(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 993 162**

(51) Int. Cl.:

G06F 9/50 (2006.01)

G06N 3/02 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(96) Fecha de presentación y número de la solicitud europea: **23.03.2018 E 18163725 (7)**

(97) Fecha y número de publicación de la concesión europea: **03.07.2024 EP 3396544**

(54) Título: **Uso compartido y compresión de datos eficientes entre sistemas de procesamiento**

(30) Prioridad:

24.04.2017 US 201715495081

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
23.12.2024

(73) Titular/es:

INTEL CORPORATION (100.00%)
2200 Mission College Blvd.
Santa Clara, CA 95054, US

(72) Inventor/es:

APPU, ABHISHEK R.;
KOKER, ALTUG;
WEAST, JOHN C.;
MACPHERSON, MIKE B.;
KIM, DUKHWAN;
HURD, LINDA L.;
BAGHSORKHI, SARA S.;
GOTTSCHLICH, JUSTIN E.;
SURTI, PRASOONKUMAR;
SAKTHIVEL, CHANDRASEKARAN y
RAY, JOYDEEP

(74) Agente/Representante:

LEHMANN NOVO, María Isabel

DESCRIPCIÓN

Uso compartido y compresión de datos eficientes entre sistemas de procesamiento

CAMPO

Las realizaciones descritas en el presente documento se relacionan en general con el procesamiento de datos y, más particularmente, con la facilitación de una herramienta para facilitar el uso compartido y la compresión de datos eficientes entre sistemas de procesamiento.

ESTADO DE LA TÉCNICA ANTERIOR

El procesamiento paralelo actual de datos gráficos incluye sistemas y métodos desarrollados para realizar operaciones específicas sobre datos gráficos, tales como, por ejemplo, interpolación lineal, teselación, rasterización, mapeo de textura, prueba de profundidad, etc. Tradicionalmente, los procesadores gráficos usan unidades computacionales de función fija para procesar datos gráficos; sin embargo, más recientemente, porciones de los procesadores gráficos se han hecho programables, lo que permite que tales procesadores admitan una gama más amplia de operaciones para procesar datos de vértice y de fragmento.

Para aumentar aún más el rendimiento, los procesadores gráficos suelen implementar técnicas de procesamiento, tales como la canalización, que intentan procesar, en paralelo, la mayor cantidad posible de datos gráficos en las diferentes partes de la canalización de gráficos. Los procesadores de gráficos paralelos con arquitecturas de múltiples hilos y única instrucción (SIMT) se diseñan para maximizar la cantidad de procesamiento paralelo en la canalización de gráficos. En una arquitectura SIMT, los grupos de hilos en paralelo intentan ejecutar juntos instrucciones de programa de manera sincrónica con la mayor frecuencia posible para aumentar la eficiencia de procesamiento. Puede encontrarse una vista global general del software y hardware para arquitecturas SIMT en Shane Cook, CUDA Programming, capítulo 3, páginas 37-51 (2013) y/o Nicholas Wilt, CUDA Handbook, A Comprehensive Guide to GPU Programming, secciones 2.6.2 a 3.1.2 (junio de 2013).

El documento US 2003/005030 A1 da a conocer la comunicación entre dispositivos autónomos que utilizan una red neuronal.

El documento US 2016/217369 A1 da a conocer la compresión de datos en redes neuronales.

El aprendizaje automático ha tenido éxito en la solución de muchos tipos de tareas. Los cálculos que surgen cuando se entrenan y se usan algoritmos de aprendizaje automático (por ejemplo, redes neuronales) se prestan naturalmente a implementaciones paralelas eficientes. En consecuencia, los procesadores paralelos, tales como las unidades de procesamiento de gráficos de propósito general (GPGPU), han desempeñado un papel importante en la implementación práctica de las redes neuronales profundas. Los procesadores de gráficos paralelos con arquitecturas de múltiples hilos y única instrucción (SIMT) se diseñan para maximizar la cantidad de procesamiento paralelo en la canalización de gráficos. En una arquitectura SIMT, los grupos de hilos en paralelo intentan ejecutar juntos instrucciones de programa de manera sincrónica con la mayor frecuencia posible para aumentar la eficiencia de procesamiento. La eficiencia proporcionada por las implementaciones paralelas de algoritmos de aprendizaje automático permite el uso de redes de alta capacidad y permite que esas redes se entrenen en conjuntos de datos más grandes.

Las técnicas convencionales no permiten compartir datos entre sistemas de procesamiento a través de una biblioteca, lo que es ineficiente y engorroso. Además, las técnicas convencionales se limitan a la compresión de datos y, por lo tanto, no anticipan la expansión o reexpansión de los modelos de compresión.

BREVE DESCRIPCIÓN DE LOS DIBUJOS

Las realizaciones se ilustran a modo de ejemplo, y no a modo de limitación, en las figuras de los dibujos adjuntos en los que números de referencia similares se refieren a elementos similares. Para que puedan entenderse en detalle las características antes citadas, se proporciona una descripción más particular, resumida anteriormente de manera breve, haciendo referencia a las realizaciones, algunas de las cuales se ilustran en los dibujos adjuntos. Sin embargo, cabe señalar que los dibujos adjuntos ilustran únicamente realizaciones típicas y, por lo tanto, no deben considerarse limitativos de su alcance, ya que los dibujos pueden ilustrar otras realizaciones igualmente efectivas.

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

Las **Figuras 2A-2D** ilustran componentes de un procesador paralelo, según una realización.

Las **Figuras 3A-3B** son diagramas de bloques de multiprocesadores de gráficos, según realizaciones.

Las **Figuras 4A-4F** ilustran una arquitectura a modo de ejemplo en la que una pluralidad de unidades de procesamiento de gráficos están acopladas de manera comunicativa a una pluralidad de procesadores de múltiples núcleos.

La **Figura 5** es un diagrama conceptual de una canalización de procesamiento de gráficos, según una realización.

La **Figura 6** ilustra un dispositivo informático que aloja un mecanismo de distribución de recursos inteligente según una realización.

La **Figura 7** ilustra un mecanismo de distribución de recursos inteligente según una realización.

La **Figura 8A** ilustra una estructura de comunicación en forma de árbol para facilitar la distribución energéticamente eficiente del aprendizaje profundo según una realización.

La **Figura 8B** ilustra una estructura de proceso para facilitar la comunicación efectiva en el aprendizaje profundo distribuido según una realización.

La **Figura 9** ilustra un método para facilitar la eficiencia en energía, comunicación y depuración en máquinas autónomas según una realización.

La **Figura 10** ilustra una pila de software de aprendizaje automático, según una realización.

La **Figura 11** ilustra una unidad de procesamiento de gráficos de propósito general altamente paralela, según una realización.

La **Figura 12** ilustra un sistema informático multi-GPU, según una realización.

Las **Figuras 13A-13B** ilustran capas de redes neuronales profundas a modo de ejemplo.

La **Figura 14** ilustra el entrenamiento y despliegue de una red neuronal profunda.

La **Figura 15** ilustra el entrenamiento y despliegue de una red neuronal profunda

La **Figura 16** es un diagrama de bloques que ilustra el aprendizaje distribuido.

La **Figura 17** ilustra un sistema de inferencia en un chip (SOC) a modo de ejemplo adecuado para realizar inferencias utilizando un modelo entrenado.

La **Figura 18** es un diagrama de bloques de una realización de un sistema informático con un procesador que tiene uno o más núcleos de procesador y procesadores de gráficos.

La **Figura 19** es un diagrama de bloques de una realización de un procesador que tiene uno o más núcleos de procesador, un controlador de memoria integrado y un procesador de gráficos integrado.

La **Figura 20** es un diagrama de bloques de una realización de un procesador de gráficos 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.

La **Figura 21** es un diagrama de bloques de una realización de un motor de procesamiento de gráficos para un procesador de gráficos.

La **Figura 22** es un diagrama de bloques de otra realización de un procesador de gráficos.

La **Figura 23** es un diagrama de bloques de la lógica de ejecución de hilos que incluye una matriz de elementos de procesamiento.

La **Figura 24** ilustra un formato de instrucción de la unidad de ejecución de procesador de gráficos según una realización.

La **Figura 25** es un diagrama de bloques de otra realización de un procesador de gráficos que incluye una canalización de gráficos, una canalización de medios, un motor de visualización, lógica de ejecución de hilos y una canalización de salida de renderizado.

La **Figura 26A** es un diagrama de bloques que ilustra un formato de comando de procesador de gráficos según una realización.

La **Figura 26B** es un diagrama de bloques que ilustra una secuencia de comandos de procesador de gráficos según una realización.

La **Figura 27** ilustra una arquitectura de software de gráficos a modo de ejemplo para un sistema de procesamiento de datos según una realización.

La **Figura 28** es un diagrama de bloques que ilustra un sistema de desarrollo de núcleo IP que puede utilizarse para fabricar un circuito integrado para realizar operaciones según una realización.

La **Figura 29** es un diagrama de bloques que ilustra un circuito integrado de sistema en chip a modo de ejemplo que puede fabricarse utilizando uno o más núcleos IP, según una realización.

La **Figura 30** es un diagrama de bloques que ilustra un procesador de gráficos ilustrativo de un sistema en circuito integrado en chip.

La **Figura 31** es un diagrama de bloques que ilustra un procesador de gráficos a modo de ejemplo adicional de un sistema en un circuito integrado en chip.

DESCRIPCIÓN DETALLADA

Las realizaciones dan a conocer una técnica novedosa para facilitar que se compartan datos entre sistemas de procesamiento utilizando una biblioteca de superficie de modo que los datos producidos en un procesador de gráficos, procesador de aplicaciones, etc., se puedan recuperar de la biblioteca de superficie y utilizar por otro si se trabaja en la misma convolución. Las realizaciones dan a conocer, además, una técnica novedosa para facilitar la expansión o reexpansión de modelos comprimidos para mejorar el rendimiento y la eficiencia de la comunicación.

Cabe señalar que, expresiones o acrónimos como "red neuronal convolucional", "CNN", "red neuronal", "NN", "red neuronal profunda", "DNN", "red neuronal recurrente", "RNN" y /o similares pueden ser referenciados de manera intercambiable a lo largo de todo este documento. Además, expresiones como "máquina autónoma" o simplemente "máquina", "vehículo autónomo" o simplemente "vehículo", "agente autónomo" o simplemente "agente", "dispositivo autónomo" o "dispositivo informático", "robot", y/o similares, se pueden hacer referencia de manera intercambiable a lo largo de todo este documento.

En algunas realizaciones, una unidad de procesamiento de gráficos (GPU) está acoplada comunicativamente a núcleos de anfitrión/procesador para acelerar las operaciones de gráficos, las operaciones de aprendizaje automático, las operaciones de análisis de patrones y varias funciones de GPU de propósito general (GPGPU). La GPU puede estar acoplada comunicativamente al procesador/núcleos de anfitrión a través de un bus u otra interconexión (por ejemplo, una interconexión de alta velocidad tal como PCIe o NVLink). En otras realizaciones, la GPU puede estar integrada en el mismo paquete o chip que los núcleos y estar acoplada comunicativamente a los núcleos a través de un bus/interconexión de procesador interno (es decir, interno al paquete o chip). Independientemente de la manera en que se conecta la GPU, los núcleos del procesador pueden asignar trabajo a la GPU en forma de secuencias de comandos/instrucciones contenidas en un descriptor de trabajo. La GPU utiliza entonces circuitos/lógica dedicados para procesar de manera eficiente estos comandos/instrucciones.

En la siguiente descripción, se exponen numerosos detalles específicos. Sin embargo, las realizaciones, tal como se describen en este documento, pueden implementarse sin estos detalles específicos. En otros casos, no se han mostrado en detalle circuitos, estructuras y técnicas bien conocidos para no complicar la comprensión de esta descripción.

Descripción general del sistema I

La **Figura 1** es un diagrama de bloques que ilustra un sistema informático 100 configurado para implementar uno o más aspectos de las realizaciones 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 del 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 un componente separado dentro de un componente de un conjunto de circuitos integrados o puede estar integrado dentro de uno o más procesadores 102. El concentrador de memoria 105 se acopla con un subsistema de E/S 111 mediante un enlace de comunicaciones 106. El subsistema de E/S 111 incluye un concentrador de E/S 107 que puede permitir que el sistema informático 100 reciba la entrada de uno o más dispositivos de entrada 108. Adicionalmente, el concentrador de E/S 107 puede permitir a un controlador de visualización, que puede estar incluido en el uno o varios procesadores 102, proporcionar salidas a uno o varios dispositivos de visualización 110A. En una 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 realización, el subsistema de procesamiento 101 incluye uno o más procesadores paralelos 112 acoplados al concentrador de memoria 105 por medio de un bus u otro enlace de comunicación 113. El enlace de comunicación 113 puede ser uno de cualquier número de tecnologías o protocolos de enlace de comunicación basados en normas, tales como, pero sin limitarse a, PCI Express, o puede ser una interfaz de comunicaciones o estructura de comunicaciones específica de un proveedor. En una realización, el uno o más procesadores paralelos 112 forman un sistema de procesamiento paralelo o vectorial computacionalmente enfocado que incluye un gran número de núcleos de procesamiento y/o agrupaciones de procesamiento, tal como un procesador de muchos núcleos integrados (MIC). En una realización, el uno o más procesadores paralelos 112 forman un subsistema de procesamiento de gráficos que puede proporcionar píxeles a uno del uno o más dispositivos de visualización 110A acoplados por medio 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 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 varios otros dispositivos que pueden agregarse a través de uno o más dispositivos de adición 120. El adaptador de red 118 puede ser un adaptador Ethernet u otro adaptador de red cableado. El adaptador de red inalámbrica 119 puede incluir uno o más de un dispositivo de red de Wi-Fi, de Bluetooth, de comunicación de campo cercano (NFC) o de otro tipo que incluye una o más radios inalámbricas.

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

En una realización, el uno o más procesadores paralelos 112 incorporan circuitos optimizados para el procesamiento de gráficos y vídeo, que incluyen, por ejemplo, circuitos de salida de vídeo, y constituyen una unidad de procesamiento de gráficos (GPU). En otra realización, el uno o más procesadores paralelos 112 incorporan circuitería optimizada para procesamiento de propósito general, mientras conservan la arquitectura computacional subyacente, descrita en mayor detalle en el presente documento. En otra realización más, los componentes del sistema informático 100 pueden integrarse con uno o más elementos del sistema distintos en un único circuito integrado. Por ejemplo, el uno o más procesadores paralelos 112, el concentrador de memoria 105, el procesador o procesadores 102 y el concentrador de E/S 107 pueden integrarse en un circuito integrado de sistema en chip (SoC). Alternativamente, los componentes del sistema informático 100 pueden integrarse en un único paquete para formar una configuración de sistema en paquete (SIP). En una realización, al menos una parte de los componentes del sistema informático 100 puede integrarse en un módulo multichip (MCM), que puede interconectarse con otros módulos multichip para formar un sistema informático modular.

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

Algunos de los componentes particulares mostrados en el presente documento son opcionales y pueden no estar incluidos en todas las implementaciones del sistema informático 100. Por ejemplo, puede admitirse cualquier número de tarjetas o periféricos complementarios, o pueden eliminarse algunos componentes. Además, algunas arquitecturas pueden utilizar una terminología diferente para componentes similares a los ilustrados en la **Figura 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 **Figura 2A** ilustra un procesador paralelo 200, de acuerdo con una realización. Los diversos componentes del procesador paralelo 200 pueden implementarse utilizando uno o más dispositivos de circuitos integrados, tales como procesadores programables, circuitos integrados específicos de la aplicación (ASIC) o matrices de puertas

programables en campo (FPGA). El procesador paralelo 200 ilustrado es una variante del uno o más procesadores paralelos 112 mostrados en la **Figura 1**, de acuerdo con una realización.

En una 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 se puede conectar directamente a otros dispositivos. En una realización, la unidad de E/S 204 se conecta con otros dispositivos mediante una interfaz de concentrador o de conmutador, tal como un 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 se conecta con una interfaz de anfitrión 206 y una barra transversal de memoria 216, donde la interfaz de anfitrión 206 recibe comandos dirigidos a realizar operaciones de procesamiento y la barra transversal de memoria 216 recibe comandos dirigidos a realizar operaciones de memoria.

Cuando la interfaz de anfitrión 206 recibe una memoria intermedia de comandos a través de la unidad de E/S 204, la interfaz de anfitrión 206 puede dirigir operaciones de trabajo para ejecutar esos comandos a un extremo frontal 208. En una realización, el extremo frontal 208 se acopla con un planificador 210, que está configurado para distribuir comandos u otros elementos de trabajo a una matriz de agrupaciones de procesamiento 212. En una realización, el planificador 210 garantiza que la matriz de agrupaciones de procesamiento 212 está configurada adecuadamente y se encuentra en un estado válido antes de que se distribuyan las tareas a las agrupaciones de procesamiento de la matriz de agrupaciones de procesamiento 212.

La matriz 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 matriz de agrupaciones de procesamiento 212 puede ejecutar un gran número de hilos concurrentes. El planificador 210 puede asignar trabajo a las agrupaciones 214A-214N de la matriz de agrupaciones de procesamiento 212 usando varios 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 ser manejada dinámicamente por el planificador 210, o puede ser asistida en parte por la lógica del compilador durante la compilación de la lógica del programa configurada para su ejecución por la matriz de agrupaciones de procesamiento 212.

En una realización, se pueden asignar diferentes agrupaciones 214A-214N de la matriz de agrupaciones de procesamiento 212 para procesar diferentes tipos de programas o para realizar diferentes tipos de cálculos.

La matriz de agrupaciones de procesamiento 212 puede configurarse para realizar varios tipos de operaciones de procesamiento en paralelo. En una realización, la matriz de agrupaciones de procesamiento 212 está configurada para realizar operaciones de cálculo paralelo de propósito general. Por ejemplo, la matriz de agrupaciones de procesamiento 212 puede incluir lógica para ejecutar tareas de procesamiento, incluyendo el filtrado de datos de vídeo y/o audio, la realización de operaciones de modelado, incluyendo operaciones de física, y la realización de transformaciones de datos.

En una realización, la matriz de agrupaciones de procesamiento 212 está configurada para realizar operaciones de procesamiento de gráficos en paralelo. En realizaciones en las que el procesador paralelo 200 está configurado para realizar operaciones de procesamiento de gráficos, la matriz de agrupaciones de procesamiento 212 puede incluir lógica adicional para admitir la ejecución de dichas operaciones de procesamiento de gráficos, incluidas, pero sin limitarse a, lógica de muestreo de textura para realizar operaciones de textura, así como lógica de teselación y otra lógica de procesamiento de vértices. Adicionalmente, la matriz 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 desde la memoria del sistema a través de la unidad de E/S 204 para su procesamiento. Durante el procesamiento, los datos transferidos se pueden almacenar en la memoria en chip (por ejemplo, la memoria del procesador paralelo 222) durante el procesamiento, y luego escribirse nuevamente en la memoria del sistema.

En una realización, cuando la unidad de procesamiento paralelo 202 se utiliza para realizar el procesamiento de gráficos, el planificador 210 puede configurarse para dividir la carga de trabajo de procesamiento en tareas de tamaño aproximadamente igual, para permitir mejor la distribución de las operaciones de procesamiento de gráficos a múltiples agrupaciones 214A-214N de la matriz de agrupaciones de procesamiento 212. En algunas realizaciones, porciones de la matriz de agrupaciones de procesamiento 212 pueden configurarse para realizar diferentes tipos de procesamiento. Por ejemplo, una primera porción puede configurarse para realizar un sombreado de vértices y una generación de topología, una segunda porción puede configurarse para realizar sombreado de teselación y de geometría, y una tercera porción puede configurarse para realizar sombreado de píxeles u otras operaciones de espacio de pantalla, para producir una imagen representada para su visualización. Datos intermedios producidos por una o más de las agrupaciones 214A-214N pueden almacenarse en memorias intermedias para permitir que los datos intermedios se transmitan entre las agrupaciones 214A-214N para su procesamiento adicional.

Durante el funcionamiento, la matriz 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 el extremo frontal 208. Para las operaciones de procesamiento de gráficos, las tareas de procesamiento pueden incluir índices de datos que se procesarán, 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 procesarán los datos (por ejemplo, qué programa se ejecutará). El planificador 210 puede configurarse para extraer los índices que corresponden a las tareas o puede recibir los índices desde el extremo frontal 208. El extremo frontal 208 se puede configurar para garantizar que la matriz de agrupaciones de procesamiento 212 esté configurada en un estado válido antes de que se inicie la carga de trabajo especificada por las memorias intermedias de comandos entrantes (por ejemplo, memorias intermedias de lotes, memorias intermedias de inserción, etc.).

Cada una de la una o más instancias de la unidad de procesamiento paralelo 202 se puede acoplar con la memoria de procesador paralelo 222. Se puede acceder a la memoria de procesador paralelo 222 a través de la barra transversal de memoria 216, que puede recibir solicitudes de memoria de la matriz de agrupaciones de procesamiento 212 así como de la unidad de E/S 204. La barra transversal 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, a través de la unidad de subdivisión 220N) que pueden acoplarse cada una a una parte (por ejemplo, unidad de memoria) de la memoria de procesador paralelo 222. En una implementación, el número de unidades de subdivisión 220A-220N está configurado para que sea igual al número de unidades de memoria, de manera que una primera unidad de subdivisión 220A tiene una primera unidad de memoria 224A correspondiente, una segunda unidad de subdivisión 220B tiene una unidad de memoria 224B correspondiente y una N-ésima unidad de subdivisión 220N tiene una N-ésima unidad de memoria 224N correspondiente. En otras realizaciones, el número de unidades de subdivisión 220A-220N puede no ser igual al número de dispositivos de memoria.

En diversas realizaciones, las unidades de memoria 224A-224N pueden incluir diversos tipos de dispositivos de memoria, que incluyen memoria de acceso aleatorio dinámica (DRAM) o memoria gráfica de acceso aleatorio, tal como una memoria gráfica de acceso aleatorio sincrónica (SGRAM), que incluye una memoria gráfica de doble tasa de datos (GDDR). En una realización, las unidades de memoria 224A-224N también pueden incluir memoria apilada 3D, que incluye, entre otras, memoria de ancho de banda alto (HBM). Los expertos en la materia apreciarán que la implementación específica de las unidades de memoria 224A-224N puede variar y puede seleccionarse a partir de uno de diversos diseños convencionales. Los objetivos de renderizado, tales como las memorias intermedias de fotogramas o los mapas de textura, se pueden almacenar en las unidades de memoria 224A-224N, lo que permite que las unidades de subdivisión 220A-220N escriban partes de cada objetivo de renderizado en paralelo para utilizar de manera eficiente el ancho de banda disponible de la memoria de procesador paralelo 222. En algunas realizaciones, puede excluirse una instancia local de la memoria de procesador paralelo 222 en favor de un diseño de memoria unificado que utiliza memoria de sistema junto con memoria caché local.

En una realización, cualquiera de las agrupaciones 214A-214N de la matriz 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 transversal de memoria 216 se puede configurar para transferir la salida de cada agrupación 214A-214N a cualquier unidad de subdivisión 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 transversal de memoria 216 para leer desde, o escribir en varios dispositivos de memoria externos. En una realización, la barra transversal 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 realización, la barra transversal de memoria 216 puede usar canales virtuales para separar flujos de tráfico entre las agrupaciones 214A-214N y las unidades de subdivisión 220A-220N.

Si bien se ilustra una única instancia de la unidad de procesamiento paralelo 202 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 adicional o se pueden interconectar múltiples tarjetas adicionales. Las diferentes instancias de la unidad de procesamiento paralelo 202 pueden configurarse para interoperar incluso si las diferentes instancias tienen diferentes números de núcleos de procesamiento, diferentes cantidades de memoria de procesador paralelo local y/u otras diferencias de configuración. Por ejemplo, y en una realización, algunas instancias de la unidad de procesamiento paralelo 202 pueden incluir unidades de coma flotante de mayor precisión con relación a otras instancias. Los sistemas que incorporan una o más instancias de la unidad de procesamiento paralelo 202 o el procesador paralelo 200 pueden implementarse en una diversidad de configuraciones y factores de forma, incluyendo, pero sin limitarse a, ordenadores personales de escritorio, portátiles o de mano, servidores, estaciones de trabajo, consolas de juegos y/o sistemas integrados.

La **Figura 2B** es un diagrama de bloques de una unidad de subdivisión 220, de acuerdo con una realización. En una realización, la unidad de subdivisión 220 es una instancia de una de las unidades de subdivisión 220A-220N de la

Figura 2A. Como se ilustra, la unidad de subdivisión 220 incluye una memoria caché L2 221, una interfaz de memoria intermedia 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 de almacenamiento recibidas desde la barra transversal de memoria 216 y la ROP 226. Los desaciertos de lectura y las solicitudes de escritura diferida urgente son emitidas por la memoria caché L2 221 a la interfaz de memoria intermedia de fotogramas 225 para su procesamiento. Las actualizaciones sucias también se pueden enviar a la memoria intermedia de fotogramas a través de la interfaz de memoria intermedia de fotogramas 225 para un procesamiento oportunista. En una realización, la interfaz de memoria intermedia de fotogramas 225 interactúa con una de las unidades de memoria en la memoria del procesador paralelo, tales como las unidades de memoria 224A-224N de la **Figura 2A** (por ejemplo, dentro de la memoria del procesador paralelo 222).

En las aplicaciones de gráficos, la ROP 226 es una unidad de procesamiento que realiza operaciones de rasterización, tales como estarcido, prueba z, mezcla y similares. La ROP 226 emite, a continuación, datos de gráficos procesados que se almacenan en memoria de gráficos. En algunas realizaciones, la ROP 226 incluye lógica de compresión para comprimir z o datos de color que se escriben en la memoria y descomprimir z o datos de color que se leen de la memoria. En algunas realizaciones, la ROP 226 está incluida dentro de cada agrupación de procesamiento (por ejemplo, la agrupación 214A-214N de la **Figura 2A**) en lugar de dentro de la unidad de subdivisión 220. En dicha realización, las solicitudes de lectura y escritura de datos de píxeles se transmiten a través de la barra transversal de memoria 216 en lugar de datos de fragmentos de píxeles.

Los datos de gráficos procesados pueden visualizarse en un dispositivo de visualización, tal como uno del uno o más dispositivos de visualización 110 de la **Figura 1**, enrutarse para su procesamiento adicional por el procesador o procesadores 102, o enrutarse para su procesamiento adicional por una de las entidades de procesamiento dentro del procesador paralelo 200 de la **Figura 2A**.

La **Figura 2C** es un diagrama de bloques de una agrupación de procesamiento 214 dentro de una unidad de procesamiento paralelo, de acuerdo con una realización. En una realización, la agrupación de procesamiento es una instancia de una de las agrupaciones de procesamiento 214A-214N de la **Figura 2A**. La agrupación de procesamiento 214 puede configurarse para ejecutar muchos hilos en paralelo, donde el término "hilo" se refiere a una instancia de un programa particular que se ejecuta en un conjunto particular de datos de entrada. En algunas realizaciones, se utilizan técnicas de emisión de instrucciones de instrucción única, múltiples datos (SIMD) para admitir la ejecución paralela de una gran cantidad de hilos sin proporcionar múltiples unidades de instrucción independientes. En otras realizaciones, se usan técnicas de única instrucción, múltiples hilos (SIMT) para admitir la ejecución paralela de un gran número de hilos generalmente sincronizados, usando una unidad de instrucciones común configurada para emitir instrucciones en un conjunto de motores de procesamiento dentro de cada una de las agrupaciones de procesamiento. A diferencia del régimen de ejecución de SIMD, donde todos los motores de procesamiento ejecutan habitualmente instrucciones idénticas, la ejecución de SIMT permite que diferentes hilos sigan más fácilmente rutas de ejecución divergentes a través de un programa de hilos dado. Los expertos en la materia comprenderán que un régimen de procesamiento SIMD representa un subconjunto funcional de un régimen de procesamiento SIMT.

El funcionamiento de la agrupación de procesamiento 214 puede controlarse a través de un gestor de canalización 232 que distribuye tareas de procesamiento a procesadores paralelos SIMT. El gestor de canalización 232 recibe instrucciones del planificador 210 de la **Figura 2A** y gestiona la ejecución de esas instrucciones a través de un multiprocesador de gráficos 234 y/o una unidad de texturas 236. El multiprocesador de gráficos 234 ilustrado es una instancia a modo de ejemplo de un procesador paralelo de SIMT. Sin embargo, se pueden incluir varios tipos de procesadores paralelos de SIMT de diferentes arquitecturas dentro de la agrupación de procesamiento 214. Una o más instancias del multiprocesador de gráficos 234 pueden incluirse dentro de una agrupación de procesamiento 214. El multiprocesador de gráficos 234 puede procesar datos y una barra transversal de datos 240 se puede utilizar para distribuir los datos procesados a uno de los múltiples destinos posibles, incluyendo otras unidades de sombreado. El gestor de canalización 232 puede facilitar la distribución de los datos procesados mediante la especificación de destinos para los datos procesados que se distribuirán a través de la barra transversal 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 lógicas aritméticas, unidades de carga y almacenamiento, etc.). La lógica de ejecución funcional puede configurarse de una manera canalizada en la que pueden emitirse nuevas instrucciones antes de que se hayan completado instrucciones previas. Se puede proporcionar lógica de ejecución funcional. La lógica funcional admite una diversidad de operaciones que incluyen aritmética de números enteros y de coma flotante, operaciones de comparación, operaciones booleanas, desplazamiento de bits y de cálculo de diversas funciones algebraicas. En una realización, puede aprovecharse el mismo hardware de unidades funcionales para realizar diferentes operaciones, y puede estar presente cualquier combinación de unidades funcionales.

Las instrucciones transmitidas a la agrupación de procesamiento 214 constituyen un hilo. Un conjunto de hilos que se ejecutan en el conjunto de motores de procesamiento paralelo es un grupo de hilos. Un grupo de hilos ejecuta el mismo programa en diferentes datos de entrada. Cada hilo dentro de un grupo de hilos puede asignarse a un motor de procesamiento diferente dentro de un multiprocesador de gráficos 234. Un grupo de hilos puede incluir menos hilos que el número de motores de procesamiento dentro del multiprocesador de gráficos 234. Cuando un grupo de hilos

incluye menos hilos 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 está procesando ese grupo de hilos. Un grupo de hilos también puede incluir más hilos que el número de motores de procesamiento dentro del multiprocesador de gráficos 234. Cuando el grupo de hilos incluye más hilos que el número de motores de procesamiento dentro del multiprocesador de gráficos 234, el procesamiento se puede realizar a lo largo de ciclos de reloj consecutivos. En una realización, múltiples grupos de hilos pueden ejecutarse concurrentemente en un multiprocesador de gráficos 234.

En una realización, el multiprocesador de gráficos 234 incluye una memoria caché interna para realizar operaciones de carga y de almacenamiento. En una 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 de la agrupación de procesamiento 214. Cada multiprocesador de gráficos 234 también tiene acceso a las memorias caché L2 dentro de las unidades de subdivisión (por ejemplo, las unidades de subdivisión 220A-220N de la **Figura 2A**) que se comparten entre todas las agrupaciones de procesamiento 214 y pueden usarse para transferir datos entre hilos. El multiprocesador de gráficos 234 puede acceder también a memoria global fuera de chip, que puede incluir una o más de memoria de procesador paralelo local y/o memoria de sistema. Cualquier memoria externa a la unidad de procesamiento paralelo 202 se puede utilizar como memoria global. Las realizaciones 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 mapear direcciones virtuales a direcciones físicas. En otras realizaciones, una o más instancias de la MMU 245 pueden residir dentro de la interfaz de memoria 218 de la **Figura 2A**. La MMU 245 incluye un conjunto de entradas de tabla de página (PTE) utilizadas para mapear una dirección virtual a una dirección física de un mosaico (se hablará más sobre la) y, opcionalmente, un índice de línea de memoria caché. La MMU 245 puede incluir memorias intermedias de conversión anticipada de direcciones (TLB) o memorias caché que pueden residir dentro del multiprocesador de gráficos 234 o de la memoria caché L1 o la agrupación de procesamiento 214. La dirección física se procesa para distribuir la localidad de acceso a los datos de superficie para permitir un intercalado eficiente de solicitudes entre unidades de subdivisión. El índice de línea de memoria caché se puede usar para determinar si una solicitud para una línea de memoria caché es un acierto o un fallo.

En aplicaciones gráficas y de computación, 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 mapeo de textura, por ejemplo, determinar posiciones de muestra de textura, leer datos de textura y filtrar los datos de textura. Los datos de textura se leen desde una memoria caché L1 de textura interna (no mostrada) o en algunas realizaciones desde 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 del sistema, según sea necesario. Cada multiprocesador de gráficos 234 proporciona tareas procesadas a la barra transversal 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 transversal de memoria 216. Una unidad preROP 242 (unidad de operaciones de prerrasterización) está configurada para recibir datos del multiprocesador de gráficos 234, dirigir los datos a las unidades ROP, que pueden estar ubicadas con unidades de subdivisión como se describe en este documento (por ejemplo, las unidades de subdivisión 220A-220N de la Figura 2A). La unidad preROP 242 puede realizar optimizaciones para la combinación de colores, organizar datos de color de píxeles y realizar traducciones de direcciones.

Se apreciará que la arquitectura de núcleo descrita en el presente documento es ilustrativa y que son posibles variaciones y modificaciones. Cualquier número de unidades de procesamiento, por ejemplo, multiprocesador de gráficos 234, unidades de textura 236, preROP 242, etc., pueden incluirse dentro de una agrupación de procesamiento 214. Además, aunque solo se muestra una agrupación de procesamiento 214, una unidad de procesamiento paralela como se describe en este documento puede incluir cualquier número de instancias de la agrupación de procesamiento 214. En una realización, cada agrupación de procesamiento 214 puede ser configurada para operar independientemente de otras agrupaciones de procesamiento 214 usando unidades de procesamiento separadas y distintas, memorias caché L1, etc.

La **Figura 2D** muestra un multiprocesador de gráficos 234, de acuerdo con una realización. En tal realización, el multiprocesador de gráficos 234 se acopla con el gestor de canalización 232 de la agrupación de procesamiento 214. El multiprocesador de gráficos 234 tiene una canalización de ejecución que incluye, pero sin limitación, una memoria caché de instrucciones 252, una unidad de instrucciones 254, una unidad de mapeo 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 acoplados con la memoria caché 272 y la memoria compartida 270 a través de una interconexión de memoria y memoria caché 268.

En una realización, la memoria caché de instrucciones 252 recibe un flujo de instrucciones para ejecutarse desde el gestor de canalización 232. 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 hilos (por ejemplo, urdimbres), con cada hilo del grupo de hilos asignado a una unidad de ejecución diferente dentro del núcleo de GPGPU 262. Una instrucción puede acceder a cualquiera de un espacio de direcciones local, compartido o global especificando una dirección dentro de un espacio de direcciones unificado. La unidad de mapeo de direcciones 256 puede usarse para traducir direcciones en el espacio de direcciones unificado a una dirección de memoria distinta a la que pueden acceder 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 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 realización, el archivo de registros 258 se divide entre cada una de las unidades funcionales de tal manera que a cada unidad funcional se le asigna una porción dedicada del archivo de registros 258. En una realización, el archivo de registros 258 se divide entre las diferentes urdimbres que ejecuta el multiprocesador de gráficos 324.

Los núcleos de GPGPU 262 pueden incluir cada uno unidades de coma flotante (FPU) y/o unidades lógicas aritméticas de números enteros (ALU) 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, según las realizaciones. Por ejemplo, y en una 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 precisión doble. En una realización, las FPU pueden implementar la norma IEEE 754-2008 para la aritmética de coma flotante o permitir una aritmética de coma flotante de precisión variable. El multiprocesador de gráficos 324 puede incluir adicionalmente una o más funciones fijas o unidades de funciones especiales para realizar funciones específicas, tales como copiar rectángulos u operaciones de combinación de píxeles. En una realización, uno o varios de los núcleos de GPGPU puede incluir también lógica de función fija o especial.

La interconexión de memoria y memoria caché 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 realización, la interconexión de memoria y memoria caché 268 es una interconexión de barra transversal 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 operar a la misma frecuencia que los núcleos de GPGPU 262, por lo tanto, la transferencia de datos entre los núcleos de GPGPU 262 y el archivo de registros 258 es de muy baja latencia. La memoria compartida 270 se puede usar para permitir la comunicación entre hilos 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 memoria caché datos de textura comunicados entre las unidades funcionales y la unidad de textura 236. La memoria compartida 270 también se puede utilizar como memoria caché gestionada por programas. Los hilos que se ejecutan en los núcleos de GPGPU 262 pueden almacenar datos mediante programación dentro de la memoria compartida además de los datos almacenados en memoria caché automáticamente que se almacenan dentro de la memoria caché 272.

Las **Figuras 3A-3B** ilustran multiprocesadores de gráficos adicionales, de acuerdo con realizaciones. Los multiprocesadores de gráficos 325, 350 ilustrados son variantes del multiprocesador de gráficos 234 de la **Figura 2C**. Los multiprocesadores de gráficos 325, 350 ilustrados se pueden configurar como un multiprocesador de transmisión (SM) capaz de ejecutar simultáneamente una gran cantidad de hilos de ejecución.

La **Figura 3A** muestra un multiprocesador de gráficos 325 de acuerdo con una 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 **Figura 2D**. Por ejemplo, el multiprocesador de gráficos 325 puede incluir múltiples instancias de la unidad de instrucciones 332A-332B, el archivo de registros 334A-334B y la(s) unidad(es) de textura 344A-344B. El multiprocesador de gráficos 325 también incluye múltiples conjuntos de gráficos o unidades de ejecución de cálculo (p. ej., 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 realización, las unidades de recursos de ejecución tienen una memoria caché de instrucciones común 330, una memoria caché de textura y/o datos 342 y una memoria compartida 346. Los diversos componentes pueden comunicarse por medio de una estructura de interconexión 327. En una realización, la estructura de interconexión 327 incluye uno o más conmutadores de barra transversal para permitir la comunicación entre los diversos componentes del multiprocesador de gráficos 325.

La **Figura 3B** muestra un multiprocesador de gráficos 350 de acuerdo con una 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 instrucciones, archivos de registros, núcleos de GPGPU y unidades de carga-almacenamiento, como se ilustra en la **Figura 2D** y en la **Figura 3A**. Los recursos de ejecución 356A-356D pueden funcionar en conjunto con la unidad o unidades de texturas 360A-360D para operaciones de textura, mientras que comparten una memoria caché de instrucciones 354 y la memoria compartida 362. En una 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 textura y/o datos 358A-358B. Los diversos componentes

pueden comunicarse mediante una estructura de interconexión 352 similar a la estructura de interconexión 327 de la **Figura 3A**.

Los expertos en la materia comprenderán que la arquitectura descrita en las **Figuras 1**, 2A-2D y 3A-3B es descriptiva y no limitativa en cuanto al alcance de las presentes realizaciones. Por lo tanto, las técnicas descritas en el presente documento pueden implementarse en cualquier unidad de procesamiento configurada adecuadamente, incluyendo, sin limitación, uno o más procesadores de aplicaciones móviles, una o más unidades de procesamiento central (CPU) de escritorio o servidor, incluyendo CPU de múltiples núcleos, una o más unidades de procesamiento paralelo, tales como la unidad de procesamiento paralelo 202 de la **Figura 2A**, así como uno o más procesadores de gráficos o unidades de procesamiento de propósito especial, sin apartarse del alcance de las realizaciones descritas en el presente documento.

En algunas realizaciones, un procesador paralelo o GPGPU descrito en el presente documento está acoplado de manera comunicativa a núcleos de anfitrión/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 acoplarse de manera comunicativa al procesador/núcleos de anfitrión a través de un bus u otra interconexión (por ejemplo, una interconexión de alta velocidad tal como PCIe o NVLink). En otras realizaciones, la GPU puede estar integrada en el mismo paquete o chip que los núcleos y estar acoplada comunicativamente a los núcleos a través de un bus/interconexión de procesador interno (es decir, interno al paquete o chip). Independientemente de la manera en que se conecta la GPU, los núcleos del procesador pueden asignar trabajo a la GPU en forma de secuencias de comandos/instrucciones contenidas en un descriptor de trabajo. La GPU utiliza entonces circuitos/lógica dedicados para procesar de manera eficiente estos comandos/instrucciones.

Técnicas de interconexión de GPU y procesador anfitrión

La **Figura 4A** ilustra una arquitectura a modo de ejemplo 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 realización, los enlaces de alta velocidad 440-443 admiten un caudal de comunicación de 4 GB/s, 30 GB/s, 80 GB/s o superior, dependiendo de la implementación. Se pueden utilizar varios protocolos de interconexión, incluidos, entre otros, PCIe 4.0 o 5.0 y NVLink 2.0. Sin embargo, los principios subyacentes de la memoria descriptiva no se limitan a ningún protocolo de comunicación o rendimiento en particular.

Además, en una realización, dos o más de las GPU 410-413 están interconectadas a través de enlaces de alta velocidad 444-445, que se pueden implementar 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 estar conectados a través de un enlace de alta velocidad 433 que puede ser un bus multiprocesador simétrico (SMP) que funciona a 20 GB/s, 30 GB/s, 120 GB/s o más. Como alternativa, toda la comunicación entre los diversos componentes de sistema que se muestran en la **Figura 4A** se puede lograr usando los mismos protocolos/enlaces (por ejemplo, a través de una estructura de interconexión común). Sin embargo, como se mencionó, los principios subyacentes de la memoria descriptiva no están limitados a ningún tipo particular de tecnología de interconexión.

En una realización, cada procesador de múltiples núcleos 405-406 está acoplado comunicativamente a una memoria de procesador 401-402, a través de interconexiones de memoria 430-431, respectivamente, y cada GPU 410-413 está acoplada comunicativamente a la memoria GPU 420-423 a través de interconexiones de memoria GPU 450-453, respectivamente. Las interconexiones de memoria 430-431 y 450-453 pueden utilizar las mismas o diferentes tecnologías de acceso a la 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 dinámico (DRAM) (incluyendo DRAM apiladas), SDRAM DDR gráfica (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 realización, una parte de las memorias puede ser memoria volátil y otra parte puede ser memoria no volátil (por ejemplo, utilizando una jerarquía de memoria de dos niveles (2LM)).

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

La **Figura 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 realización. El módulo de aceleración de gráficos 446 puede incluir uno o más chips de GPU integrados en una tarjeta de línea que se acopla al procesador 407 mediante

el enlace de alta velocidad 440. Alternativamente, el módulo de aceleración de gráficos 446 puede estar integrado en el mismo paquete o chip que el procesador 407.

El procesador 407 ilustrado incluye una pluralidad de núcleos 460A-460D, cada uno con una memoria intermedia de conversión anticipada 461A-461D y una o varias memorias caché 462A-462D. Los núcleos pueden incluir varios otros componentes para ejecutar instrucciones y procesar datos que no se ilustran para evitar oscurecer los principios subyacentes de la memoria descriptiva (por ejemplo, unidades de búsqueda de instrucciones, unidades de predicción de bifurcación, decodificadores, unidades de ejecución, memorias intermedias de reordenación, etc.). Las memorias caché 462A-462D pueden comprender memorias caché de nivel 1 (L1) y nivel 2 (L2). Además, una o varias memorias caché compartidas 426 pueden incluirse en la jerarquía de almacenamiento en memoria caché y ser compartidas por conjuntos de núcleos 460A-460D. Por ejemplo, una 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 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 se conectan con la memoria de sistema 441, que puede incluir las memorias de procesador 401-402.

La coherencia se mantiene para los datos e instrucciones almacenados en las distintas memorias caché 462A-462D, 456 y la memoria del sistema 441 a través de una comunicación entre núcleos a través de un bus de coherencia 464. Por ejemplo, cada memoria caché puede tener una lógica/circuitería de coherencia de memoria caché asociada con la misma con la que comunicarse a través del bus de coherencia 464 en respuesta a lecturas o escrituras detectadas en líneas de caché particulares. En una implementación, se implementa un protocolo de monitorización de caché a través del bus de coherencia 464 para monitorizar los accesos de memoria caché. Las técnicas de monitorización/coherencia de la memoria caché son bien entendidas por los expertos en la materia y no se describirán en detalle aquí para evitar oscurecer los principios subyacentes de la memoria descriptiva.

En una realización, un circuito proxy 425 acopla comunicativamente 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 la memoria caché como un par de los núcleos. En particular, una interfaz 435 proporciona conectividad al circuito proxy 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 440.

En una implementación, un circuito de integración de acelerador 436 proporciona servicios de gestión de caché, acceso a memoria, gestión de contexto y gestión de interrupción en nombre de una pluralidad de motores de procesamiento de gráficos 431, 432, N del módulo de aceleración de gráficos 446. Los motores de procesamiento de gráficos 431, 432, N pueden comprender cada uno una unidad de procesamiento de gráficos (GPU) separada. Como 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/decodificadores 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 paquete, una tarjeta de línea o un chip común.

En una realización, el circuito de integración del acelerador 436 incluye una unidad de gestión de memoria (MMU) 439 para realizar varias funciones de gestión de memoria, 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 del sistema 441. La MMU 439 también puede incluir una memoria intermedia de conversión anticipada de direcciones (TLB) (no mostrada) para almacenar en memoria caché las conversiones de dirección virtual/efectiva a física/real. En una implementación, una memoria caché 438 almacena comandos y datos para un acceso eficaz por los motores de procesamiento de gráficos 431-432, N. En una realización, los datos almacenados en la memoria caché 438 y en las memorias de gráficos 433-434, N se mantienen coherentes con las memorias caché de núcleo 462A-462D, 456 y la memoria de sistema 411. Como se mencionó, esto se puede lograr a través del circuito proxy 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, enviando actualizaciones a la memoria caché 438 relacionadas con modificaciones/accesos de líneas de caché en las memorias caché de procesador 462A-462D, 456 y recibiendo actualizaciones de la memoria caché 438).

Un conjunto de registros 445 almacenan datos de contexto para hilos ejecutados por los motores de procesamiento de gráficos 431-432, N y un circuito de gestión de contexto 448 gestiona los contextos de hilo. Por ejemplo, el circuito de gestión de contexto 448 puede realizar operaciones de guardar y restaurar para guardar y restaurar contextos de los diversos hilos durante conmutaciones de contextos (por ejemplo, donde un primer hilo se guarda y un segundo hilo se almacena para que el segundo hilo pueda ser ejecutado por un motor de procesamiento de gráficos). Por ejemplo, en una conmutación de contexto, el circuito de gestión de contexto 448 puede almacenar valores de registro actuales en un área designada en memoria (por ejemplo, identificada por un puntero de contexto). A continuación, puede restaurar los valores de registro cuando vuelve al contexto. En una realización, un circuito de gestión de interrupciones 447 recibe y procesa interrupciones recibidas de dispositivos del sistema.

