(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 962 813**

(51) Int. Cl.:

G06T 1/20 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(96) Fecha de presentación y número de la solicitud europea: **08.03.2018 E 21180506 (4)**

(97) Fecha y número de publicación de la concesión europea: **09.08.2023 EP 3937119**

(54) Título: **Mecanismo de cómputo disperso del aprendizaje automático**

(30) Prioridad:

09.04.2017 US 201715482791

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
21.03.2024

(73) Titular/es:

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

(72) Inventor/es:

NURVITADHI, ERIKO;
VEMBU, BALAJI;
LIN, TSUNG-HAN;
SINHA, KAMAL;
BARIK, RAJKISHORE y
GALOPPO VON BORRIES, NICOLAS C.

(74) Agente/Representante:

LEHMANN NOVO, María Isabel

DESCRIPCIÓN

Mecanismo de cómputo disperso del aprendizaje automático

CAMPO TÉCNICO

Las realizaciones se refieren en general al procesamiento de datos y, más particularmente, al procesamiento de datos mediante una unidad de procesamiento de gráficos de fin general.

ANTECEDENTES DE LA DESCRIPCIÓN

El procesamiento de datos de gráficos paralelo actual incluye sistemas y métodos desarrollados para realizar operaciones específicas en datos de gráficos, tales como, por ejemplo, interpolación lineal, teselación, rasterización, mapeo de texturas, prueba de profundidad, etc. De manera tradicional, los procesadores de gráficos usan unidades computacionales de función fija para procesar datos de gráficos; sin embargo, más recientemente, se han hecho programables porciones de los procesadores de gráficos, lo que posibilita que tales procesadores soporten una gama más amplia de operaciones para procesar datos de vértices y de fragmentos.

Para aumentar adicionalmente el rendimiento, los procesadores de gráficos típicamente implementan técnicas de procesamiento, tales como encauzamiento en canalizaciones, que intentan procesar, en paralelo, tantos datos de gráficos como sea posible a lo largo de todas las diferentes partes de la canalización de gráficos. Los procesadores de gráficos paralelos con arquitecturas de múltiples hilos y única instrucción (SIMT) están diseñados para maximizar la cantidad de procesamiento paralelo en la canalización de gráficos. En una arquitectura de SIMT, grupos de hilos paralelos intentan ejecutar conjuntamente instrucciones de programa de manera sincrónica tan a menudo como sea posible para aumentar la eficiencia de procesamiento. Puede encontrar una descripción general del software y hardware para arquitecturas SIMT en Shane Cook, Programación CUDA Capítulo 3, páginas 37-51 (2013).

Moczulski Marcin y otros, "ACDC: A Structured Efficient Linear Layer", páginas 1-12, URL: https://arxiv.org/pdf/1511.05946.pdf, presenta un módulo de red neuronal profundo, diferenciable y completamente conectado (capa ACDC) compuesto de matrices diagonales de parámetros con cascadas profundas de capas ACDC que se aproximan a capas lineales.

SUMARIO

La presente invención se define en las reivindicaciones independientes. Las reivindicaciones dependientes definen realizaciones de las mismas. Cualquier "realización" o "ejemplo" que se divulgue en la siguiente descripción pero que no esté cubierto por las reivindicaciones debe considerarse presentado únicamente con fines ilustrativos.

BREVE DESCRIPCIÓN DE LOS DIBUJOS

Para que la forma en que las características citadas anteriormente de las presentes realizaciones puedan entenderse en detalle, se puede obtener una descripción más particular de las realizaciones, brevemente resumida anteriormente, con referencia a las realizaciones, algunas de las cuales se ilustran en los dibujos adjuntos. Sin embargo, se ha de observar que los dibujos adjuntos ilustran únicamente realizaciones típicas y, por lo tanto, no han de considerarse limitantes de su alcance.

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

La **Figura 2A-2D** ilustra unos componentes de procesador paralelo, de acuerdo con una realización;

Las Figuras 3A-3B son diagramas de bloques de multiprocesadores de gráficos, de acuerdo con realizaciones;

Las **Figuras 4A-4F** ilustran una arquitectura ilustrativa en la que una pluralidad de GPU están comunicativamente acopladas a una pluralidad de procesadores de múltiples núcleos;

La **Figura 5** ilustra un conducto de procesamiento de gráficos, de acuerdo con una realización;

La **Figura 6** ilustra un dispositivo informático que emplea un mecanismo de cómputo disperso, de acuerdo con una realización;

La **Figura 7A** ilustra una multiplicación de matrices ilustrativa;

La **Figura 7B** ilustra una realización de un elemento de procesamiento que tiene un planificador disperso;

La **Figura 7C** ilustra una realización de un elemento de procesamiento que tiene un seguidor disperso;

Las **Figuras 7D&7E** ilustran realizaciones de un procesador de gráficos;

La **Figura 8** ilustra una pila de software de aprendizaje automático, de acuerdo con una realización;

La **Figura 9** ilustra una unidad de procesamiento de gráficos de fin general altamente paralela, de acuerdo con una realización;

La **Figura 10** ilustra un sistema informático de múltiples GPU, de acuerdo con una realización;

Las **Figuras 11A-11B** ilustran capas de redes neuronales profundas ilustrativas;

La **Figura 12** ilustra una red neuronal recurrente ilustrativa;

La **Figura 13** ilustra el entrenamiento y despliegue de una red neuronal profunda.

La **Figura 14** es un diagrama de bloques que ilustra un aprendizaje distribuido;

La **Figura 15** ilustra un sistema de inferencia ilustrativa en un chip (SOC) adecuado para realizar inferencias utilizando un modelo entrenado;

La **Figura 16** es un diagrama de bloques de un sistema de procesamiento, de acuerdo con una realización;

La **Figura 17** es un diagrama de bloques de un procesador de acuerdo con una realización;

La **Figura 18** es un diagrama de bloques de un procesador de gráficos, de acuerdo con una realización;

La **Figura 19** es un diagrama de bloques de un motor de procesamiento gráfico de un procesador de gráficos de acuerdo con algunas realizaciones;

La **Figura 20** es un diagrama de bloques de un procesador de gráficos proporcionado por una realización adicional;

La **Figura 21** ilustra la lógica de ejecución de hilo que incluye una matriz de elementos de procesamiento empleados en algunas realizaciones;

La **Figura 22** es un diagrama de bloques que ilustra unos formatos de instrucción de procesador de gráficos de acuerdo con algunas realizaciones;

La **Figura 23** es un diagrama de bloques de un procesador de gráficos, de acuerdo con una realización;

La **Figura 24A-24B** ilustra un formato de comando de procesador de gráficos y secuencia de comandos, de acuerdo con algunas realizaciones;

La **Figura 25** ilustra una arquitectura de software de gráficos ilustrativa para un sistema de procesamiento de datos de acuerdo con algunas realizaciones;

La **Figura 26** es un diagrama de bloques que ilustra un sistema de desarrollo de núcleo de IP, de acuerdo con una realización;

La **Figura 27** es un diagrama de bloques que ilustra un sistema ilustrativo en un circuito integrado de chip, de acuerdo con una realización;

La **Figura 28** es un diagrama de bloques que ilustra un procesador de gráficos ilustrativo adicional; y

La **Figura 29** es un diagrama de bloques que ilustra un procesador de gráficos ilustrativo adicional de un sistema en un circuito integrado de chip, de acuerdo con una realización.

DESCRIPCIÓN DETALLADA

En realizaciones, se divulgan mecanismos para realizar un mecanismo de procesamiento de matriz dispersa. En algunas realizaciones, el mecanismo de procesamiento incluye elementos de procesamiento que incluyen un programador para identificar operandos que tienen un valor cero y evitar la programación de los operandos que tienen el valor cero en la unidad de multiplicación. En otras realizaciones, el mecanismo de procesamiento incluye lógica de seguimiento de patrones para detectar uno o más segmentos de datos dispersos en un bloque de datos almacenado

y registrar una ubicación de dirección para cada segmento detectado de datos dispersos. Aún en otras realizaciones, el mecanismo de procesamiento comprime matrices dispersas y almacena una o más matrices dispersas utilizadas con frecuencia en una memoria intermedia comprimida dispersa para su ejecución para el procesamiento. En una realización adicional, el mecanismo de procesamiento divide una pluralidad de unidades de ejecución (EU) y asigna cada partición de EU para ejecutar hilos asociados con una capa de red neuronal.

En la siguiente descripción, se exponen numerosos detalles específicos para proporcionar una comprensión más minuciosa. Sin embargo, será evidente para un experto en la técnica que las formas de realización descritas en el presente documento pueden ponerse en práctica sin uno o más de estos detalles específicos. En otros casos, no se han descrito características bien conocidas para evitar oscurecer los detalles de las presentes formas de realización.

Visión general del sistema

La **Figura 1** es un diagrama de bloques que ilustra un sistema informático 100 configurado para implementar uno o más aspectos de las formas de realización descritas en el presente documento. El sistema informático 100 incluye un subsistema de procesamiento 101 que tiene uno o más procesador o procesadores 102 y una memoria de sistema 104 que se comunica mediante 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 conjunto de chips o puede estar integrado dentro del uno o más procesador o procesadores 102. El concentrador de memoria 105 se acopla con un subsistema de E/S 111 mediante un enlace de comunicación 106. El subsistema de E/S 111 incluye un concentrador de E/S 107 que puede permitir que el sistema informático 100 reciba una entrada desde uno o más dispositivos de entrada 108. Adicionalmente, el concentrador de E/S 107 puede permitir un controlador de visualización, que puede estar incluido en el uno o más procesadores 102, para que proporcione salidas a uno o más dispositivos de visualización 110A. En una realización, el uno o más dispositivos de visualización 110A acoplados con el concentrador de E/S 107 pueden incluir un dispositivo de visualización local, interno o embebido.

En una forma de realización, el subsistema de procesamiento 101 incluye uno o más procesadores en paralelo 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 estándares, como por ejemplo, entre otros, una 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 grupos 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 emitir píxeles a uno del uno o más dispositivos de visualización 110A acoplados mediante el concentrador de E/S 107. El uno o más procesadores paralelo(s) 112 también puede incluir un controlador de pantalla y una interfaz de pantalla (no mostrada) para permitir una conexión directa a uno o más dispositivo(s) de pantalla 11DB.

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. Puede usarse 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 inalámbrico 119 que pueden integrarse en la plataforma, y diversos otros dispositivos que pueden añadirse mediante uno o más dispositivos de adición 120. El adaptador de red 118 puede ser un adaptador de Ethernet u otro adaptador de red alámbrica. El adaptador de red inalámbrica 119 puede incluir uno o más de un dispositivo de Wi-Fi, Bluetooth, de comunicación de campo cercano (NFC) u otra red que incluye una o más radios inalámbricas.

El sistema informático 100 puede incluir otros componentes no explícitamente mostrados, incluyendo USB u otras conexiones de puerto, unidades de almacenamiento óptico, dispositivos de captura de vídeo, y similares, que también puede conectarse al concentrador de E/S 107. Las rutas de comunicación que interconectan los diversos componentes en la Figura 1 pueden implementarse usando cualquier protocolo adecuado, tal como protocolos (por ejemplo, PCI-Express) basados en PCI (Interconexión de Componentes Periféricos), o cualesquiera otras interfaces de comunicación de bus o de punto a punto y/o protocolo o protocolos, tal como la interconexión de alta velocidad NVLink, o protocolos de interconexión conocidos en la técnica.

En una realización, el uno o más procesadores paralelos 112 incorporan circuitería optimizada para procesamiento de gráficos y vídeo, que incluye, por ejemplo, circuitería de salida de vídeo y constituye una unidad de procesamiento de gráficos (GPU). En otra realización, los 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 todavía otra forma de realización, componentes del sistema informático 100 se pueden integrar con otros uno o más elementos de sistema en un único circuito integrado. Por ejemplo, los 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). Como alternativa, los componentes del sistema informático 100 pueden integrarse en un único paquete para formar una configuración de sistema en paquete (SIP). En una forma de realización, al menos una porción de los componentes del sistema informático 100 se pueden

integrar en un módulo multimicroprocesador (MCM), que se puede interconectar con otros módulos multimicroprocesador en un sistema informático modular.

Se apreciará que, el sistema informático 100 mostrado en la presente memoria sea ilustrativo y que sean posibles variaciones y modificaciones. La topología de conexión, que incluye el número y disposición de puentes, el número de procesador o procesadores 102, y el número de procesador o procesadores paralelos 112, puede modificarse como se desee. Por ejemplo, en algunas realizaciones, la memoria de 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 de sistema 104 mediante el concentrador de memoria 105 y el procesador o procesadores 102. En otras topologías alternativas, el procesador o procesadores paralelos 112 están conectados al concentrador de E/S 107 o directamente a uno del uno o más procesador o 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 del procesador o procesadores 102 adjuntos mediante múltiples zócalos, que pueden acoplarse con dos o más instancias del procesador o procesadores paralelos 112.

Alguno de los componentes particulares mostrados en el presente documento es opcional y puede no estar incluido en todas las implementaciones del sistema informático 100. Por ejemplo, puede soportarse cualquier número de tarjetas o periféricos de adición, o pueden eliminarse algunos componentes. Adicionalmente, algunas arquitecturas pueden usar diferente terminología para componentes similares a aquellos ilustrados en la Figura 1. Por ejemplo, el concentrador de memoria 105 puede denominarse un puente norte en algunas arquitecturas, mientras que el concentrador de E/S 107 puede denominarse un puente sur.

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

En una forma de realización, el procesador en paralelo 200 incluye una unidad de procesamiento en paralelo 202. La unidad de procesamiento paralelo incluye una unidad de E/S 204 que posibilita la comunicación con otros dispositivos, que incluyen otras instancias de la unidad de procesamiento paralelo 202. La unidad de E/S 204 se puede conectar directamente a otros dispositivos. En una forma de realización, la unidad de E/S 204 se conecta con otros dispositivos por medio de la utilización de una interfaz de concentrador o de conmutador, como por ejemplo 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 las 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 comando mediante la unidad de E/S 204, la interfaz de anfitrión 206 puede dirigir operaciones de trabajo para realizar aquellos comandos a un extremo delantero 208. En una realización, el extremo delantero 208 se acopla con un planificador 210, que está configurado para distribuir comandos u otros elementos de trabajo a una matriz de grupo de procesamiento 212. En una forma de realización, el programador 210 garantiza que la matriz de grupos de procesamiento 212 se configura adecuadamente y se encuentra en un estado válido antes de que se distribuyan las tareas a los grupos de procesamiento de la matriz de grupos de procesamiento 212.

La matriz de grupo de procesamiento 212 puede incluir hasta "N" grupos de procesamiento (por ejemplo, del grupo 214A, el grupo 214B al grupo 214N). Cada grupo 214A-214N de la matriz de grupo de procesamiento 212 puede ejecutar un gran número de hilos concurrentes. El planificador 210 puede asignar trabajo a los grupos 214A-214N de la matriz de grupo de procesamiento 212 usando diversos algoritmos de planificación y/o distribución de trabajo, que pueden variar dependiendo de la carga de trabajo que surge para cada tipo de programa o cómputo. La planificación puede manejarse dinámicamente por el planificador 210, o puede ser ayudada, en parte, por lógica de compilador durante la compilación de la lógica de programa configurada para la ejecución por la matriz de grupo de procesamiento 212.

En una realización, pueden asignarse diferentes grupos 214A-214N de la matriz de grupo de procesamiento 212 para procesar diferentes tipos de programas o para realizar diferentes tipos de cómputos.

La matriz de grupo de procesamiento 212 se puede configurar para realizar varios tipos de operaciones de procesamiento en paralelo. En una realización, la matriz de grupo de procesamiento 212 está configurada para realizar operaciones de cómputo en paralelo de propósito general. Por ejemplo, la matriz de grupo de procesamiento 212 puede incluir lógica para ejecutar tareas de procesamiento que incluyen el filtrado de datos de vídeo y/o audio y/u operaciones de modelado, incluidas operaciones físicas y la realización de transformaciones de datos.

En una realización, la matriz de grupo 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 grupo de procesamiento 212 puede incluir lógica adicional para soportar la ejecución de dichas operaciones de procesamiento de gráficos, incluyendo, entre otras, lógica de muestreo de textura para realizar operaciones de textura, como así como lógica de teselación y otra lógica de procesamiento de vértices. Adicionalmente, la matriz de grupos de procesamiento 212 puede configurarse para ejecutar programas sombreadores relacionados con el procesamiento de gráficos tales como, pero sin limitación, 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 de sistema mediante la unidad de E/S 204 para su procesamiento. Durante el procesamiento, los datos transferidos pueden almacenarse en memoria en chip (por ejemplo, memoria de procesador paralelo 222) durante el procesamiento y, a continuación, escribirse de vuelta en memoria de sistema.

En una realización, cuando se usa la unidad de procesamiento paralelo 202 para realizar el procesamiento de gráficos, el planificador 210 puede estar configurado 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 grupos 214A-214N de la matriz de grupo de procesamiento 212. En algunas realizaciones, porciones de la matriz de grupo de procesamiento 212 se pueden configurar 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 los grupos 214A-214N pueden almacenarse en memorias intermedias para permitir que los datos intermedios se transmitan entre los grupos 214A-214N para su procesamiento adicional.

Durante el funcionamiento, la matriz de grupos de procesamiento 212 puede recibir tareas de procesamiento que hay que ejecutar mediante el planificador 210, que recibe comandos que definen tareas de procesamiento desde el extremo frontal 208. Para operaciones de procesamiento de gráficos, las tareas de procesamiento pueden incluir índices de datos que hay que procesar, por ejemplo, datos de superficie (parche), datos de primitiva, datos de vértice y/o datos de píxel, así como parámetros de estado y comandos que definen cómo han de procesarse los datos (por ejemplo, qué programa ha de ejecutarse). 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 puede configurarse para garantizar que la matriz de grupos de procesamiento 212 está configurada en un estado válido antes de que se inicie la carga de trabajo especificada por las memorias intermedias de comando de entrada (por ejemplo, memorias intermedias de lotes, memorias intermedias de inserción, etc.).

Cada una de las una o más instancias de la unidad de procesamiento paralelo 202 puede acoplarse con la memoria de procesador paralelo 222. Puede accederse a la memoria de procesador paralelo 222 mediante la barra transversal de memoria 216, que puede recibir solicitudes de memoria desde la matriz de grupos de procesamiento 212, así como la unidad de E/S 204. La barra transversal de memoria 216 puede acceder a la memoria de procesador paralelo 222 mediante una interfaz de memoria 218. La interfaz de memoria 218 puede incluir múltiples unidades de partición (por ejemplo, la unidad de partición 220A, la unidad de partición 220B a la unidad de partición 220N), cada una acoplable a una porción (por ejemplo, la unidad de memoria) de la memoria de procesador paralelo 222. En una implementación, el número de unidades de partición 220A-220N está configurado para que sea igual al número de unidades de memoria, de manera que una primera unidad de partición 220A tiene una correspondiente primera unidad de memoria 224A, una segunda unidad de partición 220B tiene una correspondiente unidad de memoria 224B y una unidad de partición de orden N 220N tiene una correspondiente unidad de memoria de orden N 224N. En otras realizaciones, el número de unidades de partición 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 de acceso aleatorio de gráficos, tal como la memoria de acceso aleatorio de gráficos síncrona (SGRAM), que incluye la memoria de tasa de datos doble de gráficos (GDDR). En una realización, las unidades de memoria 224A-224N pueden incluir también memoria 3D apilada, que incluye, pero sin limitación, memoria de ancho de banda alto (HBM). Los expertos en la técnica apreciarán que la implementación específica de las unidades de memoria 224A-224N puede variar, y puede seleccionarse de uno de diversos diseños convencionales. Los objetivos de representación, tales como las memorias intermedias de tramas o los mapas de textura pueden almacenarse a través de las unidades de memoria 224A-224N, permitiendo que las unidades de partición 220A-220N escriban porciones de cada objetivo de representación en paralelo para usar de manera efectiva 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 en conjunto con memoria caché local.

En una realización, una cualquiera de los grupos 214A-214N de la matriz de grupo 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 puede estar configurada para transferir la salida de cada grupo 214A-214N en cualquier unidad de partición 220A-220N o en otro grupo 214A-214N, que puede realizar

operaciones de procesamiento adicionales en la salida. Cada grupo 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 diversos 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 posibilita que las unidades de procesamiento dentro de las diferentes grupos 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 los grupos 214A-214N y las unidades de partición 220A-220N.

Aunque se ilustra una única instancia de la unidad de procesamiento paralelo 202 dentro del procesador paralelo 200, puede incluirse cualquier número de instancias de la unidad de procesamiento paralelo 202. Por ejemplo, pueden proporcionarse múltiples instancias de la unidad de procesamiento paralelo 202 en una única tarjeta de complemento, o pueden interconectarse múltiples tarjetas de complemento. Las diferentes instancias de la unidad de procesamiento paralelo 202 pueden estar configuradas 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 forma de realización, algunas instancias de la unidad de procesamiento en paralelo 202 pueden incluir unidades de punto flotante de precisión más alta 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 limitación, ordenadores personales de sobremesa, 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 partición 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 caché L2 221, una interfaz de memoria intermedia de trama 225 y una ROP 226 (unidad de operaciones de rasterización). La caché L2 221 es una caché de lectura/escritura que está configurada para realizar operaciones de carga y almacenamiento recibidas desde la barra transversal de memoria 216 y el ROP 226. Los errores de lectura y las solicitudes urgentes de reescritura son enviados por la memoria caché L2 221 a la interfaz de memoria intermedia de tramas 225 para su procesamiento. Pueden enviarse también las actualizaciones sucias a la memoria intermedia de tramas mediante la interfaz de memoria intermedia de tramas 225 para su procesamiento oportunista. En una realización, la interfaz de memoria intermedia de trama 225 interconecta con una de las unidades de memoria en la memoria de procesador paralelo, tal como las unidades de memoria 224A-224N de la Figura 2 (por ejemplo, dentro de la memoria de procesador paralelo 222).

En las aplicaciones de gráficos, la ROP 226 es una unidad de procesamiento que lleva a cabo operaciones de rasterización como por ejemplo estarcido, prueba z, mezcla y similares. La ROP 226 emite entonces datos de gráficos procesados que se almacenan en una memoria de gráficos. En algunas realizaciones, la ROP 226 incluye una lógica de compresión para comprimir datos de z o de color que se escriben en la memoria y descomprimir datos de z o de color que se leen desde la memoria. En algunas realizaciones, la ROP 226 está incluida dentro de cada grupo de procesamiento (por ejemplo, el grupo 214A-214N de la Figura 2) en lugar de dentro de la unidad de partición 220. En tal realización, las solicitudes de lectura y escritura para datos de píxeles se transmiten a través de la barra transversal de memoria 216 en lugar de los datos de fragmento de píxel.

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, encaminarse para su procesamiento adicional por el procesador o procesadores 102, o encaminarse 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 un grupo de procesamiento 214 dentro de una unidad de procesamiento paralelo, de acuerdo con una realización. En una realización, el grupo de procesamiento es una instancia de una de los grupos de procesamiento 214A-214N de la Figura 2. El grupo 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 y datos múltiples (SIMD) para soportar la ejecución en paralelo de una gran cantidad de hilos sin proporcionar múltiples unidades de instrucción independientes. En otras realizaciones, se usan técnicas de única instrucción de múltiples hilos (SIMT) para soportar la ejecución paralela de un gran número de hilos generalmente sincronizados, usando una unidad de instrucción común configurada para emitir instrucciones en un conjunto de motores de procesamiento dentro de cada uno de los grupos de procesamiento. A diferencia del régimen de ejecución SIMD, donde todos los motores de procesamiento ejecutan normalmente instrucciones idénticas, la ejecución SIMT permite que diferentes hilos sigan más fácilmente trayectorias de ejecución divergentes a través de un programa de hilos dado. Los expertos en la técnica entenderán que un régimen de procesamiento SIMD representa un subconjunto funcional de un régimen de procesamiento SIMT.

El funcionamiento del grupo de procesamiento 214 se puede controlar a través de un gestor de conducto 232 que distribuye las tareas de procesamiento a los procesadores en paralelo SIMT. El administrador de canalización 232 recibe instrucciones del programador 210 de la Figura 2 y gestiona la ejecución de esas instrucciones a través de un

multiprocesador de gráficos 234 y/o una unidad de textura 236. El multiprocesador de gráficos ilustrado 234 es un ejemplo de procesador paralelo SIMT. Sin embargo, se pueden incluir varios tipos de procesadores paralelos SIMT de diferentes arquitecturas dentro del grupo de procesamiento 214. Una o más instancias del multiprocesador de gráficos 234 pueden incluirse dentro de un grupo de procesamiento 214. El multiprocesador de gráficos 234 puede procesar datos y puede usarse una barra transversal de datos 240 para distribuir los datos procesados a uno de múltiples posibles destinos, que incluyen otras unidades sombreadoras. El gestor de canalizaciones 232 puede facilitar la distribución de datos procesados especificando destinos para que se distribuyan datos procesados mediante la barra transversal de datos 240.

Cada multiprocesador de gráficos 234 dentro del grupo de procesamiento 214 puede incluir un conjunto idéntico de lógica de ejecución funcional (por ejemplo, unidades aritmético-lógicas, unidades de carga-almacén, etc.). La lógica de ejecución funcional puede configurase en forma en tubería en la que pueden emitirse nuevas instrucciones antes de que estén completadas instrucciones anteriores. La lógica de ejecución funcional soporta una diversidad de operaciones que incluyen aritmética de números enteros y de coma flotante, operaciones de comparación, operaciones booleanas, desplazamiento de bits y del cómputo de diversas funciones algebraicas. En una realización, puede aprovecharse el mismo hardware funcional-unitario para realizar diferentes operaciones y puede estar presente cualquier combinación de unidades funcionales.

Las instrucciones transmitidas al grupo de procesamiento 214 constituyen un hilo. Un conjunto de hilos que se ejecutan a través del conjunto de motores de procesamiento paralelo es un grupo de hilos. Un grupo de hilos ejecuta el mismo programa sobre 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 encontrarse en espera durante ciclos en los que se está procesando ese grupo de hilos. Un grupo de hilos puede incluir también 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 durante ciclos de reloj consecutivos. En una realización, pueden ejecutarse múltiples grupos de hilos 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 caché interna y usar una memoria caché (por ejemplo, la caché de L1 308) dentro del grupo de procesamiento 214. Cada multiprocesador de gráficos 234 también tiene acceso a cachés L2 dentro de las unidades de partición (por ejemplo, unidades de partición 220A-220N de la Figura 2) que se comparten entre todos los grupos de procesamiento 214 y se pueden usar para transferir datos entre hilos. El multiprocesador de gráficos 234 también puede acceder a la memoria global fuera del chip, que puede incluir una o más memorias de procesador paralelo local y/o memoria del sistema. Puede usarse cualquier memoria externa a la unidad de procesamiento paralelo 202 como memoria global. Las realizaciones en las que el grupo de procesamiento 214 incluye múltiples instancias del multiprocesador de gráficos 234 pueden compartir instrucciones y datos comunes, que pueden almacenarse en la caché L1 308.

Cada grupo de procesamiento 214 puede incluir una MMU 245 (unidad de gestión de memoria) que está configurada para asignar 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 2. La MMU 245 incluye un conjunto de entradas de tabla de páginas (PTE) que se utilizan para asignar una dirección virtual a una dirección física de un mosaico (más información sobre mosaico) y, opcionalmente, un índice de línea de caché. La MMU 245 puede incluir memorias intermedias de traducción adelantada (TLB) de dirección o cachés que pueden residir dentro del multiprocesador de gráficos 234 o la caché L1 o el grupo de procesamiento 214. La dirección física se procesa para distribuir la localidad de acceso a los datos de superficie para permitir un entrelazado eficiente de solicitudes entre unidades de partición. El índice de línea de caché puede usarse para determinar si una solicitud para una línea de caché es un acierto o un fallo.

En aplicaciones de gráficos e informática, se puede configurar un grupo de procesamiento 214 de modo 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 muestras de textura, leer datos de textura y filtrar los datos de textura. Los datos de textura se leen desde una caché L1 de textura interna (no mostrada) o, en algunas realizaciones, desde la caché L1 dentro del multiprocesador de gráficos 234 y se extraen desde una caché L2, memoria de procesador paralelo local o memoria de sistema, según sea necesario. Cada multiprocesador de gráficos 234 emite tareas procesadas a la barra transversal de datos 240 para proporcionar la tarea procesada a otro grupo de procesamiento 214 para su procesamiento adicional o para almacenar la tarea procesada en una caché L2, memoria de procesador paralelo local o memoria de sistema mediante la barra transversal de memoria 216. Una preROP 242 (unidad de operaciones previa a la trama) está configurada para recibir datos del multiprocesador de gráficos 234, datos directos a unidades ROP, que pueden ubicarse con unidades de partición como se describe en el presente documento (por ejemplo, unidades de partición 220A-220N de la Figura 2). 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. Puede incluirse cualquier número de unidades de procesamiento, por ejemplo, el multiprocesador de gráficos 234, las unidades de texturas 236, las preROP 242, etc., dentro de un grupo de procesamiento 214. Además, aunque únicamente se muestra un grupo de procesamiento 214, la unidad de procesamiento paralelo, como se describe en el presente documento, puede incluir cualquier número de instancias del grupo de procesamiento 214. En una realización, cada grupo de procesamiento 214 puede configurarse para funcionar independientemente de otros grupos de procesamiento 214 usando unidades de procesamiento separadas y distintas, cachés L1, etc.

La **Figura 2D** muestra un multiprocesador de gráficos 234, de acuerdo con una realización. En tales realizaciones, el multiprocesador de gráficos 234 se acopla con el administrador de canalización 232 del grupo de procesamiento 214. El multiprocesador de gráficos 234 tiene un canal de ejecución que incluye, entre otros, una memoria caché de instrucciones 252, una unidad de instrucciones 254, una unidad de mapeo de direcciones 256, un archivo de registro 258, uno o más núcleos 262 de unidad de procesamiento de gráficos de propósito general (GPGPU) y un o más unidades de carga/almacenamiento 266. Los núcleos de GPGPU 262 y las unidades de carga/almacén 266 están acoplados con la memoria caché 272 y la memoria compartida 270 mediante una interconexión de memoria y caché 268.

En una realización, la caché de instrucciones 252 recibe un flujo de instrucciones para ejecutarse desde el gestor de canalizaciones 232. Las instrucciones se almacenan en caché en la caché de instrucciones 252 y se envían para su ejecución por la unidad de instrucciones 254. La unidad de instrucciones 254 puede despachar instrucciones como grupos de hilos (por ejemplo, envolventes), 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 del espacio de direcciones local, compartido o global, especificando una dirección dentro de un espacio de direcciones unificado. La unidad de mapeo de direcciones 256 se puede usar para traducir direcciones en el espacio de direcciones unificado en una dirección de memoria distinta a la que pueden acceder las unidades de carga/almacenamiento 266.

El archivo de registro 258 proporciona un conjunto de registros para las unidades funcionales del multiprocesador de gráficos 324. El archivo de registro 258 proporciona almacenamiento temporal para operandos conectados a las rutas de datos de las unidades funcionales (por ejemplo, núcleos GPGPU 262, unidades de carga/almacenamiento 266) del multiprocesador de gráficos 324. En una realización, el archivo de registro 258 se divide entre cada una de las unidades funcionales de manera que cada unidad funcional está asignada a una porción dedicada del archivo de registro 258. En una realización, el archivo de registro 258 se divide entre las diferentes envolventes que se ejecutan por el multiprocesador de gráficos 324.

Cada uno de los núcleos GPGPU 262 puede incluir unidades de punto flotante (FPU) y/o unidades lógicas aritméticas enteras (ALU) que se usan para ejecutar instrucciones del multiprocesador de gráficos 324. Los núcleos de GPGPU 262 pueden ser similares en arquitectura o pueden diferir en arquitectura, de acuerdo con las realizaciones. Por ejemplo, y en una realización, una primera porción de los núcleos GPGPU 262 incluye una FPU de precisión simple y una ALU entera, mientras que una segunda porción de los núcleos GPGPU incluye una FPU de doble precisión. En una realización, las FPU pueden implementar el estándar IEEE 754-2008 para aritmética de punto flotante o habilitar aritmética de punto 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 forma de realización, uno o más de los núcleos GPGPU puede incluir también lógica de función fija o especial.

La interconexión de memoria y caché 268 es una red de interconexión que conecta cada una de las unidades funcionales del multiprocesador de gráficos 324 al archivo de registro 258 y a la memoria compartida 270. En una realización, la interconexión de memoria y caché 268 es una interconexión de barra transversal que permite que la unidad de carga/almacén 266 implemente operaciones de carga y almacén entre la memoria compartida 270 y el archivo de registro 258. El archivo de registro 258 puede funcionar 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 registro 258 es de muy baja latencia. La memoria compartida 270 puede usarse para permitir la comunicación entre hilos que se ejecutan en las unidades funcionales dentro del multiprocesador de gráficos 234. La memoria caché 272 puede usarse como una caché de datos, por ejemplo, para almacenar en caché datos de textura comunicados entre las unidades funcionales y la unidad de texturas 236. La memoria compartida 270 puede usarse también como un programa gestionado almacenado en caché. Los hilos que se ejecutan en los núcleos GPGPU 262 pueden almacenar datos mediante programación dentro de la memoria compartida además de los datos almacenados en 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 ilustrados 325, 350 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 instrucción 332A-332B, el archivo de registro 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 cómputo (por ejemplo, núcleo GPGPU 336A-336B, núcleo GPGPU 337A-337B, núcleo 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 a través 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 instrucción, archivos de registro, núcleos GPGPU y unidades de almacenamiento de carga, como se ilustra en la Figura 2D y la Figura 3A. Los recursos de ejecución 356A-356D pueden funcionar en conjunto con la(s) unidad(es) de textura 360A-360D para operaciones de textura, mientras comparten una memoria caché de instrucciones 354 y una memoria compartida 362. En una 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 a través de una estructura de interconexión 352 similar a la estructura de interconexión 327 de la Figura 3A.

Los expertos en la técnica entenderán que la arquitectura descrita en las Figuras 1, 2A-2D y 3A-3B son descriptivas y no limitativas 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, incluidos, entre otros, uno o más procesadores de aplicaciones móviles, una o más unidades centrales de procesamiento (CPU) de escritorio o de servidor, incluidas CPU de múltiples núcleos, una o más unidades de procesamiento paralelas, tales como la unidad de procesamiento paralelo 202 de la Figura 2, así como uno o más procesadores gráficos o unidades de procesamiento de propósito especial, sin salirse del ámbito de las realizaciones descritas en el presente documento.

En algunas realizaciones, un procesador paralelo o GPGPU como se describe en el presente documento está acoplado comunicativamente a núcleos de anfitrión/procesador para acelerar operaciones gráficas, 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 integrarse en el mismo paquete o chip que los núcleos y estar acoplada de manera comunicativa a los núcleos a través de un bus/interconexión de procesador interno (es decir, internamente al paquete o chip). Independientemente de la manera en la que esté conectada la GPU, los núcleos de procesador pueden asignar trabajo a la GPU en forma de secuencias de comandos/instrucciones contenidas en un descriptor de trabajo. Luego, la GPU utiliza circuitos/lógica dedicados para procesar eficientemente estos comandos/instrucciones.

Técnicas para interconexión de GPU a procesador de anfitrión

La **Figura 4A** ilustra una arquitectura ilustrativa en la que una pluralidad de GPU 410-413 están acopladas comunicativamente a una pluralidad de procesadores multinúcleo 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 soportan un rendimiento 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 invención no están limitados a ningún protocolo o rendimiento de comunicación particular.

Además, en una forma de realización, dos o más de las GPU 410-413 se interconectan 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. Del mismo modo, dos o más de los procesadores multi-núcleo 405-406 se pueden conectar a través de enlaces de alta velocidad 433 que pueden ser buses multi-procesador simétricos (SMP) operando a 20GB/s, 30GB/s, 120GB/s o más. Alternativamente, toda la comunicación entre los diversos componentes del sistema que se muestran en la **Figura 4A** se puede lograr utilizando los mismos protocolos/enlaces (por ejemplo, a través de una estructura de interconexión común). Sin embargo, como se mencionó, los principios subyacentes de la invención no se limitan a ningún tipo particular de tecnología de interconexión.

En una forma de realización, cada procesador multinúcleo 405-406 se acopla con capacidad de comunicación a una memoria de procesador 401-402, por medio de interconexiones de memoria 430-431, respectivamente, y cada GPU 410-413 se acopla con capacidad de comunicación 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 tecnologías de acceso de memoria, o unas diferentes. A modo de ejemplo, y no como limitación, las memorias de procesador 401-402 y las memorias de GPU 420-423 pueden ser memorias volátiles, tal como memorias de acceso aleatorio dinámicas (DRAM) (que incluyen DRAM apiladas), SDRAM DDR de gráficos (GDDR) (por ejemplo, GDDR5, GDDR6), o memoria de ancho de banda alto (HBM) y/o pueden ser memorias no volátiles, tales como 3D XPoint o Nano-Ram. En una realización, alguna porción de las memorias puede ser memoria volátil y otra porción puede ser memoria no volátil (por ejemplo, usando una jerarquía de memoria de dos niveles (2LM)).

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

La **Figura 4B** ilustra detalles adicionales para una interconexión entre un procesador multinúcleo 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 GPU integrados en una tarjeta de línea que está acoplada al procesador 407 a través del enlace de alta velocidad 440. Alternativamente, el módulo de aceleración de gráficos 446 puede integrarse en el mismo paquete o chip que el procesador 407.

El procesador ilustrado 407 incluye una pluralidad de núcleos 460A-460D, cada uno con una memoria intermedia de búsqueda de traducción 461A-461D y uno o más cachés 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 invención (por ejemplo, unidades de búsqueda de instrucciones, unidades de predicción de bifurcación, decodificadores, unidades de ejecución, memorias intermedias de reordenamiento, etc.). Las cachés 462A-462D pueden comprender cachés de nivel 1 (L1) y de nivel 2 (L2). Además, una o más 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 caché L1, doce cachés L2 compartidas y doce cachés L3 compartidas. En esta realización, una de las cachés L2 y L3 es compartida 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 almacenadas en las diversas memorias caché 462A-462D, 456 y la memoria del sistema 441 mediante comunicación entre núcleos a través de un bus de coherencia 464. Por ejemplo, cada caché puede tener una lógica/circuitería de coherencia de caché asociada con la misma para 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 caché. Las técnicas de fisgoneo/coherencia de caché son bien conocidas por los expertos en la materia y no se describirán en detalle en el presente caso para evitar complicar los principios subyacentes de la invención.

En una realización, un circuito intermediario 425 acopla de manera comunicativa el módulo de aceleración de gráficos 446 al bus de coherencia 464, permitiendo que el módulo de aceleración de gráficos 446 participe en el protocolo de coherencia de caché como un homólogo de los núcleos. En particular, una interfaz 435 proporciona conectividad al circuito intermediario 425 a través del enlace de alta velocidad 440 (por ejemplo, un bus PCIe, NVLink, etc.) y una interfaz 437 conecta el módulo de aceleración de gráficos 446 al enlace 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 interrupciones en nombre de una pluralidad de motores de procesamiento de gráficos 431, 432, N del módulo de aceleración de gráficos 446. Cada uno de los motores de procesamiento de gráficos 431, 432, N puede comprender una unidad de procesamiento de gráficos (GPU) separada. Alternativamente, los motores de procesamiento de gráficos 431, 432, N pueden comprender diferentes tipos de motores de procesamiento de gráficos dentro de una GPU, tales como unidades de ejecución de gráficos, motores de procesamiento de medios (por ejemplo, codificadores/descodificadores de vídeo), muestreadores y motores blit. En otras palabras, el módulo de aceleración de gráficos puede ser una GPU con una pluralidad de motores de procesamiento de gráficos 431-432, N o los motores de procesamiento de gráficos 431-432, N pueden ser CPU individuales integradas en un paquete, tarjeta de línea o chip común.

