

19



OFICINA ESPAÑOLA DE
PATENTES Y MARCAS

ESPAÑA



11 Número de publicación: **2 934 735**

51 Int. Cl.:

G06F 7/58

(2006.01)

12

TRADUCCIÓN DE PATENTE EUROPEA

T3

86 Fecha de presentación y número de la solicitud internacional: **07.04.2017** **PCT/CN2017/079720**

87 Fecha y número de publicación internacional: **11.10.2018** **WO18184205**

96 Fecha de presentación y número de la solicitud europea: **07.04.2017** **E 17904773 (3)**

97 Fecha y número de publicación de la concesión europea: **16.11.2022** **EP 3607412**

54 Título: **Sistemas y métodos para generar números aleatorios gaussianos con aceleración por hardware**

45 Fecha de publicación y mención en BOPI de la traducción de la patente:
24.02.2023

73 Titular/es:

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

72 Inventor/es:

GUO, YIWEN;
YAO, ANBANG;
CAI, DONGQI;
WANG, LIBIN;
XU, LIN;
HU, PING;
WANG, SHANDONG y
CHENG, WENHUA

74 Agente/Representante:

LEHMANN NOVO, María Isabel

ES 2 934 735 T3

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

DESCRIPCIÓN

Sistemas y métodos para generar números aleatorios gaussianos con aceleración por hardware

5 CAMPO TÉCNICO

Las formas de realización se refieren, en general, al procesamiento de datos y más en particular al procesamiento de datos mediante una unidad de procesamiento gráfico de uso general. En particular, las formas de realización se refieren a sistemas y métodos para generar números aleatorios gaussianos con aceleración por hardware.

10 ANTECEDENTES DE LA INVENCION

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

Para aumentar aún más el rendimiento, los procesadores de gráficos suelen poner en práctica técnicas de procesamiento tal como canalización que intentan procesar, en paralelo, la mayor cantidad posible de datos gráficos a lo largo de las diferentes partes de la canalización de gráficos. Los procesadores de gráficos en paralelo con arquitecturas de una única instrucción y múltiples subprocesos (SIMT) están diseñados para maximizar la magnitud de procesamiento en paralelo en la canalización de gráficos. En una arquitectura SIMT, los grupos de subprocesos en paralelo intentan ejecutar las instrucciones del programa de forma sincronizada con la mayor frecuencia posible para aumentar la eficiencia del procesamiento. Se puede encontrar una descripción general del software y del hardware para las arquitecturas SIMT en Shane Cook, Programación CUDA, Capítulo 3, páginas 37 a 51 (2013).

Para la generación de números aleatorios, se suelen utilizar métodos Box-muller, inversión de función de densidad acumulativa (CDF), transformación, rechazo y recursivo. El método de inversión CDF simplemente invierte la CDF para obtener un número aleatorio a partir de una distribución deseada. Los métodos de transformación implican la transformación directa de números aleatorios uniformes en una distribución gaussiana. La tercera categoría, el rechazo, nuevamente comienza con números aleatorios uniformes y una transformación, pero tiene la etapa adicional de rechazar de manera condicional algunos de los valores transformados. La recursión, la categoría final, utiliza combinaciones lineales de números gaussianos generados previamente para obtener nuevos resultados. El método Box-muller requiere mucho tiempo y es lento para generar números aleatorios gaussianos.

El documento US 2003/177155 A1 da a conocer una matriz celular para generar un flujo de números aleatorios en una distribución gaussiana a partir de un flujo de números aleatorios en una distribución uniforme. La matriz incluye células de repetición que reciben un bit como entrada, almacenan el bit, suman el bit a un bit previamente almacenado y obtienen un solo bit suma como salida. Las células incluyen un sumador y uno o más registros de almacenamiento de un único bit.

La invención se define por un aparato y un método para generar números aleatorios gaussianos según las reivindicaciones independientes. Las formas de realización preferidas se definen en las reivindicaciones dependientes.

BREVE DESCRIPCIÓN DE LOS DIBUJOS

Para que la manera en que las características mencionadas con anterioridad de las presentes formas de realización se puedan entender en detalle, se puede obtener una descripción más particular de las formas de realización, resumidas brevemente con anterioridad, con referencia a las formas de realización, algunas de las cuales se ilustran en los dibujos adjuntos. Sin embargo, conviene señalar que los dibujos adjuntos ilustran solamente formas de realización típicas y, por lo tanto, no deben considerarse limitativos de su alcance.

La Figura 1 es un diagrama de bloques que ilustra un sistema informático configurado para poner en práctica uno o más aspectos de las formas de realización descritas en este documento.

Las Figura s 2A-2D ilustran los componentes de un procesador en paralelo, según una forma de realización.

Las Figura 3A-3B son diagramas de bloques de multiprocesadores de gráficos, según formas de realización.

Las Figura s 4A-4F ilustran una arquitectura, a modo de ejemplo, en donde una pluralidad de GPUs se acoplan de manera comunicativa a una pluralidad de procesadores multinúcleo.

La Figura 5 ilustra una canalización de procesamiento de gráficos, según una forma de realización.

- La Figura 6 ilustra un método 600 para generar números aleatorios gaussianos con aceleración por hardware según una forma de realización.
- 5 La Figura 7 ilustra un diagrama de bloques de un sistema para generar números aleatorios gaussianos con aceleración por hardware según una forma de realización.
- La Figura 8 ilustra un diagrama de tiempo para la generación de números aleatorios gaussianos de conformidad con una forma de realización.
- 10 La Figura 9 ilustra una pila de software de aprendizaje máquina, según una forma de realización.
- La Figura 10 ilustra una unidad de procesamiento gráfico de uso general altamente paralela, según una forma de realización.
- 15 La Figura 11 ilustra un sistema informático multi-GPU, según una forma de realización.
- Las Figuras 12A-B ilustran capas de redes neuronales profundas a modo de ejemplo.
- La Figura 13 ilustra una red neuronal recurrente a modo de ejemplo.
- 20 La Figura 14 ilustra el entrenamiento y el despliegue de una red neuronal de profundidad.
- La Figura 15 es un diagrama de bloques que ilustra el aprendizaje distribuido.
- 25 La Figura 16 ilustra un sistema de inferencia, a modo de ejemplo, en un circuito integrado (SOC) adecuado para realizar inferencias utilizando un modelo capacitado.
- La Figura 17 es un diagrama de bloques de un sistema de procesamiento 1700, según una forma de realización
- 30 La Figura 18 es un diagrama de bloques de una forma de realización de un procesador 1800 que tiene uno o más núcleos de procesador 1802A-1802N, un controlador de memoria integrado 1814 y un procesador de gráficos integrado 1808.
- La Figura 19 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 una pluralidad de núcleos de procesamiento.
- 35 La Figura 20 es un diagrama de bloques de un motor de procesamiento de gráficos 2010 de un procesador de gráficos de conformidad con algunas formas de realización.
- 40 La Figura 21 es un diagrama de bloques de otra forma de realización de un procesador de gráficos 2100.
- La Figura 22 ilustra la lógica de ejecución de subprocesos 2200 que incluye una matriz de elementos de procesamiento empleados en algunas formas de realización de un GPE.
- 45 La Figura 23 es un diagrama de bloques que ilustra los formatos de instrucción 2300 de un procesador de gráficos según algunas formas de realización.
- La Figura 24 es un diagrama de bloques de otra forma de realización de un procesador de gráficos 2400.
- 50 La Figura 25A es un diagrama de bloques que ilustra un formato de comando de procesador de gráficos 2500 según algunas formas de realización.
- La Figura 25B es un diagrama de bloques que ilustra una secuencia de comandos del procesador de gráficos 2510 según una forma de realización.
- 55 La Figura 26 ilustra una arquitectura de software de gráficos a modo de ejemplo para un sistema de procesamiento de datos 2600 según algunas formas de realización.
- 60 La Figura 27 es un diagrama de bloques que ilustra un sistema de desarrollo de núcleo de IP 2700 que puede utilizarse para fabricar un circuito integrado para realizar operaciones de conformidad con una forma de realización; y
- Las Figuras 28-30 ilustran ejemplos de circuitos integrados y procesadores de gráficos asociados que pueden fabricarse utilizando uno o más núcleos de IP, de conformidad con varias formas de realización descritas en este documento.
- 65

Además, de lo que se ilustra, se pueden incluir otros circuitos y lógica, incluyendo procesadores/núcleos de gráficos adicionales, controladores de interfaz periféricos o núcleos de procesador de uso general.

DESCRIPCIÓN DETALLADA DE LA INVENCION

En algunas formas de realización, una unidad de procesamiento gráfico (GPU) se acopla de manera comunicativa a núcleos de host/procesador para acelerar las operaciones de gráficos, las operaciones de aprendizaje máquina, las operaciones de análisis de patrones y diversas funciones de GPU de uso general (GPGPU). La GPU se puede acoplar de manera comunicativa a los núcleos de host/procesador mediante un bus u otra interconexión (por ejemplo, una interconexión de alta velocidad tales como PCIe o NVLink). En otras formas de realización, la GPU puede integrarse en el mismo paquete o circuito integrado que los núcleos y acoplarse de manera comunicativa a los núcleos mediante un bus/interconexión de procesador interno (es decir, interno al paquete o al circuito integrado). De manera independiente, de la forma en que se conecte la GPU, los núcleos del procesador pueden asignar trabajo a la GPU en forma de secuencias de comandos/instrucciones contenidas en un descriptor de trabajo. A continuación, la GPU utiliza circuitos/lógica dedicados para procesar, de manera eficiente, estos comandos/instrucciones.

En la siguiente descripción, se exponen numerosos detalles específicos para proporcionar una comprensión más completa. Sin embargo, será evidente para un experto en esta 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.

Descripción general del sistema

La Figura 1 es un diagrama de bloques que ilustra un sistema informático 100 configurado para poner en práctica uno o más aspectos de las formas de realización descritas en este documento. El sistema informático 100 incluye un subsistema de procesamiento 101 que tiene uno o más procesadores 102 y una memoria del sistema 104 que se 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 circuitos integrados o puede integrarse dentro de uno o más procesadores 102. El concentrador de memoria 105 se acopla con un subsistema de E/S 111 mediante un enlace de comunicación 106. El subsistema de E/S 111 incluye un concentrador de E/S 107 que puede permitir al sistema informático 100 recibir entradas desde uno o más dispositivos de entrada 108. Además, el concentrador de E/S 107 puede habilitar un controlador de pantalla, que puede estar incluido en uno o más procesadores 102, para proporcionar salidas a uno o más dispositivo(s) de visualización 110A. En una forma de realización, uno o más dispositivos de visualización 110A, acoplados con el concentrador de E/S 107, pueden incluir un dispositivo de visualización local, interno o incorporado.

En una forma de realización, el subsistema de procesamiento 101 incluye uno o más procesadores en paralelo 112 acoplados al concentrador de memoria 105 mediante un bus u otro enlace de comunicación 113. El enlace de comunicación 113 puede ser uno de entre cualquier número de tecnologías de enlace de comunicación basadas en normas o protocolos, tales como, sin limitación, PCI Express, o puede ser una interfaz de comunicaciones o una estructura de comunicaciones específica del proveedor. En una forma de realización, uno o más procesadores en paralelo 112 forman un sistema de procesamiento vectorial o en paralelo concentrado de forma computacional que pueden incluir un gran número de núcleos de procesamiento y/o grupos de procesamiento, tales como un procesador de núcleos integrados (MIC). En una forma de realización, los uno o más procesadores en paralelo 112 forman un subsistema de procesamiento de gráficos que puede enviar píxeles a uno o más dispositivos de visualización 110A acoplados mediante el concentrador de E/S 107. Los uno o más procesadores en paralelo(s) 112 también puede incluir un controlador de pantalla y una interfaz de pantalla (no ilustrada) para permitir una conexión directa a uno o más dispositivos de visualización 110B.

Dentro del subsistema de E/S 111, una unidad de almacenamiento del sistema 114 se puede conectar al concentrador de E/S 107 para proporcionar un mecanismo de almacenamiento para el sistema informático 100. Se puede utilizar un conmutador de E/S 116 para proporcionar un mecanismo de interfaz para permitir las conexiones entre el concentrador de E/S 107 y otros componentes, tal 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 agregarse mediante uno o más dispositivo(s) de adición 120. El adaptador de red 118 puede ser un adaptador Ethernet u otro adaptador de red por cable. El adaptador de red inalámbrico 119 puede incluir uno o más de entre Wi-Fi, Bluetooth, comunicación de campo cercano (NFC) u otro dispositivo de red que incluya uno o más dispositivos de radio inalámbricos.

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

En una forma de realización, uno o más procesadores en paralelo 112 incorporan circuitos optimizados para procesamiento de gráficos y de vídeo, incluyendo, por ejemplo, circuitos de salida de vídeo, y constituyen una unidad de procesamiento gráfico (GPU). En otra forma de realización, uno o más procesadores en paralelo 112 incorporan circuitos optimizados para procesamiento de uso general, al tiempo que conservan la arquitectura computacional subyacente, descrita con mayor detalle en este documento. En otra forma de realización adicional, los componentes del sistema informático 100 pueden integrarse con uno o más elementos del sistema en un único circuito integrado. Por ejemplo, uno o más procesadores en paralelo 112, el concentrador de memoria 105, los procesadores 102 y el concentrador de E/S 107 pueden integrarse en un sistema de circuito integrado (SoC). De manera alternativa, los componentes del sistema informático 100 pueden integrarse en un paquete único para formar una configuración de sistema en paquete (SIP). En una forma de realización, al menos una parte de los componentes del sistema informático 100 se puede integrar en un módulo de circuito integrado múltiple (MCM), que se puede interconectar con otros módulos de circuito integrado múltiples en un sistema informático modular.

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

Algunos de los componentes particulares que se muestran en este documento son opcionales y pueden no estar incluidos en todas las puestas en práctica del sistema informático 100. Por ejemplo, se puede admitir cualquier número de tarjetas adicionales o periféricos, o se pueden eliminar algunos componentes. Además, algunas arquitecturas pueden utilizar una terminología diferente para componentes similares a los ilustrados en la Figura 1. Por ejemplo, el concentrador de memoria 105 puede denominarse *Northbridge* en algunas arquitecturas, mientras que el concentrador de E/S 107 puede denominarse *Southbridge*.

La Figura 2A ilustra un procesador en paralelo 200, según una forma de realización. Los diversos componentes del procesador en paralelo 200 pueden ponerse en práctica utilizando uno o más dispositivos de circuito integrado, tales como procesadores programables, circuitos integrados específicos de aplicación (ASIC) o matrices de puertas programables en campo (FPGA). El procesador en paralelo ilustrado 200 es una variante de uno o más procesadores en paralelo 112 mostrados en la Figura 1, según una forma de realización.

En una forma de realización, el procesador en paralelo 200 incluye una unidad de procesamiento en paralelo 202. La unidad de procesamiento en paralelo incluye una unidad de E/S 204 que permite la comunicación con otros dispositivos, incluyendo otras instancias de la unidad de procesamiento en 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 mediante el uso de una interfaz de concentrador o conmutador, tal como el concentrador de memoria 105. Las conexiones entre el concentrador de memoria 105 y la unidad de E/S 204 forman un enlace de comunicación 113. Dentro de la unidad de procesamiento en paralelo 202, la unidad de E/S 204 se conecta con una interfaz host 206 y una barra transversal de memoria 216, en donde la interfaz host 206 recibe comandos dirigidos a realizar operaciones de procesamiento y la barra transversal de memoria 216 recibe comandos dirigidos a realizar operaciones de memoria.

Cuando la interfaz host 206 recibe una memoria intermedia de comando mediante la unidad de E/S 204, la interfaz host 206 puede dirigir las operaciones de trabajo para realizar dichos comandos hacia un extremo frontal 208. En una forma de realización, el extremo frontal 208 se acopla con un planificador 210, que está configurado para distribuir comandos u otros elementos de trabajo a una matriz de grupos de procesamiento 212. En una forma de realización, el planificador 210 garantiza que la matriz de grupos de procesamiento 212 esté configurada de manera correcta y en un estado válido antes de que las tareas se distribuyan a los grupos de procesamiento de la matriz de grupos de procesamiento 212. En una forma de realización, el planificador 210 se pone en práctica mediante la lógica de firmware que se ejecuta en un microcontrolador. El planificador 210, puesto en práctica por el microcontrolador, se puede configurar para realizar operaciones complejas de programación y de distribución del trabajo con granularidad gruesa y fina, lo que permite una rápida preferencia y conmutador de contextos de los subprocesos que se ejecutan en la matriz de procesamiento 212. En una forma de realización, el software de host puede sondear las cargas de trabajo para la programación en la matriz de procesamiento 212 mediante uno de múltiples timbres de procesamiento de gráficos. Entonces, las cargas de trabajo pueden distribuirse de manera automática mediante la matriz de procesamiento 212 a través de la lógica del planificador 210 dentro del microcontrolador del planificador.

La matriz de grupos de procesamiento 212 puede incluir hasta "N" grupos de procesamiento (p. ej., grupo 214A, grupo 214B, grupo 214N). Cada grupo 214A-214N de la matriz de grupos de procesamiento 212 puede ejecutar una gran

cantidad de subprocesos de manera simultánea. El planificador 210 puede asignar trabajo a los grupos 214A-214N de la matriz de grupos de procesamiento 212 utilizando diversos algoritmos de programación y/o distribución del trabajo, que pueden variar dependiendo de la carga de trabajo que surja para cada tipo de programa o cálculo. La programación puede ser gestionada de manera dinámica por el planificador 210, o puede ser asistida, en parte, por la lógica del compilador durante la compilación de la lógica del programa configurada para ser ejecutada por la matriz de grupos de procesamiento 212. En una forma de realización, diferentes grupos 214A-214N de la matriz de grupos de procesamiento 212 se pueden asignar para procesar diferentes tipos de programas o para realizar diferentes tipos de cálculos.

La matriz de grupos de procesamiento 212 se puede configurar para realizar diversos tipos de operaciones de procesamiento en paralelo. En una forma de realización, la matriz de grupos de procesamiento 212 está configurada para realizar operaciones informáticas paralelas de uso general. Por ejemplo, la matriz de grupos de procesamiento 212 puede incluir lógica para ejecutar tareas de procesamiento, que incluyen el filtrado de datos de vídeo y/o audio, realizar operaciones de modelado, incluyendo operaciones físicas, y realizar transformaciones de datos.

En una forma de realización, la matriz de grupos de procesamiento 212 está configurada para realizar operaciones de procesamiento de gráficos en paralelo. En formas de realización en las que el procesador en paralelo 200 está configurado para realizar operaciones de procesamiento de gráficos, la matriz de grupos de procesamiento 212 puede incluir lógica adicional para admitir la ejecución de dichas operaciones de procesamiento de gráficos, que incluyen, pero sin limitación, a lógica de muestreo de textura para realizar operaciones de textura, así como lógica de teselado y otra lógica de procesamiento de vértices. Además, la matriz de grupos de procesamiento 212 puede configurarse para ejecutar programas de sombreadores relacionados con el procesamiento de gráficos tales como, pero sin limitación, sombreadores de vértices, sombreadores de teselado, sombreadores de geometría y sombreadores de píxeles. La unidad de procesamiento en paralelo 202 puede transferir datos desde la memoria del sistema mediante la unidad de E/S 204 para su procesamiento. Durante el procesamiento, los datos transmitidos pueden almacenarse en la memoria del circuito integrado (por ejemplo, la memoria de procesadores en paralelo 222) durante el procesamiento y a continuación, volver a ser objeto de escritura en la memoria del sistema.