En una implementación, las direcciones virtuales/efectivas de un motor de procesamiento de gráficos 431 se traducen a direcciones reales/físicas en la memoria del sistema 411 por la MMU 439. Una realización del circuito de integración de acelerador 436 admite múltiples (por ejemplo, 4, 8, 16) módulos de aceleración de gráficos 446 y/u otros dispositivos de aceleración. El módulo de aceleración de gráficos 446 puede estar dedicado a una única aplicación ejecutada en el procesador 407 o puede ser compartido entre múltiples aplicaciones. En una 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 pueden subdividirse en "secciones" que se asignan a diferentes máquinas virtuales (VM) y/o aplicaciones en función de los requisitos de procesamiento y las prioridades asociadas con las VM y/o aplicaciones.

Por 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 caché de sistema. Además, el circuito de integración de acelerador 436 puede proporcionar funciones de virtualización para que el procesador anfitrión gestione la virtualización de los motores de procesamiento de gráficos, las interrupciones y la gestión de memoria.

Como los recursos de hardware de los motores de procesamiento de gráficos 431-432, N se mapean explícitamente al espacio de direcciones real visto por el procesador anfitrión 407, cualquier procesador anfitrión puede direccionar estos recursos directamente utilizando un valor de dirección efectiva. Una función del circuito de integración de acelerador 436, en una realización, es la separación física de los motores de procesamiento de gráficos 431-432, N para que aparezcan ante el sistema como unidades independientes.

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

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

La **Figura 4C** ilustra otra realización en la que el circuito de integración del acelerador 436 está integrado dentro del procesador 407. En esta 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 del acelerador 436 a través de la interfaz 437 y la interfaz 435 (que, nuevamente, pueden utilizar cualquier forma de bus o protocolo de interfaz). El circuito de integración del acelerador 436 puede realizar las mismas operaciones que las descritas con respecto a la **Figura 4B**, pero potencialmente con un mayor caudal dada su proximidad al bus de coherencia 462 y a las memorias caché 462A-462D, 426.

Una realización admite diferentes modelos de programación, incluido un modelo de programación de proceso dedicado (sin virtualización del módulo de aceleración de gráficos) y modelos de programación compartidos (con virtualización). Esta última puede incluir modelos de programación que están controlados por el circuito de integración del acelerador 436 y modelos de programación que están controlados por el módulo de aceleración de gráficos 446.

En una 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 única aplicación puede canalizar otras solicitudes de aplicación a los motores de gráficos 431-432, N, proporcionando virtualización dentro de una VM/subdivisión.

En los modelos de programación de proceso dedicado, los motores de procesamiento de gráficos 431-432, N, pueden ser 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 de cada sistema operativo. En sistemas de subdivisión única sin un 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.

Para el modelo de programación compartido, el módulo de aceleración de gráficos 446 o un motor de procesamiento de gráficos individual 431-432, N selecciona un elemento de proceso utilizando un identificador de proceso. En una realización, los elementos de proceso se almacenan en la memoria de sistema 411 y se pueden direccionar utilizando las técnicas de conversión de dirección efectiva a dirección real descritas en el presente documento. El identificador de proceso puede ser un valor específico de la implementación proporcionado al proceso anfitrión cuando registra su contexto con el motor de procesamiento de gráficos 431-432, N (es decir, llamando al software del sistema para

agregar el elemento de proceso a la lista enlazada de elementos de proceso). Los 16 bits inferiores del identificador de proceso pueden ser el desplazamiento del elemento de proceso dentro de la lista enlazada de elementos de proceso.

La **Figura 4D** ilustra una sección de integración de acelerador 490 a modo de ejemplo. Como se usa en el presente documento, una "sección" comprende una parte 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 los elementos de proceso 483. En una realización, los elementos de proceso 483 se almacenan en respuesta a invocaciones de GPU 481 desde las aplicaciones 480 ejecutadas en el procesador 407. Un elemento de proceso 483 contiene el estado del proceso para la aplicación correspondiente 480. Un descriptor de trabajo (WD) 484 contenido en el elemento de proceso 483 puede ser un único trabajo solicitado por una aplicación o puede contener un puntero a una cola de trabajos. En el último caso, el WD 484 es un puntero a la cola de solicitud de trabajo en el espacio de direcciones de la aplicación 482.

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 en el sistema. Las realizaciones incluyen una infraestructura para configurar el estado del proceso y enviar un WD 484 a un módulo de aceleración de gráficos 446 para iniciar un trabajo en un entorno virtualizado.

En una implementación, el modelo de programación de proceso dedicado es específico de la implementación. En este modelo, un único proceso posee el módulo de aceleración de gráficos 446 o un motor de procesamiento de gráficos 431 individual. Como el módulo de aceleración de gráficos 446 es propiedad de un solo proceso, el hipervisor inicializa el circuito de integración del acelerador 436 para la subdivisión propietaria y el sistema operativo inicializa el circuito de integración del acelerador 436 para el proceso propietario en el momento en que se asigna el módulo de aceleración de gráficos 446.

En funcionamiento, una unidad de extracción de WD 491 en la sección de integración del acelerador 490 extrae 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 se pueden almacenar en registros 445 y ser usados 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 realización de la MMU 439 incluye circuitos de recorrido de páginas/segmentos para acceder a tablas de segmentos/páginas 486 dentro del espacio de direcciones virtuales 485 del sistema operativo. El circuito de gestión de interrupciones 447 puede procesar eventos de interrupción 492 recibidos desde el módulo de aceleración de gráficos 446. Al realizar operaciones de gráficos, una dirección efectiva 493 generada por un motor de procesamiento de gráficos 431-432, N se convierte a una dirección real mediante la MMU 439.

En una 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 inicializarse por el hipervisor o el sistema operativo. Cada uno de estos registros duplicados se puede incluir en una sección de integración de acelerador 490. Se muestran los registros a modo de ejemplo que pueden inicializarse por el hipervisor en la **Tabla 1.**

Tabla 1 - Registros inicializados del hipervisor

1	Registro de control de sección
2	Puntero a área de procesos planificados de dirección real (RA)
3	Registro de anulación de máscara de autoridad
4	Desplazamiento de entrada de la tabla de vectores de interrupción
5	Límite de entrada de la tabla de vectores de interrupción
6	Registro de estado
7	ID de subdivisión lógica
8	Puntero de registro de utilización del acelerador del hipervisor de dirección real (RA)
9	Registro de descripción de almacenamiento

En la **Tabla 2** se muestran registros a modo de ejemplo que pueden ser inicializados por el sistema operativo.

Tabla 2 - Registros inicializados por el sistema operativo

1	Identificación de proceso y de hilo
2	Puntero guardado/restauración de contexto de dirección efectiva (EA)
3	Puntero de registro de utilización de acelerador de dirección virtual (VA)

4	Puntero de tabla de segmentos de almacenamiento de dirección virtual (VA)
5	Máscara de autoridad
6	Descriptor de trabajo

En una realización, cada WD 484 es específico para un módulo de aceleración de gráficos en particular 446 y/o motores 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 **Figura 4E** ilustra detalles adicionales para una realización de un modelo compartido. Esta 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 mediante un hipervisor 496 que virtualiza los motores de módulo de aceleración de gráficos para el sistema operativo 495.

Los modelos de programación compartidos permiten que todos o un subconjunto de procesos de todas o un subconjunto de subdivisiones del sistema utilicen un módulo de aceleración de gráficos 446. Existen dos modelos de programación en los que el módulo de aceleración de gráficos 446 es compartido por múltiples procesos y subdivisiones: compartido por secciones de tiempo y compartido dirigido por gráficos.

En este modelo, el hipervisor de sistema 496 posee el módulo de aceleración de gráficos 446 y pone su función a disposición de todos los sistemas operativos 495. Para que un módulo de aceleración de gráficos 446 admita la virtualización por parte del hipervisor del sistema 496, el módulo de aceleración de gráficos 446 puede cumplir los siguientes requisitos: 1) La solicitud de trabajo de una aplicación debe ser autónoma (es decir, no es necesario mantener el estado entre trabajos), o el módulo de aceleración de gráficos 446 debe proporcionar un mecanismo de guardado y recuperación de contexto. 2) El módulo de aceleración de gráficos 446 garantiza que la solicitud de trabajo de una aplicación se complete en una cantidad de tiempo especificada, incluidos los errores de conversión, o el módulo de aceleración de gráficos 446 proporciona la capacidad de dar prioridad al procesamiento del trabajo. 3) El módulo de aceleración de gráficos 446 debe garantizar equidad entre procesos cuando opera en el modelo de programación compartido dirigido.

En una realización, para el modelo compartido, se requiere que la aplicación 480 realice una llamada de sistema del 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 de área de guardado/restauración de contexto (CSRP). El tipo del módulo de aceleración de gráficos 446 describe la función de aceleración dirigida como objetivo para la llamada de sistema. El tipo del módulo de aceleración de gráficos 446 puede ser un valor específico del sistema. El WD está formateado específicamente para el módulo de aceleración de gráficos 446 y puede tener la forma de un comando del módulo de aceleración de gráficos 446, un puntero de dirección efectiva a una estructura definida por el usuario, un 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 realización, el valor de AMR es el estado de AMR que se debe usar para el proceso actual. El valor pasado al sistema operativo es similar a que una aplicación establezca el AMR. Si las implementaciones del circuito de integración del acelerador 436 y del 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 del 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 realización, el CSRP es uno de los registros 445 que contienen la dirección efectiva de un área en el espacio de direcciones de la aplicación 482 para que el módulo de aceleración de gráficos 446 guarde y restaure el estado del contexto. Este puntero es opcional si no se requiere guardar ningún estado entre trabajos o cuando se da prioridad a un trabajo. El área de guardado/restauración del contexto puede estar anclada en la memoria del sistema.

Tras recibir la llamada de sistema, el sistema operativo 495 puede verificar que la aplicación 480 se ha registrado y que se le ha dado la autoridad para usar el módulo de aceleración de gráficos 446. El sistema operativo 495 luego llama al hipervisor 496 con la información que se muestra en la **Tabla 3**.

Tabla 3 - Parámetros de llamada de SO a hipervisor

1	Un descriptor de trabajo (WD)
2	Un valor de Registro de máscara de autoridad (AMR) (potencialmente enmascarado).
3	Un Puntero de área de guardado/restauración de contexto (CSRP) de dirección efectiva (EA)
4	Un ID de proceso (PID) y un ID de hilo opcional (TID)
5	Un puntero de registro de utilización de acelerador (AURP) de dirección virtual (VA)

6	La dirección virtual del puntero de la tabla de segmentos de almacenamiento (SSTP)
7	Un número de servicio de interrupción lógica (LISN)

Al recibir la llamada del hipervisor, el hipervisor 496 verifica que el sistema operativo 495 se haya registrado y se le haya otorgado la autoridad para usar el módulo de aceleración de gráficos 446. A continuación, el hipervisor 496, pone el elemento de proceso 483 en la lista enlazada de elementos de proceso para el correspondiente tipo de módulo de aceleración de gráficos 446. 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) (potencialmente enmascarado).
3	Un Puntero de área de guardado/restauración de contexto (CSRP) de dirección efectiva (EA)
4	Un ID de proceso (PID) y un ID de hilo opcional (TID)
5	Un puntero de registro de utilización de acelerador (AURP) de dirección virtual (VA)
6	La dirección virtual del puntero de la tabla de segmentos de almacenamiento (SSTP)
7	Un número de servicio de interrupción lógica (LISN)
8	Tabla de vectores de interrupción, derivada de los parámetros de llamada de hipervisor.
9	Un valor de registro de estado (SR)
10	Un identificador de subdivisión lógica (LPID)
11	Un puntero de registro de utilización de acelerador de hipervisor de dirección real (RA)
12	El registro de descriptor de almacenamiento (SDR)

En una realización, el hipervisor inicializa una pluralidad de registros 445 de la sección de integración del acelerador 490.

Como se ilustra en la **Figura 4F**, una realización emplea una memoria unificada direccionable a través de un espacio de dirección 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 dirección de memoria virtual/efectiva para acceder a las memorias de procesador 401-402 y viceversa, simplificando así la programabilidad. En una realización, una primera porción del espacio de direcciones virtual/efectiva se asigna a la memoria del procesador 401, una segunda porción a la segunda memoria del procesador 402, una tercera porción a la memoria de la GPU 420, y así sucesivamente. De este modo, todo el espacio de memoria virtual/efectiva (al que a veces se hace referencia como el espacio de dirección efectiva) se distribuye entre 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 asignada a esa memoria.

En una realización, la circuitería de gestión de desvío/coherencia 494A-494E dentro de una o más de las MMU 439A-439E garantiza la coherencia de caché entre las memorias caché de los procesadores de anfitrión (por ejemplo, 405) y las GPU 410-413 e implementa técnicas de desvío que indican las memorias físicas en las que deben almacenarse ciertos tipos de datos. Aunque se ilustran múltiples instancias de la circuitería de gestión de desvío/coherencia 494A-494E en la **Figura 4F**, la circuitería de desvío/coherencia puede implementarse dentro de la MMU de uno o más procesadores de anfitrión 405 y/o dentro del circuito de integración de acelerador 436.

Una realización permite que la memoria conectada a GPU 420-423 se asigne como parte de la memoria del sistema y se acceda a ella usando tecnología de memoria virtual compartida (SVM), pero sin sufrir los típicos inconvenientes de rendimiento asociados a la coherencia total de memoria caché del sistema. La capacidad de acceder a la memoria adjunta a la GPU 420-423 como memoria del sistema sin una sobrecarga onerosa de coherencia de caché proporciona un entorno operativo beneficioso para la descarga de la GPU. Esta disposición permite que el software del procesador anfitrión 405 configure operandos y acceda a los resultados de los cálculos, sin la sobrecarga de las copias de datos DMA de E/S tradicionales. Estas copias tradicionales implican llamadas de controlador, interrupciones y accesos de E/S mapeadas en la memoria (MMIO) que son todos ineficientes en relación con los accesos a la memoria simple. Al mismo tiempo, la capacidad de acceder a la memoria conectada a la GPU 420-423 sin sobrecargas de coherencia de memoria caché puede ser crucial para el tiempo de ejecución de un cálculo descargado. En caso de tráfico de memoria de escritura de transmisión en continuo sustancial, 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 eficiencia de la

configuración de los operandos, la eficiencia 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 de la GPU.

En una implementación, la selección entre el desvío de la GPU y el desvío del procesador anfitrión está impulsada por una estructura de datos de seguimiento de desvío. Se puede utilizar una tabla de desvío, 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 adjunta a la GPU. La tabla de desvíos puede implementarse en un intervalo de memoria robado de una o más memorias adjuntas a la GPU 420-423, con o sin una memoria caché de desvío en la GPU 410-413 (por ejemplo, para almacenar en memoria caché entradas usadas de manera frecuente/reciente de la tabla de desvíos). Como alternativa, la tabla de desvío completa puede mantenerse dentro de la GPU.

En una implementación, se accede a la entrada de la tabla de desvío asociada con cada acceso a la memoria adjunta a la GPU 420-423 antes del acceso real a la memoria de la GPU, lo que provoca las siguientes operaciones. En primer lugar, las solicitudes locales de la GPU 410-413 que encuentran su página en el desvío de la GPU se reenvían directamente a una memoria de GPU correspondiente 420-423. Las solicitudes locales de la GPU que encuentran su página en el desvío del anfitrión se reenvían al procesador 405 (por ejemplo, a través de un enlace de alta velocidad como se explicó anteriormente). En una realización, las solicitudes del procesador 405 que encuentran la página solicitada en el desvío del procesador anfitrión completan la solicitud como una lectura de memoria normal. Como alternativa, las solicitudes dirigidas a una página con desvío de GPU pueden redirigirse a la GPU 410-413. Luego, la GPU puede realizar la transición de la página a un desvío del procesador anfitrión si no está utilizando la página actualmente.

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

Un mecanismo para cambiar el estado de desvío emplea una llamada 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 para indicarle que cambie el estado de desvío y, para algunas transiciones, realice una operación de vaciado de caché en el anfitrión. La operación de vaciado de caché es necesaria para una transición de desvío del procesador anfitrión 405 al desvío de GPU, pero no es necesaria para la transición opuesta.

En una realización, se mantiene la coherencia de caché representando temporalmente las páginas con desvío de GPU que no pueden almacenare en caché por el procesador de anfitrión 405. Para acceder a estas páginas, el procesador 405 puede solicitar acceso desde la GPU 410, que puede conceder o no el acceso de inmediato, según la implementación. Por tanto, para reducir la comunicación entre el procesador 405 y la GPU 410 es beneficioso garantizar que las páginas con desvío por la GPU sean aquellas que requiere la GPU, pero no el procesador anfitrión 405 y viceversa.

Canalización de procesamiento de gráficos

La **Figura 5** ilustra una canalización de procesamiento de gráficos 500, de acuerdo con una realización. En una realización, un procesador de gráficos puede implementar la canalización de procesamiento de gráficos 500 ilustrada. El procesador de gráficos puede incluirse dentro de los subsistemas de procesamiento paralelo como se describe en este documento, tal como el procesador paralelo 200 de la **Figura 2A**, que, en una realización, es una variante del procesador o procesadores paralelos 112 de la **Figura 1**. Los diversos sistemas de procesamiento paralelo pueden implementar la canalización de procesamiento de gráficos 500 mediante una o más instancias de la unidad de procesamiento paralelo (por ejemplo, la unidad de procesamiento paralelo 202 de la **Figura 2A**) como se describe en el presente documento. Por ejemplo, una unidad sombreadora (por ejemplo, el multiprocesador de gráficos 234 de la **Figura 2D**) puede configurarse para realizar las funciones de una o más de una unidad de procesamiento de vértices 504, una unidad de control de proceso 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 del ensamblador de datos 502, los ensambladores de primitivas 506, 514, 518, la unidad de teselación 510, el rasterizador 522 y la unidad de operaciones de rasterización 526 también pueden ser realizadas por otros motores de procesamiento dentro de una agrupación de procesamiento (por ejemplo, la agrupación de procesamiento 214 de la **Figura 3A**) y una unidad de subdivisión correspondiente (por ejemplo, la unidad de subdivisión 220A-220N de la **Figura 2C**). La canalización de procesamiento de gráficos 500 puede implementarse también usando unidades de procesamiento especializadas para una o más funciones. En una realización, una o más porciones de la canalización de procesamiento de gráficos 500 pueden realizarse mediante una lógica de procesamiento paralelo dentro de un procesador de propósito general (por ejemplo, una CPU). En una realización, una o más partes de la canalización de procesamiento de gráficos 500 pueden acceder a la memoria en chip (por ejemplo, la memoria del procesador paralelo 222 como en la Figura 2A) a través de una interfaz de memoria 528, que puede ser una instancia de la interfaz de memoria 218 de la **Figura 2A**.

En una 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 envía los datos de vértices, incluidos los atributos

de vértices, 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 según lo especificado 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 puede programarse para transformar los datos de vértices desde una representación de coordenadas basada en objetos hasta 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 vértices desde la unidad de procesamiento de vértices 504. El ensamblador de primitivas 506 lee los atributos de vértice almacenados según sea necesario y construye primitivas de gráficos para su procesamiento por la unidad de procesamiento de control de teselación 508. Las primitivas de gráficos incluyen triángulos, segmentos de línea, puntos, parches, etc., según lo admitido por varias interfaces de programación de aplicaciones (API) de procesamiento de gráficos.

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 de una representación de entrada del parche (por ejemplo, las bases del parche) a una representación que es adecuada para su uso en la evaluación de la superficie por 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 la vista asociado con el 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 opera en coordenadas parametrizadas del parche subdividido para generar una representación de superficie y atributos de vértice para cada vértice asociado con las primitivas geométricas.

Una segunda instancia de un ensamblador de primitivas 514 recibe atributos de vértices de la unidad de procesamiento de evaluación de teselación 512, lee atributos de vértice almacenados según sea necesario y construye primitivas de gráficos para su procesamiento por parte de 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 del ensamblador de primitivas 514 según lo especificado por los programas de sombreado de geometría. En una realización, la unidad de procesamiento de geometría 516 está programada para subdividir las primitivas de gráficos en una o más primitivas de gráficos nuevas y calcular los parámetros utilizados para rasterizar las nuevas primitivas de gráficos.

En algunas realizaciones, la unidad de procesamiento de geometría 516 puede añadir o eliminar elementos en el flujo de geometría. La unidad de procesamiento de geometría 516 envía los parámetros y vértices que especifican las primitivas de gráficos nuevas al ensamblador de primitivas 518. El ensamblador de primitivas 518 recibe los parámetros y vértices de la unidad de procesamiento de geometría 516 y construye primitivas de gráficos para su procesamiento por una unidad de escala, selección y recorte de ventana gráfica 520. La unidad de procesamiento de geometría 516 lee datos que están almacenados en la memoria del procesador paralelo o en la memoria del sistema para su uso en el procesamiento de los datos de geometría. La unidad de escala, selección y recorte de la ventana gráfica 520 realiza el recorte, la selección y el escalado de la ventana gráfica y envía primitivas de gráficos procesadas a un rasterizador 522.

El rasterizador 522 puede realizar selección de profundidad y otras optimizaciones basadas en la profundidad. El rasterizador 522 también realiza una conversión de exploración en las nuevas primitivas de gráficos para generar fragmentos y emitir 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 sombreadores de fragmentos o programas de sombreadores de píxeles. La unidad de procesamiento de fragmentos/píxeles 524 transforma fragmentos o píxeles recibidos del rasterizador 522, según lo especificado por los programas de sombreado de fragmentos o píxeles. Por ejemplo, la unidad de procesamiento de fragmentos/píxeles 524 puede programarse para realizar operaciones que incluyen, entre otras, mapeo de texturas, sombreado, combinación, corrección de texturas y corrección de perspectiva para producir fragmentos o píxeles sombreados que se envían 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 bien en la memoria de sistema para su uso cuando se procesan los datos de fragmentos. Los programas de sombreado de fragmentos o de píxeles pueden estar configurados para sombrear a granularidad de muestra, de píxel, de mosaico u otras dependiendo de la tasa 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, entre otras, estarcido, prueba z, combinación y similares, y envía datos de píxeles como datos gráficos procesados para almacenarlos en la memoria de gráficos, por ejemplo, la **memoria** del procesador paralelo **222 como en la Figura 2A**, y/o la **memoria** del sistema **104 como en la Figura 1**, para mostrarse en el uno

o más dispositivos de visualización 110 o para su posterior procesamiento por uno del uno o más procesadores 102 o procesadores paralelos 112. En algunas realizaciones, 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 desde la memoria.

La **Figura 6** ilustra un dispositivo informático 600 que aloja un mecanismo de compartición de datos y comprensión/expansión ("mecanismo de compartición y expansión") 610 según una realización. El dispositivo informático 600 representa un dispositivo de comunicación y procesamiento de datos que incluye (pero sin limitación) dispositivos portátiles inteligentes, teléfonos inteligentes, dispositivos de realidad virtual (VR), pantallas montadas en la cabeza (HMD), ordenadores móviles, dispositivos del Internet de las cosas (IoT), ordenadores portátiles, ordenadores de sobremesa, ordenadores de servidor, etc., y ser similar o igual que el dispositivo informático 100 de la **Figura 1;** en consecuencia, por brevedad, claridad y facilidad de comprensión, muchos de los detalles establecidos anteriormente con referencia a las **Figuras 1-5** no se analizan ni se repiten adicionalmente más adelante.

El dispositivo informático 600 puede incluir además (sin limitaciones) una máquina autónoma o un agente artificialmente inteligente, tal como un agente o máquina mecánica, un agente o máquina electrónica, un agente o máquina virtual, un agente o máquina electromecánica, etc. Ejemplos de máquinas o agentes artificialmente inteligentes pueden incluir (sin limitación) robots, vehículos autónomos (por ejemplo, automóviles autónomos, aviones autónomos, barcos autónomos, etc.), equipos autónomos (vehículos de construcción autónomos, equipos médicos autónomos, etc.), y/o similares. A través de todo este documento, "dispositivo informático" puede denominarse de manera intercambiable "máquina autónoma" o "agente artificialmente inteligente" o simplemente "robot".

Se contempló que si bien a lo largo de este documento se hace referencia a “vehículo autónomo” y “conducción autónoma”, las realizaciones no están limitadas como tales. Por ejemplo, "vehículo autónomo" no se limita a un automóvil, sino que puede incluir cualquier número y tipo de máquinas autónomas, tales como robots, equipos autónomos, dispositivos domésticos autónomos y/o similares, y una o más tareas u operaciones relacionadas con tales máquinas autónomas pueden denominarse de manera intercambiable con la conducción autónoma.

El dispositivo informático 600 puede incluir, además, (sin limitaciones) grandes sistemas informáticos, tales como servidores, ordenadores de sobremesa, etc., y puede incluir, además, decodificadores (por ejemplo, decodificadores de televisión por cable basados en Internet, etc.), dispositivos basados en sistemas de posicionamiento global (GPS), etc. El dispositivo informático 600 puede incluir dispositivos informáticos móviles que funcionan como dispositivos de comunicación, tales como teléfonos móviles, incluidos los teléfonos inteligentes, asistentes digitales personales (PDA), tabletas, ordenadores portátiles, lectores electrónicos, televisores inteligentes, plataformas de televisión, dispositivos ponibles (por ejemplo, gafas, relojes, pulseras, tarjetas inteligentes, joyas, prendas de vestir, etc.), reproductores multimedia, etc. Por ejemplo, en una realización, el dispositivo informático 600 puede incluir un dispositivo informático móvil que emplea una plataforma informática que aloja un circuito integrado ("IC"), tal como un sistema en un chip ("SoC" o "SOC"), que integra varios componentes de hardware y/o software del dispositivo informático 600 en un único chip.

Como se ilustra, en una realización, el dispositivo informático 600 puede incluir cualquier número y tipo de componentes de hardware y/o software, tales como (sin limitación) unidad de procesamiento de gráficos ("GPU" o simplemente "procesador de gráficos") 614, controlador de gráficos (también denominado "controlador de GPU", "lógica del controlador gráfico", "lógica del controlador", controlador de modo de usuario (UMD), UMD, entorno de controlador de modo de usuario (UMDF), UMDF o simplemente "controlador") 616, unidad central de procesamiento ("CPU" o simplemente "procesador de aplicaciones") 612, memoria 608, dispositivos de red, controladores o similares, así como fuentes de entrada/salida (E/S) 604, tales como pantallas táctiles, paneles táctiles, teclados táctiles, teclados virtuales o normales, ratones virtuales o normales, puertos, conectores, etc. El dispositivo informático 600 puede incluir un sistema operativo (OS) 606 que sirve como interfaz entre los recursos de hardware y/o físicos del dispositivo informático 600 y un usuario. Se contempla que el procesador de gráficos 614 y el procesador de aplicaciones 612 pueden ser uno o más procesador o procesador 102 de la **Figura 1.**

Debe apreciarse que para determinadas implementaciones puede preferirse un sistema menos o más equipado que el ejemplo descrito anteriormente. Por lo tanto, la configuración del dispositivo informático 600 puede variar de una implementación a otra dependiendo de numerosos factores, tales como limitaciones de precio, requisitos de rendimiento, mejoras tecnológicas u otras circunstancias.

Las realizaciones pueden implementarse como cualquiera o una combinación de: uno o más microchips o circuitos integrados interconectados usando una placa base, lógica cableada, software almacenado por un dispositivo de memoria y ejecutado por un microprocesador, firmware, un circuito integrado de aplicación específica (ASIC), y /o una matriz de puertas programables en campo (FPGA). Los términos "lógica", "módulo", "componente", "motor" y "mecanismo" pueden incluir, a modo de ejemplo, software o hardware y/o combinaciones de software y hardware.

En una realización, el mecanismo de compartición y expansión 610 puede estar alojado o ser facilitado por el sistema operativo 606 del dispositivo informático 600. En otra realización, el mecanismo de compartición y expansión 610 puede estar alojado por o ser parte de la unidad de procesamiento de gráficos ("GPU" o simplemente "procesador de

gráficos") 614 o el firmware del procesador de gráficos 614. Por ejemplo, el mecanismo de compartición y expansión 610 puede estar integrado en o implementado como parte del hardware de procesamiento del procesador de gráficos 614. De manera similar, en otra realización más, el mecanismo de compartición y expansión 610 puede estar alojado por, o ser parte de la unidad de procesamiento central ("CPU" o simplemente "procesador de aplicaciones") 612. Por ejemplo, el mecanismo de compartición y expansión 610 puede estar integrado en, o implementarse como parte del hardware de procesamiento del procesador de aplicaciones 612. En otra realización más, el mecanismo de compartición y expansión 610 puede estar alojado por, o ser parte de cualquier número y tipo de componentes del dispositivo informático 600, tal como por ejemplo una parte del mecanismo de compartición y expansión 610 puede estar alojada por, o ser parte del sistema operativo 606, otra porción puede estar alojada por, o ser parte del procesador de gráficos 614, otra porción puede estar alojada por, o ser parte del procesador de aplicaciones 612, mientras que una o más porciones del mecanismo de compartición y expansión 610 pueden estar alojadas por, o ser parte del sistema operativo 606 y/o cualquier número y tipo de dispositivos del dispositivo informático 600. Se contempla que una o más porciones o componentes del mecanismo de compartición y expansión 610 pueden emplearse como hardware, software y/o firmware.

Se contempla que las realizaciones no se limitan a ninguna implementación o alojamiento particular del mecanismo de compartición y expansión 610 y que el mecanismo de compartición y expansión 610 y uno o más de sus componentes pueden implementarse como hardware, software, firmware o cualquier combinación de los mismos.

El dispositivo informático 600 puede alojar interfaz o interfaces de red para proporcionar acceso a una red, tal como una LAN, una red de área extensa (WAN), una red de área metropolitana (MAN), una red de área personal (PAN), Bluetooth, una red en la nube, una red móvil (por ejemplo, 3ª generación (3G), 4ª generación (4G), etc.), una intranet, Internet, etc. La interfaz o interfaces de red pueden incluir, por ejemplo, una interfaz de red inalámbrica que tiene una antena, que puede representar una o más antenas. La interfaz o interfaces de red también pueden incluir, por ejemplo, una interfaz de red por cable para comunicarse con dispositivos remotos por medio de un cable de red, que puede ser, por ejemplo, un cable Ethernet, un cable coaxial, un cable de fibra óptica, un cable serie o un cable paralelo.

Se pueden proporcionar realizaciones, por ejemplo, como un producto de programa informático que puede incluir uno o más medios legibles por máquina que tienen almacenados en los mismos instrucciones ejecutables por máquina que, cuando se ejecutan por una o más máquinas tales como un ordenador, una red de ordenadores u otros dispositivos electrónicos, pueden dar como resultado que la una o más máquinas lleven a cabo operaciones de acuerdo con las realizaciones descritas en el presente documento. Un medio legible por máquina puede incluir, pero sin limitación, disquetes, discos ópticos, CD-ROM (memorias de sólo lectura en disco compacto) y discos magnetoópticos, ROM, RAM, EPROM (memorias de sólo lectura programables y borrables), EEPROM (memorias de sólo lectura programables y borrables eléctricamente), tarjetas magnéticas u ópticas, memoria flash u otro tipo de soporte/medio legible por máquina adecuado para almacenar instrucciones ejecutables por máquina.

Además, las realizaciones pueden descargarse como un producto de programa informático, en donde el programa puede transferirse desde un ordenador remoto (por ejemplo, un servidor) a un ordenador solicitante (por ejemplo, un cliente) por medio de una o más señales de datos incorporadas en y/o moduladas por una onda portadora u otro medio de propagación a través de un enlace de comunicación (por ejemplo, un módem y/o conexión de red).

A través de todo del documento, el término "usuario" puede denominarse de manera intercambiable "espectador", "observador", "persona", "individuo", "usuario final" y/o similares. Cabe señalar que, a lo largo de todo este documento, se puede hacer referencia a términos como "dominio de gráficos" de manera intercambiable con "unidad de procesamiento de gráficos", "procesador de gráficos" o simplemente "GPU" y, de manera similar, "dominio de CPU" o "dominio de anfitrión" se pueden hacer referencia de manera intercambiable a "unidad de procesamiento de ordenador", "procesador de aplicaciones" o simplemente "CPU".

Cabe señalar que, términos y expresiones como "nodo", "nodo informático", "servidor", "dispositivo de servidor", "ordenador en la nube", "servidor en la nube", "ordenador de servidor en la nube", "máquina", "máquina de anfitrión", "dispositivo", "dispositivo informático", "ordenador", "sistema informático" y similares, pueden usarse de manera intercambiable en este documento. Cabe señalar además que, términos como "aplicación", "aplicación de software", "programa", "programa de software", "paquete", "paquete de software" y similares, pueden usarse de manera intercambiable a lo largo de todo este documento. Además, términos como "trabajo", "entrada", "solicitud", "mensaje" y similares se pueden usar de manera intercambiable en este documento.

La **Figura 7** ilustra el mecanismo de compartición y expansión 610 de la **Figura 6** según una realización. Para abreviar, muchos de los detalles ya analizados con referencia a las **Figuras 1-6** no se repiten ni se analizan a continuación. En una realización, el mecanismo de compartición y expansión 610 puede incluir cualquier número y tipo de componentes, tales como (sin limitaciones): lógica de detección/observación 701; lógica de generación/mapeo de bibliotecas 703; lógica de compartición/recuperación de datos 705; lógica de comunicación/compatibilidad 707; y lógica de compresión/expansión 709.

Como se mencionó anteriormente, la compartición de datos es una manera eficiente de tener acceso a todos los datos relevantes sin pasar por largos procesos y procedimientos o reinventar la rueda. Sin embargo, las técnicas

convencionales de compartición de datos están limitadas en cuanto a su uso, ya que no permiten compartir datos entre sistemas de procesamiento, donde un sistema de procesamiento puede recuperar cualquier parte de los datos compartidos si esa parte es relevante para el trabajo que se realiza por o en el sistema de procesamiento.

Las realizaciones proporcionan una técnica novedosa para ofrecer compartir datos producidos en cualquier número y tipo de sistemas o dispositivos de procesamiento, tal como el procesador de gráficos 614, el procesador de aplicaciones 612, la matriz de puertas programables en campo (FPGA), el circuito integrado específico de la aplicación (ASIC) y/o similares, utilizando una biblioteca de superficies. Por ejemplo, en una realización, cualquier procesador que trabaje en la misma convolución puede recuperar datos de la biblioteca de superficies, donde estos datos han sido producidos por otro procesador. En otra realización, estos datos pueden almacenarse de forma persistente entre ejecuciones.

En una realización, la lógica de detección/observación 701 puede utilizarse para detectar y observar procesadores, tal como el procesador de gráficos 614, mientras trabajan, por ejemplo, en convolución, donde esta información puede compartirse con la lógica de compartición/recuperación 703 y la lógica de biblioteca 705. Por ejemplo, si el procesador de gráficos 614 está trabajando en redes neuronales de convolución (CNN), el procesador de gráficos 614 puede verse facilitado por la lógica de compartición/recuperación 703 para almacenar los datos de la red neuronal (NN) intermedios como superficie de datos en la biblioteca de superficies 731 ubicada en una o más bases de datos en la nube o centros de datos, tales como la(s) base(s) de datos 730, en comunicación con los dispositivos informáticos 600, 740 a través de uno o más medios de comunicación 725, tales como la red en la nube.

En una realización, la lógica de biblioteca 703 puede utilizarse para generar una o más bibliotecas de superficies, tales como la biblioteca de superficies 731, según se desee o se necesite, ya que la lógica de detección/observación 701 detecta que el procesador de gráficos 614 está trabajando en la convolución y tiene datos que pueden almacenarse en la biblioteca de superficies 731 y usarse posteriormente por uno o más dispositivos de procesamiento adicionales, tales como el procesador de aplicaciones 612, el procesador de aplicaciones 742, el procesador de gráficos 744, etc. En otra realización, si hay una cantidad suficiente de espacio disponible en las bibliotecas de superficies, es posible que no se generen bibliotecas de superficies adicionales o nuevas y la lógica de compartición/recuperación de datos 703 puede utilizarse simplemente para activar el procesador de gráficos 614 para que almacene sus datos en una de las bibliotecas, tal como la biblioteca de superficies 731, y establecer un mapeo de los datos almacenados con el dispositivo de procesamiento correspondiente, tal como el procesador de gráficos 614, como lo facilita la lógica de biblioteca 705.

Una vez que los datos se han almacenado, en una realización, ahora están disponibles para otros dispositivos de procesamiento. ya sea el procesador de aplicaciones 612 en el dispositivo informático 600, o uno o más dispositivos de procesamiento en otro sistema de aprendizaje profundo, tal como los procesadores de aplicaciones y gráficos 742, 744 en el dispositivo informático 740, a través del medio de comunicación 730. Por ejemplo, si el procesador de gráficos 744 también está trabajando en la misma convolución o problema o similar que el procesador de gráficos 614, el procesador de gráficos 744 puede acceder a los datos NN intermedios en la biblioteca de superficies 731, como lo facilita la lógica de compartición/recuperación 703.

Por ejemplo, si los dispositivos informáticos 600, 740 son dos máquinas autónomas (por ejemplo, vehículos, robots, etc.) que están uno al lado del otro, que tienen una vista similar, experimentan las mismas condiciones ambientales, etc., los dos procesadores de gráficos 612, 742, respectivamente, pueden compartir los datos NN utilizando la biblioteca de superficies 731 en la(s) base(s) de datos 730 a través del medio o medios de comunicación 725.

Además, en una realización, cualquier superficie producida en esta materia puede comprimirse opcionalmente mediante la lógica de compresión/expansión 709 para un tiempo de transmisión aún más rápido. Además, las superficies compartidas, como las facilita la biblioteca compartida 731, pueden usarse para verificar los resultados mediante múltiples sistemas de aprendizaje profundo, tales como las máquinas autónomas 600, 740, como se ilustra con más detalle en la **Figura 8A**.

Las realizaciones proporcionan, además, una técnica novedosa para la reexpansión de modelos comprimidos para lograr un alto rendimiento y eficiencia de comunicación. Por ejemplo, ciertos modelos pueden ser demasiado grandes para enviarlos por aire y, por lo tanto, la compresión de modelos puede usarse para reducir la cantidad de capas y uno o más de los modelos pueden expandirse nuevamente a su tamaño original, tal como en la computación de alto rendimiento (HPC).

Las técnicas convencionales simplemente están limitadas en su enfoque para comunicar modelos de datos entre varias máquinas autónomas, tales como vehículos, etc., lo que es ineficiente en la mayoría de los casos, como cuando hay millones de vehículos autónomos, tales como las máquinas autónomas 600 y 740, involucrados en la carretera, lo que requiere una gran cantidad de ancho de banda para entregar modelos de datos a través de uno o más medios de comunicación 725.

Las realizaciones proporcionan una técnica novedosa para comprimir modelos de datos, a la vez que los expanden con un artefacto para permitir una comunicación fluida a través de uno o más medios de comunicación 725 sin que resulte costoso, como por ejemplo en términos de recursos del sistema o de la red, ancho de banda, etc.

Por ejemplo, en el caso de que haya una gran cantidad de vehículos autónomos, tales como las máquinas autónomas 600, 740, en la carretera, sería deseable facilitar la comunicación de información entre dichos vehículos, tal como datos de tráfico, información del tiempo, alertas de emergencia, etc., con la frecuencia y rapidez deseadas o necesarias. Sin embargo, los sistemas convencionales no son capaces de expandir modelos comprimidos.

En una realización, como se ilustra adicionalmente con respecto a la **Figura 8B**, la lógica de compresión/expansión 709 se utiliza para comprimir un modelo de datos y asignar un artefacto al modelo comprimido de modo que el artefacto sirva tanto como una extensión del modelo comprimido como una forma de identificación si el modelo comprimido se comunica desde una máquina 600 a otra máquina 740 a través del medio o medios de comunicación 725.

Se contempla que el "artefacto" o el uso del artefacto es simplemente según una realización y que puede haber otras varias técnicas mediante las cuales un modelo comprimido puede volver a combinarse para obtener el modelo original, dichas técnicas adicionales pueden incluir (sin limitación) el uso de un reentrenamiento "ligero" dentro del vehículo, "pistas" de vehículos/controladores similares y/o similares.

Por ejemplo, en una realización, un modelo original o regular de datos se comprime aplicando un artefacto mediante la lógica de compresión/expansión 709, donde este modelo comprimido y el artefacto correspondiente se comunican desde una máquina autónoma 600 a otra máquina autónoma 740 a través de uno o más medios de comunicación 725 (por ejemplo, la nube, Internet, etc.). El artefacto se recibe entonces en la máquina autónoma 740 (en un vehículo autónomo, por ejemplo), seguido de la recepción de una combinación de modelo comprimido y artefacto. Los dos se separan entonces y la máquina autónoma 740 puede ahora utilizar el modelo en su modelo original y sin comprimir.

Además, la lógica de comunicación/compatibilidad 707 puede utilizarse para facilitar la comunicación y compatibilidad necesarias entre cualquier número de dispositivos del dispositivo informático 600 y varios componentes del mecanismo de compartición y expansión 610.

La lógica de comunicación/compatibilidad 707 puede usarse para facilitar la comunicación dinámica y la compatibilidad entre el dispositivo informático 600 y cualquier número y tipo de otros dispositivos informáticos (tales como dispositivo informático móvil, ordenador de escritorio, dispositivo informático de servidor, etc.); dispositivos o componentes de procesamiento (tales como CPU, GPU, etc.); dispositivos de captura/localización/detección (tales como componentes de captura/detección que incluyen cámaras, cámaras de detección de profundidad, sensores de cámara, sensores de rojo, verde, azul ("RGB" o "rgb"), micrófonos, etc.); dispositivos de visualización (tales como componentes de salida, que incluyen pantallas de visualización, áreas de visualización, proyectores de visualización, etc.); componentes de reconocimiento de usuario/contexto y/o sensores/dispositivos de identificación/verificación (tales como sensores/detectores biométricos, escáneres, etc.); base o bases de datos 730, tales como memoria o dispositivos de almacenamiento, bases de datos y/o fuentes de datos (tales como dispositivos de almacenamiento de datos, discos duros, unidades de estado sólido, discos duros, tarjetas o dispositivos de memoria, circuitos de memoria, etc.); medio o medios de comunicación 725, tales como uno o más canales o redes de comunicación (por ejemplo, redes en la nube, Internet, intranets, redes celulares, redes de proximidad, tales como Bluetooth, Bluetooth de baja energía (BLE), Bluetooth inteligente, Wi-Fi de proximidad, identificación por radiofrecuencia (RFID), comunicación de campo cercano (NFC), red de área corporal (BAN), etc.); comunicaciones inalámbricas o por cable y protocolos pertinentes (por ejemplo, Wi-Fi®, WiMAX, Ethernet, etc.); técnicas de conectividad y gestión de ubicación; aplicaciones de software/sitios web (por ejemplo, sitios web de redes sociales y/o comerciales, etc., aplicaciones comerciales, juegos y otras aplicaciones de entretenimiento, etc.); y lenguajes de programación, etc., garantizando al mismo tiempo la compatibilidad con tecnologías, parámetros, protocolos, normas, etc., cambiantes.

Además, cualquier uso de una marca, palabra, término, frase, nombre y/o acrónimo en particular, tales como "detectar", "observar", "entrenar", "seleccionar", "comprimir", "asociar", "aplicar", "compartir", "almacenar", "recuperar", "biblioteca de superficies", "modelo comprimido", "modelo expandido", "expandir", "conjunto de entrenamiento", "agente", "máquina", "vehículo", "robot", "conducir", "CNN", "DNN", "NN", "unidad de ejecución", "UE", "memoria local compartida", "SLM", "flujos de gráficos", "caché", "caché de gráficos", "GPU", "procesador de gráficos", "dominio de GPU", "GPGPU", "CPU", "procesador de aplicaciones", "dominio de CPU", "controlador de gráficos", "carga de trabajo", "aplicación", "canalización de gráficos", "procesos de canalización", "API", "API 3D", "OpenGL®", "DirectX®", "hardware", "software", "agente", "controlador de gráficos", "controlador de gráficos en modo kernel", "controlador en modo usuario", "entorno de controlador en modo usuario", "memoria intermedia", "memoria intermedia de gráficos", "tarea", "proceso", "operación", "aplicación de software", "juego", etc., no deben interpretarse como si limitaran las realizaciones a software o dispositivos que llevan esa etiqueta en productos o en literatura externa a este documento.

Se contempla que se puede agregar y/o quitar cualquier cantidad y tipo de componentes del mecanismo de compresión y expansión 610 para facilitar diversas realizaciones, incluyendo agregar, quitar y/o mejorar ciertas características. Para mayor brevedad, claridad y facilidad de comprensión del mecanismo de compresión y expansión 610, muchos de los componentes estándar y/o conocidos, tales como los de un dispositivo informático, no se muestran ni se analizan

aquí. Se contempla que las realizaciones, como se describen en el presente documento, no se limiten a ninguna tecnología, topología, sistema, arquitectura y/o norma particular y sean lo suficientemente dinámicas para adoptar y adaptarse a cualquier cambio futuro.

La **Figura 8A** ilustra una configuración de red 800 para compartir y recuperar datos entre sistemas de procesamiento de acuerdo con una realización. Para abreviar, muchos de los detalles previamente analizados con referencia a las **Figuras 1-7** pueden no analizarse o repetirse posteriormente en este punto. Cualquier proceso relacionado con la configuración 800 puede ser realizado por lógica de procesamiento que puede comprender hardware (por ejemplo, circuitos, lógica dedicada, lógica programable, etc.), software (tal como instrucciones que se ejecutan en un dispositivo de procesamiento) o una combinación de los mismos, como lo facilita el mecanismo de compartición y expansión 610 de la **Figura 6**. Los procesos asociados con la configuración 800 pueden ilustrarse o enumerarse en secuencias lineales para mayor brevedad y claridad en la presentación; sin embargo, se contempla que cualquier número de ellos puede realizarse en paralelo, de manera asincrónica o en diferentes órdenes. Además, las realizaciones no están limitadas a ninguna ubicación arquitectónica, entorno, configuración o estructura particular de procesos y/o componentes, tal como la configuración 800.

Como se ilustra aquí y se describe con referencia a la **Figura 7**, la nueva técnica para compartir y recuperar datos, facilitada por el mecanismo de compartición y expansión 610, puede alojarse y usarse por cualquier número y tipo de dispositivos informáticos, tal como las máquinas autónomas 600, 740, 810 (por ejemplo, vehículos, robots, etc.) a través de uno o más medios de comunicación 725, tales como una red en la nube.

En la realización ilustrada, la(s) base(s) de datos 730, tales como la(s) base(s) de datos en la nube o el centro o centros de datos, pueden alojar una o más bibliotecas de superficie, tal como la biblioteca de superficies 731, donde los datos, tales como los datos de NN intermedios, pueden almacenarse, recuperarse y compartirse por cualquier número y tipo de dispositivos de procesamiento, tales como los procesadores 612, 614, 742, 744 y 812, 814 de las máquinas autónomas 600, 740 y 810, respectivamente.