En una realización, el circuito de integración del acelerador 436 incluye una unidad de gestión de memoria (MMU) 439 para realizar diversas funciones de gestión de memoria tales como traducciones de memoria virtual a física (también denominadas traducciones de memoria efectiva a real) y protocolos de acceso a memoria para acceder a la memoria del sistema 441. La MMU 439 puede incluir también una memoria intermedia de traducción adelantada (TLB) (no mostrada) para almacenar en caché las traducciones de dirección virtual/efectiva a física/real. En una implementación, una caché 438 almacena comandos y datos para un acceso eficiente por parte de los motores de procesamiento de gráficos 431-432, N. En una realización, los datos almacenados en la caché 438 y las memorias gráficas 433-434, N

se mantienen coherentes con las cachés centrales. 462A-462D, 456 y memoria del sistema 411. Como se mencionó, esto se puede lograr a través del circuito proxy 425 que participa en el mecanismo de coherencia de la caché en nombre de la caché 438 y las memorias 433-434, N (por ejemplo, enviando actualizaciones a la caché 438 relacionadas con modificaciones/accesos de líneas de caché en las cachés del procesador 462A-462D, 456 y recibiendo actualizaciones desde la 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 cambios de contextos (por ejemplo, cuando se guarda un primer hilo y se almacena un segundo hilo de modo que el segundo hilo pueda ejecutarse mediante un motor de procesamiento de gráficos). Por ejemplo, en un cambio de contexto, el circuito de gestión de contexto 448 puede almacenar valores de registro actuales en una región designada en la memoria (por ejemplo, identificada por un puntero de contexto). A continuación, puede restablecer los valores de registro cuando se vuelve al contexto. En una realización, un circuito de gestión de interrupciones 447 recibe y procesa interrupciones recibidas desde los dispositivos de sistema.

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

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

Debido a que los recursos de hardware de los motores de procesamiento de gráficos 431-432, N se asignan explícitamente al espacio de direcciones real visto por el procesador principal 407, cualquier procesador principal puede direccionar estos recursos directamente usando un valor de dirección efectivo. 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 de modo que aparecen al sistema como unidades independientes.

Como se ha mencionado, 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 gráficas 433-434, M almacenan instrucciones y datos que son procesados por cada uno de los motores de procesamiento de gráficos 431-432, N. Las memorias gráficas 433-434, M pueden ser memorias volátiles tales como DRAM (incluidas DRAM apiladas), memoria GDDR (por ejemplo, GDDR5, GDDR6) o HBM, y/o pueden ser memorias no volátiles como 3D XPoint o Nano-Ram.

En una realización, para reducir el tráfico de datos a través del enlace 440, se usan técnicas de desvío para garantizar que los datos almacenados en las memorias de gráficos 433-434, M son datos que serán usados de la manera más frecuente por los motores de procesamiento de gráficos 431-432, N y preferentemente no serán usados por los núcleos 460A-460D (al menos, no con frecuencia). De manera similar, el mecanismo de desvío intenta mantener datos que necesitan los núcleos (y, preferentemente, no los motores de procesamiento de gráficos 431-432, N) dentro de las cachés 462A-462D, 456 de los núcleos y 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, se puede 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 rendimiento mayor dada su proximidad al bus de coherencia 462 y 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). Estos últimos pueden incluir modelos de programación controlados por el circuito de integración acelerador 436 y modelos de programación 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 aplicación única puede canalizar otras solicitudes de aplicación a los motores gráficos 431-432, N, proporcionando virtualización dentro de una VM/partición.

En los modelos de programación de proceso especializado, los motores de procesamiento de gráficos 431-432, N, pueden estar compartidos por múltiples subdivisiones de aplicación/VM. Los modelos compartidos requieren que un hipervisor de sistema virtualice los motores de procesamiento de gráficos 431-432, N para permitir el acceso de cada sistema operativo. Para sistemas de partición única sin hipervisor, los motores de procesamiento de gráficos 431-432, N son propiedad del 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 compartida, el módulo de aceleración de gráficos 446 o un motor de procesamiento de gráficos individual 431-432, N selecciona un elemento de proceso usando un manejador de proceso. En una realización, los elementos del proceso se almacenan en la memoria del sistema 411 y se pueden direccionar utilizando las técnicas de traducción de dirección efectiva a dirección real descritas en el presente documento. El manejador de proceso puede ser un valor específico de la implementación proporcionado al proceso de anfitrión cuando se registra su contexto con el motor de procesamiento de gráficos 431-432, N (es decir, llamando al software de sistema para añadir el elemento de proceso a la lista vinculada de elementos de proceso). Los 16 bits inferiores del manejador de proceso pueden ser el desplazamiento del elemento de proceso dentro de la lista vinculada de elementos de proceso.

La **Figura 4D** ilustra un corte de integración de acelerador ilustrativa 490. Como se usa en el presente documento, un "corte" comprende una porción específica de los recursos de procesamiento del circuito de integración del acelerador 436. El espacio de direcciones efectivas de aplicación 482 dentro de la memoria del sistema 411 almacena 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 de 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 482 de la aplicación.

El módulo de aceleración de gráficos 446 y/o los motores de procesamiento de gráficos individuales 431-432, N pueden compartirse por todos o un subconjunto de los procesos en el sistema. Las realizaciones de la invención incluyen una infraestructura para configurar el estado de proceso y enviar un WD 484 a un módulo de aceleración de gráficos 446 para empezar 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 individual 431. Debido a que el módulo de aceleración de gráficos 446 es de propiedad de un único proceso, el hipervisor inicializa el circuito de integración de acelerador 436 para la partición de propiedad y el sistema operativo inicializa el circuito de integración de acelerador 436 para el proceso de propiedad en el momento cuando se asigna el módulo de aceleración de gráficos 446.

En la operación, una unidad de extracción de WD 491 en el corte de integración de acelerador 490 extrae el siguiente WD 484 que incluye una indicación del trabajo que va a hacerse por uno de los motores de procesamiento de gráficos del módulo de aceleración de gráficos 446. Los datos del WD 484 pueden almacenarse en registros 445 y ser utilizados 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 las tablas 486 de segmentos/páginas dentro del espacio 485 de direcciones virtuales del sistema operativo. El circuito de gestión de interrupción 447 puede procesar eventos de interrupción 492 recibidos del módulo de aceleración de gráficos 446. Al realizar operaciones gráficas, una dirección efectiva 493 generada por un motor de procesamiento de gráficos 431-432, N es traducida a una dirección real por la MMU 439.

En una realización, se duplica el mismo conjunto de registros 445 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 puede incluirse en un corte de integración de acelerador 490. Se muestran los registros ilustrativos que pueden inicializarse por el hipervisor en la **Tabla 1.**

Tabla 1 - Registros inicializados por el hipervisor

1	Registro de control de corte
2	Puntero de área de procesos planificados de dirección real (RA)
3	Registro de anulación de máscara de autoridad
4	Desplazamiento de entrada de tabla de vectores de interrupción
5	Límite de entrada de la tabla de vectores de interrupción

6	Registro de estado
7	ID de partición lógica
8	Puntero de registro de utilización de acelerador de hipervisor de dirección real (RA)
9	Registro de descripción de almacenamiento

Los registros de ejemplo que pueden ser inicializados por el sistema operativo se muestran en la **Tabla 2.**

Tabla 2 - Registros inicializados por sistema operativo

1	Identificación de procesos y hilos
2	Puntero de guardar/restaurar 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 de un módulo de aceleración de gráficos 446 particular y/o motor de procesamiento de gráficos 431-432, N. Contiene toda la información que un motor de procesamiento de gráficos 431-432, N requiere para realizar su trabajo o puede ser un puntero a una ubicación de memoria donde la aplicación ha configurado una cola de comandos de trabajo para 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 real 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 compartida permiten que todos o un subconjunto de procesos de todas o un subconjunto de particiones en el sistema utilicen un módulo de aceleración de gráficos 446. Hay dos modelos de programación donde el módulo de aceleración de gráficos 446 se comparte por múltiples procesos y subdivisiones: compartido en cortes de tiempo y compartido dirigido a gráficos.

En este modelo, el hipervisor del 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 soporte virtualización por el hipervisor de sistema 496, el módulo de aceleración de gráficos 446 puede adherirse a 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 restauración de contexto. 2) El módulo de aceleración de gráficos 446 garantiza que la solicitud de trabajo de una aplicación se completará en un período de tiempo específico, incluido cualquier error de traducción, o el módulo de aceleración de gráficos 446 proporciona la capacidad de adelantarse al procesamiento del trabajo, 3) La aceleración de gráficos Al módulo 446 se le debe garantizar la equidad entre procesos cuando opera en el modelo de programación compartida dirigida.

En una realización, para el modelo compartido, se requiere que la aplicación 480 realice una llamada al sistema operativo 495 con un tipo de módulo de aceleración de gráficos 446, un descriptor de trabajo (WD), un valor de registro de máscara de autoridad (AMR) y un guardado de contexto. /puntero de área de restauración (CSRP). El tipo de módulo de aceleración de gráficos 446 describe la función de aceleración dirigida para la llamada de sistema. El tipo del módulo de aceleración de gráficos 446 puede ser un valor específico del sistema. Al WD se le da formato específicamente para el módulo de aceleración de gráficos 446 y puede estar en forma de un comando del módulo de aceleración de gráficos 446, un puntero de dirección eficaz a una estructura definida por el usuario, un puntero de dirección eficaz a una cola de comandos o cualquier otra estructura de datos para describir el trabajo que debe hacer el módulo de aceleración de gráficos 446. En una realización, el valor de AMR es el estado de AMR que hay que 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 soportan un Registro de Anulación de Máscara de Autoridad de Usuario (UAMOR), el sistema operativo puede aplicar el valor UAMOR actual al valor AMR antes de pasar el AMR en la llamada del hipervisor. El hipervisor 496 puede aplicar opcionalmente el valor de registro de anulación de máscara de autoridad (AMOR) actual antes de colocar el AMR en el elemento de proceso 483. En una realización, el CSRP es uno de los registros 445 que contiene 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 de contexto. Este puntero es opcional si no se requiere que se grabe estado entre trabajos o cuando se anticipa un trabajo. El área de guardado/restauración de contexto puede estar anclada en la memoria del sistema.

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

Tabla 3 - Parámetros de llamada de SO a hipervisor

1	Un descriptor de trabajo (VVD)
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. El hipervisor 496, a continuación, pone el elemento de proceso 483 en la lista de elementos de proceso vinculados 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 ID de partición 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 del corte 490 de integración del acelerador.

Como se ilustra en **Figura 4F,** una realización de la invención emplea una memoria unificada direccionable a través de un espacio de direcciones de memoria virtual común usado para acceder a las memorias físicas del procesador 401-402 y las memorias GPU 420-423. En esta implementación, las operaciones ejecutadas en las GPU 410-413 utilizan el mismo espacio de direcciones de memoria virtual/efectiva para acceder a las memorias de los procesadores 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 401 del procesador, una segunda porción a la memoria 402 del segundo procesador, una tercera porción a la memoria 420 de la GPU, y así sucesivamente. El espacio de memoria virtual/efectivo total (denominado, en ocasiones, el espacio de direcciones efectivo) está distribuido, de esta manera, a lo largo de cada una de las memorias de procesador 401-402 y de las memorias de GPU 420-423, permitiendo que cualquier procesador o GPU acceda a cualquier memoria física con una dirección virtual mapeada a esa memoria.

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

Una realización permite que la memoria conectada a la GPU 420-423 se asigne como parte de la memoria del sistema y se acceda a ella utilizando tecnología de memoria virtual compartida (SVM), pero sin sufrir los típicos inconvenientes de rendimiento asociados con la coherencia total de la memoria caché del sistema. La capacidad de que se acceda a

la memoria adjunta a la GPU 420-423 como memoria de sistema sin sobrecarga de coherencia de caché onerosa proporciona un entorno de operación beneficioso para la descarga de la GPU. Esta disposición permite que el software del procesador de anfitrión 405 establezca operandos y acceda a resultados de cómputo, sin la sobrecarga de las copias de datos de DMA de E/S tradicionales. Tales copias tradicionales implican llamadas de controlador, interrupciones y accesos de E/S correlacionados con memoria (MMIO) que son, todos ellos, ineficientes en relación con los accesos de memoria sencillos. Al mismo tiempo, la capacidad de acceder a la memoria anexada a GPU 420-423 sin sobrecargas de coherencia de la caché puede ser crítica para el tiempo de ejecución de un cómputo descargado. En casos con tráfico sustancial de memoria de escritura de transmisión por flujo continuo, por ejemplo, la sobrecarga de coherencia de la caché puede reducir significativamente el ancho de banda de escritura eficaz observado por una GPU 410-413. La eficiencia del establecimiento de operandos, la eficiencia del acceso a resultados y la eficiencia del cómputo de GPU desempeñan, todas ellas, un papel en la determinación de la eficacia de la descarga de GPU.

En una implementación, la selección entre el desvío de GPU y el desvío de procesador de anfitrión es controlada por una estructura de datos de rastreador de desvío. Se puede usar una tabla de polarización, por ejemplo, que puede ser una estructura granular de página (es decir, controlada en la granularidad de una página de memoria) que incluye 1 o 2 bits por página de memoria conectada a la GPU. La tabla de polarización se puede implementar en un rango de memoria robada de una o más memorias 420-423 conectadas a la GPU, con o sin una caché de polarización en la GPU 410-413 (por ejemplo, para almacenar en caché las entradas utilizadas con frecuencia/recientemente de la tabla de polarización). Como alternativa, toda la tabla de desvíos puede mantenerse dentro de la GPU.

En una implementación, se accede a la entrada de la tabla de polarización asociada con cada acceso a la memoria conectada 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 correspondiente memoria de GPU 420-423. Las solicitudes locales de la GPU que encuentran su página en el sesgo del anfitrión se reenvían al procesador 405 (por ejemplo, a través de un enlace de alta velocidad como se analizó anteriormente). En una realización, las solicitudes del procesador 405 que encuentran la página solicitada en la polarización del procesador principal completan la solicitud como una lectura de memoria normal. Alternativamente, las solicitudes dirigidas a una página orientada a GPU pueden reenviarse a la GPU 410-413. La GPU puede hacer entonces que la página realice una transición a un desvío de procesador de anfitrión si no está usando actualmente la página.

El estado de sesgo 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 de API (por ejemplo, OpenCL), que, a su vez, llama al controlador de dispositivos de la GPU que, a su vez, envía un mensaje a (o pone en cola un descriptor de comandos para) la GPU que le indica que cambie el estado de desvío y, para algunas transiciones, que realice una operación de vaciado de caché en el anfitrión. Se requiere la operación de vaciado de caché para una transición desde un desvío del procesador de anfitrión 405 a un desvío de GPU, pero no se requiere para la transacción opuesta.

En una realización, la coherencia de la caché se mantiene haciendo que las páginas orientadas a la GPU no puedan almacenarse en caché temporalmente por el procesador anfitrión 405. Para acceder a estas páginas, el procesador 405 puede solicitar acceso a la GPU 410, que puede otorgar o no acceso de inmediato, dependiendo de la implementación. Por tanto, para reducir la comunicación entre el procesador 405 y la GPU 410 es beneficioso garantizar que las páginas orientadas a la GPU sean aquellas que requiere la GPU pero no el procesador principal 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 el canal de procesamiento de gráficos 500 ilustrado. El procesador de gráficos puede incluirse dentro de los subsistemas de procesamiento paralelo como se describe en el presente documento, tal como el procesador paralelo 200 de la Figura 2, que, en una realización, es una variante de los procesadores paralelos 112 de la Figura 1. Los diversos sistemas de procesamiento paralelo pueden implementar la canalización de procesamiento de gráficos 500 a través de una o más instancias de la unidad de procesamiento paralelo (por ejemplo, la unidad de procesamiento paralelo 202 de la Figura 2) como se describe en el presente documento. Por ejemplo, una unidad de sombreado (por ejemplo, el multiprocesador de gráficos 234 de la Figura 3) puede configurarse para realizar las funciones de una o más de una unidad de procesamiento de vértices 504, una unidad de procesamiento de control de teselación 508, una unidad de procesamiento de evaluación de teselación 512, una unidad de procesamiento de geometría unidad de procesamiento 516, y una unidad de procesamiento de fragmentos/píxeles 524. Las funciones del ensamblador de datos 502, los ensambladores primitivos 506, 514, 518, la unidad de teselación 510, el rasterizador 522 y la unidad de operaciones de ráster 526 también pueden ser realizadas por otros motores de procesamiento dentro de un grupo de procesamiento (por ejemplo, el grupo de procesamiento 214 de la Figura 3) y una unidad de partición correspondiente (por ejemplo, unidad de partición 220A-220N de la Figura

2). La canalización de procesamiento de gráficos 500 también puede implementarse usando unidades de procesamiento dedicadas 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 porciones del proceso de procesamiento de gráficos 500 pueden acceder a la memoria en el chip (por ejemplo, la memoria del procesador paralelo 222 como en la Figura 2) a través de una interfaz de memoria 528, que puede ser una instancia de la interfaz de memoria 218 de la Figura 2.

En una realización, el ensamblador de datos 502 es una unidad de procesamiento que recopila datos de vértices para superficies y primitivas. El ensamblador de datos 502 luego envía los datos de vértice, incluidos los atributos de vértice, a la unidad de procesamiento de vértice 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 están almacenados en la memoria caché, local o del sistema para su uso en el procesamiento de los datos de vértice y puede programarse para transformar los datos de vértice de una representación de coordenadas basada en objetos a un espacio de coordenadas de espacio mundial o un espacio de coordenadas de dispositivo normalizado.

Una primera instancia de un ensamblador de primitivas 506 recibe atributos de vértice desde la unidad de procesamiento de vértices 50. El ensamblador de primitivas 506 lee atributos de vértices 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 y así sucesivamente, según sea soportado por diversas 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 desde 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 superficies por parte de la unidad de procesamiento de evaluación de teselación 512. La unidad de procesamiento de control de teselación 508 también puede computar factores de teselación para bordes de parches geométricos. Se aplica un factor de teselación a un único borde y cuantifica un nivel dependiente de la vista del detalle 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 desde la unidad de procesamiento de evaluación de teselación 512, que lee los atributos de vértices almacenados según sea necesario y construye primitivas de gráficos para su procesamiento por la unidad de procesamiento de geometría 516. La unidad de procesamiento de geometría 516 es una unidad de ejecución programable que ejecuta programas de sombreado de geometría para transformar primitivas de gráficos recibidas desde el ensamblador de primitivas 514 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 parámetros usados para rasterizar las nuevas primitivas de gráficos.

En algunas realizaciones, la unidad de procesamiento de geometría 516 puede añadir o borrar elementos en el flujo de geometría. La unidad de procesamiento de geometría 516 envía los parámetros y vértices que especifican nuevas primitivas gráficas al ensamblador primitivo 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 mediante 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 de procesador paralelo o en la memoria de sistema para su uso en el procesamiento de los datos de geometría. La unidad de escala, selección y recorte de ventana gráfica 520 realiza recorte, selección y escalado de ventana gráfica y genera primitivas gráficas procesadas a un rasterizador 522. El rasterizador 522 puede realizar optimizaciones de selección de profundidad y otras basadas en la profundidad. El rasterizador 522 también realiza la conversión de exploración en las nuevas primitivas de gráficos para generar fragmentos y emitir aquellos fragmentos y datos de cobertura asociados a la unidad de procesamiento de fragmentos/píxeles 524. El escaneo del rasterizador 522 convierte las nuevas primitivas gráficas y envía datos de fragmentos y cobertura a la unidad de procesamiento de fragmentos/píxeles 524.

La unidad de procesamiento de píxeles de fragmentos 524 es una unidad de ejecución programable que está configurada para ejecutar programas de sombreado de fragmentos o programas de sombreado de píxeles. La unidad de procesamiento de fragmentos/píxeles 524 transforma fragmentos o píxeles recibidos 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, pero sin limitación, mapeo de textura, sombreado, mezcla, corrección de textura y corrección de perspectiva para producir fragmentos o píxeles sombreados que se emiten a una unidad de operaciones de rasterización 526. La unidad de procesamiento de

fragmentos/píxeles 524 puede leer datos que se almacenan en cualquiera de la memoria de procesador paralelo o la memoria de sistema para su uso cuando se procesan los datos de fragmento. Los programas de sombreador de fragmentos o de píxeles pueden estar configurados para sombrear a granularidad de muestra, de píxel, de pieza u otras dependiendo de las tasas de muestreo configuradas para las unidades de procesamiento.

La unidad de operaciones de trama 526 es una unidad de procesamiento que realiza operaciones de trama que incluyen, entre otras, plantilla, prueba z, combinación y similares, y genera datos de píxeles como datos de gráficos procesados para almacenarlos en la memoria de gráficos (por ejemplo, memoria de procesador paralelo 222 como en la Figura 1, para ser mostrado en uno o más dispositivos de visualización 110 o para procesamiento adicional por uno de 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 una realización de un dispositivo informático 600 que emplea un mecanismo de procesamiento de matriz dispersa. El dispositivo informático 600 (por ejemplo, dispositivos portátiles inteligentes, dispositivos de realidad virtual (VR), pantallas montadas en la cabeza (HMD), ordenadores móviles, dispositivos de Internet de las cosas (IoT), ordenadores portátiles, ordenadores de escritorio, ordenadores servidor, etc.) pueden ser lo mismo que el sistema de procesamiento de datos 100 de la Figura 1 y, en consecuencia, por brevedad, claridad y facilidad de comprensión, muchos de los detalles indicados anteriormente con referencia a las Figuras 1 - 5 no se analizan ni se repiten en lo sucesivo. Como se ilustra, en una realización, el dispositivo informático 600 se muestra alojando un mecanismo de procesamiento de matriz dispersa 610.

Como se ilustra, en una realización, el mecanismo de procesamiento de matriz dispersa 610 puede alojarse en la GPU 614. Sin embargo, en otras realizaciones, el mecanismo de procesamiento de matriz dispersa 610 puede alojarse en el controlador de gráficos 616. Aún en otras realizaciones, el mecanismo de procesamiento de matriz dispersa 610 puede estar alojado en o parte del firmware de la unidad central de procesamiento ("CPU" o "procesador de aplicaciones") 612. Por brevedad, claridad y facilidad de comprensión, a lo largo del resto de este documento, el mecanismo de procesamiento de matriz dispersa 610 puede analizarse como parte del controlador de gráficos 616; sin embargo, las realizaciones no están limitadas como tales.

En otra realización más, el mecanismo de procesamiento de matriz dispersa 610 puede alojarse como lógica de software o firmware mediante el sistema operativo 606. Aún en una realización adicional, el mecanismo de procesamiento de matriz dispersa 610 puede ser alojado parcial y simultáneamente por múltiples componentes del dispositivo informático 600, tales como uno o más de controlador de gráficos 616, GPU 614, firmware de GPU, CPU 612, firmware de CPU, sistema operativo 606, y/o similares. Se contempla que el mecanismo de procesamiento de matriz dispersa 610 o uno o más de sus componentes puedan implementarse como hardware, software y/o firmware.

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

El dispositivo informático 600 puede incluir cualquier número y tipo de dispositivos de comunicación, tales como grandes sistemas informáticos, tales como ordenadores servidores, ordenadores de escritorio, etc., y puede incluir además decodificadores (por ejemplo, decodificadores de televisión por cable basados en Internet, etc.), dispositivos basados en el sistema de posicionamiento global (GPS), etc. El dispositivo informático 600 puede incluir dispositivos informáticos móviles que sirven como dispositivos de comunicación, tales como teléfonos móviles que incluyen teléfonos inteligentes, asistentes digitales personales (PDA), tabletas, ordenadores portátiles, lectores electrónicos, televisores inteligentes, plataformas de televisión, dispositivos portátiles (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 diversos 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) GPU 614, controlador de gráficos (también denominado "controlador de GPU", "lógica del controlador de gráficos", "lógica de controlador", controlador en modo de usuario (UMD), UMD, marco de controlador en modo de usuario (UMDF), UMDF, o simplemente "controlador") 616, CPU 612, memoria 608, dispositivos de red, controladores o similares, así como fuentes de entrada/salida (E/S) 604, tales como pantallas táctiles, paneles táctiles, almohadillas táctiles, teclados virtuales o normales, ratones, puertos, conectores, etc.

El dispositivo informático 600 puede incluir un sistema operativo (OS) 606 que sirve como interfaz entre el hardware y/o los recursos físicos del dispositivo informático 600 y un usuario. Se contempla que la CPU 612 puede incluir uno o

más procesadores, tales como el(los) procesador(es) 102 de la Figura 1, mientras que la GPU 614 puede incluir uno o más procesadores gráficos (o multiprocesadores).

Cabe señalar que términos como "nodo", "nodo informático", "servidor", "dispositivo servidor", "ordenador en la nube", "servidor en la nube", "ordenador servidor en la nube", "máquina", "máquina anfitriona", "dispositivo", "dispositivo informático", "ordenador", "sistema informático" y similares, pueden usarse indistintamente 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 indistintamente a lo largo de este documento. Además, términos como "trabajo", "entrada", "solicitud", "mensaje" y similares se pueden utilizar indistintamente en este documento.

Se contempla, y como se describe más detalladamente con referencia a las Figuras 1-5, que algunos procesos del proceso de gráficos descritos anteriormente se implementan en software, mientras que el resto se implementa en hardware. Se puede implementar una canalización de gráficos en un diseño de coprocesador de gráficos, donde la CPU 612 está diseñada para funcionar con la GPU 614 que puede incluirse o ubicarse conjuntamente con la CPU 612. En una realización, la GPU 614 puede emplear cualquier número y tipo de lógica de software y hardware convencional para realizar las funciones convencionales relacionadas con la representación de gráficos, así como lógica de software y hardware novedosa para ejecutar cualquier número y tipo de instrucciones.

Como se mencionó anteriormente, la memoria 608 puede incluir una memoria de acceso aleatorio (RAM) que comprende una base de datos de aplicaciones que tiene información de objetos. Un concentrador controlador de memoria, tal como el concentrador de memoria 105 de la Figura 1, puede acceder a los datos en la RAM y reenviarlos a la GPU 614 para el procesamiento del canal de gráficos. La RAM puede incluir RAM de velocidad de datos doble (RAM DDR), RAM de salida de datos extendida (RAM EDO), etc. La CPU 612 interactúa con una canalización de gráficos de hardware para compartir la funcionalidad de canalización de gráficos.

Los datos procesados se almacenan en una memoria intermedia en la tubería de gráficos de hardware y la información de estado se almacena en la memoria 608. La imagen resultante luego se transfiere a fuentes de E/S 604, tales como un componente de visualización para visualizar la imagen. Se contempla que el dispositivo de visualización pueda ser de varios tipos, tales como tubo de rayos catódicos (CRT), transistor de película delgada (TFT), pantalla de cristal líquido (LCD), conjunto de diodos orgánicos emisores de luz (OLED), etc., para visualizar información a un usuario.

La memoria 608 puede comprender una región preasignada de una memoria intermedia (por ejemplo, memoria intermedia de cuadros); sin embargo, un experto en la técnica debería entender que las realizaciones no están tan limitadas y que se puede utilizar cualquier memoria accesible al canal de gráficos inferior. El dispositivo informático 600 puede incluir además un concentrador de control (ICH) 107 de entrada y salida (E/S) como se hace referencia en la Figura 1, como una o más fuentes de E/S 604, etc.

La CPU 612 puede incluir uno o más procesadores para ejecutar instrucciones con el fin de realizar cualquier rutina de software que implemente el sistema informático. Las instrucciones frecuentemente implican algún tipo de operación realizada sobre los datos. Tanto los datos como las instrucciones pueden almacenarse en la memoria del sistema 608 y en cualquier caché asociada. La memoria caché normalmente está diseñada para tener tiempos de latencia más cortos que la memoria del sistema 608; por ejemplo, la memoria caché podría integrarse en el(los) mismo(s) chip(es) de silicio que el(los) procesador(es) y/o construirse con celdas de RAM estática (SRAM) más rápidas, mientras que la memoria del sistema 608 podría construirse con celdas de RAM dinámica (DRAM) más lentas. Al tender a almacenar instrucciones y datos utilizados con mayor frecuencia en la memoria caché en lugar de en la memoria del sistema 608, mejora la eficiencia del rendimiento general del dispositivo informático 600. Se contempla que en algunas realizaciones, la GPU 614 puede existir como parte de la CPU 612 (tal como parte de un paquete físico de CPU), en cuyo caso, la memoria 608 puede ser compartida por la CPU 612 y la GPU 614 o mantenerse separada.

La memoria del sistema 608 puede estar disponible para otros componentes dentro del dispositivo informático 600. Por ejemplo, cualquier dato (por ejemplo, datos de gráficos de entrada) recibido desde varias interfaces al dispositivo informático 600 (por ejemplo, teclado y ratón, puerto de impresora, puerto de red de área local (LAN), puerto de módem, etc.) o recuperado de un puerto interno El elemento de almacenamiento del dispositivo informático 600 (por ejemplo, unidad de disco duro) a menudo se pone en cola temporalmente en la memoria del sistema 608 antes de ser operado por uno o más procesadores en la implementación de un programa de software. De manera similar, los datos que un programa de software determina que deben enviarse desde el dispositivo informático 600 a una entidad externa a través de una de las interfaces del sistema informático, o almacenarse en un elemento de almacenamiento interno, a menudo se ponen en cola temporalmente en la memoria del sistema 608 antes de ser transmitidos o almacenados.

Además, por ejemplo, se puede usar un ICH para garantizar que dichos datos se pasen adecuadamente entre la memoria del sistema 608 y su interfaz de sistema informático correspondiente apropiada (y el dispositivo de almacenamiento interno si el sistema informático está diseñado así) y puede tener enlaces punto a punto bidireccionales entre sí mismo y las fuentes/dispositivos de E/S observados 604. De manera similar, se puede usar un MCH para gestionar las diversas solicitudes en competencia para los accesos a la memoria del sistema 608 entre la

CPU 612 y la GPU 614, interfaces y elementos de almacenamiento interno que pueden surgir aproximadamente en el tiempo entre sí.

Las fuentes de E/S 604 pueden incluir uno o más dispositivos de E/S que se implementan para transferir datos hacia y/o desde el dispositivo informático 600 (por ejemplo, un adaptador de red); o, para un almacenamiento no volátil a gran escala dentro del dispositivo informático 600 (por ejemplo, unidad de disco duro). Se puede usar un dispositivo de entrada de usuario, incluidas claves alfanuméricas y de otro tipo, para comunicar información y ordenar selecciones a la GPU 614. Otro tipo de dispositivo de entrada de usuario es el control del cursor, tal como un ratón, una bola de seguimiento, una pantalla táctil, un panel táctil o teclas de dirección del cursor para comunicar información de dirección y selecciones de comandos a la GPU 614 y para controlar el movimiento del cursor en el dispositivo de visualización. Se pueden emplear conjuntos de cámaras y micrófonos del dispositivo informático 600 para observar gestos, grabar audio y vídeo y recibir y transmitir comandos visuales y de audio.

El dispositivo informático 600 puede incluir además interfaz(es) de red para proporcionar acceso a una red, tal como una LAN, una red de área amplia (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. Las interfaces de red pueden incluir, por ejemplo, una interfaz de red inalámbrica que tenga una antena, que puede representar una o más antenas. Las interfaces de red también pueden incluir, por ejemplo, una interfaz de red cableada para comunicarse con dispositivos remotos a través 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.

Las interfaces de red pueden proporcionar acceso a una LAN, por ejemplo, conforme a los estándares IEEE 802.11b y/o IEEE 802.11g, y/o la interfaz de red inalámbrica puede proporcionar acceso a una red de área personal, por ejemplo, conforme según los estándares Bluetooth. También pueden ser compatibles otras interfaces y/o protocolos de red inalámbrica, incluidas versiones anteriores y posteriores de los estándares. Además de, o en lugar de, la comunicación a través de los estándares de LAN inalámbrica, las interfaces de red pueden proporcionar comunicación inalámbrica utilizando, por ejemplo, protocolos de división de tiempo, acceso múltiple (TDMA), protocolos de sistemas globales para comunicaciones móviles (GSM), protocolos de código de división, protocolos de Acceso Múltiple (CDMA) y/o cualquier otro tipo de protocolos de comunicaciones inalámbricas.

Las interfaces de red pueden incluir una o más interfaces de comunicación, como un módem, una tarjeta de interfaz de red u otros dispositivos de interfaz conocidos, como los utilizados para el acoplamiento a Ethernet, Token Ring u otros tipos de dispositivos físicos cableados o accesorios inalámbricos con el fin de proporcionar un enlace de comunicación para soportar una LAN o una WAN, por ejemplo. De esta manera, el sistema informático también puede estar acoplado a una serie de dispositivos periféricos, clientes, superficies de control, consolas o servidores a través de un infraestructura de red convencional, incluida, por ejemplo, una Intranet o Internet.

Debe apreciarse que para ciertas 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. Ejemplos del dispositivo electrónico o sistema informático 600 pueden incluir (sin limitación) un dispositivo móvil, un asistente digital personal, un dispositivo informático móvil, un teléfono inteligente, un teléfono celular, un auricular, un buscapersonas unidireccional, un buscapersonas bidireccional, un dispositivo de mensajería, un ordenador, un ordenador personal (PC), un ordenador de escritorio, una ordenador portátil, un ordenador manual, un ordenador portátil, una tableta, un servidor, una matriz de servidores o granja de servidores, un servidor web, un servidor de red, un servidor de Internet, una estación de trabajo, un miniordenador, una ordenador principal, un superordenador, un dispositivo de red, un dispositivo web, un sistema informático distribuido, sistemas multiprocesador, sistemas basados en procesadores, electrónica de consumo, electrónica de consumo programable, televisión, televisión digital, decodificador, punto de acceso inalámbrico, estación base, estación de abonado, centro de abonado móvil, controlador de red de radio, enrutador, concentrador, puerta de enlace, puente, conmutador, máquina o combinaciones de los mismos.

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). El término "lógica" puede incluir, a modo de ejemplo, software o hardware y/o combinaciones de software y hardware.

Se pueden proporcionar realizaciones, por ejemplo, como un producto de programa informático que puede incluir uno o más medios legibles por máquina que tienen almacenadas instrucciones ejecutables por máquina que, cuando se ejecutan por una o más máquinas tales como un ordenador, una red de ordenadores u otras dispositivos electrónicos, pueden dar como resultado que una o más máquinas realicen operaciones de acuerdo con las realizaciones descritas en el presente documento. Un medio legible por máquina puede incluir, entre otros, disquetes, discos ópticos, CD-ROM (memorias de disco compacto de sólo lectura) 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 una ordenador remota (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 modulado 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).

Las operaciones de multiplicación de matrices dispersas son importantes en diversas aplicaciones, incluidas las redes neuronales. Una matriz dispersa es una matriz en la que la mayoría de los elementos son cero (o algún otro valor matemáticamente irrelevante). Las matrices dispersas suelen ser el resultado de datos de imagen recibidos que indican que una imagen (o región de imagen) incluye información que no es útil. Una multiplicación de matrices generalmente se realiza utilizando un método bloqueado, como se muestra en la **Figura 7A.** Por lo tanto, una GPU tradicional toma dos bloques de matriz de entrada como entradas y produce un bloque de matriz de salida. Sin embargo, cuando se opera con matrices dispersas, estos bloques de entrada incluyen en su mayoría valores cero que no contribuyen a los resultados acumulados en la matriz de salida (por ejemplo, multiplicar por cero produce cero). Según una realización, el mecanismo 610 de procesamiento de matrices dispersas incluye un programador 613 que identifica dinámicamente operandos que tienen valores cero en una matriz que se está procesando.

La **Figura 7B** ilustra una realización de dicho programador 613 incluido dentro de un elemento de procesamiento de GPU 700. Como se muestra en la **Figura 7B,** El elemento de procesamiento 700 incluye lógica 701 para leer operandos incluidos en instrucciones recibidas. También se incluyen la unidad de cómputo 702 y la lógica de resultados de escritura 703. En una realización, el programador 613 detecta e identifica ubicaciones de memoria de operandos que tienen valores cero. En tal realización, el programador 613 recupera valores de operandos almacenados de la memoria (o caché) a medida que se reciben instrucciones en la GPU 614.

Una vez recuperado, se determina si el valor de un operando es cero. Tras la determinación de que el valor de un operando es cero, el programador 613 impide la programación de multiplicación de esos operandos en la unidad de multiplicación 702. En consecuencia, sólo se programan y procesan operandos distintos de cero en la unidad de cómputo 702, mientras que el programador 703 escribe un valor cero para escribir la lógica de resultados 703 para operandos cero. Aunque se muestra que reside dentro de la lógica 701, otras realizaciones pueden presentar un programador 703 que reside externo a la lógica 701.

En una realización adicional, el mecanismo de procesamiento de matriz dispersa 610 incluye además un rastreador de patrones dispersos 615 para detectar uno o más segmentos dispersos de datos (por ejemplo, patrones de dispersión) dentro de un bloque de datos almacenado, y utiliza los patrones para convertir cómputos de matrices potencialmente densas en cómputos escasos. En una realización, el rastreador de patrones dispersos 615 detecta patrones de dispersión en datos (por ejemplo, datos de imágenes) que están almacenados en la memoria/caché.

Se espera que los futuros sistemas de aprendizaje profundo almacenen miles de millones de imágenes que serán procesadas. Normalmente, una imagen se puede dividir en segmentos que representan información útil y otra que no es importante. Por ejemplo, si la mitad de la imagen está vacía, es posible rastrear esta información en el nivel de memoria (por ejemplo, a través de un controlador de memoria), en el nivel de página (en el sistema operativo) o en el nivel de jerarquía de caché). Esta información es útil durante la ejecución de una aplicación para eliminar la realización de operaciones de cómputo para segmentos vacíos sin importancia (o dispersos).

La **Figura 7C** ilustra una realización de un rastreador de patrones dispersos 615, que incluye lógica de reconocimiento de patrones 708 y segmentos dispersos (o lógica de segmentos) 709. Según una realización, la lógica de reconocimiento de patrones 708 realiza una operación de cuadro delimitador en un bloque de datos (por ejemplo, datos de imágenes) paginando datos de imágenes almacenados en la memoria para determinar una similitud de varios segmentos dentro del bloque de datos. Los segmentos de datos dentro de un cuadro delimitador que tienen el mismo valor pueden considerarse datos dispersos.

En una realización, la lógica de reconocimiento de patrones 708 se coordina con un controlador de memoria para rastrear los datos almacenados en un dispositivo de memoria. En otras realizaciones, la lógica de reconocimiento de patrones 708 rastrea la información en el nivel de jerarquía de caché. Aún en otras realizaciones, la lógica de reconocimiento de patrones 708 rastrea la información en el nivel de tabla de páginas a través del sistema operativo 606. En una realización adicional, la lógica de reconocimiento de patrones 708 puede implementarse para analizar grandes volúmenes de datos densos para determinar segmentos que pueden procesarse como operaciones dispersas. Como resultado, la lógica de segmento 709 registra las ubicaciones de dirección de segmentos dispersos identificados por la lógica de reconocimiento de patrones 708. En una realización, los segmentos dispersos 709 comprenden punteros a los componentes del segmento disperso. Como se analizó anteriormente, las multiplicaciones de matrices para operaciones dispersas se pueden omitir, reduciendo así la carga de procesamiento en la GPU 614.

Aún en una realización adicional, el mecanismo de procesamiento de matrices dispersas 610 incluye lógica de compresión 617 para comprimir matrices dispersas. En tal realización, las representaciones de matrices dispersas

comprimidas se generan dinámicamente basándose en un índice disperso (por ejemplo, definido por un % de entradas distintas de cero en la matriz). En esta realización, el formato comprimido de matriz dispersa se puede representar con un valor distinto de cero señalado por el índice de fila y columna.

Según una realización, la lógica de compresión 617 recibe el segmento disperso determinado por la lógica de reconocimiento de patrones 708 y determina si los datos cumplen un umbral predeterminado para ser considerados dispersos. Por ejemplo, una matriz MxN puede considerarse escasa si se determina que un número Y de entradas dentro de la matriz son valores cero. La lógica de compresión 617 comprime matrices que se determina que son dispersas y almacena las matrices comprimidas en una memoria intermedia comprimida dispersa para su ejecución en la GPU 614.

La **Figura 7D** ilustra una realización de la GPU 614 que incluye una memoria intermedia comprimido disperso 712 y una pluralidad de unidades de ejecución (EU) 710. En una realización, la memoria intermedia 712 comprimida dispersa incluye entradas de almacenamiento de matriz dispersa comprimidas 712(0) - 712(n) que son procesadas por los EU 710. En tal realización, la lógica de compresión 717 almacena matrices dispersas utilizadas con frecuencia. Antes del procesamiento por parte de los EU 710, la lógica de compresión 617 descomprime una matriz comprimida nuevamente a su formato original. En una realización, todos los EU 710 pueden usar las mismas matrices comprimidas para los hilos que se van a ejecutar. Sin embargo, en otras realizaciones, cada EU 710 puede usar una matriz dispersa única para los cómputos. Por lo tanto, las matrices dispersas a las que se accede con frecuencia se almacenan y leen localmente para evitar la transmisión de datos desde la memoria caché a través de una interconexión larga.