En una forma de realización, cuando la unidad de procesamiento en paralelo 202 se utiliza para realizar el procesamiento de gráficos, el planificador 210 puede configurarse para dividir la carga de trabajo de procesamiento de gráficos a múltiples grupos 214A-214N de la matriz de grupos de procesamiento 212. En algunas formas de realización, partes de la matriz de grupos de procesamiento 212 se pueden configurar para realizar diferentes tipos de procesamiento. Por ejemplo, una primera parte puede configurarse para realizar sombreado de vértices y generación de topología, una segunda parte puede configurarse para realizar teselado y sombreado de geometría, y una tercera parte puede configurarse para realizar sombreado de píxeles u otras operaciones de espacio de pantalla, para obtener una imagen renderizada para su visualización. Los datos intermedios obtenidos por uno 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 un procesamiento posterior.

Durante el funcionamiento, la matriz de grupos de procesamiento 212 puede recibir tareas de procesamiento para ser ejecutadas 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 para ser procesados, por ejemplo, datos de superficie (parche), datos primitivos, datos de vértices y/o datos de píxeles, así como parámetros de estado y comandos que definen cómo se procesarán los datos (por ejemplo, qué programa se ejecutará). El planificador 210 puede configurarse para obtener los índices correspondientes 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 esté configurada en un estado válido antes de que se inicie la carga de trabajo especificada por las memorias intermedias de comandos entrantes (por ejemplo, memorias intermedias por lotes, memorias intermedias de inserción, etc.).

Cada una o más instancias de la unidad de procesamiento en paralelo 202 pueden acoplarse con la memoria de procesadores en paralelo 222. Se puede acceder a la memoria de procesadores en 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 procesadores en paralelo 222 mediante una interfaz de memoria 218. La interfaz de memoria 218 puede incluir múltiples unidades de partición (por ejemplo, unidad de partición 220A, unidad de partición 220B, a unidad de partición 220N, inclusive) que pueden acoplarse cada una a una parte (por ejemplo, unidad de memoria) de la memoria de procesadores en paralelo 222. En una puesta en práctica, el número de unidades de partición 220A-220N está configurado para ser igual al número de unidades de memoria, de modo que una primera unidad de partición 220A tiene una primera unidad de memoria 224A correspondiente, una segunda unidad de partición 220B tiene una unidad de memoria correspondiente 224B, y una N-ésima unidad de partición 220N tiene una N-ésima unidad de memoria 224N correspondiente. En otras formas de realización, el número de unidades de partición 220A-220N puede no ser igual al número de dispositivos de memoria.

En varias formas de realización, las unidades de memoria 224A-224N pueden incluir diversos tipos de dispositivos de memoria, incluyendo la memoria de acceso aleatorio dinámica (DRAM) o memoria de acceso aleatorio gráfica, tal como la memoria de acceso aleatorio gráfica síncrona (SGRAM), incluyendo memoria de tasa de datos dobles gráfica (G17R). En una forma de realización, las unidades de memoria 224A-224N también pueden incluir memoria apilada en 3D, que incluye, entre otras, memoria de alto ancho de banda (HBM). Los expertos en esta técnica apreciarán que la puesta en práctica específica de las unidades de memoria 224A-224N puede variar y puede seleccionarse entre diversos diseños convencionales. Los objetivos de procesamiento, tales como las memorias intermedias de tramas o los mapas de textura, pueden almacenarse en las unidades de memoria 224A-224N, lo que permite que las unidades de partición 220A-220N realicen la escritura de partes de cada objetivo de procesamiento en paralelo para utilizar de manera eficiente el ancho de banda disponible de la memoria de procesadores en paralelo 222. En algunas formas de realización, una instancia local de la memoria de procesadores en paralelo 222 puede excluirse en favor de un diseño de memoria unificada que utiliza la memoria del sistema junto con la memoria caché local.

En una forma de realización, cualquiera de los grupos 214A-214N de la matriz de grupos de procesamiento 212 puede procesar datos que serán objeto de escritura en cualquiera de las unidades de memoria 224A-224N dentro de la memoria de procesadores en paralelo 222. La barra transversal de memoria 216 se puede configurar para transferir la salida de cada grupo 214A-214N a cualquier unidad de partición 220A-220N o a 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 mediante la barra transversal de memoria 216 para ser objeto de lectura o de escritura en diversos dispositivos externos de memoria. En una forma de 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 procesadores en paralelo 222, lo que permite que las unidades de procesamiento, dentro de los diferentes grupos de procesamiento 214A-214N, se comuniquen con la memoria del sistema u otra memoria que no sea local a la unidad de procesamiento en paralelo 202. En una forma de realización, la barra transversal de memoria 216 puede utilizar canales virtuales para separar flujos de tráfico entre los grupos 214A-214N y las unidades de partición 220A-220N.

Aunque se ilustra una sola instancia de la unidad de procesamiento en paralelo 202 dentro del procesador en paralelo 200, se puede incluir cualquier número de instancias de la unidad de procesamiento en paralelo 202. Por ejemplo, se pueden proporcionar múltiples instancias de la unidad de procesamiento en paralelo 202 en una sola tarjeta complementaria, o se pueden interconectar múltiples tarjetas complementarias. Las diferentes instancias de la unidad de procesamiento en paralelo 202 pueden configurarse para inter-operar entre sí incluso si las diferentes instancias tienen diferentes números de núcleos de procesamiento, diferentes cantidades de memoria de procesadores en 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 coma flotante de mayor precisión en relación con otras instancias. Los sistemas que incorporan una o más instancias de la unidad de procesamiento en paralelo 202 o el procesador en paralelo 200 se pueden poner en práctica en una diversidad de configuraciones y factores de forma, que incluyen, sin limitación, ordenadores de escritorio, portátiles u ordenador de bolsillo, servidores, estaciones de trabajo, consolas de juegos, y/o sistemas incorporados.

La Figura 2B es un diagrama de bloques de una unidad de partición 220, según una forma de realización. En una forma de realización, la unidad de partición 220 es una instancia de una de las unidades de partición 220A-220N de la Figura 2A. Tal como se ilustra, la unidad de partición 220 incluye una memoria caché L2 221, una interfaz de memoria intermedia de tramas 225 y una ROP 226 (unidad de operaciones de trama). La memoria caché L2 221 es una memoria caché de lectura/escritura que está configurada para realizar operaciones de carga y de almacenamiento recibidas desde la barra transversal de memoria 216 y ROP 226. Los errores de lectura y las solicitudes urgentes de reescritura son enviadas por la memoria caché L2 221 a la interfaz de memoria intermedia de tramas 225 para su procesamiento. Las actualizaciones también se pueden enviar a la memoria intermedia de tramas mediante la interfaz de memoria intermedia de tramas 225 para su procesamiento. En una forma de realización, la interfaz de memoria intermedia de tramas 225 interactúa con una de las unidades de memoria en la memoria de procesadores en paralelo, tal como las unidades de memoria 224A-224N de la Figura 2 (por ejemplo, dentro de la memoria de procesadores en paralelo 222).

En aplicaciones gráficas, la ROP 226 es una unidad de procesamiento que realiza operaciones de trama tales como estarcido, prueba z, combinación y similares. A continuación, la unidad ROP 226 emite datos gráficos procesados que se almacenan en la memoria de gráficos. En algunas formas de realización, la unidad ROP 226 incluye lógica de compresión para comprimir datos de profundidad o de color que son objeto de escritura en la memoria y descomprimir datos de profundidad o de color que son objeto de lectura de la memoria. La lógica de compresión puede ser una lógica de compresión sin pérdidas que hace uso de uno o más de múltiples algoritmos de compresión. El tipo de compresión que realiza la unidad ROP 226 puede variar según las características estadísticas de los datos que se van a comprimir. Por ejemplo, en una forma de realización, la compresión de color delta se realiza en datos de profundidad y de color sobre una base de por mosaico.

En algunas formas de realización, la unidad ROP 226 se incluye 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 dicha forma de realización, las solicitudes de lectura y de escritura de datos de píxeles se transmiten mediante la barra transversal de memoria

216 en lugar de datos de fragmentos de píxeles. Los datos gráficos procesados pueden mostrarse en un dispositivo de visualización, tal como uno o más dispositivos de visualización 110 de la Figura 1, enrutado para su posterior procesamiento por el(los) procesador(es) 102, o enrutado para su posterior procesamiento por una de las entidades de procesamiento dentro del procesador en 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 en paralelo, según una forma de realización. En una forma de realización, el grupo de procesamiento es una instancia de uno de los grupos de procesamiento 214A-214N de la Figura 2. El grupo de procesamiento 214 se puede configurar para ejecutar múltiples subprocesos en paralelo, en donde el término "subproceso" se refiere a una instancia de un programa particular que se ejecuta en un conjunto particular de datos de entrada. En algunas formas de realización, se utilizan técnicas de emisión de instrucciones de instrucción única, datos múltiples (SIMD) para soportar la ejecución paralela de un gran número de subprocesos sin proporcionar múltiples unidades de instrucción independientes. En otras formas de realización, se utilizan técnicas de instrucción única, subprocesos múltiples (SIMT) para admitir la ejecución en paralelo de un gran número de subprocesos, por lo general sincronizados, utilizando una unidad de instrucción común configurada para emitir instrucciones a un conjunto de motores de procesamiento dentro de cada uno de los grupos de procesamiento. A diferencia de un régimen de ejecución SIMD, en donde todos los motores de procesamiento suelen ejecutar instrucciones idénticas, la ejecución SIMT permite que diferentes subprocesos sigan más fácilmente rutas de ejecución divergentes mediante un programa de subproceso determinado. Los expertos en esta técnica comprenderán 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 mediante un gestor de canalización 232 que distribuye las tareas de procesamiento a los procesadores en paralelo SIMT. El gestor de canalización 232 recibe instrucciones del planificador 210 de la Figura 2 y gestiona la ejecución de esas instrucciones mediante un multiprocesador de gráficos 234 y/o una unidad de textura 236. El multiprocesador de gráficos ilustrado 234 es un ejemplo de un procesador en paralelo SIMT. Sin embargo, se pueden incluir diversos tipos de procesadores en paralelo SIMT de diferentes arquitecturas dentro del grupo de procesamiento 214. Una o más instancias del multiprocesador de gráficos 234 se pueden incluir dentro de un grupo de procesamiento 214. El multiprocesador de gráficos 234 puede procesar datos y una barra transversal de datos 240 se puede utilizar para distribuir los datos procesados a uno de los múltiples destinos posibles, incluyendo otras unidades de sombreadores. El gestor de canalización 232 puede facilitar la distribución de datos procesados especificando destinos para que los datos procesados se distribuyan 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 lógicas aritméticas que tienen circuitos sumadores, unidades de carga/almacenamiento, etc.). La lógica de ejecución funcional se puede configurar de manera canalizada en donde se pueden emitir nuevas instrucciones antes de que se completen las instrucciones anteriores. La lógica de ejecución funcional admite una diversidad de operaciones que incluyen aritmética de números enteros y de coma flotante, operaciones de comparación, operaciones booleanas, desplazamiento de bits y cálculo de diversas funciones algebraicas. En una forma de realización, se puede aprovechar el mismo hardware de unidad funcional para realizar diferentes operaciones y puede estar presente cualquier combinación de unidades funcionales.

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

En una forma de realización, el multiprocesador de gráficos 234 incluye una memoria caché interna para realizar operaciones de carga y de almacenamiento. En una forma de realización, el multiprocesador de gráficos 234 puede prescindir de una memoria caché interna y utilizar una memoria caché (por ejemplo, memoria caché L1 308) dentro del grupo de procesamiento 214. Cada multiprocesador de gráficos 234 también tiene acceso a memorias caché L2 dentro de las unidades de partición (por ejemplo, unidades de partición 220A-220N de la Figura 2) que se comparten entre todos los grupos de procesamiento 214 y se pueden utilizar para transmitir datos entre subprocesos. El multiprocesador de gráficos 234 también puede acceder a la memoria global fuera del circuito integrado, que puede incluir una o más memorias de procesadores en paralelo locales y/o memorias del sistema. Cualquier memoria externa a la unidad de procesamiento en paralelo 202 puede utilizarse como memoria global. Las formas de realización 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 memoria caché L1 308.

Cada grupo de procesamiento 214 puede incluir una 26U 245 (unidad de gestión de memoria) que está configurada para asignar direcciones virtuales en direcciones físicas. En otras formas de realización, una o más instancias de la unidad 26U 245 pueden residir dentro de la interfaz de memoria 218 de la Figura 2. La unidad 26U 245 incluye un conjunto de entradas de tabla de páginas (PTEs) que se utilizan para asignar una dirección virtual a una dirección física de un mosaico (obtener más información sobre el establecimiento de mosaico) y, de manera opcional, un índice de línea de memoria caché. La unidad 26U 245 puede incluir memorias intermedias de búsqueda de traslación de direcciones (TLB) o memorias caché que pueden residir dentro del multiprocesador de gráficos 234 o en la memoria caché L1 o en el grupo de procesamiento 214. La dirección física se procesa para distribuir la ubicación de acceso a datos de superficie para permitir la intercalación eficiente de solicitudes entre unidades de partición. El índice de línea de memoria caché se puede utilizar para determinar si una solicitud de una línea de memoria caché es un acierto o un error.

En las aplicaciones informáticas y de gráficos, un grupo de procesamiento 214 puede configurarse de manera que cada multiprocesador de gráficos 234 esté acoplado a una unidad de textura 236 para realizar operaciones de asignación de texturas, por ejemplo, determinar posiciones de muestra de texturas, realizar la lectura de datos de texturas y filtrar los datos de texturas. Los datos de texturas son objeto de lectura desde una memoria caché L1 de textura interna (no ilustrada) o en algunas formas de realización desde la memoria caché L1 dentro del multiprocesador de gráficos 234 y se extraen desde una memoria caché L2, memoria de procesador en paralelo local o memoria del sistema, según sea necesario. Cada multiprocesador de gráficos 234 envía tareas procesadas a la barra transversal de datos 240 para proporcionar la tarea procesada a otro grupo de procesamiento 214 para su posterior procesamiento o para almacenar la tarea procesada en una memoria caché L2, memoria de procesador en paralelo local o memoria del sistema mediante la barra transversal de memoria 216. Una unidad 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 tal como se describe en este 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 traslaciones de direcciones.

Se apreciará que la arquitectura central aquí descrita es ilustrativa y que son posibles variaciones y modificaciones. Cualquier número de unidades de procesamiento, por ejemplo, multiprocesador de gráficos 234, unidades de textura 236, preROPs 242, etc., puede incluirse dentro de un grupo de procesamiento 214. Además, aunque solamente se muestra un grupo de procesamiento 214, una unidad de procesamiento paralela tal como se describe en este documento puede incluir cualquier número de instancias del grupo de procesamiento 214. En una forma de realización, cada grupo de procesamiento 214 se puede configurar para operar de manera independiente respecto a otros grupos de procesamiento 214 utilizando unidades de procesamiento separadas y distintas, memorias caché L1, etc.

La Figura 2D muestra un multiprocesador de gráficos 234, según una forma de realización. En tal forma de realización, el multiprocesador de gráficos 234 se acopla con el gestor de canalización 232 del grupo de procesamiento 214. El multiprocesador de gráficos 234 tiene una canalización de ejecución que incluye, pero no se limita a, una memoria caché de instrucciones 252, una unidad de instrucción 254, una unidad de asignación de direcciones 256, un fichero de registro 258, uno o más núcleos de unidades de procesamiento de gráficos de uso general (GPGPU) 262 y una o más unidades de carga/almacenamiento 266. Los núcleos de GPGPU 262 y las unidades de carga/almacenamiento 266 se acoplan con una memoria caché 272 y una memoria compartida 270 mediante una interconexión de memoria compartida y memoria caché 268.

En una forma de realización, la memoria caché de instrucciones 252 recibe un flujo de instrucciones para ejecutar desde el gestor de canalización 232. Las instrucciones se almacenan en la memoria caché de instrucciones 252 y se envían para su ejecución por la unidad de instrucción 254. La unidad de instrucción 254 puede enviar instrucciones como grupos de subprocesos (por ejemplo, tejidos funcionales), con cada subproceso del grupo de subprocesos asignado a una unidad de ejecución diferente dentro del núcleo GPGPU 262. Una instrucción puede acceder a cualquier espacio de direcciones local, compartido o global especificando una dirección dentro de un espacio de dirección unificado. La unidad de asignación de direcciones 256 puede utilizarse para trasladar 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 fichero de registro 258 proporciona un conjunto de registros para las unidades funcionales del multiprocesador de gráficos 234. El fichero 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 234. En una forma de realización, el fichero de registro 258 se divide entre cada una de las unidades funcionales de manera que a cada unidad funcional se le asigna una parte dedicada del fichero de registro 258. En una forma de realización, el fichero de registro 258 es dividido entre los diferentes tejidos funcionales que está ejecutando el multiprocesador de gráficos 234.

Cada uno de los núcleos GPGPU 262 puede incluir unidades de coma flotante (FPUs) y/o unidades lógicas aritméticas (ALUs) de enteros que se utilizan para ejecutar instrucciones del multiprocesador de gráficos 234. Las ALUs pueden incluir circuitos sumadores (por ejemplo, sistema 700, circuito sumador 750) de conformidad con formas de realización

del presente diseño. Los núcleos GPGPU 262 pueden tener una arquitectura similar o pueden diferir en la arquitectura, según las formas de realización. Por ejemplo, y en una forma de realización, una primera parte de los núcleos de GPGPU 262 incluye una FPU de precisión simple y una ALU de número entero, mientras que una segunda parte de los núcleos de GPGPU incluye una FPU de precisión doble. En una forma de realización, las FPUs pueden poner en práctica la norma I18E 754-2008 para aritmética de coma flotante o permitir aritmética de coma flotante de precisión variable. El multiprocesador de gráficos 234 puede incluir de manera adicional una o más unidades de funciones especiales o de funciones fijas para realizar funciones específicas tales como operaciones de copia de rectángulos o combinación de píxeles. En una forma de realización, uno o más de los núcleos de GPGPU también pueden incluir una lógica de función fija o especial.

En una forma de realización, los núcleos GPGPU 262 incluyen lógica SIMD capaz de realizar una única instrucción en múltiples conjuntos de datos. En una forma de realización, los núcleos GPGPU 262 pueden ejecutar físicamente las instrucciones SIMD4, SIMD8 y SIMD16 y lógicamente ejecutar las instrucciones SIMD1, SIMD2 y SIMD32. Las instrucciones SIMD para los núcleos GPGPU pueden generarse en el momento de la compilación mediante un compilador de sombreador o generarse de manera automática al ejecutar programas escritos y compilados para arquitecturas de SIMT o de datos múltiples de programa único (SPMD). Múltiples subprocesos de un programa configurado para el modelo de ejecución SIMT pueden ejecutarse mediante una única instrucción SIMD. Por ejemplo, y en una forma de realización, ocho subprocesos SIMT que realizan operaciones iguales o similares pueden ejecutarse en paralelo mediante una única unidad lógica SIMD8.

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

Las Figuras 3A-3B ilustran multiprocesadores de gráficos adicionales, según formas de realización. 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 pueden configurarse como un multiprocesador de flujo continuo (SM) capaz de ejecutar, de manera simultánea, un gran número de subprocesos de ejecución.

La Figura 3A muestra un multiprocesador de gráficos 325 según una forma de realización adicional. El multiprocesador de gráficos 325 incluye múltiples instancias adicionales de unidades de recursos de ejecución en relación con el 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 fichero 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 cálculo (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 forma de realización, las unidades de recursos de ejecución tienen una memoria caché de instrucciones 330 común, una memoria caché de textura y/o datos 342 y una memoria compartida 346.