La **Figura 8B** ilustra una secuencia de transacciones 850 para compartir y recuperar datos entre sistemas de procesamiento de acuerdo con una realización. Para abreviar, muchos de los detalles previamente analizados con referencia a las **Figuras 1-7** pueden no analizarse o repetirse posteriormente en este punto. Cualquiera de los procesos relacionados con la secuencia de transacciones 850 pueden ser realizados por lógica de procesamiento que puede comprender hardware (por ejemplo, circuitos, lógica dedicada, lógica programable, etc.), software (tal como instrucciones que se ejecutan en un dispositivo de procesamiento) o una combinación de los mismos, tal como lo facilita el mecanismo de compartición y expansión 610 de la **Figura 6**. Los procesos asociados con la secuencia de transacciones 850 pueden ilustrarse o indicarse en secuencias lineales para mayor brevedad y claridad en la presentación; sin embargo, se contempla que cualquier número de ellos pueda realizarse en paralelo, de forma asincrónica o en diferentes órdenes. Además, las realizaciones no se limitan a ninguna ubicación arquitectónica, entorno, configuración o estructura particular de procesos y/o componentes, tal como la secuencia de transacciones 850.

Como se ilustra, en una realización, se selecciona el modelo original 851A y se comprime en el bloque 853 y se expande con el artefacto 855A, lo que da como resultado el modelo comprimido 857A. Este modelo comprimido 857A se transmite luego junto con el artefacto desde una máquina autónoma o cualquier dispositivo informático a otra máquina autónoma o cualquier otro dispositivo informático en el bloque 859 a través del medio de comunicación 725 (por ejemplo, red en la nube, Internet, red de proximidad, Bluetooth, etc.).

El modelo comprimido transmitido se recibe en la máquina autónoma receptora en 861, donde se recibe como un paquete combinado 863 que tiene una combinación del modelo comprimido 857B (que es el mismo que el modelo comprimido 857A) y el artefacto 855B (que es el mismo que el artefacto 855A). En la máquina autónoma, se elimina el artefacto 855B y el modelo comprimido 857B se expande nuevamente al modelo sin comprimir 851B (que es el mismo que el modelo original sin comprimir 851).

La **Figura 9** ilustra un método 900 para facilitar la compartición de datos entre dispositivos de procesamiento utilizando una biblioteca de superficies de acuerdo con una realización. Para abreviar, muchos de los detalles previamente analizados con referencia a las **Figuras 1-8B** pueden no analizarse o repetirse posteriormente en este punto. Cualquier proceso relacionado con el método 900 puede ser realizado por lógica de procesamiento que puede comprender hardware (por ejemplo, circuitos, lógica dedicada, lógica programable, etc.), software (tal como instrucciones que se ejecutan en un dispositivo de procesamiento) o una combinación de los mismos, tal como lo facilita el mecanismo de compartición y expansión 610 de la **Figura 6**. Los procesos asociados con el método 900 pueden ilustrarse o indicarse en secuencias lineales para mayor brevedad y claridad en la presentación; sin embargo, se contempla que cualquier número de ellos puede realizarse en paralelo, de manera asincrónica o en diferentes órdenes.

El método 900 comienza en el bloque 901 con la detección de un procesador, tal como un procesador de gráficos en una máquina autónoma, que trabaja en una CNN. En el bloque 903, se facilita que el procesador almacene cualquier dato NN intermedio relacionado con la CNN, tal como una superficie de datos de una biblioteca de superficies en una

base de datos, tal como una base de datos en la nube, a través de un medio de comunicación, tal como una red en la nube. En una realización, esta biblioteca de superficies y otras bibliotecas de superficies similares pueden crearse en varias bases de datos o centros de datos y ponerse a disposición de cualquier número y tipo de procesadores en varias máquinas autónomas.

En el bloque 905, otro procesador de esta u otra máquina autónoma está trabajando en la misma CNN y accede a la superficie de datos almacenada en la biblioteca de superficies por el procesador de gráficos. Este procesador puede ser otro procesador, puede ser un procesador de aplicaciones en la misma u otra máquina autónoma u otro procesador de gráficos u otros procesadores similares en la misma máquina autónoma o en otra. En el bloque 907, este segundo procesador, tal como un procesador de gráficos, en otra máquina autónoma accede entonces a la superficie de datos y procede a recuperar estos datos almacenados para usarlos para trabajar en la CNN. Además, la superficie producida de este modo puede comprimirse opcionalmente para un tiempo de transmisión más rápido.

Descripción general del aprendizaje automático

Un algoritmo de aprendizaje automático es un algoritmo que puede aprender basándose en un conjunto de datos. Se pueden diseñar realizaciones de algoritmos de aprendizaje automático para modelar abstracciones de alto nivel dentro de un conjunto de datos. Por ejemplo, pueden usarse algoritmos de reconocimiento de imágenes para determinar a cuál de varias categorías pertenece una entrada dada; los algoritmos de regresión pueden emitir un valor numérico dada una entrada; y pueden usarse los algoritmos de reconocimiento de patrones para generar texto traducido o para realizar texto a habla y/o reconocimiento de habla.

Un tipo ilustrativo de algoritmo de aprendizaje automático es una red neuronal. Hay muchos tipos de redes neuronales; un tipo sencillo de red neuronal es una red de realimentación prospectiva. Una red de realimentación prospectiva puede implementarse como un grafo acíclico en el que los nodos están dispuestos en capas. Típicamente, una topología de red de realimentación prospectiva 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 la salida en la capa de salida. Los nodos de red están completamente conectados mediante bordes a los nodos en capas adyacentes, pero no hay bordes entre nodos dentro de cada capa. Los datos recibidos en los nodos de una capa de entrada de una red de realimentación prospectiva se propagan (es decir, "se realimentan prospectivamente") a los nodos de la capa de salida mediante una función de activación que calcula los estados de los nodos de cada capa sucesiva en la red basándose en coeficientes ("pesos") asociados, respectivamente, con cada uno de los bordes que conectan las capas. Dependiendo del modelo específico que represente el algoritmo que se esté ejecutando, la salida del algoritmo de red neuronal puede adoptar diversas formas.

Antes de que pueda usarse un algoritmo de aprendizaje automático para modelar un problema particular, se entrena el algoritmo usando un conjunto de datos de entrenamiento. Entrenar una red neuronal implica seleccionar una topología de red, usar un conjunto de datos de entrenamiento que representa un problema que es modelado por la red, y ajustar los pesos hasta que el modelo de red rinde 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 "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 con 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 usado para entrenar el algoritmo. El proceso de entrenamiento puede ser computacionalmente intensivo y puede requerir una cantidad significativa de tiempo en un procesador convencional de propósito general. En consecuencia, se utiliza hardware de procesamiento paralelo para entrenar muchos tipos de algoritmos de aprendizaje automático. Esto es particularmente útil para optimizar el entrenamiento de redes neuronales, debido a que los cómputos realizados en el ajuste de los coeficientes en redes neuronales se prestan de manera natural 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 **Figura 10** es un diagrama generalizado de una pila de software de aprendizaje automático 1000. Una aplicación de aprendizaje automático 1002 se puede configurar para entrenar una red neuronal utilizando un conjunto de datos de entrenamiento o para utilizar una red neuronal profunda entrenada para implementar la inteligencia automática. La aplicación de aprendizaje automático 1002 puede incluir una funcionalidad de entrenamiento y de inferencia para una red neuronal y/o software especializado que puede usarse para entrenar una red neuronal antes del despliegue. La aplicación de aprendizaje automático 1002 puede implementar cualquier tipo de inteligencia automática incluyendo, pero sin limitación, reconocimiento de imágenes, mapeo y localización, navegación autónoma, síntesis de habla, formación de imágenes médicas o traducción de idioma.

La aceleración de hardware para la aplicación de aprendizaje automático 1002 se puede habilitar a través de un entorno de aprendizaje automático 1004. La estructura de aprendizaje automático 1004 puede proporcionar una biblioteca de primitivas de aprendizaje automático. Las primitivas de aprendizaje automático son operaciones básicas que se realizan comúnmente mediante algoritmos de aprendizaje automático. Sin la estructura de aprendizaje automático 1004, se requeriría que los desarrolladores de algoritmos de aprendizaje automático crearan y optimizaran la lógica computacional principal asociada con el algoritmo de aprendizaje automático, y que volvieran a optimizar la lógica computacional a medida que se desarrollan nuevos procesadores paralelos. En cambio, la aplicación de aprendizaje automático se puede configurar para realizar los cálculos necesarios utilizando las primitivas proporcionadas por el entorno de aprendizaje automático 1004. Las primitivas ilustrativas incluyen convoluciones de tipo tensor, funciones de activación y agrupamiento, que son operaciones computacionales que se realizan mientras se entrena una red neuronal convolucional (CNN). La estructura de aprendizaje automático 1004 puede proporcionar también primitivas para implementar subprogramas de álgebra lineal básicos realizados por muchos algoritmos de aprendizaje automático, tales como operaciones matriciales y vectoriales.

El entorno de aprendizaje automático 1004 puede procesar los datos de entrada recibidos de la aplicación de aprendizaje automático 1002 y generar la entrada adecuada para un entorno de cálculo 1006. El entorno de cálculo 1006 puede abstraer las instrucciones subyacentes proporcionadas al controlador GPGPU 1008 para permitir que el entorno de aprendizaje automático 1004 aproveche la aceleración de hardware a través del hardware GPGPU 1010 sin requerir que el entorno de aprendizaje automático 1004 tenga un conocimiento profundo de la arquitectura del hardware GPGPU 1010. Además, el entorno de cálculo 1006 puede habilitar la aceleración de hardware para el entorno de aprendizaje automático 1004 en una variedad de tipos y generaciones del hardware GPGPU 1010.

Aceleración de aprendizaje automático de GPGPU

La **Figura 11** ilustra una unidad de procesamiento de gráficos de uso general altamente paralela 1100, de acuerdo con una realización. En una realización, la unidad de procesamiento de propósito general (GPGPU) 1100 se puede configurar para que sea particularmente eficiente en el procesamiento del tipo de cargas de trabajo computacionales asociadas con el entrenamiento de redes neuronales profundas. Además, la GPGPU 1100 se puede vincular directamente a otras instancias de la GPGPU para crear un grupo de múltiples GPU para mejorar la velocidad de entrenamiento para redes neuronales particularmente profundas.

La GPGPU 1100 incluye una interfaz de anfitrión 1102 para habilitar una conexión con un procesador anfitrión. En una realización, la interfaz de anfitrión 1102 es una interfaz PCI Express. Sin embargo, la interfaz de anfitrión también puede ser una interfaz de comunicaciones específica del proveedor o una estructura de comunicaciones. La GPGPU 1100 recibe comandos desde el procesador de anfitrión y usa un planificador global 1104 para distribuir hilos de ejecución asociados con estos comandos a un conjunto de agrupaciones de cálculo 1106A-H. Las agrupaciones de cálculo 1106A-H comparten una memoria caché 1108. La memoria caché 1108 puede servir como una caché de nivel superior para memorias caché dentro de las agrupaciones de cálculo 1106A-H.

La GPGPU 1100 incluye memoria 1114A-B acoplada con las agrupaciones de cálculo 1106A-H a través de un conjunto de controladores de memoria 1112A-B. En diversas realizaciones, la memoria 1114A-B puede incluir diversos 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 sincrónica (SGRAM), que incluye memoria de tasa de datos doble de gráficos (GDDR). En una realización, las unidades de memoria 224A-N pueden incluir también memoria 3D apilada, que incluye, pero sin limitación, memoria de alto ancho de banda (HBM).

En una realización, cada agrupación de cálculo GPLAB06A-H incluye un conjunto de multiprocesadores de gráficos, tales como el multiprocesador de gráficos 400 de la Figura 4A. Los multiprocesadores de gráficos de la agrupación de cálculo tienen múltiples tipos de unidades de lógica de enteros y de coma flotante que pueden realizar operaciones computacionales con un intervalo de precisiones que incluyen unas adecuadas para cálculos de aprendizaje automático. Por ejemplo, y en una realización, al menos un subconjunto de las unidades de coma flotante en cada una de las agrupaciones de cálculo 1106A-H puede estar configurado para realizar operaciones de coma flotante de 16 bits o de 32 bits, mientras que un subconjunto diferente de las unidades de coma flotante puede estar configurado para realizar operaciones de coma flotante de 64 bits.

Múltiples instancias de la GPGPU 1100 pueden configurarse para funcionar como una agrupación de cálculo. El mecanismo de comunicación usado por la agrupación de cálculo para la sincronización y el intercambio de datos varía a lo largo de las realizaciones. En una realización, las múltiples instancias de la GPGPU 1100 se comunican a través de la interfaz de anfitrión 1102. En una realización, la GPGPU 1100 incluye un concentrador de E/S 1108 que acopla la GPGPU 1100 con un enlace de GPU 1110 que permite una conexión directa a otras instancias de la GPGPU. En una realización, el enlace de GPU 1110 está acoplado a un puente de GPU a GPU dedicado que permite la comunicación y sincronización entre múltiples instancias de la GPGPU 1100. En una realización, el enlace GPU 1110 se acopla con una interconexión de alta velocidad para transmitir y recibir datos a otras GPGPU o procesadores paralelos. En una realización, las múltiples instancias de la GPGPU 1100 están ubicadas en sistemas de procesamiento de datos separados y se comunican a través de un dispositivo de red al que se puede acceder a través

de la interfaz de anfitrión 1102. En una realización, el enlace GPU 1110 se puede configurar para permitir una conexión a un procesador principal además de o como alternativa a la interfaz de anfitrión 1102.

Si bien la configuración ilustrada de la GPGPU 1100 se puede configurar para entrenar redes neuronales, una realización proporciona una configuración alternativa de la GPGPU 1100 que se puede configurar para su implementación dentro de una plataforma de inferencia de alto rendimiento o baja potencia. En una configuración de inferencia, la GPGPU 1100 incluye menos de las agrupaciones de cálculo 1106A-H con relación a la configuración de entrenamiento. Adicionalmente, una tecnología de memoria asociada con la memoria 1114A-B puede diferir entre las configuraciones de inferencia y de entrenamiento. En una realización, la configuración de inferencia de la GPGPU 1100 puede admitir las instrucciones específicas de inferencia. Por ejemplo, una configuración de inferencia puede proporcionar soporte para una o más instrucciones de producto de producto escalar de número entero de 8 bits, que se utilizan comúnmente durante las operaciones de inferencia para redes neuronales implementadas.

La **Figura 12** ilustra un sistema informático multi-GPU 1200, de acuerdo con una realización. El sistema informático de múltiples GPU 1200 puede incluir un procesador 1202 acoplado a múltiples GPGPU 1206A-D mediante un conmutador de interfaz de anfitrión 1204. El conmutador de interfaz de anfitrión 1204, en una realización, es un dispositivo de conmutador PCI express que acopla el procesador 1202 a un bus PCI express a través del cual el procesador 1202 puede comunicarse con el conjunto de GPGPU 1206A-D. Cada una de las múltiples GPGPU 1206A-D puede ser una instancia de la GPGPU 1100 de la Figura 11. Las GPGPU 1206A-D pueden interconectarse mediante un conjunto de enlaces de GPU a GPU de punto a punto de alta velocidad 1216. Los enlaces de GPU a GPU de alta velocidad se pueden conectar a cada una de las GPGPU 1206A-D a través de un enlace de GPU especializado, tal como el enlace de GPU 1110 como en la Figura 11. Los enlaces de GPU de P2P 1216 posibilitan la comunicación directa entre cada una de las GPGPU 1206A-D sin requerir la comunicación a través del bus de interfaz de anfitrión a la que está conectado el procesador 1202. Con el tráfico de GPU a GPU dirigido a los enlaces de GPU P2P, el bus de interfaz de anfitrión permanece disponible para el acceso a la memoria del sistema o para comunicarse con otras instancias del sistema informático multi-GPU 1200, por ejemplo, a través de uno o más dispositivos de red. Aunque en la realización ilustrada las GPGPU 1206A-D se conectan al procesador 1202 mediante el conmutador de interfaz de anfitrión 1204, en una realización, el procesador 1202 incluye el soporte directo para los enlaces de GPU de P2P 1216 y puede conectarse directamente a las GPGPU 1206A-D.

Implementaciones de red neuronal de aprendizaje automático

La arquitectura informática proporcionada por las realizaciones descritas en este documento se puede configurar para realizar los tipos de procesamiento paralelo que son particularmente adecuados para el entrenamiento y la implementación de redes neuronales para el aprendizaje automático. Una red neuronal puede generalizarse como una red de funciones que tienen una relación de grafo. Como es bien sabido en la técnica, existen diversos tipos de implementaciones de redes neuronales utilizadas en el aprendizaje automático. Un tipo ejemplar de red neuronal es la red de realimentación prospectiva, tal como se describió anteriormente.

Un segundo tipo ilustrativo de red neuronal es la red neuronal convolucional (CNN). Una CNN es una red de realimentación prospectiva especializada para procesar datos que tienen una topología conocida, similar a una cuadrícula, tal como datos de imágenes. En consecuencia, las CNN se utilizan comúnmente para aplicaciones de visión computacional y reconocimiento de imágenes, pero también se pueden utilizar para otros tipos de reconocimiento de patrones, tal como el procesamiento del habla y el lenguaje. Los nodos en la capa de entrada de CNN están organizados en un conjunto de "filtros" (detectores de características inspirados por los campos receptivos encontrados en la retina), y la salida de cada conjunto de filtros se propaga a nodos en capas sucesivas de la red. Los cálculos para una CNN incluyen la aplicación de la operación matemática de convolución a cada filtro para producir la salida de ese filtro. La 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 la terminología de redes convolucionales, la primera función de la convolución puede denominarse entrada, mientras que la segunda función puede denominarse núcleo de convolución. La salida puede denominarse mapa de características. Por ejemplo, la entrada a 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 están adaptados por el proceso de entrenamiento para la red neuronal.

Las redes neuronales recurrentes (RNN) son una familia de redes de realimentación prospectiva que incluyen conexiones de realimentación entre capas. Las RNN posibilitan el modelado de datos secuenciales compartiendo datos de parámetros a través de diferentes partes de la red neuronal. La arquitectura para una RNN incluye ciclos. Los ciclos representan la influencia de un valor presente 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 realimentación para procesar la entrada posterior en una secuencia. Esta característica hace que las RNN sean particularmente útiles para el procesamiento de idioma debido a la naturaleza variable en la que pueden componerse los datos de idioma.

Las figuras descritas a continuación presentan ejemplos de redes de realimentación prospectiva, CNN y RNN, y describen un proceso general para entrenar e implementar respectivamente cada uno de esos tipos de redes. Se entenderá que estas descripciones son ilustrativas y no limitantes en cuanto a cualquier realización específica descrita

en el presente documento y los conceptos ilustrados pueden aplicarse, de manera general, a redes neuronales profundas y técnicas de aprendizaje automático en general.

Las redes neuronales a modo de ejemplo descritas anteriormente se pueden utilizar para realizar aprendizaje profundo. El aprendizaje profundo es aprendizaje automático que utiliza redes neuronales profundas. Las redes neuronales profundas usadas en aprendizaje profundo son redes neuronales artificiales compuestas de múltiples capas ocultas, a diferencia de redes neuronales poco profundas que incluyen únicamente una sola capa oculta. El entrenamiento de redes neuronales más profundas es, en general, más intensivo desde el punto de vista computacional. Sin embargo, las capas ocultas adicionales de la red permiten el reconocimiento de patrones de varios pasos 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 generalmente incluyen una red de extremo frontal para realizar el reconocimiento de características acoplada a una red de extremo posterior 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. Un aprendizaje profundo posibilita que se realice un aprendizaje automático sin requerir que se realice una ingeniería de características artesanal para el modelo. En cambio, las redes neuronales profundas pueden aprender características en función de la estructura estadística o la correlación dentro de los datos de entrada. Las características aprendidas se pueden proporcionar a un modelo matemático que puede mapear características detectadas a una salida. El modelo matemático utilizado por la red generalmente está especializado para la tarea específica que se va a realizar, y se utilizarán diferentes modelos para realizar diferentes tareas.

Una vez que se ha estructurado la red neuronal, puede aplicarse un modelo de aprendizaje a la red para entrenar la red para realizar 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 método común usado para entrenar redes neuronales. Se presenta un vector de entrada a la red para su procesamiento. La salida de la red se compara con la salida deseada usando 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 luego hacia atrás hasta que cada neurona tiene un valor de error asociado que representa aproximadamente su contribución a la salida original. La red puede aprender, a continuación, de esos errores usando un algoritmo, tal como el algoritmo de descenso de gradiente estocástico, para actualizar los pesos de la red neuronal.

Las Figuras 13A-B ilustran una red neuronal convolucional ilustrativa. La Figura 13A ilustra diversas capas dentro de una CNN. Como se muestra en la Figura 13A, una CNN ilustrativa usada para modelar el procesamiento de imagen puede recibir la entrada 1302 que describe los componentes rojo, verde y azul (RGB) de una imagen de entrada. La entrada 1302 puede procesarse por múltiples capas convolucionales (por ejemplo, la capa convolucional 1304, la capa convolucional 1306). La salida de las múltiples capas convolucionales puede procesarse opcionalmente por un conjunto de capas completamente conectadas 1308. Las neuronas en una capa completamente conectada tienen conexiones completas con todas las activaciones en la capa anterior, tal como se describió anteriormente para una red de realimentación prospectiva. La salida de las capas completamente conectadas 1308 puede usarse para generar un resultado de salida a partir de la red. Las activaciones dentro de las capas completamente conectadas 1308 se pueden calcular utilizando la multiplicación de matrices en lugar de la convolución. No todas las implementaciones de CNN hacen uso de capas completamente conectadas DPLA08. Por ejemplo, en algunas implementaciones, la capa convolucional 1306 puede generar la salida de la CNN.

Las capas convolucionales están escasamente conectadas, lo que difiere de la configuración de red neuronal tradicional que se encuentra en las capas completamente conectadas 1308. 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 están escasamente conectadas porque la salida de la convolución de un campo es la entrada (en lugar del valor de estado respectivo de cada uno de los nodos en el campo) a los nodos de la capa posterior, tal como se ilustra. Los núcleos asociados con las capas convolucionales realizan operaciones de convolución, cuya salida se envía a la siguiente capa. La reducción de la dimensionalidad realizada dentro de las capas convolucionales es un aspecto que posibilita que la CNN escale para procesar imágenes grandes.

La **Figura 13B** ilustra etapas de cálculo a modo de ejemplo dentro de una capa convolucional de una CNN. La entrada a una capa convolucional 1312 de una CNN puede procesarse en tres etapas de una capa convolucional 1314. Las tres etapas pueden incluir una etapa de convolución 1316, una etapa de detección 1318 y una etapa de agrupamiento 1320. La capa convolucional 1314 puede entonces enviar datos a una capa convolucional sucesiva. La capa convolucional final de la red puede generar datos de mapeo de características de salida o proporcionar entrada a una capa completamente conectada, por ejemplo, para generar un valor de clasificación para la entrada a la CNN.

En la etapa de convolución 1316 se realizan varias convoluciones en paralelo para producir un conjunto de activaciones lineales. La etapa de convolución 1316 puede incluir una transformación afín, que es cualquier transformación que se pueda especificar como una transformación lineal más una traslación. Las transformaciones afines incluyen rotaciones, traslaciones, escalado y combinaciones de estas transformaciones. La etapa de

convolución calcula la salida de funciones (por ejemplo, neuronas) que están conectadas a regiones específicas en la entrada, que se pueden determinar como la región local asociada con 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 etapa de convolución 1316 define un conjunto de activaciones lineales que se procesan por etapas sucesivas de la capa convolucional 1314.

Las activaciones lineales pueden ser procesadas por una etapa de detección 1318. En la etapa de detección 1318, cada activación lineal es procesada por 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. Pueden usarse 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 $f(x) = \text{máx}(0, x)$, de modo que la activación tiene un umbral en cero.

La etapa de agrupamiento 1320 usa una función de agrupación que sustituye la salida de la capa convolucional 1306 con un sumario estadístico de las salidas cercanas. La función de agrupamiento puede usarse para introducir la invarianza de traslación en la red neuronal, de manera que traslaciones pequeñas en la entrada no cambian las salidas agrupadas. La invarianza a la traslación local puede ser útil en situaciones donde la presencia de una característica en los datos de entrada es más importante que la ubicación precisa de la característica. Pueden usarse diversos tipos de funciones de agrupamiento durante la etapa de agrupamiento 1320, que incluye agrupamiento máximo, agrupamiento promedio y agrupamiento de norma l2. Adicionalmente, algunas implementaciones de CNN no incluyen una fase de agrupamiento. En su lugar, tales implementaciones sustituyen una etapa de convolución adicional que tiene un paso mayor en relación con las etapas de convolución anteriores.

La salida de la capa convolucional 1314 puede ser procesada luego por la siguiente capa 1322. La siguiente capa 1322 puede ser una capa convolucional adicional o una de las capas completamente conectadas 1308. Por ejemplo, la primera capa convolucional 1304 de la **Figura 13A** puede enviarse a la segunda capa convolucional 1306, mientras que la segunda capa convolucional puede enviarse a una primera capa de las capas completamente conectadas 1308.

La **Figura 14** ilustra una red neuronal recurrente ilustrativa 1400. En una red neuronal recurrente (RNN), el estado anterior de la red influye en la salida del estado actual de la red. Las RNN pueden crearse en una diversidad de maneras usando una diversidad de funciones. El uso de las RNN generalmente gira en torno al uso de modelos matemáticos para predecir el futuro basándose en una secuencia previa de entradas. Por ejemplo, puede usarse una RNN para realizar modelado de idioma estadístico para predecir una palabra próxima dada en una secuencia de palabras anterior. La RNN 1400 ilustrada puede describirse como que tiene una capa de entrada 1402 que recibe un vector de entrada, capas ocultas 1404 para implementar una función recurrente, un mecanismo de realimentación 1405 para habilitar una 'memoria' de estados anteriores y una capa de salida 1406 para generar un resultado. La RNN 1400 opera basándose en pasos de tiempo. El estado de la RNN en un paso de tiempo determinado se ve influenciado en función del paso de tiempo anterior a través del mecanismo de realimentación 1405. Para una etapa de tiempo dada, se define el estado de las capas ocultas 1404 por el estado anterior y la entrada en la etapa de tiempo actual. Puede procesarse una entrada inicial (x_1) en una primera etapa de tiempo por la capa oculta 1404. Puede procesarse una segunda entrada (x_2) por la capa oculta 1404 usando 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 una no linealidad, tal como la función tangente hiperbólica (Tanh) o una variante de la función rectificadora $f(x) = \text{máx}(0,x)$. Sin embargo, la función matemática específica utilizada en las capas ocultas 1404 puede variar dependiendo de los detalles de implementación específicos de la RNN 1400.

Además de las redes CNN y RNN básicas descritas, pueden habilitarse variaciones en esas redes. Una variante de RNN ilustrativa es la RNN de memoria a corto plazo larga (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 de creencia profunda convolucional, que tiene una estructura similar a una CNN y se entrena de una manera similar a una red de creencia profunda. Una red de creencias profundas (DBN) es una red neuronal generativa que se compone de múltiples capas de variables estocásticas (aleatorias). Las DBN pueden entrenarse capa a capa usando aprendizaje no supervisado voraz. Los pesos aprendidos de la DBN se pueden utilizar para proporcionar redes neuronales preentrenadas determinando un conjunto inicial óptimo de pesos para la red neuronal.

La **Figura 15** ilustra el entrenamiento y despliegue de una red neuronal profunda. Una vez que se ha estructurado una red determinada para una tarea, la red neuronal se entrena utilizando un conjunto de datos de entrenamiento 1502. Se han desarrollado diversas estructuras de entrenamiento 1504 para posibilitar la aceleración de hardware del proceso de entrenamiento. Por ejemplo, la estructura de aprendizaje automático 1004 de la Figura 10 puede configurarse como una estructura de entrenamiento 1004. La estructura de entrenamiento 1004 puede conectarse a una red neuronal no entrenada 1506 y permitir que la red neuronal no entrenada se entrene utilizando los recursos de procesamiento paralelo descritos en este documento para generar una red neuronal entrenada 1508.

Para iniciar el proceso de entrenamiento, los pesos iniciales pueden elegirse aleatoriamente o mediante un entrenamiento previo utilizando una red de creencias profundas. El ciclo de entrenamiento se puede realizar de manera supervisada o no supervisada.

El aprendizaje supervisado es un método de aprendizaje en el que se realiza un entrenamiento como una operación mediada, tal como cuando el conjunto de datos de entrenamiento 1502 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 contra un conjunto de salidas esperadas o deseadas. Los errores se propagan hacia atrás a través del sistema. La estructura de entrenamiento 1504 puede ajustarse para ajustar los pesos que controlan la red neuronal no entrenada 1506. La estructura de entrenamiento 1504 puede proporcionar herramientas para monitorizar qué tan bien converge la red neuronal no entrenada 1506 hacia un modelo adecuado para generar respuestas correctas en función de los datos de entrada conocidos. El proceso de entrenamiento tiene lugar repetidamente a medida que se ajustan los pesos de la red para perfeccionar la salida generada por la red neuronal. El proceso de entrenamiento puede continuar hasta que la red neuronal alcanza una precisión estadísticamente deseada asociada con una red neuronal entrenada 1508. La red neuronal entrenada 1508 se puede implementar para implementar cualquier número de operaciones de aprendizaje automático.

El aprendizaje no supervisado es un método de aprendizaje en el que la red intenta entrenarse a sí misma usando datos no etiquetados. Por lo tanto, para el aprendizaje no supervisado, el conjunto de datos de entrenamiento 1502 incluirá datos de entrada sin ningún dato de salida asociado. La red neuronal no entrenada 1506 puede aprender agrupamientos dentro de la entrada no etiquetada y puede determinar cómo las entradas individuales están relacionadas 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 1507 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 en un conjunto de datos de entrada que se desvían de los patrones normales de los datos.

También pueden emplearse variaciones al entrenamiento supervisado y no supervisado. El aprendizaje semisupervisado es una técnica en la que el conjunto de datos de entrenamiento 1502 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 1508 se adapte a los nuevos datos 1512 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 en términos computacionales para un solo nodo de cómputo. En lugar de usar un único nodo de cálculo, puede usarse una red distribuida de nodos computacionales para acelerar el proceso de entrenamiento.

La **Figura 16** es un diagrama de bloques que ilustra el aprendizaje distribuido. El aprendizaje distribuido es un modelo de entrenamiento que usa múltiples nodos informáticos distribuidos para realizar un entrenamiento supervisado o no supervisado de una red neuronal. Cada uno de los nodos computacionales distribuidos puede incluir uno o más procesadores de anfitrión 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 1100 como en la Figura 1100. Como se ha ilustrado, un aprendizaje distribuido puede realizarse con el paralelismo de modelo 1602, el paralelismo de datos 1604 o una combinación del paralelismo de modelo y de datos 1604.

En el paralelismo de modelos 1602, diferentes nodos computacionales en un sistema distribuido pueden realizar cálculos de entrenamiento para diferentes partes de una sola red. Por ejemplo, cada capa de una red neuronal puede entrenarse 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 con diferentes capas de la red neuronal posibilita el entrenamiento de redes neuronales muy grandes en las que los pesos para todas las capas no se ajustarían en la memoria de un único nodo computacional. En algunos casos, el paralelismo de modelo puede ser particularmente útil al realizar entrenamiento no supervisado de redes neuronales grandes.

En el paralelismo de datos 1604, los diferentes nodos de la red distribuida tienen una instancia completa del modelo y cada nodo recibe una porción diferente de los datos. Luego, se combinan los resultados de los diferentes nodos. Aunque son posibles diferentes enfoques al paralelismo de datos, los enfoques de entrenamiento paralelo de datos requieren, todos ellos, una técnica de combinación de resultados y de sincronización de los parámetros de modelo entre cada nodo. Los enfoques ilustrativos para la combinación de datos incluyen promediado de parámetros y paralelismo de datos basado en actualizaciones. El promedio de parámetros entrena cada nodo en un subconjunto de los datos de entrenamiento y establece los parámetros globales (por ejemplo, pesos, desvíos) en el promedio de los parámetros de cada nodo. El promediado de parámetros usa un servidor de parámetros central que mantiene los datos de parámetro. El paralelismo de datos basado en actualizaciones es similar al promediado de parámetros excepto en que, en lugar de transferir parámetros desde los nodos al servidor de parámetros, se transfieren las actualizaciones al modelo. Además, el paralelismo de datos basado en actualizaciones se puede realizar de manera descentralizada, donde las actualizaciones se comprimen y se transfieren entre nodos.

El paralelismo de modelo y de datos combinado 1606 puede implementarse, por ejemplo, en un sistema distribuido en el que cada nodo computacional incluye múltiples GPU. Cada nodo puede tener una instancia completa del modelo con GPU separadas dentro de cada nodo que se usan para entrenar diferentes porciones del modelo.

El entrenamiento distribuido ha aumentado la sobrecarga en relación con el entrenamiento en una sola máquina. Sin embargo, los procesadores paralelos y las GPGPU descritas en este documento pueden implementar varias técnicas para reducir la sobrecarga del entrenamiento distribuido, incluidas técnicas para permitir la transferencia de datos de GPU a GPU de alto ancho de banda y la sincronización de datos remota acelerada.

Aplicaciones de aprendizaje automático a modo de ejemplo

El aprendizaje automático se puede aplicar para resolver una variedad de problemas tecnológicos, incluidos, entre otros, la visión informática, la conducción y navegación autónomas, el reconocimiento de voz y el procesamiento del lenguaje. La visión informática ha sido tradicionalmente una de las áreas de investigación más activas para aplicaciones de aprendizaje automático. Las aplicaciones de la visión informática van desde la reproducción de las capacidades visuales humanas, tal como el reconocimiento de rostros, hasta la creación de nuevas categorías de capacidades visuales. Por ejemplo, las aplicaciones de visión informática pueden configurarse para reconocer ondas de sonido de las vibraciones inducidas en los objetos visibles en un vídeo. El aprendizaje automático acelerado por procesadores paralelos permite que las aplicaciones de visión informática se entrenen utilizando un conjunto de datos de entrenamiento significativamente más grande que el que era posible anteriormente y permite que los sistemas de inferencia se implementen utilizando procesadores paralelos de bajo consumo.

El aprendizaje automático acelerado por procesador paralelo tiene aplicaciones de conducción autónoma que incluyen reconocimiento de señales de carretera y de carril, evitación de obstáculos, navegación y control de 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 este documento pueden permitir un entrenamiento rápido de las redes neuronales cada vez más complejas que se utilizan para soluciones de conducción autónoma y permiten la implementación de procesadores de inferencia de bajo consumo en una plataforma móvil adecuada para la integración en vehículos autónomos.

Las redes neuronales profundas aceleradas por procesadores paralelos 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 redes neuronales profundas ha permitido la sustitución de los modelos ocultos de Markov (HMM) y los modelos de mezcla gaussiana (GMM) utilizados anteriormente para ASR.

El aprendizaje automático acelerado por procesador paralelo puede usarse también para acelerar el procesamiento de lenguaje natural. Los procedimientos de aprendizaje automático pueden hacer uso de algoritmos de inferencia estadística para producir modelos que sean robustos a entradas erróneas o desconocidas. Las aplicaciones de procesador de lenguaje natural ilustrativas incluyen la traducción mecánica automática entre idiomas humanos.

Las plataformas de procesamiento paralelo utilizadas para el aprendizaje automático se pueden dividir en plataformas de entrenamiento y plataformas de implementación. Las plataformas de entrenamiento son generalmente altamente paralelas e incluyen optimizaciones para acelerar el entrenamiento de nodo único multi-GPU y el entrenamiento multi-nodo, multi-GPU. Los procesadores paralelos a modo de ejemplo adecuados para entrenamiento incluyen la unidad de procesamiento de gráficos de propósito general altamente paralela 1100 de la Figura 1100 y el sistema informático multi-GPU 1200 de la Figura 1200. Por el contrario, las plataformas de aprendizaje automático desplegadas incluyen, en general, procesadores paralelos de potencia inferior adecuados para su uso en productos tales como cámaras, robots autónomos y vehículos autónomos.

La **Figura 17** ilustra un sistema de inferencia ilustrativo en un chip (SOC) 1700 adecuado para realizar inferencia usando un modelo entrenado. El SOC 1700 puede integrar componentes de procesamiento que incluyen un procesador de medios 1702, un procesador de visión 1704, una GPGPU 1706 y un procesador de múltiples núcleos 1708. El SOC 1700 puede incluir, además, una memoria en chip 1705 que puede habilitar un grupo de datos en chip compartido al que puede acceder cada uno de los componentes de procesamiento. Los componentes de procesamiento pueden optimizarse para una operación de baja potencia para posibilitar el despliegue en una diversidad de plataformas de aprendizaje automático, incluyendo vehículos autónomos y robots autónomos. Por ejemplo, una implementación del SOC 1700 puede usarse como una porción del sistema de control principal para un vehículo autónomo. Cuando el SOC 1700 está configurado para su uso en vehículos autónomos, el SOC está diseñado y configurado para cumplir con los estándares de seguridad funcional pertinentes de la jurisdicción de implementación.

Durante el funcionamiento, el procesador de medios 1702 y el procesador de visión 1704 pueden trabajar en conjunto para acelerar las operaciones de visión por ordenador. El procesador de medios 1702 puede posibilitar la decodificación de latencia baja de múltiples flujos de vídeo de alta resolución (por ejemplo, 4K, 8K). Los flujos de vídeo decodificados pueden escribirse en una memoria intermedia en la memoria en chip 1705. El procesador de visión 1704

puede luego analizar el vídeo decodificado y realizar operaciones de procesamiento preliminares en los fotogramas del vídeo decodificado en preparación para procesar los fotogramas utilizando un modelo de reconocimiento de imágenes entrenado. Por ejemplo, el procesador de visión 1704 puede acelerar las operaciones convolucionales para una CNN que se usa para realizar el reconocimiento de imagen en los datos de vídeo de alta resolución, mientras se realizan cálculos de modelo de extremo trasero por la GPGPU 1706.

El procesador de múltiples núcleos 1708 puede incluir lógica de control para ayudar con la secuenciación y sincronización de transferencias de datos y operaciones de memoria compartida realizadas por el procesador de medios 1702 y el procesador de visión 1704. El procesador de múltiples núcleos 1708 también puede funcionar como un procesador de aplicaciones para ejecutar aplicaciones de software que pueden hacer uso de la capacidad de cálculo de inferencia de la GPGPU 1706. Por ejemplo, puede implementarse al menos una porción de la lógica de navegación y de conducción en software que se ejecuta en el procesador de múltiples núcleos 1708. Dicho software puede emitir directamente cargas de trabajo computacionales a la GPGPU 1706 o las cargas de trabajo computacionales pueden emitirse al procesador de múltiples núcleos 1708, que puede descargar al menos una parte de esas operaciones a la GPGPU 1706.

La GPGPU 1706 puede incluir agrupaciones de cálculo, tal como una configuración de baja potencia de las agrupaciones de cálculo 1106A-1106H dentro de la unidad de procesamiento de gráficos de propósito general altamente paralela 1100. Las agrupaciones de cálculo dentro de la GPGPU 1706 pueden admitir instrucciones que están específicamente optimizadas para realizar cálculos de inferencia en una red neuronal entrenada. Por ejemplo, la GPGPU 1706 puede admitir instrucciones para realizar cálculos de precisión baja, tales como operaciones vectoriales de números enteros de 8 bits y de 4 bits.

Vista general del sistema II

La **Figura 18** es un diagrama de bloques de un sistema de procesamiento 1800, de acuerdo con una realización. En diversas realizaciones, el sistema 1800 incluye uno o más procesadores 1802 y uno o más procesadores de gráficos 1808, y puede ser un sistema de escritorio de un solo procesador, un sistema de estación de trabajo multiprocesador o un sistema de servidor que tiene una gran cantidad de procesadores 1802 o núcleos de procesador 1807. En una realización, el sistema 1800 es una plataforma de procesamiento incorporada dentro de un circuito integrado de sistema en chip α (SoC) para su uso en dispositivos móviles, portátiles o integrados.

Una realización del sistema 1800 puede incluir, o estar incorporada dentro de una plataforma de juego basada en servidor, una consola de juegos, que incluye una consola de juegos y multimedia, una consola de juegos móvil, una consola de juegos portátil o una consola de juegos en línea. En algunas realizaciones, el sistema 1800 es un teléfono móvil, un teléfono inteligente, un dispositivo informático de tipo tableta o un dispositivo de Internet móvil. El sistema de procesamiento de datos 1800 también puede incluir, acoplarse con, 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 realizaciones, el sistema de procesamiento de datos 1800 es un dispositivo de televisión o decodificador que tiene uno o más procesadores 1802 y una interfaz gráfica generada por uno o más procesadores de gráficos 1808.

En algunas realizaciones, el uno o varios procesadores 1802 incluyen, cada uno, uno o varios núcleos de procesador 1807 para procesar instrucciones que, cuando se ejecutan, realizan operaciones para el software del sistema y del usuario. En algunas realizaciones, cada uno del uno o varios núcleos de procesador 1807 está configurado para procesar un conjunto de instrucciones específico 1809. En algunas realizaciones, el conjunto de instrucciones 1809 puede facilitar el cálculo de conjunto de instrucciones complejo (CISC), el cálculo de conjunto de instrucciones reducido (RISC) o el cálculo mediante una palabra de instrucción muy larga (VLIW). Múltiples núcleos de procesador 1807 pueden procesar cada uno un conjunto de instrucciones 1809 diferente, que puede incluir instrucciones para facilitar la emulación de otros conjuntos de instrucciones. El núcleo de procesador 1807 también puede incluir otros dispositivos de procesamiento, tal como un procesador de señal digital (DSP).

En algunas realizaciones, el procesador 1802 incluye una memoria caché 1804. Dependiendo de la arquitectura, el procesador 1802 puede tener una única memoria caché interna o múltiples niveles de memoria caché interna. En algunas realizaciones, la memoria caché se comparte entre varios componentes del procesador 1802. En algunas realizaciones, el procesador 1802 también utiliza una memoria caché externa (por ejemplo, una memoria caché de nivel 3 (L3) o una memoria caché de último nivel (LLC)) (no se muestra), que puede compartirse entre los núcleos de procesador 1807 utilizando técnicas de coherencia de caché conocidas. Un archivo de registros 1806 se incluye adicionalmente en el procesador 1802 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 instrucciones). Algunos registros pueden ser registros de propósito general, mientras que otros registros pueden ser específicos para el diseño del procesador 1802.

En algunas realizaciones, el procesador 1802 está acoplado a un bus de procesador 1810 para transmitir señales de comunicación tales como señales de dirección, de datos o de control entre el procesador 1802 y otros componentes del sistema 1800. En una realización, el sistema 1800 utiliza una arquitectura de sistema de "concentrador" a modo

de ejemplo, que incluye un centro de controlador de memoria 1816 y un centro de controlador de entrada y salida (E/S) 1830. Un centro de controlador de memoria 1816 facilita la comunicación entre un dispositivo de memoria y otros componentes del sistema 1800, mientras que un concentrador del controlador de E/S (ICH) 1830 proporciona conexiones a dispositivos de E/S a través de un bus de E/S local. En una realización, la lógica del concentrador del controlador de memoria 1816 está integrada dentro del procesador.

El dispositivo de memoria 1820 puede ser un dispositivo de memoria de acceso aleatorio dinámico (DRAM), un dispositivo de memoria de acceso aleatorio estático (SRAM), un dispositivo de memoria flash, un dispositivo de memoria de cambio de fase o algún otro dispositivo de memoria que tenga un rendimiento adecuado para servir como memoria de proceso. En una realización, el dispositivo de memoria 1820 puede funcionar como memoria del sistema para el sistema 1800, para almacenar datos 1822 e instrucciones 1821 para su uso cuando el uno o más procesadores 1802 ejecutan una aplicación o un proceso. El concentrador del controlador de memoria 1816 también se acopla con un procesador de gráficos externo opcional 1812, que puede comunicarse con el uno o más procesadores de gráficos 1808 en los procesadores 1802 para realizar operaciones gráficas y multimedia.

En algunas realizaciones, el ICH 1830 permite que dispositivos periféricos se conecten al dispositivo de memoria 1820 y al procesador 1802 por medio de un bus de E/S de alta velocidad. Los dispositivos periféricos de E/S incluyen, pero sin limitarse a, un controlador de audio 1846, una interfaz de firmware 1828, un transceptor inalámbrico 1826 (por ejemplo, Wi-Fi, Bluetooth), un dispositivo de almacenamiento de datos 1824 (por ejemplo, unidad de disco duro, memoria flash, etc.) y un controlador de E/S heredado 1840 para acoplar dispositivos heredados (por ejemplo, un sistema personal 2 (PS/2)) al sistema. Uno o más controladores de bus serie universal (USB) 1842 conectan dispositivos de entrada, tales como combinaciones de teclado y ratón 1844. Un controlador de red 1834 también puede acoplarse al ICH 1830. En algunas realizaciones, un controlador de red de alto rendimiento (no mostrado) se acopla con el bus de procesador 1810. Se apreciará que el sistema 1800 mostrado es ilustrativo y no limitante, ya que pueden usarse otros tipos de sistemas de procesamiento de datos que están configurados de manera diferente. Por ejemplo, el concentrador del controlador de E/S 1830 puede estar integrado dentro del uno o más procesadores 1802, o el concentrador del controlador de memoria 1816 y el concentrador del controlador de E/S 1830 pueden estar integrados en un procesador de gráficos externo discreto, tal como el procesador de gráficos externo 1812.

La **Figura 19** es un diagrama de bloques de una realización de un procesador 1900 que tiene uno o más núcleos de procesador 1902A-1902N, un controlador de memoria integrado 1914 y un procesador de gráficos integrado 1908. Los elementos de la **Figura 19** que tienen los mismos números de referencia (o nombres) que los elementos de cualquier otra figura del presente documento pueden operar o funcionar de cualquier manera similar a la descrita en otra parte del presente documento, pero no se limitan a ello. El procesador 1900 puede incluir núcleos adicionales hasta e incluyendo el núcleo adicional 1902N representado por los recuadros de línea discontinua. Cada uno de los núcleos de procesador 1902A-1902N incluye una o más unidades de caché internas 1904A-1904N. En algunas realizaciones, cada núcleo de procesador también tiene acceso a una o más unidades de caché compartidas 1906.

Las unidades de caché internas 1904A-1904N y las unidades de caché compartidas 1906 representan una jerarquía de memoria caché dentro del procesador 1900. La jerarquía de memoria caché puede incluir al menos un nivel de memoria caché de instrucciones y 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 caché, donde el nivel más alto de memoria caché antes de la memoria externa se clasifica como LLC. En algunas realizaciones, la lógica de coherencia de caché mantiene la coherencia entre las diversas unidades de memoria caché 1906 y 1904A-1904N.