La GPU 614 se puede implementar para realizar otras operaciones de aprendizaje profundo. Por ejemplo, la GPU 614 puede realizar procesamiento de capas de redes neuronales. Un patrón que se ejecuta con frecuencia en casi todas las redes neuronales profundas es que una capa de convolución (C) sigue una capa de polarización (B) seguida de una capa de unidad lineal rectificada (ReLu (R)) seguida de una capa de grupo (P). Hoy en día, la mayoría de los sistemas ejecutan estas capas una tras otra (por ejemplo, en las GPU, C, B, R y P se asignan como núcleos individuales) o se asignan fusionando CBR seguido de P como dos núcleos separados.

En ambos escenarios, se necesita más de una invocación del kernel; incurriendo así en gastos generales de transferencia de datos adicionales. Según una realización, la GPU 614 está configurada de manera que las UE se particionan y se asignan para realizar ciertas operaciones, y se reenvían resultados intermedios entre ellas para lograr un alto rendimiento. La **Figura 7E** ilustra una realización de la GPU 614 que tiene EU particionadas 720.

Como se muestra en la **Figura 7E,** los EU 720(1) - 720(10) se asignan para ejecutar hilos de capa de convolución, mientras que los EU 720(11) - 720(13), EU 720(14) - 720(16) y EU 720(17) - 720(19)) realizan, sesgan, ReLu y ponen en común, respectivamente, la ejecución de hilos en capas. Además, se reenvían los datos entre la capa EU 720. Por ejemplo, los datos de C pueden enviarse a la jerarquía de caché de B tan pronto como se complete, configurando una canalización.

Según una realización, la partición y asignación de EU 720 se puede establecer de antemano basándose en el conocimiento del dominio. En tal realización, los mecanismos de cómputo EU 720 pueden dividirse estáticamente, de modo que la asignación de EU permanezca igual durante la vida útil de una aplicación específica. En otras realizaciones, los EU 720 pueden dividirse de manera óptima para cada invocación de ejecución de la GPU 614. Aún en otras realizaciones, la configuración puede ser dinámica de modo que se cambie por grupo de hilos durante el envío. En aún otras realizaciones, la partición se puede implementar para realizar el procesamiento de otros tipos de capas de red neuronal, además de las capas C, B, R y P, determinando un patrón común y configurando una canalización para ejecutarlas de forma ráster en el GPU en lugar de realizarlos individualmente.

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 voz y/o reconocimiento del 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. Normalmente, una topología de red predictiva 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 representado por el algoritmo que se ejecuta, la salida del algoritmo de red neuronal puede tomar varias 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, una señal de error que representa se calcula 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 capacitación puede ser intensivo desde el punto de vista computacional y puede requerir una cantidad significativa de tiempo en un procesador convencional de uso 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, ya que los cómputos realizados para ajustar los coeficientes en las redes neuronales se prestan naturalmente a implementaciones paralelas. Específicamente, muchos algoritmos de aprendizaje automático y aplicaciones de software se han adaptado a hacer uso del hardware de procesamiento paralelo dentro de dispositivos de procesamiento de gráficos de fin general.

La **Figura 8** es un diagrama generalizado de una pila de software de aprendizaje automático 800. Se puede configurar una aplicación de aprendizaje automático 802 para entrenar una red neuronal usando un conjunto de datos de entrenamiento o para usar una red neuronal profunda entrenada para implementar inteligencia de máquina. La aplicación de aprendizaje automático 802 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 802 puede implementar cualquier tipo de inteligencia automática incluyendo, pero sin limitación, reconocimiento de imágenes, correlación y localización, navegación autónoma, síntesis de habla, formación de imágenes médicas o traducción de idioma.

Puede habilitarse una aceleración de hardware para la aplicación de aprendizaje automático 802 mediante una estructura de aprendizaje automático 804. La estructura de aprendizaje automático 804 puede proporcionar una biblioteca de primitivas de aprendizaje automático. Las primitivas de aprendizaje automático son operaciones básicas que comúnmente realizan algoritmos de aprendizaje automático. Sin la estructura de aprendizaje automático 804, 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 su lugar, la aplicación de aprendizaje automático puede estar configurada para realizar los cómputos necesarios usando las primitivas proporcionadas por la estructura de aprendizaje automático 804. Las primitivas ilustrativas incluyen convoluciones tensoriales, 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 804 también puede proporcionar primitivas para implementar subprogramas de álgebra lineal básicos realizados por muchos algoritmos de aprendizaje automático, tales como operaciones matriciales y vectoriales.

La estructura de aprendizaje automático 804 puede procesar datos de entrada recibidos desde la aplicación de aprendizaje automático 802 y generar la entrada apropiada en una estructura de cómputo 806. La estructura de cómputo 806 puede abstraer las instrucciones subyacentes proporcionadas al controlador de GPGPU 808 para permitir que la estructura de aprendizaje automático 804 se aproveche de la aceleración de hardware mediante el hardware de GPGPU 810 sin requerir que la estructura de aprendizaje automático 804 tenga un conocimiento íntimo de la arquitectura del hardware de GPGPU 810. Adicionalmente, la estructura de cómputo 806 puede habilitar la aceleración de hardware para la estructura de aprendizaje automático 804 a lo largo de una diversidad de tipos y generaciones del hardware de GPGPU 810.

Aceleración de aprendizaje automático de GPGPU

La **Figura 9** ilustra una unidad de procesamiento de gráficos de uso general altamente paralela 900, de acuerdo con una realización. En una realización, la unidad de procesamiento de propósito general (GPGPU) 900 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 900 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 900 incluye una interfaz de anfitrión 902 para habilitar una conexión con un procesador de anfitrión. En una realización, la interfaz de anfitrión 902 es una interfaz PCI Express. Sin embargo, la interfaz de anfitrión puede ser también una interfaz de comunicaciones o estructura de comunicaciones específico de proveedor. La GPGPU 900 recibe comandos del procesador anfitrión y utiliza un programador global 904 para distribuir hilos de ejecución asociados con esos comandos a un conjunto de grupos de cómputo 906AH. Los grupos de cómputo 906A-H comparten una memoria caché 908. La memoria caché 908 puede servir como una memoria caché de nivel superior para memorias caché dentro de los grupos de cómputo 906A-H.

La GPGPU 900 incluye memoria 914A-B acoplada con los grupos de cómputo 906AH a través de un conjunto de controladores de memoria 912A-B. En diversas realizaciones, la memoria 914A-B puede incluir varios tipos de dispositivos de memoria que incluyen memoria de acceso aleatorio dinámico (DRAM) o memoria de acceso aleatorio de gráficos, tal como memoria de acceso aleatorio de gráficos síncronos (SGRAM), incluida memoria de velocidad 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 ancho de banda alto (HBM).

En una realización, cada grupo de cómputo GPLAB06A-H incluye un conjunto de multiprocesadores de gráficos, como el multiprocesador de gráficos 400 de la Figura 4A. Los multiprocesadores de gráficos del grupo de cómputo agrupan múltiples tipos de unidades de lógica de enteros y de coma flotante que pueden realizar operaciones computacionales con un rango de precisiones que incluyen unas adecuadas para cómputos de aprendizaje automático. Por ejemplo, y en una realización, al menos un subconjunto de las unidades de punto flotante en cada uno de los grupos de cómputo 906A-H se puede configurar para realizar operaciones de punto flotante de 16 o 32 bits, mientras que un subconjunto diferente de las unidades de punto flotante se puede configurar para realizar operaciones de punto flotante de 64 bits.

Se pueden configurar varias instancias de GPGPU 900 para que funcionen como un grupo de cómputo. El mecanismo de comunicación usado por el grupo de cómputo para la sincronización y el intercambio de datos varía a lo largo de las realizaciones. En una realización, las múltiples instancias de la GPGPU 900 se comunican a través de la interfaz de anfitrión 902. En una realización, la GPGPU 900 incluye un concentrador de E/S 908 que acopla la GPGPU 900 con un enlace de GPU 910 que permite una conexión directa a otras instancias de la GPGPU. En una realización, el enlace de GPU 910 está acoplado a un puente de GPU a GPU dedicado que permite la comunicación y sincronización entre múltiples instancias de la GPGPU 900. En una realización, el enlace GPU 910 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 900 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 902. En una realización, el enlace GPU 910 se puede configurar para permitir una conexión a un procesador principal además de o como alternativa a la interfaz principal 902.

Si bien la configuración ilustrada de la GPGPU 900 se puede configurar para entrenar redes neuronales, una realización proporciona una configuración alternativa de la GPGPU 900 que se puede configurar para su implementación dentro de una plataforma de inferencia de alto rendimiento o baja potencia. En una configuración de inferenciación, la GPGPU 900 incluye menos de los grupos de cómputo de los grupos de cómputo 906A-H en relación con la configuración de entrenamiento. Además, la tecnología de memoria asociada con la memoria 914A-B puede diferir entre las configuraciones de inferencia y entrenamiento. En una realización, la configuración de inferencia de la GPGPU 900 puede admitir la inferencia de instrucciones específicas. Por ejemplo, una configuración de inferencia puede proporcionar soporte para una o más instrucciones de producto escalar de números enteros de 8 bits, que se usan comúnmente durante operaciones de inferencia para redes neuronales implementadas.

La **Figura 10** ilustra un sistema informático multi-GPU 1000, de acuerdo con una realización. El sistema informático de múltiples GPU 1000 puede incluir un procesador 1002 acoplado a múltiples GPGPU 1006A-D mediante un conmutador de interfaz de anfitrión 1004. El conmutador de interfaz de anfitrión 1004, en una realización, es un dispositivo de conmutador PCI express que acopla el procesador 1002 a un bus PCI express a través del cual el procesador 1002 puede comunicarse con el conjunto de GPGPU 1006A-D. Cada una de las múltiples GPGPU 1006A-D puede ser una instancia de la GPGPU 900 de la Figura 9. Las GPGPU 1006A-D pueden interconectarse mediante un conjunto de enlaces de GPU a GPU de punto a punto de alta velocidad 1016. Los enlaces de GPU a GPU de alta velocidad se pueden conectar a cada una de las GPGPU 1006A-D a través de un enlace de GPU dedicado, tal como el enlace de GPU 910 como en la Figura 9. Los enlaces de GPU P2P 1016 permiten la comunicación directa entre cada una de las GPGPU 1006A-D sin requerir comunicación a través del bus de interfaz de anfitrión al que está conectado el procesador 1002. Con el tráfico de GPU a GPU dirigido a los enlaces de GPU de P2P, el bus de interfaz de anfitrión permanece disponible para el acceso de memoria de sistema o para comunicarse con otras instancias del sistema informático de múltiples GPU 1000, por ejemplo, mediante uno o más dispositivos de red. Aunque, en la realización ilustrada, las GPGPU 1006A-D se conectan al procesador 1002 mediante el conmutador de interfaz de anfitrión 1004, en una realización, el procesador 1002 incluye un soporte directo para los enlaces de GPU de P2P 1016 y puede conectarse directamente a las GPGPU 1006A-D.

Implementaciones de red neuronal de aprendizaje automático

La arquitectura informática proporcionada por las realizaciones descritas en el presente documento puede configurarse para realizar los tipos de procesamiento paralelo que son particularmente adecuados para entrenar y desplegar redes neuronales para un aprendizaje automático. Una red neuronal puede generalizarse como una red de funciones que tienen una relación de grafo. Como es bien conocido en la técnica, existe una variedad de tipos de implementaciones de redes neuronales utilizadas en el aprendizaje automático. Un tipo ilustrativo de red neuronal es la red de avance, como se describió anteriormente.

Un segundo tipo ilustrativo de red neuronal es la red neuronal convolucional (CNN). Una CNN es una red neuronal de retroalimentación especializada para procesar datos que tienen una topología conocida en forma de cuadrícula, como datos de imágenes. En consecuencia, las CNN se usan comúnmente para aplicaciones de reconocimiento de imágenes y de visión de cómputo, pero también pueden usarse para otros tipos de reconocimiento de patrones, tales como procesamiento de habla y de idioma. Los nodos en la capa de entrada de CNN están organizados en un conjunto de "filtros" (detectores de funciones inspirados en los campos receptivos que se encuentran en la retina), y la salida de cada conjunto de filtros se propaga a los nodos en capas sucesivas de la red. Los cómputos 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 una clase especializada de operación matemática realizada por dos funciones para producir una tercera función que es una versión modificada de una de las dos funciones originales. En terminología de 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 el mapa de características. Por ejemplo, la entrada a una capa de convolución puede ser una matriz multidimensional de datos que definen 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 neuronales de retroalimentación que incluyen conexiones de retroalimentación 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 el futuro, ya que al menos una porción de los datos de salida del RNN se utiliza como retroalimentación para procesar entradas posteriores en una secuencia. Esta característica hace a las RNN particularmente útiles para procesamiento de idioma debido a la naturaleza variable en la que pueden estar compuestos los datos de idioma.

Las figuras que se describen a continuación presentan redes de predictiva, CNN y RNN ilustrativas, además de describir 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 en general a redes neuronales profundas y técnicas de aprendizaje automático en general.

Las redes neuronales ilustrativas anteriormente descritas pueden usarse para realizar aprendizaje profundo. El aprendizaje profundo es el 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 etapas múltiples que da como resultado un error de salida reducido en relación con las técnicas superficiales de aprendizaje automático.

Las redes neuronales profundas usadas en aprendizaje automático incluyen típicamente una red de extremo frontal para realizar un reconocimiento de característica acoplada a una red de extremo trasero que representa un modelo matemático que puede realizar operaciones (por ejemplo, clasificación de objetos, reconocimiento de habla, etc.) basándose en la representación de característica proporcionada en el modelo. El aprendizaje profundo posibilita que se realice el aprendizaje automático sin requerir que se realice ingeniería de características artesanal para el modelo. En su lugar, las redes neuronales profundas pueden presentar características basándose en una estructura estadística o correlación dentro de los datos de entrada. Las características aprendidas se pueden proporcionar a un modelo matemático que puede asignar 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 está estructurada 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. Luego, los valores de error se propagan hacia atrás hasta que cada neurona tiene un valor de error asociado que representa aproximadamente su contribución a la salida original. Luego, la red puede aprender de esos errores utilizando un algoritmo, como el algoritmo de descenso de gradiente estocástico, para actualizar los pesos de la red neuronal.

Las **Figuras 11A y 11B** ilustran una red neuronal convolucional ilustrativa. La **Figura 11A** ilustra varias capas dentro de una CNN. Como se muestra en la **Figura 11A**, una CNN ilustrativa utilizada para modelar el procesamiento de imágenes puede recibir la entrada 1102 que describe los componentes rojo, verde y azul (RGB) de una imagen de entrada. La entrada 1102 puede procesarse mediante múltiples capas convolucionales (por ejemplo, capa convolucional 1104, capa convolucional 1106). La salida desde las múltiples capas convolucionales opcionalmente puede ser procesada por un conjunto de capas completamente conectadas 1108. Las neuronas en una capa completamente conectada tienen conexiones completas a todas las activaciones en la capa previa, como se ha descrito previamente para una red de realimentación prospectiva. La salida desde las capas completamente conectadas 1108 puede usarse para generar un resultado de salida a partir de la red. Las activaciones dentro de las capas completamente conectadas 908 pueden computarse usando una multiplicación matricial en lugar de una convolución. No todas las implementaciones de CNN hacen uso de las capas completamente conectadas 1108. Por ejemplo, en algunas implementaciones, la capa convolucional 1106 puede generar la salida de la CNN.

Las capas convolucionales están conectadas de manera dispersa, que difiere de la configuración de red neuronal tradicional encontrada en las capas completamente conectadas 1108. Las capas de red neuronal tradicionales están completamente conectadas, de manera que cada unidad de salida interacciona con cada unidad de entrada. Sin embargo, las capas convolucionales están conectadas de manera dispersa debido a que se introduce la salida de la convolución de un campo (en lugar del valor de estado respectivo de cada uno de los nodos en el campo) en los nodos de la capa subsiguiente, como se ha ilustrado. Los núcleos asociados con las capas convolucionales realizan operaciones de convolución, cuya salida se envía a la capa siguiente. La reducción de dimensionalidad realizada dentro de las capas convolucionales es un aspecto que habilita a la CNN para que realice un ajuste a escala para procesar imágenes grandes.

La **Figura 11B** ilustra etapas de cómputo ilustrativas dentro de una capa convolucional de una CNN. La entrada a una capa convolucional 1112 de una CNN puede procesarse en tres fases de una capa convolucional 1114. Las tres etapas pueden incluir una etapa de convolución 1116, una etapa de detector 1118 y una etapa de grupo 1120. La capa de convolución 1114 puede emitir entonces datos a una capa convolucional sucesiva. La capa convolucional final de la red puede generar datos de correlación de características de salida o proporcionar una entrada a una capa completamente conectada, por ejemplo, para generar un valor de clasificación para la entrada a la CNN.

En la fase de convolución 1116 se realizan varias convoluciones en paralelo para producir un conjunto de activaciones lineales. La fase de convolución 1116 puede incluir una transformación afín, que es cualquier transformación que pueda especificarse como una transformación lineal más una traslación. Las transformaciones afines incluyen rotaciones, traslaciones, escalamientos 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 puede determinar como la región local asociada con la neurona. Las neuronas calculan un producto vectorial 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 1116 define un conjunto de activaciones lineales que son procesadas por etapas sucesivas de la capa convolucional 1114.

Las activaciones lineales pueden ser procesadas por una fase de detección 1118. En la fase de detección 1118, 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 $f(x) = \max(0, x)$, de modo que la activación tiene un umbral de cero.

La etapa de grupo 1 120 utiliza una función de grupo que reemplaza la salida de la capa convolucional 1106 con una estadística resumida de las salidas cercanas. La función de grupo se puede utilizar para introducir invariancia de traducción en la red neuronal, de modo que pequeñas traducciones a la entrada no cambien las salidas agrupadas. La invarianza a la traducción local puede ser útil en escenarios 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 fase de agrupamiento 1120, incluyendo agrupamiento máximo, agrupamiento promedio y agrupamiento de normas 12. Adicionalmente, algunas implementaciones de CNN no incluyen una fase de agrupamiento. En su lugar, tales implementaciones sustituyen una fase de convolución adicional que tiene un paso mayor en relación con las fases de convolución anteriores.

La salida de la capa convolucional 1114 puede procesarse a continuación por la siguiente capa 1122. La siguiente capa 1122 puede ser una capa convolucional adicional o una de las capas completamente conectadas 1108. Por ejemplo, la primera capa convolucional 1104 de la **Figura 11A** puede enviarse a la segunda capa convolucional 1106, mientras que la segunda capa convolucional puede enviarse a una primera capa de las capas completamente conectadas 1108.

La **Figura 12** ilustra una red neuronal recurrente ilustrativa 1200. 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 RNN generalmente gira en torno al uso de modelos matemáticos para predecir el futuro en función de 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. Se puede describir que la RNN 1200 ilustrado tiene una capa de entrada 1202 que recibe un vector de entrada, capas ocultas 1204 para implementar una función recurrente, un mecanismo de retroalimentación 1205 para permitir una 'memoria' de estados anteriores y una capa de salida 1206 para generar un resultado. La RNN 1200 funciona en función de pasos de tiempo. El estado de la RNN en un escalón de tiempo dado se ve influenciado basándose en el escalón de tiempo previo mediante el mecanismo de realimentación 1205. Para un escalón de tiempo dado, el estado de las capas ocultas 1204 está definido por el estado anterior y la entrada en el escalón de tiempo actual. Una entrada inicial (x_1) en un primer escalón de tiempo puede ser procesada por la capa oculta 1204. Una segunda entrada (x_2) puede ser procesada por la capa oculta 1204 usando información de estado que se determina durante el procesamiento de la entrada inicial (x_1). Un estado dado puede computarse 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, como la función tangente hiperbólica (Tanh) o una variante de la función rectificadora $f(x) = \max(0,x)$. Sin embargo, la función matemática específica utilizada en las capas ocultas 1004 puede variar dependiendo de los detalles de implementación específicos de la RNN 1200.

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 larga a corto plazo (LSTM). Las RNN de LSTM son capaces de aprender dependencias a largo plazo que pueden ser necesarias para procesar secuencias de idioma más largas. Una variante de 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 creencia profunda (DBN) es una red neuronal generativa que está compuesta por múltiples capas de variables estocásticas (aleatorias). Las DBN pueden entrenarse capa a capa usando aprendizaje no supervisado voraz. Los pesos aprendidos del DBN se pueden utilizar para proporcionar redes neuronales previas al entrenamiento determinando un conjunto inicial óptimo de pesos para la red neuronal.

La **Figura 13** 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 1302. Se han desarrollado diversas estructuras de entrenamiento 1304 para posibilitar la aceleración de hardware del proceso de entrenamiento. Por ejemplo, el marco de aprendizaje automático 804 de la Figura 8 puede configurarse como un marco de entrenamiento 1304. El marco de entrenamiento 1304 puede conectarse a una red neuronal no entrenada 1306 y permitir que la red neuronal no entrenada se entrene utilizando los recursos de procesamiento paralelo descritos en el presente documento para generar una red neuronal entrenada 1308.

Para iniciar el proceso de entrenamiento, los pesos iniciales pueden elegirse aleatoriamente o mediante preentrenamiento usando una red de creencia profunda. El ciclo de entrenamiento puede realizarse entonces de una manera o bien supervisada o bien no supervisada.

El aprendizaje supervisado es un método de aprendizaje en el que un entrenamiento se realiza como una operación mediada, tal como cuando el conjunto de datos de entrenamiento 1302 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. Luego, los errores se propagan nuevamente a través del sistema. El marco de entrenamiento 1304 puede ajustarse para ajustar los pesos que controlan la red neuronal no entrenada 1306. El marco de entrenamiento 1304 puede proporcionar herramientas para monitorear qué tan bien está convergendo la red neuronal no entrenada 1306 hacia un modelo adecuado para generar respuestas correctas basadas en 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 1308. Luego, la red neuronal entrenada 1308 se puede implementar para implementar cualquier número de operaciones de aprendizaje automático.

El aprendizaje no supervisado es un método automático en el que la red intenta entrenarse a sí misma usando datos no etiquetados. Por tanto, para la formación de equipos no supervisados, el conjunto de datos de entrenamiento 1302 incluirá datos de entrada sin ningún dato de salida asociado. La red neuronal no entrenada 1306 puede aprender grupos dentro de la entrada sin etiquetar y puede determinar cómo se relacionan las entradas individuales con el conjunto de datos general. El entrenamiento no supervisado puede usarse para generar una correlación de autoorganización, que es un tipo de red neuronal entrenada 1307 que puede realizar operaciones útiles en cuanto a la reducción de la dimensionalidad de los datos. El entrenamiento no supervisado también puede usarse para realizar una 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 1302 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 1308 se adapte a los nuevos datos 1312 sin olvidar el conocimiento inculcado dentro de la red durante el entrenamiento inicial.

Ya sea supervisado o no supervisado, el proceso de entrenamiento para redes neuronales particularmente profundas puede ser demasiado intensivo desde el punto de vista computacional para un único nodo de cómputo. En lugar de usar un único nodo de cómputo, puede usarse una red distribuida de nodos computacionales para acelerar el proceso de entrenamiento.

La **Figura 14** es un diagrama de bloques que ilustra el aprendizaje distribuido. El aprendizaje distribuido es un modelo de entrenamiento que utiliza 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 principales 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 900 como en la Figura 9. Como se ilustra, el aprendizaje distribuido se puede realizar con el paralelismo de modelo 1402, el paralelismo de datos 1404 o una combinación de modelo y paralelismo de datos 1204.

En el modelo de paralelismo 1402, diferentes nodos computacionales en un sistema distribuido pueden realizar cómputos de entrenamiento para diferentes partes de una única red. Por ejemplo, cada capa de una red neuronal puede ser entrenada por un nodo de procesamiento diferente del sistema distribuido. Los beneficios del paralelismo de modelo incluyen la capacidad de ajustar a escala a modelos particularmente grandes. La división de los cómputos asociados con diferentes capas de la red neuronal habilita el entrenamiento de redes neuronales muy grandes en las que los pesos de todas las capas no cabrían en la memoria de un único nodo computacional. En algunas instancias, el paralelismo de modelo puede ser particularmente útil en la ejecución de un entrenamiento no supervisado de redes neuronales grandes.

En el paralelismo de datos 1404, 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. Si bien son posibles diferentes enfoques para el paralelismo de datos, todos los enfoques de entrenamiento en paralelo de datos requieren una técnica para combinar resultados y sincronizar los parámetros del modelo entre cada nodo. Los enfoques ilustrativos para combinar datos incluyen el promedio de parámetros y el 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, desviaciones) al promedio de los parámetros de cada nodo. El promedio de parámetros utiliza un servidor de parámetros central que mantiene los datos de los parámetros. El paralelismo de datos basado en actualizaciones es similar al promedio de parámetros excepto que en lugar de transferir parámetros desde los nodos al servidor de parámetros, se transfieren las actualizaciones al modelo. Adicionalmente, el paralelismo de datos basado en la actualización puede realizarse de una manera descentralizada, donde se comprimen las actualizaciones y se transfieren entre nodos.

El paralelismo combinado de modelo y datos 1406 se puede implementar, por ejemplo, en un sistema distribuido en el que cada nodo computacional incluye múltiples OPU. Cada nodo puede tener una instancia completa del modelo con CPU separadas dentro de cada nodo que se utilizan para entrenar diferentes porciones del modelo.

La capacitación distribuida ha aumentado los gastos generales en relación con la capacitación en una sola máquina. Sin embargo, los procesadores paralelos y las GPGPU descritos 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 gran ancho de banda y la sincronización remota acelerada de datos.

Aplicaciones de aprendizaje automático ilustrativas

El aprendizaje automático se puede aplicar para resolver una variedad de problemas tecnológicos, incluidos, entre otros, la visión por ordenador, 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 por ordenador van desde la reproducción de habilidades visuales humanas, como reconocer rostros, hasta la creación de nuevas categorías de habilidades 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 entrenar aplicaciones de visión por ordenador utilizando un conjunto de datos de entrenamiento significativamente más grande de lo que antes era factible y permite implementar sistemas de inferencia utilizando procesadores paralelos de baja potencia.

El aprendizaje automático acelerado por procesador paralelo tiene aplicaciones de conducción autónoma que incluyen el reconocimiento de señales de carril y carretera, evitación de obstáculos, navegación y control de conducción. Las técnicas de aprendizaje automático aceleradas pueden usarse para entrenar modelos de conducción basándose en conjuntos de datos que definen las respuestas apropiadas a entrada de entrenamiento específica. Los procesadores paralelos descritos en este documento pueden permitir un entrenamiento rápido de las redes neuronales cada vez

más complejas utilizadas para soluciones de conducción autónoma y permiten la implementación de procesadores de inferencia de baja potencia en una plataforma móvil adecuada para la integración en vehículos autónomos.

Las redes neuronales profundas aceleradas de procesador paralelo han posibilitado enfoques de aprendizaje automático para reconocimiento de voz automático (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 que utiliza redes neuronales profundas ha permitido reemplazar 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 del lenguaje natural. Los procedimientos de aprendizaje automático pueden utilizar algoritmos de inferencia estadística para producir modelos que sean resistentes a entradas erróneas o desconocidas. Las aplicaciones de procesador de lenguaje natural ilustrativas incluyen traducción de máquina automática entre idiomas humanos.

Las plataformas de procesamiento paralelo usadas para aprendizaje automático pueden dividirse en plataformas de entrenamiento y plataformas de despliegue. Las plataformas de capacitación generalmente son muy paralelas e incluyen optimizaciones para acelerar la capacitación de un solo nodo con múltiples GPU y la capacitación de múltiples nodos y múltiples GPU. Los procesadores paralelos ilustrativos adecuados para la formación incluyen la unidad de procesamiento de gráficos de uso general altamente paralelo y el sistema informático multi-GPU. 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 15** ilustra un sistema de inferencia ilustrativa en un chip (SOC) 1500 adecuado para realizar inferencias utilizando un modelo entrenado. El SOC 1500 puede integrar componentes de procesamiento que incluyen un procesador de medios 1502, un procesador de visión 1504, una GPGPU 1506 y un procesador de múltiples núcleos 1508. El SOC 1500 puede incluir adicionalmente memoria en el chip 1505 que puede posibilitar un grupo de datos en chip compartido que es accesible por cada uno de los componentes de procesamiento. Los componentes de procesamiento pueden optimizarse para la operación de baja potencia para posibilitar el despliegue a una diversidad de plataformas de aprendizaje automático, que incluyen vehículos autónomos y robots autónomos. Por ejemplo, puede usarse una implementación del SOC 1500 como una porción del sistema de control principal para un vehículo autónomo. Cuando el SOC 1500 está configurado para su uso en vehículos autónomos, el SOC está diseñado y configurado para cumplir con las normas de seguridad funcional relevantes de la jurisdicción de despliegue.

Durante el funcionamiento, el procesador de medios 1502 y el procesador de visión 1504 pueden trabajar conjuntamente para acelerar las operaciones de visión por ordenador. El procesador de medios 1502 puede habilitar la descodificación de latencia baja de múltiples flujos de vídeo de alta resolución (por ejemplo, 4K, 8K). Los flujos de vídeo descodificados se pueden escribir en una memoria intermedia en la memoria en chip 1505. El procesador de visión 1304 puede analizar entonces el vídeo descodificado y realizar operaciones de procesamiento preliminares sobre las tramas del vídeo descodificado como preparación al procesamiento de las tramas usando un modelo de reconocimiento de imágenes entrenado. Por ejemplo, el procesador de visión 1504 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 cómputos de modelo de extremo trasero por la GPGPU 1506.

El procesador de múltiples núcleos 1508 puede incluir una lógica de control de asistencia en la secuenciación y la sincronización de transferencias de datos y operaciones de memoria compartida realizadas por el procesador de medios 1502 y el procesador de visión 1504. El procesador de múltiples núcleos 1308 también puede funcionar como un procesador de aplicaciones para ejecutar aplicaciones de software que pueden hacer uso de la capacidad de cómputo de inferenciación de la GPGPU 1506. Por ejemplo, al menos una porción de la lógica de navegación y de conducción se puede implementar en software que se ejecuta en el procesador de múltiples núcleos 1508. Dicho software puede emitir directamente cargas de trabajo computacionales a la GPGPU 1506 o las cargas de trabajo computacionales pueden enviarse al procesador multinúcleo 1508, que puede descargar al menos una porción de esas operaciones a la GPGPU 1 506.

La GPGPU 1506 puede incluir grupos de cómputo, tales como una configuración de baja potencia de los grupos de cómputo 906A-906H dentro de la unidad de procesamiento de gráficos de propósito general altamente paralela 900. Los grupos de cómputo dentro de la GPGPU 1506 pueden soportar instrucciones que están optimizadas específicamente para realizar cómputos de inferencia en una red neuronal entrenada. Por ejemplo, la GPGPU 1506 puede soportar instrucciones para realizar cómputos de baja precisión tales como operaciones vectoriales de números enteros de 8 bits y 4 bits.

Sistema de procesamiento de gráficos ilustrativo adicional

Los detalles de las realizaciones descritas anteriormente pueden incorporarse dentro de los sistemas y dispositivos de procesamiento de gráficos descritos a continuación. El sistema de procesamiento de gráficos y los dispositivos de las **Figuras 16-29** ilustran sistemas alternativos y hardware de procesamiento de gráficos que pueden implementar cualquiera y todas las técnicas descritas anteriormente.

Descripción general adicional del sistema de procesamiento de gráficos de ejemplo

La **Figura 16** es un diagrama de bloques de un sistema de procesamiento 1600, de acuerdo con una forma de realización. En diversas formas de realización, el sistema 1600 incluye uno o más procesadores 1602 y uno o más procesadores de gráficos 1608, y puede ser un sistema de escritorio de procesador único, un sistema de estación de trabajo multiprocesador o un sistema de servidor que tiene un gran número de procesadores 1602 o núcleos de procesador 1607. En una forma de realización, el sistema 1600 es una plataforma de procesamiento incorporada dentro de un circuito integrado de sistema en un microprocesador (SoC) para su uso en dispositivos móviles, portátiles o integrados.

Una realización del sistema 1600 puede incluir, o incorporarse dentro de, una plataforma de juegos basada en servidor, una consola de juegos, incluyendo una consola de juegos y de medios, una consola de juegos móvil, una consola de juegos de mano o una consola de juegos en línea. En algunas realizaciones, el sistema 1600 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 1600 también puede incluir, acoplarse con o integrarse 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 1600 es un dispositivo de televisión o de descodificador de salón que tiene uno o más procesadores 1602 y una interfaz gráfica generada por uno o más procesadores de gráficos 1608.

En algunas realizaciones, cada uno de los uno o más procesadores 1602 incluye uno o más núcleos de procesador 1607 para procesar instrucciones que, cuando se ejecutan, realizan operaciones para software de usuario y sistema. En algunas realizaciones, cada uno de los uno o más núcleos de procesador 1607 está configurado para procesar un conjunto de instrucciones 1609 específico. En algunas realizaciones, el conjunto de instrucciones 1609 puede facilitar el cómputo de un conjunto de instrucciones complejo (CISC), el cómputo de un conjunto de instrucciones reducido (RISC) o el cómputo mediante una palabra de instrucción muy larga (VLIW). Múltiples núcleos de procesador 1607 pueden procesar, cada uno, un conjunto de instrucciones 1609 diferente, que puede incluir instrucciones para facilitar la emulación de otros conjuntos de instrucciones. El núcleo del procesador 1607 también puede incluir otros dispositivos de procesamiento, tales como un procesador de señal digital (DSP).

En algunas realizaciones, el procesador 1602 incluye la memoria caché 1604. Dependiendo de la arquitectura, el procesador 1602 puede tener una única caché interna o múltiples niveles de caché interna. En algunas realizaciones, la memoria caché se comparte entre varios componentes del procesador 1602. En algunas formas de realización, el procesador 1602 también utiliza una memoria caché externa (por ejemplo, una memoria caché de nivel 3 (L3) o una memoria caché de último nivel (LLC)) (no mostrada), que se puede compartir entre los núcleos 1607 del procesador utilizando técnicas conocidas de coherencia de caché. Se incluye adicionalmente, en el procesador 1602, un archivo de registro 1606 que puede incluir diferentes tipos de registros para almacenar diferentes tipos de datos (por ejemplo, registros de número entero, registros de coma flotante, registros de estado y un registro de puntero de instrucción). Algunos registros pueden ser registros de propósito general, mientras que otros registros pueden ser específicos del diseño del procesador 1602.

En algunas realizaciones, el procesador 1602 está acoplado con un bus de procesador 1610 para transmitir señales de comunicación tales como dirección, datos o señales de control entre el procesador 1602 y otros componentes en el sistema 1600. En una realización, el sistema 1600 usa una arquitectura de sistema de 'concentrador' ilustrativa, incluyendo un concentrador de controlador de memoria 1616 y un concentrador de controlador de entrada-salida (E/S) 1630. Un concentrador de controlador de memoria 1616 facilita la comunicación entre un dispositivo de memoria y otros componentes del sistema 1600, mientras que un concentrador de controlador de E/S (ICH) 1630 proporciona conexiones a dispositivos de E/S mediante un bus de E/S local. En una realización, la lógica del concentrador de controlador de memoria 1616 está integrada dentro del procesador.

El dispositivo de memoria 1620 puede ser un dispositivo de memoria de acceso aleatorio dinámica (DRAM), un dispositivo de memoria de acceso aleatorio estática (SRAM), dispositivo de memoria flash, dispositivo de memoria de cambio de fase o algún otro dispositivo de memoria que tiene un rendimiento adecuado para servir como una memoria de proceso. En una realización, el dispositivo de memoria 1620 puede funcionar como memoria del sistema para el sistema 1600, para almacenar datos 1622 e instrucciones 1621 para su uso cuando uno o más procesadores 1602 ejecutan una aplicación o proceso. El concentrador de controlador de memoria 1616 también se acopla con un procesador de gráficos externo opcional 1612, que puede comunicarse con el uno o más procesadores de gráficos 1608 en los procesadores 1602 para realizar operaciones de gráficos y de medios.

En algunas realizaciones, el ICH 1630 permite que los periféricos se conecten al dispositivo de memoria 1620 y al procesador 1602 a través de un bus de E/S de alta velocidad. Los periféricos de E/S incluyen, entre otros, un controlador de audio 1646, una interfaz de firmware 1628, un transceptor inalámbrico 1626 (por ejemplo, Wi-Fi, Bluetooth), un dispositivo de almacenamiento de datos 1624 (por ejemplo, unidad de disco duro, unidad flash memoria, etc.), y un controlador de E/S heredado 1640 para acoplar dispositivos heredados (por ejemplo, Sistema Personal 2 (PS/2)) al sistema. Uno o más controladores de bus serie universal (USB) 1642 conectan dispositivos de entrada, tales

como las combinaciones de teclado y ratón 1644. Un controlador de red 1634 también puede acoplarse con el ICH 1630. En algunas realizaciones, un controlador de red de alto rendimiento (no mostrado) se acopla con el bus de procesador 1610. Se apreciará que el sistema 1600 mostrado es ilustrativo y no limitante, debido a que también pueden usarse otros tipos de sistemas de procesamiento de datos que están configurados de manera diferente. Por ejemplo, el concentrador de controlador de E/S 1630 puede integrarse dentro de los uno o más procesadores 1602, o el concentrador de controlador de memoria 1616 y el concentrador de controlador de E/S 1630 pueden integrarse en un procesador de gráficos externo discreto, tal como el procesador de gráficos externo 1612.

La **Figura 17** es un diagrama de bloques de una realización de un procesador 1 700 que tiene uno o más núcleos de procesador 1702A-1702N, un controlador de memoria integrado 1714 y un procesador de gráficos integrado 1708. Esos elementos de la **Figura 17** 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 1700 puede incluir núcleos adicionales hasta e incluyendo el núcleo adicional 1702N representado por los recuadros en líneas discontinuas. Cada uno de los núcleos de procesador 1702A-1702N incluye una o más unidades de caché internas 1704A-1704N. En algunas realizaciones, cada núcleo de procesador también tiene acceso a una o más unidades en caché compartidas 1706.

Las unidades de caché internas 1704A-1704N y las unidades de caché compartidas 1706 representan una jerarquía de memoria caché dentro del procesador 1700. La jerarquía de memoria caché puede incluir al menos un nivel de caché de instrucciones y datos dentro de cada núcleo de procesador y uno o más niveles de caché de nivel medio compartido, tal como Nivel 2 (L2), Nivel 3 (L3), Nivel 4 (L4), u otros niveles de caché, donde el nivel más alto de caché antes de la memoria externa se clasifica como LLC. En algunas formas de realización, la lógica de coherencia de caché mantiene la coherencia entre las diversas unidades de caché 1706 y 1704A-1704N.

En algunas formas de realización, el procesador 1700 también puede incluir un conjunto de una o más unidades de controlador de bus 1716 y un núcleo de agente de sistema 1710. Las una o más unidades controladoras de bus 1716 gestionan un conjunto de buses de periféricos, tales como uno o más buses de interconexión de componentes periféricos (por ejemplo, PCI, PCI Express). El núcleo de agente de sistema 1710 proporciona funcionalidad de gestión para los diversos componentes de procesador. En algunas realizaciones, el núcleo de agente del sistema 1710 incluye uno o más controladores de memoria integrados 1714 para gestionar el acceso a varios dispositivos de memoria externos (no mostrados).

En algunas realizaciones, uno o más de los núcleos de procesador 1702A-1702N incluyen soporte para múltiples hilos simultáneos. En una realización de este tipo, el núcleo de agente de sistema 1710 incluye componentes para coordinar y operar los núcleos 1702A-1702N durante el procesamiento de múltiples hilos. El núcleo de agente de sistema 1710 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 1702A-1702N y el procesador de gráficos 1708.

En algunas realizaciones, el procesador 1700 incluye adicionalmente un procesador de gráficos 1708 para ejecutar operaciones de procesamiento de gráficos. En algunas realizaciones, el procesador de gráficos 1708 se acopla con el conjunto de unidades de caché compartidas 1706 y el núcleo de agente del sistema 1710, incluido uno o más controladores de memoria integrados 1714. En algunas realizaciones, un controlador de pantalla 1711 está acoplado con el procesador de gráficos 1708 para controlar la salida del procesador de gráficos a una o más pantallas acopladas. En algunas realizaciones, el controlador de pantalla 1711 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 1708 o el núcleo de agente del sistema 1710.