Los diversos componentes pueden comunicarse mediante un tejido funcional de interconexión 327. En una forma de realización, el tejido funcional de interconexión 327 incluye uno o más conmutadores de barra cruzada para permitir la comunicación entre los diversos componentes del multiprocesador de gráficos 325. En una forma de realización, el tejido funcional de interconexión 327 es una capa de tejido funcional de red de alta velocidad separada sobre la que se apila cada componente del multiprocesador de gráficos 325. Los componentes del multiprocesador de gráficos 325 se comunican con componentes distantes mediante el tejido funcional de interconexión 327. Por ejemplo, los núcleos GPGPU 336A-336B, 337A-337B y 3378A-338B pueden comunicarse cada uno con la memoria compartida 346 mediante el tejido funcional de interconexión 327. El tejido funcional de interconexión 327 puede arbitrar la comunicación dentro del multiprocesador de gráficos 325 para asegurar una asignación justa de ancho de banda entre los componentes.

La Figura 3B muestra un multiprocesador de gráficos 350 según una forma de realización adicional. El procesador de gráficos incluye múltiples conjuntos de recursos de ejecución 356A-356D, en donde cada conjunto de recursos de ejecución incluye múltiples unidades de instrucción, ficheros de registro, núcleos GPGPU y unidades de carga/almacenamiento, tal como se ilustra en la Figura 2D y en la Figura 3A. Los recursos de ejecución 356A-356D pueden trabajar 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 forma de realización, los recursos de ejecución 356A-356D pueden compartir una memoria caché de instrucciones 354 y una memoria compartida 362, así como múltiples instancias de memoria caché de datos y/o textura 358A-358B. Los diversos componentes pueden comunicarse mediante un tejido funcional de interconexión 352 similar al tejido funcional 327 de la Figura 3A.

Los expertos en esta técnica comprenderán 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 formas de realización. Por lo tanto, las técnicas aquí descritas pueden ponerse en práctica en cualquier unidad de procesamiento configurada correctamente, incluyendo, sin limitación, uno o más procesadores de aplicaciones móviles, una o más unidades de procesamiento central (CPUs) de escritorio o servidor, incluyendo las CPUs de multinúcleos, una o más unidades de procesamiento en paralelo, tales como la unidad de procesamiento en paralelo 202 de la Figura 2, así como uno o más procesadores de gráficos o unidades de procesamiento de uso especial, sin apartarse del alcance de las formas de realización descritas en este documento.

En algunas formas de realización, un procesador en paralelo o GPGPU tal como se describe en el presente documento se acopla de manera comunicativa a los núcleos del procesador/host para acelerar las operaciones gráficas, las operaciones de aprendizaje máquina, las operaciones de análisis de patrones y diversas funciones de GPU de uso general (GPGPU). La GPU se puede acoplar de manera comunicativa a los núcleos del procesador/host mediante un bus u otra interconexión (por ejemplo, una interconexión de alta velocidad tal como PCIe o NVLink). En otras formas de realización, la GPU puede integrarse en el mismo paquete o circuito integrado que los núcleos y acoplarse de manera comunicativa a los núcleos mediante un bus/interconexión de procesador interno (es decir, interno al paquete o circuito integrado), con independencia de la forma en que se conecta la GPU, los núcleos del procesador pueden asignar trabajo a la GPU en forma de secuencias de comandos/instrucciones contenidas en un descriptor de trabajo. A continuación, la GPU utiliza circuitos/lógica dedicados para procesar, de manera eficiente, estos comandos/instrucciones.

Técnicas para la GPU para la interconexión del procesador/host

La Figura 4A ilustra una arquitectura, a modo de ejemplo, en donde una pluralidad de GPUs 410-413 se acoplan de manera comunicativa a una pluralidad de procesadores multinúcleo 405-406 mediante enlaces de alta velocidad 440-443 (por ejemplo, buses, interconexiones punto a punto, etc.). En una forma de realización, los enlaces de alta velocidad 440-443 admiten un rendimiento de comunicación de 4 GB/s, 30 GB/s, 80 GB/s o superior, según la puesta en práctica. Se pueden utilizar diversos protocolos de interconexión, incluyendo, sin limitación, PCIe 4.0 o 5.0 y NVLink 2.0. Sin embargo, los principios subyacentes de la invención no se limitan a ningún protocolo de comunicación o rendimiento en particular.

Además, en una forma de realización, dos o más de las GPUs 410-413 están interconectadas mediante enlaces de alta velocidad 444-445, que pueden ponerse en práctica utilizando los mismos o diferentes protocolos/enlaces que los utilizados para los enlaces de alta velocidad 440-443. De manera similar, dos o más de los procesadores multinúcleo 405-406 se pueden conectar mediante un enlace de alta velocidad 433 que pueden ser buses de multiprocesador simétrico (SMP) que funcionan a 20 GB/s, 30 GB/s, 120 GB/s o una velocidad superior. De manera alternativa, toda la comunicación entre los diversos componentes del sistema que se muestran en la Figura 4A puede lograrse utilizando los mismos protocolos/enlaces (por ejemplo, mediante un tejido funcional de interconexión común). Tal como se describió con anterioridad, sin embargo, 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 está acoplado, de manera comunicativa, a una memoria de procesador 401-402, mediante las interconexiones de memoria 430-431, respectivamente, y cada GPU 410-413 está acoplada, de manera comunicativa, a la memoria GPU 420-423 a través de las interconexiones de memoria GPU 450-453, respectivamente. Las interconexiones de memoria 430-431 y 450-453 pueden utilizar las mismas o diferentes tecnologías de acceso a la memoria. A modo de ejemplo, y sin limitación, las memorias del procesador 401-402 y las memorias de la GPU 420-423 pueden ser memorias volátiles tal como las Memorias de Acceso Aleatorio Dinámicas (DRAMs) (incluyendo las Memorias DRAMs apiladas), Memoria de Gráficos 17R SDRAM (G17R) (por ejemplo, G17R5, G17R6) o Memoria de Alto Ancho de Banda (HBM) y/o pueden ser memorias no volátiles tal como 3D XPoint o Nano-Ram. En una forma de realización, una parte de las memorias puede ser una memoria volátil y otra parte puede ser una memoria no volátil (por ejemplo, utilizando una jerarquía de memoria de dos niveles (2LM)).

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

La Figura 4B ilustra datos adicionales para una interconexión entre un procesador multinúcleo 407 y un módulo de aceleración de gráficos 446 de conformidad con una forma de realización. El módulo de aceleración de gráficos 446 puede incluir uno o más circuitos integrados GPU incorporados en una tarjeta de línea que está acoplada al procesador 407 mediante el enlace de alta velocidad 440. De manera alternativa, el módulo de aceleración de gráficos 446 puede estar integrado en el mismo paquete o circuito integrado que el procesador 407.

El procesador 407 ilustrado incluye una pluralidad de núcleos 460A-460D, cada uno con una memoria intermedia de búsqueda de traslación 461A-461D y una o más memorias caché 462A-462D. Los núcleos pueden incluir diversos 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 reordenación, etc.). Las memorias caché 462A-462D pueden comprender memoria caché de nivel 1 (L1) y 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 los núcleos 460A-460D. Por ejemplo, una forma de realización del procesador 407 incluye 24 núcleos, cada uno con su propia memoria caché L1, doce memorias caché L2 compartidas y doce memorias caché L3 compartidas. En esta forma de realización, dos núcleos adyacentes comparten una de las memorias caché L2 y L3. El procesador 407 y el módulo de integración del acelerador de gráficos 446 se conectan con la memoria del sistema 441, que puede incluir las memorias del procesador 401-402.

La coherencia se mantiene para los datos y las instrucciones almacenadas en las diversas memorias caché 462A-462D, 456 y la memoria del sistema 441 mediante la comunicación entre núcleos a través de un bus de coherencia 464. Por ejemplo, cada memoria caché puede tener una lógica/circuitos de coherencia de memoria caché asociados con ella para comunicarse mediante un bus de coherencia 464 en respuesta a lecturas o escrituras detectadas en líneas de memoria caché particulares. En una puesta en práctica, se realiza un protocolo de espionaje de memoria caché sobre el bus de coherencia 464 para espiar los accesos a la memoria caché. Las técnicas de indagación/coherencia de memoria caché son bien conocidas por los expertos en esta técnica y no se describirán aquí en detalle para evitar oscurecer los principios subyacentes de la invención.

En una forma de realización, un circuito proxy 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 memoria caché como un homólogo de los núcleos. En particular, una interfaz 435 proporciona conectividad al circuito proxy 425 mediante un 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 puesta en práctica, un circuito de integración de acelerador 436 proporciona servicios de gestión de memoria caché, acceso a memoria, gestión de contexto y gestión de interrupciones en función de una pluralidad de motores de procesamiento de gráficos 431, 432, N del módulo de aceleración de gráficos 446. Los motores de procesamiento de gráficos 431, 432, N pueden comprender, cada uno, una unidad de procesamiento gráfico (GPU) separada. De manera alternativa, los motores de procesamiento de gráficos 431, 432, N pueden comprender diferentes tipos de motores de procesamiento de gráficos dentro de una GPU, tales como unidades de ejecución de gráficos, motores de procesamiento de medios (por ejemplo, codificadores/decodificadores de vídeo), muestreadores y motores *blit*. Dicho de otro modo, 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 unidades GPUs individuales integradas en un paquete común, tarjeta de línea o circuito integrado.

En una forma de realización, el circuito de integración del acelerador 436 incluye una unidad de gestión de memoria (26U) 439 para realizar diversas funciones de gestión de memoria tales como traslaciones de memoria virtual a física (también denominadas traslaciones de memoria efectiva a real) y protocolos de acceso a la memoria para acceder a la memoria del sistema 441. La unidad 26U 439 también puede incluir una memoria intermedia de búsqueda de traslación (TLB) (no ilustrada) para almacenar en memoria caché las traslaciones de direcciones virtuales/efectivas a físicas/reales. En una puesta en práctica, una memoria caché 438 almacena comandos y datos para un acceso eficiente por parte de los motores de procesamiento de gráficos 431-432, N. En una forma de realización, los datos almacenados en la memoria caché 438 y las memorias de gráficos 433-434, N se mantienen coherentes con las memorias caché centrales 462A-462D, 456 y la memoria del sistema 411. Tal como se describió con anterioridad, lo que antecede puede lograrse mediante un circuito proxy 425 que participa en el mecanismo de coherencia de la memoria caché en función de la memoria caché 438 y de las memorias 433-434, N (por ejemplo, enviando actualizaciones a la memoria caché 438 relacionadas con modificaciones/accesos de líneas de memoria caché en las memorias caché del procesador 462A-462D, 456 y recibir actualizaciones desde la memoria caché 438).

Un conjunto de registros 445 almacena datos de contexto para subprocesos ejecutados por los motores de procesamiento de gráficos 431-432, N y un circuito de gestión de contexto 448 gestiona los contextos de subproceso. Por ejemplo, el circuito de gestión de contexto 448 puede realizar operaciones de guardar y restaurar para guardar y restaurar contextos de diversos subprocesos durante los cambios de contexto (por ejemplo, cuando se guarda un primer subproceso y se almacena un segundo subproceso para que el segundo subproceso pueda ser ejecutado por un motor de procesamiento de gráficos). Por ejemplo, en un conmutador de contextos, el circuito de gestión de

contexto 448 puede almacenar valores de registro actuales en una zona designada en la memoria (por ejemplo, identificada por un puntero de contexto). A continuación, puede restaurar los valores de registro al volver al contexto. En una forma de realización, un circuito de gestión de interrupciones 447 recibe y procesa las interrupciones recibidas desde los dispositivos del sistema.

En una puesta en práctica, las direcciones virtuales/efectivas de un motor de procesamiento de gráficos 431 se trasladan a direcciones reales/físicas en la memoria del sistema 411 por la unidad 26U 439. Una forma de realización del circuito de integración del acelerador 436 admite múltiples (por ejemplo, 4, 8, 16) módulos aceleradores de gráficos 446 y/u otros dispositivos aceleradores. El módulo acelerador de gráficos 446 puede estar dedicado a una sola aplicación ejecutada en el procesador 407 o puede compartirse entre múltiples aplicaciones. En una forma de realización, se presenta un entorno de ejecución de gráficos virtualizados en donde los recursos de los motores de procesamiento de gráficos 431-432, N se comparten con múltiples aplicaciones o máquinas virtuales (VMs). Los recursos pueden subdividirse en "segmentos" que se asignan a diferentes VMs y/o aplicaciones en función de los requisitos de procesamiento y las prioridades asociadas con las VMs y/o las aplicaciones.

Por lo tanto, el circuito de integración del acelerador actúa como un puente al sistema para el módulo de aceleración de gráficos 446 y proporciona traslación de direcciones y servicios de memoria caché del sistema. Además, el circuito de integración del acelerador 436 puede proporcionar instalaciones de virtualización para que el procesador host gestione la virtualización de los motores de procesamiento de gráficos, las interrupciones y la gestión de la 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 host 407, cualquier procesador host puede direccionar estos recursos directamente utilizando un valor de dirección efectivo. Una función del circuito de integración del acelerador 436, en una forma de realización, es la separación física de los motores de procesamiento de gráficos 431-432, N para que aparezcan ante el sistema como unidades independientes.

Tal como se describió con anterioridad, en la forma de realización ilustrada, una o más memorias de gráficos 433-434, M están acopladas a cada uno de los motores de procesamiento de gráficos 431-432, N, respectivamente. Las memorias de gráficos 433-434, M almacenan instrucciones y datos que se procesan por cada uno de los motores de procesamiento de gráficos 431-432, N. Las memorias de gráficos 433-434, M pueden ser memorias volátiles tales como memorias DRAMs (incluyendo memorias DRAMs apiladas), memoria G17R (p. ej., G17R5, G17R6), o HBM, y/o pueden ser memorias no volátiles tales como 3D XPoint o Nano-Ram.

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

La Figura 4C ilustra otra forma de realización en donde el circuito de integración del acelerador 436 está integrado dentro del procesador 407. En esta forma de realización, los motores de procesamiento de gráficos 431-432, N se comunican directamente mediante enlace de alta velocidad 440 con el circuito de integración del acelerador 436 mediante la interfaz 437 e interfaz 435 (que, de nuevo, puede utilizar cualquier forma de bus o de 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 a un mayor rendimiento dada su proximidad al bus de coherencia 462 y a las memorias caché 462A-462D, 426.

Una forma de realización admite diferentes modelos de programación que incluyen 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 que son controlados por el circuito de integración del acelerador 436 y modelos de programación que son controlados por el módulo de aceleración de gráficos 446.

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

En los modelos de programación de proceso dedicado, los motores de procesamiento de gráficos 431-432, N, pueden ser compartidos por múltiples particiones de aplicación/VM. Los modelos compartidos requieren un hipervisor del sistema para virtualizar 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 poseídos por el 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 utilizando un identificador de proceso. En una forma de realización, los elementos de proceso se almacenan en la memoria del sistema 411 y se pueden direccionar utilizando las técnicas de traslación de dirección efectiva a dirección real descritas en este documento. El identificador del proceso puede ser un valor específico de la puesta en práctica proporcionado al proceso anfitrión al registrar su contexto con el motor de procesamiento de gráficos 431-432, N (es decir, solicitar al software del sistema que agregue el elemento del proceso a la lista vinculada del elemento del proceso). Los 16 bits inferiores del identificador de proceso pueden ser el desplazamiento del elemento de proceso dentro de la lista enlazada de elementos de proceso.

La Figura 4D ilustra un segmento de integración de acelerador 490 a modo de ejemplo. Tal como se aquí utiliza, un "segmento" comprende una parte específica de los recursos de procesamiento del circuito de integración de acelerador 436. El espacio de dirección efectiva de la aplicación 482 dentro de la memoria del sistema 411 almacena elementos de proceso 483. En una forma de realización, los elementos de proceso 483 se almacenan en respuesta a las invocaciones de GPU 481 de las aplicaciones 480 ejecutadas en el procesador 407. Un elemento de proceso 483 contiene el estado del proceso para la aplicación correspondiente 480. Un descriptor de trabajo (WD) 484 contenido en el elemento de proceso 483 puede ser un trabajo único 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 solicitudes de trabajo en el espacio de direcciones de la aplicación 482.

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

En una puesta en práctica, el modelo de programación de procesos dedicados es específico de la puesta en práctica. 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 poseído por un proceso único, el hipervisor inicializa el circuito de integración del acelerador 436 para la partición propietaria y el sistema operativo inicializa el circuito de integración del acelerador 436 para el proceso de propiedad en el momento en que se asigna el módulo de aceleración de gráficos 446.

En funcionamiento, una unidad de búsqueda de WD 491 en el segmento de integración del acelerador 490 busca el siguiente WD 484 que incluye una indicación del trabajo que debe realizar uno de los motores de procesamiento de gráficos del módulo de aceleración de gráficos 446. Los datos desde el WD 484 pueden almacenarse en registros 445 y utilizarse por la unidad 26U 439, el circuito de gestión de interrupciones 447 y/o el circuito de gestión de contexto 446 tal como se ilustra. Por ejemplo, una forma de realización de la unidad 26U 439 incluye un circuito de recorrido de segmento/página para acceder a las tablas de segmento/páginas 486, dentro del espacio de dirección virtual del sistema operativo OS 485, el circuito de gestión de interrupciones 447 puede procesar eventos de interrupción 492 recibidos desde el módulo de aceleración de gráficos 446. Cuando se realizan operaciones de gráficos una dirección efectiva 493, generada por un motor de procesamiento de gráficos 431-432, N, se traslada a una dirección real por la unidad 26U 439.

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

Tabla 1 - Registros inicializados del hipervisor

	Registro de control de segmentos
	Puntero de zona de procesos programados de dirección real (RA)
	Registro de anulación de máscara de autoridad
	Compensación de entrada de la tabla de vectores de interrupción
	Límite de entrada de la tabla de vectores de interrupción
	Registro de estado
	ID de partición lógica
	Puntero de registro de utilización del acelerador de hipervisor de dirección real (RA)
	Registro de Descripción del Almacenamiento

En la Tabla 2 se muestran ejemplos de registros que pueden ser inicializados por el sistema operativo.

Tabla 2 - Registros inicializados por el sistema operativo

	Identificación de procesos y subprocesos
	Puntero de guardar/restaurar de contexto de dirección efectiva (EA)
	Puntero de registro de utilización del acelerador de dirección virtual (VA)
	Puntero de tabla de segmento de almacenamiento de dirección virtual (VA)
	Máscara de autoridad
	Descriptor de trabajo

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

La Figura 4E ilustra datos adicionales para una forma de realización de un modelo compartido. Esta forma de realización incluye un espacio de direcciones reales de hipervisor 498 en donde se almacena una lista de elementos de proceso 499. Se puede acceder al espacio de direcciones reales del hipervisor 498 mediante un hipervisor 496 que virtualiza los motores del módulo de aceleración de gráficos para el sistema operativo 495.

Los modelos de programación compartidos permiten que todos, o un subconjunto de procesos de la totalidad o un subconjunto de particiones en el sistema, utilicen un módulo de aceleración de gráficos 446. Existen dos modelos de programación en donde el módulo de aceleración de gráficos 446 es compartido por múltiples procesos y particiones: compartidas en intervalos de tiempo y compartidas dirigidas por gráficos.

En este modelo, el hipervisor del sistema 496 posee el módulo de aceleración de gráficos 446 y hace que su función esté disponible para todos los sistemas operativos 495. Para que un módulo de aceleración de gráficos 446 admita la virtualización por parte del hipervisor del sistema 496, el módulo de aceleración de gráficos 446 puede 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 guardar y restaurar contextos. 2) El módulo de aceleración de gráficos 446 garantiza que la solicitud de trabajo de una aplicación se complete en un período de tiempo específico, incluyendo los errores de traslación, o el módulo de aceleración de gráficos 446 que proporciona la capacidad de adelantarse al procesamiento del trabajo. 3) El módulo de aceleración de gráficos 446 debe garantizar la equidad entre procesos cuando opera en el modelo de programación compartida dirigida.