En algunas realizaciones, el procesador 1900 también puede incluir un conjunto de una o más unidades de controlador de bus 1916 y un núcleo de agente de sistema 1910. La una o varias unidades de controlador de bus 1916 gestionan un conjunto de buses periféricos, tales como uno o varios buses de interconexión de componentes periféricos (por ejemplo, PCI, PCI Express). El núcleo de agente de sistema 1910 proporciona funcionalidad de gestión para los diversos componentes de procesador. En algunas realizaciones, el núcleo de agente de sistema 1910 incluye uno o más controladores de memoria integrados 1914 para gestionar el acceso a varios dispositivos de memoria externa (no se muestra).

En algunas realizaciones, uno o más de los núcleos de procesador 1902A-1902N incluyen soporte para múltiples hilos simultáneos. En dicha realización, el núcleo de agente de sistema 1910 incluye componentes para coordinar y operar los núcleos 1902A-1902N durante el procesamiento de múltiples hilos. El núcleo de agente de sistema 1910 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 1902A-1902N y el procesador de gráficos 1908.

En algunas realizaciones, el procesador 1900 incluye adicionalmente el procesador de gráficos 1908 para ejecutar operaciones de procesamiento de gráficos. En algunas realizaciones, el procesador de gráficos 1908 se acopla con el conjunto de unidades de memoria caché compartidas 1906 y el núcleo de agente de sistema 1910, incluidos el uno o más controladores de memoria integrados 1914. En algunas realizaciones, un controlador de visualización 1911 está acoplado con el procesador de gráficos 1908 para impulsar la salida del procesador de gráficos a una o más pantallas acopladas. En algunas realizaciones, el controlador de visualización 1911 puede ser un módulo separado acoplado

con el procesador de gráficos a través de al menos una interconexión, o puede estar integrado dentro del procesador de gráficos 1908 o el núcleo de agente de sistema 1910.

En algunas realizaciones, se usa una unidad de interconexión basada en anillo 1912 para acoplar los componentes internos del procesador 1900. Sin embargo, se puede utilizar una unidad de interconexión alternativa, tal como una interconexión punto a punto, una interconexión conmutada u otras técnicas, incluidas técnicas bien conocidas en la técnica. En algunas realizaciones, el procesador de gráficos 1908 se acopla con la interconexión en anillo 1912 a través de un enlace de E/S 1913.

El enlace de E/S 1913 ilustrativo representa al menos una de múltiples variedades de interconexiones de E/S, incluyendo una interconexión de E/S en paquete que facilita la comunicación entre diversos componentes de procesador y un módulo de memoria integrado de alto rendimiento 1918, tal como un módulo de eDRAM. En algunas realizaciones, cada uno de los núcleos de procesador 1902-1902N y el procesador de gráficos 1908 utilizan módulos de memoria integrados 1918 como una memoria caché de último nivel compartida.

En algunas realizaciones, los núcleos de procesador 1902A-1902N son núcleos homogéneos que ejecutan la misma arquitectura de conjunto de instrucciones. En otra realización, los núcleos de procesador 1902A-1902N son heterogéneos en términos de arquitectura de conjunto de instrucciones (ISA), donde uno o más de los núcleos de procesador 1902A-N 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 realización, los núcleos de procesador 1902A-1902N son heterogéneos en términos de microarquitectura, donde uno o más núcleos que tienen un consumo de energía relativamente más alto se acoplan con uno o más núcleos de energía que tienen un consumo de energía más bajo. Además, el procesador 1900 se puede implementar en uno o más chips o como un circuito integrado SoC que tiene los componentes ilustrados, además de otros componentes.

La **Figura 20** es un diagrama de bloques de un procesador de gráficos 2000, 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 realizaciones, el procesador de gráficos se comunica a través de una interfaz de E/S mapeada en memoria a registros en el procesador de gráficos y con comandos colocados en la memoria del procesador. En algunas realizaciones, el procesador de gráficos 2000 incluye una interfaz de memoria 2014 para acceder a memoria. La interfaz de memoria 2014 puede ser una interfaz con la memoria local, una o más memorias caché internas, una o más memorias caché externas compartidas y/o con la memoria del sistema.

En algunas realizaciones, el procesador de gráficos 2000 también incluye un controlador de visualización 2002 para dirigir los datos de salida de visualización a un dispositivo de visualización 2020. El controlador de visualización 2002 incluye hardware para uno o más planos de superposición para la visualización y composición de múltiples capas de elementos de interfaz de usuario o de vídeo. En algunas realizaciones, el procesador de gráficos 2000 incluye un motor de códec de vídeo 2006 para codificar, decodificar o transcodificar medios hacia, desde o entre uno o más formatos de codificación de medios, incluidos, entre otros, los formatos del Grupo de expertos en imágenes en movimiento (MPEG) tales como MPEG-2, los formatos de codificación de vídeo avanzada (AVC) tales como H.264/MPEG-4 AVC, así como los formatos de la Sociedad de ingenieros de imágenes en movimiento y televisión (SMPTE) 421M/VC-1 y formatos del Grupo conjunto de expertos en fotografía (JPEG) tales como JPEG y Motion JPEG (MJPEG).

En algunas realizaciones, el procesador gráfico 2000 incluye un motor de transferencia de imágenes en bloques (BLIT) 2004 para realizar operaciones de rasterización bidimensionales (2D) incluyendo, por ejemplo, transferencias de bloques con límites de bits. Sin embargo, en una realización, las operaciones gráficas en 2D se realizan usando uno o más componentes del motor de procesamiento de gráficos (GPE) 2010. En algunas realizaciones, el motor de procesamiento de gráficos 2010 es un motor de cálculo para realizar operaciones gráficas, incluidas operaciones gráficas tridimensionales (3D) y operaciones multimedia.

En algunas realizaciones, el GPE 2010 incluye una canalización 3D 2012 para realizar operaciones 3D, tal como representar imágenes y escenas tridimensionales usando funciones de procesamiento que actúan sobre formas de primitivas 3D (p. ej., rectángulo, triángulo, etc.). La canalización 3D 2012 incluye elementos de función fija y programables que realizan diversas tareas dentro del elemento y/o generan hilos de ejecución en un subsistema 3D/de medios 2015. Aunque se puede usar la canalización 3D 2012 para realizar operaciones de medios, una realización del GPE 2010 también incluye una canalización de medios 2016 que se usa específicamente para realizar operaciones de medios, tales como postprocesamiento de vídeo y mejora de imagen.

En algunas realizaciones, la canalización de medios 2016 incluye funciones fijas o unidades lógicas programables para realizar una o varias operaciones de medios especializadas, tales como aceleración de decodificación de vídeo, desentrelazado de vídeo y aceleración de codificación de vídeo en lugar de, o en nombre del motor de códec de vídeo 2006. En algunas realizaciones, la canalización de medios 2016 incluye adicionalmente una unidad de generación de hilos para generar hilos para su ejecución en el subsistema 3D/de medios 2015. Los hilos generados realizan cálculos para las operaciones de medios en una o varias unidades de ejecución de gráficos incluidas en el subsistema 3D/de medios 2015.

En algunas realizaciones, el subsistema 3D/de medios 2015 incluye lógica para ejecutar hilos generados por la canalización 3D 2012 y la canalización de medios 2016. En una realización, las canalizaciones envían solicitudes de ejecución de hilos al subsistema 3D/de medios 2015, que incluye lógica de despacho de hilos para arbitrar y despachar las diversas solicitudes a los recursos de ejecución de hilos disponibles. Los recursos de ejecución incluyen una matriz de unidades de ejecución de gráficos para procesar los hilos 3D y multimedia. En algunas realizaciones, el subsistema 3D/de medios 2015 incluye una o más memorias caché internas para instrucciones de hilos y datos. En algunas realizaciones, el subsistema también incluye una memoria compartida, incluyendo registros y una memoria direccionable, para compartir datos entre hilos y para almacenar datos de salida.

Procesamiento 3D/de medios

La **Figura 21** es un diagrama de bloques de un motor de procesamiento de gráficos 2110 de un procesador de gráficos de acuerdo con algunas realizaciones. En una realización, el motor de procesamiento de gráficos (GPE) 2110 es una versión del GPE 2010 mostrado en la **Figura 20.** Elementos de la **Figura 21** que tienen los mismos números de referencia (o nombres) que los elementos de cualquier otra figura del presente documento pueden operar o funcionar de cualquier manera similar a la descrita en otra parte del presente documento, pero no se limitan a ello. Por ejemplo, se ilustra la canalización 3D 2012 y la canalización de medios 2016 de la **Figura 20**. La canalización de medios 2016 es opcional en algunas realizaciones del GPE 2110 y puede no incluirse de forma explícita en el GPE 2110. Por ejemplo, y en al menos una realización, un procesador de medios y/o imágenes independiente está acoplado al GPE 2110.

En algunas realizaciones, el GPE 2110 se acopla con o incluye un transmisor en continuo de comandos 2103, que proporciona un flujo de comandos a la canalización 3D 2012 y/o a la canalización de medios 2016. En algunas realizaciones, el transmisor en continuo de comandos 2103 está acoplado a memoria, que puede ser memoria de sistema, o una o más de memoria caché interna y memoria caché compartida. En algunas realizaciones, el transmisor en continuo de comandos 2103 recibe comandos de la memoria y envía los comandos a la canalización 3D 2012 y/o a la canalización de medios 2016. Los comandos son directivas extraídas de una memoria intermedia de anillo, que almacena comandos para la canalización 3D 2012 y la canalización de medios 2016. En una realización, la memoria intermedia de anillo puede incluir adicionalmente memorias intermedias de comandos por lotes que almacenan lotes de múltiples comandos. Los comandos para la canalización 3D 2012 también pueden incluir referencias a datos almacenados en la memoria, tales como por ejemplo, entre otros, datos de vértices y de geometría para la canalización 3D 2012 y/o datos de imagen y objetos de memoria para la canalización de medios 2016. La canalización 3D 2012 y la canalización de medios 2016 procesan los comandos y los datos realizando operaciones mediante la lógica dentro de las respectivas canalizaciones o despachando uno o más hilos de ejecución a una matriz de núcleos de gráficos 2114.

En diversas realizaciones, la canalización 3D 2012 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 despacho de hilos de ejecución a la matriz de núcleos de gráficos 2114. La matriz de núcleos de gráficos 2114 proporciona un bloque unificado de recursos de ejecución. La lógica de ejecución multipropósito (por ejemplo, unidades de ejecución) dentro de la matriz de núcleos de gráficos 2114 incluye soporte para varios lenguajes de sombreado de API 3D y puede ejecutar múltiples hilos de ejecución simultáneos asociados con múltiples sombreadores.

En algunas realizaciones, la matriz de núcleos de gráficos 2114 también incluye lógica de ejecución para realizar funciones multimedia, tales como procesamiento de vídeo y/o de imágenes. En una realización, las unidades de ejecución incluyen, además, una lógica de propósito general que es programable 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 conjunto con la lógica de propósito general dentro del núcleo o núcleos del procesador 1807 de la **Figura 18** o el núcleo 1902A-1902N como en la **Figura 19.**

Los datos de salida generados por hilos que se ejecutan en la matriz de núcleos de gráficos 2114 pueden emitir datos a memoria en una memoria intermedia de retorno unificada (URB) 2118. La URB 2118 puede almacenar datos para múltiples hilos. En algunas realizaciones, la URB 2118 se puede utilizar para enviar datos entre diferentes hilos que se ejecutan en la matriz de núcleos de gráficos 2114. En algunas realizaciones, la URB 2118 se puede utilizar, además, para la sincronización entre hilos en la matriz de núcleos de gráficos y la lógica de función fija dentro de la lógica de función compartida 2120.

En algunas realizaciones, la matriz de núcleos de gráficos 2114 es escalable, de modo que la matriz incluye una cantidad variable de núcleos de gráficos, cada uno con una cantidad variable de unidades de ejecución en función del nivel de potencia y rendimiento objetivos del GPE 2110. En una realización, los recursos de ejecución son escalables dinámicamente, de modo que los recursos de ejecución se pueden habilitar o deshabilitar según sea necesario.

La matriz de núcleos de gráficos 2114 se acopla con la lógica de función compartida 2120 que incluye múltiples recursos que se comparten entre los núcleos de gráficos en la matriz de núcleos de gráficos. Las funciones compartidas dentro de la lógica de función compartida 2120 son unidades lógicas de hardware que proporcionan funcionalidad suplementaria especializada a la matriz de núcleos de gráficos 2114. En diversas realizaciones, la lógica de función compartida 2120 incluye, entre otras cosas, la lógica del muestreador 2121, la lógica matemática 2122 y la lógica de comunicación entre hilos (ITC) 2123. Además, algunas realizaciones implementan una o más memorias caché 2125 dentro de la lógica de función compartida 2120. Una función compartida se implementa cuando la demanda de una función especializada dada es insuficiente para su inclusión en la matriz de núcleos de gráficos 2114. En su lugar, se implementa una única instanciación de dicha función especializada como una entidad independiente en la lógica de función compartida 2120 y se comparte entre los recursos de ejecución dentro de la matriz de núcleos de gráficos 2114. El conjunto preciso de funciones que se comparten entre la matriz de núcleos de gráficos 2114 y se incluyen dentro de la matriz de núcleos de gráficos 2114 varía entre las realizaciones.

La **Figura 22** es un diagrama de bloques de otra realización de un procesador de gráficos 2200. Elementos de la **Figura 22** que tienen los mismos números de referencia (o nombres) que los elementos de cualquier otra figura del presente documento pueden operar o funcionar de cualquier manera similar a la descrita en otra parte del presente documento, pero no se limitan a ello.

En algunas realizaciones, el procesador de gráficos 2200 incluye una interconexión en anillo 2202, un extremo frontal de canalización 2204, un motor de medios 2237 y núcleos de gráficos 2280A-2280N. En algunas realizaciones, la interconexión en anillo 2202 acopla el procesador de gráficos a otras unidades de procesamiento, que incluyen otros procesadores de gráficos o uno o más núcleos de procesadores de propósito general. En algunas realizaciones, el procesador de gráficos es uno de muchos procesadores integrados dentro de un sistema de procesamiento de múltiples núcleos.

En algunas realizaciones, el procesador de gráficos 2200 recibe lotes de comandos a través de la interconexión en anillo 2202. Los comandos entrantes son interpretados por un transmisor en continuo de comandos 2203 en el extremo frontal de la canalización 2204. En algunas realizaciones, el procesador de gráficos 2200 incluye lógica de ejecución escalable para realizar procesamiento de geometría 3D y procesamiento de medios a través del núcleo o núcleos de gráficos 2280A-2280N. Para los comandos de procesamiento de geometría 3D, el transmisor en continuo de comandos 2203 suministra comandos a la canalización de geometría 2236. Para al menos algunos comandos de procesamiento de medios, el transmisor en continuo de comandos 2203 suministra los comandos a un extremo frontal de vídeo 2234, que se acopla con un motor de medios 2237. En algunas realizaciones, el motor de medios 2237 incluye un motor de calidad de vídeo (VQE) 2230 para el postprocesamiento de vídeo e imágenes y un motor de codificación/decodificación multiformato (MFX) 2233 para proporcionar codificación y decodificación de datos de medios acelerada por hardware. En algunas realizaciones, la canalización de geometría 2236 y el motor de medios 2237 generan cada uno hilos de ejecución para los recursos de ejecución de hilos proporcionados por al menos un núcleo de gráficos 2280A.

En algunas realizaciones, el procesador de gráficos 2200 incluye recursos de ejecución de hilos escalables que presentan núcleos modulares 2280A-2280N (a veces denominados secciones de núcleo), cada uno con múltiples subnúcleos 2250A-2250N, 2260A-2260N (a veces denominados subsecciones de núcleo). En algunas realizaciones, el procesador de gráficos 2200 puede tener cualquier número de núcleos de gráficos 2280A a 2280N. En algunas realizaciones, el procesador de gráficos 2200 incluye un núcleo de gráficos 2280A que tiene al menos un primer subnúcleo 2250A y un segundo subnúcleo de núcleo 2260A. En otras realizaciones, el procesador de gráficos es un procesador de baja potencia con un único subnúcleo (por ejemplo, 2250A). En algunas realizaciones, el procesador de gráficos 2200 incluye múltiples núcleos de gráficos 2280A-2280N, incluyendo cada uno un conjunto de primeros subnúcleos 2250A-2250N y un conjunto de segundos subnúcleos 2260A-2260N. Cada subnúcleo del conjunto de primeros subnúcleos 2250A-2250N incluye al menos un primer conjunto de unidades de ejecución 2252A-2252N y muestreadores de medios/texturas 2254A-2254N. Cada subnúcleo en el conjunto de segundos subnúcleos 2260A-2260N incluye al menos un segundo conjunto de unidades de ejecución 2262A-2262N y muestreadores 2264A-2264N. En algunas realizaciones, cada subnúcleo 2250A-2250N, 2260A-2260N comparte un conjunto de recursos compartidos 2270A-2270N. En algunas realizaciones, los recursos compartidos incluyen memoria caché y lógica de operación de píxeles compartidas. También se pueden incluir otros recursos compartidos en las diversas realizaciones del procesador de gráficos.

Lógica de ejecución

La **Figura 23** ilustra la lógica de ejecución de hilos 2300 que incluye una matriz de elementos de procesamiento empleada en algunas realizaciones de un GPE. Elementos de la **Figura 23** que tienen los mismos números de referencia (o nombres) que los elementos de cualquier otra figura del presente documento pueden operar o funcionar de cualquier manera similar a la descrita en otra parte del presente documento, pero no se limitan a ello.

En algunas realizaciones, la lógica de ejecución de hilos 2300 incluye un sombreador de píxeles 2302, un despachador de hilos 2304, una memoria caché de instrucciones 2306, una matriz de unidades de ejecución escalables que incluye una pluralidad de unidades de ejecución 2308A-2308N, un muestreador 2310, una memoria caché de datos 2312 y

un puerto de datos 2314. En una realización, los componentes incluidos están interconectados a través de una estructura de interconexión que se vincula a cada uno de los componentes. En algunas realizaciones, la lógica de ejecución de hilos 2300 incluye una o más conexiones a la memoria, tal como la memoria del sistema o la memoria caché, a través de uno o más de la memoria caché de instrucciones 2306, el puerto de datos 2314, el muestreador 2310 y la matriz de unidades de ejecución 2308A-2308N. En algunas realizaciones, cada unidad de ejecución (p. ej., 2308A) es un procesador de vector individual que puede ejecutar múltiples hilos simultáneos y procesar múltiples elementos de datos en paralelo para cada hilo. En algunas realizaciones, la matriz de unidades de ejecución 2308A-2308N incluye cualquier número de unidades de ejecución individuales.

En algunas realizaciones, la matriz de unidades de ejecución 2308A-2308N se usa principalmente para ejecutar programas de "sombreador". En algunas realizaciones, las unidades de ejecución en la matriz 2308A-2308N ejecutan 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 (p. ej., programas de vértices, programas de geometría, sombreadores de vértices), procesamiento de píxeles (p. ej., sombreadores de píxeles, sombreadores de fragmentos) y procesamiento de propósito general (p. ej., sombreadores de computación y medios).

Cada unidad de ejecución en la matriz de unidades de ejecución 2308A-2308N opera en matrices 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 aritméticas lógicas (ALU) o unidades de coma flotante (FPU) de un procesador de gráficos en particular. En algunas realizaciones, las unidades de ejecución 2308A-2308N admiten tipos de datos de números enteros y de coma flotante.

El conjunto de instrucciones de la unidad de ejecución incluye instrucciones de instrucción única, múltiples datos (SIMD) o instrucciones de instrucción única, múltiples hilos (SIMT). Los diversos elementos de datos se pueden almacenar como un tipo de datos empaquetado en un registro y la unidad de ejecución procesará los diversos elementos basándose en el tamaño de datos de los elementos. Por ejemplo, cuando se opera en un vector de 256 bits de ancho, los 256 bits del vector se almacenan en un registro y la unidad de ejecución opera en el vector como cuatro elementos de datos empaquetados de 64 bits separados (elementos de datos de tamaño de palabra cuádruple (QW)), ocho elementos de datos empaquetados de 32 bits separados (elementos de datos de tamaño de palabra doble (DW)), dieciséis elementos de datos empaquetados de 16 bits separados (elementos de datos de tamaño de palabra (W)), o treinta y dos elementos de datos de 8 bits separados (elementos de datos de tamaño de byte (B)). Sin embargo, son posibles diferentes anchos de vector y tamaños de registro.

Una o más memorias caché de instrucciones internas (por ejemplo, 2306) están incluidas en la lógica de ejecución de hilos 2300 para almacenar en caché las instrucciones de hilos para las unidades de ejecución. En algunas realizaciones, una o más memorias caché de datos (por ejemplo, 2312) están incluidas en datos de hilo de caché durante la ejecución de hilo. En algunas realizaciones, se incluye un muestreador 2310 para proporcionar un muestreo de textura para operaciones 3D y muestreo de medios para operaciones de medios. En algunas realizaciones, el muestreador 2310 incluye funcionalidad de textura especializada o muestreo de medios para procesar los datos de textura o de medios durante el proceso de muestreo antes de proporcionar los datos muestreados a una unidad de ejecución.

Durante la ejecución, las canalizaciones de gráficos y medios envían solicitudes de iniciación de hilos a la lógica de ejecución de hilos 2300 a través de la lógica de generación y despacho de hilos. En algunas realizaciones, la lógica de ejecución de hilos 2300 incluye un despachador de hilos local 2304 que arbitra las solicitudes de inicio de hilos de las canalizaciones de gráficos y medios y genera instancias a los hilos solicitados en una o más unidades de ejecución 2308A-2308N. Por ejemplo, la canalización de geometría (por ejemplo, 2236 de la **Figura 22**) despacha hilos de procesamiento de vértices, teselación o procesamiento de geometría a la lógica de ejecución de hilos 2300 (**Figura 23**). En algunas realizaciones, el despachador de hilos 2304 también puede procesar solicitudes de generación de hilos en tiempo de ejecución de los programas de sombreado en ejecución.

Una vez que se ha procesado y rasterizado un grupo de objetos geométricos en datos de píxeles, se invoca el sombreador de píxeles 2302 para calcular más información de salida y hacer que los resultados se escriban en superficies de salida (por ejemplo, memorias intermedias de color, memorias intermedias de profundidad, memorias intermedias de estarcido, etc.). En algunas realizaciones, el sombreador de píxeles 2302 calcula los valores de los diversos atributos de vértice que han de interpolarse a través del objeto rasterizado. En algunas realizaciones, el sombreador de píxeles 2302 a continuación ejecuta un programa de sombreador de píxeles suministrado por la interfaz de programación de aplicaciones (API). Para ejecutar el programa de sombreado de píxeles, el sombreador de píxeles 2302 despacha hilos a una unidad de ejecución (por ejemplo, 2308A) a través del despachador de hilos 2304. En algunas realizaciones, el sombreador de píxeles 2302 usa la lógica de muestreo de textura en el muestreador 2310 para acceder a datos de textura en mapas de textura almacenados en memoria. Las operaciones aritméticas sobre 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 del procesamiento posterior.

En algunas realizaciones, el puerto de datos 2314 proporciona un mecanismo de acceso a la memoria para que la lógica de ejecución de hilos 2300 envíe datos procesados a la memoria para su procesamiento en una canalización de salida de procesador de gráficos. En algunas realizaciones, el puerto de datos 2314 incluye o se acopla a una o más memorias caché (por ejemplo, memoria caché de datos 2312) para almacenar en memoria caché datos para el acceso a la memoria a través del puerto de datos.

La **Figura 24** es un diagrama de bloques que ilustra los formatos de instrucción 2400 del procesador de gráficos de acuerdo con algunas realizaciones. En una o más realizaciones, las unidades de ejecución del procesador de gráficos admiten un conjunto de instrucciones que tiene instrucciones en múltiples formatos. Los recuadros de líneas continuas ilustran los componentes que se incluyen generalmente en una instrucción de unidad de ejecución, mientras que las líneas discontinuas incluyen componentes que son opcionales o que sólo se incluyen en un subconjunto de las instrucciones. En algunas realizaciones, el formato de instrucción 2400 descrito e ilustrado son macroinstrucciones, en el sentido de que son instrucciones suministradas a la unidad de ejecución, en oposición a microoperaciones resultantes de la decodificación de instrucciones una vez que se procesa la instrucción.

En algunas realizaciones, las unidades de ejecución de procesador de gráficos admiten de manera nativa instrucciones en un formato de instrucción de 128 bits 2410. Un formato de instrucción compacta de 64 bits 2430 está disponible para algunas instrucciones basándose en la instrucción, las opciones de instrucción y el número de operandos seleccionados. El formato de instrucción de 128 bits nativo 2410 proporciona acceso a todas las opciones de instrucción, mientras que algunas opciones y operaciones están restringidas en el formato de instrucción de 64 bits 2430. Las instrucciones nativas disponibles en el formato de instrucción de 64 bits 2430 varían según la realización. En algunas realizaciones, la instrucción se compacta en parte utilizando un conjunto de valores de índice en un campo de índice 2413. El hardware de la unidad de ejecución hace referencia a un conjunto de tablas de compactación basándose en los valores de índice y usa las salidas de tabla de compactación para reconstruir una instrucción nativa en el formato de instrucción de 128 bits 2410.

Para cada formato, el código de operación de instrucción 2412 define la operación que la unidad de ejecución debe realizar. Las unidades de ejecución ejecutan cada instrucción en paralelo a lo largo 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 en cada canal de color que representa un elemento de textura o un elemento de imagen. Por defecto, la unidad de ejecución realiza cada instrucción a través de todos los canales de datos de los operandos. En algunas realizaciones, el campo de control de instrucción 2414 permite el control sobre ciertas opciones de ejecución, tal como la selección de canales (por ejemplo, predicación) y el orden de los canales de datos (por ejemplo, mezcla). Para las instrucciones de 128 bits 2410, un campo de tamaño de ejecución 2416 limita la cantidad de canales de datos que se ejecutarán en paralelo. En algunas realizaciones, el campo de tamaño de ejecución 2416 no está disponible para su uso en el formato de instrucción compacto de 64 bits 2430.

Algunas instrucciones de unidad de ejecución tienen hasta tres operandos, incluidos dos operandos de origen, src0 2420, src1 2422 y un destino 2418. En algunas realizaciones, las unidades de ejecución admiten instrucciones de destino dual, donde uno de los destinos está implícito. Las instrucciones de manipulación de datos pueden tener un tercer operando de origen (por ejemplo, SRC2 2424), donde el código de operación de instrucción 2412 determina la cantidad de operandos de origen. El último operando de origen de una instrucción puede ser un valor inmediato (por ejemplo, precodificado) que se pasa con la instrucción.

En algunas realizaciones, el formato de instrucción de 128 bits 2410 incluye una información de modo de acceso/dirección 2426 que especifica, por ejemplo, si se utiliza el modo de direccionamiento de registro directo o el modo de direccionamiento de registro indirecto. Cuando se usa el modo de direccionamiento de registro directo, la dirección de registro de uno o más operandos viene proporcionada directamente por bits en la instrucción 2410.

En algunas realizaciones, el formato de instrucción de 128 bits 2410 incluye un campo de modo de acceso/dirección 2426, que especifica un modo de dirección y/o un modo de acceso para la instrucción. En una realización, el modo de acceso define una alineación de acceso a datos para la instrucción. Algunas realizaciones admiten modos de acceso que incluyen un modo de acceso alineado de 16 bytes y un modo de acceso alineado de 1 byte, donde la alineación de bytes del modo de acceso determina la alineación de acceso de los operandos de instrucción. Por ejemplo, cuando está en un primer modo, la instrucción 2410 puede utilizar direccionamiento alineado de bytes para los operandos de origen y destino y cuando está en un segundo modo, la instrucción 2410 puede utilizar direccionamiento alineado de 16 bytes para todos los operandos de origen y destino.

En una realización, la parte de modo de dirección del campo de modo de acceso/dirección 2426 determina si la instrucción debe usar el direccionamiento directo o indirecto. Cuando se usa el modo de direccionamiento de registro directo, los bits en la instrucción 2410 proporcionan directamente la dirección de registro de uno o más operandos. Cuando se usa un modo de direccionamiento de registro indirecto, la dirección de registro de uno o más operandos puede calcularse basándose en un valor de registro de dirección y un campo inmediato de dirección en la instrucción.

En algunas realizaciones, las instrucciones se agrupan en función de los campos de bits del código de operación 2412 para simplificar la decodificación del código de operación 2440. 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 precisa del código de operación que se muestra es simplemente un ejemplo. En algunas realizaciones, un grupo de código de operación de movimiento y lógica 2442 incluye instrucciones lógicas y de movimiento de datos (por ejemplo, mover (mov), comparar (cmp)). En algunas realizaciones, el grupo de movimiento y lógica 2442 comparte los cinco bits más significativos (MSB), donde las instrucciones de movimiento (mov) tienen la forma de 0000xxxxb y las instrucciones lógicas tienen la forma de 0001xxxxb. Un grupo de instrucciones de control de flujo 2444 (por ejemplo, llamar, saltar (jmp)) incluye instrucciones en la forma de 0010xxxxb (por ejemplo, 0x20). Un grupo de instrucciones misceláneas 2446 incluye una combinación de instrucciones, incluidas las instrucciones de sincronización (por ejemplo, esperar, enviar) en la forma de 0011xxxxb (por ejemplo, 0x30). Un grupo de instrucciones matemáticas paralelas 2448 incluye instrucciones aritméticas por componentes (p. ej., sumar, multiplicar (mul)) en forma de 0100xxxxb (p. ej., 0x40). El grupo de cálculo matemático paralelo 2448 realiza las operaciones aritméticas en paralelo a través de canales de datos. El grupo de cálculo matemático vectorial 2450 incluye instrucciones aritméticas (por ejemplo, dp4) en forma de 0101xxxxb (por ejemplo, 0x50). El grupo de cálculo matemático vectorial realiza operaciones aritméticas, tales como cálculos de producto escalar en operandos vectoriales.

Canalización de gráficos

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

En algunas realizaciones, el procesador de gráficos 2500 incluye una canalización de gráficos 2520, una canalización de medios 2530, un motor de visualización 2540, lógica de ejecución de hilos 2550 y una canalización de salida de renderizado 2570. En algunas realizaciones, el procesador de gráficos 2500 es un procesador de gráficos dentro 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 por las escrituras de registro en uno o más registros de control (no mostrados) o mediante comandos emitidos al procesador de gráficos 2500 mediante una interconexión en anillo 2502. En algunas realizaciones, la interconexión en anillo 2502 acopla el procesador de gráficos 2500 a otros componentes de procesamiento, tales como otros procesadores de gráficos o procesadores de propósito general. Los comandos desde la interconexión en anillo 2502 se interpretan por un transmisor en continuo de comandos 2503, que suministra instrucciones a componentes individuales de la canalización de gráficos 2520 o la canalización de medios 2530.

En algunas realizaciones, el transmisor en continuo de comandos 2503 dirige la operación de un extractor de vértices 2505 que lee datos de vértices desde memoria y ejecuta comandos de procesamiento de vértices proporcionados por el emisor por flujo continuo de comandos 2503. En algunas realizaciones, el extractor de vértices 2505 proporciona datos de vértices a un sombreador de vértices 2507, que realiza operaciones de transformación de espacio de coordenadas y de iluminación para cada vértice. En algunas realizaciones, el extractor de vértices 2505 y el sombreador de vértices 2507 ejecutan instrucciones de procesamiento de vértices despachando hilos de ejecución a las unidades de ejecución 2552A, 2552B a través de un despachador de hilos 2531.

En algunas realizaciones, las unidades de ejecución 2552A, 2552B son una matriz de procesadores vectoriales que tienen un conjunto de instrucciones para realizar operaciones de gráficos y de medios. En algunas realizaciones, las unidades de ejecución 2552A, 2552B tienen una memoria caché L1 2551 conectada que es específica para cada matriz o compartida entre las matrices. La memoria caché se puede configurar como una memoria caché de datos, una memoria caché de instrucciones o una memoria caché única subdividida para contener datos e instrucciones en diferentes particiones.

En algunas realizaciones, la canalización de gráficos 2520 incluye componentes de teselación para realizar una teselación acelerada por hardware de objetos 3D. En algunas realizaciones, un sombreador de casco programable 2511 configura las operaciones de teselación. Un sombreador de dominio programable 2517 proporciona una evaluación de extremo posterior de la salida de teselación. Un teselador 2513 opera en la dirección del sombreador de casco 2511 y contiene una lógica de propósito especial para generar un conjunto de objetos geométricos detallados en función de un modelo geométrico aproximado que se proporciona como entrada a la canalización de gráficos 2520. En algunas realizaciones, si no se utiliza la teselación, se pueden omitir los componentes de teselación 2511, 2513, 2517.

En algunas realizaciones, objetos geométricos completos pueden ser procesados por un sombreador de geometría 2519 a través de uno o más hilos despachados a unidades de ejecución 2552A, 2552B o pueden proseguir directamente al recortador 2529. En algunas realizaciones, el sombreador de geometría opera en objetos geométricos enteros, en lugar de en vértices o parches de vértices como en etapas anteriores de la canalización de gráficos. Si la teselación está deshabilitada, el sombreador de geometría 2519 recibe la entrada del sombreador de vértices 2507. En algunas realizaciones, el sombreador de geometría 2519 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 2529 procesa datos de vértices. El recortador 2529 puede ser un recortador de función fija o un recortador programable que tenga funciones de recorte y sombreador de geometría. En algunas realizaciones, un componente de pruebas de profundidad y rasterización 2573 de la canalización de salida de renderizado 2570 envía sombreadores de píxeles para convertir los objetos geométricos en sus representaciones por píxel. En algunas realizaciones, la lógica de ejecución de hilos 2550 incluye lógica de sombreado de píxeles. En algunas realizaciones, una aplicación puede omitir la rasterización y acceder a datos de vértices no rasterizados a través de una unidad de salida de flujo 2523.

El procesador de gráficos 2500 tiene un bus de interconexión, una estructura de interconexión o algún otro mecanismo de interconexión que permite el paso de datos y mensajes entre los componentes principales del procesador. En algunas realizaciones, las unidades de ejecución 2552A, 2552B y la memoria o memorias caché asociadas 2551, el muestreador de texturas y medios 2554 y la memoria caché de texturas/muestreador 2558 se interconectan a través de un puerto de datos 2556 para realizar acceso a la memoria y comunicarse con los componentes de la canalización de salida de renderizado del procesador. En algunas realizaciones, el muestreador 2554, las memorias caché 2551, 2558 y las unidades de ejecución 2552A, 2552B tienen cada una rutas de acceso a la memoria independientes.

En algunas realizaciones, la canalización de salida de renderizado 2570 contiene un rasterizador y un componente de prueba de profundidad 2573 que convierte objetos basados en vértices en una representación asociada basada en píxeles. En algunas realizaciones, la canalización de salida de renderizado 2570 incluye una unidad de generador de ventanas/enmascaramiento para realizar la rasterización de triángulos y líneas de función fija. En algunas realizaciones también están disponibles una memoria caché de renderizado 2578 y una memoria caché de profundidad 2579 asociadas. Un componente de operaciones de píxeles 2577 realiza operaciones basadas en píxeles sobre los datos, aunque, en algunas instancias, las operaciones de píxeles asociadas con operaciones 2D (por ejemplo, transferencias de imagen de bloque de bits con mezcla) se realizan por el motor 2D 2541, o se sustituyen en el momento de la visualización por el controlador de visualización 2543 usando planos de visualización de superposición. En algunas realizaciones, está disponible una caché L3 compartida 2575 para todos los componentes de gráficos, permitiendo compartir datos sin el uso de memoria de sistema principal.

En algunas realizaciones, la canalización de medios 2530 del procesador de gráficos incluye un motor de medios 2537 y un extremo frontal de vídeo 2534. En algunas realizaciones, el extremo frontal de vídeo 2534 recibe comandos de canalización desde el transmisor en continuo de comandos 2503. En algunas realizaciones, la canalización de medios 2530 incluye un transmisor en continuo de comandos independiente. En algunas realizaciones, el extremo frontal de vídeo 2534 procesa comandos de medios antes de enviar el comando al motor de medios 2537. En algunas realizaciones, el motor de medios 2537 incluye una funcionalidad de generación de hilos para generar hilos para su despacho a la lógica de ejecución de hilos 2550 a través del despachador de hilos 2531.

En algunas realizaciones, el procesador de gráficos 2500 incluye un motor de visualización 2540. En algunas realizaciones, el motor de visualización 2540 es externo al procesador 2500 y se acopla con el procesador de gráficos a través de la interconexión de anillo 2502, o algún otro bus o estructura de interconexión. En algunas realizaciones, el motor de visualización 2540 incluye un motor 2D 2541 y un controlador de visualización 2543. En algunas realizaciones, el motor de visualización 2540 contiene una lógica de propósito especial capaz de funcionar independientemente de la canalización 3D. En algunas realizaciones, el controlador de visualización 2543 se acopla con un dispositivo de visualización (no mostrado), que puede ser un dispositivo de visualización integrado en el sistema, tal como en un ordenador portátil, o un dispositivo de visualización externo conectado a través de un conector de dispositivo de visualización.

En algunas realizaciones, la canalización de gráficos 2520 y la canalización de medios 2530 se pueden configurar para realizar operaciones basadas en múltiples interfaces de programación de gráficos y medios y no son específicas de ninguna interfaz de programación de aplicaciones (API). En algunas realizaciones, el software de controlador para el procesador de gráficos traduce llamadas de API que son específicas de una biblioteca de medios o de gráficos particular a comandos que pueden procesarse por el procesador de gráficos. En algunas realizaciones, se proporciona soporte para la biblioteca Open Graphics (OpenGL) y Open Computing Language (OpenCL) del grupo Khronos, la biblioteca Direct3D de Microsoft Corporation, o se puede proporcionar soporte tanto para OpenGL como para D3D. También se puede proporcionar soporte para la Biblioteca de Visión Informática de Código Abierto (OpenCV). También se admitiría una API futura con una canalización 3D compatible si pudiera hacerse un mapeo de la canalización de la API futura a la canalización del procesador de gráficos.

Programación de canalización de gráficos

La **Figura 26A** es un diagrama de bloques que ilustra un formato de comando de procesador de gráficos 2600 según algunas realizaciones. La **Figura 26B** es un diagrama de bloques que ilustra una secuencia de comandos de procesador de gráficos 2610 de acuerdo con una realización. Los recuadros con líneas continuas en la **Figura 26A** 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 2600 ilustrativo de la **Figura 26A** incluye campos de

datos para identificar un cliente objetivo 2602 del comando, un código de operación de comando (código de operación) 2604 y los datos pertinentes 2606 para el comando. También se incluye un subcódigo de operación 2605 y un tamaño de comando 2608 en algunos comandos.

En algunas realizaciones, el cliente 2602 especifica la unidad cliente del dispositivo de gráficos que procesa los datos de comando. En algunas realizaciones, un analizador de comandos del procesador de gráficos examina el campo de cliente de cada comando para condicionar el procesamiento posterior del comando y dirigir los datos de comando a la unidad de cliente apropiada. En algunas realizaciones, las unidades cliente del procesador de gráficos incluyen una unidad de interfaz de memoria, una unidad de renderizado, una unidad 2D, una unidad 3D y una unidad multimedia. Cada unidad cliente tiene una canalización de procesamiento correspondiente que procesa los comandos. Una vez que la unidad cliente recibe el comando, la unidad cliente lee el código de operación 2604 y, si está presente, el subcódigo de operación 2605 para determinar la operación a realizar. La unidad cliente ejecuta el comando utilizando la información del campo de datos 2606. Para algunos comandos se espera un tamaño de comando explícito 2608 para especificar el tamaño del comando. En algunas realizaciones, el analizador de comandos determina automáticamente el tamaño de al menos algunos de los comandos basándose en el código de operación de comando. En algunas realizaciones, los comandos se alinean a través de múltiplos de una palabra doble.

El diagrama de flujo de la **Figura 26B** muestra una secuencia de comandos de procesador de gráficos 2610 a modo de ejemplo. En algunas realizaciones, el software o firmware de un sistema de procesamiento de datos que caracteriza una realización de un procesador de gráficos utiliza una versión de la secuencia de comandos mostrada para configurar, ejecutar y terminar un conjunto de operaciones gráficas. Se muestra una secuencia de comandos de muestra y se describe para los fines de ejemplo únicamente ya que las realizaciones no se limitan a estas comandos específicas o para esta secuencia de comandos. Por otra parte, los comandos se pueden emitir como un lote de comandos en una secuencia de comandos, de tal forma que el procesador de gráficos procesará la secuencia de comandos en al menos parcialmente concurrencia.

En algunas realizaciones, la secuencia de comandos del procesador de gráficos 2610 puede comenzar con un comando de vaciado de canalización 2612 para hacer que cualquier canalización de gráficos activa complete los comandos actualmente pendientes para la canalización. En algunas realizaciones, la canalización de 3D 2622 y la canalización de medios 2624 no operan al mismo tiempo. El vaciado de la canalización se lleva a cabo para hacer que la canalización de gráficos activo complete cualesquiera comandos pendientes. En respuesta a un vaciado de la canalización, el analizador de comandos para el procesador de gráficos pausará el procesamiento de comandos hasta que los motores de dibujo activos completen las operaciones pendientes y se invaliden las memorias caché de lectura relevantes. Opcionalmente, cualquier dato en la caché del renderizador que se marque como 'sucia' se puede descargar a la memoria. En algunas realizaciones, el comando de vaciado de canalización 2612 se puede usar para la sincronización de la canalización o antes de hacer que el procesador de gráficos pase un estado de baja potencia.

En algunas realizaciones, se utiliza un comando de selección de canalización 2613 cuando una secuencia de comandos requiere que el procesador de gráficos cambie explícitamente entre canalizaciones. En algunas realizaciones, un comando de selección de canalización 2613 se requiere sólo una vez dentro de un contexto de ejecución antes de emitir comandos de canalización a menos que el contexto sea para emitir comandos para ambas canalizaciones. En algunas realizaciones, se requiere un comando de vaciado de canalización 2612 inmediatamente antes de una conmutación de canalización mediante el comando de selección de canalización 2613.

En algunas realizaciones, un comando de control de canalización 2614 configura una canalización de gráficos para la operación y se usa para programar la canalización 3D 2622 y la canalización de medios 2624. En algunas realizaciones, el comando de control de canalización 2614 configura el estado de la canalización para la canalización activa. En una realización, el comando de control de canalización 2614 se utiliza para la sincronización de la canalización y para borrar datos de una o más memorias caché dentro de la canalización activa antes de procesar un lote de comandos.

En algunas realizaciones, los comandos para el estado de la memoria intermedia de retorno 2616 se utilizan para configurar un conjunto de memorias intermedias de retorno para que las respectivas canalizaciones escriban datos. Algunas operaciones de canalización requieren la asignación, selección o configuración de una o más memorias intermedias de retorno en las que las operaciones escriben datos intermedios durante el procesamiento. En algunas realizaciones, el procesador de gráficos también utiliza una o más memorias intermedias de retorno para almacenar datos de salida y para realizar una comunicación entre hilos. En algunas realizaciones, la configuración del estado de la memoria intermedia de retorno 2616 incluye la selección del tamaño y número de memorias intermedias de retorno a utilizar para un conjunto de operaciones de canalización.

Los comandos restantes en la secuencia de comandos difieren en función de la canalización activa para las operaciones. Basándose en una determinación de la canalización 2620, la secuencia de comandos se adapta a la canalización 3D 2622 que comienza con el estado de canalización 3D 2630, o a la canalización de medios 2624 que comienza en el estado de canalización de medios 2640.

Los comandos para el estado de canalización 3D 2630 incluyen los comandos de ajuste de estado 3D para el estado de memoria intermedia de vértice, estado de elemento de vértice, estado de color constante, estado de memoria intermedia de profundidad y otras variables de estado que han de configurarse antes de que se procesen los comandos de primitiva 3D. Los valores de estos comandos se determinan, al menos en parte, basándose en la API 3D particular en uso. En algunas realizaciones, los comandos de estado de canalización 3D 2630 también pueden deshabilitar o eludir selectivamente ciertos elementos de la canalización si esos elementos no se utilizarán.

En algunas realizaciones, el comando de primitiva 3D 2632 se utiliza para enviar primitivas 3D para que sean procesadas por la canalización 3D. Los comandos y parámetros asociados que se pasan al procesador de gráficos a través del comando de primitiva 3D 2632 se reenvían a la función de extracción de vértices en la canalización de gráficos. La función de extracción de vértices utiliza los datos del comando de primitiva 3D 2632 para generar estructuras de datos de vértices. Las estructuras de datos de vértices se almacenan en una o más memorias intermedias de retorno. En algunas realizaciones, el comando de primitiva 3D 2632 se utiliza para realizar operaciones de vértice en primitivas 3D a través de sombreadores de vértices. Para procesar sombreadores de vértices, la canalización 3D 2622 despacha hilos de ejecución de sombreador a unidades de ejecución de procesador de gráficos.

En algunas realizaciones, se activa la canalización 3D 2622 mediante un comando o evento de ejecución 2634. En algunas realizaciones, una escritura de registro activa la ejecución de comando. En algunas realizaciones, se activa la ejecución mediante un comando 'ir' o 'disparar' en la secuencia de comandos. En una realización se activa la ejecución de comando usando un comando de sincronización de canalización para vaciar la secuencia de comandos a través de la canalización de gráficos. La canalización 3D realizará un 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 extremo posterior de píxeles para esas operaciones.

En algunas realizaciones, la secuencia de comandos 2610 del procesador de gráficos sigue la ruta de la canalización de medios 2624 cuando realiza operaciones de medios. En general, el uso específico y la forma de programación de la canalización de medios 2624 depende de las operaciones de medios o de cálculo a realizar. Se pueden descargar operaciones de decodificación de medios específicas a la canalización de medios durante la decodificación de medios. En algunas realizaciones, puede desviarse también la canalización de medios y puede realizarse la decodificación de medios, en su totalidad o en parte, usando recursos proporcionados por uno o más núcleos de procesamiento de propósito general. En una realización, la canalización de medios también incluye elementos para operaciones de la unidad de procesador de gráficos de propósito general (GPGPU), donde el procesador de gráficos se usa para realizar operaciones vectoriales de SIMD usando programas sombreadores computacionales que no están relacionados explícitamente con la representación de primitivas de gráficos.

En algunas realizaciones, se configura la canalización de medios 2624 de una manera similar que la canalización 3D 2622. Un conjunto de comandos para configurar el estado de canalización de medios 2640 se despacha o coloca en una cola de comandos antes de los comandos de objeto de medios 2642. En algunas realizaciones, los comandos para el estado de canalización de medios 2640 incluyen datos para configurar los elementos de canalización de medios que se usarán para procesar los objetos de medios. Esto incluye datos para configurar la lógica de decodificación de vídeo y de codificación de vídeo dentro de la canalización de medios, tal como el formato de codificación o de decodificación. En algunas realizaciones, los comandos para el estado de canalización de medios 2640 también admiten el uso de uno o más punteros a elementos de estado "indirectos" que contienen un lote de configuraciones de estado.