En algunas realizaciones, se usa una unidad de interconexión basada en anillo 1712 para acoplar los componentes internos del procesador 1700. 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 1708 se acopla con la interconexión en anillo 1712 mediante un enlace de E/S 1713.

El enlace de E/S ilustrativo 1713 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 1718, tal como un módulo de eDRAM. En algunas realizaciones, cada uno de los núcleos de procesador 1702A-1702N y el procesador de gráficos 1708 usan módulos de memoria integrados 1718 como una caché de último nivel compartida.

En algunas realizaciones, los núcleos de procesador 1702A-1702N son núcleos homogéneos que ejecutan la misma arquitectura de conjunto de instrucciones. En otra realización, los núcleos de procesador 1702A-1702N son heterogéneos en términos de arquitectura de conjunto de instrucciones (ISA), donde uno o más de los núcleos de procesador 1702A-1702N 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 1702A-1702N son heterogéneos en términos de microarquitectura, donde uno o más núcleos que tienen un consumo de energía relativamente superior se acoplan con uno o más núcleos de

potencia que tienen un consumo de energía inferior. Adicionalmente, el procesador 1700 se puede implementar en uno o más chips o como un circuito integrado de SoC que tiene los componentes ilustrados, además de otros componentes.

La **Figura 18** es un diagrama de bloques de un procesador de gráficos 1800, que puede ser una unidad de procesamiento gráfico discreta, o puede ser un procesador de gráficos integrado con varios núcleos de procesamiento. En algunas formas de realización, el procesador de gráficos se comunica por medio de una interfaz de E/S asignada en memoria con los registros del procesador de gráficos y con las órdenes colocadas en la memoria del procesador. En algunas realizaciones, el procesador de gráficos 1 800 incluye una interfaz de memoria 1814 para acceder a la memoria. La interfaz de memoria 1814 puede ser una interfaz para la memoria local, una o más cachés internas, una o más cachés externas compartidas y/o la memoria del sistema.

En algunas realizaciones, el procesador de gráficos 1800 también incluye un controlador de visualización 1802 para enviar datos de salida de visualización a un dispositivo de visualización 1820. El controlador de visualización 1802 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 1800 incluye un motor de códec de vídeo 1 806 para codificar, decodificar o transcodificar medios hacia, desde o entre uno o más formatos de codificación de medios, incluidos, entre otros, formatos del Grupo de Expertos en Imágenes en Movimiento (MPEG), tales como MPEG-2, formatos de codificación de vídeo avanzada (AVC) como H.264/MPEG-4 AVC, así como la Sociedad de Ingenieros de Cine y Televisión (SMPTE) 421 M/VC-1 y el Grupo Conjunto de Expertos Fotográficos (JPEG) como los formatos JPEG y Motion JPEG (MJPEG).

En algunas realizaciones, el procesador de gráficos 1800 incluye un motor de transferencia de imágenes en bloques (BLIT) 1804 para realizar operaciones de rasterización bidimensionales (2D) que incluyen, por ejemplo, transferencias de bloques con límites de bits. Sin embargo, en una realización, se realizan operaciones de gráficos 2D usando uno o más componentes del motor de procesamiento de gráficos (GPE) 1810. En algunas realizaciones, el GPE 1810 es un motor de cómputo para realizar operaciones de gráficos, incluyendo operaciones de gráficos tridimensionales (3D) y operaciones de medios.

En algunas realizaciones, el GPE 1810 incluye una canalización de 3D 1812 para realizar operaciones 3D, tales como representar imágenes y escenas tridimensionales usando funciones de procesamiento que actúan sobre formas de primitivas 3D (por ejemplo, rectángulo, triángulo, etc.). La canalización de 3D 1812 incluye elementos de función programable y fija que realizan diversas tareas dentro del elemento y/o generan hilos de ejecución en un subsistema de 3D/de medios 1815. Aunque la canalización de 3D 1812 se puede usar para realizar operaciones de medios, una realización del GPE 1810 también incluye una canalización de medios 1816 que se usa específicamente para realizar operaciones de medios, tales como post-procesamiento de vídeo y potenciación de imagen.

En algunas realizaciones, la canalización de medios 1816 incluye funciones fijas o unidades lógicas programables para realizar una o más operaciones de medios especializadas, tales como aceleración de decodificación de video, desentrelazado de video y aceleración de codificación de video en lugar de, o en nombre del motor de códec de video 1806. En algunas realizaciones, la canalización de medios 1816 incluye adicionalmente una unidad de generación de hilos para generar hilos para su ejecución en el subsistema 3D/de medios 1815. Los hilos generados realizan cómputos para las operaciones de medios en una o más unidades de ejecución de gráficos incluidas en el subsistema de 3D/de medios 1815.

En algunas realizaciones, el subsistema de 3D/de medios 1815 incluye una lógica para ejecutar hilos generados por la canalización de 3D 1812 y la canalización de medios 1816. En una realización, las canalizaciones envían solicitudes de ejecución de hilos al subsistema de 3D/de medios 1815, incluyendo una lógica de despacho de hilos para arbitrar y despachar las diversas solicitudes a recursos de ejecución de hilos disponibles. Los recursos de ejecución incluyen una matriz de unidades de ejecución de gráficos para procesar los hilos de 3D y de medios. En algunas realizaciones, el subsistema de 3D/de medios 1815 incluye una o más cachés internas para datos e instrucciones de hilo. En algunas realizaciones, el subsistema también incluye memoria compartida, incluyendo registros y memoria direccionable, para compartir datos entre hilos y para almacenar datos de salida.

Motor de procesamiento de gráficos

La **Figura 19** es un diagrama de bloques de un motor de procesamiento gráfico 1910 de un procesador de gráficos de acuerdo con algunas formas de realización. En una realización, el motor de procesamiento de gráficos (GPE) 1910 es una versión del GPE 1810 mostrado en la **Figura 18.** 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. Por ejemplo, la tubería 3 D 1812 y la tubería de medios 1816 de la **Figura 18** están ilustradas. La canalización de medios 1816 es opcional en algunas realizaciones del GPE 1910 y puede no incluirse explícitamente dentro del GPE 1910. Por ejemplo, y en al menos una realización, un procesador de medios y/o de imágenes separado se acopla al GPE 1910.

En algunas realizaciones, el GPE 1910 se acopla con o incluye un transmisor por flujo continuo de comandos 1903, que proporciona un flujo de comandos a la canalización de 3D 1812 y/o a las canalizaciones de medios 1816. En algunas realizaciones, el transmisor por flujo continuo de comandos 1903 está acoplado con la memoria, que puede ser una memoria de sistema, o una o más de una memoria caché interna y una memoria caché compartida. En algunas realizaciones, el transmisor por flujo continuo de comandos 1903 recibe comandos desde la memoria y envía los comandos a la canalización de 3D 1812 y/o a la canalización de medios 1816. Los comandos son directivas obtenidas de una memoria intermedia de anillo, que almacena comandos para el canal 3D 1812 y el canal de medios 1816. En una realización, la memoria intermedia en anillo puede incluir adicionalmente memorias intermedias de comandos por lotes que almacenan lotes de múltiples comandos. Los comandos para la canalización de 3D 1812 también pueden incluir referencias a datos almacenados en memoria, tales como, pero sin limitación, datos de vértice y de geometría para la canalización de 3D 1812 y/o datos de imagen y objetos de memoria para la canalización de medios 1816. La canalización de 3D 1812 y la canalización de medios 1816 procesan los comandos y datos realizando operaciones mediante una lógica dentro de las canalizaciones respectivas o despachando uno o más hilos de ejecución a una matriz de núcleo de gráficos 1914.

En diversas realizaciones, la canalización de 3D 1812 puede ejecutar uno o más programas de sombreado, tales como sombreadores de vértices, sombreadores de geometría, sombreadores de píxeles, sombreadores de fragmentos, sombreadores de cómputo u otros programas de sombreado, procesando las instrucciones y despachando hilos de ejecución a la matriz de núcleo de gráficos 1914. La matriz de núcleo de gráficos 1914 proporciona un bloque unificado de recursos de ejecución. La lógica de ejecución multipropósito (por ejemplo, unidades de ejecución) dentro de la matriz de núcleo gráfico 1914 incluye soporte para varios lenguajes de sombreador API 3D y puede ejecutar múltiples hilos de ejecución simultáneos asociados con múltiples sombreadores.

En algunas realizaciones, la matriz de núcleo de gráficos 1914 también incluye una lógica de ejecución para realizar funciones de medios, tales como procesamiento de vídeo y/o de imagen. En una realización, las unidades de ejecución incluyen adicionalmente una lógica de fin general que es programable para realizar operaciones computacionales de fin 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(s) del procesador 1607 de la **Figura 16** o núcleo 1702A-1702N como en la **Figura 17.**

Los datos de salida generados por hilos que se ejecutan en la matriz central de gráficos 1914 pueden enviar datos a la memoria en una memoria intermedia de retorno unificado (URB) 1918. La URB 1918 puede almacenar datos para múltiples hilos. En algunas realizaciones, la URB 1918 puede usarse para enviar datos entre diferentes hilos que se ejecutan en la matriz de núcleo de gráficos 1914. En algunas realizaciones, la URB 1918 se puede usar adicionalmente para la sincronización entre hilos en la matriz central de gráficos y la lógica de función fija dentro de la lógica de función compartida 1920.

En algunas realizaciones, la matriz de núcleos de gráficos 1914 es ajustable a escala, de modo que la matriz incluye un número variable de núcleos de gráficos, teniendo cada uno un número variable de unidades de ejecución basándose en la potencia objetivo y en el nivel de rendimiento del GPE 1910. En una realización, los recursos de ejecución son dinámicamente escalables, de modo que los recursos de ejecución pueden habilitarse o deshabilitarse según sea necesario,

La matriz de núcleos de gráficos 1914 se acopla con la lógica de función compartida 1920 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 funciones compartidas 1920 son unidades de lógica de hardware que proporcionan una funcionalidad complementaria especializada a la matriz de núcleo de gráficos 1914. En diversas realizaciones, la lógica de funciones compartidas 1920 incluye, pero sin limitación, la lógica del muestreador 1921, del cómputo matemático 1922 y de la comunicación entre hilos (ITC) 1923. Adicionalmente, algunas realizaciones implementan una o más cachés 1925 dentro de la lógica de funciones compartidas 1920. Se implementa una función compartida donde la demanda de una función especializada dada es insuficiente para su inclusión dentro de la matriz de núcleo de gráficos 1914. En su lugar, una única instanciación de esa función especializada se implementa como una entidad autónoma en la lógica de funciones compartidas 1920 y se comparte entre los recursos de ejecución dentro de la matriz de núcleo de gráficos 1914. El conjunto preciso de funciones que se comparten entre la matriz de núcleo de gráficos 1914 y se incluyen dentro de la matriz de núcleo de gráficos 1914 varía entre realizaciones.

La **Figura 20** es un diagrama de bloques de otra forma de realización de un procesador de gráficos 2000. Elementos de la **Figura 20** 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 2000 incluye una interconexión en anillo 2002, un extremo frontal de canalización 2004, un motor de medios 2037 y núcleos de gráficos 2080A-2080N. En algunas realizaciones, la interconexión en anillo 2002 acopla el procesador de gráficos a otras unidades de procesamiento, que incluyen otros procesadores de gráficos o uno o más núcleos de procesadores de fin 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 2000 recibe lotes de comandos mediante la interconexión en anillo 2002. Los comandos entrantes son interpretados por un transmisor de comandos 2003 en el extremo frontal de canalización 2004. En algunas realizaciones, el procesador de gráficos 2000 incluye una lógica de ejecución escalable para realizar procesamiento de geometría 3D y procesamiento de medios mediante el núcleo o núcleos de gráficos 2080A-2080N. Para los comandos de procesamiento de geometría 3D, el transmisor por flujo continuo de comandos 2003 suministra comandos a la canalización de geometría 2036. Para al menos algunos comandos de procesamiento de medios, el transmisor por flujo continuo de comandos 2003 suministra los comandos a un extremo delantero de vídeo 2034, que se acopla con un motor de medios 2037. En algunas realizaciones, el motor de medios 2037 incluye un motor de calidad de vídeo (VQE) 2030 para post procesamiento de vídeo y de imagen y un motor de codificación/decodificación de múltiples formatos (MFX) 2033 para proporcionar codificación y decodificación de datos de medios acelerados por hardware. En algunas realizaciones, la canalización de geometría 2036 y el motor de medios 2037 generan, cada uno, hilos de ejecución para los recursos de ejecución de hilos proporcionados por al menos un núcleo de gráficos 2080A.

En algunas realizaciones, el procesador de gráficos 2000 incluye recursos de ejecución de hilos ajustables a escala que cuentan con los núcleos modulares 2080A-2080N (denominados, en ocasiones, cortes de núcleo), teniendo cada uno múltiples subnúcleos 2050A-550N, 2060A-2060N (denominados, en ocasiones, subcortes de núcleo). En algunas realizaciones, el procesador de gráficos 2000 puede tener cualquier número de núcleos de gráficos 2080A a 2080N. En algunas realizaciones, el procesador de gráficos 2000 incluye un núcleo de gráficos 2080A que tiene al menos un primer subnúcleo 2050A y un segundo subnúcleo 2060A. En otras realizaciones, el procesador de gráficos es un procesador de baja potencia con un único subnúcleo (por ejemplo, 2050A). En algunas realizaciones, el procesador de gráficos 2000 incluye múltiples núcleos de gráficos 2080A-2080N, incluyendo cada uno un conjunto de primeros subnúcleos 2050A-2050N y un conjunto de segundos subnúcleos 2060A-2060N. Cada subnúcleo en el conjunto de primeros subnúcleos 2050A-2050N incluye al menos un primer conjunto de unidades de ejecución 2052A-2052N y muestreadores de medios/texturas 2054A-2054N. Cada subnúcleo del conjunto de segundos subnúcleos 2060A-2060N incluye al menos un segundo conjunto de unidades de ejecución 2062A-2062N y muestreadores 2064A-2064N. En algunas realizaciones, cada subnúcleo 2050A-2050N, 2060A-2060N comparte un conjunto de recursos compartidos 2070A-2070N. En algunas formas de realización, los recursos compartidos incluyen memoria caché compartida y lógica de operación de píxeles. Se pueden incluir también otros recursos compartidos en las diversas formas de realización del procesador de gráficos.

Unidades de ejecución

La **Figura 21** ilustra la lógica de ejecución de hilos 2100, que incluye una matriz de elementos de procesamiento empleados en algunas formas de realización de un GPE. 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.

En algunas realizaciones, la lógica de ejecución de hilos 2100 incluye un procesador de sombreado 2102, un despachador de hilos 2104, un caché de instrucciones 2106, una matriz de unidades de ejecución escalable que incluye una pluralidad de unidades de ejecución 21 08A-2108N, un muestreador 2110, un caché de datos 2112 y un puerto de datos 2114. En una realización, la matriz de unidades de ejecución escalables puede escalar dinámicamente habilitando o deshabilitando una o más unidades de ejecución (por ejemplo, cualquiera de las unidades de ejecución 2108A, 21 08B, 21 08C, 2108D, a 2108N-l y 2108N) en función de los requisitos computacionales de una carga de trabajo. 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 formas de realización, la lógica de ejecución de hilos 2100 incluye una o más conexiones a memoria, como por ejemplo una memoria de sistema o memoria caché, a través de una o más de la caché de instrucciones 2106, el puerto de datos 2114, el muestreador 2110 y las unidades de ejecución 2108A-2108N. En algunas realizaciones, cada unidad de ejecución (por ejemplo, 2108A) es una unidad computacional de propósito general programable autónoma que es capaz de ejecutar múltiples hilos de hardware simultáneos mientras se procesan múltiples elementos de datos en paralelo para cada hilo. En diversas realizaciones, la matriz de unidades de ejecución 2108A-2108N es ajustable a escala para incluir cualquier número de unidades de ejecución individuales.

En algunas realizaciones, las unidades de ejecución 2108A-2108N se usan principalmente para ejecutar programas sombreadores. Un procesador de sombreado 2102 puede procesar los diversos programas de sombreado y distribuir hilos de ejecución asociados con los programas de sombreado a través de un despachador de hilos 2104. En una realización, el despachador de hilos incluye una lógica para arbitrar solicitudes de iniciación de un hilo desde las canalizaciones de gráficos y de medios e instanciar los hilos solicitados en una o más unidades de ejecución en las unidades de ejecución 2108A-2108N. Por ejemplo, la tubería de geometría (por ejemplo, 2036 de la **Figura 20)** puede enviar sombreadores de vértices, teselación o geometría a la lógica de ejecución del hilo 2100 **(Figura 21)** para procesar. En algunas formas de realización, el despachador de hilos 2104 también puede procesar solicitudes de generación de hilos en tiempo de ejecución desde los programas de sombreado en ejecución.

En algunas realizaciones, las unidades de ejecución 2108A-2108N soportan un conjunto de instrucciones que incluye un soporte nativo para muchas instrucciones de sombreador de gráficos 3D convencionales, de modo que los programas sombreadores desde bibliotecas de gráficos (por ejemplo, Direct 3D y OpenGL) se ejecutan con una traducción mínima. Las unidades de ejecución soportan un procesamiento de vértices y de geometría (por ejemplo, programas de vértices, programas de geometría, sombreadores de vértices), un procesamiento de píxeles (por ejemplo, sombreadores de píxeles, sombreadores de fragmentos) y un procesamiento de fin general (por ejemplo, sombreadores de cómputo y de medios). Cada una de las unidades de ejecución 2108A-2108N es capaz de múltiples emisiones de ejecución de múltiples datos de instrucción única (SIMD), y un funcionamiento de múltiples hilos posibilita un entorno de ejecución eficiente frente a accesos de memoria de latencia superior. Cada hilo de hardware dentro de cada unidad de ejecución tiene un archivo de registro de ancho de banda alto dedicado y un estado de hilo independiente asociado. La ejecución es de múltiples emisiones por reloj a canalizaciones aptas para operaciones de números enteros, y de coma flotante de precisión sencilla y doble, capacidad de ramal de SIMD, operaciones lógicas, operaciones transcendentales y otras operaciones misceláneas. Mientras se esperan los datos de la memoria o una de las funciones compartidas, la lógica de dependencia dentro de las unidades de ejecución 2108A-2108N hace que un hilo en espera pase a inactividad hasta que se devuelvan los datos solicitados. Mientras el hilo en espera está inactivo, pueden dedicarse recursos de hardware a procesar otros hilos. Por ejemplo, durante un retardo asociado con una operación de sombreador de vértices, una unidad de ejecución puede realizar operaciones para un sombreador de píxeles, un sombreador de fragmentos u otro tipo de programa de sombreado, incluyendo un sombreador de vértices diferente.

Cada unidad de ejecución en las unidades de ejecución 2108A-2108N opera sobre 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, enmascaramiento y control de flujo de elementos de datos dentro de las instrucciones. La cantidad de canales puede ser independiente de la cantidad de unidades físicas aritméticas lógicas (ALU) o unidades de coma flotante (FPU) para un procesador de gráficos en particular. En algunas realizaciones, las unidades de ejecución 2108A-2108N soportan tipos de datos de números enteros y de coma flotante.

El conjunto de instrucciones de la unidad de ejecución incluye instrucciones SIMD. Los diversos elementos de datos se pueden almacenar como un tipo de datos empaquetados en un registro y la unidad de ejecución procesará los diversos elementos en función del tamaño de los datos de los elementos. Por ejemplo, cuando se 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 (datos de tamaño de palabra cuádruple (QW) elementos), 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 byte (B)). Sin embargo, son posibles diferentes tamaños de anchuras de vector y registros.

Una o más cachés de instrucción internas (por ejemplo, 2106) están incluidas en la lógica de ejecución de hilo 2100 a las instrucciones de hilo de caché para las unidades de ejecución. En algunas realizaciones, se incluyen uno o más cachés de datos (por ejemplo, 21 12) para almacenar en caché los datos del hilo durante la ejecución del hilo. En algunas realizaciones, se incluye un muestreador 2110 para proporcionar muestreo de textura para operaciones 3D y muestreo de medios para operaciones de medios. En algunas formas de realización, el muestreador 2110 incluye funcionalidad especializada de muestreo de textura o de medios para procesar 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 de medios envían solicitudes de iniciación de hilo a la lógica de ejecución de hilos 2100 mediante una lógica de generación y de despacho de hilos. Una vez que se ha procesado y rasterizado un grupo de objetos geométricos para obtener datos de píxel, se invoca una lógica de procesador de píxeles (por ejemplo, lógica de sombreador de píxeles, lógica de sombreador de fragmentos, etc.) dentro del procesador sombreador 2102 para computar adicionalmente información de salida y hacer que se escriban resultados para emitir superficies (por ejemplo, memorias intermedias de color, memorias intermedias de profundidad, memorias intermedias de estarcido, etc.). En algunas realizaciones, un sombreador de píxeles o un sombreador de fragmentos calcula los valores de los diversos atributos de vértice que se van a interpolar a lo largo del objeto rasterizado. En algunas realizaciones, una lógica de procesador de píxeles dentro del procesador sombreador 2102 ejecuta entonces un programa sombreador de píxeles o de fragmentos suministrado por una interfaz de programación de aplicaciones (API). Para ejecutar el programa sombreador, el procesador sombreador 2102 despacha hilos a una unidad de ejecución (por ejemplo, 2108A) mediante el despachador de hilos 2104. En algunas realizaciones, el sombreador de píxeles 2102 usa una lógica de muestreo de textura en el muestreador 2110 para acceder a datos de textura en correlaciones de textura almacenadas en memoria. Unas operaciones aritméticas sobre los datos de textura y los datos de geometría de entrada computan datos de color de píxel para cada fragmento geométrico, o descartan el procesamiento adicional de uno o más píxeles.

En algunas realizaciones, el puerto de datos 2114 proporciona un mecanismo de acceso de memoria para que la lógica de ejecución de hilos 2100 emita datos procesados a la memoria para su procesamiento en una canalización de salida de procesador de gráficos. En algunas realizaciones, el puerto de datos 2114 incluye o se acopla a una o

más memorias caché (por ejemplo, la caché de datos 2112) para almacenar en caché datos para un acceso de memoria mediante el puerto de datos.

La **Figura 22** es un diagrama de bloques que ilustra los formatos de instrucción de un procesador de gráficos 2200 de acuerdo con algunas realizaciones. En una o más formas de realización, las unidades de ejecución del procesador de gráficos soportan un conjunto de instrucciones que tiene instrucciones en múltiples formatos. Las cajas 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 formas de realización, el formato de instrucción 2200 descrito e ilustrado son macro-instrucciones, en el sentido de que las mismas son instrucciones suministradas a la unidad de ejecución, en contraposición a micro-operaciones resultantes de la decodificación de instrucciones una vez que se ha procesado la instrucción.

En algunas realizaciones, las unidades de ejecución de procesador de gráficos soportan de manera nativa instrucciones en un formato de instrucción de 128 bits 2210. Un formato de instrucción compactado de 64 bits 2230 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 710 proporciona acceso a todas las opciones de instrucción, mientras que algunas opciones y operaciones están restringidas en el formato de 64 bits 2230. Las instrucciones nativas disponibles en el formato de 64 bits 2230 varían según la realización. En algunas realizaciones, la instrucción se compacta en parte usando un conjunto de valores de índice en un campo de índice 2213. El hardware de la unidad de ejecución consulta 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 2210.

Para cada formato, el código de operación de instrucción 2212 define la operación que ha de realizar la unidad de ejecución. 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 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 2214 posibilita el control a través de ciertas opciones de ejecución, tal como la selección de canales (por ejemplo, predicación) y orden de canal de datos (por ejemplo, mezcla). Para instrucciones en el formato de instrucción de 128 bits 2210, un campo de tamaño de ejecución 2216 limita el número de canales de datos que se ejecutarán en paralelo. En algunas realizaciones, el campo de tamaño de ejecución 2216 no está disponible para su uso en el formato de instrucción compacto de 64 bits 2230.

Algunas instrucciones de la unidad de ejecución tienen hasta tres operandos, incluyendo dos operandos de origen, src0 2220, src1 2222 y un destino 2218. En algunas realizaciones, las unidades de ejecución soportan 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 2224), donde el código de operación de instrucción 2212 determina el número de operandos de origen. El último operando de origen de una instrucción puede ser un valor inmediato (por ejemplo, codificado de manera rígida) pasado con la instrucción.

En algunas realizaciones, el formato de instrucción de 128 bits 2210 incluye un campo de modo de acceso/dirección 2226 que especifica, por ejemplo, si se usa 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 es proporcionada directamente por bits en la instrucción.

En algunas realizaciones, el formato de instrucción de 128 bits 2210 incluye un campo de modo de dirección/acceso 2226, que especifica un modo de dirección y/o un modo de acceso para la instrucción. En una realización, el modo de acceso se usa para definir una alineación de acceso de datos para la instrucción. Algunas realizaciones soportan 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 puede usar un direccionamiento alineado por byte para los operandos de origen y de destino y, cuando está en un segundo modo, la instrucción puede usar un direccionamiento alineado por 16 bytes para todos los operandos de origen y de destino.

En una realización, la porción de modo de dirección del campo de modo de acceso/dirección 2226 determina si la instrucción va a usar un direccionamiento directo o indirecto. Cuando se usa el modo de direccionamiento de registro directo, unos bits de la instrucción 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 se puede computar 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 2212 para simplificar la decodificación del código de operación 2240. 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. El grupo de código de operación preciso mostrado es simplemente un ejemplo. En algunas realizaciones, un grupo de códigos de operación lógicos y

de movimiento 2242 incluye movimiento de datos e instrucciones lógicas (por ejemplo, mover (mov), comparar (cmp)). En algunas realizaciones, el grupo de movimiento y lógica 2242 comparte los cinco bits más significativos (MSB), donde las instrucciones mover (mov) están en forma de 0000xxxxb y las instrucciones de lógica están en forma de 0001xxxxb. Un grupo de instrucciones de control de flujo 2244 (por ejemplo, llamada, salto (jmp)) incluye instrucciones en forma de 0010xxxxb (por ejemplo, 0x20). Un grupo de instrucciones misceláneas 2246 incluye una mezcla de instrucciones, incluyendo instrucciones de sincronización (por ejemplo, espera, envío) en forma de 001 1xxxxb (por ejemplo, 0x30). Un grupo de instrucciones de cómputo paralelo 2248 incluye instrucciones aritméticas a nivel de componente (por ejemplo, añadir, multiplicar (mul)) en forma de 0100xxxxb (por ejemplo, 0x40). El grupo matemático paralelo 2248 realiza operaciones aritméticas en paralelo a través de canales de datos. El grupo de cómputo matemático vectorial 2250 incluye instrucciones aritméticas (por ejemplo, dp4) en forma de 0101xxxxb (por ejemplo, 0x50). El grupo de cómputo matemático vectorial realiza la aritmética tal como los cálculos de producto escalar en operandos vectoriales.

Canalización de gráficos

La **Figura 23** es un diagrama de bloques de otra forma de realización de un procesador de gráficos 2300. 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, el procesador de gráficos 2300 incluye un canal de gráficos 2320, un canal de medios 2330, un motor de visualización 2340, una lógica de ejecución de hilos 2350 y un canal de salida de renderizado 2370. En algunas realizaciones, el procesador de gráficos 2300 es un procesador de gráficos dentro de un sistema de procesamiento de múltiples núcleos que incluye uno o más núcleos de procesamiento de fin general. El procesador de gráficos es controlado por escrituras de registro en uno o más registros de control (no mostrados) o mediante comandos emitidos al procesador de gráficos 2300 mediante una interconexión en anillo 2302. En algunas realizaciones, la interconexión en anillo 2302 acopla el procesador de gráficos 2300 a otros componentes de procesamiento, tales como otros procesadores de gráficos o procesadores de propósito general. Los comandos desde la interconexión en anillo 2302 son interpretados por un transmisor por flujo continuo de comandos 2303, que suministra instrucciones a componentes individuales de la canalización de gráficos 2320 o la canalización de medios 2330.

En algunas realizaciones, el transmisor de comandos 2303 dirige la operación de un buscador de vértices 2305 que lee datos de vértices de la memoria y ejecuta comandos de procesamiento de vértices proporcionados por el transmisor de comandos 2303. En algunas realizaciones, el extractor de vértices 2305 proporciona datos de vértices a un sombreador de vértices 2307, que realiza operaciones de transformación espacial de coordenadas y de iluminación en cada vértice. En algunas realizaciones, el extractor de vértices 2305 y el sombreador de vértices 2307 ejecutan instrucciones de procesamiento de vértices despachando hilos de ejecución a las unidades de ejecución 2352A-2352B mediante un despachador de hilos 2331.

En algunas realizaciones, las unidades de ejecución 2352A-2352B son una matriz de procesadores de vectores que tienen un conjunto de instrucciones para realizar operaciones de gráficos y de medios. En algunas realizaciones, las unidades de ejecución 2352A-2352B tienen una caché de L1 2351 anexada que es específica para cada matriz o que se comparte entre las matrices. La caché se puede configurar como una caché de datos, una caché de instrucciones o una única caché que se subdivide para contener datos e instrucciones en diferentes subdivisiones.

En algunas realizaciones, la canalización de gráficos 2320 incluye componentes de teselación para realizar teselación acelerada por hardware de objetos 3D. En algunas realizaciones, un sombreador de casco programable 811 configura las operaciones de teselación. Un sombreador de dominio programable 817 proporciona una evaluación de fondo de la salida de teselación. Un teselador 2313 opera en la dirección del hull shader 2311 y contiene una lógica de propósito especial para generar un conjunto de objetos geométricos detallados en función de un modelo geométrico en bruto que se proporciona como entrada al conducto de gráficos 2320. En algunas realizaciones, si no se usa la teselación, pueden eludirse los componentes de teselación (por ejemplo, el sombreador de casco 2311, el teselador 2313 y el sombreador de dominio 2317).

En algunas realizaciones, unos objetos geométricos completos pueden ser procesados por un sombreador de geometría 2319 mediante uno o más hilos despachados a las unidades de ejecución 2352A-2352B, o puede avanzar directamente al recortador 2329. En algunas realizaciones, el sombreador de geometría opera sobre objetos geométricos enteros, en lugar de vértices o parches de vértices como en fases previas de la canalización de gráficos. Si la teselación está deshabilitada, el sombreador de geometría 2319 recibe una entrada desde el sombreador de vértices 2307. En algunas realizaciones, el sombreador de geometría 2319 se puede programar mediante un programa sombreador de geometría para realizar un teselación de geometría si las unidades de teselación están deshabilitadas.

Antes de la rasterización, un recortador 2329 procesa datos de vértices. El recortador 2329 puede ser un recortador de función fija o un recortador programable que tiene funciones de recorte y de sombreador de geometría. En algunas realizaciones, un componente de rasterización y prueba de profundidad 2373 en el canal de salida de renderizado

2370 envía sombreadores de píxeles para convertir los objetos geométricos en sus representaciones por píxel. En algunas realizaciones, la lógica de sombreador de píxeles se incluye en la lógica de ejecución de hilos 2350. En algunas realizaciones, una aplicación puede omitir el componente de prueba de rasterizador y de profundidad 2373 y acceder a datos de vértice sin rasterizar mediante una unidad de salida de flujo 2323.

El procesador de gráficos 2300 tiene un bus de interconexión, una estructura de interconexión o algún otro mecanismo de interconexión que permite el paso de datos y de mensajes entre los componentes principales del procesador. En algunas realizaciones, las unidades de ejecución 2352A-2352B y la caché o cachés 2351 asociadas, el muestreador de textura y de medios 2354 y la caché de textura/muestreador 2358 se interconectan mediante un puerto de datos 2356 para realizar un acceso de memoria y comunicarse con componentes de canalización de salida de representación del procesador. En algunas realizaciones, el muestreador 2354, las cachés 2351, 2358 y las unidades de ejecución 2352A-2352B tienen, cada uno, rutas de acceso de memoria separadas.

En algunas realizaciones, la canalización de salida de representación 2370 contiene un componente de prueba de rasterizador y de profundidad 2373 que convierte objetos basados en vértices en una representación asociada basada en píxeles. En algunas realizaciones, la lógica de rasterizador incluye una unidad generadora de ventanas/enmascaradora para realizar una rasterización de líneas y de triángulos de función fija. Una caché de representación 2378 y una caché de profundidad 2379 asociadas también están disponibles en algunas realizaciones. Un componente de operaciones de píxel 2377 realiza operaciones basadas en píxeles sobre los datos, aunque, en algunas instancias, las operaciones de píxel asociadas con operaciones 2D (por ejemplo, transferencias de imagen de bloque de bits con mezcla) son realizadas por el motor 2D 2341, o son sustituidas en el momento de la visualización por el controlador de visualización 2343 usando planos de visualización de superposición. En algunas realizaciones, está disponible una caché de L3 compartida 2375 para todos los componentes de gráficos, permitiendo compartir datos sin el uso de memoria de sistema principal.

En algunas realizaciones, la canalización de medios del procesador de gráficos 2330 incluye un motor de medios 2337 y un extremo frontal de vídeo 2334. En algunas realizaciones, el extremo frontal de vídeo 2334 recibe comandos de canalización desde el transmisor de envío por flujo continuo 2303. En algunas realizaciones, la canalización de medios 2330 incluye un transmisor de envío por flujo continuo separado. En algunas realizaciones, el extremo frontal de vídeo 2334 procesa comandos de medios antes de enviar el comando al motor de medios 2337. En algunas realizaciones, el motor de medios 2337 incluye una funcionalidad de generación de hilos para generar hilos para su envío a la lógica de ejecución de hilos 2350 a través del despachador de hilos 2331.

En algunas realizaciones, el procesador de gráficos 2300 incluye un motor de visualización 2340. En algunas realizaciones, el motor de visualización 2340 es externo al procesador 2300 y se acopla con el procesador de gráficos a través de la interconexión de anillo 2302, o algún otro bus o estructura de interconexión. En algunas realizaciones, el motor de visualización 2340 incluye un motor 2D 2341 y un controlador de visualización 2343. En algunas realizaciones, el motor de visualización 2340 contiene lógica de propósito especial capaz de operar independientemente de la canalización 3D. En algunas realizaciones, el controlador de visualización 2343 se acopla con un dispositivo de visualización (no mostrado), que puede ser un dispositivo de visualización integrado en sistema, como en un ordenador portátil, o un dispositivo de visualización externo adjunto mediante un conector de dispositivo de visualización.

En algunas realizaciones, la canalización de gráficos 2320 y la canalización de medios 2330 se pueden configurar para realizar operaciones basándose en múltiples interfaces de programación de gráficos y de medios y no son específicas de ninguna interfaz de programación de aplicaciones (API) concreta. En algunas realizaciones, el software del controlador para el procesador de gráficos traduce llamadas API que son específicas a gráficos o a bibliotecas de medios particulares en comandos que pueden procesarse por el procesador de gráficos. En algunas realizaciones, se proporciona soporte para la biblioteca de gráficos abierta (OpcnGL), el lenguaje de cómputo abierto (OpenCL) y/o la API de cómputo y gráficos de Vulkan, todos del Grupo Khronos. En algunas realizaciones, también se puede proporcionar soporte para la biblioteca Direct3D de Microsoft Corporation. En algunas formas de realización, una combinación de estas bibliotecas puede ser compatible. También puede ser compatible con la biblioteca Open Source Computer Vision Library (OpenCV). Una futura API con un conducto de 3D compatible también puede ser compatible si se puede hacer una asignación desde el conducto de la futura API al conducto del procesador de gráficos.

Programación del conducto de gráficos

La **Figura 24A** es un diagrama de bloques que ilustra un formato de orden de procesador de gráficos 2400 de acuerdo con algunas formas de realización. La **Figura 24B** es un diagrama de bloques que ilustra una secuencia de comandos del procesador de gráficos 2410 de acuerdo con una realización. Los cuadros con líneas sólidas en la **Figura 24A** 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 ilustrativa 2400 de la **Figura 24A** incluye campos de datos para identificar un cliente objetivo 2402 del comando, un código de operación de comando (código de operación) 2404 y los datos relevantes 2406 para el comando. En algunos comandos también se incluyen un subcódigo de operación 2405 y un tamaño de comando 2408.

En algunas formas de realización, el cliente 2402 especifica la unidad cliente del dispositivo de gráficos que procesa los datos de la orden. En algunas formas de realización, un analizador de órdenes del procesador de gráficos examina el campo cliente de cada orden para condicionar el procesamiento posterior de la orden y dirigir los datos de la orden a la unidad cliente apropiada. En algunas formas de realización, 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 un conducto de procesamiento correspondiente que procesa las órdenes. Una vez que la orden es recibida por la unidad cliente, la unidad cliente lee el código de operación 2404 y, si está presente, el subcódigo de operación 2405 para determinar la operación a llevar a cabo. La unidad de cliente realiza el comando usando información en el campo de datos 2406. Para algunos comandos, se espera que un tamaño de comando explícito 2408 especifique 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 mediante múltiplos de una palabra doble.

El diagrama de flujo en la **Figura 24B** muestra una secuencia de comando de procesador de gráficos ilustrativa 2410. En algunas formas de realización, el software o firmware de un sistema de procesamiento de datos que caracteriza una forma de realización de un procesador de gráficos utiliza una versión de la secuencia de órdenes mostrada para configurar, ejecutar y terminar un conjunto de operaciones gráficas. Se muestra una secuencia de órdenes de muestra y se describe para los fines de ejemplo únicamente ya que las formas de realización no se limitan a estas órdenes específicas o para esta secuencia de órdenes. Por otra parte, las órdenes se pueden emitir como un lote de órdenes en una secuencia de órdenes, de tal forma que el procesador de gráficos procesará la secuencia de órdenes en al menos parcialmente concurrencia.

En algunas formas de realización, la secuencia de órdenes del procesador de gráficos 2410 puede comenzar con una orden de descarga de conducto 2412 para hacer que cualquier conducto de gráficos activo complete las órdenes actualmente pendientes para el conducto. En algunas formas de realización, el conducto de 3D 2422 y el conducto de medios 2424 no operan al mismo tiempo. Se realiza el vaciado de la canalización para hacer que la canalización de gráficos activa complete algún comando pendiente. En respuesta a un vaciado de canalización, el analizador de comando 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 cachés de lectura relevantes. Opcionalmente, cualquier dato en la caché del representador que se marca 'sucio' puede vaciarse a memoria. En algunas realizaciones, puede usarse el comando de vaciado de canalización 2412 para la sincronización de canalización o antes de colocar el procesador de gráficos en un estado de baja potencia.

En algunas realizaciones, se usa un comando de selección de canalización 2413 cuando una secuencia de comando requiere que el procesador de gráficos cambie explícitamente entre canalizaciones. En algunas realizaciones, se requiere un comando de selección de canalización 2413 solo 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 2412 inmediatamente antes de una conmutación de canalización mediante el comando de selección de canalización 2413.

En algunas realizaciones, un comando de control de canalización 2414 configura una canalización de gráficos para su funcionamiento y se usa para programar la canalización 3D 2422 y la canalización de medios 2424. En algunas realizaciones, el comando de control de canalización 2414 configura el estado de canalización para la canalización activa. En una realización, el comando de control de canalización 2414 se usa para la sincronización de 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, se usan comandos de estado de memoria intermedia de retorno 2416 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 usa uno o más memorias intermedias de retorno para almacenar datos de salida y realizar comunicación entre hilos. En algunas realizaciones, el estado de memoria intermedia de retorno 2416 incluye seleccionar el tamaño y el número de memorias intermedias de retorno que hay que usar para un conjunto de operaciones de canalización.

Los comandos restantes en la secuencia de comandos difieren basándose en la canalización activa para las operaciones. Basándose en una determinación de canalización 2420, la secuencia de comandos se adapta a la canalización de 3D 2422 comenzando con el estado de canalización de 3D 2430, o a la canalización de medios 2424 comenzando en el estado de canalización de medios 2440.

Los comandos para configurar el estado de canalización de 3D 2430 incluyen comandos de ajuste de estado de 3D para el estado de memoria intermedia de vértice, el estado de elemento de vértice, el estado de color constante, el 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 del estado de canalización de 3D 2430 también son capaces de deshabilitar u omitir selectivamente ciertos elementos de canalización si esos elementos no se van a usar.

En algunas realizaciones, se usa el comando de primitiva 3D 2432 para enviar que se procesen primitivas 3D por la canalización 3D. Los comandos y parámetros asociados que se pasan al procesador de gráficos mediante el comando de primitiva 3D 2432 se reenvían a la función de extracción de vértice en la canalización de gráficos. La función de búsqueda de vértices utiliza los datos de comando 2432 primitivos 3D 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, se usa el comando de primitiva 3D 2432 para realizar operaciones de vértice en primitivas 3D mediante sombreadores de vértice. Para procesar sombreadores de vértices, la canalización de 3D 2422 despacha hilos de ejecución de sombreador a unidades de ejecución de procesador de gráficos.