En una forma de realización, para el modelo compartido, se requiere que la aplicación 480 realice una solicitud de llamada al sistema operativo 495 con un tipo de módulo de aceleración de gráficos 446, un descriptor de trabajo (WD), un valor de registro de máscara de autoridad (AMR) y un puntero de zona de guardar/restaurar contextos (CSRP). El tipo de módulo de aceleración de gráficos 446 describe la función de aceleración dirigida para la llamada al sistema. El tipo de módulo de aceleración de gráficos 446 puede ser un valor específico del sistema. El WD está formateado específicamente para el módulo de aceleración de gráficos 446 y puede tener la forma de un comando del módulo de aceleración de gráficos 446, un puntero de dirección efectiva a una estructura definida por el usuario, un puntero de dirección efectiva a una cola de comandos o cualquier otra estructura de datos para describir el trabajo a realizar por el módulo de aceleración de gráficos 446. En una forma de realización, el valor de AMR es el estado de AMR a utilizar para el proceso en curso. El valor pasado al sistema operativo es similar a una aplicación que configura el AMR. Si las puestas en práctica del circuito de integración del acelerador 436 y el módulo de aceleración de gráficos 446 no admiten un Registro de Anulación de Máscara de Autoridad De Usuario (UAMOR), el sistema operativo puede aplicar el valor UAMOR actual al valor AMR antes de pasar el AMR en la llamada del hipervisor. El hipervisor 496 puede aplicar, de manera opcional, el valor actual del Registro de Anulación de Máscara de Autoridad (AMOR) antes de colocar el AMR en el elemento de proceso 483. En una forma de realización, el CSRP es uno de los registros 445 que contiene la dirección efectiva de una zona en el espacio de direcciones de la aplicación 482 para que el módulo de aceleración de gráficos 446 proceda a guardar y restaurar el estado de contexto. Este puntero es opcional si no es necesario guardar ningún estado entre trabajos o cuando se sustituye un trabajo. La zona para guardar/restaurar contextos puede estar fijada en la memoria del sistema.

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

Tabla 3: Parámetros de llamada del sistema operativo al hipervisor

	Un descriptor de trabajo (WD)
--	-------------------------------

	Un valor de registro de máscara de autoridad (AMR) (potencialmente enmascarado)
	Un puntero de Zona para Guardar/Restaurar Contextos (CSRP) de Dirección Efectiva (EA)
	Un ID de proceso (PID) y un ID de subproceso opcional (TID)
	Un puntero de registro de utilización del acelerador (AURP) de dirección virtual (VA)
	La dirección virtual del puntero de la tabla de segmentos de almacenamiento (SSTP)
	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 utilizar el módulo de aceleración de gráficos 446. El hipervisor 496, a continuación, coloca el elemento de proceso 483 en la lista vinculada de elementos de proceso para el tipo del módulo de aceleración de gráficos 446 correspondiente. El elemento de proceso puede incluir la información que se muestra en la Tabla 4.

Tabla 4 - Información de elementos de proceso

	Un descriptor de trabajo (WD)
	Un valor de registro de máscara de autoridad (AMR) (potencialmente enmascarado)
	Un puntero de Zona para Guardar/Restaurar Contextos (CSRP) de Dirección Efectiva (EA)
	Un ID de proceso (PID) y un ID de subproceso opcional (TID)
	Un puntero de registro de utilización del acelerador (AURP) de dirección virtual (VA)
	La dirección virtual del puntero de la tabla de segmentos de almacenamiento (SSTP)
	Un número de servicio de interrupción lógica (LISN)
	Tabla de vectores de interrupción, derivada de los parámetros de llamada del hipervisor.
	Un valor de registro de estado (SR)
0	Un ID de partición lógica (LPID)
1	Un puntero de registro de utilización del acelerador del hipervisor de dirección real (RA)
2	El registro descriptor de almacenamiento (SDR)

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

Tal como se ilustra en la Figura 4F, una forma de realización de la invención emplea una memoria unificada direccionable mediante un espacio de direcciones de memoria virtual común utilizado para acceder a las memorias 401-402 del procesador físico y a las memorias 420-423 de la GPU. En esta puesta en práctica, las operaciones ejecutadas en las unidades GPUs 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 forma de realización, una primera parte del espacio de direcciones virtual/efectivo se asigna a la memoria del procesador 401, una segunda parte a la segunda memoria del procesador 402, una tercera parte a la memoria GPU 420, y así sucesivamente. Todo el espacio de memoria virtual/efectivo (a veces denominado espacio de dirección efectiva) se distribuye por lo tanto en cada una de las memorias del procesador 401-402 y las memorias de GPU 420-423, lo que permite que cualquier procesador o GPU acceda a cualquier memoria física con una dirección virtual asignada a esa memoria.

En una forma de realización, el circuito de gestión de polarización/coherencia 494A-494E, dentro de una o más de las unidades 26Us 439A-439E, garantiza la coherencia de memoria caché entre las memorias caché de los procesadores host (por ejemplo, 405) y las GPUs 410-413 y pone en práctica técnicas de polarización indicando las memorias físicas en las que deben almacenarse determinados tipos de datos. Mientras que múltiples instancias de circuitos de gestión de polarización/coherencia 494A-494E se ilustran en la Figura 4F, el circuito de polarización/coherencia puede ponerse

en práctica dentro de la unidad 26U de uno o más procesadores host 405 y/o dentro del circuito de integración del acelerador 436.

Una forma de realización permite asignar la memoria 420-423 adjunta a la GPU como parte de la memoria del sistema y acceder a la misma utilizando la tecnología de memoria virtual compartida (SVM), pero sin sufrir los inconvenientes de rendimiento típicos asociados con la coherencia de la memoria caché del sistema completo. La capacidad de acceder a la memoria adjunta a la GPU 420-423 como memoria del sistema sin sobrecarga de coherencia de memoria caché onerosa proporciona un entorno operativo beneficioso para la descarga de GPU. Esta disposición permite que el software del procesador host 405 configure operandos y acceda a los resultados de los cálculos, sin la sobrecarga de las copias de datos DMA de E/S tradicionales. Dichas copias tradicionales implican llamadas de controlador, interrupciones y accesos de E/S asignados a la memoria (2610) que son todos ineficientes en relación con los accesos a memoria simples. Al mismo tiempo, la capacidad de acceder a la memoria adjunta de GPU 420-423 sin sobrecargas de coherencia de memoria caché puede ser fundamental para el tiempo de ejecución de un cálculo descargado. En casos con tráfico de memoria de escritura de transmisión sustancial, por ejemplo, la sobrecarga de coherencia de memoria caché, puede reducir de manera significativa el ancho de banda de escritura efectivo visto por una GPU 410-413. La eficiencia de la configuración del operando, la eficiencia del acceso a los resultados y la eficiencia del cálculo de la GPU juegan todas ellas un papel en la determinación de la efectividad de la descarga de la GPU.

En una puesta en práctica, la selección entre la polarización de la GPU y la polarización del procesador host está impulsada por una estructura de datos del seguidor de polarización. Se puede utilizar 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 adjunta a la GPU. La tabla de polarización puede ponerse en práctica en un margen de memoria extraída de una o más memorias 420-423 conectadas a GPU, con o sin memoria caché de polarización en la GPU 410-413 (p. ej., para almacenar en memoria caché las entradas de la tabla de polarización utilizadas con frecuencia/recientemente). De manera alternativa, toda la tabla de polarización puede mantenerse dentro de la GPU.

En una puesta en práctica, se accede a la entrada de la tabla de polarización asociada con cada acceso a la memoria 420-423 adjunta a la GPU antes del acceso real a la memoria de la GPU, lo que causa las siguientes operaciones. En primer lugar, las solicitudes locales de la GPU 410-413 que encuentran su página en polarización de GPU se reenvían directamente a una memoria de GPU 420-423 correspondiente. Las solicitudes locales de la GPU que encuentran su página en la polarización host se reenvían al procesador 405 (por ejemplo, mediante un enlace de alta velocidad tal como se describió con anterioridad). En una forma de realización, las solicitudes del procesador 405 que encuentran la página solicitada en la polarización del procesador host completan la solicitud como una lectura de memoria normal. De manera alternativa, las solicitudes dirigidas a una página con polarización de GPU pueden reenviarse a la GPU 410-413. A continuación, la GPU puede hacer la transición de la página a una polarización de procesador host si no está utilizando la página actualmente.

El estado de polarización 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 basado puramente en hardware.

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

En una forma de realización, la coherencia de la memoria caché se mantiene al hacer que el procesador host 405 no pueda almacenar en memoria caché, de manera temporal, las páginas polarizadas por GPU. Para acceder a estas páginas, el procesador 405 puede solicitar acceso a la GPU 410 que puede, o no, otorgar acceso de inmediato, dependiendo de la puesta en práctica. Por lo tanto, para reducir la comunicación entre el procesador 405 y la GPU 410, es conveniente asegurarse de que las páginas polarizadas por la GPU sean aquellas que se requieren por la GPU, pero no por el procesador host 405 y viceversa.

Canalización de procesamiento de gráficos

La Figura 5 ilustra una canalización de procesamiento de gráficos 500, según una forma de realización. En una forma de realización, un procesador de gráficos puede poner en práctica la canalización de procesamiento de gráficos 500 ilustrada. El procesador de gráficos puede incluirse dentro de los subsistemas de procesamiento en paralelo según aquí se describe, tal como el procesador en paralelo 200 de la Figura 2, que, en una forma de realización, es una variante del(de los) procesador(es) en paralelo(s) 112 de la Figura 1. Los diversos sistemas de procesamiento en paralelo pueden poner en práctica la canalización de procesamiento de gráficos 500 mediante una o más instancias de la unidad de procesamiento en paralelo (por ejemplo, la unidad de procesamiento en paralelo 202 de la Figura 2) tal como se describe en este documento. Por ejemplo, una unidad de sombreador (por ejemplo, el multiprocesador de

gráficos 234 de la Figura 3) puede configurarse para realizar las funciones de una o más de entre una unidad de procesamiento de vértices 504, una unidad de procesamiento de control de teselado 508, una unidad de procesamiento de evaluación de teselado 512, una unidad de procesamiento de geometría 516 y una unidad de procesamiento de fragmentos/píxeles 524. Las funciones del ensamblador de datos 502, ensambladores primitivos 506, 514, 518, unidad de teselado 510, rasterizador 522 y unidad de operaciones de rasterizado 526 también pueden ser realizadas por otros motores de procesamiento dentro de un grupo de procesamiento (p. ej., el grupo de procesamiento 214 de la Figura 3) y una unidad de partición correspondiente (p. ej., la unidad de partición 220A-220N de la Figura 2). La canalización de procesamiento de gráficos 500 también puede ponerse en práctica utilizando unidades de procesamiento dedicadas para una o más funciones. En una forma de realización, una o más partes de la canalización de procesamiento de gráficos 500 pueden realizarse mediante lógica de procesamiento en paralelo dentro de un procesador de uso general (por ejemplo, CPU). En una forma de realización, una o más partes de la canalización de procesamiento de gráficos 500 pueden acceder a la memoria en circuito integrado (por ejemplo, la memoria de procesadores en paralelo 222, tal como en la Figura 2) mediante una interfaz de memoria 528, que puede ser una instancia de la interfaz de memoria 218 de la Figura 2.

En una forma de realización, el ensamblador de datos 502 es una unidad de procesamiento que recoge datos de vértice para superficies y sistemas primitivos. El ensamblador de datos 502, a continuación, envía los datos de vértice, incluyendo los atributos de vértice, a la unidad de procesamiento de vértice 504. La unidad de procesamiento de vértice 504 es una unidad de ejecución programable que ejecuta programas de sombreador de vértice, iluminando y transformando datos de vértice según lo especificado por los programas de sombreador de vértice. La unidad de procesamiento de vértices 504 realiza la lectura de los datos que se almacenan en la memoria caché, local o del sistema para utilizar en el procesamiento de los datos de vértices y puede programarse para transformar los datos de vértices desde una representación de coordenadas basada en objetos a un espacio de coordenadas del espacio mundial o un espacio de coordenadas del dispositivo normalizado.

Una primera instancia de un ensamblador primitivo 506 recibe atributos de vértice desde la unidad de procesamiento de vértice 504. El ensamblador primitivo 506 realiza la lectura de los atributos de vértice almacenados según sea necesario y crea primitivos de gráficos para procesamiento mediante la unidad de procesamiento de control de teselado 508. Los primitivos de gráficos incluyen triángulos, segmentos de línea, puntos, parches, etc., compatibles con diversas interfaces de programación de aplicaciones (APIs) de procesamiento de gráficos.

La unidad de procesamiento de control de teselado 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 teselado 512. La unidad de procesamiento de control de teselado 508 también puede calcular factores de teselado para bordes de parches geométricos. Un factor de teselado se aplica a un único borde y cuantifica un nivel de detalle dependiente de la vista asociado con el borde. Una unidad de teselado 510 está configurada para recibir los factores de teselado para los bordes de un parche y para teselar el parche en múltiples primitivos geométricos tales como primitivos de línea, triángulo o cuadrilátero, que se transmiten a una unidad de procesamiento de evaluación de teselado 512. La unidad de procesamiento de evaluación de teselado 512 funciona sobre coordenadas parametrizadas del parche subdividido para generar una representación de superficie y atributos de vértice para cada vértice asociado con los primitivos geométricos.

Una segunda instancia de un ensamblador primitivo 514 recibe atributos de vértice desde la unidad de procesamiento de evaluación de teselado 512, realiza la lectura de los atributos de vértice almacenados, según sea necesario, y crea primitivos 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 sombreador de geometría para transformar los primitivos de gráficos recibidos desde el ensamblador de primitivos 514 según lo especificado por los programas de sombreador de geometría. En una forma de realización, la unidad de procesamiento de geometría 516 está programada para subdividir los primitivos de gráficos en uno o más primitivos de gráficos nuevos y calcular parámetros utilizados para rasterizar los primitivos de gráficos nuevos.

En algunas formas de realización, la unidad de procesamiento de geometría 516 puede agregar o eliminar elementos en el flujo de geometría. La unidad de procesamiento de geometría 516 envía los parámetros y vértices que especifican nuevos primitivos de gráficos al ensamblador de primitivos 518. El ensamblador de primitivos 518 recibe los parámetros y vértices desde la unidad de procesamiento de geometría 516 y crea primitivos de gráficos para su procesamiento por una unidad de recorte, selección y escala de ventana gráfica 520. La unidad de procesamiento de geometría 516 realiza la lectura de los datos que se almacenan en la memoria de los procesadores en paralelo o en la memoria del sistema para su uso en el procesamiento de los datos de geometría. La unidad de recorte, selección y escala de ventana gráfica 520 de ventana gráfica, realiza el recorte, la selección y el escalado de la ventana gráfica y envía primitivos de gráficos procesados a un rasterizador 522.

El rasterizador 522 puede realizar la selección de profundidad y otras optimizaciones basadas en profundidad. El rasterizador 522 también realiza la conversión de los primitivos de gráficos nuevos para generar fragmentos y enviar dichos fragmentos y los datos de cobertura asociados a la unidad de procesamiento de fragmentos/píxeles 524. La unidad de procesamiento de fragmentos/píxeles 524 es una unidad de ejecución programable que está configurada

para ejecutar programas de sombreador de segmentos o programas de sombreador de píxeles. La unidad de procesamiento de fragmentos/píxeles 524, que transforma fragmentos o píxeles recibidos desde el rasterizador 522, según lo especificado por los programas de sombreador de fragmentos o píxeles. Por ejemplo, la unidad de procesamiento de fragmentos/píxeles 524 puede programarse para realizar operaciones incluyendo, sin limitación, asignación de texturas, sombreado, combinación, corrección de texturas y corrección de perspectiva para obtener fragmentos sombreados o píxeles que se envían a una unidad de operaciones de trama 526. La unidad de procesamiento de fragmentos/píxeles 524 puede efectuar la lectura de datos que están almacenados en la memoria de procesadores en paralelo o en la memoria del sistema para su uso cuando se procesan los datos de fragmentos. Los programas de sombreador de fragmentos o píxeles se pueden configurar para sombrear en muestra, píxel, mosaico u otras granularidades según la frecuencia de muestreo configurada 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 tales como datos gráficos procesados para almacenarlos en la memoria de gráficos (por ejemplo, la memoria de procesadores en paralelo 222 tal como en la Figura 2, y/o la memoria del sistema para su uso cuando se procesan los datos de fragmentos. Los dispositivos de visualización 110 o para su posterior procesamiento por uno o más procesadores 102 o procesador(es) en paralelo(s) 112. En algunas formas de realización, la unidad de operaciones de trama 526 está configurada para comprimir z o datos de color que son objeto de escritura en la memoria y descomprimir z o datos de color que son objeto de lectura desde la memoria.

SISTEMAS Y MÉTODOS PARA GENERAR NÚMEROS ALEATORIOS GAUSSIANOS CON ACELERACIÓN POR HARDWARE

Las formas de realización del presente diseño proporcionan aceleración por hardware de generación de números aleatorios para aplicaciones de aprendizaje máquina y de aprendizaje de profundidad. En un ejemplo, la generación de números aleatorios gaussianos es importante para los métodos de aprendizaje máquina basados en bayesiano. Los números aleatorios gaussianos se pueden generar con un algoritmo generador de números aleatorios gaussianos (por ejemplo, sumando números aleatorios uniformes de conformidad con el teorema del límite central). Este proceso se acelera con circuitos sumadores en contraste con los enfoques convencionales para generar números aleatorios gaussianos. El circuito del sumador se puede personalizar para generar números aleatorios gaussianos.

La Figura 6 ilustra un método 600 para generar números aleatorios gaussianos con aceleración por hardware según una forma de realización. El método 600 se puede realizar mediante lógica de procesamiento que puede comprender hardware (por ejemplo, circuitos, lógica dedicada, lógica programable, etc.), software (tales como instrucciones que se ejecutan en un dispositivo de procesamiento) o una combinación de los mismos. En un ejemplo, al menos uno de entre un circuito sumador, una unidad lógica aritmética, un procesador, un multiprocesador de gráficos, un núcleo GPGPU, un grupo de cálculo informático y cualquier componente de hardware descrito en este documento realizan operaciones del método 600. Los procesos del método 600 se ilustran en secuencias lineales y también en paralelo para brevedad y claridad en la exposición; sin embargo, se contempla que cualquier número de ellos pueda realizarse en paralelo, de forma asíncrona o en diferentes órdenes.

El método 600 comienza en la operación 602 utilizando un circuito generador de números aleatorios uniformes (URNG) para generar números aleatorios uniformes para números aleatorios gaussianos. Los números aleatorios uniformes se pueden transmitir a cualquier sumador (por ejemplo, Sumador-1 de las operaciones 604 y 622, Sumador-2 de la operación 606, ... Sumador-N de las operaciones 608 y 624, etc.) del circuito sumador de un componente de hardware (por ejemplo, un procesador, un multiprocesador de gráficos, un núcleo GPGPU, un grupo de cálculo informático). En la operación 604, el Sumador-1 añade (suma) dos números aleatorios uniformes (por ejemplo, el primer y el segundo números aleatorios uniformes desde el circuito URNG) para generar una primera salida del Sumador-1. En la operación 606, el Sumador-2 añade (suma) otro número aleatorio uniforme (por ejemplo, el tercer número aleatorio uniforme del circuito URNG) con la primera salida para generar una primera salida de Sumador-2. Los sumadores posteriores (por ejemplo, Sumador-3 a Sumador-N-1, inclusive) luego continúan agregando números aleatorios uniformes adicionales a una salida de un sumador anterior hasta llegar a la operación 608. El Sumador-N en la operación 608 agrega (suma) otro número aleatorio uniforme (por ejemplo, N+1, número aleatorio uniforme del circuito URNG) con una primera salida de Sumador-N-1 para generar un primer número aleatorio gaussiano que se transmite a otra ubicación o se almacena en la memoria en la operación 610. N que puede ser cualquier valor entero.