En algunas realizaciones, los comandos de objeto de medios 2642 suministran punteros a objetos de medios para su procesamiento por la canalización de medios. Los objetos de medios incluyen memorias intermedias de memoria que contienen datos de vídeo para procesar. En algunas realizaciones, todos los estados de canalización de medios deben ser válidos antes de que se emita un comando de objeto de medios 2642. Una vez que se configura el estado de la canalización y se ponen en cola los comandos de objeto de medios 2642, la canalización de medios 2624 se activa a través de un comando de ejecución 2644 o un evento de ejecución equivalente (por ejemplo, escritura de registro). La salida de la canalización de medios 2624 puede ser procesada posteriormente por operaciones proporcionadas por la canalización 3D 2622 o la canalización de medios 2624. En algunas realizaciones, las operaciones de GPGPU se configuran y ejecutan de forma similar a las operaciones de medios.

Arquitectura de software de gráficos

La **Figura 27** ilustra una arquitectura de software de gráficos a modo de ejemplo para un sistema de procesamiento de datos 2700 de acuerdo con algunas realizaciones. En algunas realizaciones, la arquitectura de software incluye una aplicación de gráficos 3D 2710, un sistema operativo 2720 y al menos un procesador 2730. En algunas realizaciones, el procesador 2730 incluye un procesador de gráficos 2732 y uno o más núcleos de procesador de propósito general 2734. Cada uno de la aplicación de gráficos 2710 y el sistema operativo 2720 se ejecutan en la memoria de sistema 2750 del sistema de procesamiento de datos.

En algunas realizaciones, la aplicación de gráficos 3D 2710 contiene uno o más programas de sombreado que incluyen instrucciones de sombreado 2712. Las instrucciones del 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 aplicación también incluye las instrucciones ejecutables 2714 en un lenguaje máquina adecuado para su ejecución por el núcleo o núcleos de procesador de propósito general 2734. La aplicación también incluye objetos gráficos 2716 definidos por datos de vértice.

En algunas realizaciones, el sistema operativo 2720 es un sistema operativo Microsoft® Windows® de Microsoft Corporation, un sistema operativo propietario similar a UNIX o un sistema operativo de código abierto similar a UNIX que utiliza una variante del núcleo Linux. El sistema operativo 2720 puede admitir una API de gráficos 2722 tal como la API Direct3D o la API OpenGL. Cuando se utiliza la API Direct3D, el sistema operativo 2720 utiliza un compilador de sombreado de extremo frontal 2724 para compilar cualquier instrucción de sombreado 2712 en 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 precompilación de sombreadores. En algunas realizaciones, los sombreadores de alto nivel se compilan en sombreadores de bajo nivel durante la compilación de la aplicación de gráficos 3D 2710.

En algunas realizaciones, el controlador de gráficos en modo de usuario 2726 contiene un compilador de sombreado de extremo posterior 2727 para convertir las instrucciones de sombreado 2712 en una representación específica de hardware. Cuando está en uso la API de OpenGL, las instrucciones de sombreado 2712 en el lenguaje de alto nivel GLSL se pasan a un controlador de gráficos de modo de usuario 2726 para su compilación. En algunas realizaciones, el controlador de gráficos de modo de usuario 2726 usa las funciones de modo de núcleo de sistema operativo 2728 para comunicarse con un controlador de gráficos de modo de núcleo 2729. En algunas realizaciones, el controlador de gráficos de modo de núcleo 2729 se comunica con el procesador de gráficos 2732 para despachar comandos e instrucciones.

Implementaciones de núcleo de IP

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

La **Figura 28** es un diagrama de bloques que ilustra un sistema de desarrollo de núcleo IP 2800 que se puede usar para fabricar un circuito integrado para llevar a cabo operaciones de acuerdo con una realización. El sistema de desarrollo de núcleo IP 2800 se puede usar para generar diseños modulares reutilizables que se pueden incorporar a un diseño mayor o usar para construir un circuito integrado completo (por ejemplo, un circuito integrado SOC). Una instalación de diseño 2830 puede generar una simulación de software 2810 de un diseño de núcleo de IP en un lenguaje de programación de alto nivel (por ejemplo, C/C++). La simulación de software 2810 puede usare para diseñar, probar y verificar el comportamiento del núcleo de IP usando un modelo de simulación 2812. El modelo de simulación 2812 puede incluir simulaciones funcionales, de comportamiento y/o de temporización. Luego, se puede crear o sintetizar un diseño de nivel de transferencia de registro (RTL) 2815 a partir del modelo de simulación 2812. El diseño RTL 2815 es una abstracción del comportamiento del circuito integrado que modela el flujo de señales digitales entre registros de hardware, incluida la lógica asociada realizada utilizando las señales digitales modeladas. Además de un diseño RTL 2815, también pueden crearse, diseñarse o sintetizarse diseños de nivel inferior a nivel lógico o a nivel de transistor. Por lo tanto, los detalles particulares del diseño inicial y la simulación pueden variar.

El diseño RTL 2815 o equivalente puede ser sintetizado adicionalmente por la instalación de diseño en un modelo de hardware 2820, 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 más para verificar el diseño del núcleo IP. El diseño del núcleo IP puede ser almacenado para su entrega a una instalación de fabricación de terceros 2865 utilizando memoria no volátil 2840 (por ejemplo, disco duro, memoria flash, o cualquier medio de almacenamiento no volátil). Como alternativa, el diseño del núcleo IP se puede transmitir (por ejemplo, a través de Internet) a través de una conexión cableada 2850 o una conexión inalámbrica 2860. La instalación de fabricación 2865 puede entonces fabricar un circuito integrado que se basa al menos en parte en el diseño del núcleo IP. El circuito integrado fabricado puede estar configurado para realizar operaciones de acuerdo con al menos una realización descrita en el presente documento.

Circuito integrado de sistema en chip a modo de ejemplo

Las Figuras 29-31 ilustran circuitos integrados a modo de ejemplo y procesadores de gráficos asociados que se pueden fabricar utilizando uno o más núcleos IP, de acuerdo con diversas realizaciones descritas en el presente documento. Además de lo que se ilustra, se pueden incluir otros circuitos y lógicas, incluidos procesadores o núcleos gráficos adicionales, controladores de interfaz periférica o núcleos de procesador de propósito general.

La **Figura 29** es un diagrama de bloques que ilustra un circuito integrado en un microprocesador de sistema 2900 de ejemplo que se puede fabricar utilizando uno o más núcleos IP, de acuerdo con una realización. El circuito integrado 2900 a modo de ejemplo incluye uno o más procesadores de aplicación 2905 (por ejemplo, CPU), al menos un procesador de gráficos 2910, y puede incluir adicionalmente un procesador de imagen 2915 y/o un procesador de vídeo 2920, cualquiera de los cuales puede ser un núcleo IP modular de la misma o múltiples instalaciones de diseño diferentes. El circuito integrado 2900 incluye una lógica de bus o de periféricos que incluye un controlador de USB 2925, un controlador de UART 2930, un controlador de SPI/SDIO 2935 y un controlador de I^2S/I^2C 2940. Además, el circuito integrado puede incluir un dispositivo de visualización 2945 acoplado a uno o más de un controlador de interfaz multimedia de alta definición (HDMI) 2950 y una interfaz de visualización de interfaz de procesador de la industria móvil (MIPI) 2955. El almacenamiento puede ser proporcionado por un subsistema de memoria flash 2960 que incluye memoria flash y un controlador de memoria flash. La interfaz de memoria se puede proporcionar a través de un controlador de memoria 2965 para el acceso a dispositivos de memoria SDRAM o SRAM. Algunos circuitos integrados incluyen adicionalmente un motor de seguridad integrado 2970.

La **Figura 30** es un diagrama de bloques que ilustra un procesador de gráficos 3010 a modo de ejemplo de un circuito integrado de sistema en chip que puede fabricarse utilizando uno o más núcleos IP, de acuerdo con una realización. El procesador de gráficos 3010 puede ser una variante del procesador de gráficos 2910 de la **Figura 29.** El procesador de gráficos 3010 incluye un procesador de vértices 3005 y uno o más procesadores de fragmentos 3015A-3015N (por ejemplo, 3015A, 3015B, 3015C, 3015D, hasta 3015N-1 y 3015N). El procesador de gráficos 3010 puede ejecutar diferentes programas de sombreado a través de una lógica separada, de modo que el procesador de vértices 3005 está optimizado para ejecutar operaciones para programas de sombreado de vértices, mientras que el uno o más procesadores de fragmentos 3015A-3015N ejecutan operaciones de sombreado de fragmentos (por ejemplo, píxeles) para programas de sombreado de fragmentos o píxeles. El procesador de vértices 3005 realiza la etapa de procesamiento de vértices de la canalización de gráficos 3D y genera primitivas y datos de vértices. El procesador o procesadores de fragmentos 3015A-3015N usan los datos de primitiva y de vértice generados por el procesador de vértices 3005 para producir una memoria intermedia de fotogramas que se visualiza en un dispositivo de visualización. En una realización, el procesador o procesadores de fragmentos 3015A-3015N están optimizados para ejecutar programas de sombreado de fragmento según se proporciona en la API de OpenGL, que pueden usarse para realizar operaciones similares como un programa de sombreado de píxeles como se proporciona en la API de Direct 3D.

El procesador de gráficos 3010 incluye adicionalmente una o más unidades de gestión de memoria (MMU) 3020A-3020B, la memoria o memorias caché 3025A-3025B e interconexión o interconexiones de circuito 3030A-3030B. La una o más MMU 3020A-3020B proporcionan el mapeo de direcciones virtuales a físicas para el procesador de gráficos 3010, incluyendo para el procesador de vértices 3005 y/o el procesador o procesadores de fragmentos 3015A-3015N, que pueden hacer referencia a datos de vértices o imágenes/texturas almacenados en memoria, además de datos de vértices o imágenes/texturas almacenados en la una o más memorias caché 3025A-3025B. En una realización, la una o más MMU 3020A-3020B pueden estar sincronizadas con otras MMU dentro del sistema, incluyendo una o más MMU asociadas con el uno o más procesadores de aplicación 2905, el procesador de imagen 2915 y/o el procesador de vídeo 2920 de la **Figura 29**, de modo que cada procesador 2905-2920 puede participar en un sistema de memoria virtual compartido o unificado. El uno o más interconectores de circuitos 3030A-3030B permiten que el procesador de gráficos 3010 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 realizaciones.

La **Figura 31** es un diagrama de bloques que ilustra un procesador de gráficos 3110 adicional ilustrativo de un circuito integrado de sistema en chip que se puede fabricar usando uno o más núcleos IP, de acuerdo con una realización. El procesador de gráficos 3110 puede ser una variante del procesador de gráficos 2910 de la **Figura 29.** El procesador de gráficos 3110 incluye la una o más MMU 3020A-3020B, la memoria o memorias caché 3025A-3025B e interconexión o interconexiones de circuito 3030A-3030B del circuito integrado 3000 de la **Figura 30**.

El procesador de gráficos 3110 incluye uno o más núcleos de sombreado 3115A-3115N (por ejemplo, 3115A, 3115B, 3115C, 3115D, 3115E, 3115F, hasta 3015N-1 y 3015N), lo que proporciona una arquitectura de núcleo de sombreado unificada en la que un único núcleo o tipo de 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 cómputo. El número exacto de núcleos de sombreador presentes puede variar entre realizaciones e implementaciones. Además, el procesador de gráficos 3110 incluye un gestor de tareas internúcleo 3105, que actúa como un despachador de hilos para enviar hilos de ejecución a uno o más núcleos de sombreado 3115A-3115N. El procesador de gráficos 3110 incluye adicionalmente una unidad de mosaico 3118 para acelerar las operaciones de mosaico para la representación basada en mosaicos, en la que las operaciones de representación para una escena se subdividen en el espacio de la imagen. La representación basada en mosaicos se puede usar para aprovechar la coherencia espacial local dentro de una escena o para optimizar el uso de memorias caché internas.

Las referencias a "una realización", "una realización de ejemplo", "diversas realizaciones", etc., indican que la realización o realizaciones así descritas pueden incluir características, estructuras o rasgos particulares, pero no todas las realizaciones incluyen necesariamente las características, estructuras o rasgos particulares. Además, algunas realizaciones pueden tener algunas, todas o ninguna de las características descritas para otras realizaciones.

En la memoria descriptiva anterior, se han descrito realizaciones con referencia a realizaciones específicas a modo de ejemplo de las mismas. Sin embargo, será evidente que se pueden realizar varias modificaciones y cambios a las mismas sin apartarse del alcance de las realizaciones como se exponen en las reivindicaciones adjuntas. En consecuencia, la memoria descriptiva y los dibujos han de considerarse en un sentido ilustrativo más que restrictivo.

En la siguiente descripción y las reivindicaciones, puede usarse el término "acoplado" junto con sus derivados. "Acoplado" se usa para indicar que dos o más elementos cooperan o interactúan entre sí, pero pueden tener o no componentes físicos o eléctricos intermedios entre ellos.

Como se usa en las reivindicaciones, a menos que se especifique lo contrario, el uso de los adjetivos ordinales "primero", "segundo", "tercero", etc., para describir un elemento común, simplemente indica que se hace referencia a diferentes instancias de elementos similares, y no pretenden implicar que los elementos así descritos deban estar en una secuencia determinada, ya sea temporal, espacial, en clasificación o de cualquier otra manera.

REIVINDICACIONES

1. Aparato (600) configurado para facilitar la compartición de datos y la compresión y expansión de modelos de datos, comprendiendo el aparato:
lógica de detección/observación (701) configurada para detectar un primer procesador que procesa información relacionada con una red neuronal en el aparato, en el que la red neuronal incluye una red neuronal convolucional, CNN, y en el que el primer procesador comprende un primer procesador de gráficos (614) y el aparato (600) comprende una primera máquina autónoma; y
lógica de compartición/recuperación de datos (705) configurada para facilitar que el primer procesador almacene datos de red neuronal, NN, intermedios en una biblioteca (731) en una base de datos (730), en el que los datos de NN intermedios son accesibles para un segundo procesador de un dispositivo informático (740) a través de uno o más medios de comunicación,
en el que la lógica de compartición/recuperación de datos (705) está configurada, además, para facilitar que el segundo procesador acceda y recupere los datos NN intermedios de la biblioteca (731) cuando el segundo procesador realiza tareas relacionadas con la red neuronal, en el que el segundo procesador comprende un segundo procesador de gráficos y el dispositivo informático comprende una segunda máquina autónoma,
comprendiendo, además, una lógica de compresión/expansión (709) configurada para comprimir los datos NN intermedios mediante la aplicación de un artefacto y en el que el artefacto y una combinación de los datos NN intermedios comprimidos junto con el artefacto se comunican a la segunda máquina autónoma a través del uno o más medios de comunicación (725), sirviendo dicho artefacto tanto como una extensión de los datos NN intermedios comprimidos como una forma de identificación, en el que la lógica de compresión/expansión (709) está configurada, además, para facilitar la recepción del artefacto y la combinación de los datos NN intermedios comprimidos y el artefacto en la segunda máquina autónoma, en el que los datos NN intermedios comprimidos se descomprimen utilizando el artefacto.

2. Aparato, según la reivindicación 1, que comprende, además, una lógica de generación/mapeo de bibliotecas (703) configurada para generar la biblioteca (731) en la base de datos (730), en el que la lógica de generación/mapeo de bibliotecas (703) está configurada, además, para mapear los datos NN intermedios al primer procesador, en el que las máquinas autónomas primera y segunda incluyen vehículos autónomos en comunicación a través de uno o más medios de comunicación (725), en el que la base de datos (730) incluye una base de datos en la nube.

3. Aparato, según la reivindicación 1, en el que el primer procesador de gráficos (614) está ubicado junto con un procesador de aplicaciones (612) en un paquete semiconductor común.

4. Método para facilitar la compartición de datos y la compresión y expansión de modelos de datos, comprendiendo el método:
detectar un primer procesador que procesa información relacionada con una red neuronal en un primer dispositivo informático (600), en el que la red neuronal incluye una red neuronal convolucional, CNN, y en el que el primer procesador comprende un primer procesador de gráficos (614) y el primer dispositivo informático (600) comprende una primera máquina autónoma; y
facilitar que el primer procesador almacene datos de red neuronal, NN, intermedios en una biblioteca de superficies (731) en una base de datos (730), en el que los datos NN intermedios son accesibles para un segundo procesador de un dispositivo informático (740) a través de uno o más medios de comunicación,
comprendiendo, además, facilitar que el segundo procesador acceda y recupere los datos NN intermedios de la biblioteca de superficies (731) cuando el segundo procesador realiza tareas relacionadas con la red neuronal, en el que el segundo procesador comprende un segundo procesador de gráficos (744) y el dispositivo informático (740) comprende una segunda máquina autónoma, comprendiendo, además, comprimir los datos NN intermedios mediante la aplicación de un artefacto y en el que el artefacto y una combinación de los datos NN intermedios comprimidos junto con el artefacto se comunican a la segunda máquina autónoma a través del uno o más medios de comunicación (725) y en el que los datos NN intermedios comprimidos se descomprimen utilizando el artefacto, sirviendo dicho artefacto tanto como una extensión de los datos NN intermedios comprimidos como una forma de identificación.

5. Método, según la reivindicación 4, que comprende, además, al menos un de:
generar la biblioteca (731) en la base de datos (730); y/o
mapear los datos NN intermedios al primer procesador, en el que la primera y la segunda máquinas autónomas incluyen vehículos autónomos en comunicación a través de uno o más medios de comunicación (725), en el que la base de datos (730) incluye una base de datos en la nube.

6. Método, según la reivindicación 4, que comprende, además, facilitar la recepción de los datos NN intermedios comprimidos y el artefacto en la segunda máquina autónoma, en el que el primer procesador de gráficos (614) está ubicado junto con un procesador de aplicaciones (612) en un paquete semiconductor común.

7. Al menos un medio legible por máquina que comprende una pluralidad de instrucciones, cuando se ejecutan en un dispositivo informático, para implementar o realizar un método según se reivindica en cualquiera de las reivindicaciones 4 a 6.

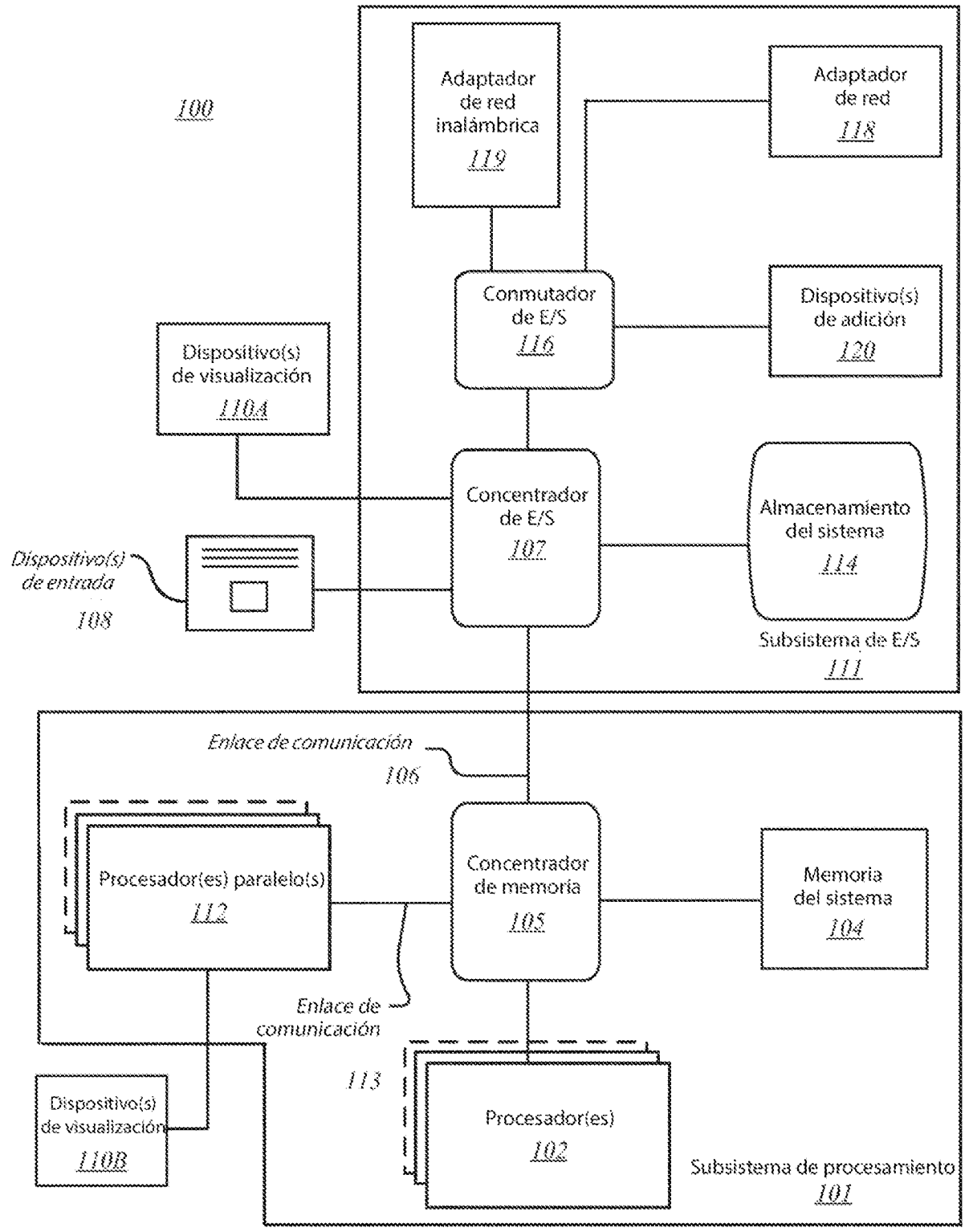

100
Adaptador de red inalámbrica 119
Adaptador de red 118
Conmutador de E/S 116
Dispositivo(s) de adición 120
Dispositivo(s) de visualización 110A
Concentrador de E/S 107
Almacenamiento del sistema 114
Dispositivo(s) de entrada 108
Subsistema de E/S 111
Enlace de comunicación 106
Procesador(es) paralelo(s)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

FIG. 1

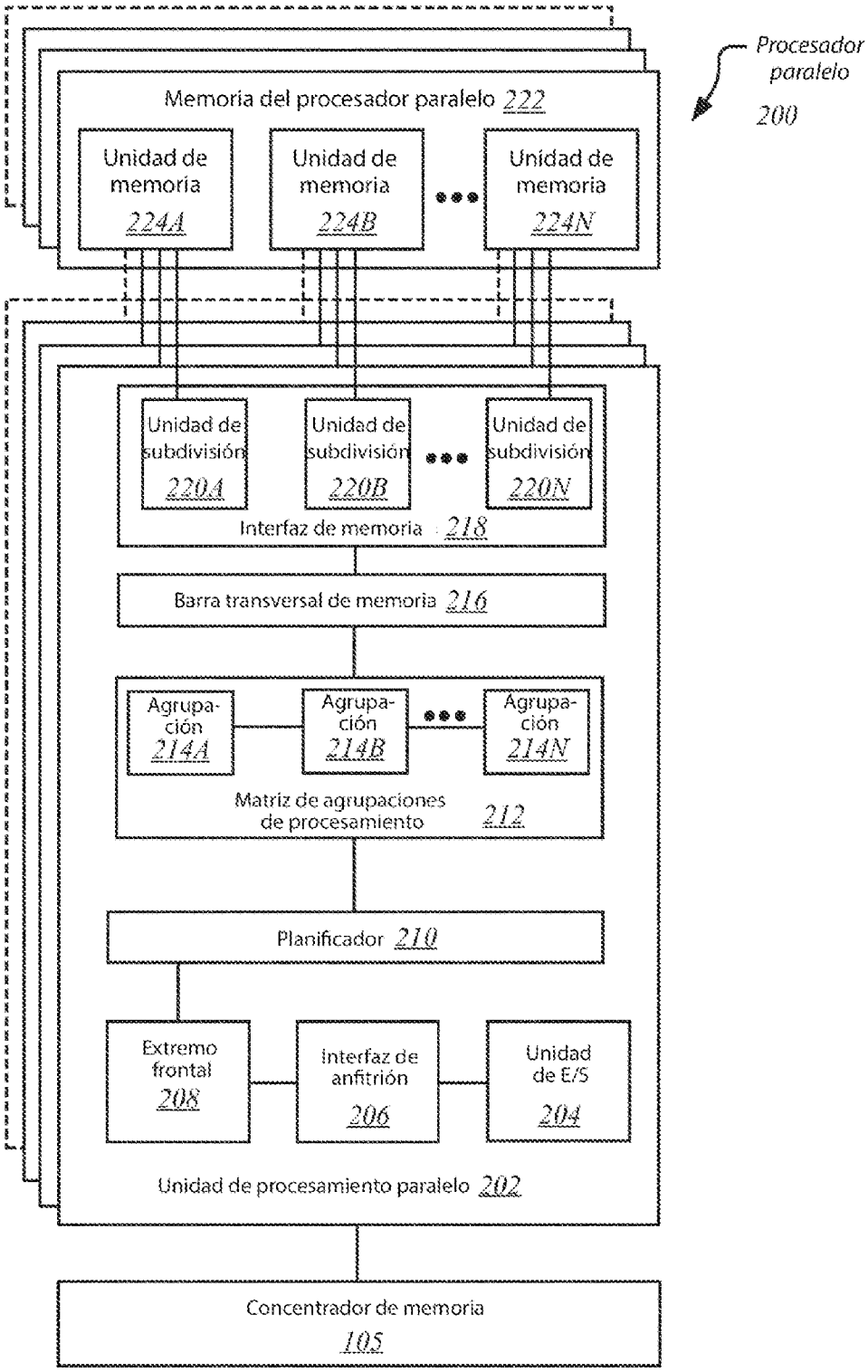

Procesador paralelo
200
Memoria del procesador paralelo 222
Unidad de memoria 224A
Unidad de memoria 224B
Unidad de memoria 224N
Unidad de subdivisión 220A
Unidad de subdivisión 220B
Unidad de subdivisión 220N
Interfaz de memoria 218
Barra transversal de memoria 216
Agrupación 214A
Agrupación 214B
Agrupación 214N
Matriz de agrupaciones de procesamiento 212
Planificador 210
Extremo frontal 208
Interfaz de anfitrión 206
Unidad de E/S 204
Unidad de procesamiento paralelo 202
Concentrador de memoria 105

FIG. 2A

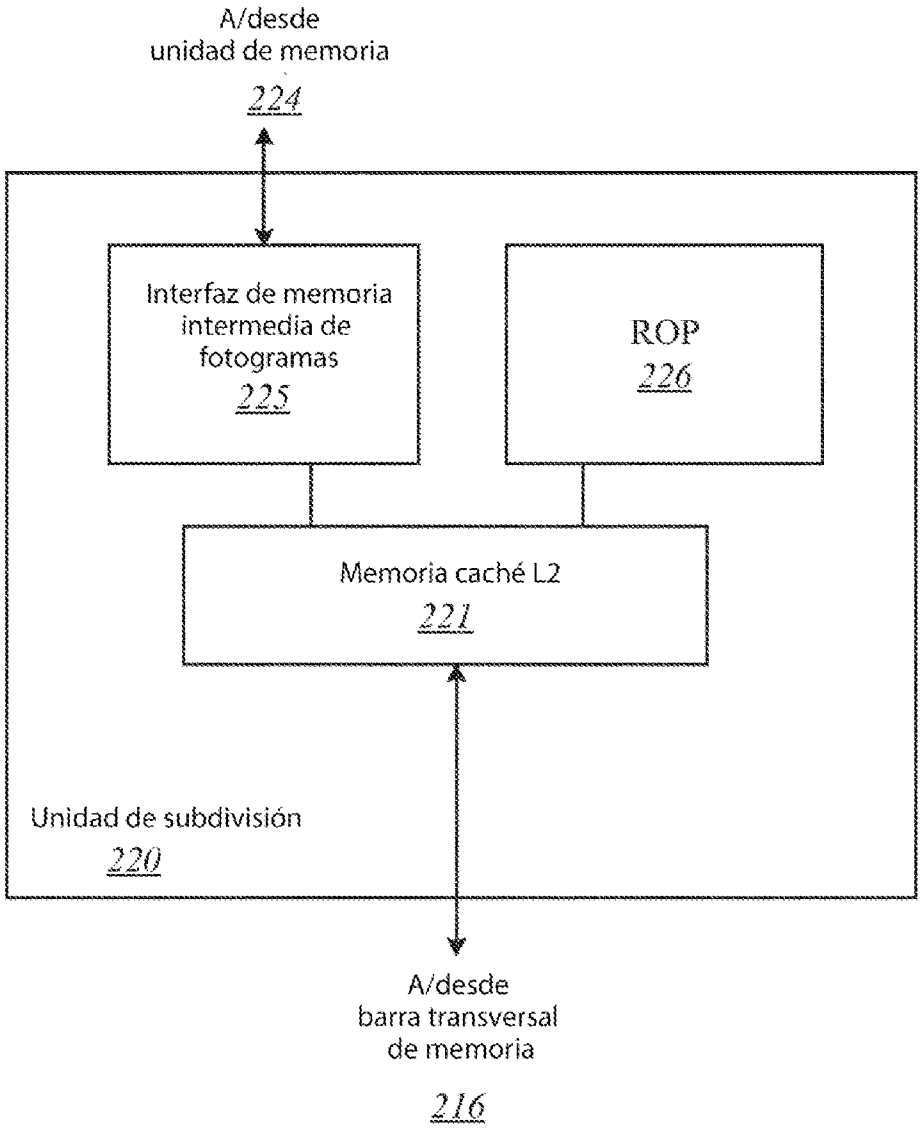

A/desde
unidad de memoria
224
Interfaz de memoria
intermedia de
fotogramas
225
ROP
226
Memoria caché L2
221
Unidad de subdivisión
220
A/desde
barra transversal
de memoria
216

FIG. 2B

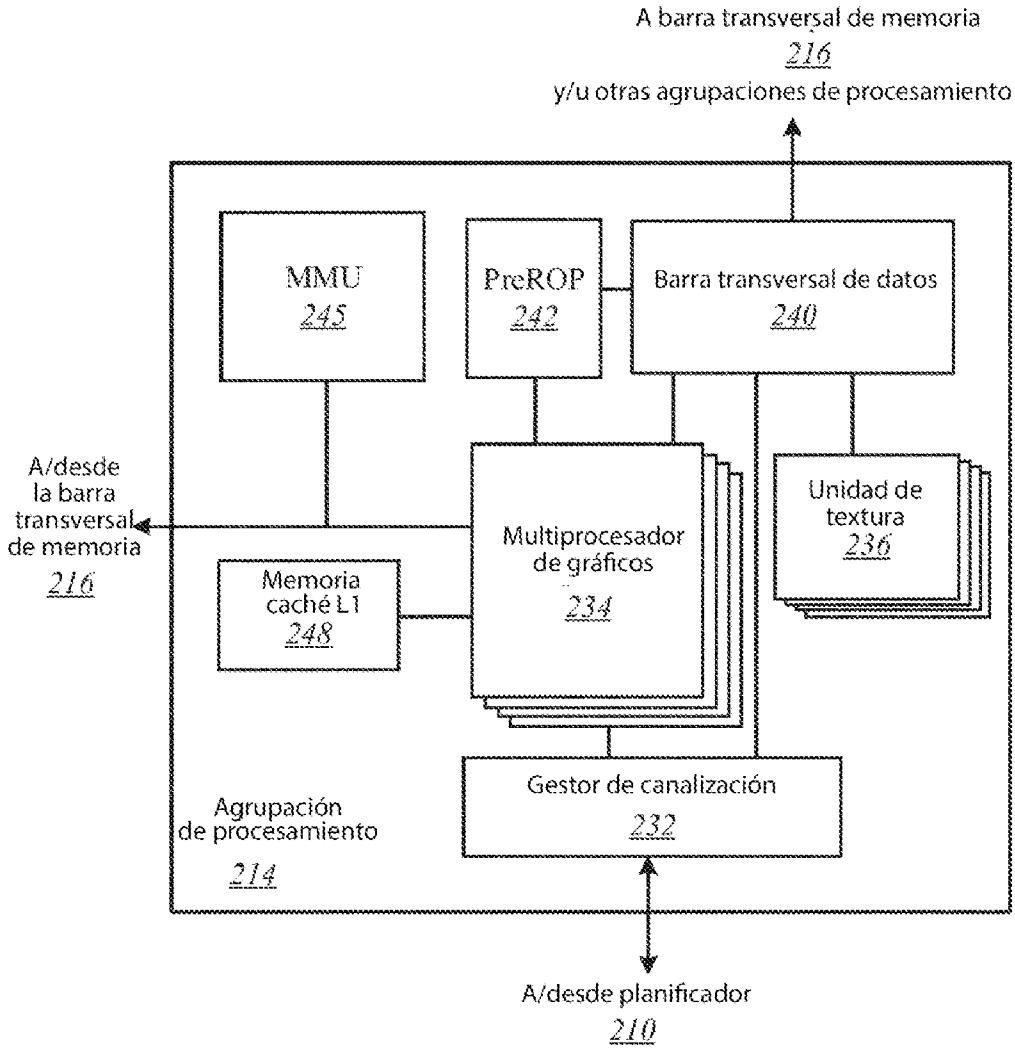

A barra transversal de memoria
216
y/u otras agrupaciones de procesamiento
MMU
245
PreROP
242
Barra transversal de datos
240
A/desde
la barra
transversal
de memoria
216
Multiprocesador
de gráficos
234
Unidad de
textura
236
Memoria
caché L1
248
Gestor de canalización
232
Agrupación
de procesamiento
214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

FIG. 2C

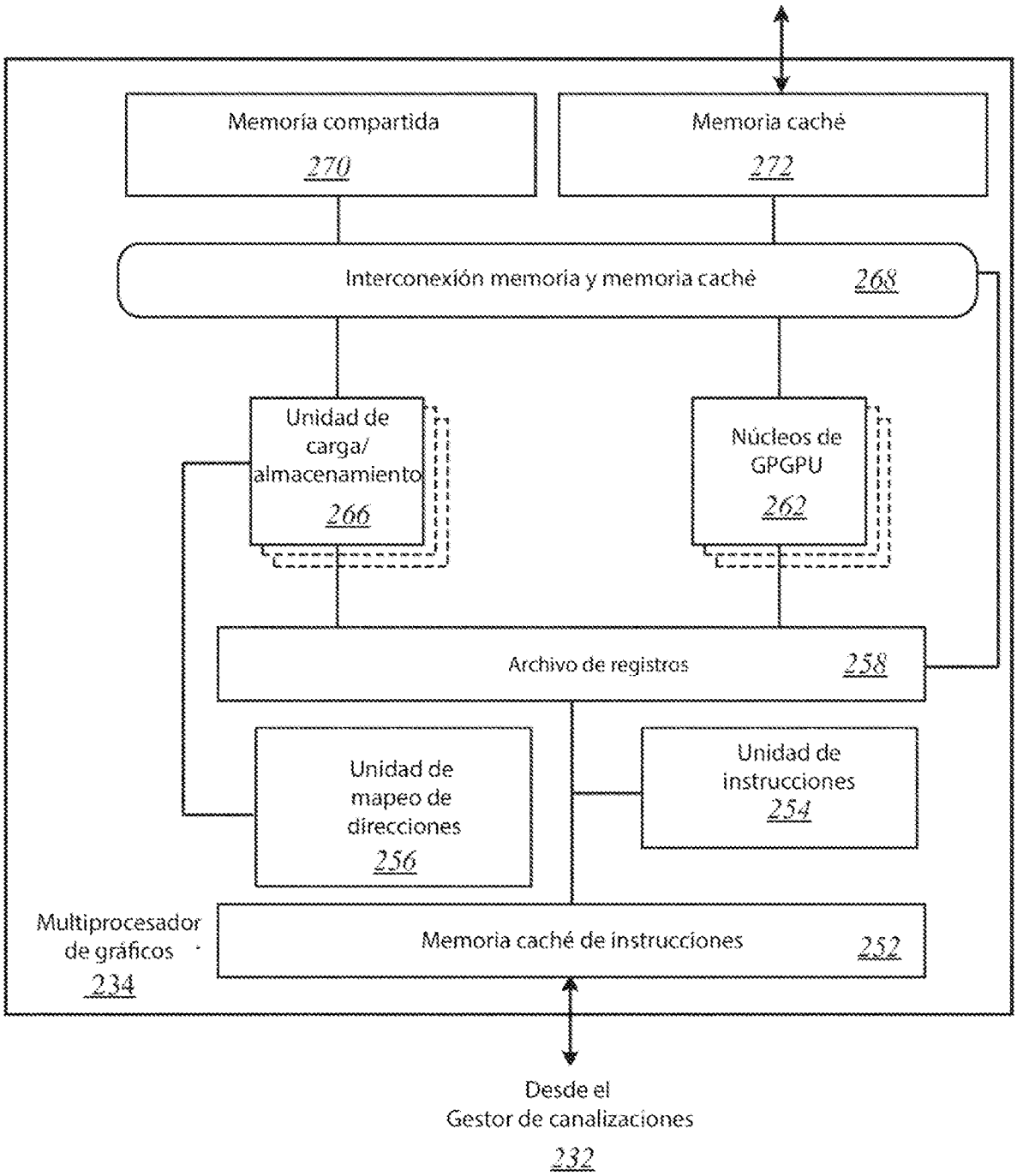

Memoria compartida
270
Memoria caché
272
Interconexión memoria y memoria caché
268
Unidad de carga/ almacenamiento
266
Núcleos de GPGPU
262
Archivo de registros
258
Unidad de mapeo de direcciones
256
Unidad de instrucciones
254
Multiprocesador de gráficos
234
Memoria caché de instrucciones
252
Desde el Gestor de canalizaciones
232

FIG. 2D

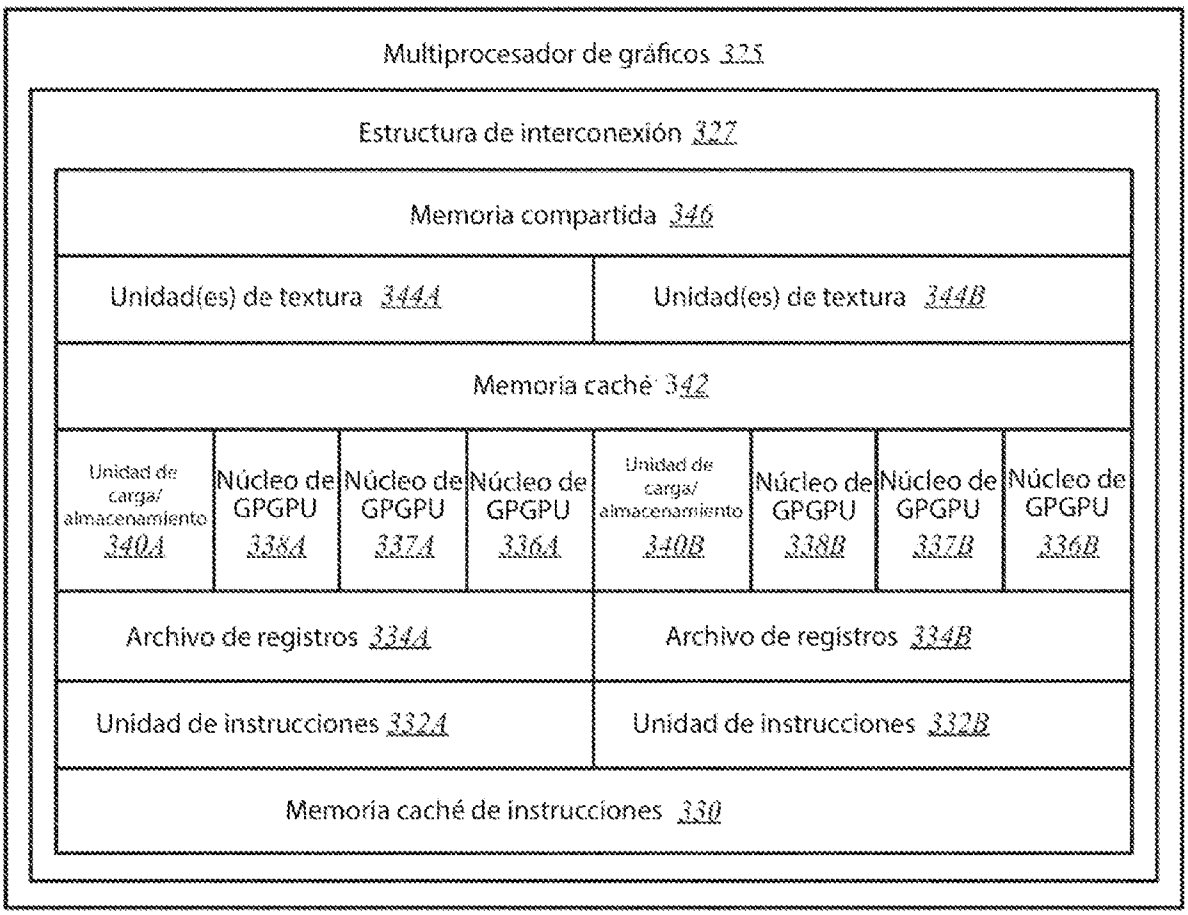

Multiprocesador de gráficos 325
Estructura de interconexión 327
Memoria compartida 346
Unidad(es) de textura 344A
Unidad(es) de textura 344B
Memoria caché 342
Unidad de carga/ almacenamiento 340A
Núcleo de GPGPU 335A
Núcleo de GPGPU 337A
Núcleo de GPGPU 336A
Unidad de carga/ almacenamiento 340B
Núcleo de GPGPU 338B
Núcleo de GPGPU 337B
Núcleo de 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

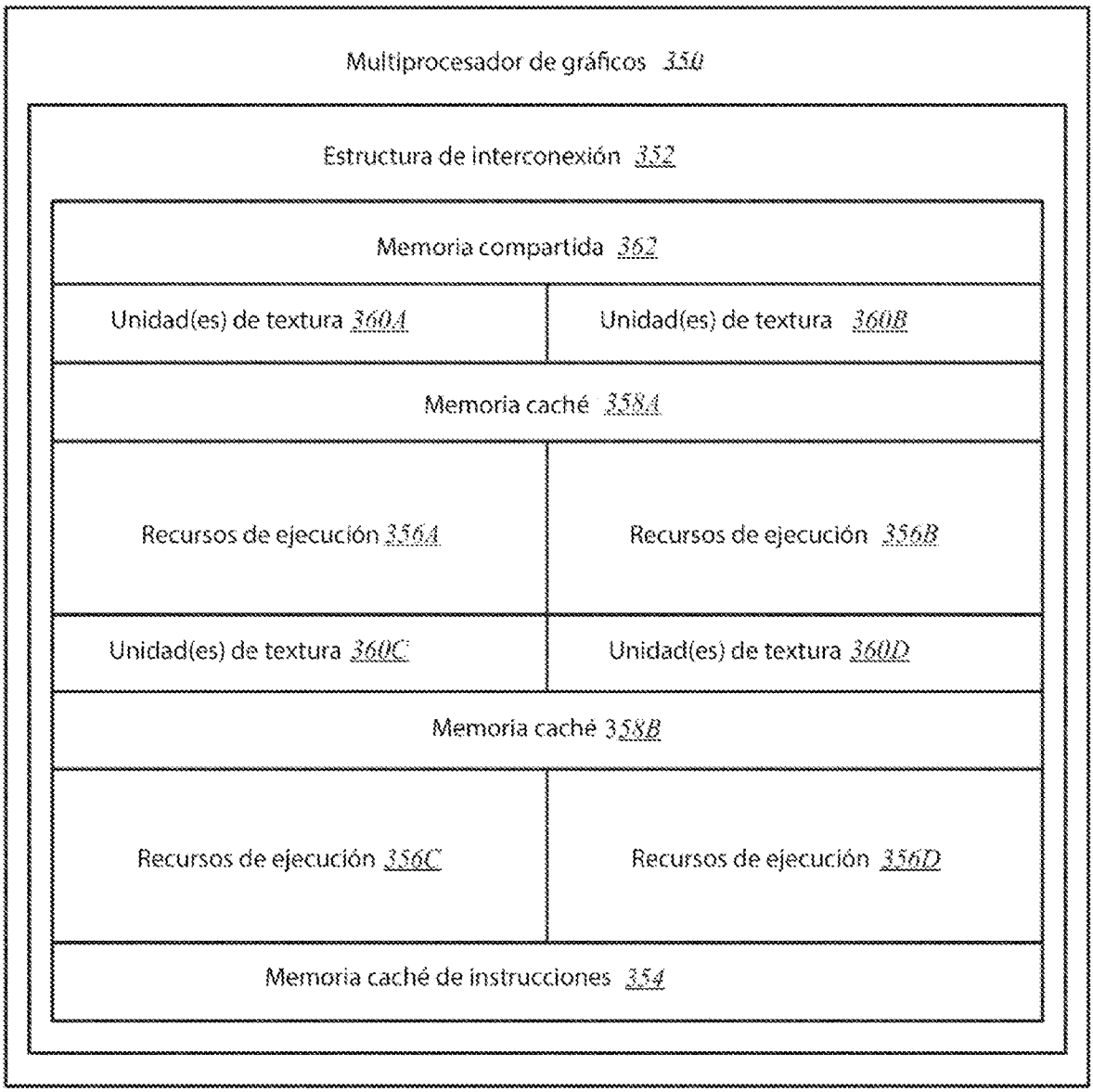

Multiprocesador de gráficos 350
Estructura de interconexión 352
Memoria compartida 362
Unidad(es) de textura 360A
Unidad(es) de textura 360B
Memoria caché 358A
Recursos de ejecución 356A
Recursos de ejecución 356B
Unidad(es) de textura 360C
Unidad(es) de textura 360D
Memoria caché 358B
Recursos de ejecución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