En algunas realizaciones, la canalización de 3D 2422 se desencadena mediante un comando o evento de ejecución 2434. En algunas realizaciones, una escritura de registro desencadena una ejecución de comando. En algunas realizaciones, la ejecución se desencadena mediante un comando 'ir' o 'poner en marcha' en la secuencia de comandos. En una realización, la ejecución de comando se desencadena usando un comando de sincronización de canalización para vaciar la secuencia de comandos a través de la canalización de gráficos. La canalización de 3D realizará un procesamiento de geometría para las primitivas 3D. Una vez que se han completado las operaciones, los objetos geométricos resultantes se rasterizan y el motor de píxeles da color a los píxeles resultantes. También se pueden incluir comandos adicionales para controlar el sombreado de píxeles y las operaciones de extremo trasero de píxeles para esas operaciones.

En algunas formas de realización, la secuencia de órdenes del procesador de gráficos 2410 sigue la trayectoria del conducto de medios 2424 cuando se llevan a cabo operaciones de medios. En general, la utilización y la manera de programación específicas para el conducto de medios 2424 depende de las operaciones de medios o de cómputo a llevar a cabo. Las operaciones de decodificación de medios específicas pueden descargarse en 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 fin general. En una realización, el canal de medios también incluye elementos para operaciones de unidad de procesador de gráficos de propósito general (GPGPU), donde el procesador de gráficos se usa para realizar operaciones de vector SIMD usando programas de sombreado computacional que no están relacionados explícitamente con la representación de primitivas de gráficos,

En algunas realizaciones, la canalización de medios 2424 está configurada de manera similar a la canalización 3D 2422. Un conjunto de comandos para configurar el estado de canalización de medios 2440 se envía o se coloca en una cola de comandos antes de que el objeto de medios ordene 2442. En algunas realizaciones, los comandos de estado de canalización de medios 2440 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 y codificación de video dentro del canal de medios, como el formato de codificación o decodificación. En algunas realizaciones, los comandos de estado de canalización de medios 2440 también soportan el uso de uno o más punteros a elementos de estado "indirecto" que contienen un lote de ajustes de estado.

En algunas realizaciones, los comandos de objeto de medios 2442 suministran punteros a objetos de medios para su procesamiento por la canalización de medios. Los objetos multimedia incluyen memorias intermedias de memoria que contienen datos de vídeo a procesar. En algunas realizaciones, todos los estados de canalización de medios han de ser válidos antes de emitir un comando de objeto de medios 2442. Una vez que se ha configurado el estado de canalización y los comandos de objeto de medios 2442 se han puesto en cola, la canalización de medios 2424 se desencadena mediante un comando de ejecución 2444 o un evento de ejecución equivalente (por ejemplo, una escritura de registro). La salida desde la canalización de medios 2424 puede post-procesarse entonces mediante operaciones proporcionadas por la canalización de 3D 2422 o la canalización de medios 2424. En algunas realizaciones, las operaciones de GPGPU se configuran y se ejecutan de una manera similar a la de las operaciones de medios.

Arquitectura de software de gráficos

La **Figura 25** ilustra una arquitectura de software de gráficos de ejemplo para un sistema de procesamiento de datos 2500 de acuerdo con algunas formas de realización. En algunas formas de realización, la arquitectura de software incluye una aplicación de gráficos 3D 2510, un sistema operativo 2520 y al menos un procesador 2530. En algunas formas de realización, el procesador 2530 incluye un procesador de gráficos 2532 y uno o más núcleos de procesador de propósito general 2534. La aplicación de gráficos 2510 y el sistema operativo 2520 se ejecutan, cada uno, en la memoria de sistema 2550 del sistema de procesamiento de datos.

En algunas realizaciones, la aplicación de gráficos 3D 2510 contiene uno o más programas de sombreado que incluyen instrucciones de sombreado 2512. Las instrucciones de lenguaje de sombreador pueden estar en un lenguaje de sombreador de alto nivel, tal como el Lenguaje de Sombreador de Alto Nivel (HLSL) o el Lenguaje de Sombreador

OpenGL (GLSL). La aplicación también incluye instrucciones ejecutables 2514 en un lenguaje máquina adecuado para su ejecución por el núcleo de procesador de fin general 2534. La aplicación también incluye los objetos de gráficos 2516 definidos por los datos de vértices.

En algunas realizaciones, el sistema operativo 2520 es un Microsoft® Windows® sistema operativo 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 kernel de Linux. El sistema operativo 2520 puede soportar una API de gráficos 2522 tal como la API Direct3D, la API OpenGL o la API Vulkan. Cuando la API Direct3D está en uso, el sistema operativo 2520 usa un compilador de sombreador frontal 2524 para compilar cualquier instrucción de sombreador 25 12 en HLSL en un lenguaje de sombreador 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 2510. En algunas realizaciones, las instrucciones de sombreador 2512 se proporcionan en una forma intermedia, tal como una versión de la representación intermedia portátil convencional (SPIR) usada por la API de Vulkan.

En algunas realizaciones, el controlador de gráficos de modo de usuario 2526 contiene un compilador de sombreador de extremo trasero 2527 para convertir las instrucciones de sombreador 2512 en una representación específica de hardware. Cuando la API OpenGL está en uso, las instrucciones de sombreado 2512 en el lenguaje de alto nivel GLSL se pasan a un controlador de gráficos 2526 en modo de usuario para su compilación. En algunas realizaciones, el controlador de gráficos de modo de usuario 2526 usa las funciones de modo de núcleo de sistema operativo 2528 para comunicarse con un controlador de gráficos de modo de núcleo 2529. En algunas realizaciones, el controlador de gráficos de modo de núcleo 2529 se comunica con el procesador de gráficos 2532 para despachar comandos e instrucciones.

Implementaciones principales 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 una lógica diversa 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. Tales representaciones, conocidas como "núcleos de IP", son unidades reutilizables de lógica para un circuito integrado que pueden almacenarse en un medio legible por máquina tangible 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 el presente documento.

La **Figura 26** es un diagrama de bloques que ilustra un sistema de desarrollo de núcleo IP 2600 que se puede utilizar para fabricar un circuito integrado para llevar a cabo operaciones de acuerdo con una forma de realización. El sistema de desarrollo de núcleo IP 2600 se puede utilizar para generar diseños modulares reutilizables que se pueden incorporar a un diseño mayor o se pueden utilizar para construir un circuito integrado completo (por ejemplo, un circuito integrado SOC). Una instalación de diseño 2630 puede generar una simulación de software 2610 de un diseño de núcleo de IP en un lenguaje de programación de alto nivel (por ejemplo, C/C++). La simulación de software 2610 se puede usar para diseñar, probar y verificar el comportamiento del núcleo IP usando un modelo de simulación 2612. El modelo de simulación 2612 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) 2615 a partir del modelo de simulación 2612. El diseño de RTL 2615 es una abstracción del comportamiento del circuito integrado que modela el flujo de señales digitales entre registros de hardware, incluyendo la lógica asociada realizada usando las señales digitales modeladas. Además de un diseño de RTL 2615, también se pueden crear, diseñar o sintetizar diseños de nivel inferior a nivel de lógica o a nivel de transistores. Por lo tanto, los detalles particulares del diseño y simulación inicial pueden variar.

El diseño RTL 2615 o equivalente puede ser sintetizado adicionalmente por la instalación de diseño en un modelo de hardware 2620, 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 de núcleo de IP puede almacenarse para su entrega a una instalación de fabricación de 3os 2665 usando memoria no volátil 2640 (por ejemplo, disco duro, memoria flash o cualquier medio de almacenamiento no volátil). Como alternativa, el diseño de núcleo de IP puede transmitirse (por ejemplo, mediante Internet) a través de una conexión alámbrica 2650 o conexión inalámbrica 2660. La instalación de fabricación 2665 puede fabricar entonces un circuito integrado que se basa, al menos en parte, en el diseño de núcleo de IP. El circuito integrado fabricado puede configurarse para realizar operaciones de acuerdo con al menos una realización descrita en el presente documento.

Circuito integrado en un microprocesador de sistema de ejemplo

Las **Figuras 27-29** ilustran circuitos integrados ilustrativos y procesadores gráficos asociados que pueden fabricarse 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 lógica, incluidos procesadores/núcleos de gráficos adicionales, controladores de interfaz periférica o núcleos de procesador de uso general.

La **Figura 27** es un diagrama de bloques que ilustra un circuito integrado en un microprocesador de sistema 2700 de ejemplo que se puede fabricar utilizando uno o más núcleos IP, de acuerdo con una forma de realización. El circuito integrado de ejemplo 2700 incluye uno o más procesadores de aplicación 2705 (por ejemplo, CPU), al menos un procesador de gráficos 2710, y puede incluir adicionalmente un procesador de imagen 2715 y/o un procesador de vídeo 2720, cualquiera de los cuales puede ser un núcleo IP modular de la misma o múltiples instalaciones de diseño diferentes. El circuito integrado 2700 incluye una lógica de bus o de periféricos que incluye un controlador de USB 2725, un controlador de UART 2730, un controlador de SPI/SDIO 2735 y un controlador de I^2S/I^2C 2740. Además, el circuito integrado puede incluir un dispositivo de visualización 2745 acoplado a uno o más de un controlador 2750 de interfaz multimedia de alta definición (HDMI) y una interfaz de visualización de interfaz de procesador de industria móvil (MIPI) 2755. El almacenamiento puede proporcionarse por un subsistema de memoria flash 2760 que incluye la memoria flash y un controlador de memoria flash. La interfaz de memoria se puede proporcionar mediante un controlador de memoria 2765 para el acceso a dispositivos de memoria SDRAM o SRAM. Algunos circuitos integrados incluyen adicionalmente un motor de seguridad integrado 2770.

La **Figura 28** es un diagrama de bloques que ilustra un procesador de gráficos 2810 de ejemplo de un circuito integrado en un microprocesador de sistema que se puede fabricar utilizando uno o más núcleos IP, de acuerdo con una forma de realización. El procesador de gráficos 2810 puede ser una variante del procesador de gráficos 2710 de la **Figura 27.** El procesador de gráficos 2810 incluye un procesador de vértices 2805 y uno o más procesadores de fragmentos 2815A-2815N (por ejemplo, 2815A, 2815B, 2815C, 2815D a 2815N-1 y 2815N). El procesador de gráficos 2810 puede ejecutar diferentes programas sombreadores mediante una lógica separada, de modo que el procesador de vértices 2805 está optimizado para ejecutar operaciones para programas sombreadores de vértices, mientras que los uno o más procesadores de fragmentos 2815A-2815N ejecutan operaciones de sombreado de fragmentos (por ejemplo, píxeles) para programas sombreadores de fragmentos o de píxeles. El procesador de vértices 2805 realiza la fase de procesamiento de vértices de la canalización de gráficos 3D y genera primitivas y datos de vértice. El procesador o procesadores de fragmentos 2815A-2815N usan los datos de primitiva y de vértice generados por el procesador de vértices 2805 para producir una memoria intermedia de tramas que se visualiza en un dispositivo de visualización. En una realización, el procesador o procesadores de fragmentos 2815A-2815N están optimizados para ejecutar programas sombreadores de fragmentos según lo previsto en la API de OpenGL, que se pueden usar para realizar operaciones similares como un programa sombreador de píxeles según lo previsto en la API de Direct 3D.

El procesador de gráficos 2810 incluye adicionalmente una o más unidades de gestión de memoria (MMU) 2820A-2820B, caché(s) 2825A-2825B e interconexiones de circuito) 2830A-2830B. La una o más MMU 2820A-2820B proporcionan mapeo de dirección virtual a física para el circuito integrado 2810, incluyendo para el procesador de vértices 2805 y/o el procesador o procesadores de fragmentos 2815A-2815N, que pueden hacer referencia a los datos de vértice o de imagen/textura almacenados en memoria, además de los datos de vértice o imagen/textura almacenados en la una o más caché o cachés 2825A-2825B. En una forma de realización, la una o más MMU 2825A-2825B se pueden sincronizar con otras MMU dentro del sistema, incluyendo una o más MMU asociadas con el uno o más procesadores de aplicación 2705, procesador de imagen 2715, y/o procesador de vídeo 2720 de la **Figura 27**, de tal forma que cada procesador 2705-2720 puede participar en un sistema de memoria virtual compartido o unificado. La(s) interconexión(es) de circuito(s) 2830A-2830B permite(n) al procesador de gráficos 2810 interactuar con otros núcleos IP dentro del SoC, ya sea por medio de un bus interno del SoC o por medio de una conexión directa, de acuerdo con las formas de realización.

La **Figura 29** es un diagrama de bloques que ilustra un procesador de gráficos adicional de ejemplo 2910 de un circuito integrado en un microprocesador de sistema que se puede fabricar utilizando uno o más núcleos IP, de acuerdo con una forma de realización. El procesador de gráficos 2910 puede ser una variante del procesador de gráficos 2710 de la **Figura 27.** El procesador de gráficos 2910 incluye una o más MMU 2820A-2820B, cachés 2825A-2825B y interconexiones de circuito 2830A-2830B del circuito integrado 2800 de la

Figura 28.

El procesador de gráficos 2910 incluye uno o más núcleos de sombreado 2915A-2915N (por ejemplo, 2915A, 2915B, 2915C, 2915D, 2915E, 2915F, a 2915N-1 y 2915N), que proporciona una arquitectura de núcleo de sombreador unificada en la que un solo núcleo o tipo o núcleo puede ejecutar todo tipo 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 formas de realización e implementaciones. Adicionalmente, el procesador de gráficos 2910 incluye un gestor de tareas inter-núcleo 2905, que actúa como un despachador de hilo para despachar hilos de ejecución a uno o más núcleos de sombreador 2915A-2915N y una unidad de mosaico 2918 para acelerar las operaciones de mosaico para la representación basada en mosaico, en las que las operaciones de representación para una escena se subdividen en

el espacio de imágenes, por ejemplo, para aprovechar la coherencia de espacio local dentro de una escena o para optimizar el uso de cachés internas.

REIVINDICACIONES

1. Un método, implementado en un procesador de gráficos, para realizar operaciones de un marco de aprendizaje automático que proporciona una biblioteca de primitivas de aprendizaje automático, siendo las primitivas de aprendizaje automático operaciones básicas a realizar mediante algoritmos de aprendizaje automático, comprendiendo el método:

cargar elementos de matrices en una primera memoria del procesador de gráficos, en donde la primera memoria es una memoria global del procesador de gráficos;
transferir un subconjunto de elementos de las matrices desde la primera memoria a una segunda memoria del procesador de gráficos, siendo la segunda memoria local a un conjunto de recursos de procesamiento del procesador de gráficos; y
desencadenar la ejecución de un núcleo de cómputo en el procesador de gráficos, en donde el núcleo de cómputo realiza una operación de multiplicación de matriz dispersa en el subconjunto de elementos de las matrices y una o más operaciones por elementos en la salida de la operación de multiplicación de matriz dispersa antes de que se transfiera la salida a la primera memoria, en donde el marco de aprendizaje automático es proporcionar una primitiva de aprendizaje automático al núcleo de cómputo para especificar una o más operaciones por elementos, en donde una o más operaciones por elementos incluyen la aplicación de una función de activación a la salida.

2. El método según la reivindicación 1, en donde la función de activación es una función unitaria lineal rectificada.

3. El método según la reivindicación 1 o 2, en donde una o más operaciones de elementos incluyen adicionalmente realizar una operación de polarización antes de realizar la función de activación.

4. El método según cualquiera de las reivindicaciones 1-3, en donde el núcleo de cómputo debe realizar la operación de multiplicación de matriz dispersa en el subconjunto de elementos de las matrices a través de una o más primitivas proporcionadas a través del marco de aprendizaje automático.

5. El método según la reivindicación 4, en donde el marco de aprendizaje automático debe proporcionar una primitiva para realizar un producto escalar de enteros de 8 bits.

6. Un medio no transitorio legible por máquina que incluye instrucciones que, cuando son ejecutadas por uno o más procesadores, hacen que uno o más procesadores realicen operaciones de un método como en cualquiera de las reivindicaciones 1 a 5.

7. Un sistema de procesamiento de datos que comprende:
un dispositivo de memoria; y
uno o más procesadores configurados para ejecutar instrucciones almacenadas en el dispositivo de memoria, en donde las instrucciones hacen que uno o más procesadores realicen operaciones de un marco de aprendizaje automático que proporciona una biblioteca de primitivas de aprendizaje automático, siendo las primitivas de aprendizaje automático operaciones básicas que debe realizar la máquina algoritmos de aprendizaje, en los que uno o más procesadores incluyen un procesador de gráficos y las instrucciones hacen que uno o más procesadores:
cargar elementos de matrices en una primera memoria del procesador de gráficos, en donde la primera memoria es una memoria global del procesador de gráficos;
transferir un subconjunto de elementos de las matrices desde la primera memoria a una segunda memoria del procesador gráfico, siendo la segunda memoria local a un conjunto de recursos de procesamiento del procesador gráfico; y
desencadenar la ejecución de un núcleo de cómputo en el procesador de gráficos, en donde el núcleo de cómputo realiza una operación de multiplicación de matriz dispersa en el subconjunto de elementos de las matrices y una o más operaciones de elementos en la salida de la operación de multiplicación de matriz dispersa antes de que se transfiera la salida a la primera memoria, en donde el marco de aprendizaje automático es proporcionar una primitiva de aprendizaje automático al núcleo de cómputo para especificar una o más operaciones por elementos, incluyendo la una o más operaciones por elementos la aplicación de una función de activación a la salida.

8. El sistema de procesamiento de datos según la reivindicación 7, en donde la función de activación es una función unitaria lineal rectificada.

9. El sistema de procesamiento de datos según la reivindicación 7 u 8, en donde una o más operaciones de elementos incluyen adicionalmente realizar una operación de polarización antes de realizar la función de activación.

10. El sistema de procesamiento de datos según cualquiera de las reivindicaciones 7-9, en donde el núcleo de cómputo debe realizar la operación de multiplicación de matriz dispersa en el subconjunto de elementos de las matrices a través de una o más primitivas proporcionadas a través del marco de aprendizaje automático.

11. El sistema de procesamiento de datos según la reivindicación 10, en donde el marco de aprendizaje automático debe proporcionar una primitiva para realizar un producto escalar de enteros de 8 bits.

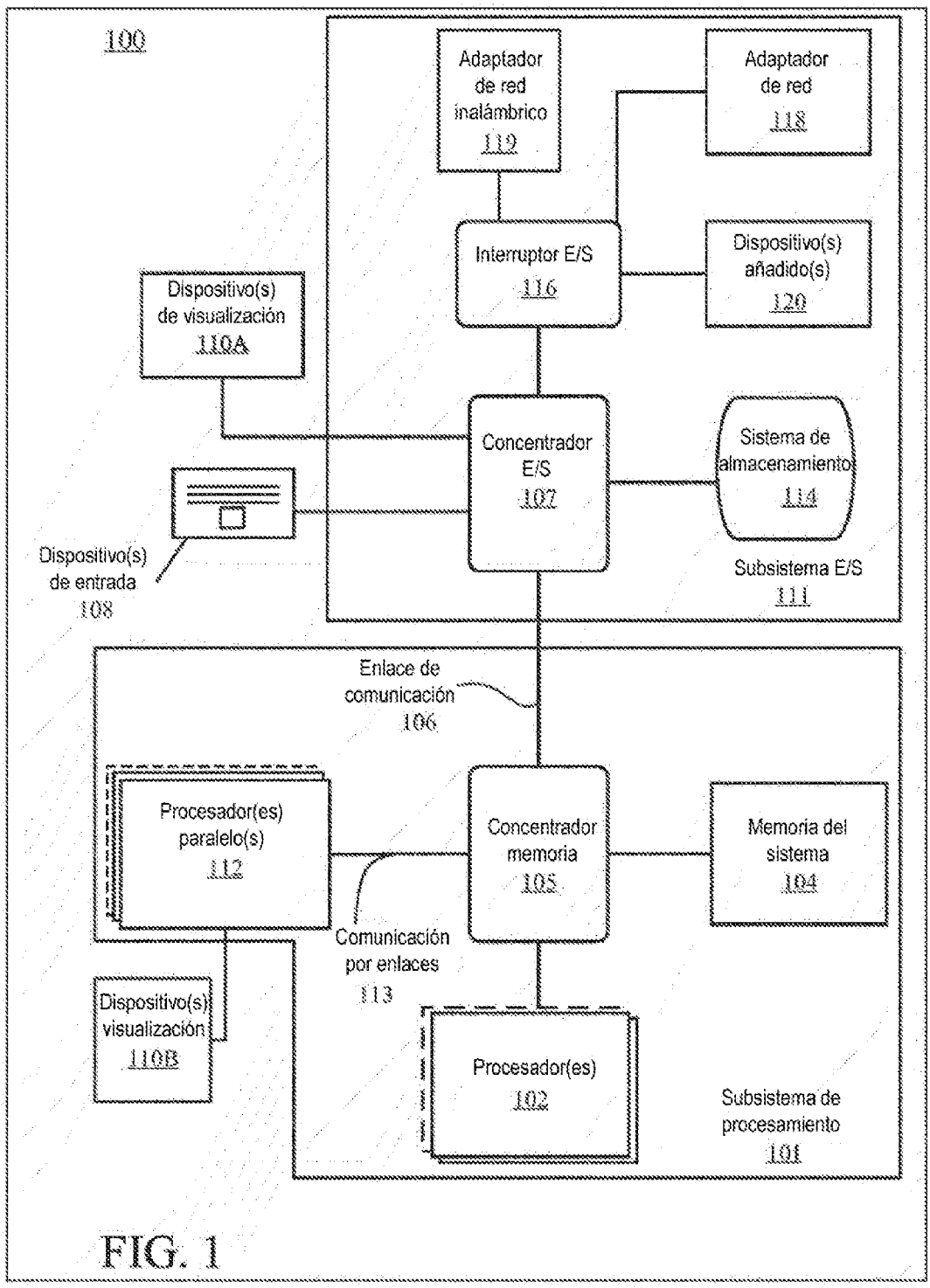

100
Adaptador de red inalámbrico 119
Adaptador de red 118
Interruptor E/S 116
Dispositivo(s) añadido(s) 120
Dispositivo(s) de visualización 110A
Concentrador E/S 107
Sistema de almacenamiento 114
Dispositivo(s) de entrada 108
Subsistema E/S 111
Enlace de comunicación 106
Procesador(es) paralelo(s) 112
Concentrador memoria 105
Memoria del sistema 104
Comunicación por enlaces 113
Dispositivo(s) visualización 110B
Procesador(es) 102
Subsistema de procesamiento 101

FIG. 1

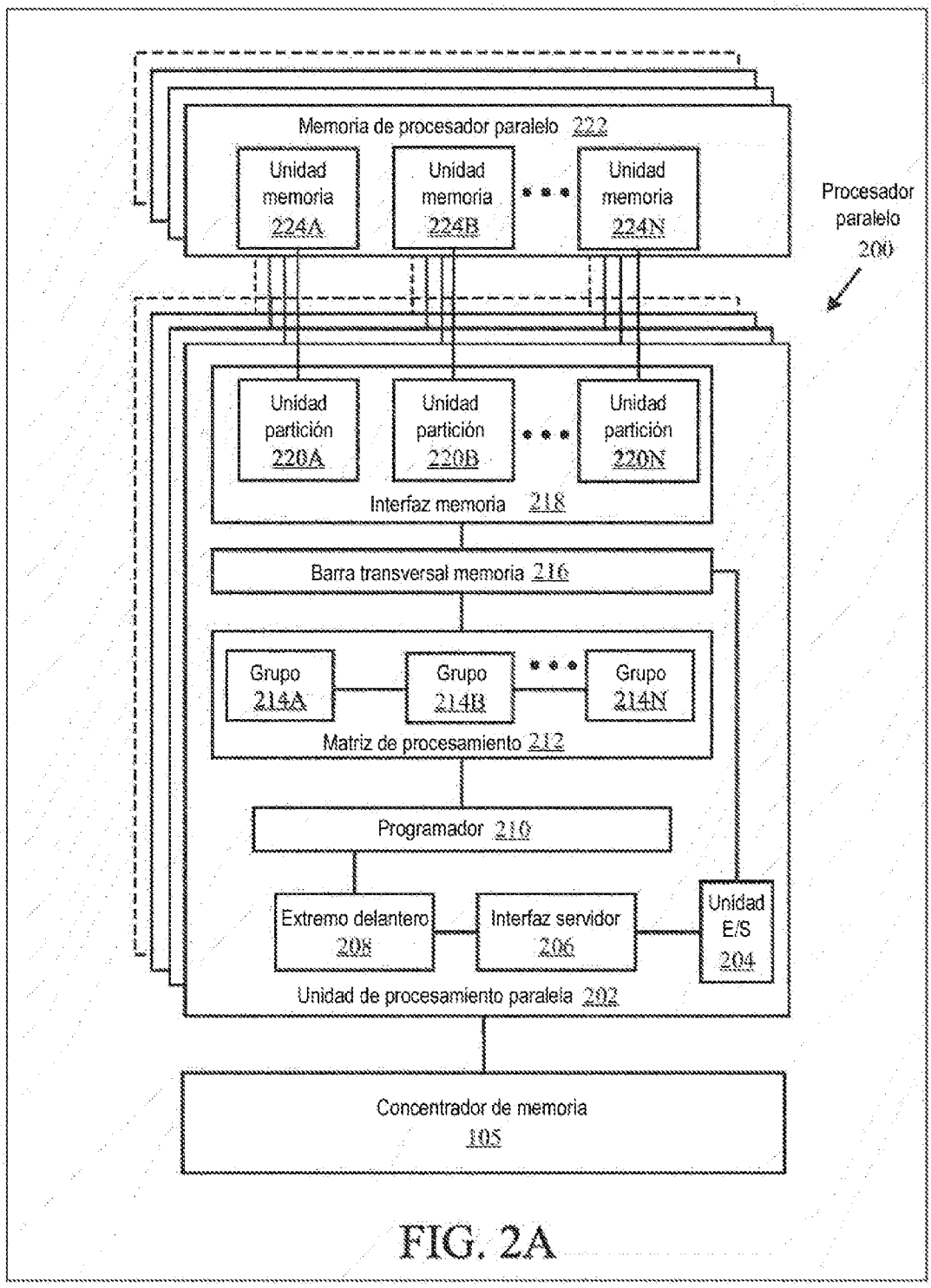

Memoria de procesador paralelo 222
Unidad memoria 224A
Unidad memoria 224B
Unidad memoria 224N
Procesador paralelo 200
Unidad partición 220A
Unidad partición 220B
Unidad partición 220N
Interfaz memoria 218
Barra transversal memoria 216
Grupo 214A
Grupo 214B
Grupo 214N
Matriz de procesamiento 212
Programador 210
Extremo delantero 208
Interfaz servidor 206
Unidad E/S 204
Unidad de procesamiento paralela 202
Concentrador de memoria 105

FIG. 2A

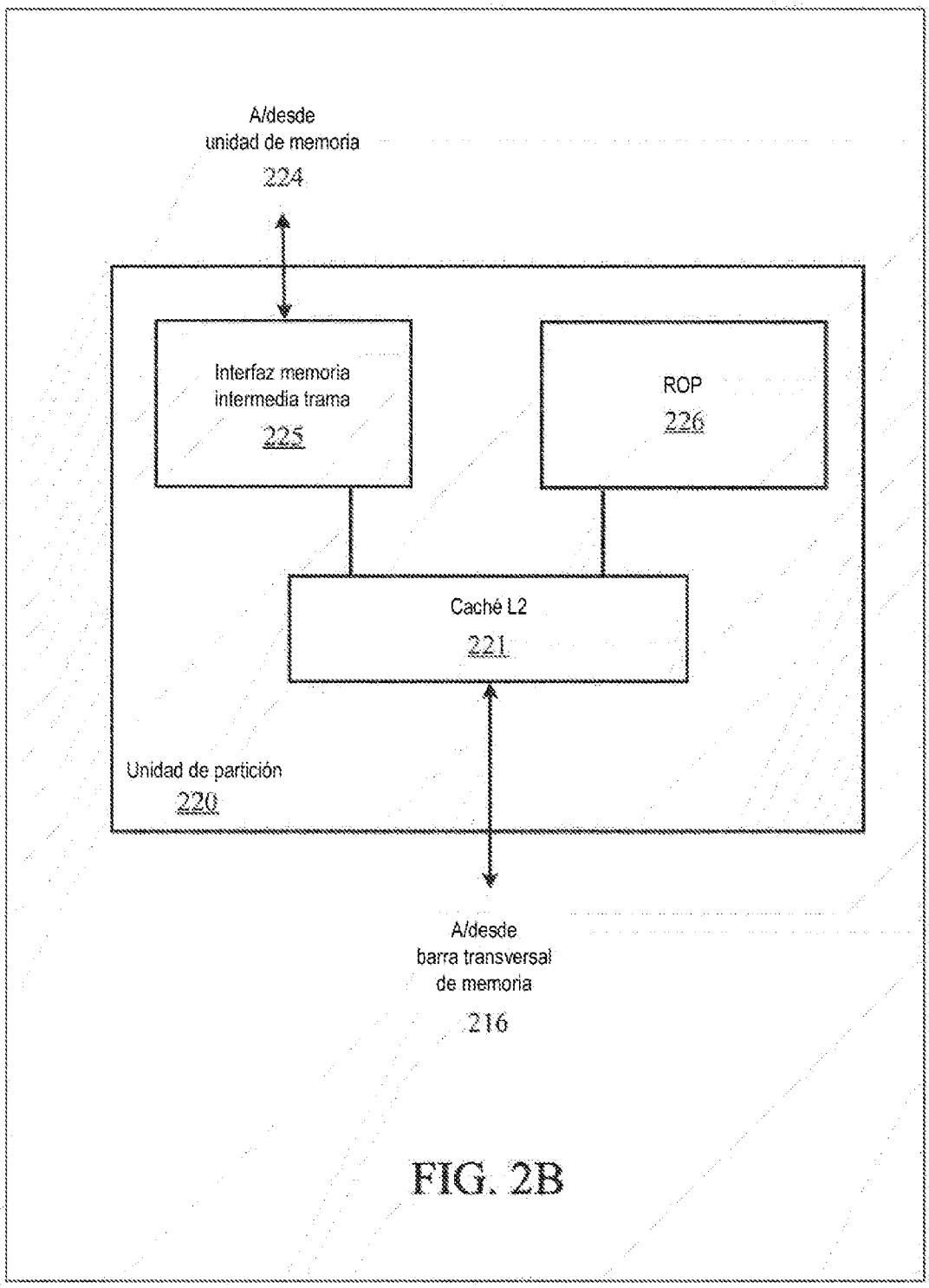

A/desde
unidad de memoria
224
Interfaz memoria
intermedia trama
225
ROP
226
Caché L2
221
Unidad de partición
220
A/desde
barra transversal
de memoria
216

FIG. 2B

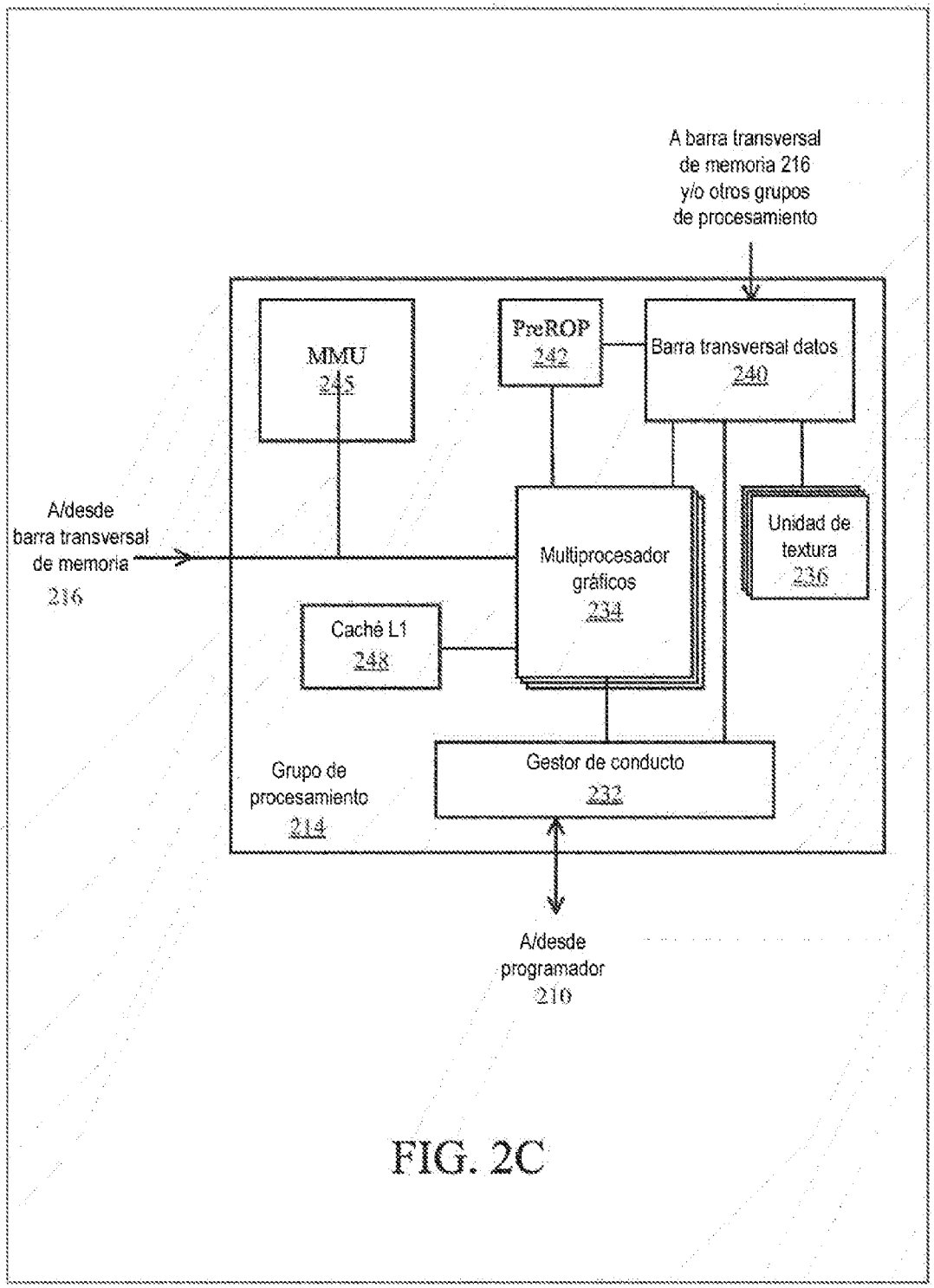

A barra transversal de memoria 216 y/o otros grupos de procesamiento
MMU 245
PreROP 242
Barra transversal datos 240
A/desde barra transversal de memoria 216
Multiprocesador gráficos 234
Unidad de textura 236
Caché L1 248
Gestor de conducto 232
Grupo de procesamiento 214
A/desde programador 210

FIG. 2C

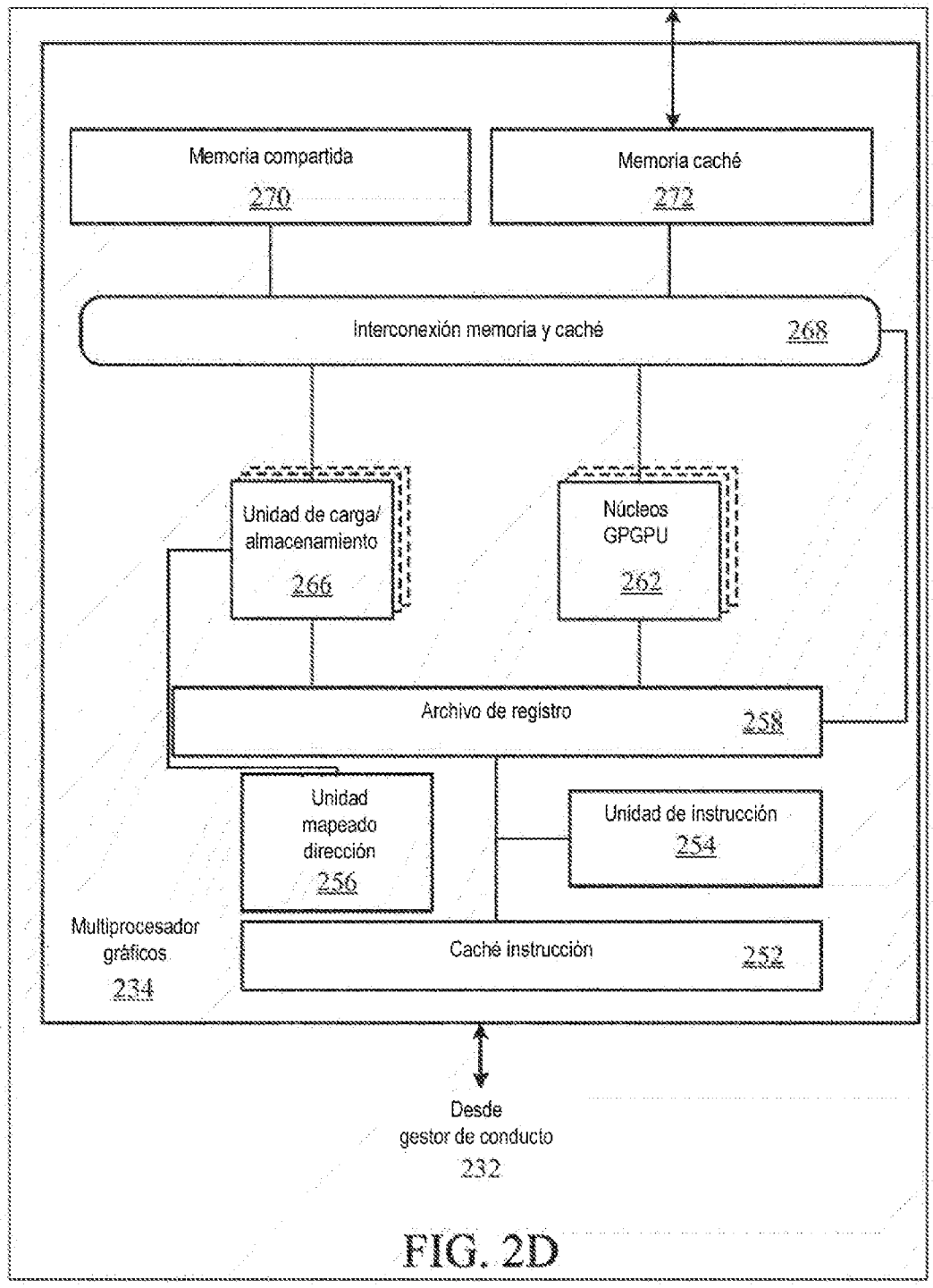

Memoria compartida
270
Memoria caché
272
Interconexión memoria y caché
268
Unidad de carga/
almacenamiento
266
Núcleos
GPGPU
262
Archivo de registro
258
Unidad
mapeado
dirección
256
Unidad de instrucción
254
Multiprocesador
gráficos
234
Caché instrucción
252
Desde
gestor de conducto
232