En la operación 622, después de que el sumador-1 haya completado la operación 604, el sumador-1 puede agregar otros dos números aleatorios uniformes (por ejemplo, diferentes números aleatorios uniformes) para generar una segunda salida del sumador-1. Los sumadores posteriores (por ejemplo, Sumador-2 a Sumador-N-1) luego continúan agregando números aleatorios uniformes adicionales a una salida de un sumador anterior hasta llegar a la operación 624. El Sumador-N en la operación 624 agrega otro número aleatorio uniforme con una segunda salida de Sumador-N-1 para generar un segundo número aleatorio gaussiano que se transmite a otra ubicación o se almacena en la memoria en la operación 626. En un ejemplo, Sumador-1 a Sumador-N se utilizan para continuar agregando números aleatorios uniformes para generar números aleatorios gaussianos adicionales. En la operación 690, un número aleatorio gaussiano m-ésimo se transmite a otra ubicación o se almacena en la memoria. En un ejemplo, el método 600 procede de conformidad con el teorema del límite central.

La Figura 7 ilustra un diagrama de bloques de un sistema para generar números aleatorios gaussianos con aceleración por hardware según una forma de realización. El sistema 700 puede ponerse en práctica con lógica de procesamiento que puede comprender hardware (por ejemplo, circuitos, lógica dedicada, lógica programable, etc.), software (tales como instrucciones que se ejecutan en un dispositivo de procesamiento) o una combinación de los mismos. En un ejemplo, el circuito sumador 750 puede ponerse en práctica en al menos una de entre una unidad lógica aritmética, un procesador, un multiprocesador de gráficos, un núcleo GPGPU, un grupo de cálculo informático y cualquier componente de hardware aquí descrito. El método 600 puede ponerse en práctica con un circuito generador de números aleatorios uniformes (URNG) 710 para generar números aleatorios uniformes y un circuito sumador 750 para generar números aleatorios gaussianos. Los números aleatorios uniformes se pueden transmitir con las comunicaciones 711-713 a cualquier sumador (por ejemplo, Sumador-720, Sumador-721, ... Sumador-N, etc.) del circuito sumador 750 de un componente de hardware (por ejemplo, un procesador, un multiprocesador de gráficos, un núcleo GPGPU, un grupo de cálculo informático, etc.).

En un ejemplo, Sumador-1 agrega dos números aleatorios uniformes (por ejemplo, el primer y segundo números aleatorios uniformes del circuito URNG 710) para generar una primera salida de Sumador-720. El Sumador-721 agrega otro número aleatorio uniforme (por ejemplo, el tercer número aleatorio uniforme del circuito URNG) con la primera salida para generar una primera salida de Sumador-721. Los sumadores posteriores (por ejemplo, Sumador-3 a Sumador-N-1, inclusive) luego continúan agregando otro número aleatorio uniforme a una salida de un sumador anterior. Sumador-N agrega otro número aleatorio uniforme (por ejemplo, el número aleatorio uniforme N+1 del circuito URNG 710) con una primera salida de Sumador-N-1 para generar un primer número aleatorio gaussiano que se puede transmitir a otra ubicación o almacenar en memoria.

Después de que el Sumador-720 haya terminado de generar la primera salida, el Sumador-720 puede agregar otros dos números aleatorios uniformes (por ejemplo, diferentes números aleatorios uniformes) para generar una segunda salida del Sumador-721. Los sumadores posteriores (por ejemplo, Sumador-2 a Sumador-N-1) luego continúan agregando otro número aleatorio uniforme a una salida de un sumador anterior. Sumador-N agrega otro número aleatorio uniforme con una segunda salida de Sumador-N-1 para generar un segundo número aleatorio gaussiano que se transmite a otra ubicación o se almacena en la memoria. Sumador-720 a Sumador-N, inclusive, se utilizan luego para continuar agregando números aleatorios uniformes para generar números aleatorios gaussianos adicionales. Los sumadores en la Figura 7 puede configurarse de cualquier manera para la suma de los números aleatorios uniformes para generar números aleatorios gaussianos. Puede utilizarse cualquier tipo de sumadores para los sumadores de la Figura 7.

La Figura 8 ilustra un diagrama de tiempo para la generación de números aleatorios gaussianos de conformidad con una forma de realización. El diagrama de temporización 800 incluye un eje de números aleatorios gaussianos vertical 812 y un eje de tiempo horizontal 810 en unidades de milisegundos. Una línea de tiempo 830 representa un primer período de tiempo para generar un primer número aleatorio gaussiano G1. El primer período de tiempo incluye una serie de retardos T_{A1} de Sumador-1, T_{A2} de Sumador-2, ... T_{AN} de Sumador-N para agregar números aleatorios uniformes para generar el primer número aleatorio gaussiano G1. Una línea de tiempo 840 representa un segundo período de tiempo que incluye una serie de retardos T_{A1} de Sumador-1, T_{A2} de Sumador-2, ... T_{AN} de Sumador-N para agregar números aleatorios uniformes para generar el segundo número aleatorio gaussiano G2. Una línea de tiempo 850 representa un tercer período de tiempo que incluye una serie de retardos T_{A1} de Sumador-1, T_{A2} de Sumador-2, ... T_{AN} de Sumador-N para agregar números aleatorios uniformes para generar el tercer número aleatorio gaussiano G3. Los períodos de tiempo primero, segundo y tercero son aproximadamente iguales.

En un ejemplo, se genera un primer número gaussiano en un primer momento que es igual a un retardo de sumador multiplicado por un número de sumadores N. Se genera un segundo número gaussiano en un segundo momento que es igual a un retardo de sumador multiplicado por un número de sumadores N+1. Se genera un m-ésimo número aleatorio gaussiano en m-ésimo momento que equivale a un retardo de sumador multiplicado por número de sumadores N+m-1.

Los números aleatorios gaussianos se pueden generar para aplicaciones de aprendizaje máquina. En particular, el método acelerado por hardware para generar una gran cantidad de números aleatorios gaussianos se puede utilizar para aplicaciones de aprendizaje máquina basadas en bayesiano que pueden necesitar una gran cantidad de números aleatorios gaussianos generados en un corto período de tiempo.

Descripción general del aprendizaje máquina

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

Un tipo, a modo de ejemplo, de algoritmo de aprendizaje máquina es una red neuronal. Existen muchos tipos de redes neuronales; un tipo simple de red neuronal es una red prealimentada. Una red prealimentada puede ponerse en práctica como un gráfico acíclico en donde los nodos están dispuestos en capas. En condiciones normales, una topología de red prealimentada 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 de utilidad para generar salida en la capa de salida. Los nodos de la red están completamente conectados mediante los bordes a los nodos en las capas adyacentes, pero no existen bordes entre los nodos dentro de cada capa. Los datos recibidos en los nodos de una capa de entrada de una red prealimentada se propagan (es decir, "prealimentan") 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 en función de coeficientes ("pesos") respectivamente asociados con cada uno de los bordes que conectan las capas. Dependiendo del modelo específico representado por el algoritmo que se está ejecutando, la salida del algoritmo de red neuronal puede adoptar varias formas.

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

La precisión de un algoritmo de aprendizaje máquina puede verse afectada en gran medida por la calidad del conjunto de datos utilizado para entrenar el algoritmo. El proceso de entrenamiento puede ser computacionalmente intensivo y puede requerir una cantidad importante de tiempo en un procesador de uso general convencional. En consecuencia, el hardware de procesamiento en paralelo se utiliza para entrenar muchos tipos de algoritmos de aprendizaje máquina. Lo que antecede es particularmente de utilidad para optimizar el entrenamiento de las redes neuronales, ya que los cálculos realizados para ajustar los coeficientes en las redes neuronales se prestan de manera natural por sí mismos a puestas en práctica en paralelo. Concretamente, muchos algoritmos de aprendizaje máquina y aplicaciones de software se han adaptado para hacer uso del hardware de procesamiento en paralelo dentro de los dispositivos de procesamiento de gráficos de uso general.

La Figura 9 es un diagrama generalizado de una pila de software 900 de aprendizaje máquina. Se puede configurar una aplicación de aprendizaje máquina 902 para entrenar una red neuronal utilizando un conjunto de datos de entrenamiento o para utilizar una red neuronal de profundidad capacitada para poner en práctica inteligencia de máquina. La aplicación de aprendizaje máquina 902 puede incluir funcionalidad de entrenamiento e inferencia para una red neuronal y/o software especializado que puede utilizarse para entrenar una red neuronal antes del despliegue. La aplicación de aprendizaje máquina 902 puede poner en práctica cualquier tipo de inteligencia de máquina que incluye, sin limitación, reconocimiento de imágenes, asignación y localización, navegación autónoma, síntesis de voz, imágenes médicas o traducción de idiomas.

La aceleración por hardware para la aplicación de aprendizaje máquina 902 se puede habilitar mediante una estructura de aprendizaje máquina 904. La estructura de aprendizaje máquina 904 puede proporcionar una biblioteca de primitivos de aprendizaje máquina. Los primitivos de aprendizaje máquina son operaciones básicas que se suelen realizar por algoritmos de aprendizaje máquina. Sin la estructura de aprendizaje máquina 904, los desarrolladores de algoritmos de aprendizaje máquina tendrían que crear y optimizar la lógica computacional principal asociada con el algoritmo de aprendizaje máquina y luego volver a optimizar la lógica computacional a medida que se desarrollan nuevos procesadores en paralelos. En cambio, la aplicación de aprendizaje máquina se puede configurar para realizar los cálculos necesarios utilizando los primitivos proporcionados por la estructura de aprendizaje máquina 904. Los primitivos, a modo de ejemplo, incluyen convoluciones de tensor, funciones de activación y agrupación, que son operaciones computacionales que se realizan mientras se entrena una red neuronal convolucional (CNN). La estructura de aprendizaje máquina 904 también puede proporcionar primitivos para poner en práctica subprogramas básicos de álgebra lineal realizados por muchos algoritmos de aprendizaje máquina, tales como operaciones matriciales y vectoriales.

La estructura de aprendizaje máquina 904 puede procesar los datos de entrada recibidos desde la aplicación de aprendizaje máquina 902 y generar la entrada apropiada para una estructura informática 906. La estructura informática 906 puede abstraer las instrucciones subyacentes proporcionadas al controlador GPGPU 908 para permitir que la estructura de aprendizaje máquina 904 se aproveche de la aceleración por hardware mediante hardware GPGPU 910 sin requerir que la estructura de aprendizaje máquina 904 tenga un conocimiento profundo de la arquitectura del hardware GPGPU 910. Además, la estructura informática 906 puede habilitar la aceleración por hardware para la estructura de aprendizaje máquina 904 mediante una diversidad de tipos y generaciones del hardware GPGPU 910.

Aceleración de aprendizaje máquina GPGPU

La Figura 10 ilustra una unidad de procesamiento de gráficos de uso general 1000 altamente paralela, según una forma de realización. En una forma de realización, la unidad de procesamiento de uso general (GPGPU) 1000 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 1000 se puede vincular directamente a otras instancias de la GPGPU para crear un grupo multi-GPU para mejorar la velocidad de entrenamiento para redes neuronales particularmente profundas.

La GPGPU 1000 incluye una interfaz host 1002 para permitir una conexión con un procesador host. En una forma de realización, la interfaz host 1002 es una interfaz PCI Express. Sin embargo, la interfaz host también puede ser una interfaz de comunicaciones o una estructura de comunicaciones específica del proveedor. La GPGPU 1000 recibe comandos del procesador host y utiliza un planificador global 1004 para distribuir subprocesos de ejecución asociados con dichos comandos a un conjunto de grupos de cálculo 1006A-H. Los grupos de cálculo 1006A-H comparten una memoria caché 1008. La memoria caché 1008 puede servir como una memoria caché de nivel superior para las memorias caché dentro de los grupos de cálculo 1006A-H.

La GPGPU 1000 incluye la memoria 1014A-B acoplada con los grupos de cálculo 1006A-H mediante un conjunto de controladores de memoria 1012A-B. En varias formas de realización, la memoria 1014A-B puede incluir diversos tipos de dispositivos de memoria, incluyendo la memoria de acceso aleatorio dinámica (DRAM) o la memoria de acceso aleatorio gráfica, tal como la memoria de acceso aleatorio gráfica síncrona (SGRAM), incluyendo la memoria de tasa de datos dobles gráfica (G17R). En una forma de realización, las unidades de memoria 224A-N también pueden incluir memoria apilada en 3D, que incluye, sin limitación, memoria de alto ancho de banda (HBM).

En una forma de realización, cada grupo de cálculo informático GPLAB06A-H incluye un conjunto de multiprocesadores de gráficos, tal como el multiprocesador de gráficos 400 de la Figura 4A. Los multiprocesadores de gráficos del grupo de cálculo informático tienen diversos tipos de unidades lógicas de coma flotante y enteros que pueden realizar operaciones computacionales en un margen de precisiones, incluyendo las adecuadas para los cálculos de aprendizaje máquina. Por ejemplo, y en una forma de realización, al menos un subconjunto de las unidades de coma flotante, en cada uno de los grupos de cálculo 1006A-H, se puede configurar para realizar operaciones de coma flotante de 16 o 32 bits, mientras que un subconjunto diferente de las unidades de coma flotante puede configurarse para realizar operaciones de coma flotante de 64 bits.

Se pueden configurar múltiples instancias de la GPGPU 1000 para operar como un grupo de cálculo informático. El mecanismo de comunicación utilizado por el grupo de cálculo informático para la sincronización y el intercambio de datos varía según las formas de realización. En una forma de realización, las múltiples instancias de la GPGPU 1000 se comunican mediante la interfaz host 1002. En una forma de realización, la GPGPU 1000 incluye un concentrador de E/S 1008 que acopla la GPGPU 1000 con un enlace GPU 1010 que permite una conexión directa con otras instancias de la GPGPU. En una forma de realización, el enlace de GPU 1010 está acoplado a un puente de GPU a GPU dedicado que permite la comunicación y sincronización entre múltiples instancias de GPGPU 1000. En una forma de realización, el enlace de GPU 1010 se acopla con una interconexión de alta velocidad para transmitir y recibir datos a otras unidades GPGPUs o procesadores en paralelo. En una forma de realización, las instancias múltiples de GPGPU 1000 están ubicadas en sistemas de procesamiento de datos separados y se comunican mediante un dispositivo de red al que se puede acceder mediante la interfaz host 1002. En una forma de realización, el enlace de GPU 1010 puede configurarse para permitir una conexión a un procesador host, además de, o como una alternativa a, la interfaz host 1002.

Si bien la configuración ilustrada de la GPGPU 1000 se puede configurar para entrenar redes neuronales, una forma de realización proporciona una configuración alternativa de la GPGPU 1000 que se puede configurar para su puesta en práctica dentro de una plataforma de inferencia de alto rendimiento o de baja potencia. En una configuración de inferencia, la GPGPU 1000 incluye menos grupos de cálculo 1006A-H en relación con la configuración de entrenamiento. Además, la tecnología de memoria asociada con la memoria 1014A-B puede diferir entre las configuraciones de inferencia y entrenamiento. En una forma de realización, la configuración de inferencia de la GPGPU 1000 puede admitir instrucciones específicas de inferencia. Por ejemplo, una configuración de inferencia puede proporcionar soporte para una o más instrucciones de productos de punto entero de 8 bits, que se suelen utilizar durante las operaciones de inferencia para las redes neuronales puesta en prácticas.

La Figura 11 ilustra un sistema informático multi-GPU 1100, según una forma de realización. El sistema informático multi-GPU 1100 puede incluir un procesador 1102 acoplado a múltiples GPGPUs 1106A-D mediante un conmutador de interfaz host 1104. El conmutador de interfaz host 1104, en una forma de realización, es un dispositivo conmutador PCI Express que acopla el procesador 1102 a un bus PCI Express a través del cual el procesador 1102 puede comunicarse con el conjunto de GPGPUs 1106A-D. Cada una de las múltiples GPGPUs 1106A-D puede ser una instancia de la GPGPU 1000 de la Figura 10. Las GPGPUs 1106A-D pueden interconectarse mediante un conjunto de enlaces de GPU a GPU punto a punto de alta velocidad 1116. Los enlaces de GPU a GPU de alta velocidad pueden conectarse a cada una de las GPGPUs 1106A-D mediante un enlace GPU dedicado, tal como el enlace GPU 1010 según se ilustra en la Figura 10. Los enlaces de GPU P2P 1116 permiten la comunicación directa entre cada una de

las unidades GPGPUs 1106A-D sin requerir comunicación mediante bus de interfaz host al que está conectado el procesador 1102. Con el tráfico de GPU a GPU dirigido a los enlaces de GPU P2P, el bus de interfaz host permanece disponible para acceder a la memoria del sistema o para comunicarse con otras instancias del sistema informático multi-GPU 1100, por ejemplo, mediante uno o más dispositivos de red. Mientras que en la forma de realización ilustrada las GPGPUs 1106A-D se conectan al procesador 1102 mediante un conmutador de interfaz host 1104, en una forma de realización, el procesador 1102 incluye soporte directo para los enlaces GPU P2P 1116 y puede conectarse directamente a las GPGPUs 1106A-D.

Puestas en práctica de redes neuronales de aprendizaje máquina

La arquitectura informática proporcionada por las formas de realización descritas en el presente documento se puede configurar para realizar los tipos de procesamiento en paralelo que son particularmente adecuados para entrenar y desplegar redes neuronales para el aprendizaje máquina. Una red neuronal se puede generalizar como una red de funciones que tienen una relación gráfica. Como es bien conocido en esta técnica, existe una diversidad de tipos de puestas en práctica de redes neuronales utilizadas en el aprendizaje máquina. Un tipo, a modo de ejemplo, de red neuronal es la red de realimentación, tal como se describió con anterioridad.

Un segundo tipo, a modo de ejemplo, de red neuronal es la red neuronal convolucional (CNN). Una CNN es una red neuronal prealimentada especializada para procesar datos que tienen una topología similar a una cuadrícula conocida, como datos de imágenes. En consecuencia, las CNNs se suelen utilizar para calcular aplicaciones de reconocimiento de imágenes y visión, pero también se pueden utilizar para otros tipos de reconocimiento de patrones, tales como el procesamiento de voz y del lenguaje. 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 cálculos para una red CNN incluyen aplicar la operación matemática de convolución a cada filtro para obtener la salida de ese filtro. La convolución es un tipo especializado de operación matemática realizada por dos funciones para obtener una tercera función que es una versión modificada de una de las dos funciones originales. En la terminología de red convolucional, la primera función de la convolución puede denominarse como la entrada, mientras que la segunda función puede denominarse como el kernel de convolución. La salida puede denominarse como el mapa de características. Por ejemplo, la entrada a una capa de convolución puede ser una matriz multidimensional de datos que define los diversos componentes de color de una imagen de entrada. El kernel de convolución puede ser una matriz multidimensional de parámetros, en donde los parámetros son adaptados por el proceso de entrenamiento para la red neuronal.

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

Las figuras que se describen a continuación presentan redes de avance, CNN y RNN, a modo de ejemplo, así como también describen un proceso general para entrenar y desplegar, respectivamente, cada uno de dichos tipos de redes. Se entenderá que estas descripciones son, a modo de ejemplo, y no limitativas en cuanto a cualquier forma de realización específica descrita en este documento y los conceptos ilustrados se pueden aplicar en general a redes neuronales profundas y técnicas de aprendizaje máquina en general.

Las redes neuronales, a modo de ejemplo, descritas con anterioridad se pueden utilizar para realizar un aprendizaje de profundidad. El aprendizaje de profundidad es el aprendizaje máquina que utiliza redes neuronales profundas. Las redes neuronales profundas utilizadas en el aprendizaje de profundidad son redes neuronales artificiales compuestas de múltiples capas ocultas, a diferencia de las redes neuronales superficiales que incluyen una sola capa oculta. Las redes neuronales más profundas suelen ser más computacionalmente intensivas para su capacitación. Sin embargo, las capas ocultas adicionales de la red permiten el reconocimiento de patrones de diversas etapas que da como resultado un error de salida reducido en relación con las técnicas superficiales de aprendizaje máquina.