FIG. 3B

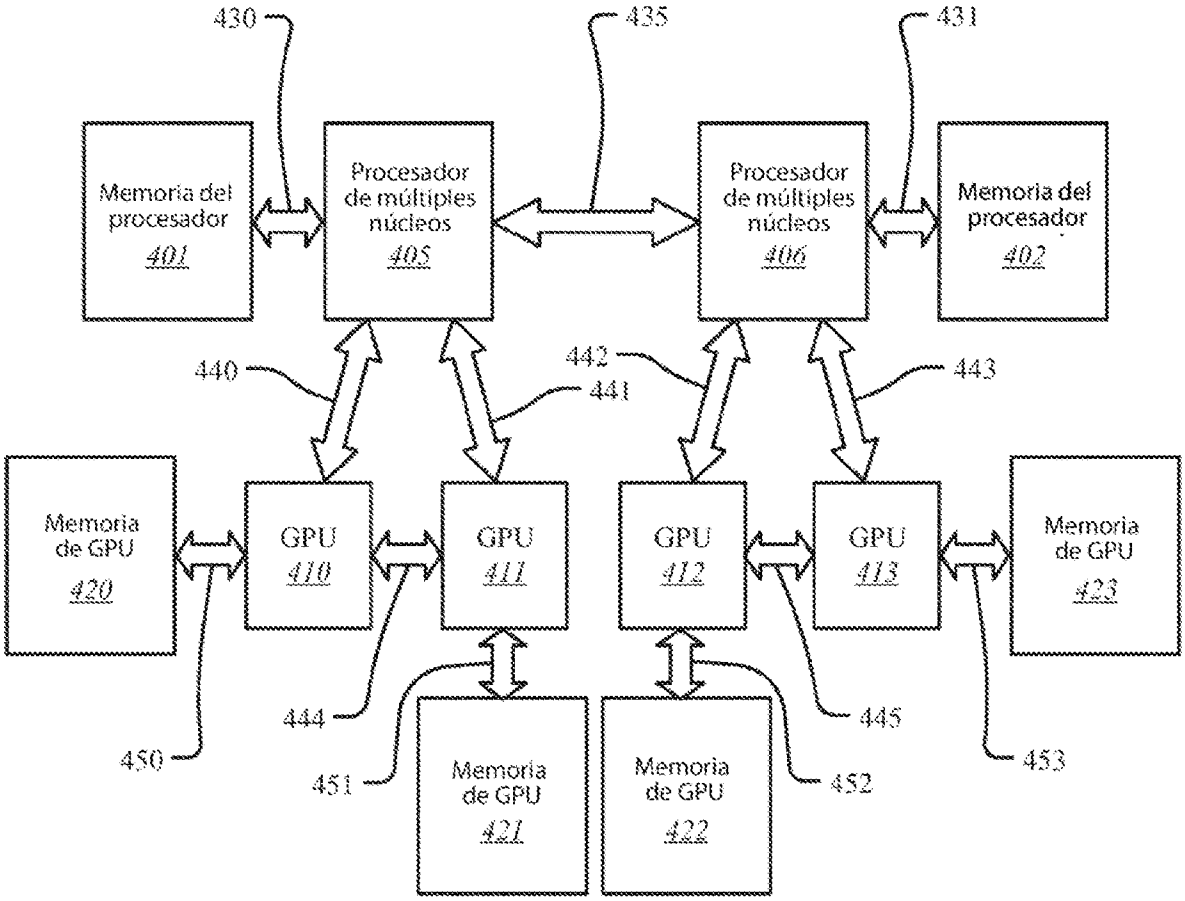

430
435
431
Memoria del procesador 401
Procesador de múltiples núcleos 405
Procesador de múltiples núcleos 406
Memoria del procesador 402
440
441
442
443
Memoria de GPU 420
GPU 410
GPU 411
GPU 412
GPU 413
Memoria de GPU 423
444
445
450
451
452
453
Memoria de GPU 421
Memoria de GPU 422

FIG. 4A

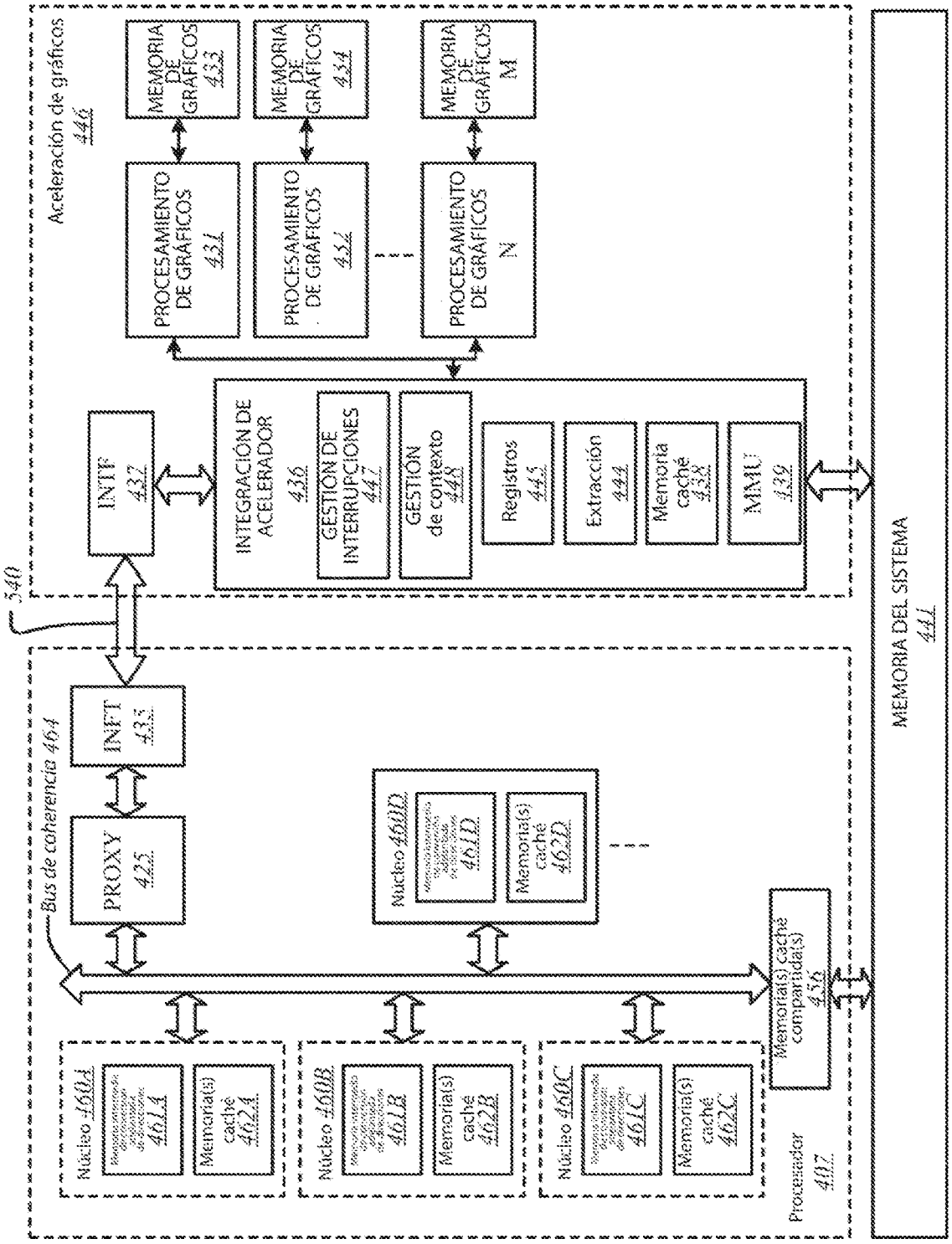

Aceleración de gráficos 446
MEMORIA DE GRÁFICOS 433
MEMORIA DE GRÁFICOS 434
MEMORIA DE GRÁFICOS M
PROCESAMIENTO DE GRÁFICOS 431
PROCESAMIENTO DE GRÁFICOS 432
PROCESAMIENTO DE GRÁFICOS N
INTEGRACIÓN DE ACELERADOR 436
GESTIÓN DE INTERRUPCIONES 447
GESTIÓN de contexto 448
Registros 445
Extracción 444
Memoria caché 438
MMU 439
INTF 437
340
INFT 435
PROXY 425
Bus de coherencia 464
Núcleo 460D
461D
Memoria(s) caché 462D
Núcleo 460A
461A
Memoria(s) caché 462A
Núcleo 460B
461B
Memoria(s) caché 462B
Núcleo 460C
461C
Memoria(s) caché 462C
Memoria(s) caché compartida(s) 456
Procesador 407
MEMORIA DEL SISTEMA 441

FIG. 4B

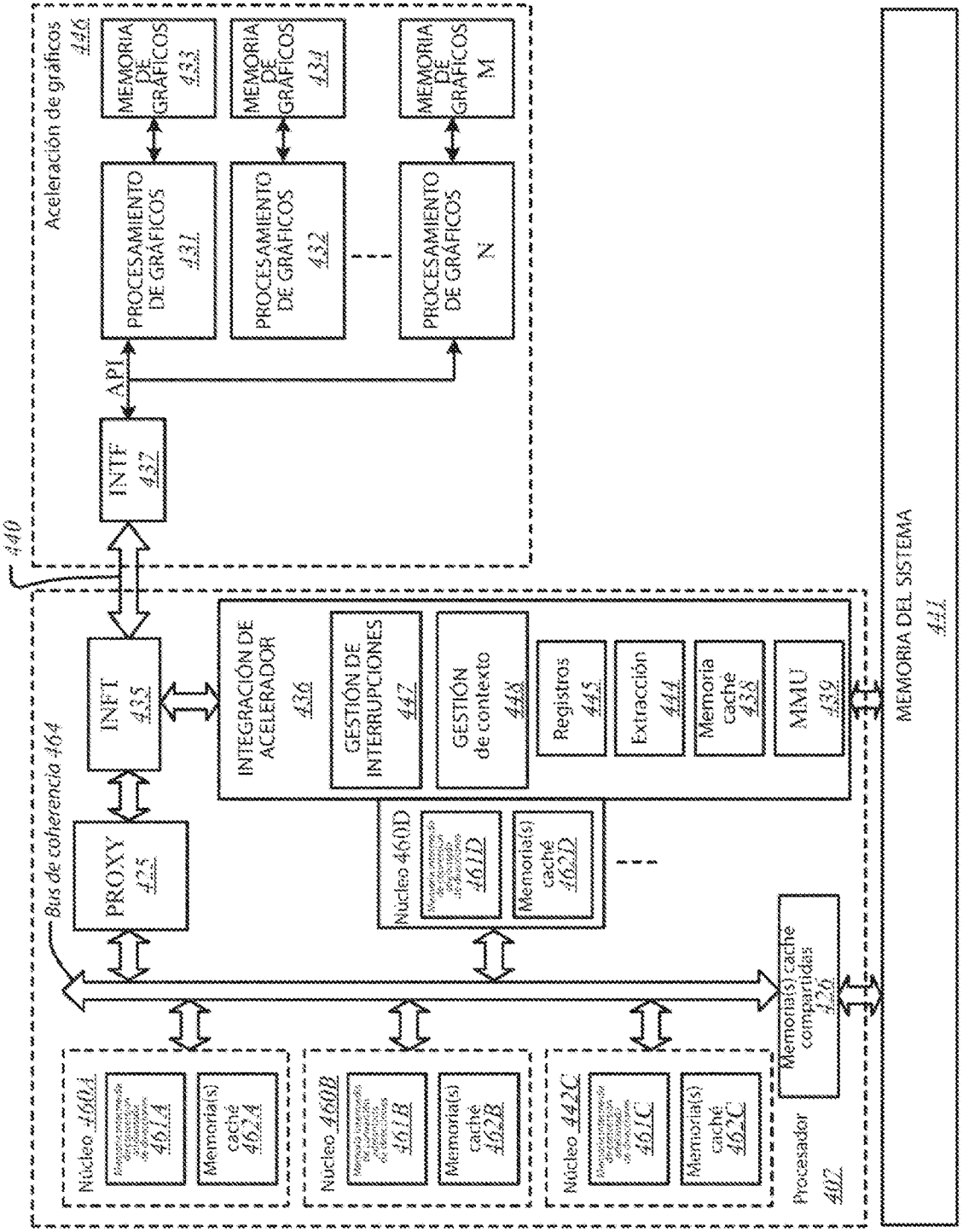

440
Aceleración de gráficos 446
MEMORIA DE GRÁFICOS 433
MEMORIA DE GRÁFICOS 434
MEMORIA DE GRÁFICOS M
PROCESAMIENTO DE GRÁFICOS 431
PROCESAMIENTO DE GRÁFICOS 432
PROCESAMIENTO DE GRÁFICOS N
API
INTF 437
Bus de coherencia 464
INFT 435
INTEGRACIÓN DE ACELERADOR 436
GESTIÓN DE INTERRUPCIONES 447
GESTIÓN de contexto 448
Registros 445
Extracción 444
Memoria caché 438
MMU 439
PROXY 425
Núcleo 460D
461D
Memoria(s) caché 462D
Memoria(s) caché compartidas 426
MEMORIA DEL SISTEMA 441
Núcleo 460A
461A
Memoria(s) caché 462A
Núcleo 460B
461B
Memoria(s) caché 462B
Núcleo 442C
461C
Memoria(s) caché 462C
Procesador 402

FIG. 4C

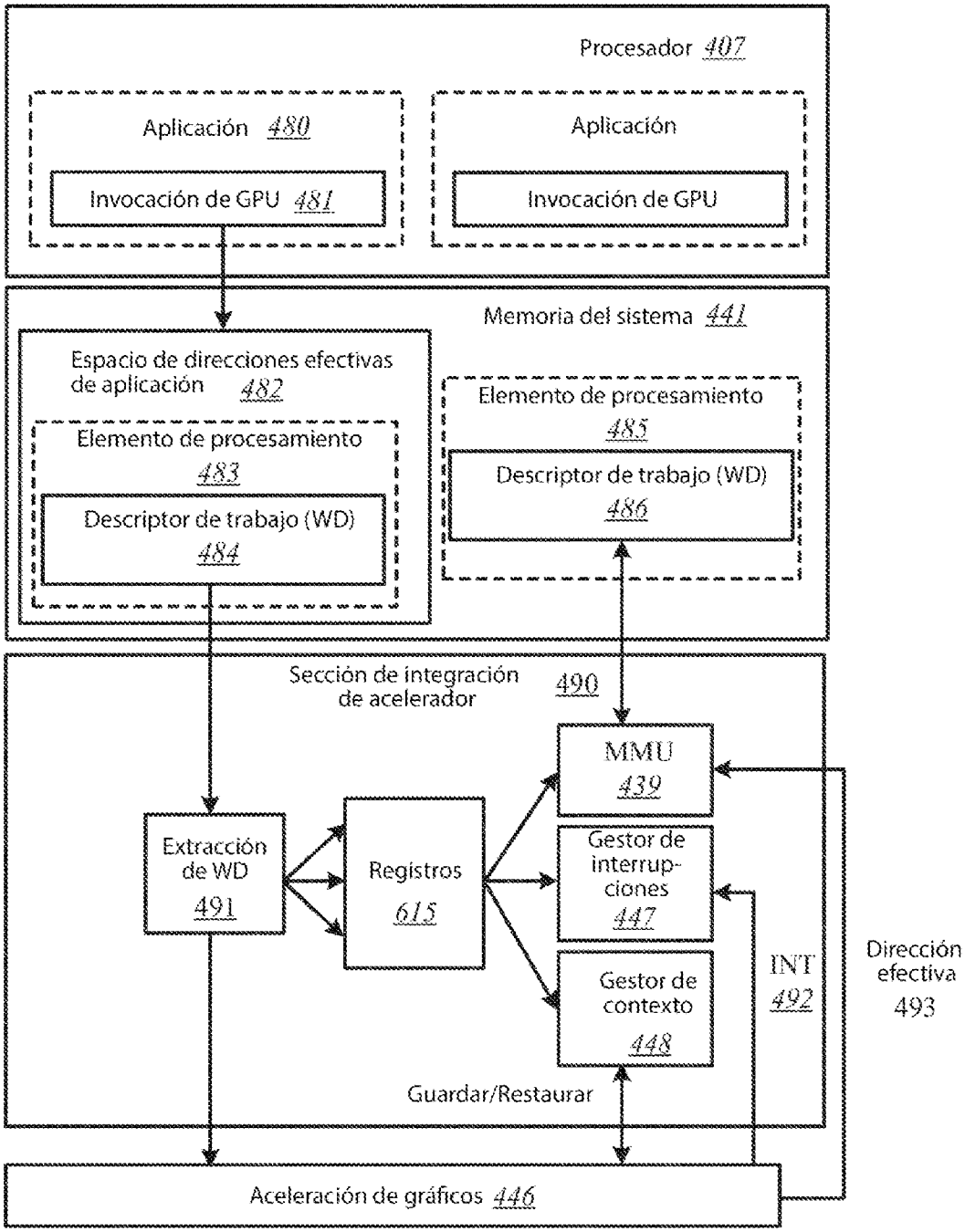

Procesador 407
Aplicación 480
Invocación de GPU 481
Aplicación
Invocación de GPU
Memoria del sistema 441
Espacio de direcciones efectivas de aplicación 482
Elemento de procesamiento 483
Descriptor de trabajo (WD) 484
Elemento de procesamiento 485
Descriptor de trabajo (WD) 486
Sección de integración de acelerador 490
MMU 439
Extracción de WD 491
Registros 615
Gestor de interrup-ciones 447
Gestor de contexto 448
INT 492
Dirección efectiva 493
Guardar/Restaurar
Aceleración de gráficos 446

FIG. 4D

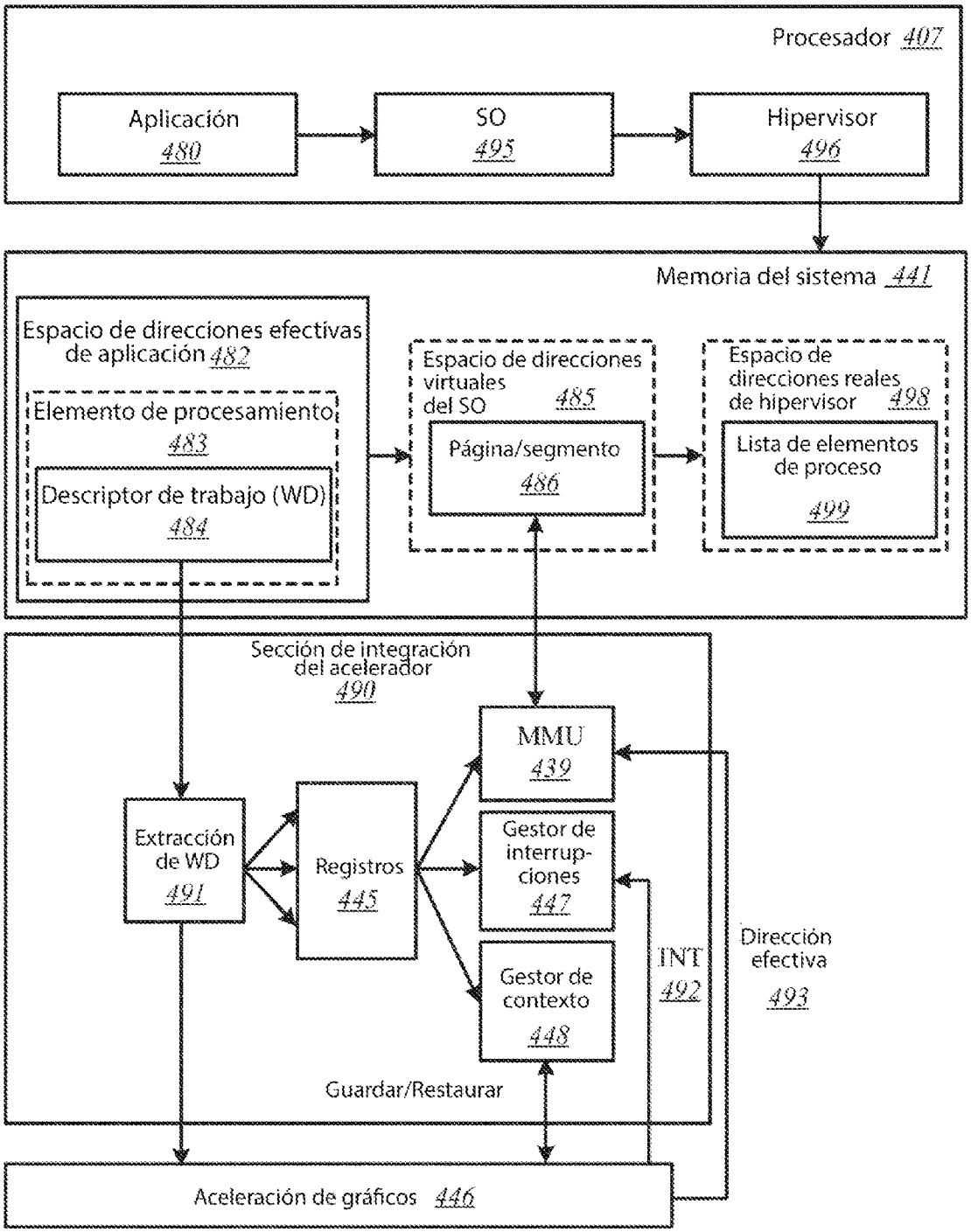

Procesador 407
Aplicación 480
SO 495
Hipervisor 496
Memoria del sistema 441
Espacio de direcciones efectivas de aplicación 482
Elemento de procesamiento 483
Descriptor de trabajo (WD) 484
Espacio de direcciones virtuales del SO 485
Página/segmento 486
Espacio de direcciones reales de hipervisor 498
Lista de elementos de proceso 499
Sección de integración del acelerador 490
MMU 439
Extracción de WD 491
Registros 445
Gestor de interrup-ciones 447
Gestor de contexto 448
INT 492
Dirección efectiva 493
Guardar/Restaurar
Aceleración de gráficos 446

FIG. 4E

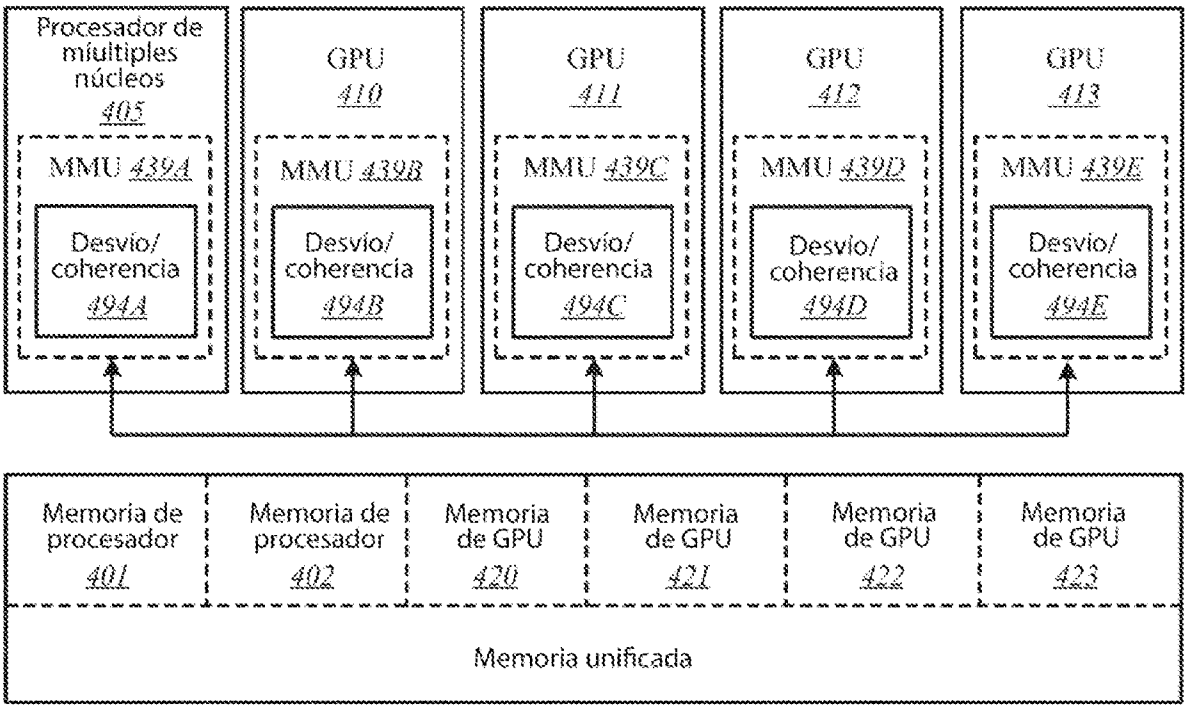

Procesador de múltiples núcleos 405
MMU 439A
Desvío/ coherencia 494A
GPU 410
MMU 439B
Desvío/ coherencia 494B
GPU 411
MMU 439C
Desvío/ coherencia 494C
GPU 412
MMU 439D
Desvío/ coherencia 494D
GPU 413
MMU 439E
Desvío/ coherencia 494E
Memoria de procesador 401
Memoria de procesador 402
Memoria de GPU 420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

FIG. 4F

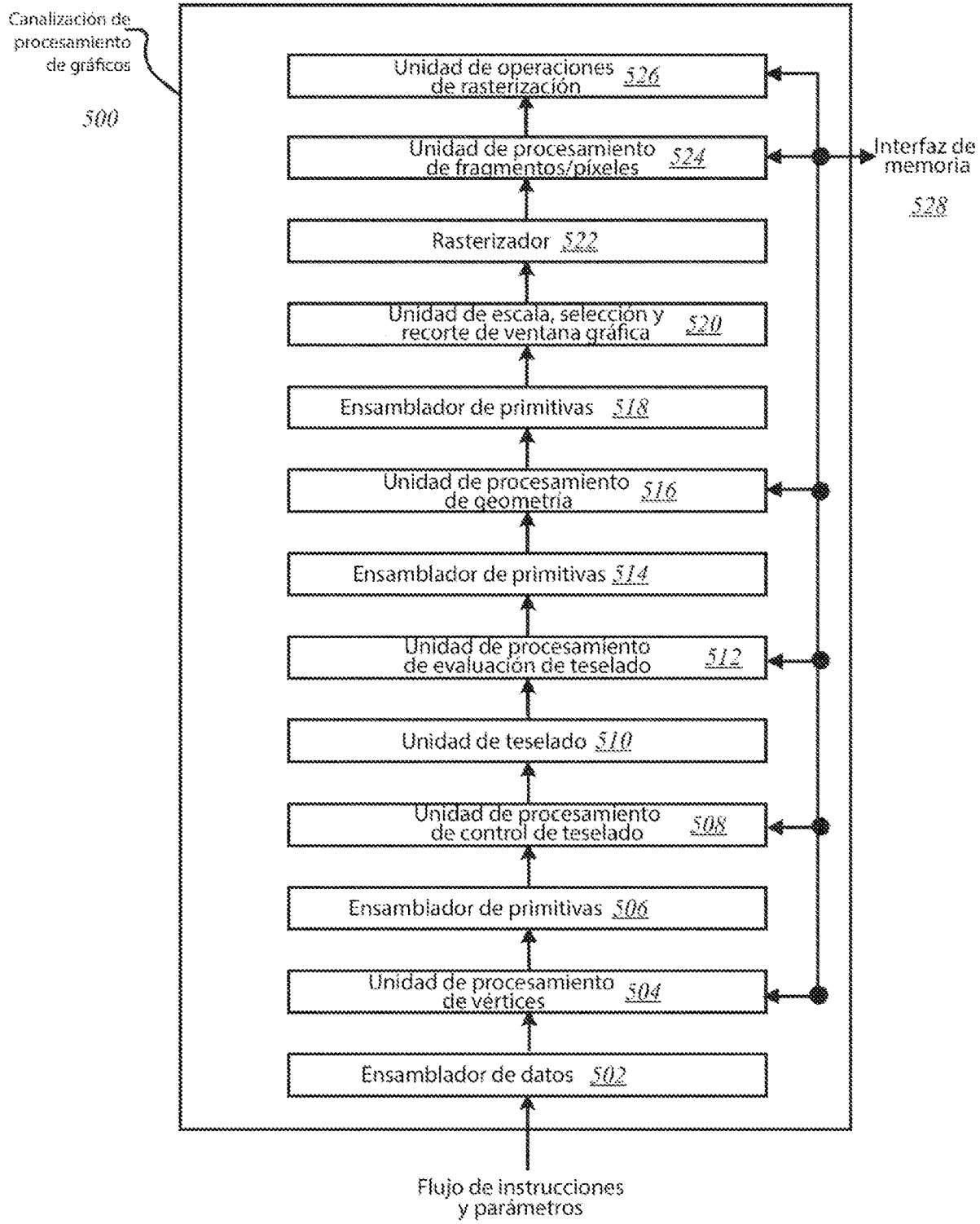

Canalización de procesamiento de gráficos
500
Unidad de operaciones de rasterización 526
Unidad de procesamiento de fragmentos/píxeles 524
Interfaz de memoria
528
Rasterizador 522
Unidad de escala, selección y recorte de ventana gráfica 520
Ensamblador de primitivas 518
Unidad de procesamiento de geometría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

FIG. 5

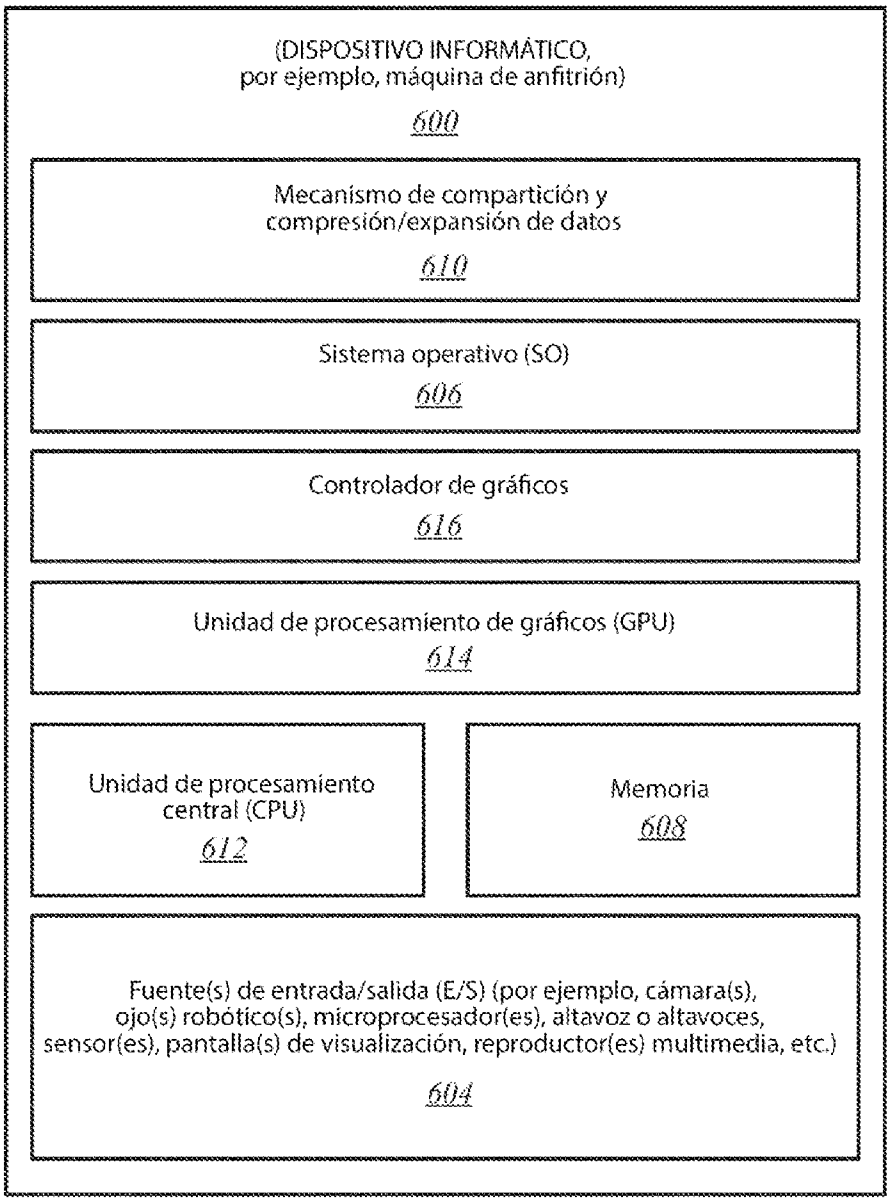

(DISPOSITIVO INFORMÁTICO,
por ejemplo, máquina de anfitrión)
600
Mecanismo de compartición y
compresión/expansión de datos
610
Sistema operativo (SO)
606
Controlador de gráficos
616
Unidad de procesamiento de gráficos (GPU)
614
Unidad de procesamiento
central (CPU)
612
Memoria
608
Fuente(s) de entrada/salida (E/S) (por ejemplo, cámara(s),
ojo(s) robótico(s), microprocesador(es), altavoz o altavoces,
sensor(es), pantalla(s) de visualización, reproductor(es) multimedia, etc.)
604

FIG. 6

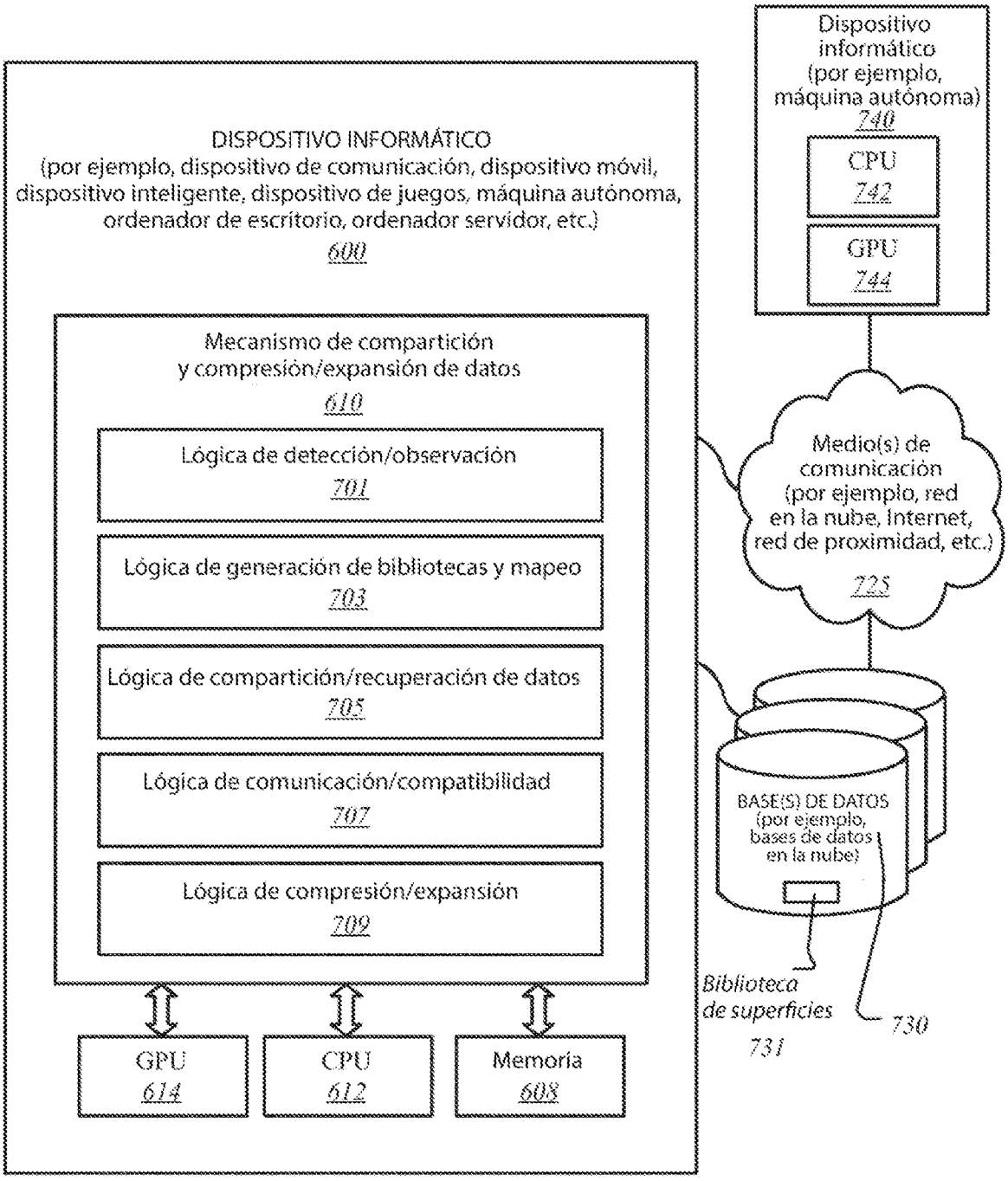

Dispositivo informático (por ejemplo, máquina autónoma)
740
CPU
742
GPU
744
DISPOSITIVO INFORMÁTICO
(por ejemplo, dispositivo de comunicación, dispositivo móvil, dispositivo inteligente, dispositivo de juegos, máquina autónoma, ordenador de escritorio, ordenador servidor, etc.)
600
Mecanismo de compartición y compresión/expansión de datos
610
Lógica de detección/observación
701
Lógica de generación de bibliotecas y mapeo
703
Lógica de compartición/recuperación de datos
705
Lógica de comunicación/compatibilidad
707
Lógica de compresión/expansión
709
GPU
614
CPU
612
Memoria
608
Medio(s) de comunicación (por ejemplo, red en la nube, Internet, red de proximidad, etc.)
725
BASE(S) DE DATOS (por ejemplo, bases de datos en la nube)
Biblioteca de superficies
731
730

FIG. 7

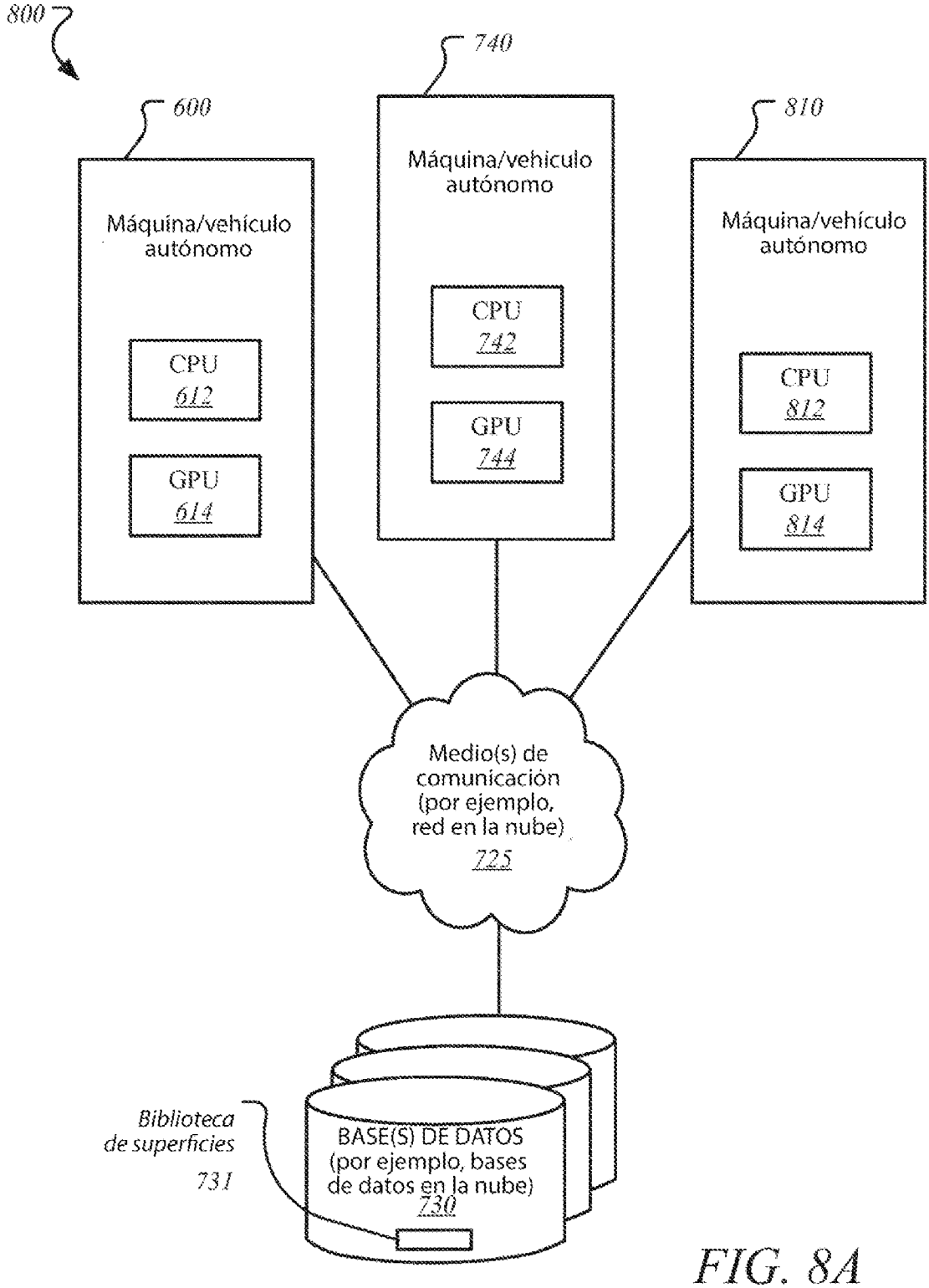

800
740
600
810
Máquina/vehículo autónomo
CPU 612
GPU 614
Máquina/vehículo autónomo
CPU 742
GPU 744
Máquina/vehículo autónomo
CPU 812
GPU 814
Medio(s) de comunicación (por ejemplo, red en la nube) 725
Biblioteca de superficies 731
BASE(S) DE DATOS (por ejemplo, bases de datos en la nube) 730

FIG. 8A

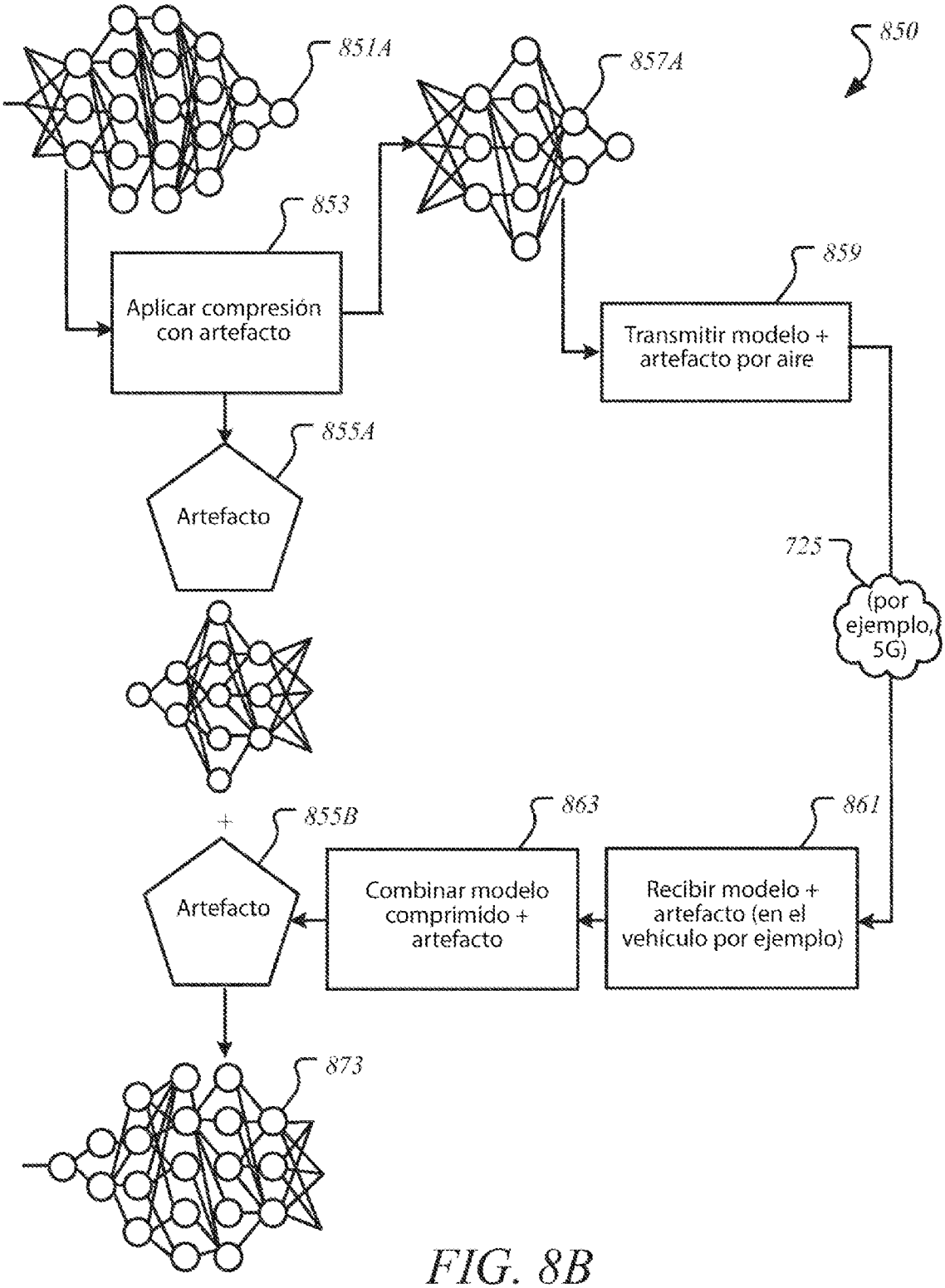

851A
857A
850
853
Aplicar compresión con artefacto
859
Transmitir modelo + artefacto por aire
855A
Artefacto
725
(por ejemplo, 5G)
+
855B
863
861
Artefacto
Combinar modelo comprimido + artefacto
Recibir modelo + artefacto (en el vehículo por ejemplo)
873