FIG. 2D

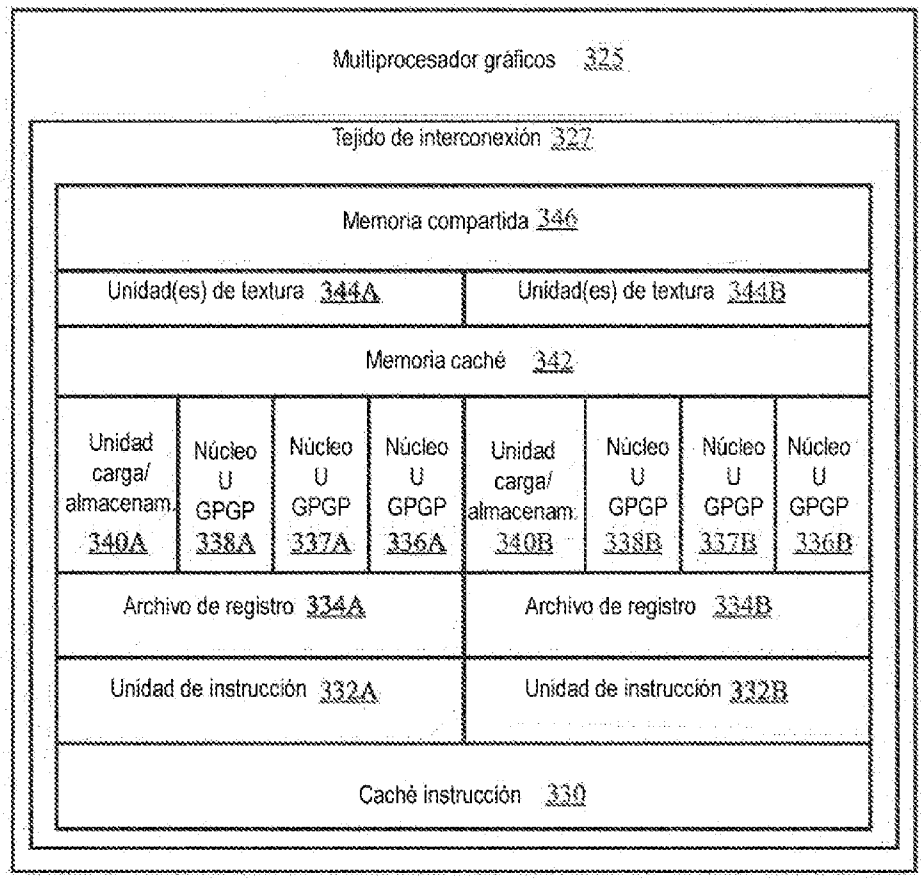

Multiprocesador gráficos 325
Tejido de interconexión 327
Memoria compartida 346
Unidad(es) de textura 344A
Unidad(es) de textura 344B
Memoria caché 342
Unidad carga/ almacenam. 340A
Núcleo U GPGP 338A
Núcleo U GPGP 337A
Núcleo U GPGP 336A
Unidad carga/ almacenam. 340B
Núcleo U GPGP 338B
Núcleo U GPGP 337B
Núcleo U GPGP 336B
Archivo de registro 334A
Archivo de registro 334B
Unidad de instrucción 332A
Unidad de instrucción 332B
Caché instrucción 330

FIG. 3A

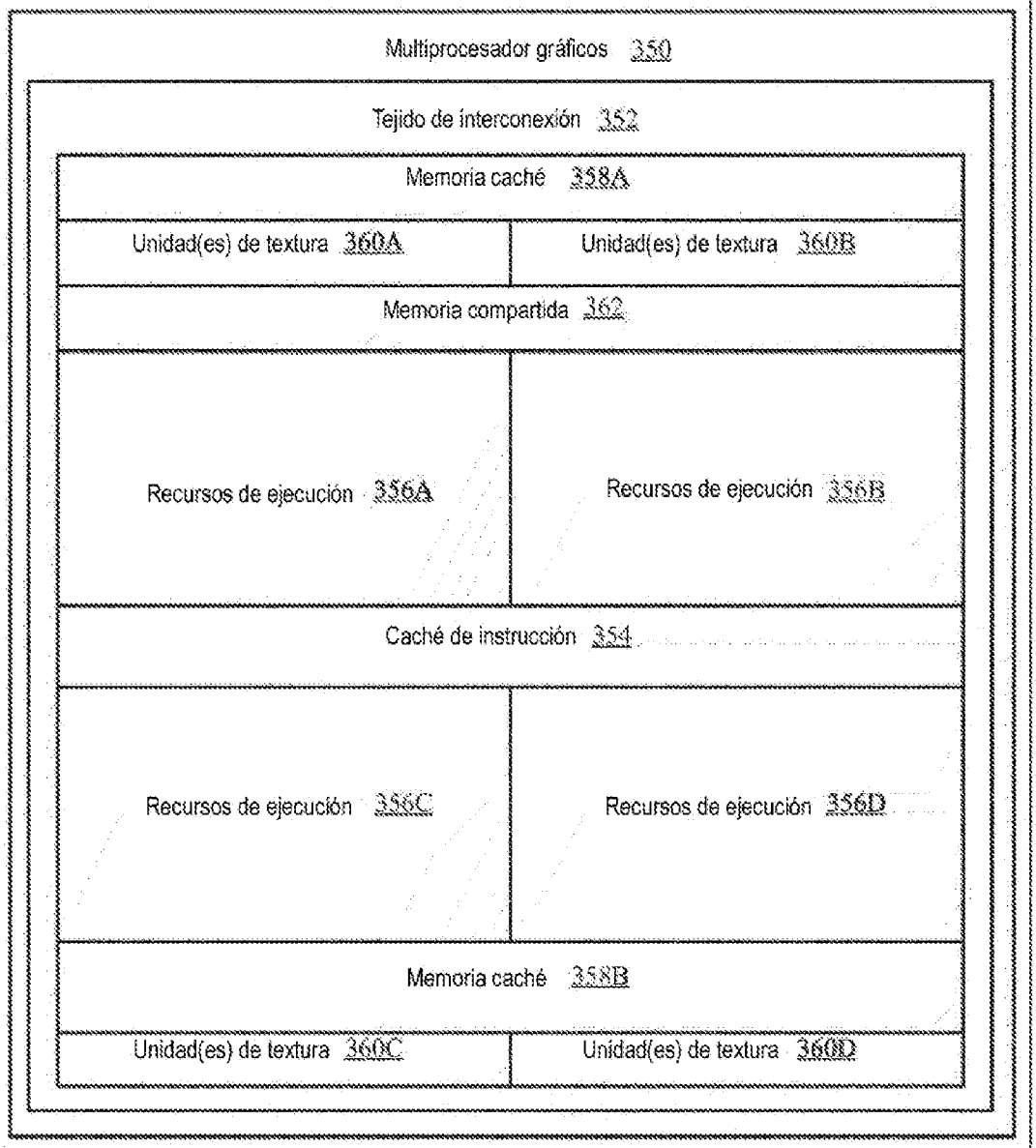

Multiprocesador gráficos 350
Tejido de interconexión 352
Memoria caché 358A
Unidad(es) de textura 360A
Unidad(es) de textura 360B
Memoria compartida 362
Recursos de ejecución 356A
Recursos de ejecución 356B
Caché de instrucción 354
Recursos de ejecución 356C
Recursos de ejecución 356D
Memoria caché 358B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

FIG. 3B

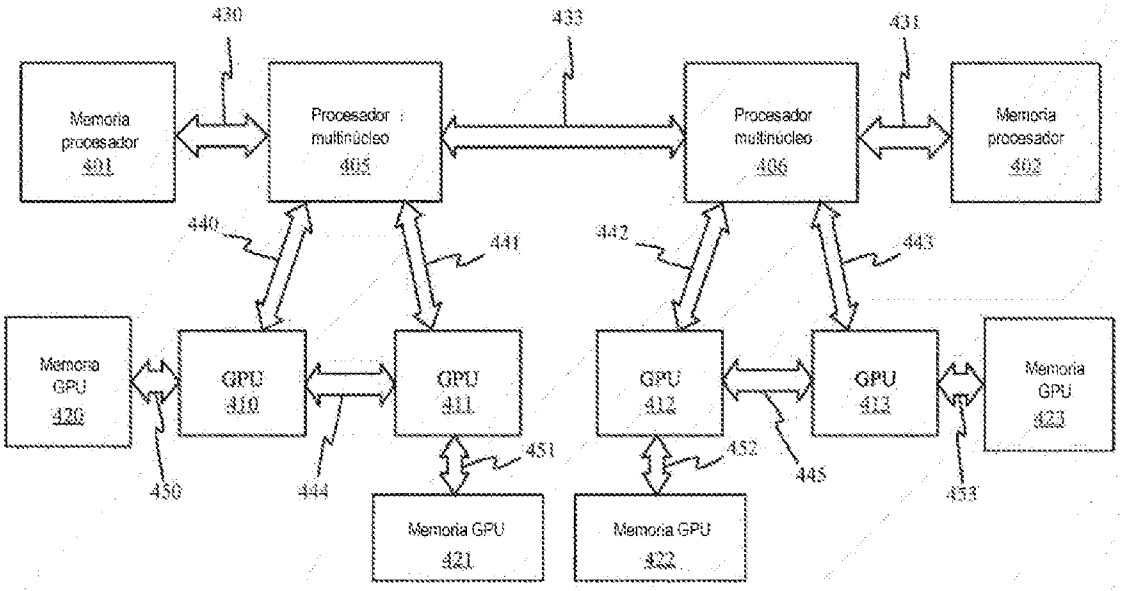

430
433
431
Memoria procesador 401
Procesador multinúcleo 405
Procesador multinúcleo 406
Memoria procesador 402
440
441
442
443
Memoria GPU 420
GPU 410
GPU 411
GPU 412
GPU 413
Memoria GPU 423
450
444
451
452
445
453
Memoria GPU 421
Memoria GPU 422

FIG. 4A

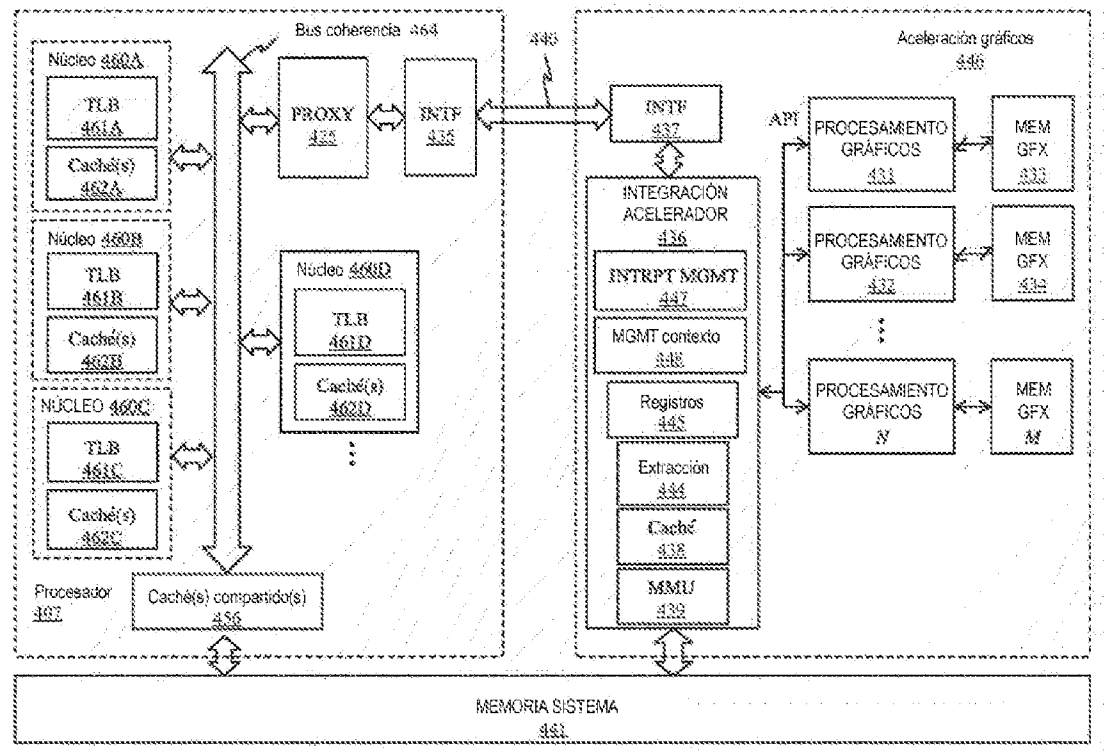

Bus coherencia 464
Núcleo 460A
TLB 461A
Caché(s) 462A
Núcleo 460B
TLB 461B
Caché(s) 462B
NÚCLEO 460C
TLB 461C
Caché(s) 462C
Núcleo 460D
TLB 461D
Caché(s) 462D
PROXY 435
INTF 436
Procesador 407
Caché(s) compartido(s) 456
446
INTF 437
INTEGRACIÓN ACELERADOR 436
INTRPT MGMT 447
MGMT contexto 448
Registros 445
Extracción 444
Caché 438
MMU 439
Aceleración gráficos 446
API
PROCESAMIENTO GRÁFICOS 431
PROCESAMIENTO GRÁFICOS 432
PROCESAMIENTO GRÁFICOS N
MEM GFX 433
MEM GFX 434
MEM GFX M
MEMORIA SISTEMA 461

FIG. 4B

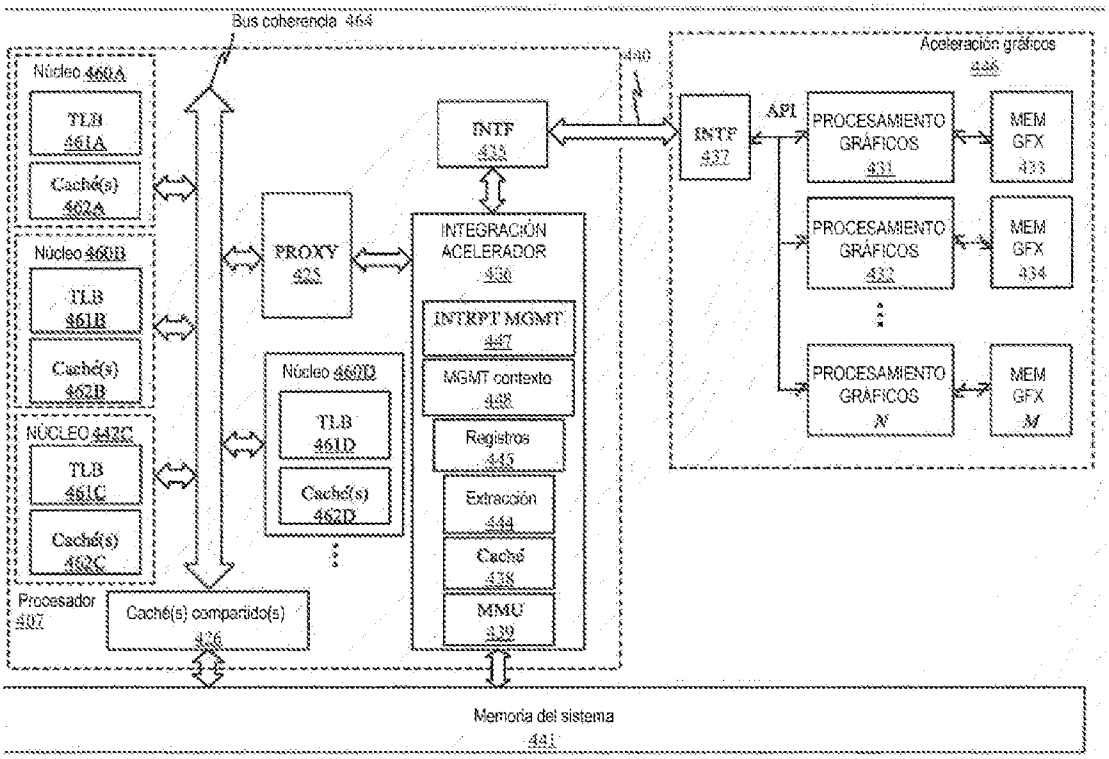

Bus coherencia 464
Núcleo 460A
TLB 461A
Caché(s) 462A
Núcleo 460B
TLB 461B
Caché(s) 462B
NÚCLEO 460C
TLB 461C
Caché(s) 462C
Procesador 407
Caché(s) compartido(s) 426
Núcleo 460D
TLB 461D
Caché(s) 462D
PROXY 422
INTF 433
INTEGRACIÓN ACELERADOR 436
INTRPT MGMT 437
MGMT contexto 438
Registros 445
Extracción 444
Caché 478
MMU 439
440
Aceleración gráficos 446
INTF 437
API
PROCESAMIENTO GRÁFICOS 431
PROCESAMIENTO GRÁFICOS 432
PROCESAMIENTO GRÁFICOS N
MEM GFX 433
MEM GFX 434
MEM GFX M
Memoria del sistema 441

FIG. 4C

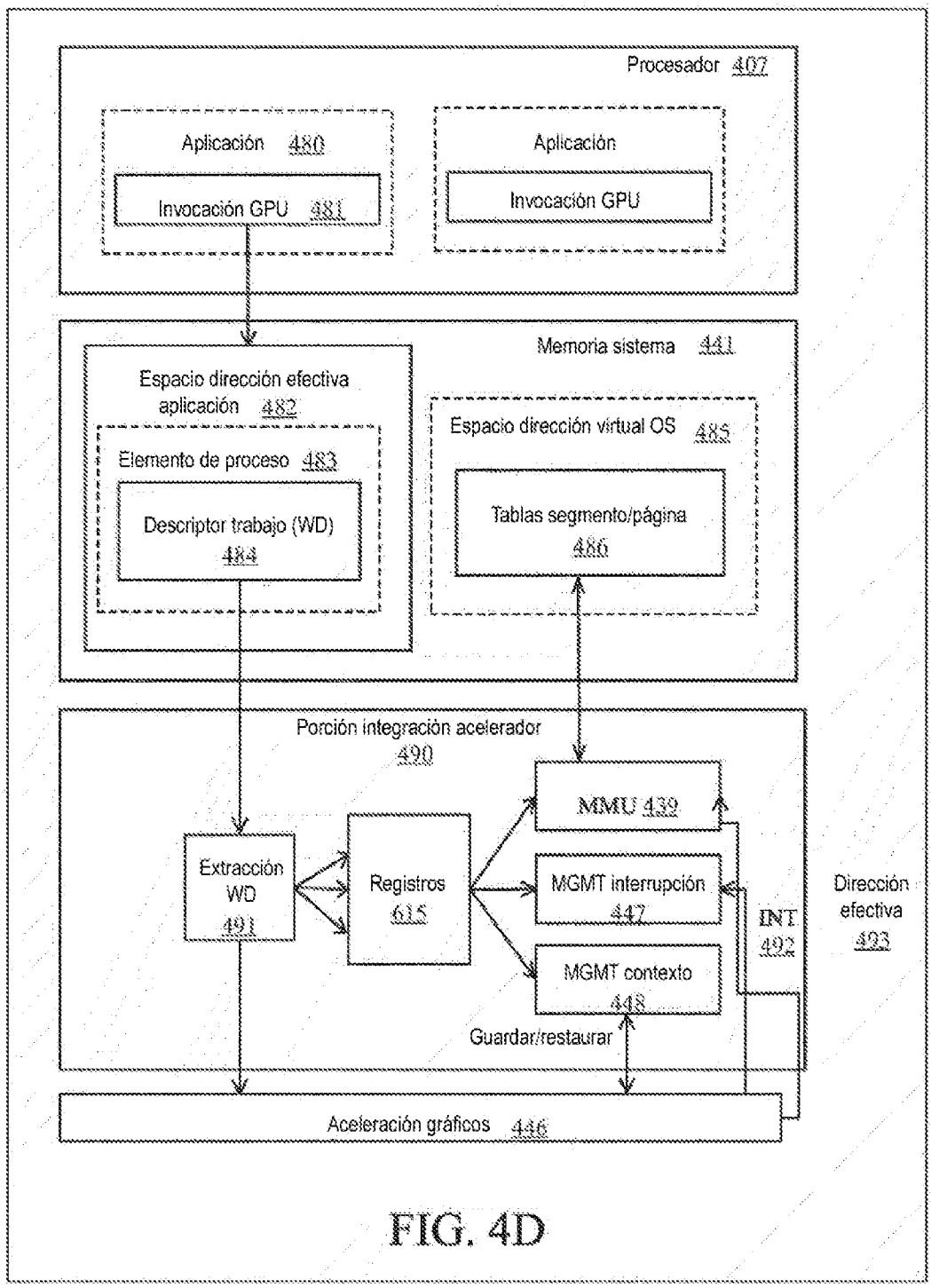

Procesador 407
Aplicación 480
Invocación GPU 481
Aplicación
Invocación GPU
Memoria sistema 441
Espacio dirección efectiva aplicación 482
Elemento de proceso 483
Descriptor trabajo (WD) 484
Espacio dirección virtual OS 485
Tablas segmento/página 486
Porción integración acelerador 490
Extracción WD 491
Registros 615
MMU 439
MGMT interrupción 447
MGMT contexto 448
Guardar/restaurar
INT 492
Dirección efectiva 423
Aceleración gráficos 446

FIG. 4D

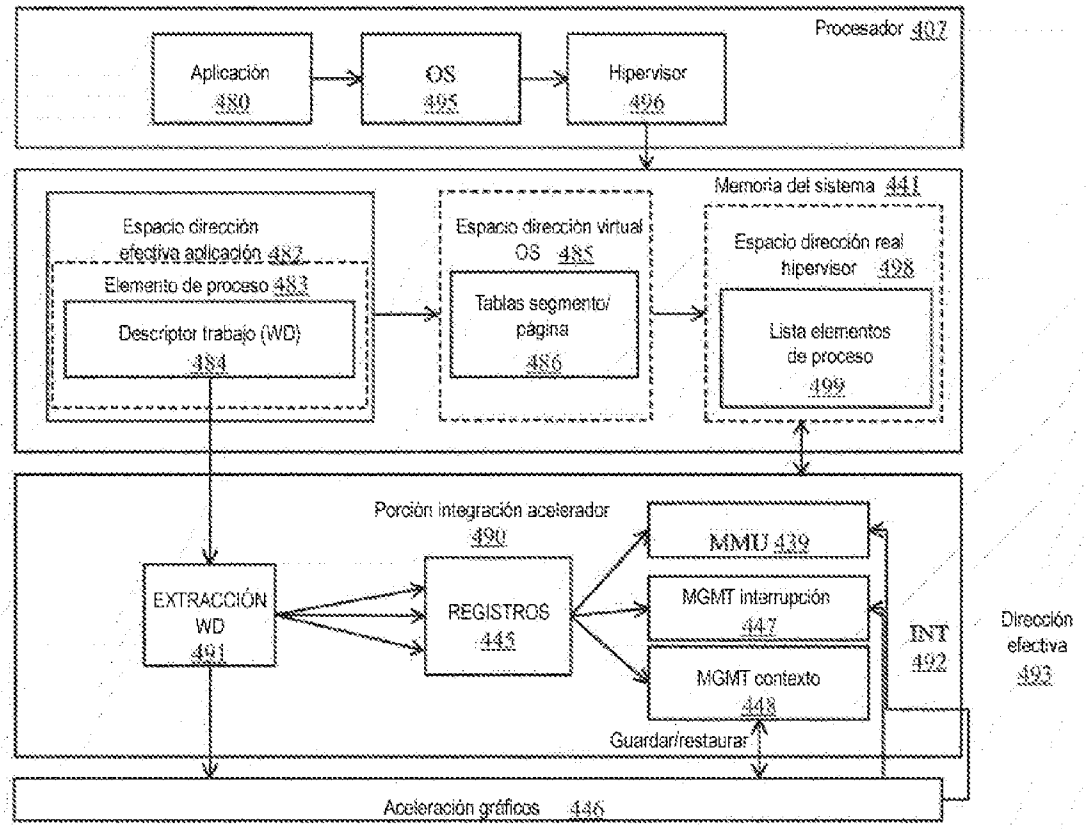

Procesador 407
Aplicación 480
OS 495
Hipervisor 496
Memoria del sistema 441
Espacio dirección efectiva aplicación 482
Elemento de proceso 483
Descriptor trabajo (WD) 484
Espacio dirección virtual OS 485
Tablas segmento/página 486
Espacio dirección real hipervisor 498
Lista elementos de proceso 499
Porción integración acelerador 490
MMU 439
EXTRACCIÓN WD 491
REGISTROS 445
MGMT interrupción 447
MGMT contexto 448
INT 492
Dirección efectiva 493
Guardar/restaurar
Aceleración gráficos 436

FIG. 4E

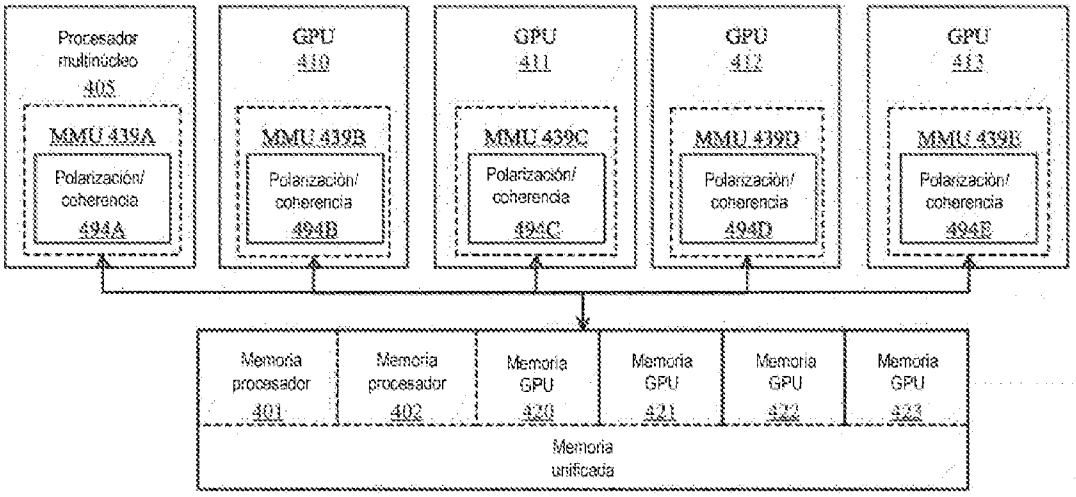

Procesador multinúcleo
405
MMU 439A
Polarización/ coherencia
494A
GPU
410
MMU 439B
Polarización/ coherencia
494B
GPU
411
MMU 439C
Polarización/ coherencia
494C
GPU
412
MMU 439D
Polarización/ coherencia
494D
GPU
413
MMU 439E
Polarización/ coherencia
494E
Memoria procesador
401
Memoria procesador
402
Memoria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

FIG. 4F

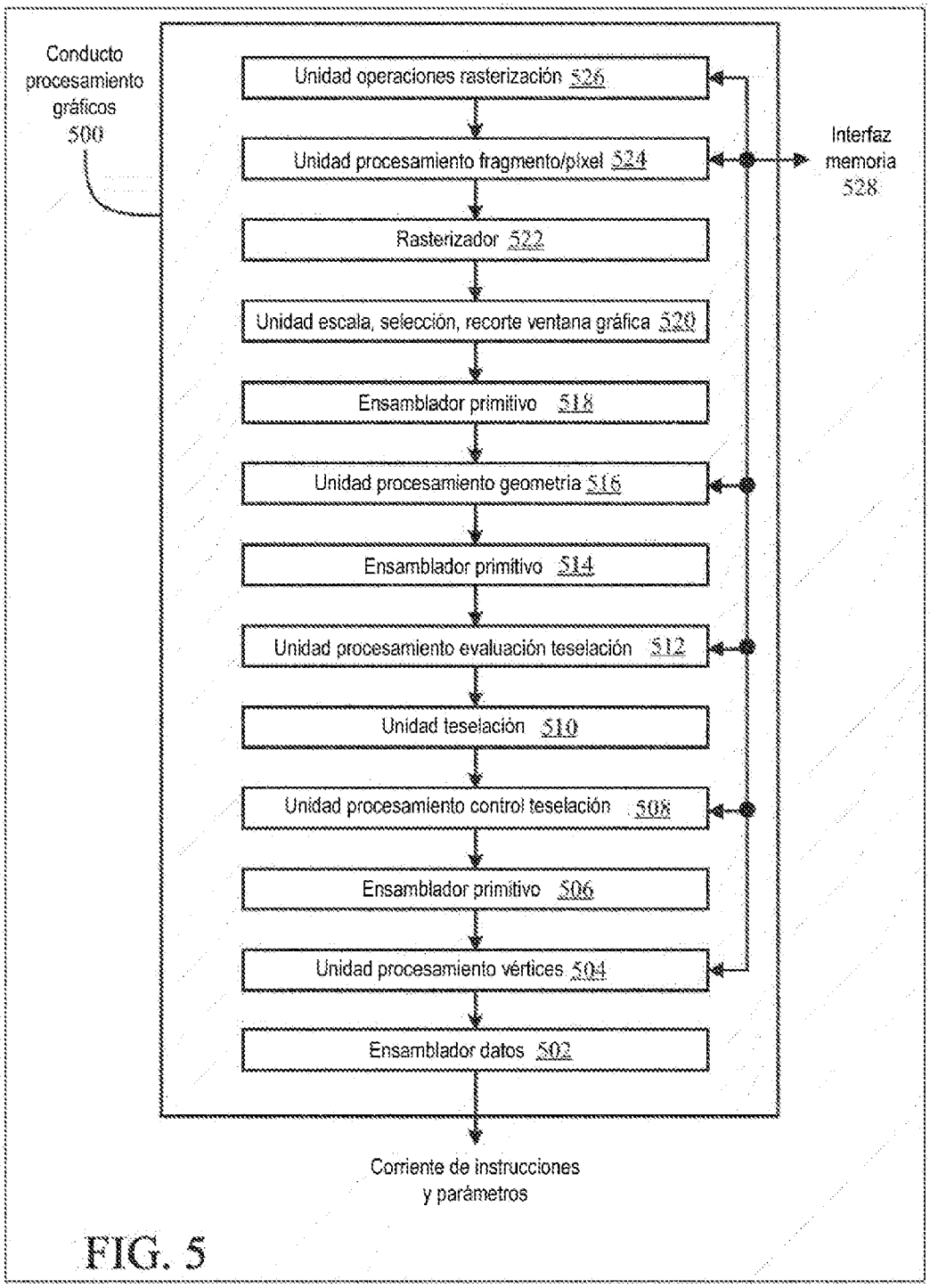

Conducto procesamiento gráficos 500
Unidad operaciones rasterización 526
Unidad procesamiento fragmento/píxel 524
Rasterizador 522
Unidad escala, selección, recorte ventana gráfica 520
Ensamblador primitivo 518
Unidad procesamiento geometría 516
Ensamblador primitivo 514
Unidad procesamiento evaluación teselación 512
Unidad teselación 510
Unidad procesamiento control teselación 508
Ensamblador primitivo 506
Unidad procesamiento vértices 504
Ensamblador datos 502
Interfaz memoria 528
Corriente de instrucciones y parámetros

FIG. 5

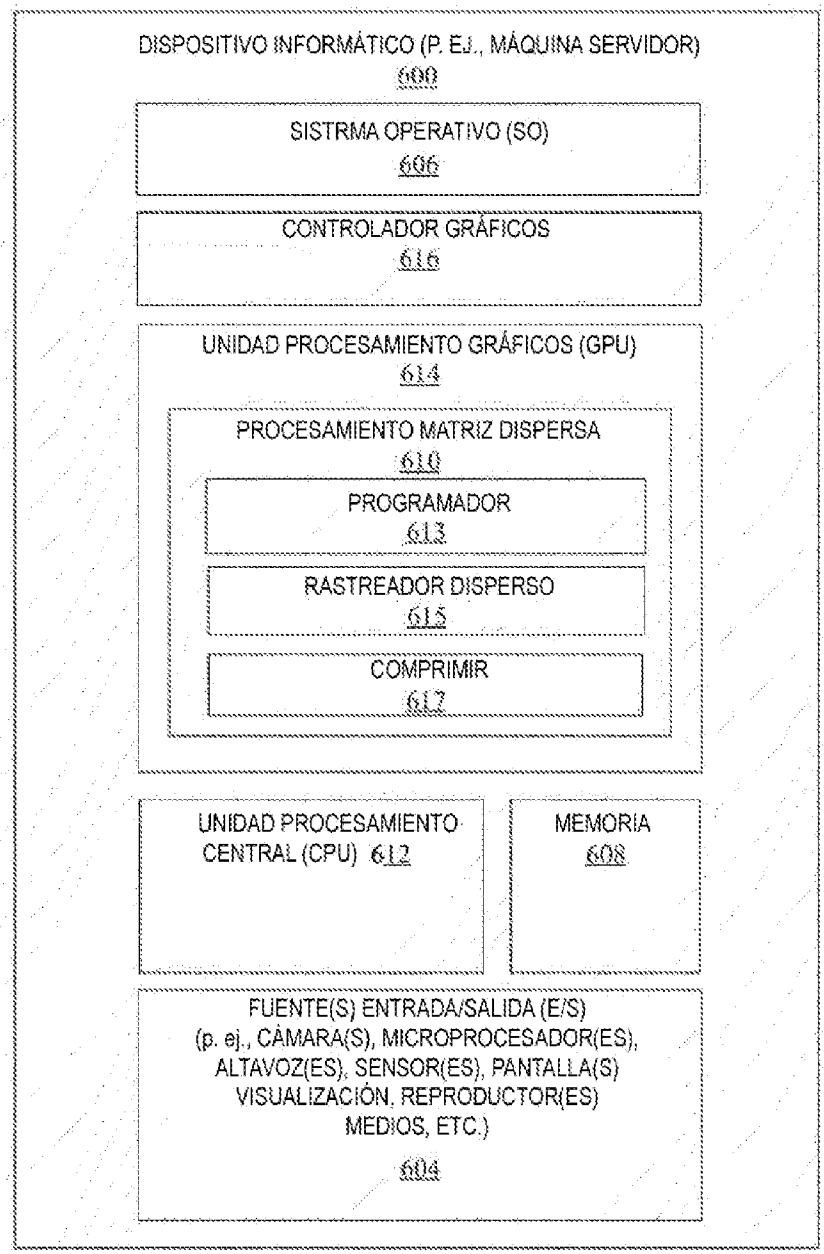

DISPOSITIVO INFORMÁTICO (P. EJ., MÁQUINA SERVIDOR)
600
SISTRMA OPERATIVO (SO)
606
CONTROLADOR GRÁFICOS
616
UNIDAD PROCESAMIENTO GRÁFICOS (GPU)
614
PROCESAMIENTO MATRIZ DISPERSA
610
PROGRAMADOR
613
RASTREADOR DISPERSO
615
COMPRIMIR
617
UNIDAD PROCESAMIENTO CENTRAL (CPU) 612
MEMORIA
608
FUENTE(S) ENTRADA/SALIDA (E/S)
(p. ej., CÁMARA(S), MICROPROCESADOR(ES), ALTAVOZ(ES), SENSOR(ES), PANTALLA(S) VISUALIZACIÓN, REPRODUCTOR(ES) MEDIOS, ETC.)
604

FIG. 6

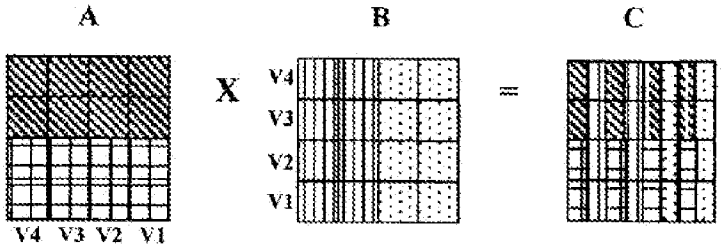


FIG. 7A

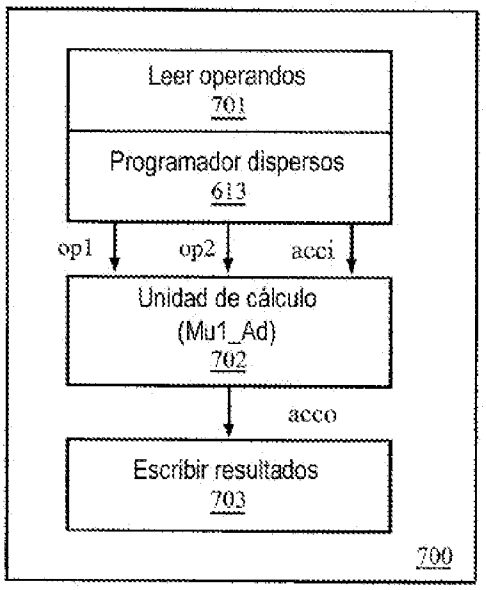


FIG. 7B

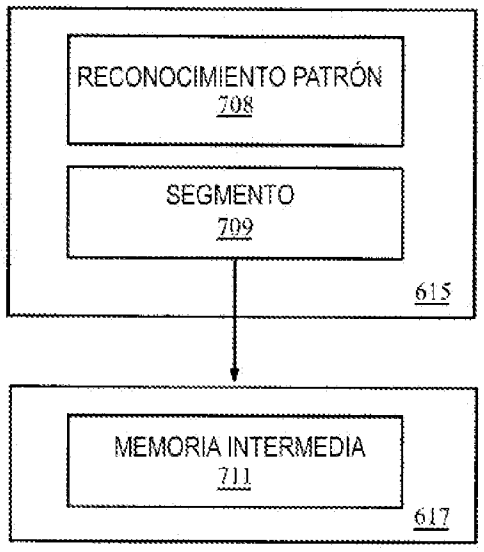


FIG. 7C

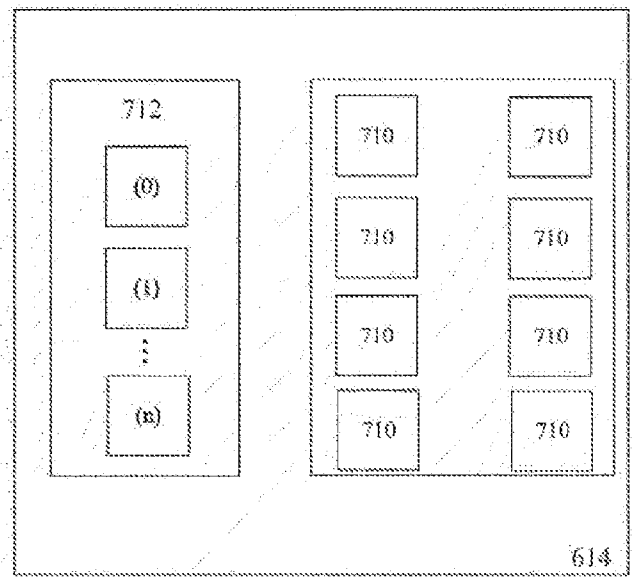

712
(0)
(1)
(n)
710
710
710
710
710
710
710
710
614

FIG. 7D

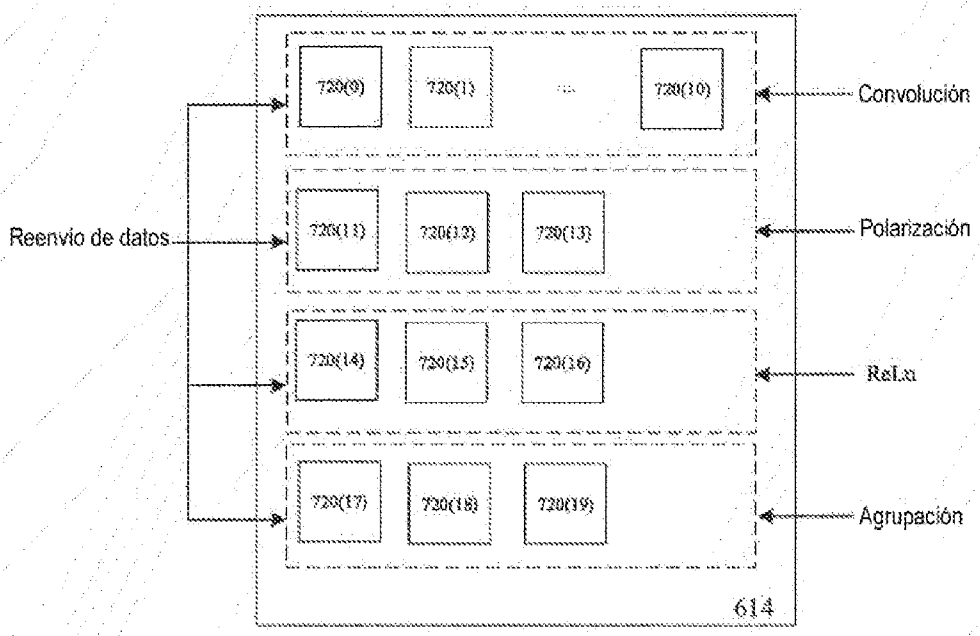

720(0)
720(1)
720(10)
720(11)
720(12)
720(13)
720(14)
720(15)
720(16)
720(17)
720(18)
720(19)
Reenvío de datos
Convolución
Polarización
ReLu
Agrupación
614