Las redes neuronales profundas utilizadas en el aprendizaje de profundidad suelen incluir una red de extremo frontal para realizar el reconocimiento de características acopladas a una red de extremo posterior, lo que representa un modelo matemático que puede realizar operaciones (p. ej., clasificación de objetos, reconocimiento de voz, etc.) basadas en la representación de características proporcionadas al modelo. El aprendizaje de profundidad permite realizar el aprendizaje máquina sin necesidad de realizar una ingeniería de características artesanal para el modelo. En cambio, las redes neuronales profundas pueden aprender características basadas en la estructura estadística o en la correlación dentro de los datos de entrada. Las funciones aprendidas se pueden proporcionar a un modelo matemático que puede asignar las funciones detectadas a una salida. El modelo matemático utilizado por la red suele estar especializado para la tarea específica a realizar, y se utilizarán diferentes modelos para realizar diferentes tareas.

Una vez que la red neuronal está estructurada, se puede aplicar 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 utilizado para entrenar redes neuronales. Un vector de entrada se presenta a la red para su procesamiento. La salida de la red se compara con la salida deseada utilizando una función de pérdida y se calcula un valor de error para cada una de las neuronas en la capa de salida. A continuación, los valores de error se propagan hacia atrás, hasta que cada neurona tenga un valor de error asociado que represente aproximadamente su aportación a la salida original. A continuación, la red puede aprender de dichos errores utilizando un algoritmo, tal como el algoritmo de descenso de gradiente estocástico, para actualizar los pesos de la red neuronal.

La Figura 12A-B ilustran una red neuronal convolucional, a modo de ejemplo. La Figura 12A ilustra varias capas dentro de una red CNN. Tal como se muestra en la Figura 12A, un ejemplo de red CNN utilizada para modelar el procesamiento de imágenes puede recibir la entrada 1202 que describe los componentes rojo, verde y azul (RGB) de una imagen de entrada. La entrada 1202 puede ser procesada por múltiples capas convolucionales (por ejemplo, capa convolucional 1204, capa convolucional 1206). La salida desde las múltiples capas convolucionales puede ser procesada de manera opcional por un conjunto de capas completamente conectadas 1208. Las neuronas en una capa completamente conectada tienen conexiones completas para todas las activaciones en la capa anterior, tal como se describió con anterioridad para una red de realimentación. La salida de las capas completamente conectadas 1208 puede utilizarse para generar un resultado de salida de la red. Las activaciones dentro de las capas completamente conectadas 1208 se pueden calcular utilizando la multiplicación de matrices en lugar de la convolución. No todas las puestas en práctica de CNN utilizan capas totalmente conectadas DPLA08. Por ejemplo, en algunas puestas en práctica, la capa convolucional 1206 puede generar una salida para la red CNN.

Las capas convolucionales están escasamente conectadas, lo que difiere de la configuración de red neuronal tradicional que se encuentra en las capas 1208 completamente conectadas. Las capas de red neuronal tradicionales están completamente conectadas, de modo que cada unidad de salida interactúa con cada unidad de entrada. Sin embargo, las capas convolucionales están escasamente conectadas porque la salida de la convolución de un campo se introduce (en lugar del valor de estado respectivo de cada uno de los nodos en el campo) a los nodos de la capa subsiguiente, tal como se ilustra. Los kernel asociados con las capas convolucionales realizan operaciones de convolución, cuya salida se envía a la siguiente capa. La reducción de la dimensionalidad realizada dentro de las capas convolucionales es un aspecto que permite escalar la red CNN para procesar imágenes de mayor tamaño.

La Figura 12B ilustra etapas de cálculo, a modo de ejemplo, dentro de una capa convolucional de una red CNN. La entrada a una capa convolucional 1212 de una red CNN puede procesarse en tres etapas de una capa convolucional 1214. Las tres etapas pueden incluir una etapa de convolución 1216, una etapa de detector 1218 y una etapa de agrupación 1220. La capa convolucional 1214 puede generar datos a una capa convolucional sucesiva. La capa convolucional final de la red puede generar datos de mapa de características de salida o proporcionar entrada a una capa completamente conectada, por ejemplo, para generar un valor de clasificación para la entrada a la red CNN.

En la etapa de convolución 1216 se realizan varias convoluciones en paralelo para obtener un conjunto de activaciones lineales. La etapa de convolución 1216 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, puesta en escala y combinaciones de estas transformaciones. La etapa de convolución calcula la salida de funciones (por ejemplo, neuronas) que están conectadas a zonas específicas en la entrada, que se puede determinar como la zona local asociada con la neurona. Las neuronas calculan un producto escalar entre los pesos de las neuronas y la zona en la entrada local a la que están conectadas las neuronas. La salida desde la etapa de convolución 1216 define un conjunto de activaciones lineales que son procesadas por etapas sucesivas de la capa convolucional 1214.

Las activaciones lineales pueden ser procesadas por una etapa de detector 1218. En la etapa de detector 1218, 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 los campos receptivos de la capa de convolución. Se pueden utilizar diversos tipos de funciones de activación no lineales. Un tipo particular es la unidad lineal rectificadora (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 agrupación 1220 utiliza una función de agrupación que sustituye la salida de la capa convolucional 1206 con una estadística de resumen de las salidas cercanas. La función de agrupación se puede utilizar para introducir la invariancia de la traslación en la red neuronal, de modo que las pequeñas traslaciones a la entrada no cambien las salidas agrupadas. La invariancia a la traslación local puede ser útil en escenarios en donde la presencia de una característica en los datos de entrada es más importante que la ubicación precisa de la característica. Se pueden utilizar diversos tipos de funciones de agrupación durante la etapa de agrupación 1220, incluyendo agrupación máxima, agrupación media y agrupación de 12 normas. Además, algunas puestas en práctica de CNN no incluyen una etapa de agrupación. En su lugar, dichas puestas en práctica sustituyen una etapa de convolución adicional que tiene una etapa mayor en relación con las etapas de convolución anteriores.

La salida desde la capa convolucional 1214 puede luego ser procesada por la siguiente capa 1222. La siguiente capa 1222 puede ser una capa convolucional adicional o una de las capas completamente conectadas 1208. Por ejemplo, la primera capa convolucional 1204 de la Figura 12A puede enviar a la segunda capa convolucional 1206, mientras que la segunda capa convolucional puede enviar a una primera capa de las capas completamente conectadas 1208.

La Figura 13 ilustra una red neuronal recurrente, a modo de ejemplo, 1300. En una red neuronal recurrente (RNN), el estado anterior de la red influye en la salida del estado actual de la red. Las redes RNNs se pueden construir de varias maneras utilizando una diversidad de funciones. El uso de las redes RNNs por lo general gira en torno a la utilización de modelos matemáticos para predecir el futuro en función de una secuencia previa de entradas. Por ejemplo, se puede utilizar una red RNN para realizar un modelado estadístico del lenguaje para predecir una próxima palabra dada una secuencia previa de palabras. La red RNN 1300 ilustrada se puede describir con una capa de entrada 1302 que recibe un vector de entrada, capas ocultas 1304 para poner en práctica una función recurrente, un mecanismo de retroalimentación 1305 para habilitar una "memoria" de estados previos y una capa de salida 1306 para generar un resultado. La red RNN 1300 opera en base a etapas de tiempo. El estado de la red RNN, en una etapa de tiempo dado, se ve influenciado en base a la etapa de tiempo anterior mediante el mecanismo de retroalimentación 1305. Para una etapa de tiempo dado, el estado de las capas ocultas 1304 está definido por el estado anterior y la entrada en una etapa de tiempo actual. La capa oculta 1304 puede procesar una entrada inicial (x_1) en una primera etapa de tiempo. La capa oculta 1304 puede procesar una segunda entrada (x_2) utilizando información de estado que se determina durante el procesamiento de la entrada inicial (x_1). Un estado dado se puede calcular como $s_t = f(Ux_t + Ws_{t-1})$, en donde U y W son matrices de parámetros. La función f es por lo general una no linealidad, tal como la función de 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 1304 puede variar según los datos de puesta en práctica específicos de la red RNN 1300.

Además, de las redes CNN y RNN básicas descritas, se pueden habilitar variaciones en dichas redes. Un ejemplo de variante de RNN es la RNN de memoria a largo plazo (LSTM). Las RNNs de LSTM son capaces de aprender dependencias a largo plazo que pueden ser necesarias para procesar secuencias de lenguaje más largas. Una variante de la red CNN es una red de creencias profundas convolucional, que tiene una estructura similar a una red CNN y está capacitada de manera similar a una red de creencias profundas. Una red de creencias profundas (DBN) es una red neuronal generativa que se compone de múltiples capas de variables estocásticas (aleatorias). Las redes DBNs se pueden entrenar capa por capa mediante un aprendizaje profundo no supervisado. Los pesos aprendidos de la red DBN se pueden utilizar para proporcionar redes neuronales previas al entrenamiento mediante la determinación de un conjunto inicial óptimo de pesos para la red neuronal.

La Figura 14 ilustra el entrenamiento y el despliegue de una red neuronal de profundidad. Una vez que se ha estructurado una red determinada para una tarea, la red neuronal se entrena utilizando un conjunto de datos de entrenamiento 1402. Se han desarrollado diversas estructuras de entrenamiento 1404 para permitir la aceleración por hardware del proceso de entrenamiento. Por ejemplo, la estructura de aprendizaje máquina 904 de la Figura 9 puede configurarse como una estructura de entrenamiento 904. La estructura de entrenamiento 904 puede conectarse a una red neuronal no capacitada 1406 y permitir que la red neuronal no capacitada se entrene utilizando los recursos de procesamiento en paralelo descritos en este documento para generar una red neuronal capacitada 1408.

Para iniciar el proceso de entrenamiento, los pesos iniciales pueden elegirse de manera aleatoria o mediante un entrenamiento previo utilizando una red de creencias profundas. Entonces, el ciclo de entrenamiento puede realizarse de forma supervisada o no supervisada.

El aprendizaje supervisado es un método de aprendizaje en donde el entrenamiento se realiza como una operación mediada, tal como cuando el conjunto de datos de entrenamiento 1402 incluye una entrada emparejada con la salida deseada para la entrada, o en donde el conjunto de datos de entrenamiento incluye una entrada que tiene una salida conocida y la salida de la red neuronal se califica de manera manual. La red procesa las entradas y compara las salidas resultantes con un conjunto de salidas esperadas o previstas. A continuación, los errores se propagan de nuevo a través del sistema. La estructura de entrenamiento 1404 puede ajustarse para ajustar los pesos que controlan la red neuronal no capacitada 1406. La estructura de entrenamiento 1404 puede proporcionar herramientas para supervisar como de bien la red neuronal no capacitada 1406 está convergiendo hacia un modelo adecuado para generar respuestas correctas basadas en datos de entrada conocidos. El proceso de entrenamiento ocurre de manera repetida a medida que se ajustan los pesos de la red para refinar la salida generada por la red neuronal. El proceso de entrenamiento puede continuar hasta que la red neuronal alcance una precisión estadísticamente deseada asociada con una red neuronal capacitada 1408. La red neuronal capacitada 1408 puede desplegarse entonces para poner en práctica cualquier número de operaciones de aprendizaje máquina.

El aprendizaje no supervisado es un método de aprendizaje en donde la red intenta entrenarse a sí misma utilizando datos no etiquetados. Por lo tanto, para el aprendizaje no supervisado, el conjunto de datos de entrenamiento 1402 incluirá datos de entrada sin ningún dato de salida asociado. La red neuronal no capacitada 1406 puede aprender agrupamientos dentro de la entrada no etiquetada y puede determinar cómo se relacionan las entradas individuales con el conjunto de datos global. El entrenamiento no supervisado se puede utilizar para generar un mapa autoorganizado, que es un tipo de red neuronal capacitada 1407 capaz de realizar operaciones útiles para reducir la

dimensionalidad de los datos. El entrenamiento no supervisado también se puede utilizar para realizar la detección de anomalías, lo que permite la identificación de puntos de datos en un conjunto de datos de entrada que se desvían de los patrones normales de los datos.

También se pueden emplear variaciones en el entrenamiento supervisado y no supervisado. El aprendizaje semisupervisado es una técnica en donde el conjunto de datos de entrenamiento 1402 incluye una combinación de datos etiquetados y no etiquetados de la misma distribución. El aprendizaje incremental es una variante del aprendizaje supervisado en donde los datos de entrada se utilizan de manera continua para entrenar aún más el modelo. El aprendizaje incremental permite que la red neuronal capacitada 1408 se adapte a los nuevos datos 1412 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 cálculo. En lugar de utilizar un único nodo de cálculo, se puede utilizar una red distribuida de nodos de cálculo para acelerar el proceso de entrenamiento.

La Figura 15 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 anfitriones y uno o más de los nodos de procesamiento de uso general, tal como la unidad de procesamiento gráfico de uso general altamente paralela 1000 como se ilustra en la Figura 1000. Tal como se ilustra, el aprendizaje distribuido se puede realizar con paralelismo de modelo 1502, paralelismo de datos 1504 o una combinación de paralelismo de modelo y datos 1504.

En el paralelismo de modelo 1502, diferentes nodos computacionales en un sistema distribuido pueden realizar cálculos de entrenamiento para diferentes partes de una sola red. Por ejemplo, cada capa de una red neuronal puede ser capacitada por un nodo de procesamiento diferente del sistema distribuido. Los beneficios del paralelismo de modelos incluyen la capacidad de poner en escala a modelos particularmente grandes. La división de los cálculos asociados con diferentes capas de la red neuronal permite el entrenamiento de redes neuronales muy grandes en las que los pesos de todas las capas no cabrían en la memoria de un único nodo computacional. En algunas instancias operativas, el paralelismo de modelos puede ser particularmente útil para realizar un entrenamiento no supervisado de grandes redes neuronales.

En el paralelismo de datos 1504, los diferentes nodos de la red distribuida tienen una instancia completa del modelo y cada nodo recibe una parte diferente de los datos. A continuación, se combinan los resultados de los diferentes nodos. Si bien son posibles diferentes enfoques para el paralelismo de datos, todos los enfoques de entrenamiento en paralelo de datos requieren una técnica de combinación de resultados y de sincronización de los parámetros del modelo entre cada nodo. Los enfoques, a modo de ejemplo, para combinar datos incluyen la promediación de parámetros y el paralelismo de datos basado en actualizaciones. La promediación de parámetros capacita a cada nodo en un subconjunto de los datos de entrenamiento y establece los parámetros globales (p. ej., pesos, polarizaciones) a la media de los parámetros de cada nodo. La promediación de parámetros utiliza un servidor de parámetros central que mantiene los datos de los parámetros. El paralelismo de datos basado en actualizaciones es similar a una promediación de parámetros, excepto que, en lugar de transferir parámetros de los nodos al servidor de parámetros, se transfieren las actualizaciones al modelo. Además, el paralelismo de datos basado en actualizaciones se puede realizar de manera descentralizada, en donde las actualizaciones se comprimen y transfieren entre nodos.

Los paralelismos de modelos y de datos combinados 1506 se pueden poner en práctica, por ejemplo, en un sistema distribuido en donde cada nodo computacional incluye múltiples GPUs. Cada nodo puede tener una instancia completa del modelo con GPUs separadas dentro de cada nodo que se utilizan para entrenar diferentes partes del modelo.

El entrenamiento distribuido ha aumentado la sobrecarga relativa al entrenamiento en una máquina única. Sin embargo, los procesadores en paralelo y las GPGPUs descritas en este documento pueden poner, cada una, en práctica diversas técnicas para reducir la sobrecarga del entrenamiento distribuido, incluyendo técnicas para permitir la transmisión de datos de GPU a GPU de gran ancho de banda y la sincronización de datos distantes acelerada.

Aplicaciones, a modo de ejemplo, de aprendizaje máquina

El aprendizaje máquina se puede aplicar para resolver una diversidad de problemas tecnológicos, que incluyen, sin limitación, visión por ordenador, conducción y navegación autónomas, reconocimiento de voz y procesamiento del lenguaje. La visión por ordenador ha sido, de manera tradicional, una de las zonas de investigación más activas para las aplicaciones de aprendizaje máquina. Las aplicaciones de la visión por ordenador van desde la reproducción de las capacidades visuales humanas, como el reconocimiento de rostros, hasta la creación de nuevas categorías de capacidades visuales. Por ejemplo, las aplicaciones de visión por ordenador se pueden configurar para reconocer ondas de sonido a partir de las vibraciones inducidas en los objetos visibles en un vídeo. El aprendizaje máquina acelerado por procesadores en paralelo permite que las aplicaciones de visión por ordenador se entrenen utilizando

un conjunto de datos de entrenamiento significativamente más grande que lo que era factible con anterioridad y permite que los sistemas de inferencia se pongan en práctica utilizando procesadores en paralelo de baja potencia.

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

Las redes neuronales profundas aceleradas por procesador en paralelo han permitido métodos de aprendizaje máquina para el reconocimiento de voz automático (ASR). ASR incluye la creación de una función que calcula la secuencia lingüística más probable dada una secuencia acústica de entrada. El aprendizaje máquina acelerado que utiliza redes neuronales profundas ha permitido la sustitución de los modelos ocultos de Markov (H26s) y de los modelos de mezcla gaussiana (G26s) que se utilizaban con anterioridad para el reconocimiento ASR.

El aprendizaje máquina acelerado por procesador en paralelo también se puede utilizar para acelerar el procesamiento del lenguaje natural. Los procedimientos de aprendizaje máquina pueden hacer uso de algoritmos de inferencia estadística para obtener modelos que sean resistentes a una entrada errónea o desconocida. Las aplicaciones, a modo de ejemplo, del procesador de lenguaje natural incluyen la traducción automática entre lenguajes humanos.

Las plataformas de procesamiento en paralelo utilizadas para el aprendizaje máquina se pueden dividir en plataformas de entrenamiento y plataformas de puesta en práctica. Las plataformas de entrenamiento suelen ser muy paralelas e incluyen optimizaciones para acelerar el entrenamiento de un único nodo de múltiples GPUs y el entrenamiento de múltiples nodos y de múltiples GPU. Los procesadores en paralelo, a modo de ejemplo, adecuados para el entrenamiento incluyen la unidad de procesamiento gráfico de uso general altamente paralela 1000 de la Figura 1000 y el sistema informático multi-GPU 1100 de la Figura 1100. Por el contrario, las plataformas de aprendizaje máquina puestas en práctica por lo general incluyen procesadores en paralelo de menor potencia adecuados para su uso en productos tales como cámaras, robots autónomos y vehículos autónomos.

La Figura 16 ilustra un sistema de inferencia, a modo de ejemplo, en un circuito integrado (SOC) 1600 adecuado para realizar inferencias utilizando un modelo capacitado. El SOC 1600 puede integrar componentes de procesamiento que incluyen un procesador de medios 1602, un procesador de visión 1604, una GPGPU 1606 y un procesador multinúcleo 1608. El SOC 1600 puede incluir, además, una memoria en circuito integrado 1605 que puede habilitar un grupo de datos en circuito integrado compartido que es accesible por cada uno de los componentes de procesamiento. Los componentes de procesamiento se pueden optimizar para una operación de baja potencia para permitir la puesta en práctica, en una diversidad de plataformas de aprendizaje máquina, incluyendo vehículos autónomos y robots autónomos. Por ejemplo, una puesta en práctica del SOC 1600 puede utilizarse como parte del sistema de control principal para un vehículo autónomo. Cuando el SOC 1600 está configurado para su uso en vehículos autónomos, el SOC está diseñado y configurado para cumplir con las normas de seguridad funcional pertinentes de la jurisdicción de puesta en práctica.