FIG. 8B

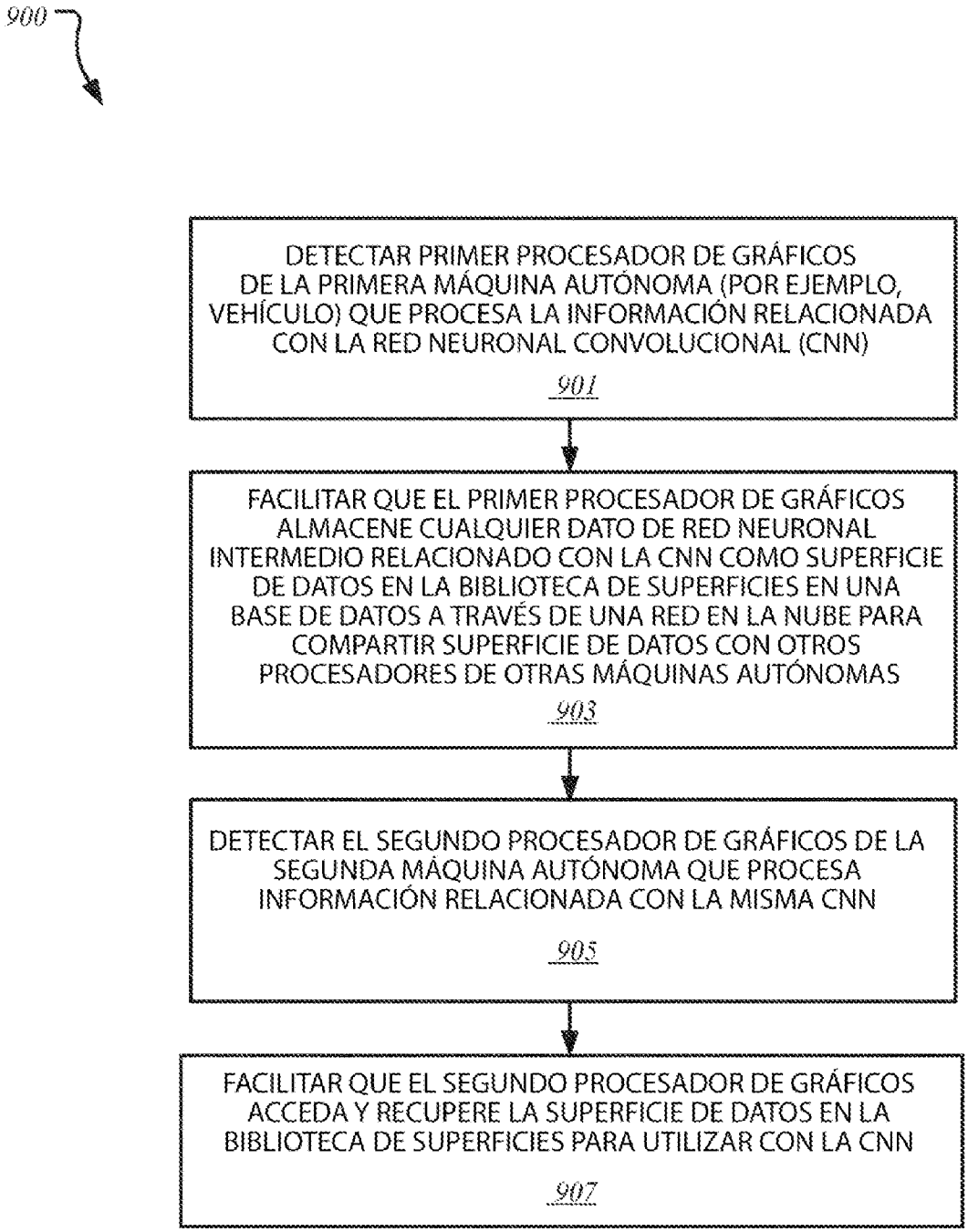

900
DETECTAR PRIMER PROCESADOR DE GRÁFICOS DE LA PRIMERA MÁQUINA AUTÓNOMA (POR EJEMPLO, VEHÍCULO) QUE PROCESA LA INFORMACIÓN RELACIONADA CON LA RED NEURONAL CONVOLUCIONAL (CNN)
901
FACILITAR QUE EL PRIMER PROCESADOR DE GRÁFICOS ALMACENE CUALQUIER DATO DE RED NEURONAL INTERMEDIO RELACIONADO CON LA CNN COMO SUPERFICIE DE DATOS EN LA BIBLIOTECA DE SUPERFICIES EN UNA BASE DE DATOS A TRAVÉS DE UNA RED EN LA NUBE PARA COMPARTIR SUPERFICIE DE DATOS CON OTROS PROCESADORES DE OTRAS MÁQUINAS AUTÓNOMAS
903
DETECTAR EL SEGUNDO PROCESADOR DE GRÁFICOS DE LA SEGUNDA MÁQUINA AUTÓNOMA QUE PROCESA INFORMACIÓN RELACIONADA CON LA MISMA CNN
905
FACILITAR QUE EL SEGUNDO PROCESADOR DE GRÁFICOS ACCEDA Y RECUPERE LA SUPERFICIE DE DATOS EN LA BIBLIOTECA DE SUPERFICIES PARA UTILIZAR CON LA CNN
907

FIG. 9

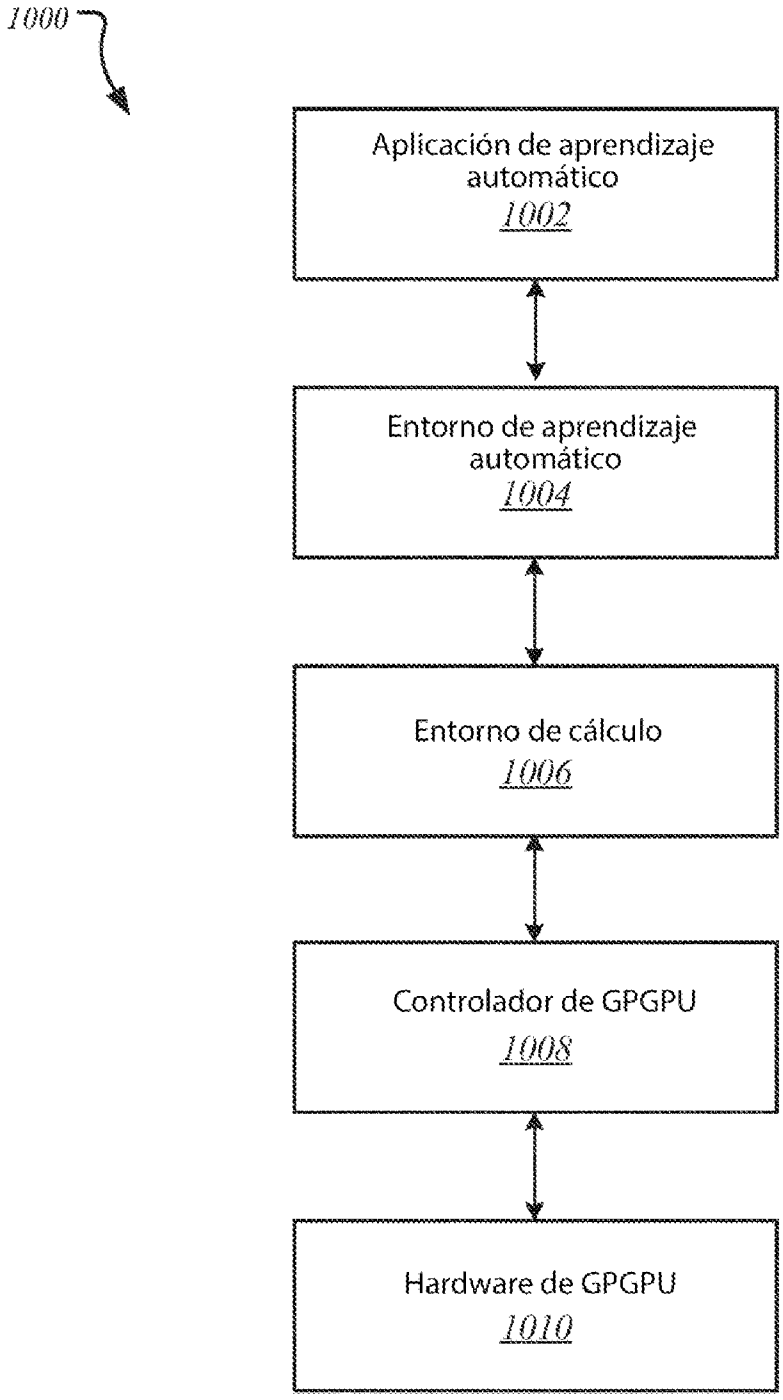

1000
Aplicación de aprendizaje automático
1002
Entorno de aprendizaje automático
1004
Entorno de cálculo
1006
Controlador de GPGPU
1008
Hardware de GPGPU
1010

FIG. 10

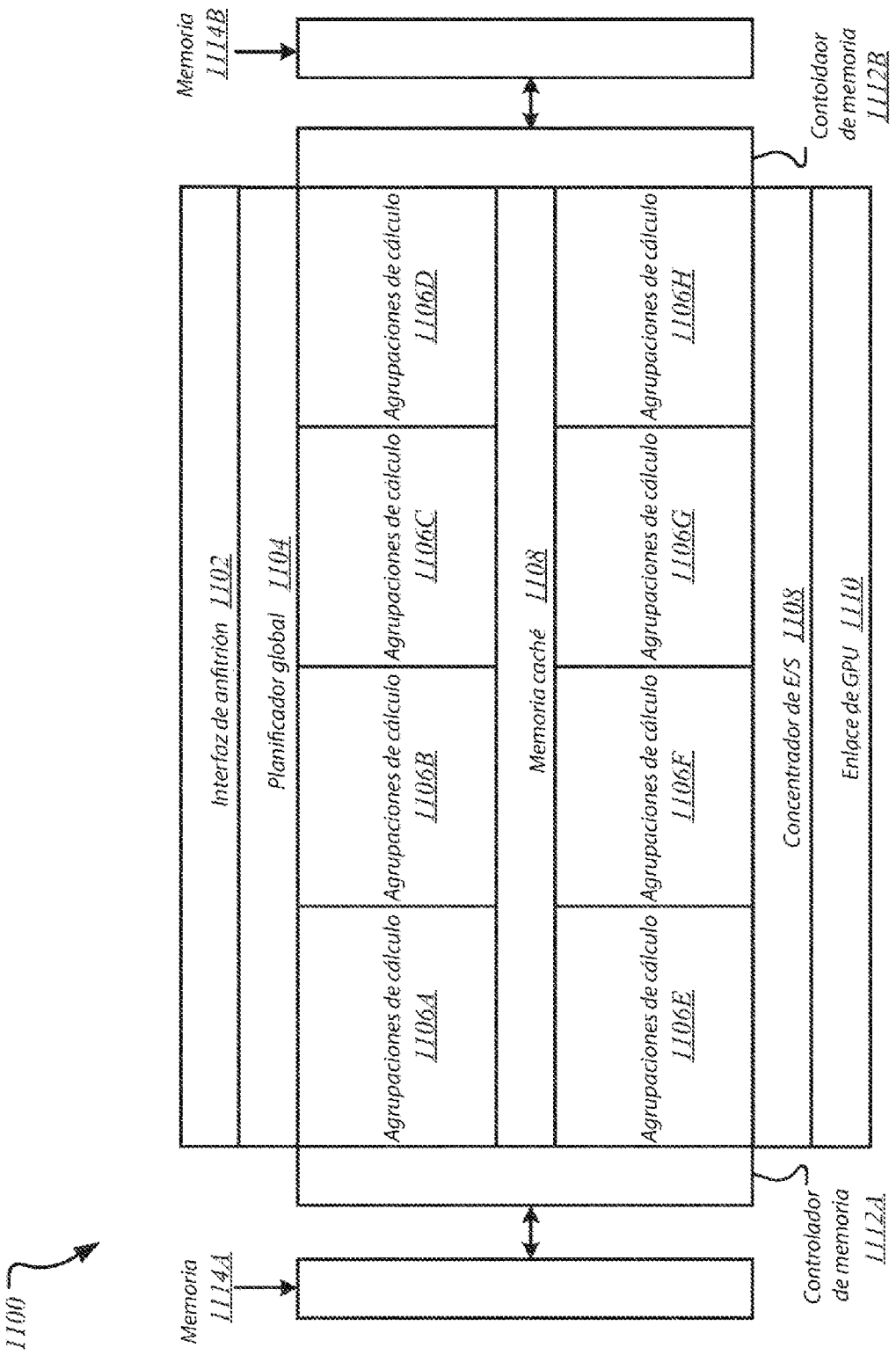

1100
Memoria 1114A
Memoria 1114B
Controlador de memoria 1112A
Contolador de memoria 1112B
Interfaz de anfitrión 1102
Planificador global 1104
Agrupaciones de cálculo 1106A
Agrupaciones de cálculo 1106B
Agrupaciones de cálculo 1106C
Agrupaciones de cálculo 1106D
Agrupaciones de cálculo 1106E
Agrupaciones de cálculo 1106F
Agrupaciones de cálculo 1106G
Agrupaciones de cálculo 1106H
Memoria caché 1108
Concentrador de E/S 1108
Enlace de GPU 1110

FIG. 11

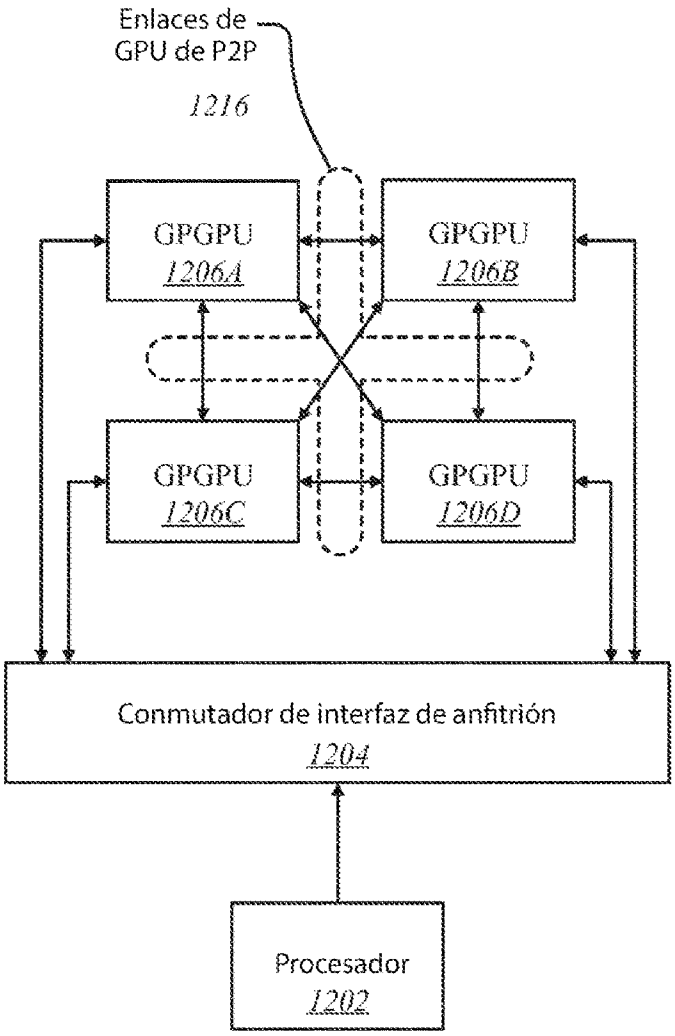

Enlaces de
GPU de P2P
1216
GPGPU
1206A
GPGPU
1206B
GPGPU
1206C
GPGPU
1206D
Conmutador de interfaz de anfitrión
1204
Procesador
1202

FIG. 12

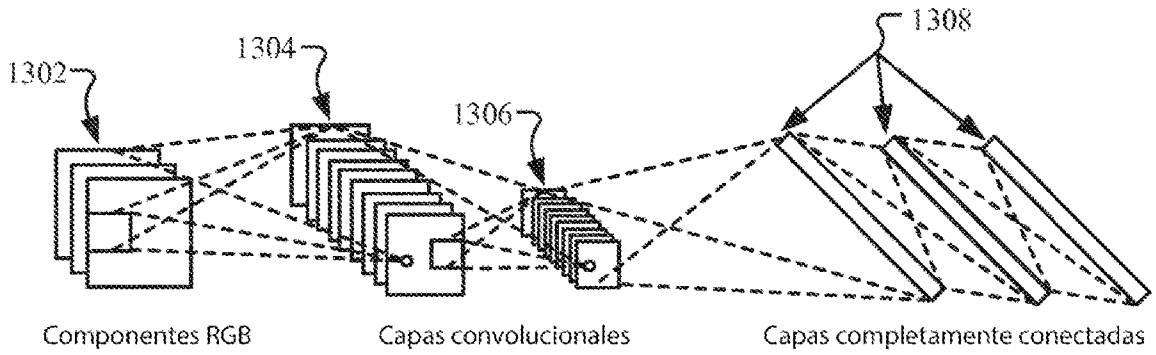


FIG. 13A

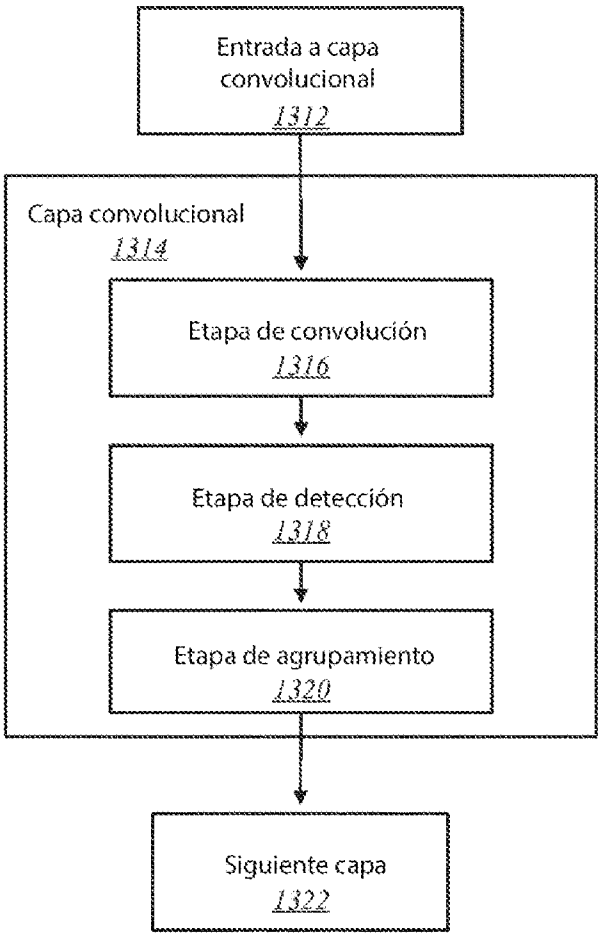


FIG. 13B

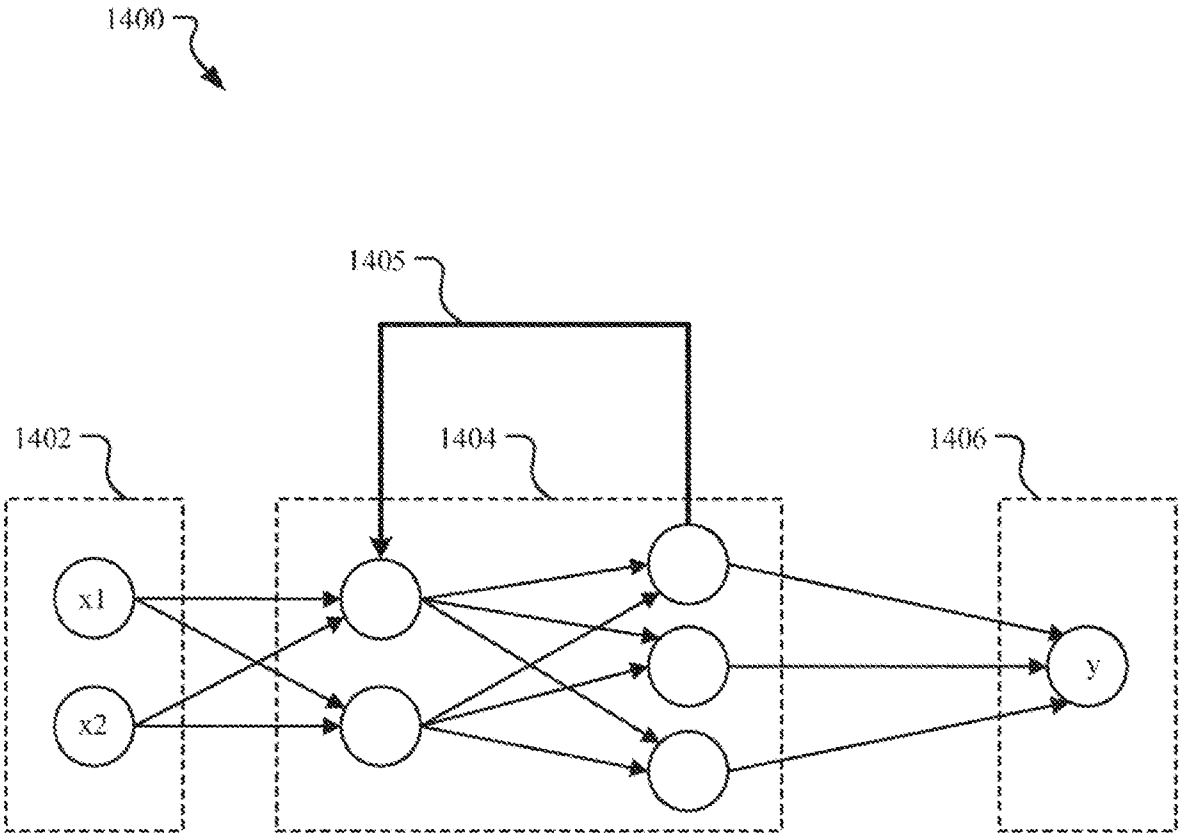


FIG. 14

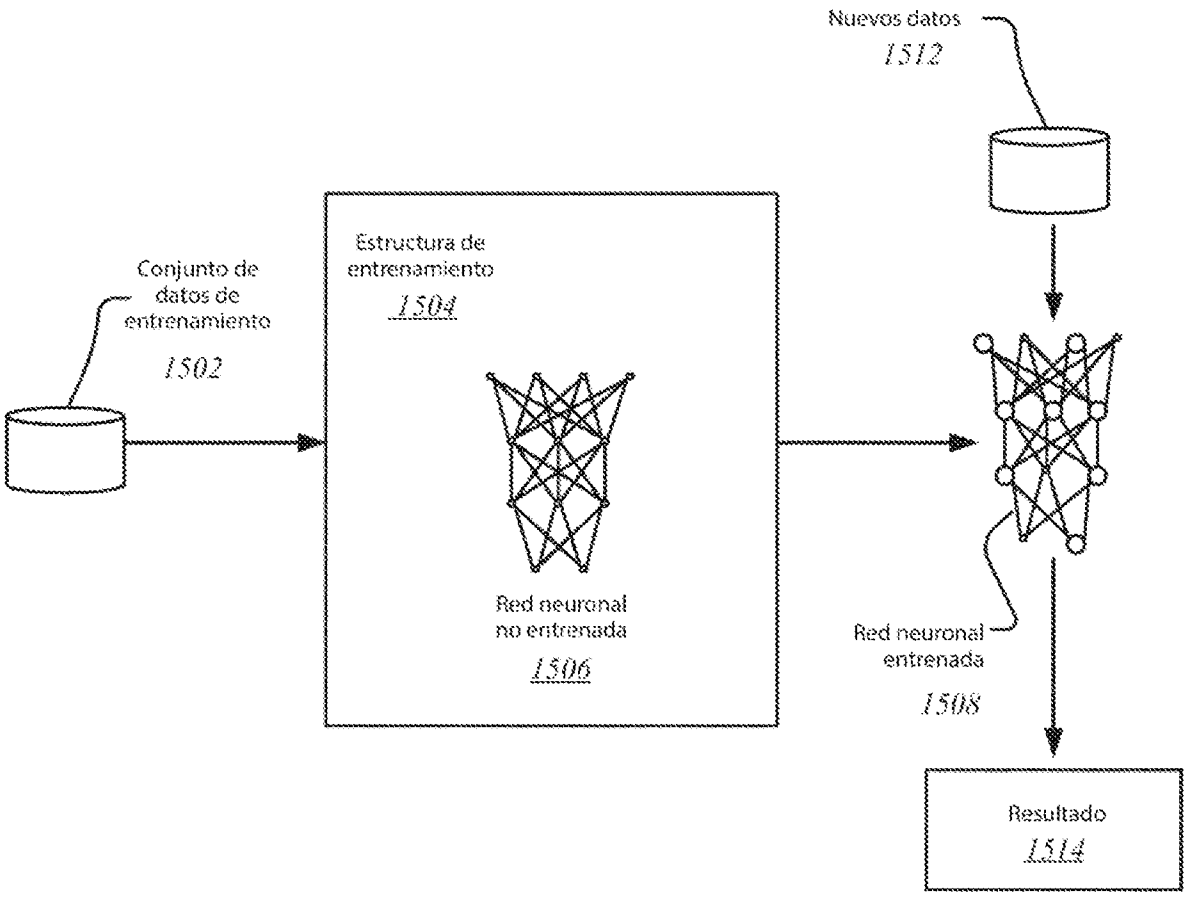

Nuevos datos
1512
Conjunto de datos de entrenamiento
1502
Estructura de entrenamiento
1504
Red neuronal no entrenada
1506
Red neuronal entrenada
1508
Resultado
1514

FIG. 15

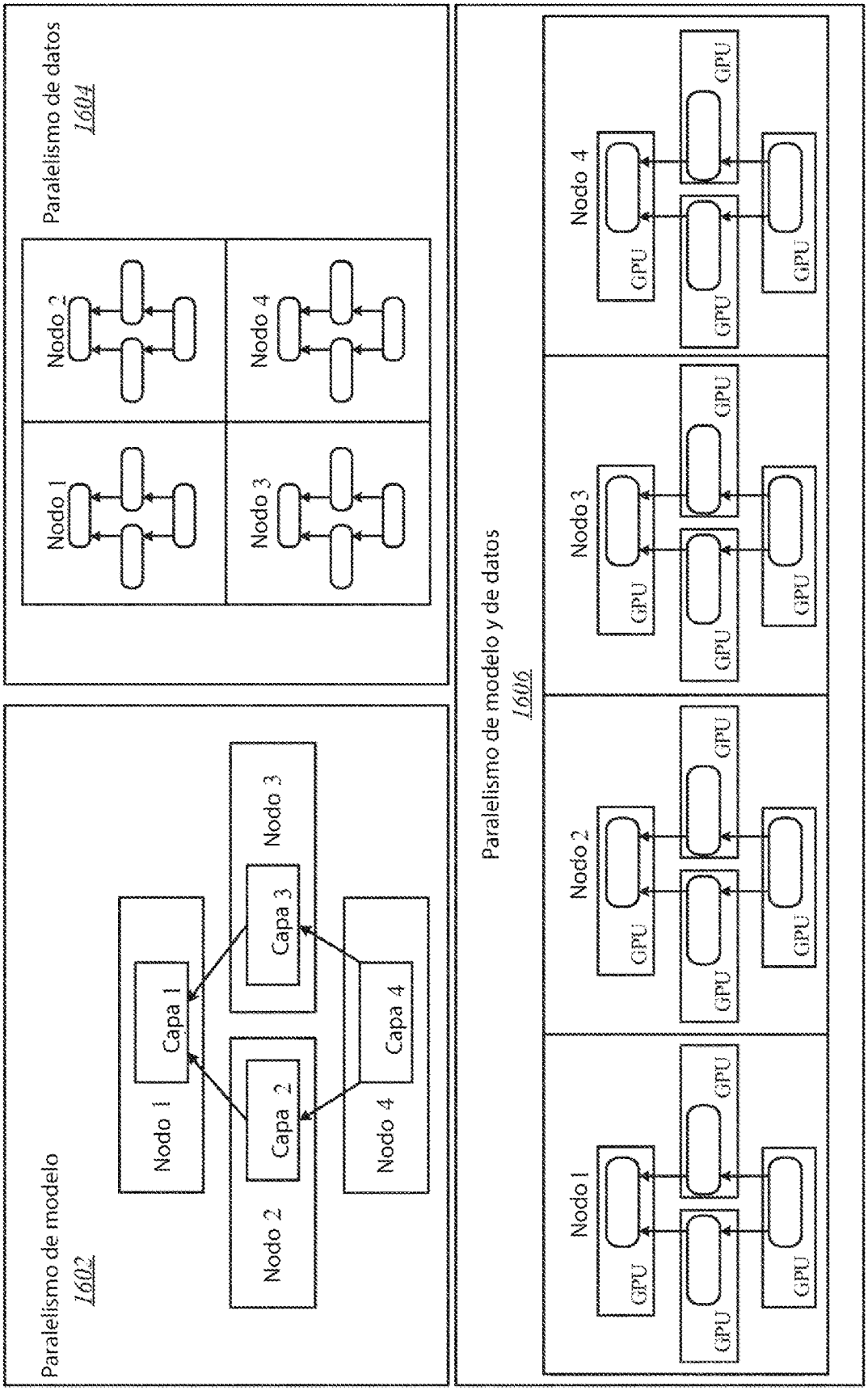

Paralelismo de modelo
1602
Nodo 1
Capa 1
Nodo 2
Capa 2
Nodo 3
Capa 3
Nodo 4
Capa 4
Paralelismo de datos
1604
Nodo 1
Nodo 2
Nodo 3
Nodo 4
Paralelismo de modelo y de datos
1606
GPU

FIG. 16

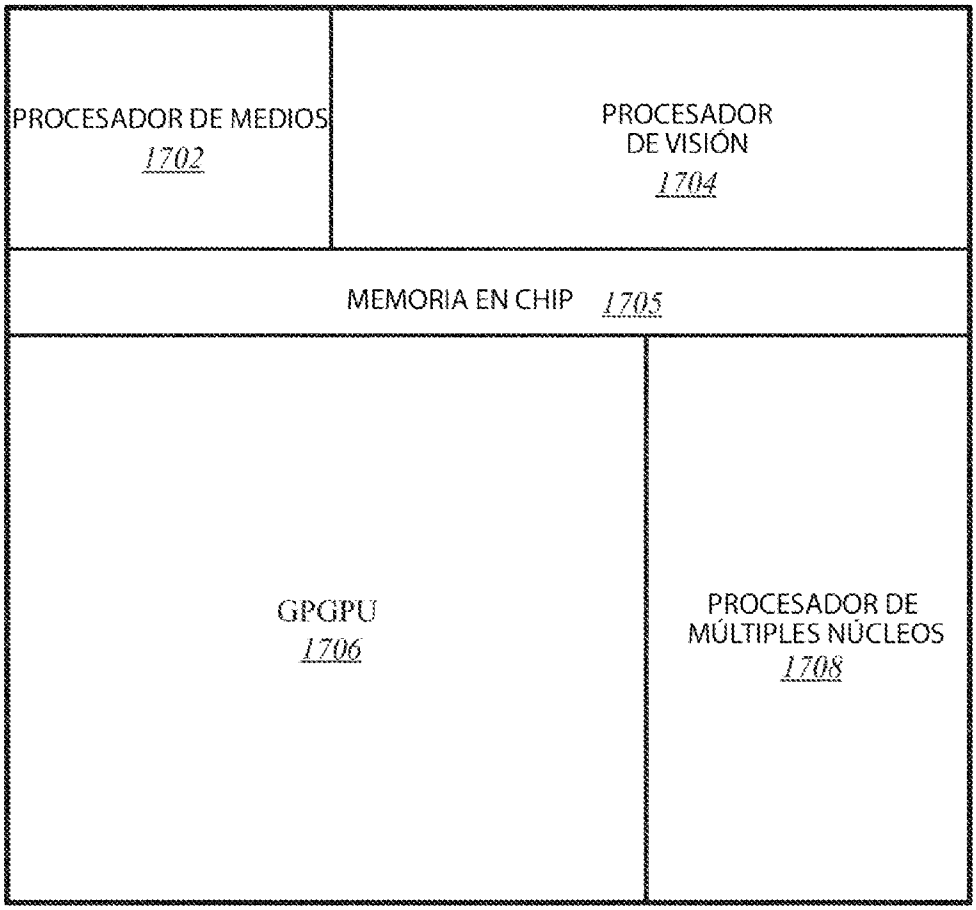

PROCESADOR DE MEDIOS
1702
PROCESADOR
DE VISIÓN
1704
MEMORIA EN CHIP 1705
GPGPU
1706
PROCESADOR DE
MÚLTIPLES NÚCLEOS
1708

FIG. 17

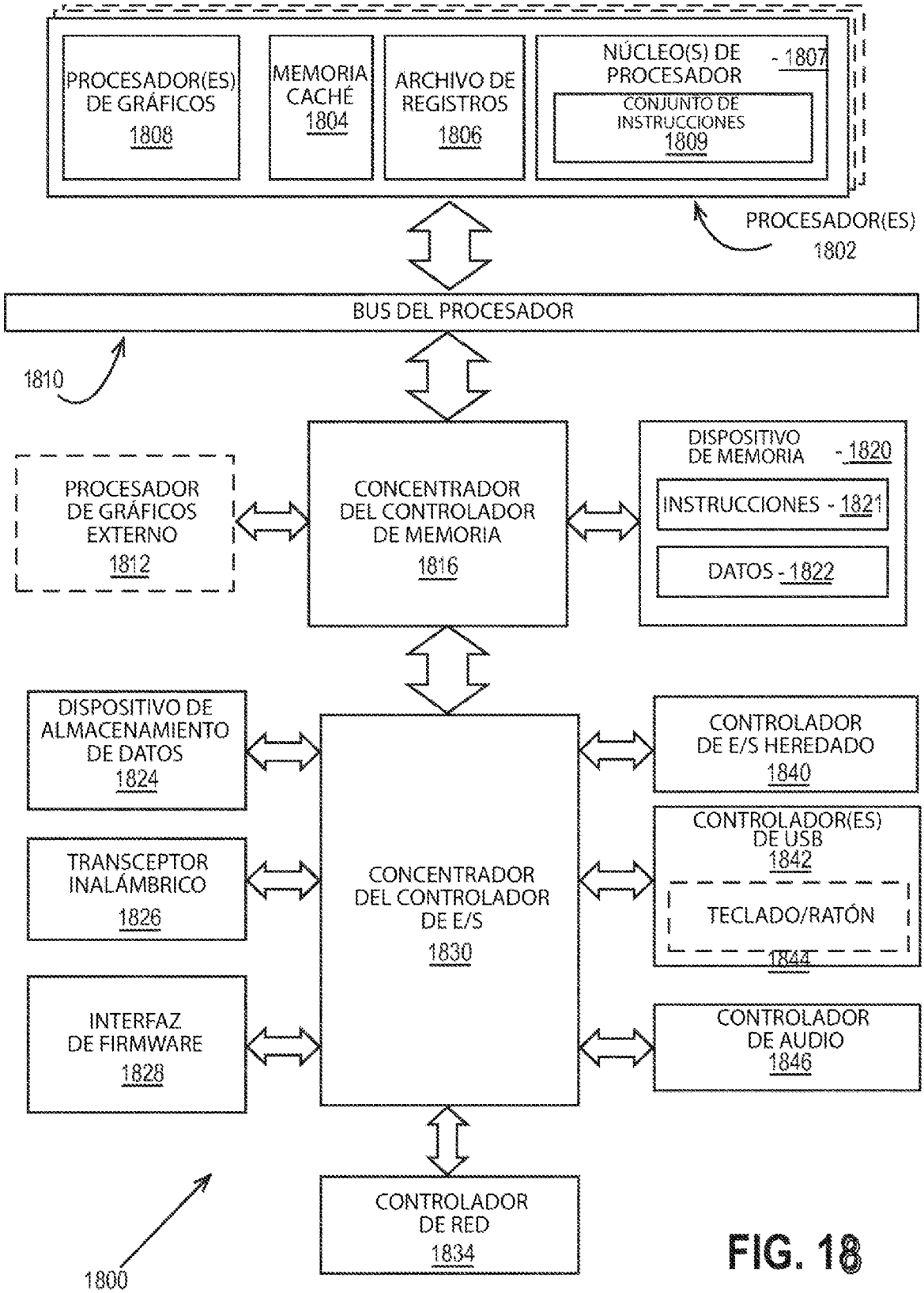

PROCESADOR(ES) DE GRÁFICOS 1808
MEMORIA CACHÉ 1804
ARCHIVO DE REGISTROS 1806
NÚCLEO(S) DE PROCESADOR 1807
CONJUNTO DE INSTRUCCIONES 1809
PROCESADOR(ES) 1802
BUS DEL PROCESADOR
1810
PROCESADOR DE GRÁFICOS EXTERNO 1812
CONCENTRADOR DEL CONTROLADOR DE MEMORIA 1816
DISPOSITIVO DE MEMORIA 1820
INSTRUCCIONES - 1821
DATOS - 1822
DISPOSITIVO DE ALMACENAMIENTO DE DATOS 1824
TRANSCEPTOR INALÁMBRICO 1826
INTERFAZ DE FIRMWARE 1828
CONCENTRADOR DEL CONTROLADOR DE E/S 1830
CONTROLADOR DE E/S HEREDADO 1840
CONTROLADOR(ES) DE USB 1842
TECLADO/RATÓN 1844
CONTROLADOR DE AUDIO 1846
CONTROLADOR DE RED 1834
1800

FIG. 18

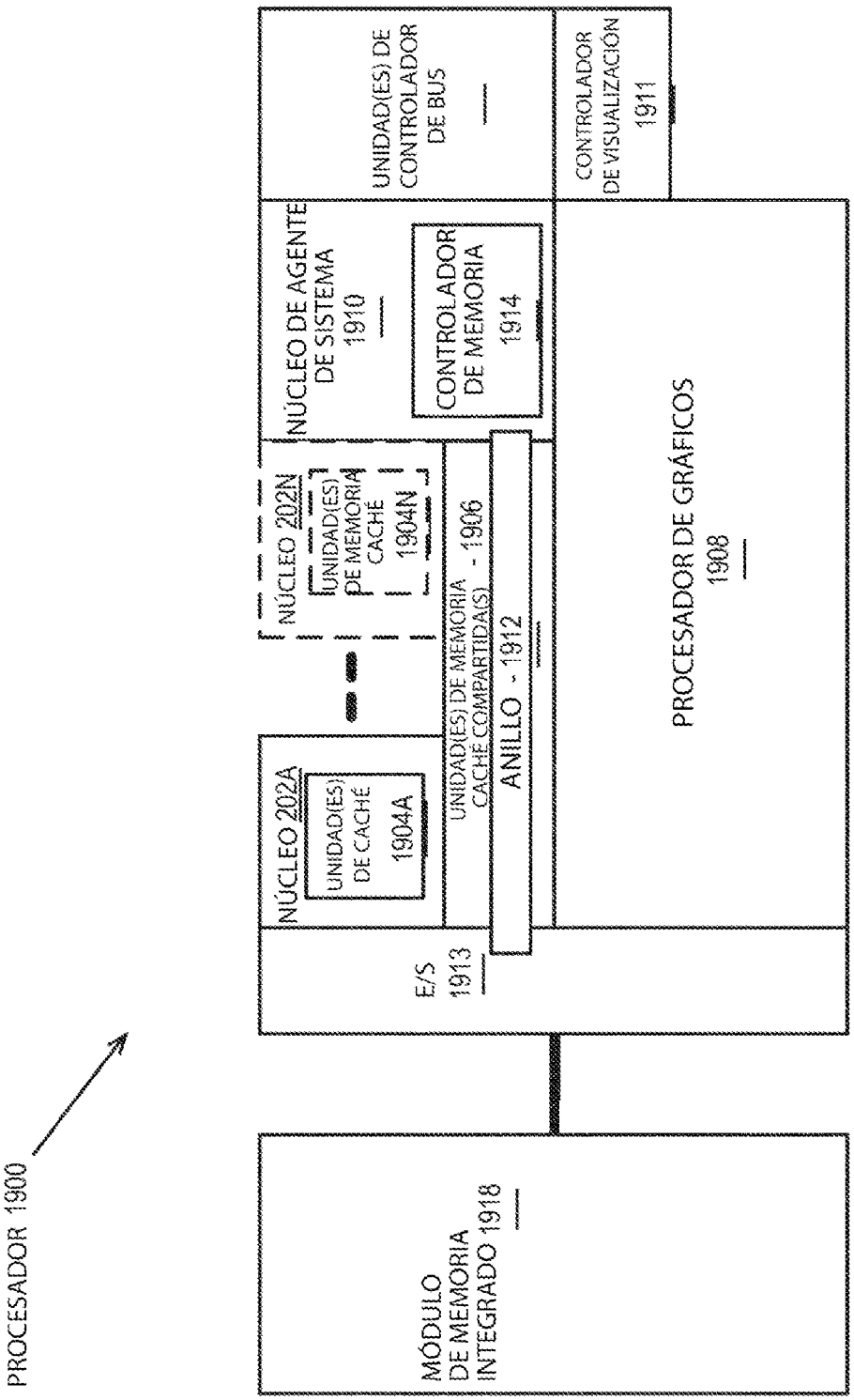

PROCESADOR 1900
MÓDULO DE MEMORIA INTEGRADO 1918
E/S 1913
NÚCLEO 202A
UNIDAD(ES) DE CACHÉ 1904A
NÚCLEO 202N
UNIDAD(ES) DE MEMORIA CACHÉ 1904N
UNIDAD(ES) DE MEMORIA CACHÉ COMPARTIDA(S) - 1906
ANILLO - 1912
NÚCLEO DE AGENTE DE SISTEMA 1910
CONTROLADOR DE MEMORIA 1914
PROCESADOR DE GRÁFICOS 1908
UNIDAD(ES) DE CONTROLADOR DE BUS 1913
CONTROLADOR DE VISUALIZACIÓN 1911

FIG. 19

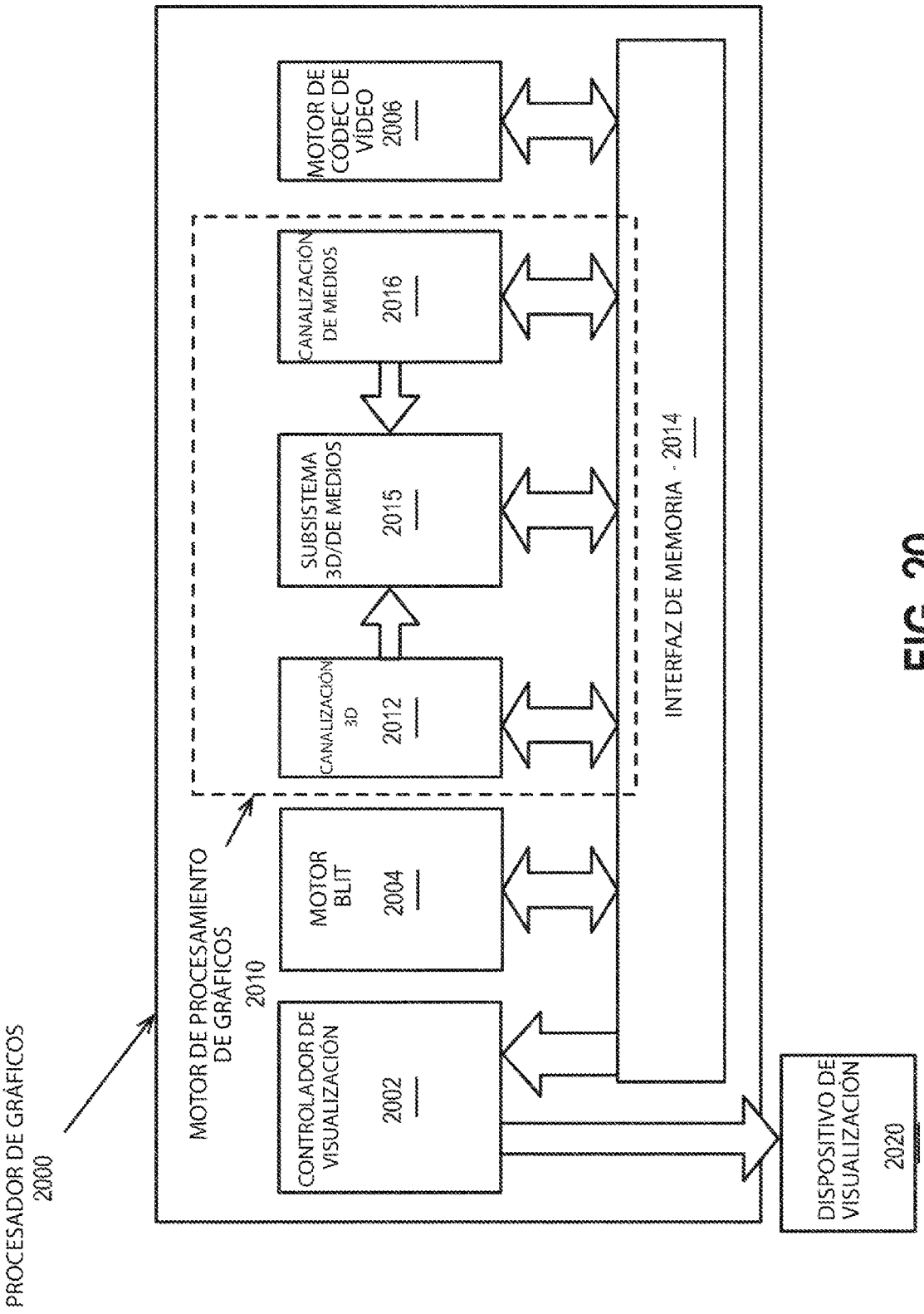

PROCESADOR DE GRÁFICOS
2000
MOTOR DE PROCESAMIENTO DE GRÁFICOS
2010
CONTROLADOR DE VISUALIZACIÓN
2002
MOTOR BLIT
2004
CANALIZACIÓN 3D
2012
SUBSISTEMA 3D/DE MEDIOS
2015
CANALIZACIÓN DE MEDIOS
2016
MOTOR DE CÓDEC DE VÍDEO
2006
INTERFAZ DE MEMORIA - 2014
DISPOSITIVO DE VISUALIZACIÓN
2020

FIG. 20

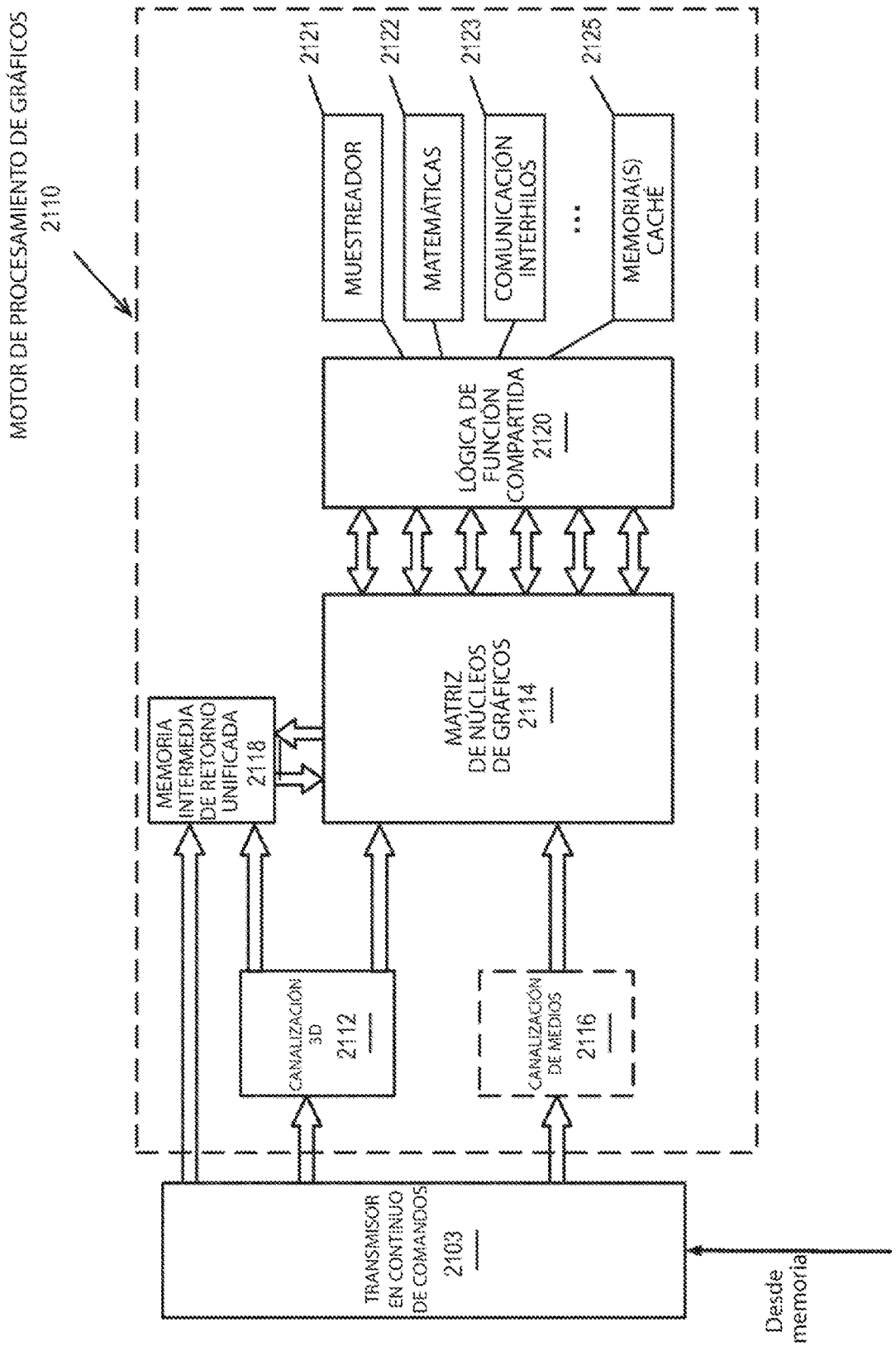

MOTOR DE PROCESAMIENTO DE GRÁFICOS
2110
2121
2122
2123
2125
MUESTREADOR
MATEMÁTICAS
COMUNICACIÓN INTERHILOS
...
MEMORIA(S) CACHÉ
LÓGICA DE FUNCIÓN COMPARTIDA
2120
MATRIZ DE NÚCLEOS DE GRÁFICOS
2114
MEMORIA INTERMEDIA DE RETORNO UNIFICADA
2118
CANALIZACIÓN 3D
2112
CANALIZACIÓN DE MEDIOS
2116
TRANSMISOR EN CONTINUO DE COMANDOS
2103
Desde memoria