FIG. 7E

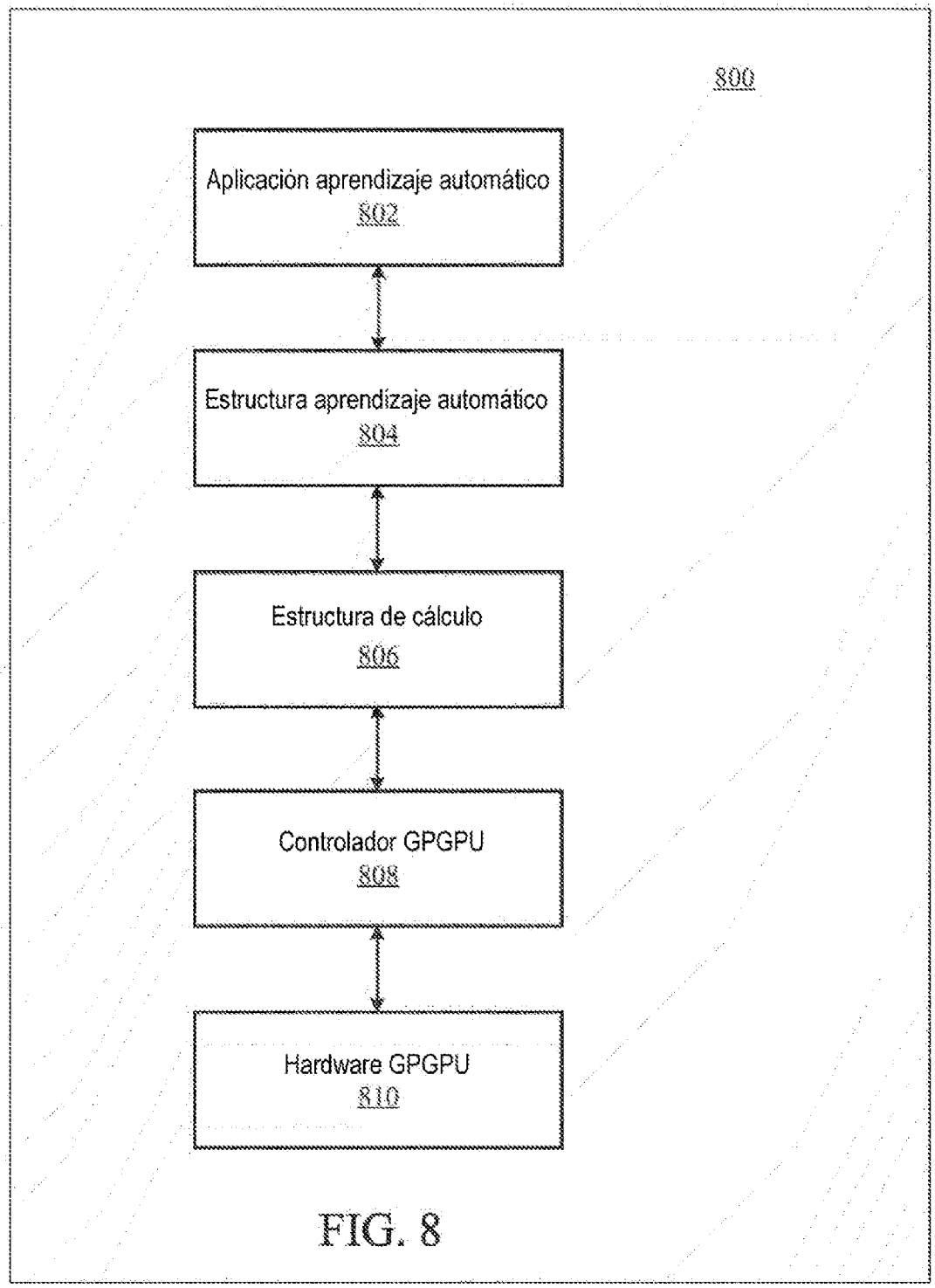

800
Aplicación aprendizaje automático
802
Estructura aprendizaje automático
804
Estructura de cálculo
806
Controlador GPGPU
808
Hardware GPGPU
810

FIG. 8

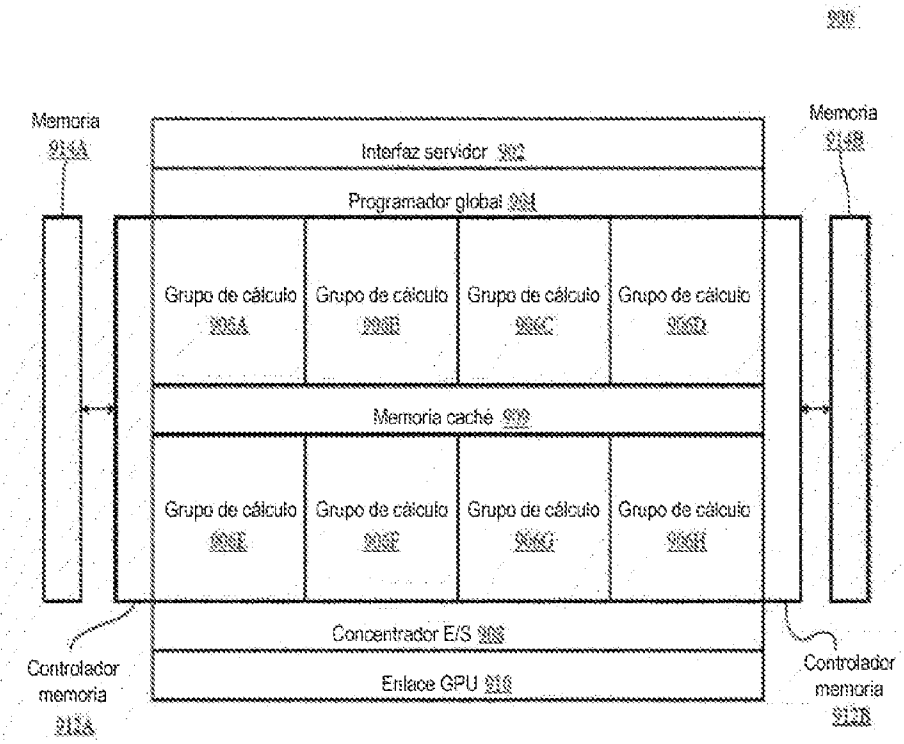

Memoria
Interfaz servidor
Memoria
Programador global
Grupo de cálculo
Grupo de cálculo
Grupo de cálculo
Grupo de cálculo
Memoria caché
Grupo de cálculo
Grupo de cálculo
Grupo de cálculo
Grupo de cálculo
Concentrador E/S
Enlace GPU
Controlador memoria
Controlador memoria

FIG. 9

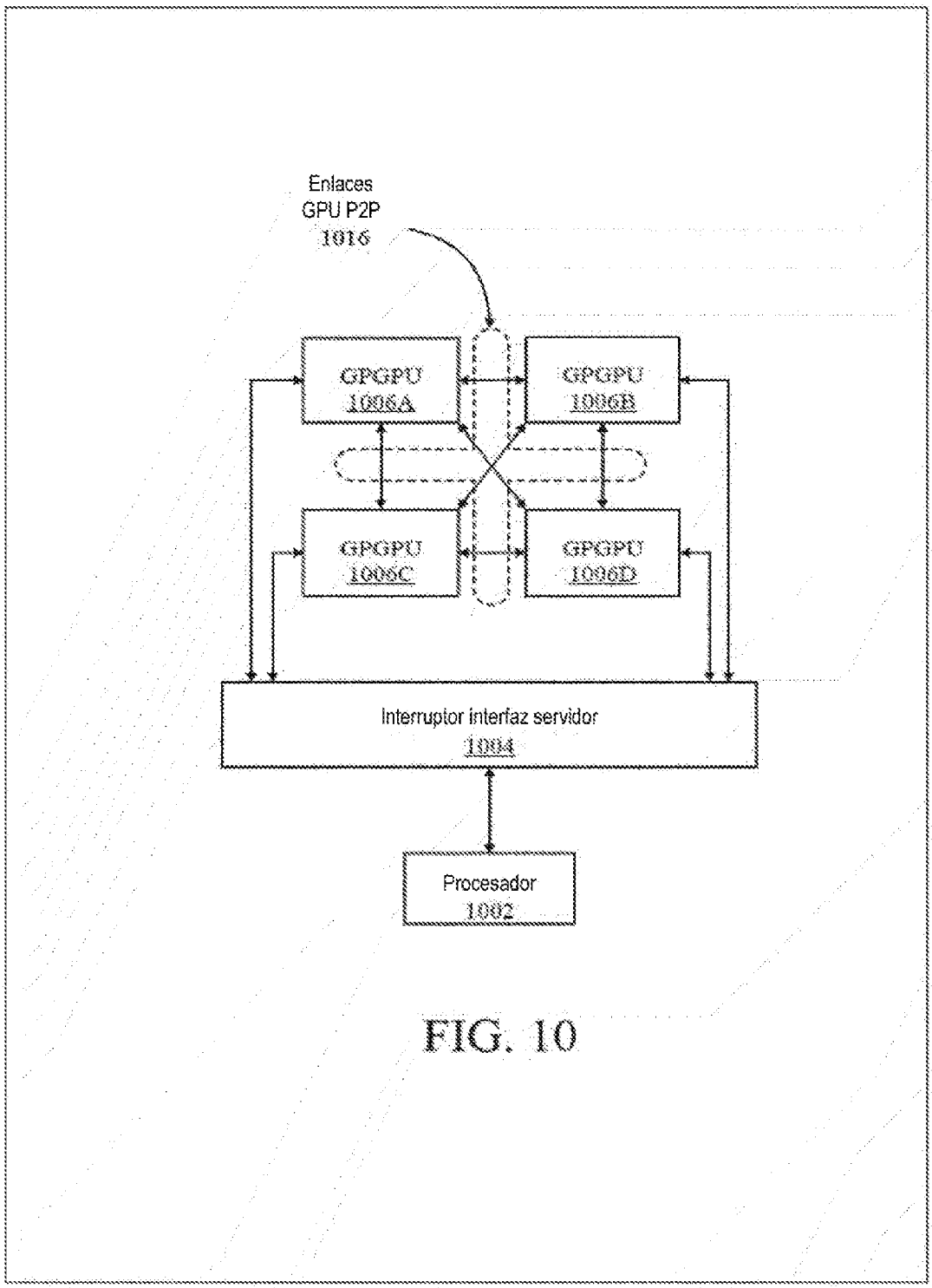

Enlaces
GPU P2P
1016
GPGPU
1006A
GPGPU
1006B
GPGPU
1006C
GPGPU
1006D
Interruptor interfaz servidor
1004
Procesador
1002

FIG. 10

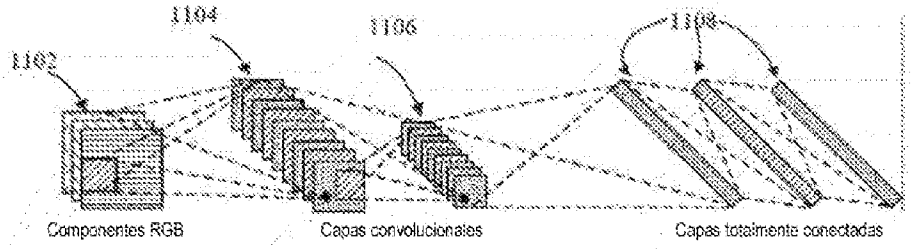


FIG. 11A

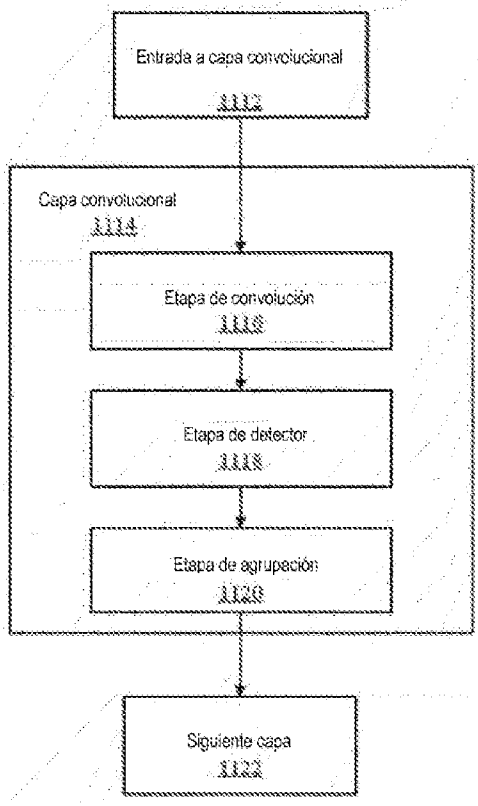


FIG. 11B

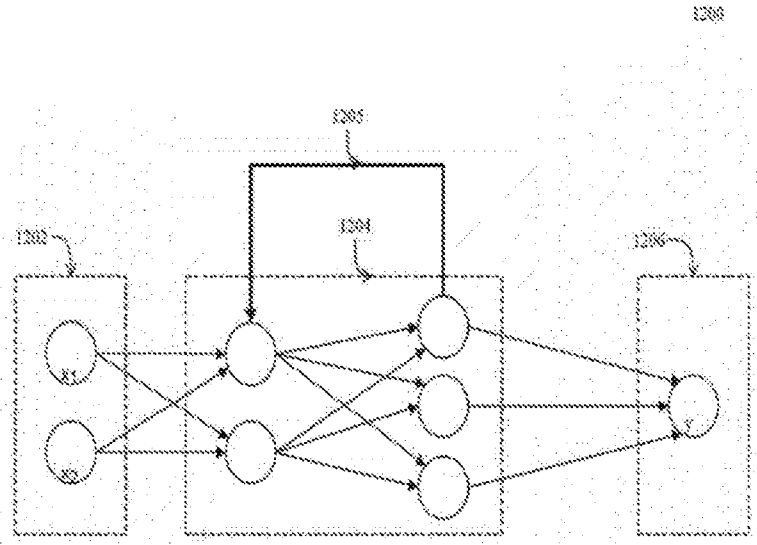


FIG. 12

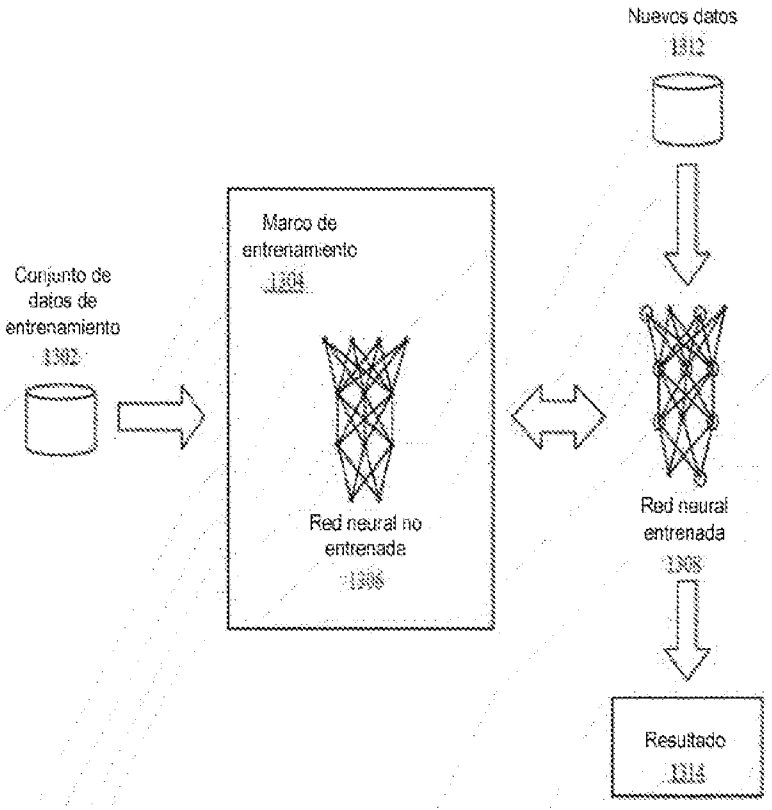

Nuevos datos
1312
Marco de entrenamiento
1304
Conjunto de datos de entrenamiento
1302
Red neural no entrenada
1306
Red neural entrenada
1308
Resultado
1314

FIG. 13

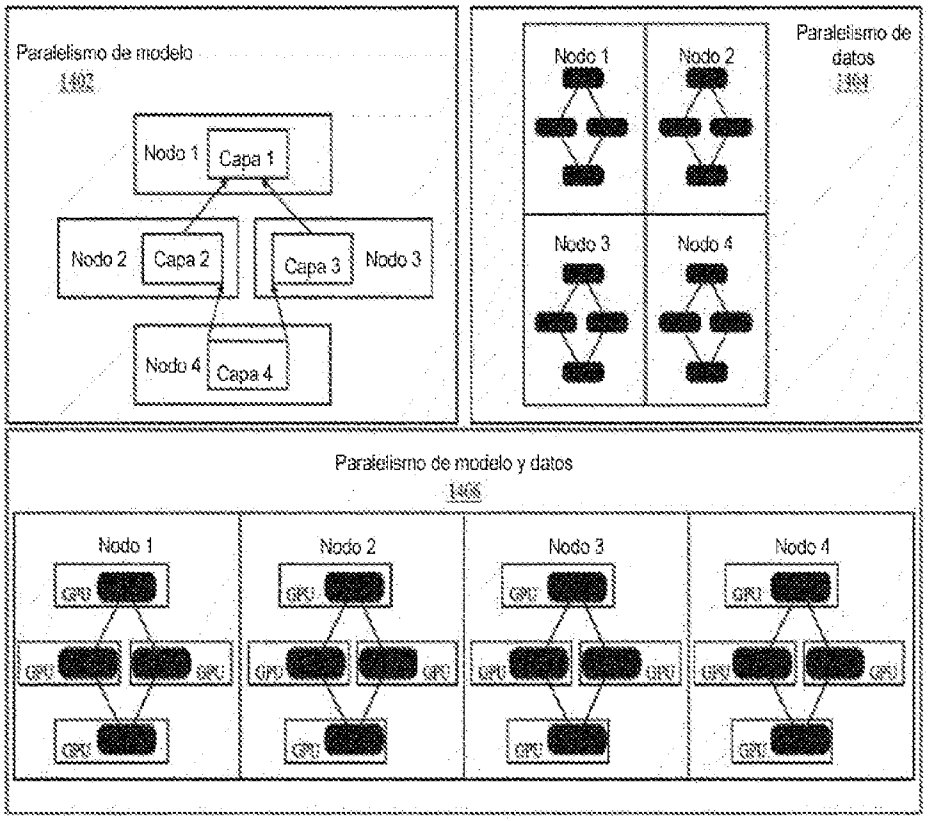

Paralelismo de modelo
1402
Nodo 1
Capa 1
Nodo 2
Capa 2
Capa 3
Nodo 3
Nodo 4
Capa 4
Nodo 1
Nodo 2
Nodo 3
Nodo 4
Paralelismo de datos
1404
Paralelismo de modelo y datos
1406
Nodo 1
Nodo 2
Nodo 3
Nodo 4
GPU

FIG. 14

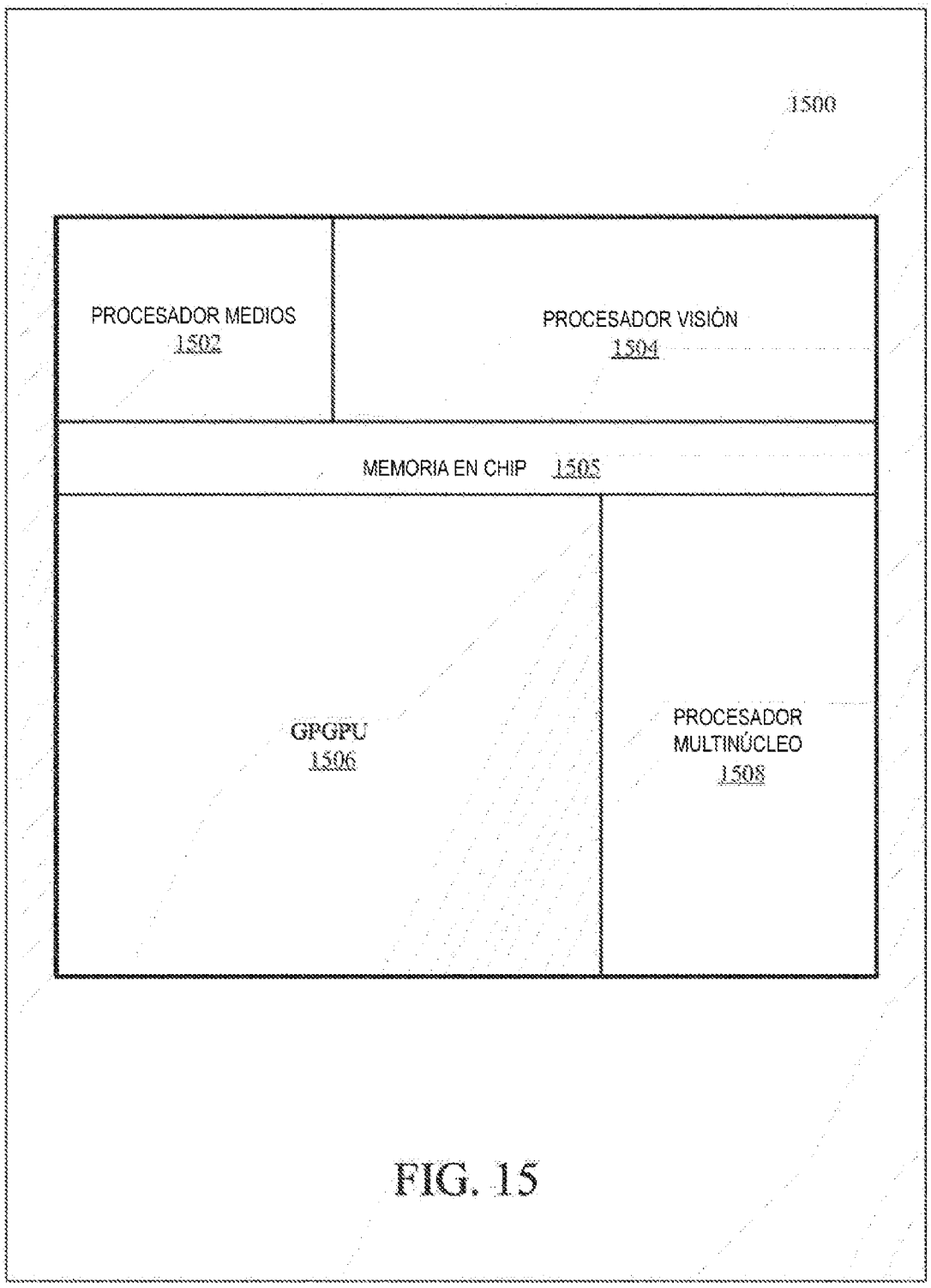

1500
PROCESADOR MEDIOS
1502
PROCESADOR VISIÓN
1504
MEMORIA EN CHIP 1505
GPGPU
1506
PROCESADOR
MULTINÚCLEO
1508

FIG. 15

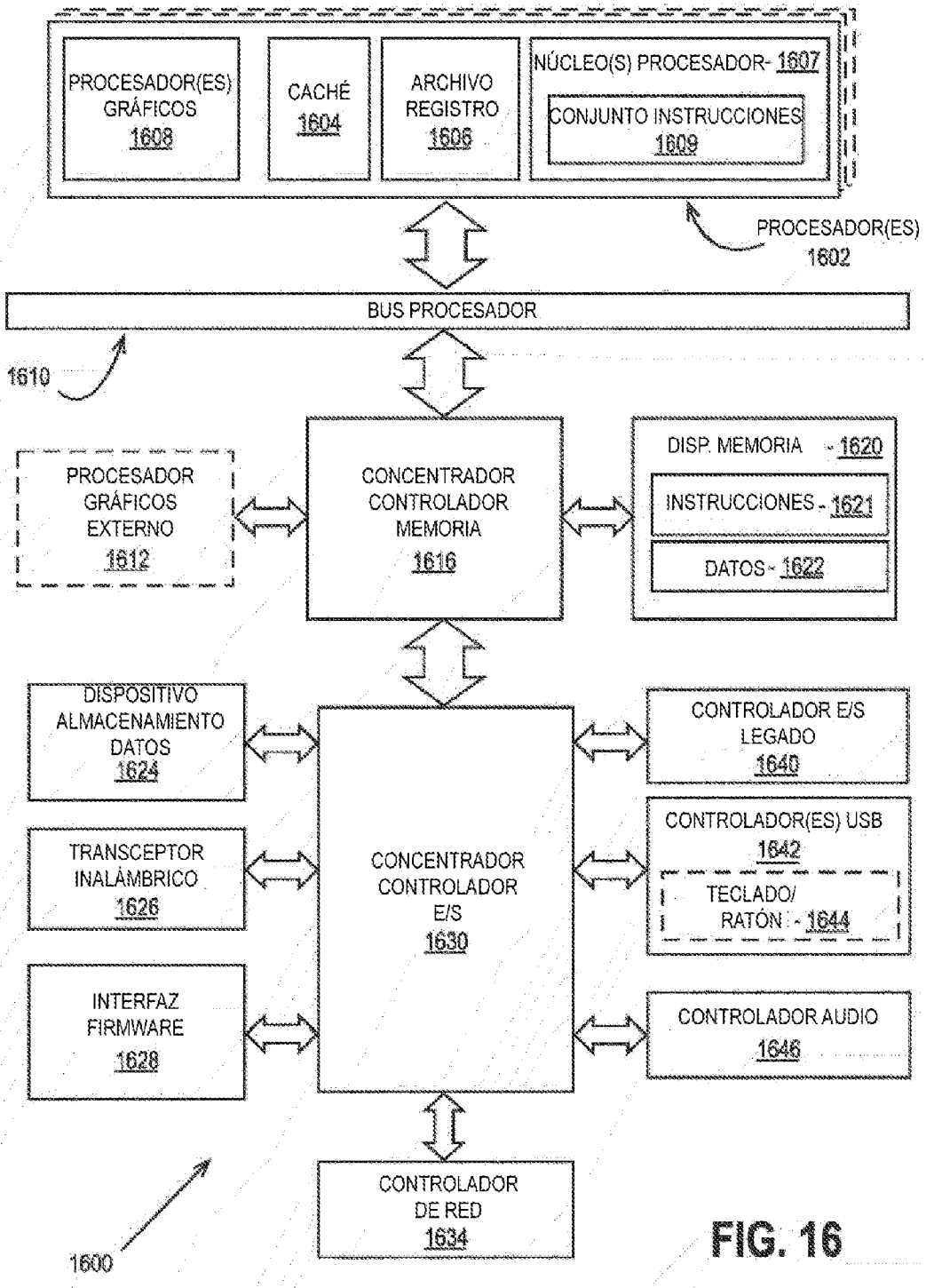

PROCESADOR(ES) GRÁFICOS 1608
CACHÉ 1604
ARCHIVO REGISTRO 1606
NÚCLEO(S) PROCESADOR- 1607
CONJUNTO INSTRUCCIONES 1609
PROCESADOR(ES) 1602
BUS PROCESADOR
1610
PROCESADOR GRÁFICOS EXTERNO 1612
CONCENTRADOR CONTROLADOR MEMORIA 1616
DISP. MEMORIA - 1620
INSTRUCCIONES - 1621
DATOS - 1622
DISPOSITIVO ALMACENAMIENTO DATOS 1624
TRANSCEPTOR INALÁMBRICO 1626
INTERFAZ FIRMWARE 1628
CONCENTRADOR CONTROLADOR E/S 1630
CONTROLADOR E/S LEGADO 1640
CONTROLADOR(ES) USB 1642
TECLADO/ RATÓN - 1644
CONTROLADOR AUDIO 1646
CONTROLADOR DE RED 1634
1600

FIG. 16

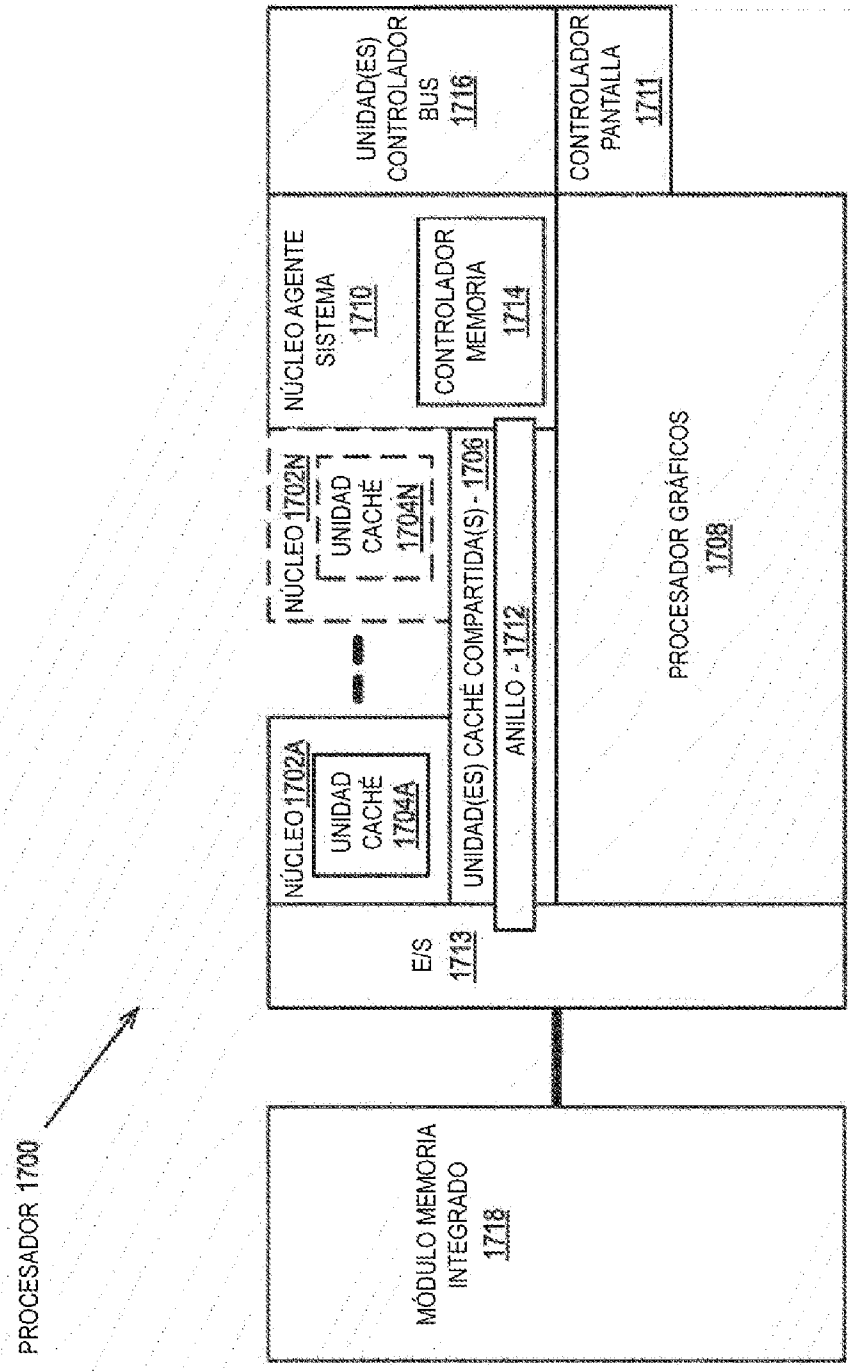

PROCESADOR 1700
MÓDULO MEMORIA INTEGRADO 1718
E/S 1713
NÚCLEO 1702A
UNIDAD CACHÉ 1704A
NÚCLEO 1702N
UNIDAD CACHÉ 1704N
UNIDAD(ES) CACHÉ COMPARTIDA(S) - 1706
ANILLO - 1712
NÚCLEO AGENTE SISTEMA 1710
CONTROLADOR MEMORIA 1714
PROCESADOR GRÁFICOS 1708
UNIDAD(ES) CONTROLADOR BUS 1716
CONTROLADOR PANTALLA 1711

FIG. 17

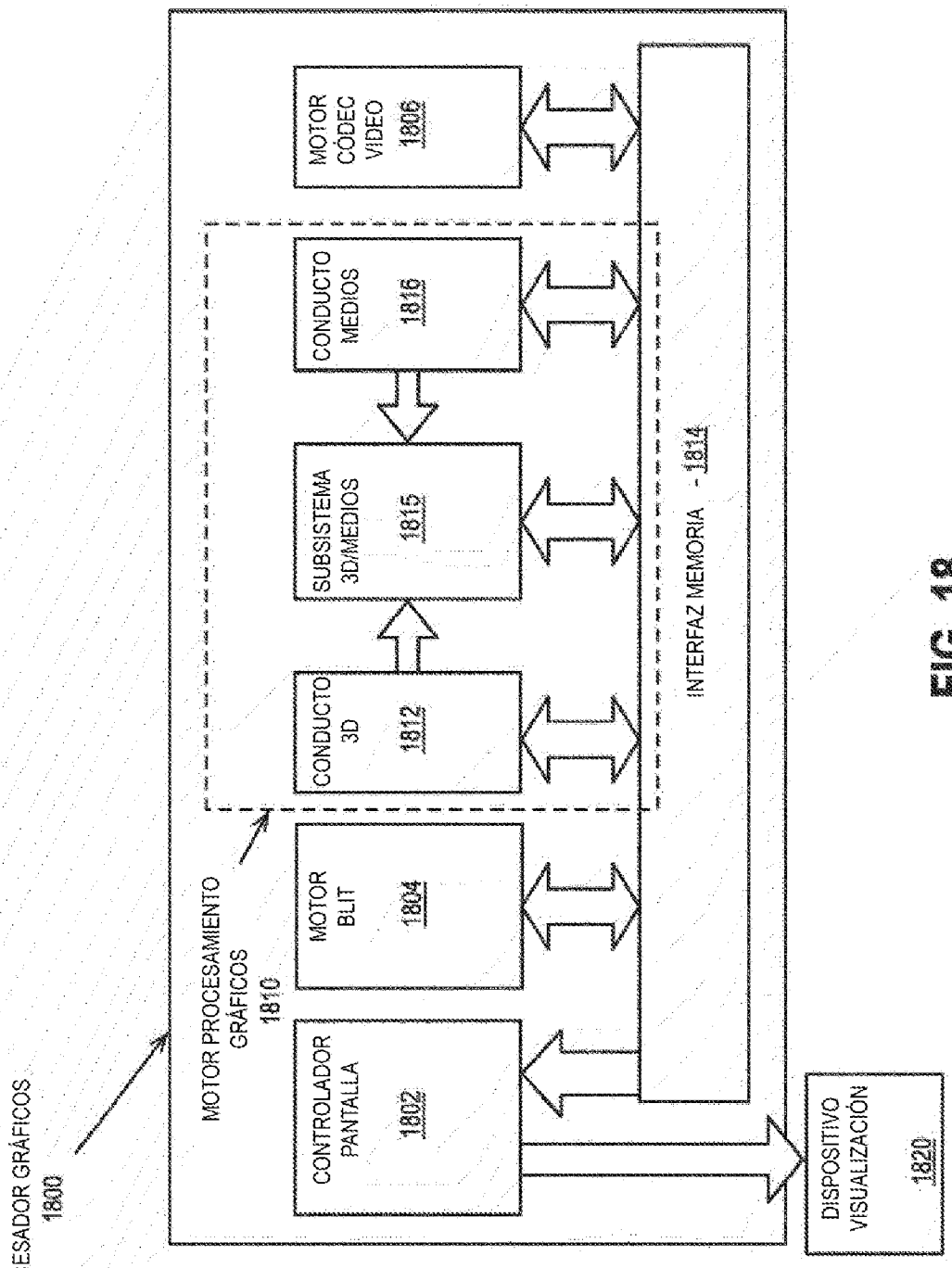

PROCESADOR GRÁFICOS
1800
MOTOR PROCESAMIENTO GRÁFICOS
1810
CONTROLADOR PANTALLA
1802
MOTOR BLIT
1804
CONDUCTO 3D
1812
SUBSISTEMA 3D/MEDIOS
1815
CONDUCTO MEDIOS
1816
MOTOR CÓDEC VIDEO
1806
INTERFAZ MEMORIA - 1814
DISPOSITIVO VISUALIZACIÓN
1820

FIG. 18

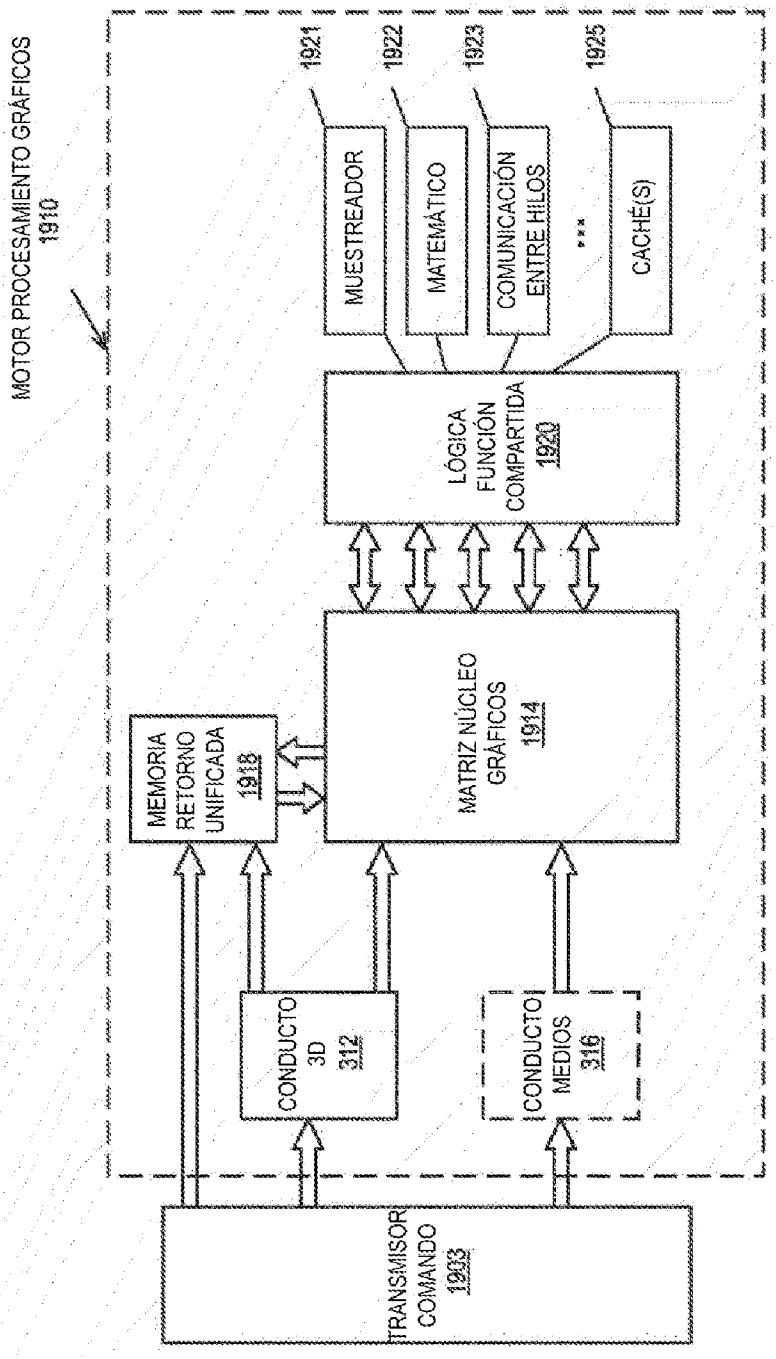

MOTOR PROCESAMIENTO GRÁFICOS
1910
1921
1922
1923
1925
MUESTREADOR
MATEMÁTICO
COMUNICACIÓN ENTRE HILOS
***
CACHÉ(S)
LÓGICA FUNCIÓN COMPARTIDA
1920
MATRIZ NÚCLEO GRÁFICOS
1914
MEMORIA RETORNO UNIFICADA
1918
CONDUCTO 3D
312
CONDUCTO MEDIOS
316
TRANSMISOR COMANDO
1903