Durante la operación, el procesador de medios 1602 y el procesador de visión 1604 pueden trabajar en conjunto para acelerar las operaciones de visión por ordenador. El procesador de medios 1602 puede habilitar la decodificación de baja latencia de múltiples flujos de vídeo de alta resolución (por ejemplo, 4K, 8K). Los flujos de vídeo decodificados pueden ser objeto de escritura en una memoria intermedia en la memoria del circuito integrado 1605. El procesador de visión 1604 puede luego analizar el vídeo decodificado y realizar operaciones de procesamiento preliminares en las tramas del vídeo decodificado en preparación del procesamiento de las tramas utilizando un modelo de reconocimiento de imagen capacitado. Por ejemplo, el procesador de visión 1604 puede acelerar las operaciones de convolución para una red CNN que se utiliza para realizar el reconocimiento de imágenes en los datos de vídeo de alta resolución, mientras que los cálculos del modelo de extremo posterior se realizan por la unidad GPGPU 1606.

El procesador multinúcleo 1608 puede incluir lógica de control para ayudar con la secuenciación y sincronización de transferencias de datos y operaciones de memoria compartida realizadas por el procesador de medios 1602 y el procesador de visión 1604. El procesador multinúcleo 1608 también puede funcionar como un procesador de aplicación para ejecutar aplicaciones de software que pueden hacer uso de la capacidad de cálculo de inferencia de la GPGPU 1606. Por ejemplo, al menos una parte de la lógica de navegación y conducción puede ponerse en práctica en el software que se ejecuta en el procesador multinúcleo 1608. Dicho software puede emitir cargas de trabajo computacionales directamente a la GPGPU 1606 o las cargas de trabajo computacionales se pueden emitir al procesador multinúcleo 1608, que puede descargar, al menos, una parte de esas operaciones a la GPGPU 1606.

La GPGPU 1606 puede incluir grupos de cálculo tales como una configuración de baja potencia de los grupos de cálculo 1006A-1006H dentro de la unidad de procesamiento gráfico de uso general altamente paralela 1000. Los grupos de cálculo dentro de la GPGPU 1606 pueden admitir instrucciones que están específicamente optimizadas para realizar cálculos de inferencia en una red neuronal capacitada. Por ejemplo, la GPGPU 1606 puede admitir

instrucciones para realizar cálculos de baja precisión, tales como operaciones vectoriales de números enteros de 8 y 4 bits.

Descripción general del sistema

La Figura 17 es un diagrama de bloques de un sistema de procesamiento 1700, según una forma de realización. En varias formas de realización, el sistema 1700 incluye uno o más procesadores 1702 y uno o más procesadores de gráficos 1708, y puede ser un sistema de escritorio de un único procesador, un sistema de estación de trabajo multiprocesador o un sistema de servidor que tiene una gran cantidad de procesadores 1702 o núcleos de procesador 1707. En una forma de realización, el sistema 1700 es una plataforma de procesamiento incorporada dentro de un sistema en un circuito integrado (SoC) para uso en dispositivos móviles, portátiles o integrados.

Una forma de realización del sistema 1700 puede incluir, o incorporarse dentro de una plataforma de juegos basada en servidor, una consola de juegos, que incluye una consola de juegos y medios, una consola de juegos móvil, una consola de juegos portátil o una consola de juegos online. En algunas formas de realización, el sistema 1700 es un teléfono móvil, un teléfono inteligente, un dispositivo informático de tableta electrónica o un dispositivo móvil de Internet. El sistema de procesamiento de datos 1700 también puede incluir, acoplarse o integrarse dentro de un dispositivo portátil, tal como un dispositivo portátil de reloj inteligente, un dispositivo de gafas inteligentes, un dispositivo de realidad aumentada o un dispositivo de realidad virtual. En algunas formas de realización, el sistema de procesamiento de datos 1700 es un televisor o un dispositivo decodificador que tiene uno o más procesadores 1702 y una interfaz gráfica generada por uno o más procesadores de gráficos 1708.

En algunas formas de realización, uno o más procesadores 1702 incluyen, cada uno, uno o más núcleos de procesador 1707 para procesar instrucciones que, cuando se ejecutan, realizan operaciones para el sistema y el software del usuario. En algunas formas de realización, cada uno de los uno o más núcleos de procesador 1707 está configurado para procesar un conjunto de instrucciones específico 1709. En algunas formas de realización, el conjunto de instrucciones 1709 puede facilitar la Computación de Conjunto de Instrucciones Complejas (CISC), la Computación de Conjunto de Instrucciones Reducida (RISC) o la Computación mediante una palabra de instrucción muy larga (VLIW). Múltiples núcleos de procesador 1707 pueden procesar, cada uno, un conjunto de instrucciones diferente 1709, que puede incluir instrucciones para facilitar la emulación de otros conjuntos de instrucciones. El núcleo del procesador 1707 también puede incluir otros dispositivos de procesamiento, tal como un Procesador de Señal Digital (DSP).

En algunas formas de realización, el procesador 1702 incluye una memoria caché 1704. Dependiendo de la arquitectura, el procesador 1702 puede tener una memoria caché interna única o múltiples niveles de memoria caché interna. En algunas formas de realización, la memoria caché se comparte entre varios componentes del procesador 1702. En algunas formas de realización, el procesador 1702 también utiliza una memoria caché externa (por ejemplo, una memoria caché de nivel 3 (L3) o memoria caché de último nivel (LLC)) (no ilustrada), que puede compartirse entre los núcleos de procesador 1707 utilizando técnicas conocidas de coherencia de memoria caché. Un fichero de registro 1706 se incluye de manera adicional en el procesador 1702 que puede incluir diferentes tipos de registros para almacenar diferentes tipos de datos (por ejemplo, registros de números enteros, registros de coma flotante, registros de estado y un registro de puntero de instrucción). Algunos registros pueden ser registros de uso general, mientras que otros registros pueden ser específicos del diseño del procesador 1702.

En algunas formas de realización, el procesador 1702 está acoplado con un bus de procesador 1710 para transmitir señales de comunicación tales como direcciones, datos o señales de control entre el procesador 1702 y otros componentes en el sistema 1700. En una forma de realización, el sistema 1700 utiliza una arquitectura de sistema de "concentrador" a modo de ejemplo, que incluye un concentrador del controlador de memoria 1716 y un concentrador de controlador de entrada/salida (E/S) 1730. Un concentrador del controlador de memoria 1716 facilita la comunicación entre un dispositivo de memoria y otros componentes del sistema 1700, mientras que un concentrador de controlador de E/S (ICH) 1730 proporciona conexiones a dispositivos de E/S mediante un bus de E/S local. En una forma de realización, la lógica del concentrador del controlador de memoria 1716 está integrada dentro del procesador.

El dispositivo de memoria 1720 puede ser un dispositivo de memoria de acceso aleatorio dinámica (DRAM), un dispositivo de memoria de acceso aleatorio estática (SRAM), un dispositivo de memoria instantánea, un dispositivo de memoria de cambio de fase o algún otro dispositivo de memoria que tenga un rendimiento adecuado para servir como memoria de proceso. En una forma de realización, el dispositivo de memoria 1720 puede funcionar como memoria de sistema para el sistema 1700, para almacenar datos 1722 e instrucciones 1721 para utilizar cuando uno o más procesadores 1702 ejecutan una aplicación o proceso. El concentrador del controlador de memoria 1716 también se acopla con un procesador de gráficos externo opcional 1712, que puede comunicarse con los uno o más procesadores de gráficos 1708 en los procesadores 1702 para realizar operaciones de gráficos y de medios.

En algunas formas de realización, ICH 1730 permite que los periféricos se conecten al dispositivo de memoria 1720 y al procesador 1702 mediante un bus de E/S de alta velocidad. Los periféricos de E/S incluyen, sin limitación, un controlador de audio 1746, una interfaz de firmware 1728, un transceptor inalámbrico 1726 (p. ej., Wi-Fi, Bluetooth), un dispositivo de almacenamiento de datos 1724 (p. ej., una unidad de disco duro, memoria instantánea, etc.), y un

controlador de E/S de legado 1740 para acoplar dispositivos de legado (por ejemplo, Sistema Personal 2 (PS/2)) al sistema. Uno o más controladores de bus serie universal (USB) 1742 conectan dispositivos de entrada, tales como combinaciones de teclado y ratón 1744. Un controlador de red 1734 también puede acoplarse con el ICH 1730. En algunas formas de realización, un controlador de red de alto rendimiento (no ilustrado) se acopla con el bus de procesador 1710. Se apreciará que el sistema 1700 mostrado es un ejemplo y no limitativo, ya que pueden utilizarse también otros tipos de sistemas de procesamiento de datos que están configurados de manera diferente. Por ejemplo, el concentrador del controlador de E/S 1730 puede estar integrado dentro de los uno o más procesadores 1702, o el concentrador del controlador de memoria 1716 y el concentrador del controlador de E/S 1730 pueden estar integrados en un procesador de gráficos externo discreto, tal como el procesador de gráficos externo 1712.

La Figura 18 es un diagrama de bloques de una forma de realización de un procesador 1800 que tiene uno o más núcleos de procesador 1802A-1802N, un controlador de memoria integrado 1814 y un procesador de gráficos integrado 1808. El procesador de gráficos 1808 puede incluir un circuito sumador 1810 (por ejemplo, un sistema 700, un circuito sumador 750) de conformidad con formas de realización del presente diseño. Los elementos de la Figura 18, 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 otro lugar del este documento, pero no se limitan a ello. El procesador 1800 puede incluir núcleos adicionales hasta e incluyendo el núcleo adicional 1802N representado por las casillas con líneas discontinuas. Cada uno de los núcleos de procesador 1802A-1802N incluye una o más unidades de memoria caché interna 1804A-1804N. En algunas formas de realización, cada núcleo de procesador también tiene acceso a una o más unidades almacenadas en memoria caché compartidas 1806.

Las unidades de memoria caché internas 1804A-1804N y las unidades de memoria caché compartidas 1806 representan una jerarquía de memoria caché dentro del procesador 1800. La jerarquía de memoria caché puede incluir, al menos, un nivel de memoria caché de datos e instrucciones dentro de cada núcleo del procesador y uno o más niveles de memoria caché compartida de nivel medio, tal como Nivel 2 (L2), Nivel 3 (L3), Nivel 4 (L4) u otros niveles de memoria caché, en donde el nivel más alto de memoria caché antes de la memoria externa se clasifica como LLC. En algunas formas de realización, la lógica de coherencia de memoria caché mantiene la coherencia entre las diversas unidades de memoria caché 1806 y 1804A-1804N.

En algunas formas de realización, el procesador 1800 también puede incluir un conjunto de una o más unidades de controlador de bus 1816 y un núcleo de agente del sistema 1810. La una o más unidades de controlador de bus 1816 gestionan un conjunto de buses periféricos, tales como uno o más buses de Interconexión de Componentes Periféricos (p. ej., PCI, PCI Express). El núcleo de agente del sistema 1810 proporciona funcionalidad de gestión para los distintos componentes del procesador. En algunas formas de realización, el núcleo del agente del sistema 1810 incluye uno o más controladores de memoria integrados 1814 para gestionar el acceso a diversos dispositivos de memoria externos (no ilustrados).

En algunas formas de realización, uno o más de los núcleos de procesador 1802A-1802N incluyen soporte para subprocesos múltiples simultáneos. En dicha forma de realización, el núcleo del agente del sistema 1810 incluye componentes para coordinar y operar los núcleos 1802A-1802N durante el procesamiento de subprocesos múltiples. El núcleo del agente del sistema 1810 puede incluir, además, una unidad de control de energía (PCU), que incluye lógica y componentes para regular el estado de energía de los núcleos del procesador 1802A-1802N y del procesador de gráficos 1808.

En algunas formas de realización, el procesador 1800 incluye, además, el procesador de gráficos 1808 para ejecutar operaciones de procesamiento de gráficos. En otras formas de realización, el procesador de gráficos 1808 se acopla con el conjunto de unidades de memoria caché compartidas 1806 y el núcleo del agente del sistema 1810, incluyendo los uno o más controladores de memoria integrados 1814. En algunas formas de realización, un controlador de pantalla 1811 está acoplado con el procesador de gráficos 1808 para impulsar la salida del procesador de gráficos a una o más pantallas acopladas. En algunas formas de realización, el controlador de pantalla 1811 puede ser un módulo separado acoplado con el procesador de gráficos mediante al menos una interconexión, o puede estar integrado dentro del procesador de gráficos 1808 o del núcleo del agente del sistema 1810.

En algunas formas de realización, se utiliza una unidad de interconexión basada en anillo 1812 para acoplar los componentes internos del procesador 1800. Sin embargo, se puede utilizar una unidad de interconexión alternativa, tal como una interconexión punto a punto, una interconexión conmutada u otras técnicas, incluyendo técnicas bien conocidas a este respecto. En algunas formas de realización, el procesador de gráficos 1808 se acopla con la interconexión de anillo 1812 mediante un enlace de E/S 1813.

El enlace de E/S, a modo de ejemplo, 1813 representa al menos una de las múltiples diversidades de interconexiones de E/S, incluyendo una interconexión de E/S en el paquete que facilita la comunicación entre diversos componentes del procesador y un módulo de memoria integrado de alto rendimiento 1818, tal como un módulo eDRAM. En algunas formas de realización, cada uno de los núcleos de procesador 1802A-1802N y del procesador de gráficos 1808 utilizan módulos de memoria integrados 1818 tal como una memoria caché de último nivel compartida.

En algunas formas de realización, los núcleos de procesador 1802A-1802N son núcleos homogéneos que ejecutan la misma arquitectura de conjunto de instrucciones. En otra forma de realización, los núcleos de procesador 1802A-1802N son heterogéneos en términos de arquitectura de conjunto de instrucciones (ISA), en donde uno o más de los núcleos de procesador 1802A-1802N ejecutan un primer conjunto de instrucciones, mientras que al menos uno de los otros núcleos ejecuta un subconjunto del primer conjunto de instrucciones o un conjunto de instrucciones diferente. En una forma de realización, los núcleos de procesador 1802A-1802N son heterogéneos en términos de microarquitectura, en donde uno o más núcleos que tienen un consumo de energía relativamente mayor se acoplan con uno o más núcleos de energía que tienen un consumo de energía más bajo. Además, el procesador 1800 se puede poner en práctica en uno o más circuitos integrados o como un circuito integrado SoC que tenga los componentes ilustrados, además, de otros componentes.

La Figura 19 es un diagrama de bloques de un procesador de gráficos 1900, que puede ser una unidad de procesamiento de gráficos discreto, o puede ser un procesador de gráficos integrado con una pluralidad de núcleos de procesamiento. En algunas formas de realización, el procesador de gráficos se comunica mediante una interfaz de E/S asignada en memoria con registros en el procesador de gráficos y con comandos ubicados en la memoria del procesador. En algunas formas de realización, el procesador de gráficos 1900 incluye una interfaz de memoria 1914 para acceder a la memoria. La interfaz de memoria 1914 puede ser una interfaz para la memoria local, una o más memoria caché interna, una o más memoria caché externa compartida y/o para la memoria del sistema.

En algunas formas de realización, el procesador de gráficos 1900 también incluye un controlador de pantalla 1902 para conducir datos de salida de visualización a un dispositivo de visualización 1920. El controlador de pantalla 1902 incluye hardware para los uno o más planos superpuestos para la visualización y composición de múltiples capas de vídeo o elementos de interfaz de usuario. En algunas formas de realización, el procesador de gráficos 1900 incluye un motor de códec de vídeo 1906 para codificar, decodificar o transcodificar medios hacia, desde o entre uno o más formatos de codificación de medios, incluyendo, sin limitación, formatos de grupos de expertos en imágenes en movimiento (MPEG) tal como MPEG-2, formatos de codificación avanzada de vídeo (AVC) tal como H.264/MPEG-4 AVC, así como la Sociedad de Ingenieros de Televisión y de Imágenes en Movimiento (SMPTE) 421M/VC-1 y el Grupo de Expertos Fotográficos Conjuntos (JPEG) tales como JPEG y JPEG de Movimiento (MJPEG).

En algunas formas de realización, el procesador de gráficos 1900 incluye un motor de transferencia de imágenes en bloque (BLIT) 1904 para realizar operaciones de rasterizador bidimensional (2D) que incluyen, por ejemplo, transferencias de bloque de límites de bits. Sin embargo, en una forma de realización, las operaciones de gráficos 2D se realizan utilizando uno o más componentes del motor de procesamiento de gráficos (GPE) 1910. En algunas formas de realización, GPE 1910 es un motor de cálculo para realizar operaciones de gráficos, incluyendo operaciones de gráficos tridimensionales (3D) y operaciones de medios.

En algunas formas de realización, GPE 1910 incluye una canalización 3D 1912 para realizar operaciones 3D, tal como renderizar imágenes y escenas tridimensionales utilizando funciones de procesamiento que actúan sobre formas de primitivos 3D (por ejemplo, rectángulo, triángulo, etc.). La canalización 3D 1912 incluye elementos programables y de funciones fijas que realizan varias tareas dentro del elemento y/o generan subprocesos de ejecución en un subsistema 3D/Medios 1915. Mientras que la canalización 3D 1912 se puede utilizar para realizar operaciones de medios, una forma de realización de GPE 1910 también incluye una canalización de medios 1916 que se utiliza específicamente para realizar operaciones de medios, tales como el posprocesamiento de vídeo y la mejora de imágenes.

En algunas formas de realización, la canalización de medios 1916 incluye funciones fijas o unidades lógicas programables para realizar una o más operaciones de medios especializadas, tales como aceleración de decodificación de vídeo, desentrelazado de vídeo y aceleración de codificación de vídeo en lugar de, o en función del motor códec de vídeo 1906. En algunas formas de realización, la canalización de medios 1916 incluye, además, una unidad de generación de subprocesos para generar subprocesos para su ejecución en Subsistema 3D/Media 1915. Los subprocesos generados realizan cálculos para las operaciones de medios en una o más unidades de ejecución de gráficos incluyendo en Subsistema 3D/Media 1915.

En algunas formas de realización, el Subsistema 3D/Media 1915 incluye lógica para ejecutar subprocesos generados por la canalización 3D 1912 y la canalización de medios 1916. En una forma de realización, las canalizaciones envían solicitudes de ejecución de subprocesos al Subsistema 3D/Media 1915, que incluye lógica de envío de subprocesos para arbitrar y enviar las diversas solicitudes a los recursos de ejecución de subprocesos disponibles. Los recursos de ejecución incluyen una matriz de unidades de ejecución de gráficos para procesar los subprocesos de medios y 3D. En algunas formas de realización, el Subsistema 3D/Media 1915 incluye una o más memoria caché interna para instrucciones y datos de subprocesos. En algunas formas de realización, el subsistema también incluye memoria compartida, incluyendo registros y memoria direccionable, para compartir datos entre subprocesos y para almacenar datos de salida.

Motor de procesamiento de gráficos

La Figura 20 es un diagrama de bloques de un motor de procesamiento de gráficos 2010 de un procesador de gráficos de conformidad con algunas formas de realización. En una forma de realización, el motor de procesamiento de gráficos

(GPE) 2010 es una versión del GPE 1910 que se muestra en la Figura 19. Los elementos de la Figura 20, que tienen los mismos números de referencia (o nombres) que los elementos de cualquier otra figura, en este documento, pueden operar o funcionar de manera similar a la descrita en otra parte del mismo, pero no se limitan a dicha circunstancia operativa. Por ejemplo, se ilustran la canalización 3D 1912 y la canalización de medios 1916 de la Figura 19. La canalización de medios 1916 es opcional en algunas formas de realización del GPE 2010 y es posible que no se incluya explícitamente dentro del GPE 2010. Por ejemplo, y en al menos una forma de realización, un procesador de imágenes y/o medios separado está acoplado al GPE 2010.