FIG. 21

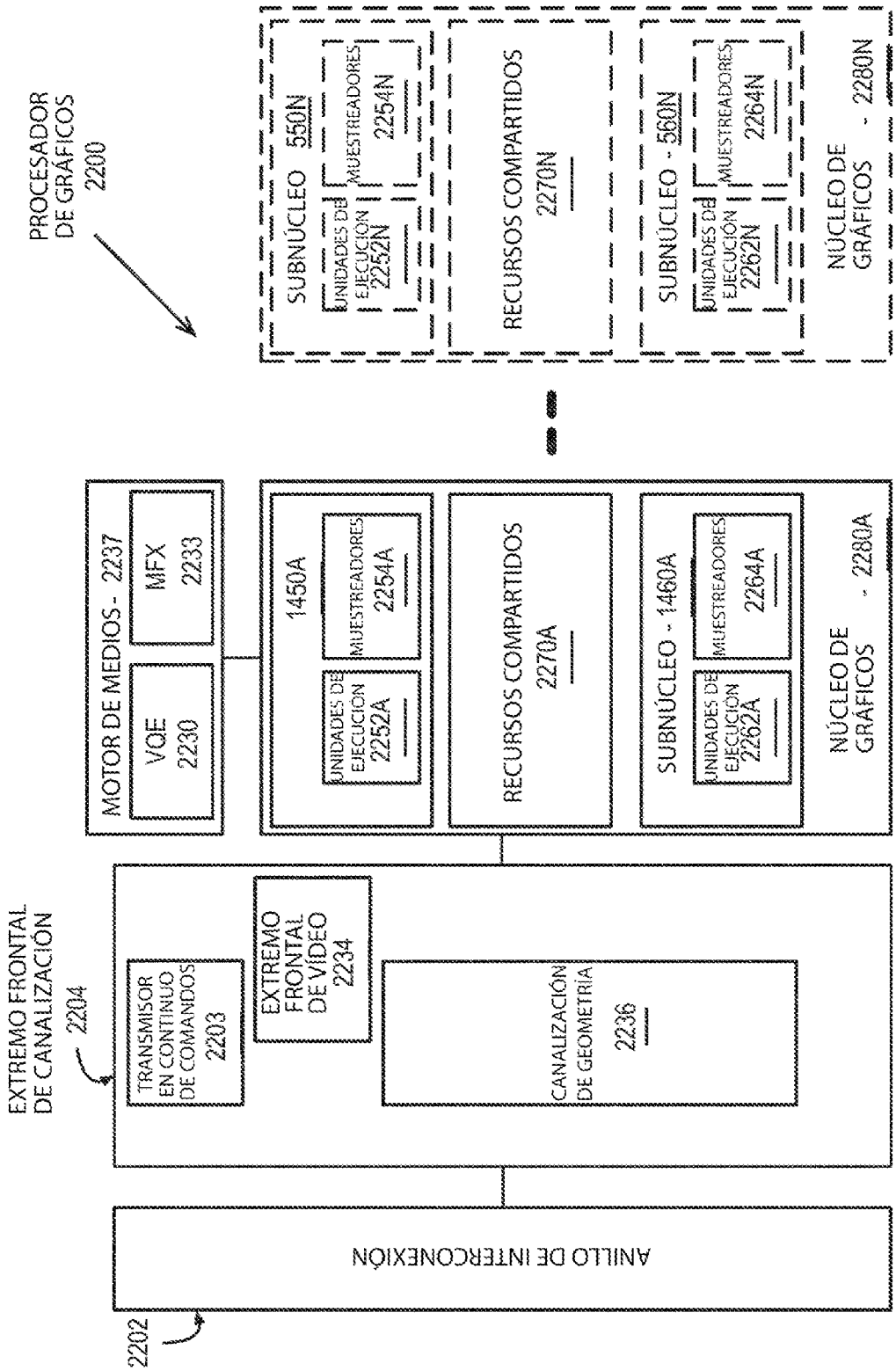

PROCESADOR DE GRÁFICOS 2200
SUBNÚCLEO 550N
UNIDADES DE EJECUCIÓN 2252N
MUESTREADORES 2254N
RECURSOS COMPARTIDOS 2270N
SUBNÚCLEO - 560N
UNIDADES DE EJECUCIÓN 2262N
MUESTREADORES 2264N
NÚCLEO DE GRÁFICOS - 2280N
MOTOR DE MEDIOS - 2237
VQE 2230
MFX 2233
1450A
UNIDADES DE EJECUCIÓN 2252A
MUESTREADORES 2254A
RECURSOS COMPARTIDOS 2270A
SUBNÚCLEO - 1460A
UNIDADES DE EJECUCIÓN 2262A
MUESTREADORES 2264A
NÚCLEO DE GRÁFICOS - 2280A
EXTREMO FRONTAL DE CANALIZACIÓN 2204
TRANSMISOR EN CONTINUO DE COMANDOS 2203
EXTREMO FRONTAL DE VÍDEO 2234
CANALIZACIÓN DE GEOMETRÍA 2236
ANILLO DE INTERCONEXIÓN
2202

FIG. 22

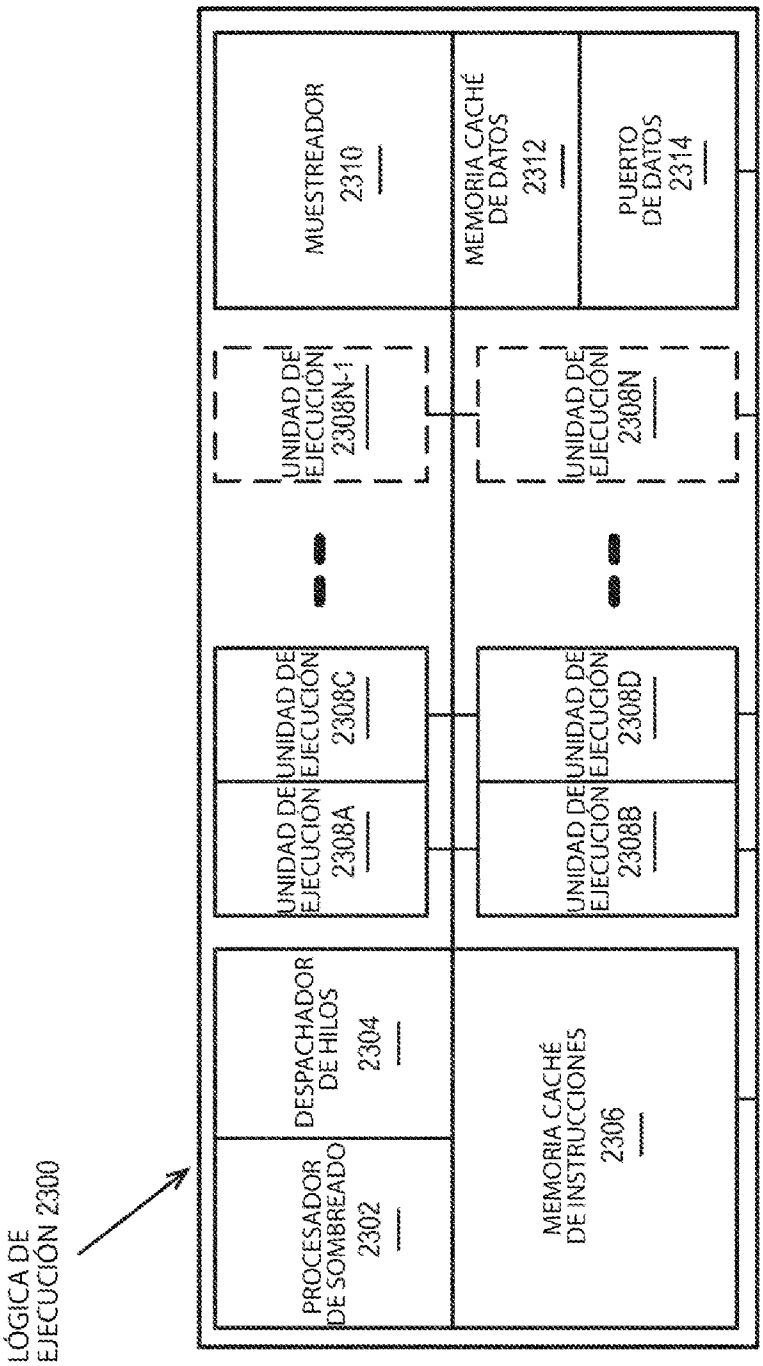

LÓGICA DE EJECUCIÓN 2300
PROCESADOR DE SOMBREADO 2302
DESPACHADOR DE HILOS 2304
MEMORIA CACHÉ DE INSTRUCCIONES 2306
UNIDAD DE EJECUCIÓN 2308A
UNIDAD DE EJECUCIÓN 2308B
UNIDAD DE EJECUCIÓN 2308C
UNIDAD DE EJECUCIÓN 2308D
UNIDAD DE EJECUCIÓN 2308N-1
UNIDAD DE EJECUCIÓN 2308N
MUESTREADOR 2310
MEMORIA CACHÉ DE DATOS 2312
PUERTO DE DATOS 2314

FIG. 23

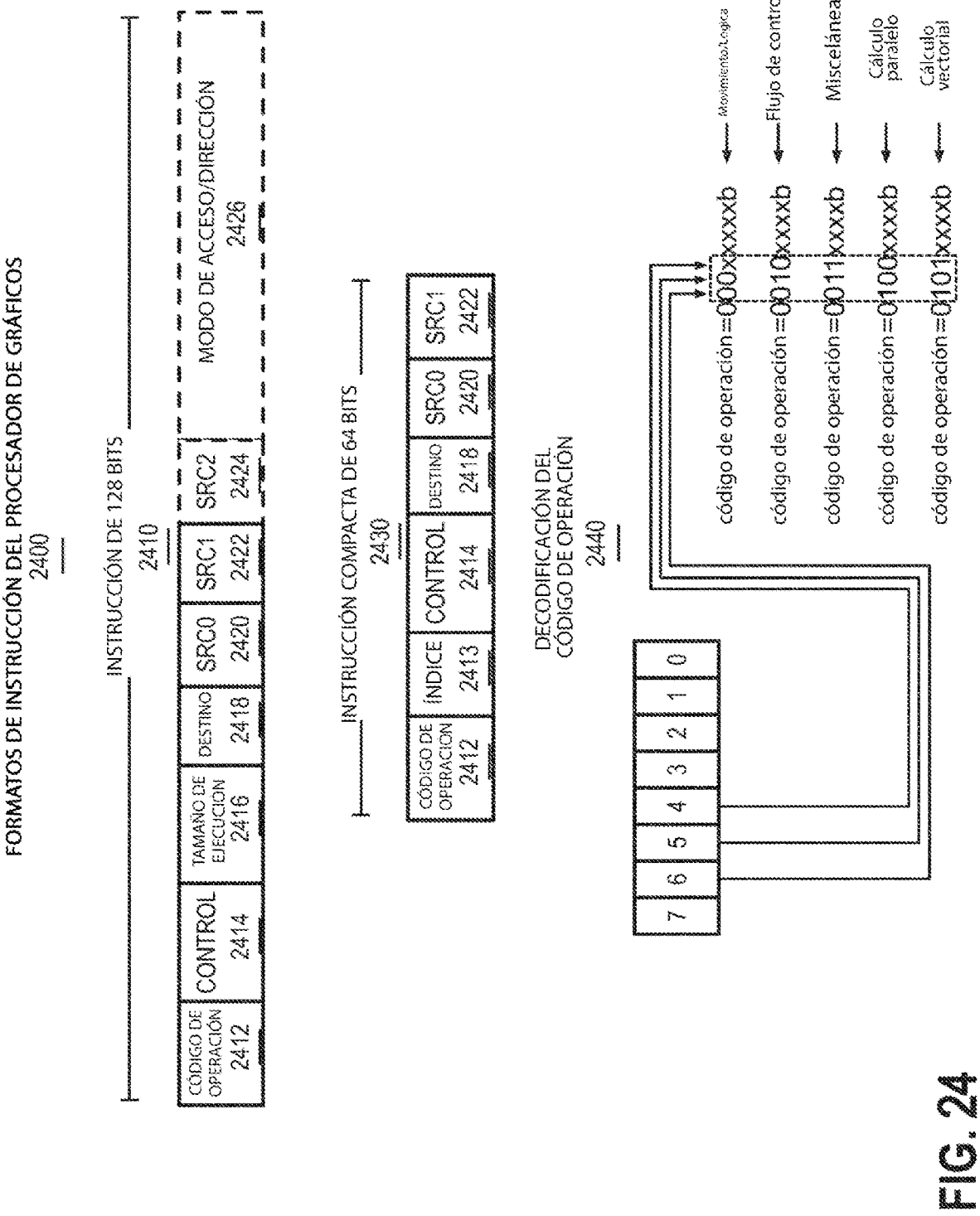

FORMATOS DE INSTRUCCIÓN DEL PROCESADOR DE GRÁFICOS
2400
INSTRUCCIÓN DE 128 BITS
2410
CÓDIGO DE OPERACIÓN 2412
CONTROL 2414
TAMAÑO DE EJECUCIÓN 2416
DESTINO 2418
SRC0 2420
SRC1 2422
SRC2 2424
MODO DE ACCESO/DIRECCIÓN 2426
INSTRUCCIÓN COMPACTA DE 64 BITS
2430
CÓDIGO DE OPERACIÓN 2412
ÍNDICE 2413
CONTROL 2414
DESTINO 2418
SRC0 2420
SRC1 2422
DECODIFICACIÓN DEL CÓDIGO DE OPERACIÓN
2440
7 6 5 4 3 2 1 0
código de operación=0000xxxxb
código de operación=0010xxxxb
código de operación=0011xxxxb
código de operación=0100xxxxb
código de operación=0101xxxxb
Movimiento/Lógica - 2442
Flujo de control - 2444
Miscelánea - 2446
Cálculo paralelo - 2448
Cálculo vectorial - 2450

FIG. 24

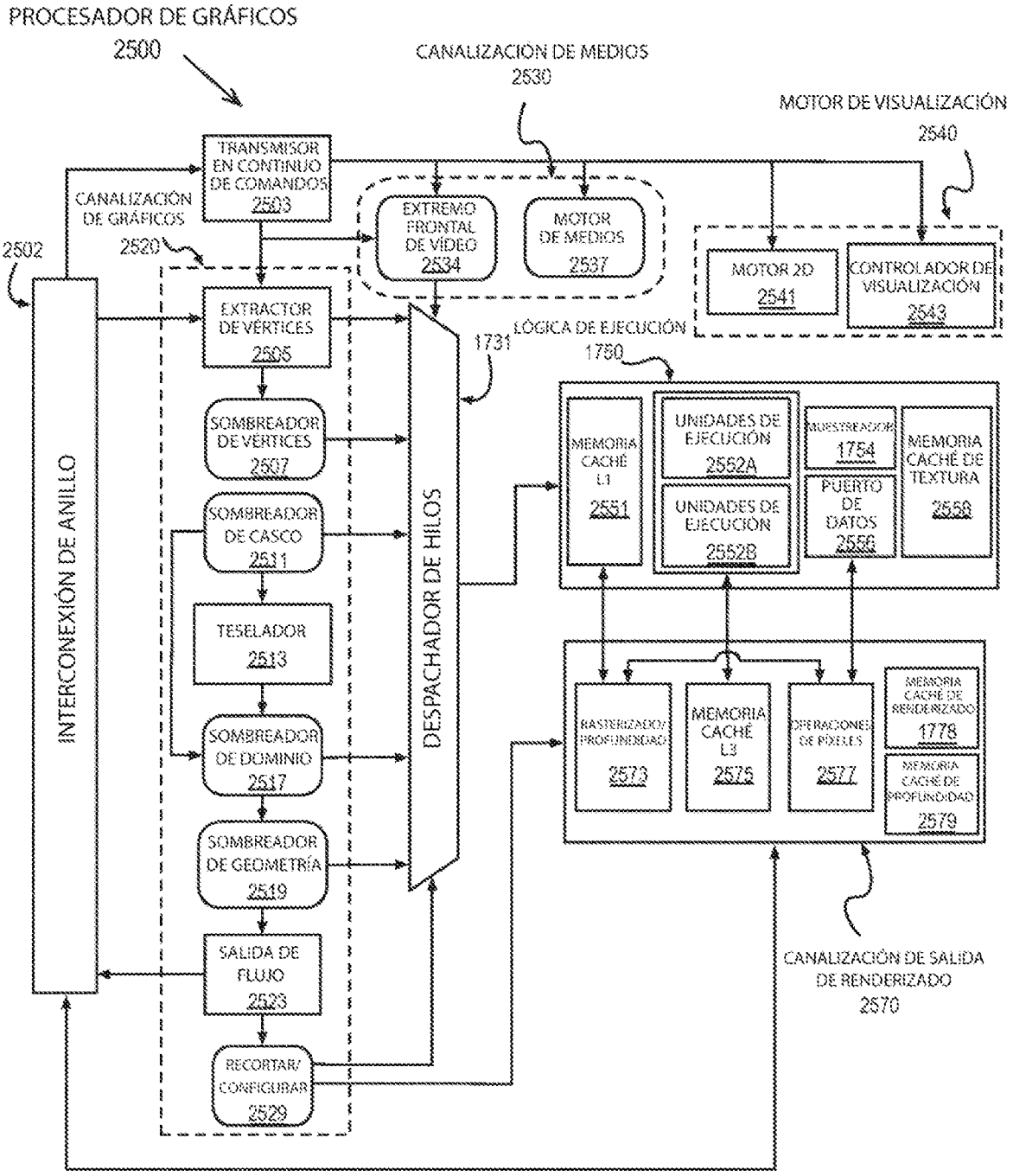

PROCESADOR DE GRÁFICOS
2500
CANALIZACIÓN DE MEDIOS
2530
MOTOR DE VISUALIZACIÓN
2540
TRANSMISOR EN CONTINUO DE COMANDOS
2503
CANALIZACIÓN DE GRÁFICOS
2520
2502
EXTREMO FRONTAL DE VÍDEO
2534
MOTOR DE MEDIOS
2537
MOTOR 2D
2541
CONTROLADOR DE VISUALIZACIÓN
2543
EXTRACTOR DE VÉRTICES
2505
1731
LÓGICA DE EJECUCIÓN
1750
SOMBREADOR DE VÉRTICES
2507
SOMBREADOR DE CASCO
2511
TESELADOR
2513
SOMBREADOR DE DOMINIO
2517
SOMBREADOR DE GEOMETRÍA
2519
SALIDA DE FLUJO
2523
RECORTAR/ CONFIGURAR
2529
INTERCONEXIÓN DE ANILLO
DESPACHADOR DE HILOS
MEMORIA CACHÉ L1
2551
UNIDADES DE EJECUCIÓN
2552A
UNIDADES DE EJECUCIÓN
2552B
MUESTREADOR
1754
PUERTO DE DATOS
2556
MEMORIA CACHÉ DE TEXTURA
2558
RASTERIZADOR/ PROFUNDIDAD
2573
MEMORIA CACHÉ L3
2575
OPERACIONES DE PÍXELES
2577
MEMORIA CACHÉ DE RENDERIZADO
1778
MEMORIA CACHÉ DE PROFUNDIDAD
2579
CANALIZACIÓN DE SALIDA DE RENDERIZADO
2570

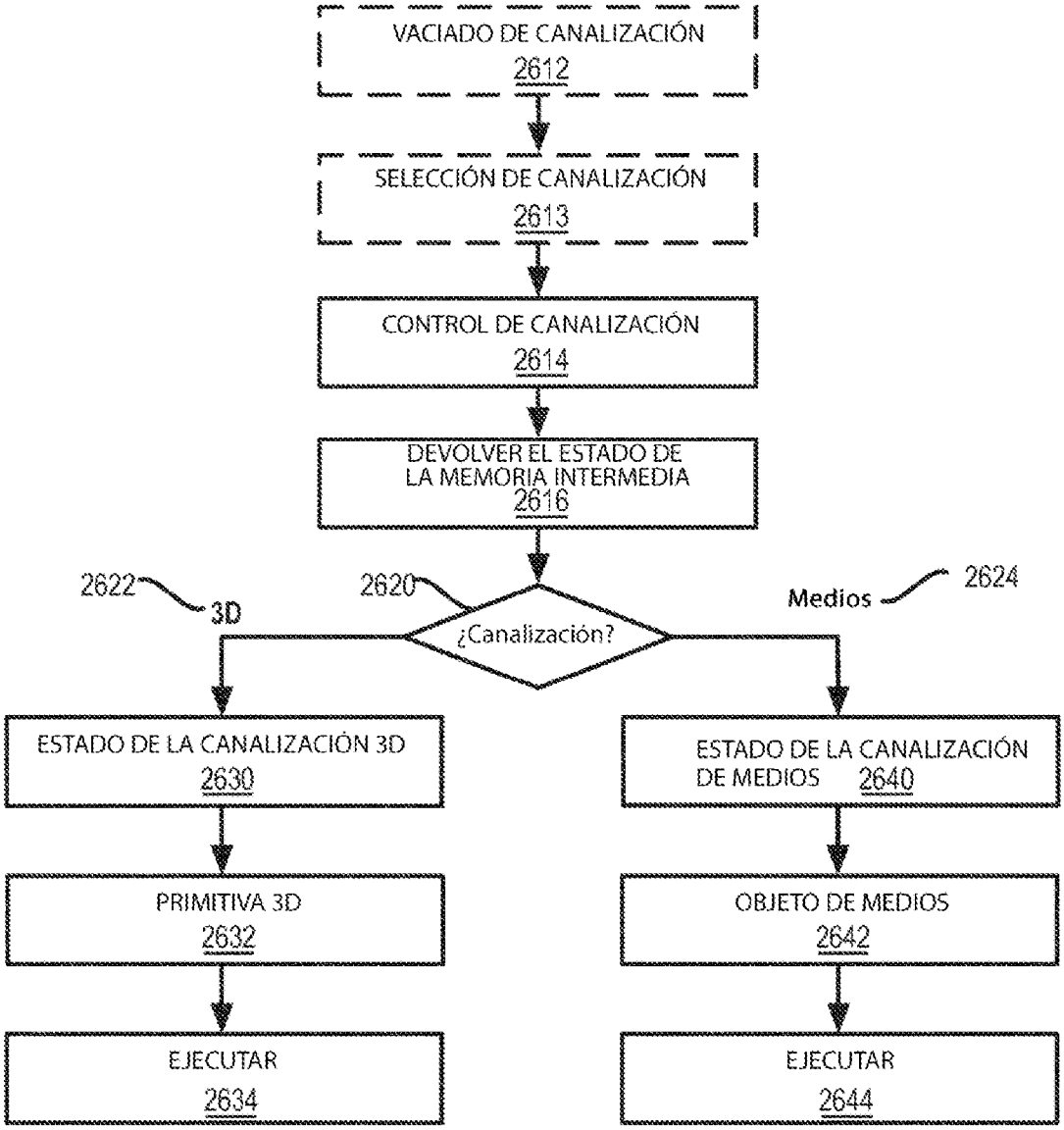
FIG. 25

FIG. 26A FORMATO DE COMANDO DE PROCESADOR DE GRÁFICOS

2600

FIG. 26B SECUENCIA DE COMANDOS DE PROCESADOR DE GRÁFICOS

2610



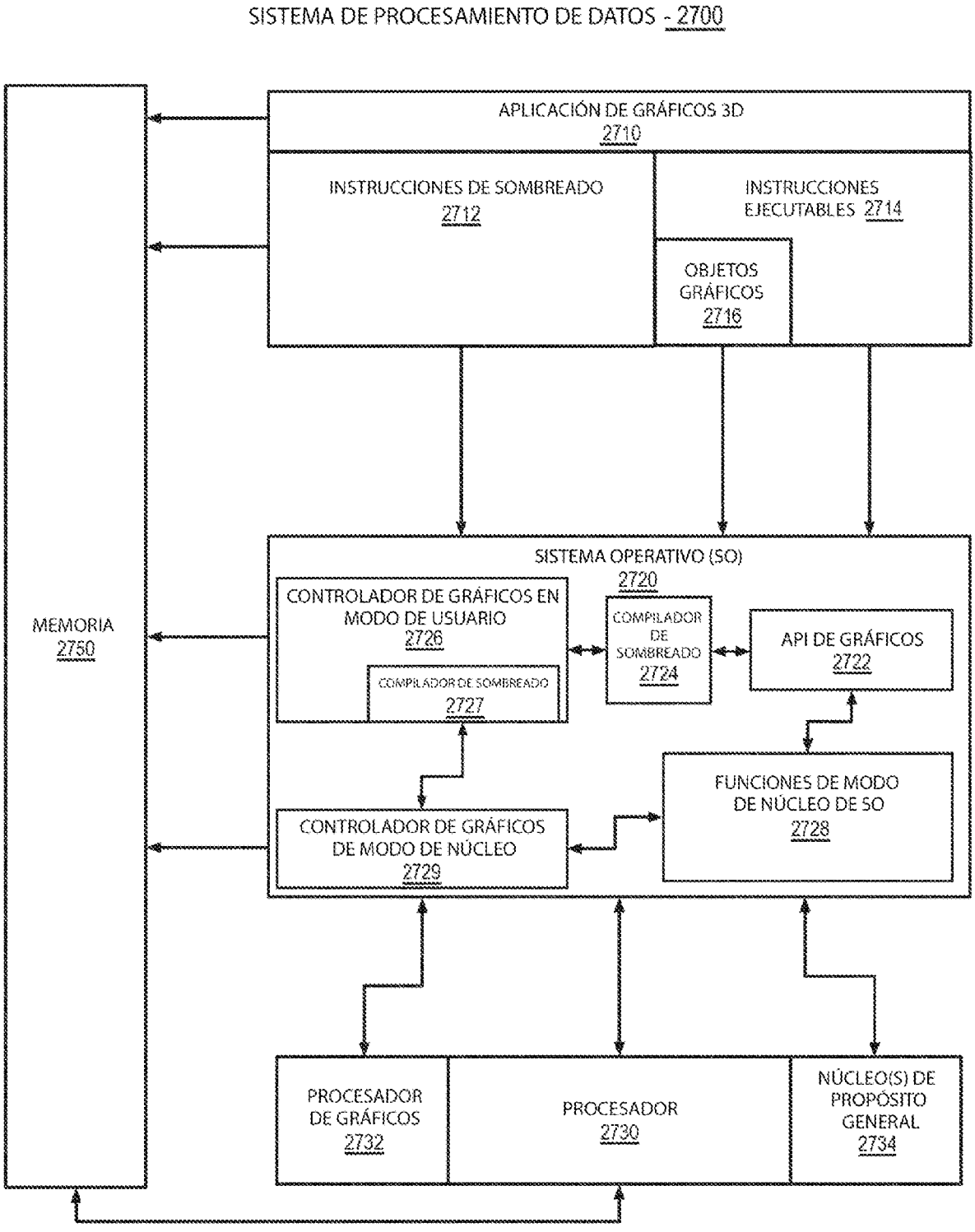

SISTEMA DE PROCESAMIENTO DE DATOS - 2700
APLICACIÓN DE GRÁFICOS 3D
2710
INSTRUCCIONES DE SOMBREADO
2712
INSTRUCCIONES EJECUTABLES 2714
OBJETOS GRÁFICOS
2716
MEMORIA
2750
SISTEMA OPERATIVO (SO)
2720
CONTROLADOR DE GRÁFICOS EN MODO DE USUARIO
2726
COMPILADOR DE SOMBREADO
2727
COMPILADOR DE SOMBREADO
2724
API DE GRÁFICOS
2722
FUNCIONES DE MODO DE NÚCLEO DE SO
2728
CONTROLADOR DE GRÁFICOS DE MODO DE NÚCLEO
2729
PROCESADOR DE GRÁFICOS
2732
PROCESADOR
2730
NÚCLEO(S) DE PROPÓSITO GENERAL
2734

FIG. 27

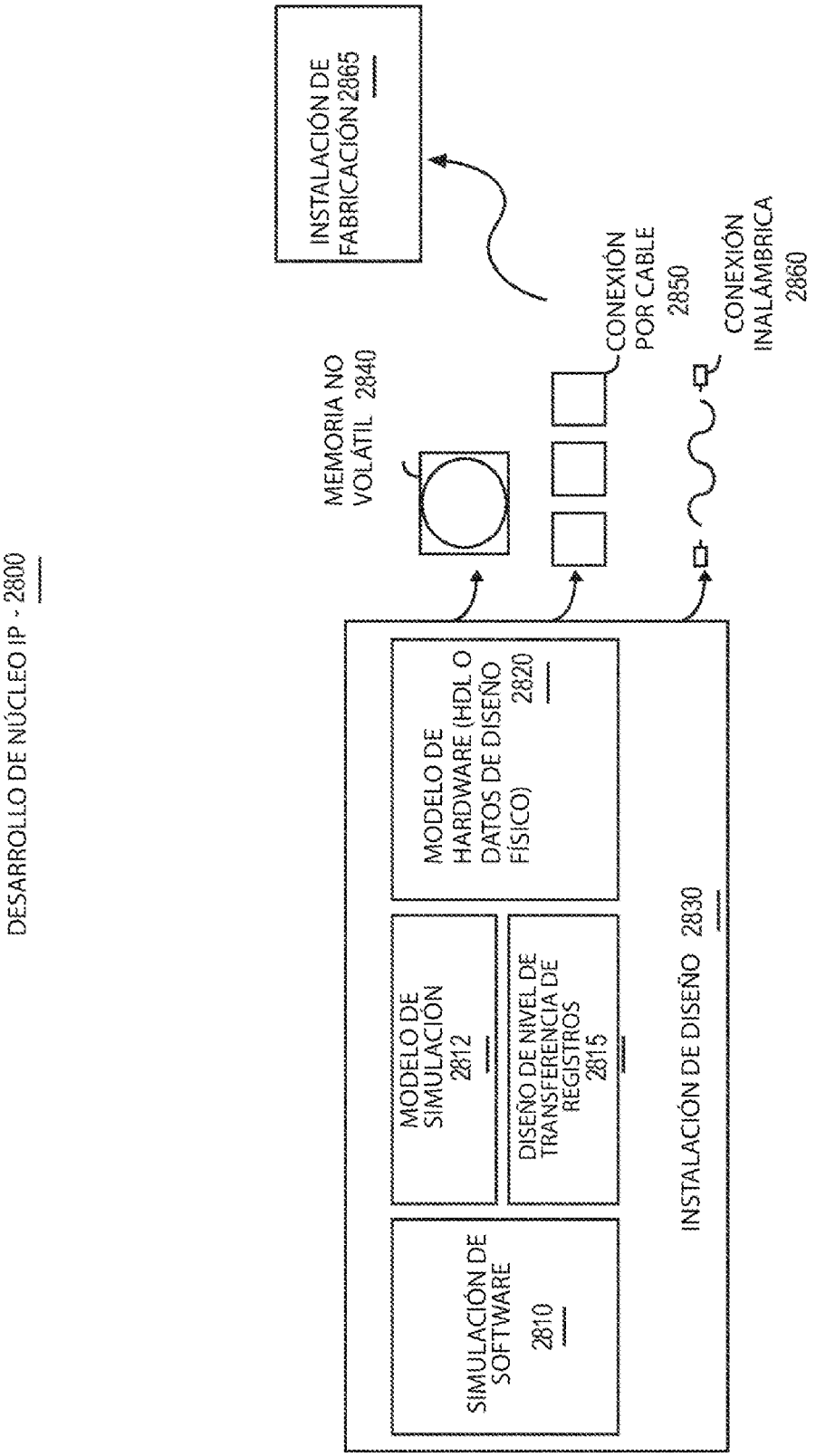

DESARROLLO DE NÚCLEO IP - 2800
INSTALACIÓN DE DISEÑO 2830
SIMULACIÓN DE SOFTWARE 2810
MODELO DE SIMULACIÓN 2812
DISEÑO DE NIVEL DE TRANSFERENCIA DE REGISTROS 2815
MODELO DE HARDWARE (HDL O DATOS DE DISEÑO FÍSICO) 2820
MEMORIA NO VOLÁTIL 2840
CONEXIÓN POR CABLE 2850
CONEXIÓN INALÁMBRICA 2860
INSTALACIÓN DE FABRICACIÓN 2865

FIG. 28

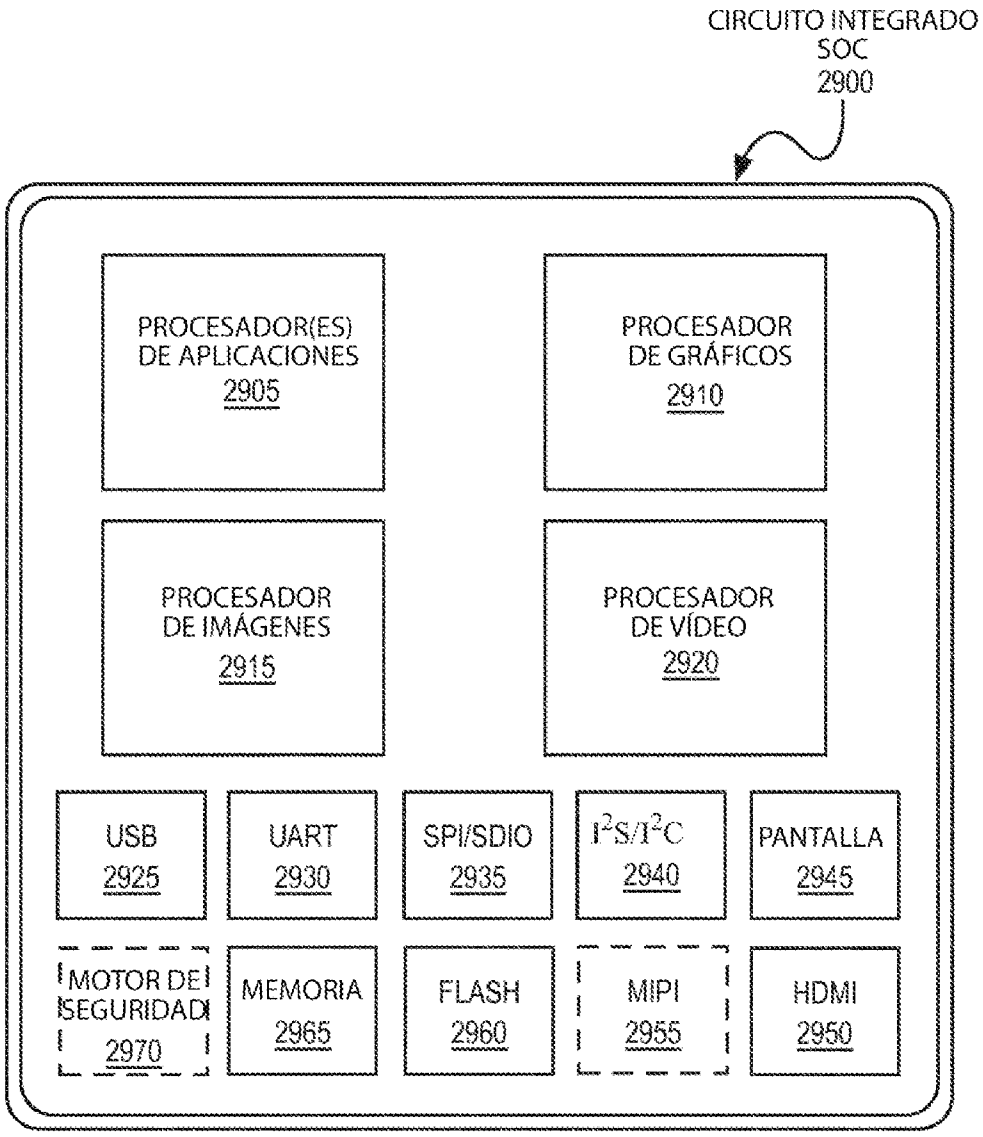

CIRCUITO INTEGRADO
SOC
2900
PROCESADOR(ES)
DE APLICACIONES
2905
PROCESADOR
DE GRÁFICOS
2910
PROCESADOR
DE IMÁGENES
2915
PROCESADOR
DE VÍDEO
2920
USB
2925
UART
2930
SPI/SDIO
2935
I2S/I2C
2940
PANTALLA
2945
MOTOR DE
SEGURIDAD
2970
MEMORIA
2965
FLASH
2960
MIPI
2955
HDMI
2950

FIG. 29

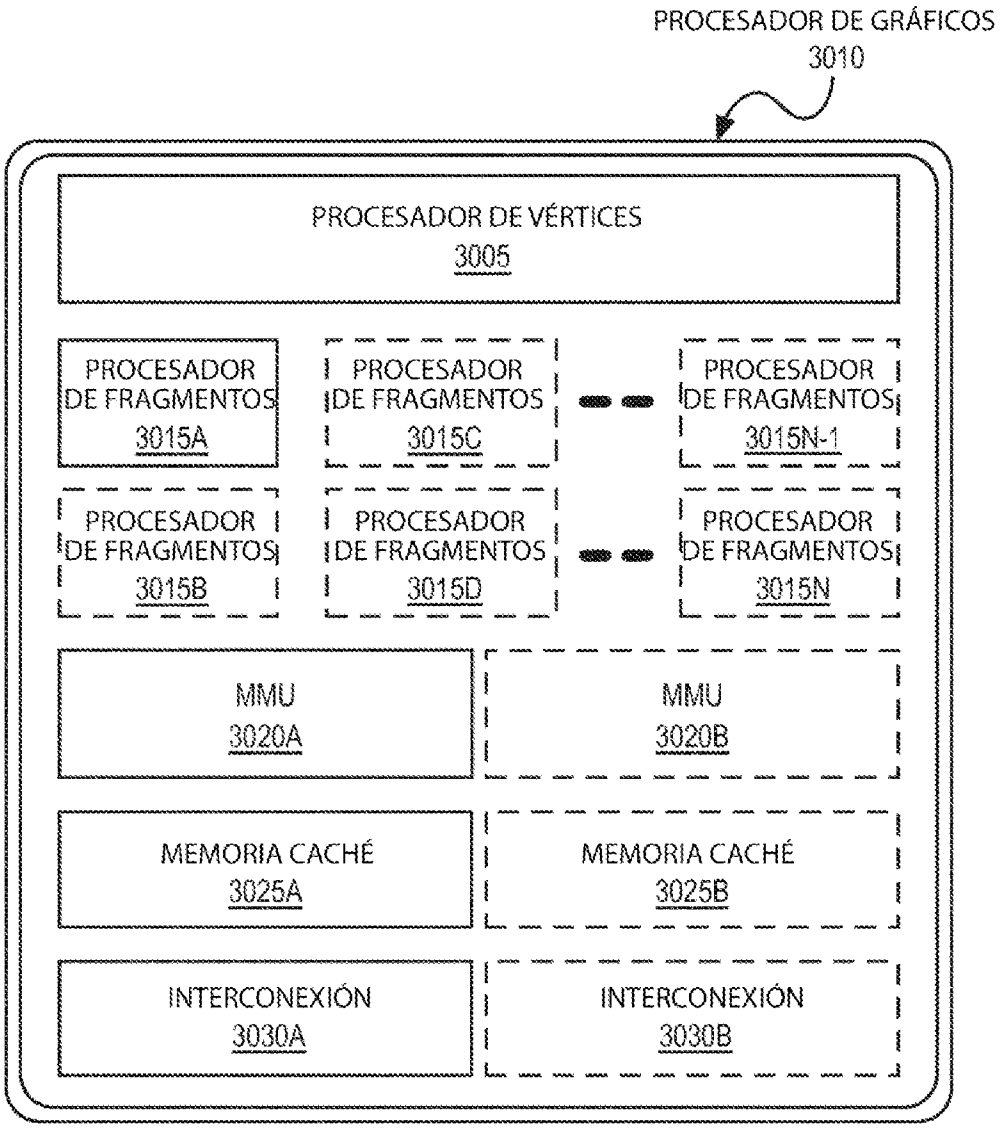

PROCESADOR DE GRÁFICOS
3010
PROCESADOR DE VÉRTICES
3005
PROCESADOR DE FRAGMENTOS
3015A
PROCESADOR DE FRAGMENTOS
3015C
PROCESADOR DE FRAGMENTOS
3015N-1
PROCESADOR DE FRAGMENTOS
3015B
PROCESADOR DE FRAGMENTOS
3015D
PROCESADOR DE FRAGMENTOS
3015N
MMU
3020A
MMU
3020B
MEMORIA CACHÉ
3025A
MEMORIA CACHÉ
3025B
INTERCONEXIÓN
3030A
INTERCONEXIÓN
3030B

FIG. 30

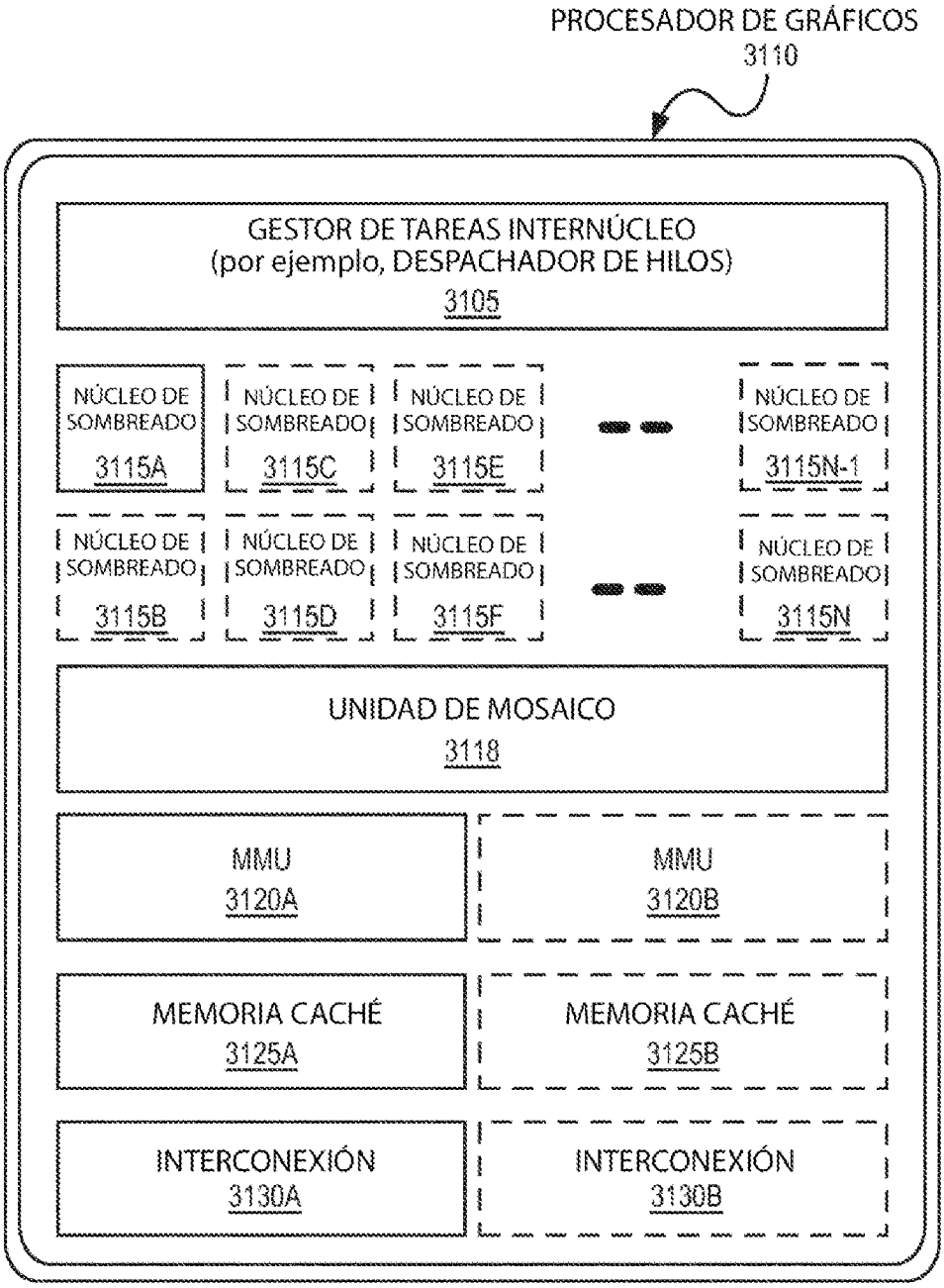

PROCESADOR DE GRÁFICOS
3110
GESTOR DE TAREAS INTERNÚCLEO
(por ejemplo, DESPACHADOR DE HILOS)
3105
NÚCLEO DE SOMBREADO
3115A
NÚCLEO DE SOMBREADO
3115C
NÚCLEO DE SOMBREADO
3115E
NÚCLEO DE SOMBREADO
3115N-1
NÚCLEO DE SOMBREADO
3115B
NÚCLEO DE SOMBREADO
3115D
NÚCLEO DE SOMBREADO
3115F
NÚCLEO DE SOMBREADO
3115N
UNIDAD DE MOSAICO
3118
MMU
3120A
MMU
3120B
MEMORIA CACHÉ
3125A
MEMORIA CACHÉ
3125B
INTERCONEXIÓN
3130A
INTERCONEXIÓN
3130B

FIG. 31