FIG. 19

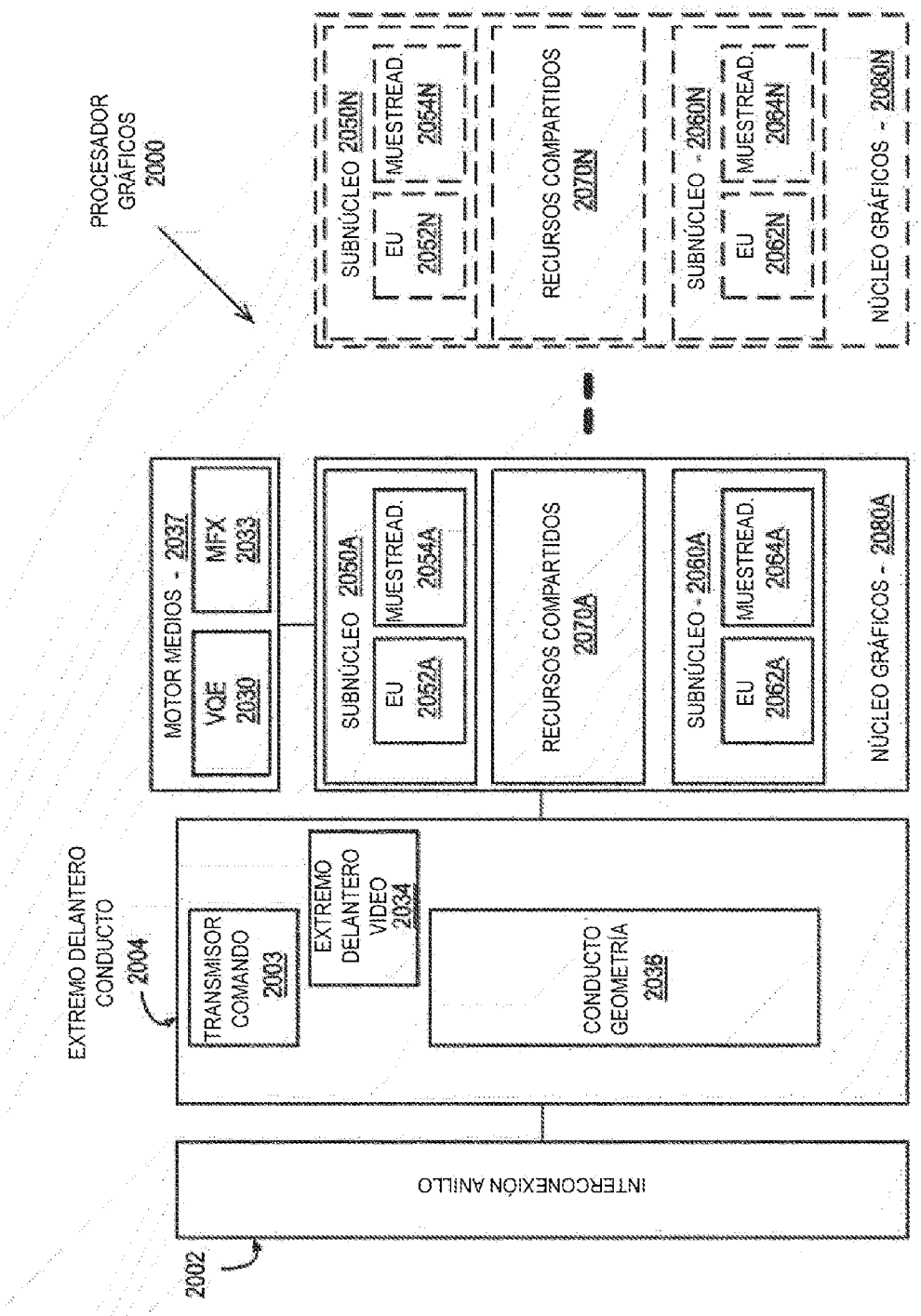

PROCESADOR GRÁFICOS 2000
EXTREMO DELANTERO CONDUCTO 2004
2002
INTERCONEXIÓN ANILLO
TRANSMISOR COMANDO 2003
EXTREMO DELANTERO VIDEO 2034
CONDUCTO GEOMETRÍA 2036
MOTOR MEDIOS - 2037
VQE 2030
MFX 2033
SUBNÚCLEO 2050A
EU 2052A
MUESTREAD. 2054A
RECURSOS COMPARTIDOS 2070A
SUBNÚCLEO - 2060A
EU 2062A
MUESTREAD. 2064A
NÚCLEO GRÁFICOS - 2080A
SUBNÚCLEO 2050N
EU 2052N
MUESTREAD. 2054N
RECURSOS COMPARTIDOS 2070N
SUBNÚCLEO - 2060N
EU 2062N
MUESTREAD. 2064N
NÚCLEO GRÁFICOS - 2080N

FIG. 20

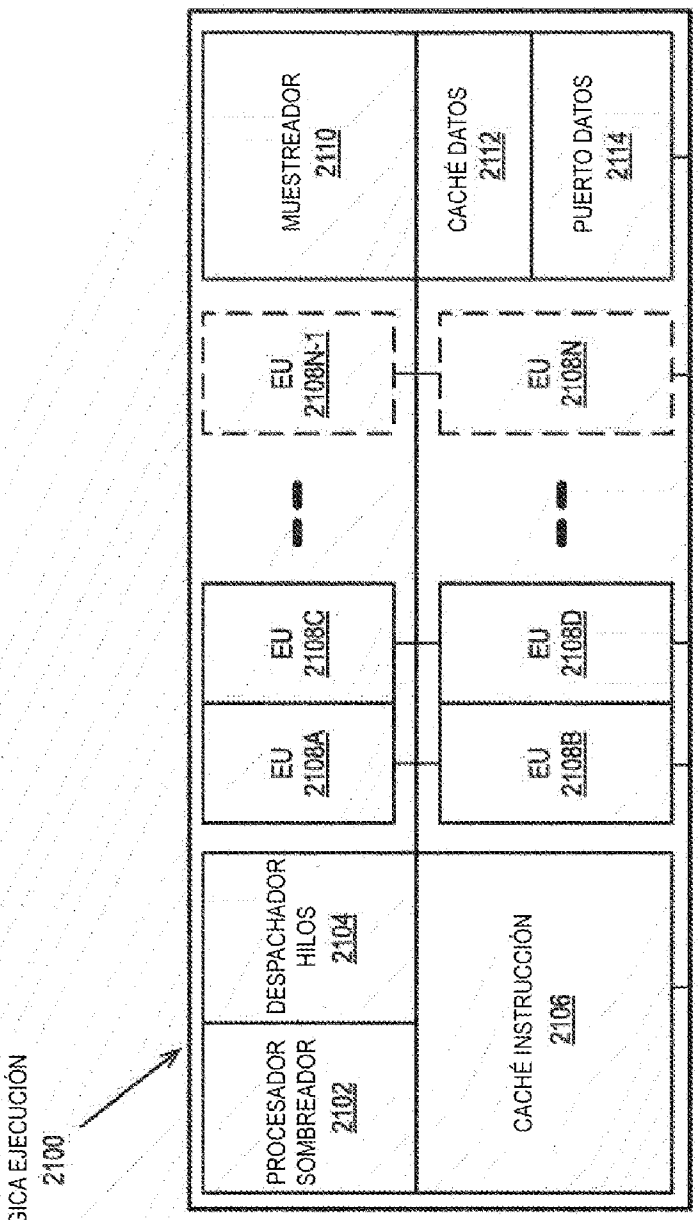

LÓGICA EJECUCIÓN
2100
PROCESADOR SOMBREADOR
2102
DESPACHADOR HILOS
2104
CACHÉ INSTRUCCIÓN
2106
EU
2108A
EU
2108B
EU
2108C
EU
2108D
EU
2108N-1
EU
2108N
MUESTREADOR
2110
CACHÉ DATOS
2112
PUERTO DATOS
2114

FIG. 21

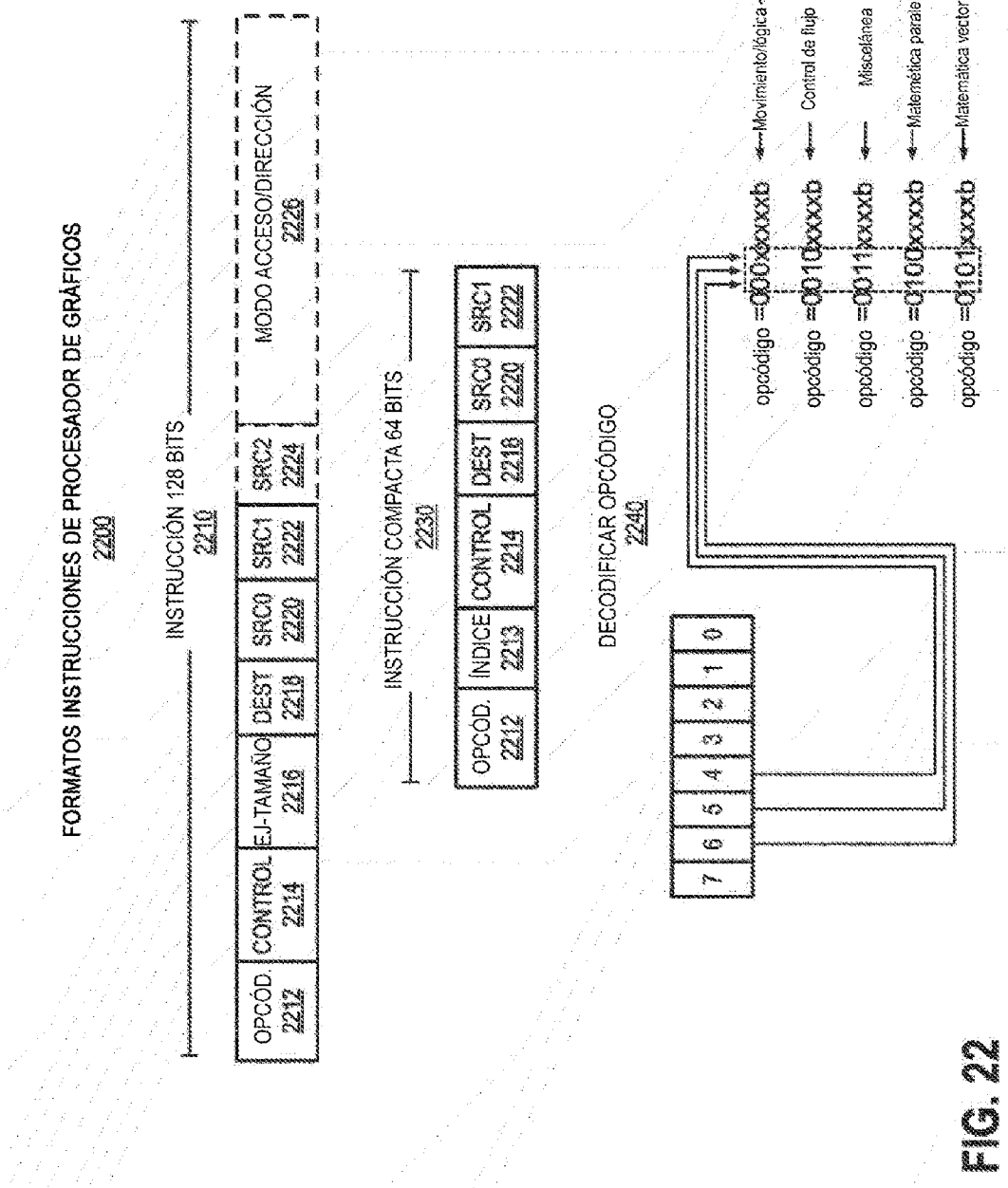

FORMATOS INSTRUCCIONES DE PROCESADOR DE GRÁFICOS
2200
INSTRUCCIÓN 128 BITS
2210
OPCÓD. 2212
CONTROL 2214
EJ-TAMAÑO 2216
DEST 2218
SRC0 2220
SRC1 2222
SRC2 2224
MODO ACCESO/DIRECCIÓN 2226
INSTRUCCIÓN COMPACTA 64 BITS
2230
OPCÓD. 2212
ÍNDICE 2213
CONTROL 2214
DEST 2218
SRC0 2220
SRC1 2222
DECODIFICAR OPCÓDIGO
2240
7 6 5 4 3 2 1 0
opcódigo =000xxxxb
opcódigo =001xxxxb
opcódigo =010xxxxb
opcódigo =011xxxxb
opcódigo =100xxxxb
opcódigo =101xxxxb
Movimiento/lógica - 2242
Control de flujo - 2244
Miscelánea - 2246
Matemática paralela - 2248
Matemática vector - 2250

FIG. 22

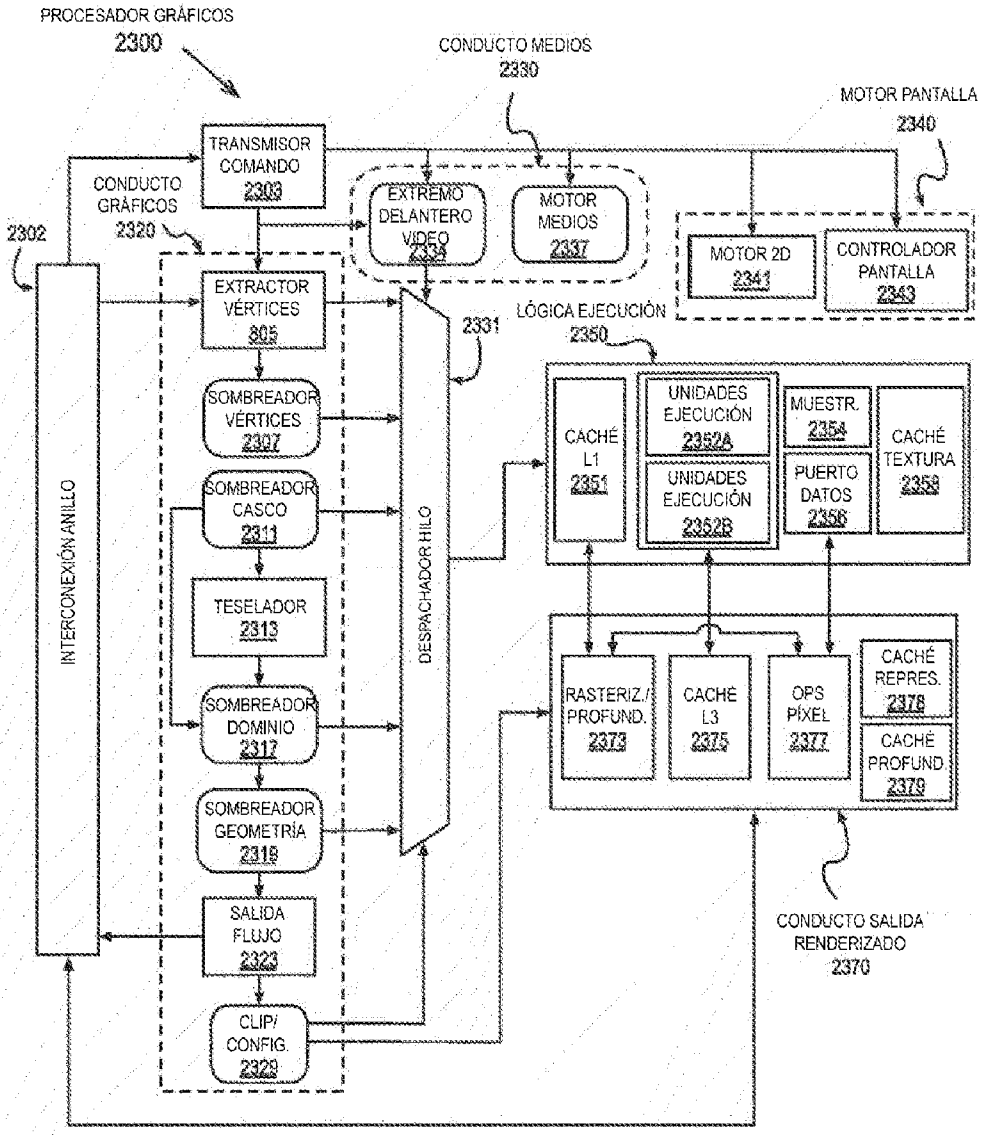
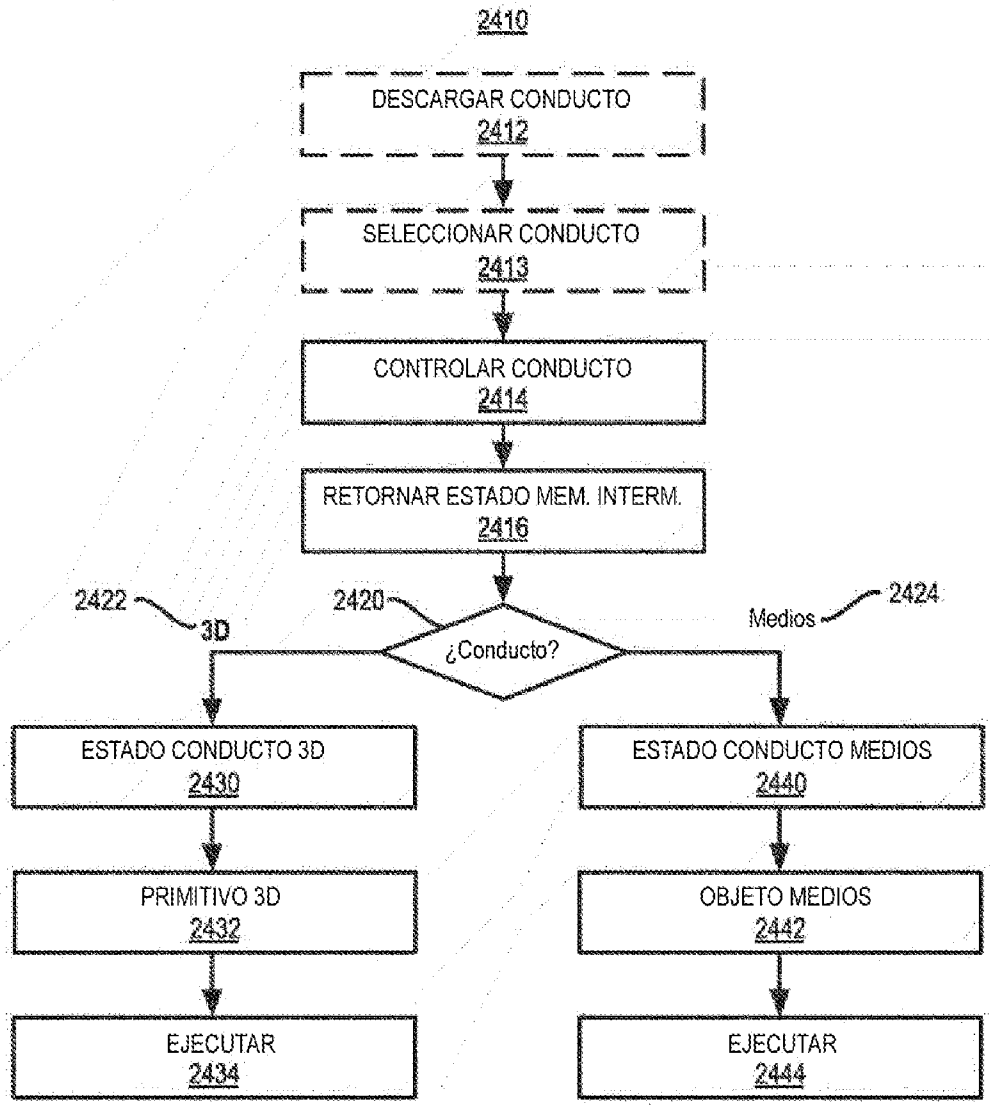

PROCESADOR GRÁFICOS
2300
CONDUCTO MEDIOS
2330
MOTOR PANTALLA
2340
TRANSMISOR COMANDO
2303
CONDUCTO GRÁFICOS
2320
2302
EXTREMO DELANTERO VÍDEO
2334
MOTOR MEDIOS
2337
MOTOR 2D
2341
CONTROLADOR PANTALLA
2343
EXTRACTOR VÉRTICES
805
2331
LÓGICA EJECUCIÓN
2350
SOMBREADOR VÉRTICES
2307
SOMBREADOR CASCO
2311
TESELADOR
2313
SOMBREADOR DOMINIO
2317
SOMBREADOR GEOMETRÍA
2319
SALIDA FLUJO
2323
CLIP/ CONFIG.
2329
INTERCONEXIÓN ANILLO
DESPACHADOR HILO
CACHÉ L1
2351
UNIDADES EJECUCIÓN
2352A
UNIDADES EJECUCIÓN
2352B
MUESTR.
2354
PUERTO DATOS
2356
CACHÉ TEXTURA
2358
RASTERIZ./ PROFUND.
2373
CACHÉ L3
2375
OPS PÍXEL
2377
CACHÉ REPRES.
2378
CACHÉ PROFUND.
2379
CONDUCTO SALIDA RENDERIZADO
2370

FIG. 23

FIG. 24A FORMATO COMANDOS DE PROCESADOR DE GRÁFICOS

FIG. 24B SECUENCIA DE COMANDOS DE PROCESADOR DE GRÁFICOS



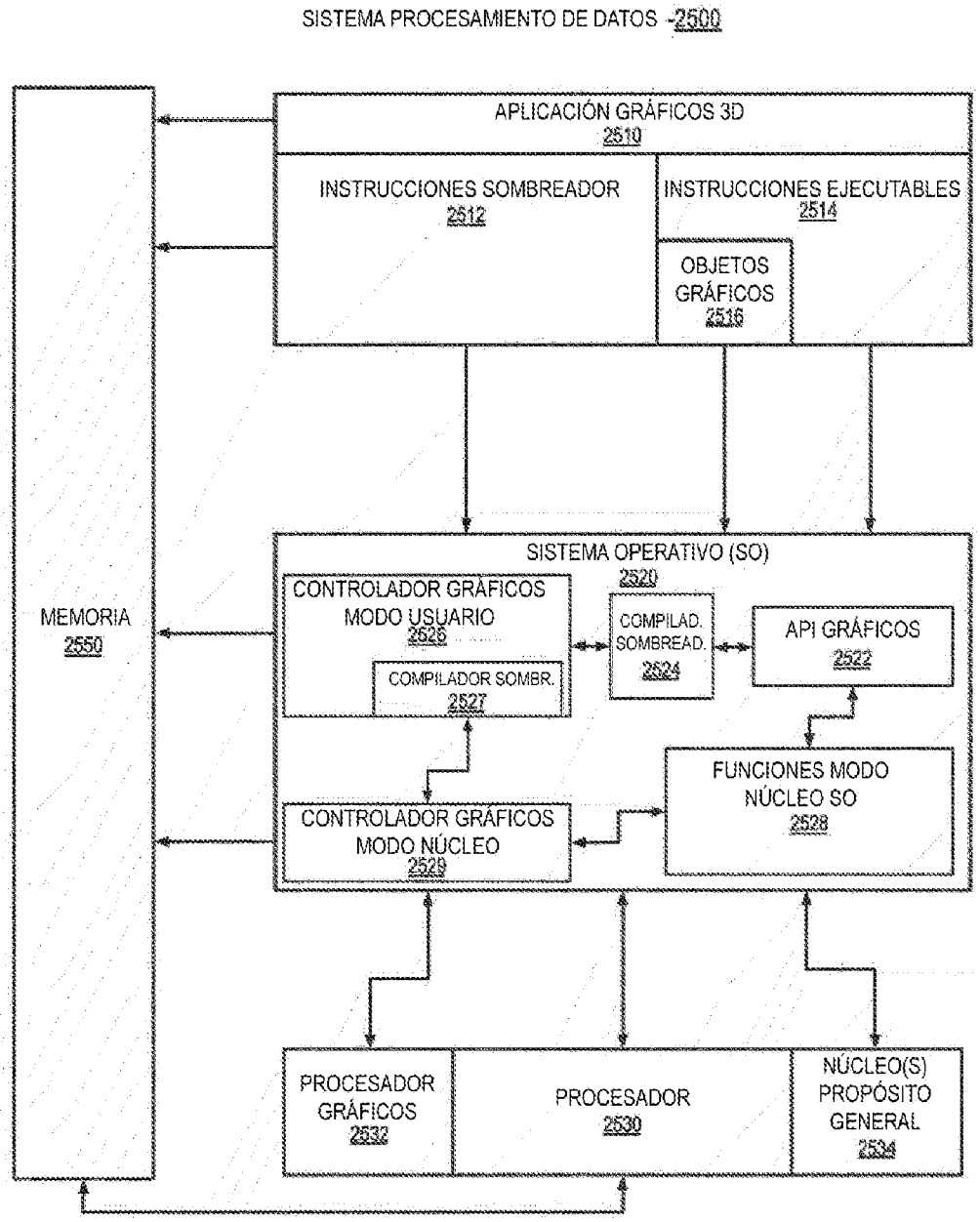

SISTEMA PROCESAMIENTO DE DATOS -2500
APLICACIÓN GRÁFICOS 3D
2510
INSTRUCCIONES SOMBREADOR
2512
INSTRUCCIONES EJECUTABLES
2514
OBJETOS GRÁFICOS
2516
MEMORIA
2550
SISTEMA OPERATIVO (SO)
2520
CONTROLADOR GRÁFICOS MODO USUARIO
2526
COMPILADOR SOMBR.
2527
COMPILAD. SOMBREAD.
2524
API GRÁFICOS
2522
FUNCIONES MODO NÚCLEO SO
2528
CONTROLADOR GRÁFICOS MODO NÚCLEO
2529
PROCESADOR GRÁFICOS
2532
PROCESADOR
2530
NÚCLEO(S) PROPÓSITO GENERAL
2534

FIG. 25

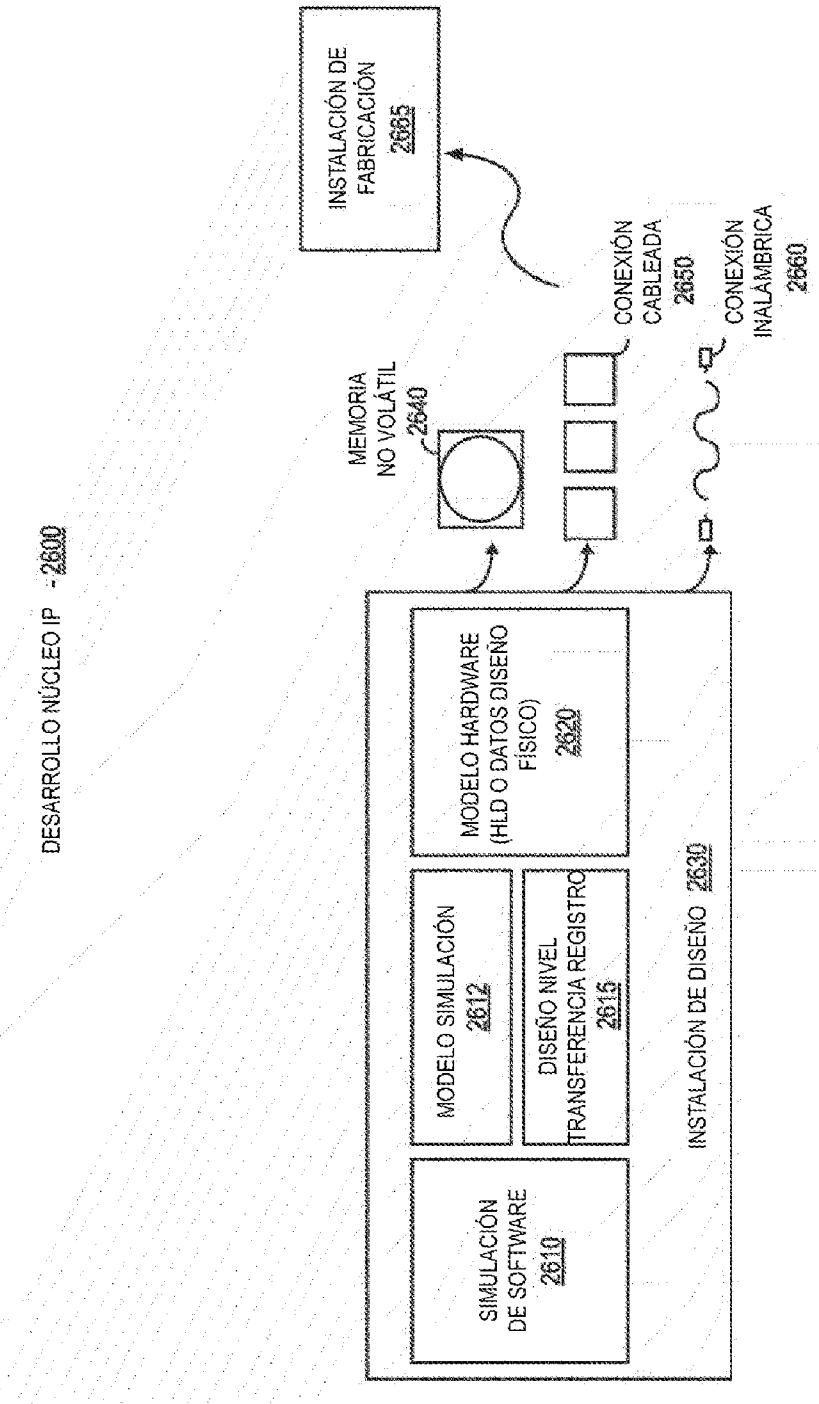

DESARROLLO NÚCLEO IP 2600
INSTALACIÓN DE FABRICACIÓN 2685
MEMORIA NO VOLÁTIL 2640
CONEXIÓN CABLEADA 2650
CONEXIÓN INALÁMBRICA 2660
SIMULACIÓN DE SOFTWARE 2610
MODELO SIMULACIÓN 2612
DISEÑO NIVEL TRANSFERENCIA REGISTRO 2615
MODELO HARDWARE (HLD O DATOS DISEÑO FÍSICO) 2620
INSTALACIÓN DE DISEÑO 2630

FIG. 26

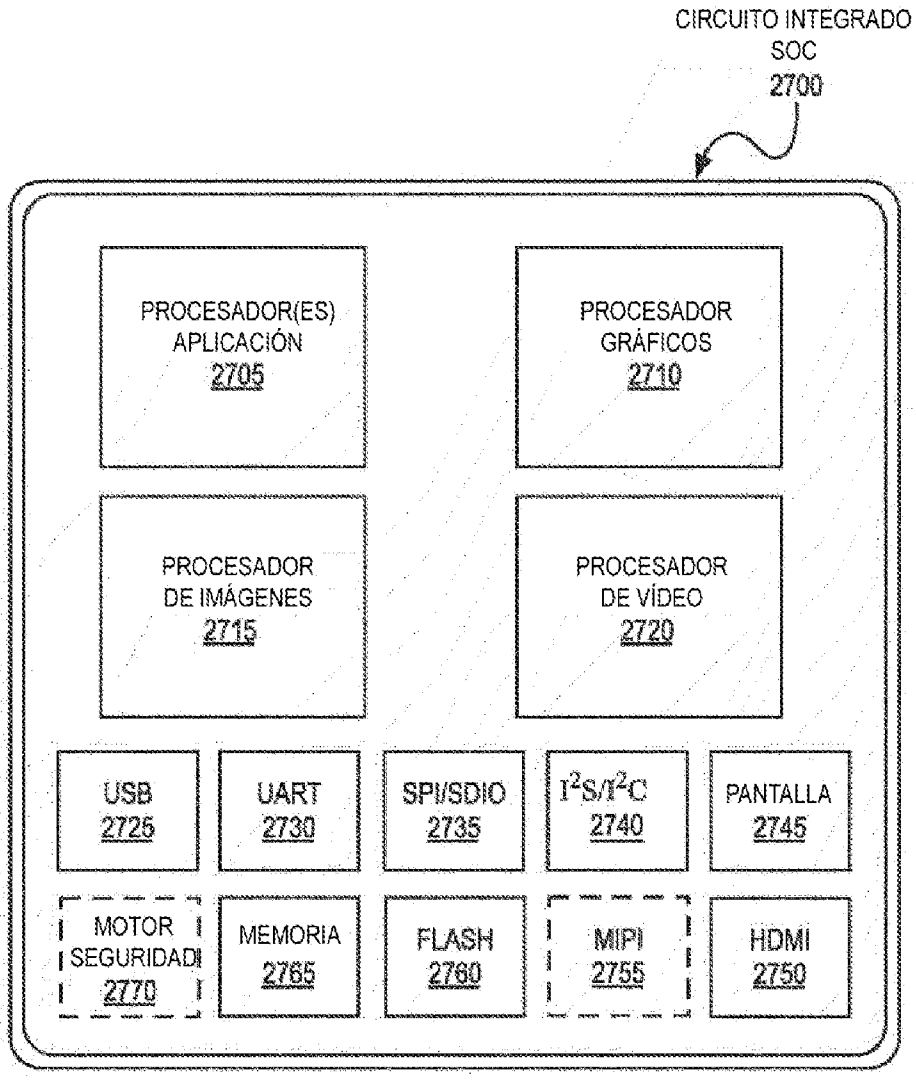

CIRCUITO INTEGRADO
SOC
2700
PROCESADOR(ES)
APLICACIÓN
2705
PROCESADOR
GRÁFICOS
2710
PROCESADOR
DE IMÁGENES
2715
PROCESADOR
DE VÍDEO
2720
USB
2725
UART
2730
SPI/SDIO
2735
I2S/I2C
2740
PANTALLA
2745
MOTOR
SEGURIDAD
2770
MEMORIA
2765
FLASH
2760
MIPI
2755
HDMI
2750

FIG. 27

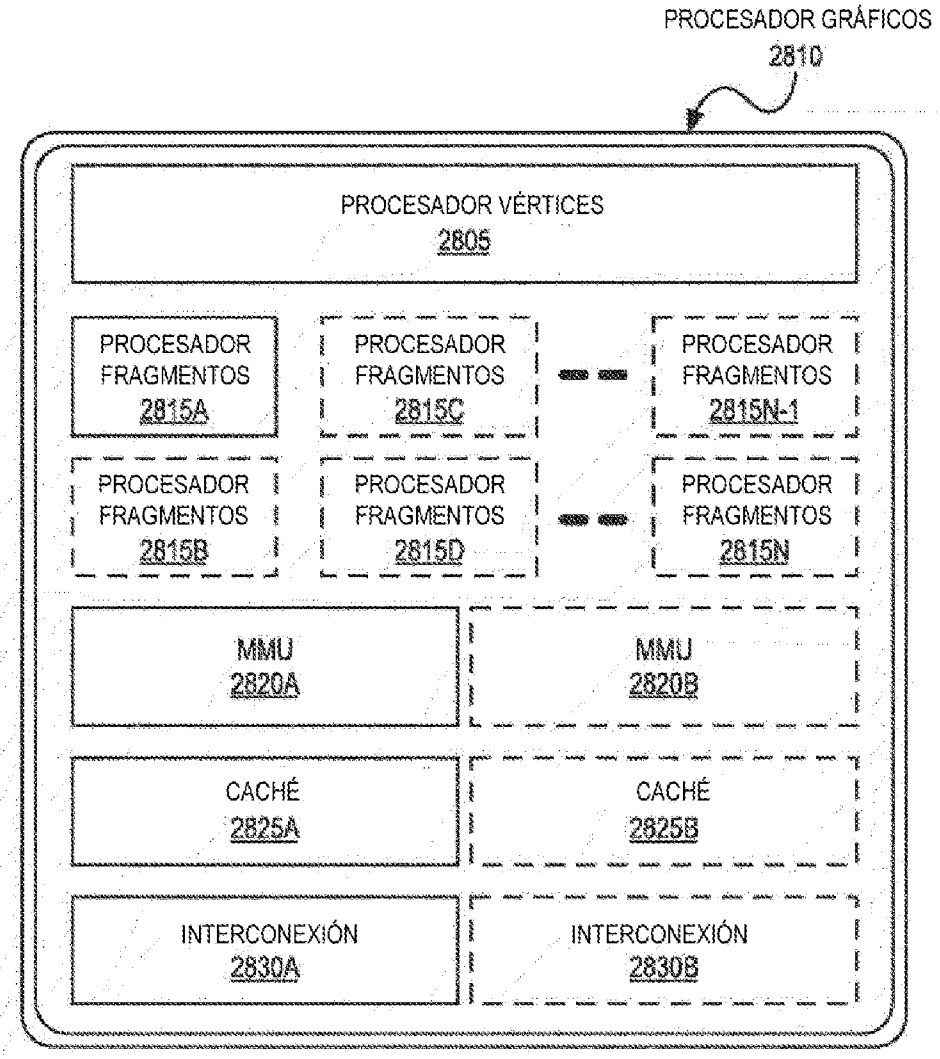

PROCESADOR GRÁFICOS
2810
PROCESADOR VÉRTICES
2805
PROCESADOR FRAGMENTOS
2815A
PROCESADOR FRAGMENTOS
2815C
PROCESADOR FRAGMENTOS
2815N-1
PROCESADOR FRAGMENTOS
2815B
PROCESADOR FRAGMENTOS
2815D
PROCESADOR FRAGMENTOS
2815N
MMU
2820A
MMU
2820B
CACHÉ
2825A
CACHÉ
2825B
INTERCONEXIÓN
2830A
INTERCONEXIÓN
2830B

FIG. 28

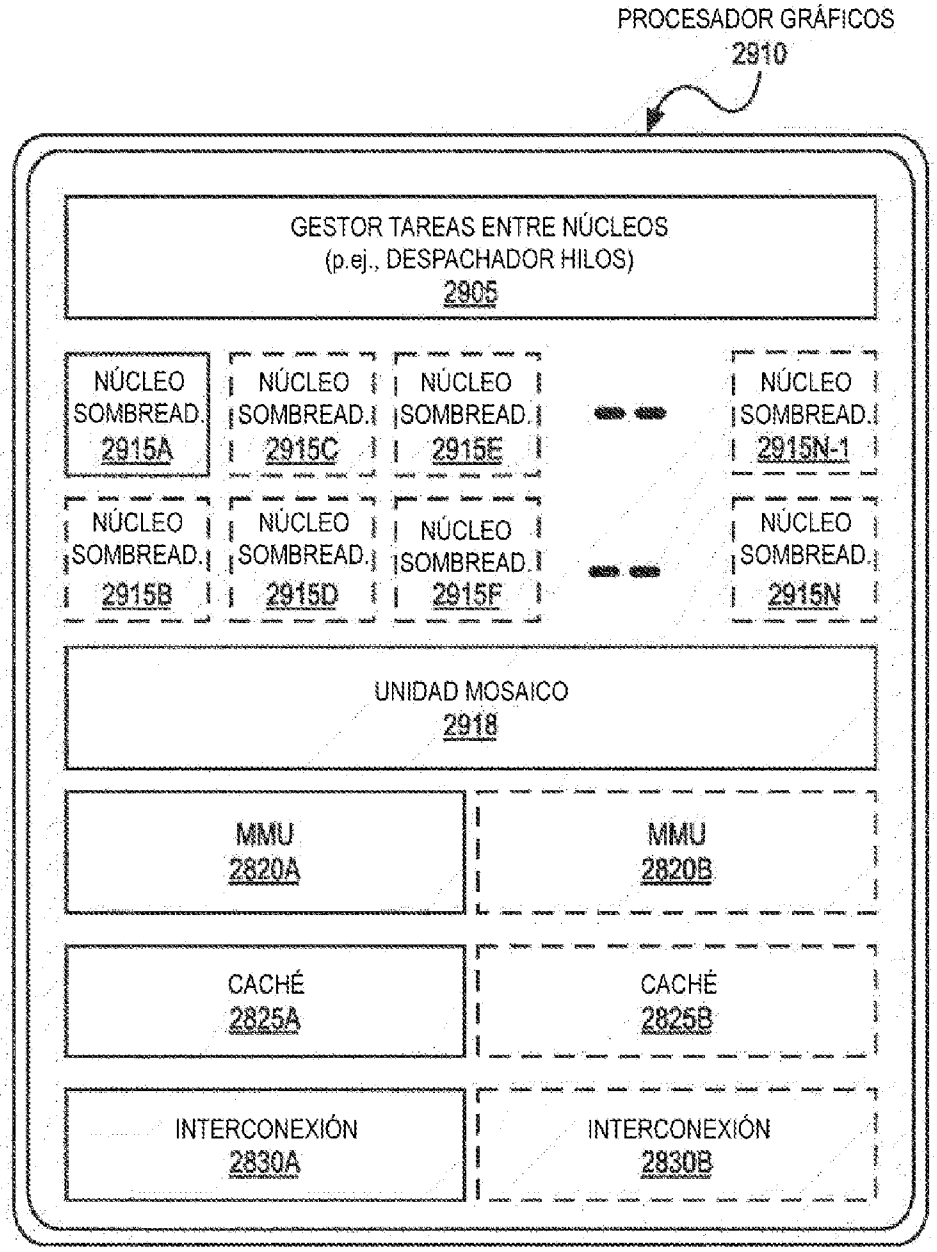

PROCESADOR GRÁFICOS
2910
GESTOR TAREAS ENTRE NÚCLEOS
(p.ej., DESPACHADOR HILOS)
2905
NÚCLEO SOMBREAD.
2915A
NÚCLEO SOMBREAD.
2915C
NÚCLEO SOMBREAD.
2915E
NÚCLEO SOMBREAD.
2915N-1
NÚCLEO SOMBREAD.
2915B
NÚCLEO SOMBREAD.
2915D
NÚCLEO SOMBREAD.
2915F
NÚCLEO SOMBREAD.
2915N
UNIDAD MOSAICO
2918
MMU
2820A
MMU
2820B
CACHÉ
2825A
CACHÉ
2825B
INTERCONEXIÓN
2830A
INTERCONEXIÓN
2830B

FIG. 29