En algunas formas de realización, GPE 2010 se acopla o incluye un transmisor de comandos 2003, que proporciona un flujo de comandos a la canalización 3D 1912 y/o a las canalizaciones de medios 1916. En algunas formas de realización, el transmisor de comandos 2003 está acoplado con memoria, que puede ser memoria del sistema, o una o más de entre la memoria caché interna y la memoria caché compartida. En algunas formas de realización, el transmisor de comandos 2003 recibe comandos de la memoria y envía los comandos a la canalización 3D 1912 y/o a la canalización de medios 1916. Los comandos son directivas obtenidas de una memoria intermedia de anillo, que almacena comandos para la canalización 3D 1912 y para la canalización de medios 1916. En una forma de realización, la memoria intermedia de anillo puede incluir de manera adicional memorias intermedias de comandos por lotes que almacenan lotes de múltiples comandos. Los comandos para la canalización 3D 1912 también pueden incluir referencias a datos almacenados en la memoria, tales como, sin limitación, datos de vértice y de geometría para la canalización 3D 1912 y/o datos de imagen y objetos de memoria para la canalización de medios 1916. La canalización 3D 1912 y la canalización de medios 1916 procesan los comandos y los datos realizando operaciones mediante la lógica dentro de las canalizaciones respectivas o enviando uno o más subprocesos de ejecución a una matriz de núcleo de gráficos 2014.

En varias formas de realización, la canalización 3D 1912 puede ejecutar uno o más programas de sombreador, tales como sombreadores de vértices, sombreadores de geometría, sombreadores de píxeles, sombreadores de fragmentos, sombreadores de cálculo u otros programas de sombreador, al procesar las instrucciones y enviar subprocesos de ejecución a la matriz de núcleo de gráficos 2014. La matriz de núcleo de gráficos 2014 proporciona un bloque unificado de recursos de ejecución. La lógica de ejecución multiusuario (p. ej., unidades de ejecución) dentro de la matriz de núcleo gráfico 2014 incluye soporte para diversos lenguajes de sombreador de API 3D y puede ejecutar múltiples subprocesos de ejecución simultáneos asociados con múltiples sombreadores.

En algunas formas de realización, la matriz de núcleo de gráficos 2014 también incluye lógica de ejecución para realizar funciones de medios, tales como procesamiento de vídeo y/o de imagen. En una forma de realización, las unidades de ejecución incluyen, además, lógica de uso general que es programable para realizar operaciones computacionales paralelas de uso general, además, de operaciones de procesamiento de gráficos. La lógica de uso general puede realizar operaciones de procesamiento en paralelo o junto con la lógica de uso general dentro del (de los) núcleo(s) del procesador 107 de la Figura 17 o del núcleo 202A-202N tal como se ilustra en la Figura 18.

Los datos de salida generados por subprocesos que se ejecutan en la matriz de núcleo de gráficos 2014 pueden enviar datos a la memoria en una memoria intermedia de retorno unificada (URB) 2018. La memoria URB 2018 puede almacenar datos para múltiples subprocesos. En algunas formas de realización, la URB 2018 se puede utilizar para enviar datos entre diferentes subprocesos que se ejecutan en la matriz de núcleo de gráficos 2014. En algunas formas de realización, la URB 2018 se puede utilizar de manera adicional para la sincronización entre subprocesos en la matriz de núcleo de gráficos y la lógica de función fija dentro de la lógica de función compartida 2020.

En algunas formas de realización, la matriz de núcleos de gráficos 2014 es escalable, de modo que la matriz incluye un número variable de núcleos de gráficos, cada uno de los cuales tiene un número variable de unidades de ejecución en función de la potencia objetivo y del nivel de rendimiento de GPE 2010. En una forma de realización, los recursos de ejecución son dinámicamente escalables, de modo que los recursos de ejecución se pueden habilitar o deshabilitar según sea necesario.

La matriz de núcleo de gráficos 2014 se acopla con la lógica de función compartida 2020 que incluye múltiples recursos que se comparten entre los núcleos de gráficos en la matriz de núcleo de gráficos. Las funciones compartidas, dentro de la lógica de funciones compartidas 2020, son unidades lógicas de hardware que proporcionan una funcionalidad complementaria especializada a la matriz de núcleo de gráficos 2014. En varias formas de realización, la lógica de funciones compartida 2020 incluye, sin limitación, un muestreador 2021, matemáticas 2022 y lógica de comunicación entre subprocesos (ITC) 2023. Además, algunas formas de realización ponen en práctica una o más memorias caché 2025 dentro de la lógica de funciones compartidas 2020. Se pone en práctica una función compartida cuando la demanda de una función especializada dada es insuficiente para incluirla dentro de la matriz de núcleo de gráficos 2014. En su lugar, una instancia única de dicha función especializada se pone en práctica como una entidad independiente en la lógica de función compartida 2020 y se comparte entre los recursos de ejecución dentro de la matriz de núcleo de gráficos 2014. El conjunto preciso de funciones que se comparten entre la matriz de núcleo de gráficos 2014 y se incluyen dentro de la matriz de núcleo de gráficos 2014 varía entre formas de realización.

La Figura 21 es un diagrama de bloques de otra forma de realización de un procesador de gráficos 2100. Los elementos de la Figura 21, que tienen los mismos números de referencia (o nombres) que los elementos de cualquier

otra figura en este documento, pueden operar o funcionar de cualquier manera similar a que se describen en otra parte del mismo, pero no se limitan a dichas circunstancias operativas.

En algunas formas de realización, el procesador de gráficos 2100 incluye una interconexión de anillo 2102, un extremo frontal de canalización 2104, un motor de medios 2137 y núcleos de gráficos 2180A-2180N. En otras formas de realización, la interconexión en anillo 2102 acopla el procesador de gráficos a otras unidades de procesamiento, incluyendo otros procesadores de gráficos o uno o más núcleos de procesador de uso general. En algunas formas de realización, el procesador de gráficos es uno de los muchos procesadores integrados dentro de un sistema de procesamiento de multinúcleo.

En algunas formas de realización, el procesador de gráficos 2100 recibe lotes de comandos mediante la interconexión de anillo 2102. Los comandos entrantes son interpretados por un transmisor de comandos 2103 en el extremo frontal de canalización 2104. En algunas formas de realización, el procesador de gráficos 2100 incluye una lógica de ejecución escalable para realizar procesamiento de geometría en 3D y procesamiento de medios mediante los núcleos gráficos 2180A-2180N. Para los comandos de procesamiento de geometría en 3D, el transmisor de comandos 2103 proporciona comandos a la canalización de geometría 2136. Para al menos algunos comandos de procesamiento de medios, el transmisor de comandos 2103 proporciona los comandos a un extremo frontal de vídeo 2134, que se acopla con un motor de medios 2137. En algunas formas de realización, el motor de medios 2137 incluye un motor de calidad de vídeo (VQE) 2130 para el post-procesamiento de imágenes y de vídeo y un motor de codificación/decodificación (MFX) 533 de multiformato para proporcionar codificación y decodificación de datos multimedia acelerada por hardware. En algunas formas de realización, la canalización de geometría 2136 y el motor de medios 2137 generan, cada uno, subprocesos de ejecución para los recursos de ejecución de subproceso proporcionados por al menos un núcleo gráfico 2180A.

En algunas formas de realización, el procesador de gráficos 2100 incluye recursos de ejecución de subprocesos escalables que presentan núcleos modulares 2180A-2180N (a veces denominados segmentos de núcleo), cada uno de los cuales tiene múltiples subnúcleos 2150A-2150N, 2160A-2160N (a veces denominados subsegmentos de núcleo). En algunas formas de realización, el procesador gráfico 2100 puede tener cualquier número de núcleo gráfico 2180A a 2180N. En algunas formas de realización, el procesador de gráficos 2100 incluye un núcleo gráfico 2180A que tiene al menos un primer subnúcleo 2150A y un segundo subnúcleo 2160A. En otras formas de realización, el procesador de gráficos es un procesador de baja potencia con un único subnúcleo (por ejemplo, 2150A). En algunas formas de realización, el procesador de gráficos 2100 incluye múltiples núcleos de gráficos 2180A-2180N, cada uno de los cuales incluye un conjunto de primeros subnúcleos 2150A-2150N y un conjunto de segundos subnúcleos 2160A-2160N. Cada subnúcleo del conjunto de primeros subnúcleos 2150A-2150N incluye al menos un primer conjunto de unidades de ejecución 2152A-2152N y muestreadores de medios/texturas 2154A-2154N. Cada subnúcleo del conjunto de segundos subnúcleos 2160A-2160N incluye al menos un segundo conjunto de unidades de ejecución 2162A-2162N y muestreadores 2164A-2164N. En algunas formas de realización, cada subnúcleo 2150A-2150N, 2160A-2160N comparte un conjunto de recursos compartidos 2170A-2170N. En algunas formas de realización, los recursos compartidos incluyen memoria caché compartida y lógica de operación de píxeles. También se pueden incluir otros recursos compartidos en las diversas formas de realización del procesador de gráficos.

Unidades de ejecución

La Figura 22 ilustra la lógica de ejecución de subprocesos 2200 que incluye una matriz de elementos de procesamiento empleados en algunas formas de realización de un GPE. Los elementos de la Figura 22, que tienen los mismos números de referencia (o nombres) que los elementos de cualquier otra figura del presente documento, pueden operar o funcionar de cualquier manera similar a la descrita en otro lugar del mismo, pero no se limitan a ello.

En algunas formas de realización, la lógica de ejecución de subprocesos 2200 incluye un procesador de sombreador 2202, un expedidor de subprocesos 2204, memoria caché de instrucciones 2206, una matriz de unidades de ejecución escalable que incluye una pluralidad de unidades de ejecución 2208A-2208N, un muestreador 2210, una memoria caché de datos 2212, y un puerto de datos 2214. En una forma de realización, la matriz de unidades de ejecución escalable puede escalar, de manera dinámica, habilitando o deshabilitando una o más unidades de ejecución (por ejemplo, cualquiera de las unidades de ejecución 2208A, 2208B, 2208C, 2208D, hasta 2208N-1 y 2208N) en función de los requisitos computacionales de una carga de trabajo. En una forma de realización, los componentes incluidos están interconectados mediante un tejido funcional de interconexión que enlaza con cada uno de los componentes. En algunas formas de realización, la lógica de ejecución de subprocesos 2200 incluye una o más conexiones a la memoria, tal como la memoria del sistema o la memoria caché, mediante una o más memoria caché de instrucciones 2206, puerto de datos 2214, muestreador 2210 y unidades de ejecución 2208A-2208N. En algunas formas de realización, cada unidad de ejecución (p. ej., 2208A) es una unidad computacional independiente programable de uso general que es capaz de ejecutar múltiples subprocesos de hardware de manera simultánea mientras procesa múltiples elementos de datos en paralelo para cada subproceso. En varias formas de realización, la matriz de unidades de ejecución 2208A-2208N es escalable para incluir cualquier número de unidades de ejecución individuales.

En algunas formas de realización, las unidades de ejecución 2208A-2208N se utilizan principalmente para ejecutar programas de sombreador. Un procesador de sombreador 2202 puede procesar los diversos programas de

sombreador y enviar subprocesos de ejecución asociados con los programas de sombreador mediante un expedidor de subprocesos 2204. En una forma de realización, el expedidor de subprocesos incluye lógica para arbitrar las solicitudes de inicio de subprocesos de las canalizaciones de gráficos y medios y para instanciar los subprocesos solicitados en una o más unidades de ejecución en las unidades de ejecución 2208A-2208N. Por ejemplo, la canalización de geometría (por ejemplo, 2136 de la Figura 21) puede enviar sombreadores de vértices, teselaciones o geometría a la lógica de ejecución de subprocesos 2200 (Figura 22) para su procesamiento. En algunas formas de realización, el expedidor de subprocesos 2204 también puede procesar solicitudes de generación de subprocesos en tiempo de ejecución desde los programas de sombreador en ejecución.

En algunas formas de realización, las unidades de ejecución 2208A-2208N admiten un conjunto de instrucciones que incluyen un soporte nativo para muchas instrucciones de sombreado de gráficos 3D estándar, de modo que los programas de sombreador de bibliotecas de gráficos (por ejemplo, Direct 3D y OpenGL) se ejecutan con una traslación mínima. Las unidades de ejecución admiten procesamiento de vértices y de geometría (p. ej., programas de vértices, programas de geometría, sombreadores de vértices), procesamiento de píxeles (p. ej., sombreadores de píxeles, sombreadores de fragmentos) y procesamiento de uso general (p. ej., sombreadores de cálculo y de medios). Cada una de las unidades de ejecución 2208A-2208N es capaz de ejecución de datos múltiples de instrucción única (SIMD) de emisión múltiple y la operación de subprocesos múltiples permite un entorno de ejecución eficiente frente a accesos de memoria de mayor latencia. Cada subproceso de hardware, dentro de cada unidad de ejecución, tiene un fichero de registro de gran ancho de banda dedicado y un estado de subproceso independiente asociado. La ejecución es de múltiples emisiones por reloj para canalizaciones capaces de operaciones de coma flotante de precisión simple y doble, capacidad de bifurcación SIMD, operaciones lógicas, operaciones trascendentales y otras operaciones varias. Mientras espera datos de la memoria o de una de las funciones compartidas, la lógica de dependencia dentro de las unidades de ejecución 2208A-2208N hace que un subproceso de espera entre en suspensión hasta que se devuelvan los datos solicitados. Mientras el subproceso en espera está inactivo, los recursos de hardware pueden dedicarse a procesar otros subprocesos. 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 sombreador, incluyendo un sombreador de vértices diferente.

Cada unidad de ejecución en las unidades de ejecución 2208A-2208N opera en matrices de elementos de datos. El número de elementos de datos es el "tamaño de ejecución" o el número de canales para la instrucción. Un canal de ejecución es una unidad lógica de ejecución para el acceso, enmascaramiento y control de flujo de elementos de datos dentro de las instrucciones. El número de canales puede ser independiente del número de Unidades de Lógica Aritmética (ALUs) físicas o Unidades de Coma Flotante (FPUs) para un procesador de gráficos en particular. En algunas formas de realización, las unidades de ejecución 2208A-2208N admiten tipos de datos enteros y de coma flotante. Las ALUs pueden incluir circuitos sumadores (por ejemplo, sistema 700, circuito sumador 750) de conformidad con formas de realización del presente diseño.

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 separados de 64 bits (elementos de datos de tamaño de palabra cuádruple (QW)), ocho elementos de datos empaquetados separados de 32 bits (elementos de datos de tamaño de palabra doble (DW)), dieciséis elementos de datos empaquetados separados de 16 bits (elementos de datos de tamaño de palabras (W)), o treinta y dos elementos de datos separados de 8 bits (elementos de datos de tamaño byte (B)). Sin embargo, son posibles diferentes anchos de vector y tamaños de registro.

Una o más memorias caché de instrucciones internas (por ejemplo, 2206) se incluyen en la lógica de ejecución de subprocesos 2200 para almacenar en memoria caché las instrucciones de subprocesos para las unidades de ejecución. En algunas formas de realización, se incluyen una o más memorias caché de datos (por ejemplo, 2212) para almacenar, en memoria caché, los datos del subproceso durante la ejecución del subproceso. En algunas formas de realización, se incluye un muestreador 2210 para proporcionar muestreo de textura para operaciones en 3D y muestreo de medios para operaciones de medios. En algunas formas de realización, el muestreador 2210 incluye una funcionalidad especializada de muestreo de texturas o medios para procesar datos de texturas o medios durante el proceso de muestreo antes de proporcionar los datos muestreados a una unidad de ejecución.

Durante la ejecución, las canalizaciones de gráficos y medios envían solicitudes de iniciación de subprocesos a la lógica de ejecución de subprocesos 2200 mediante la lógica de generación y envío de subprocesos. Una vez que un grupo de objetos geométricos ha sido procesado y rasterizado en datos de píxeles, se invoca la lógica del procesador de píxeles (por ejemplo, lógica de sombreador de píxeles, lógica de sombreador de fragmentos, etc.) dentro del procesador de sombreador 2202 para calcular más información de salida y hacer que se realice la escritura de los resultados a las superficies de salida (por ejemplo, memorias intermedias de color, memorias intermedias de profundidad, memorias intermedias de plantilla, etc.). En algunas formas de realización, un sombreador de píxeles o un sombreador de fragmentos calcula los valores de los diversos atributos de vértice que se van a interpolar en el objeto rasterizado. En algunas formas de realización, la lógica del procesador de píxeles, dentro del procesador de sombreador 2202, ejecuta luego un programa de sombreador de fragmentos o píxeles proporcionado por la interfaz

de programación de aplicaciones (API). Para ejecutar el programa de sombreador, el procesador de sombreador 2202 envía subprocesos a una unidad de ejecución (por ejemplo, 2208A) mediante el expedidor de subprocesos 2204. En algunas formas de realización, el sombreador de píxeles 2202 utiliza la lógica de muestreo de textura en el muestreador 2210 para acceder a los datos de textura en los mapas de textura almacenados en memoria. Las operaciones aritméticas sobre los datos de textura y los datos de geometría de entrada calculan los datos de color de los píxeles para cada fragmento geométrico, o descartan uno o más píxeles del procesamiento posterior.

En algunas formas de realización, el puerto de datos 2214 proporciona un mecanismo de acceso a la memoria para que la lógica de ejecución de subprocesos 2200 transmita datos procesados a la memoria para su procesamiento en una canalización de salida del procesador de gráficos. En algunas formas de realización, el puerto de datos 2214 incluye o se acopla a una o más memorias caché (por ejemplo, memoria caché de datos 2212) para almacenar datos en memoria caché para acceder a la memoria a través del puerto de datos.

La Figura 23 es un diagrama de bloques que ilustra los formatos de instrucción 2300 de un procesador de gráficos según algunas formas de realización. En una o más formas de realización, las unidades de ejecución del procesador de gráficos admiten un conjunto de instrucciones que tiene instrucciones en múltiples formatos. Las casillas con líneas continuas ilustran los componentes que, por lo general, se incluyen en una instrucción de unidad de ejecución, mientras que las líneas discontinuas incluyen componentes que son opcionales o que solamente se incluyen en un subconjunto de las instrucciones. En algunas formas de realización, el formato de instrucción 2300 descrito e ilustrado son macroinstrucciones, ya que son instrucciones suministradas a la unidad de ejecución, a diferencia de las microoperaciones que resultan de la decodificación de instrucciones una vez que se procesa la instrucción.

En algunas formas de realización, las unidades de ejecución del procesador de gráficos admiten de forma nativa instrucciones en un formato de instrucción de 128 bits 2310. Un formato de instrucción compactado de 64 bits 2330 está disponible para algunas instrucciones en función de la instrucción seleccionada, las opciones de instrucción y el número de operandos. El formato de instrucción de 128 bits nativo 2310 proporciona acceso a todas las opciones de instrucción, mientras que algunas opciones y operaciones están restringidas en el formato de instrucción de 64 bits 2330. Las instrucciones nativas disponibles en el formato de instrucción de 64 bits 2330 varían según la forma de realización. En algunas formas de realización, la instrucción se compacta en parte utilizando un conjunto de valores de índice en un campo de índice 2313. El hardware de la unidad de ejecución hace referencia a un conjunto de tablas de compactación basadas en los valores de índice y utiliza las salidas de la tabla de compactación para reconstruir una instrucción nativa en el formato de instrucción de 128 bits 2310.

Para cada formato, el código opcode de instrucción 2312 define la operación que debe realizar la unidad de ejecución. Las unidades de ejecución, ejecutan cada instrucción en paralelo mediante los múltiples elementos de datos de cada operando. Por ejemplo, en respuesta a una instrucción de adición, la unidad de ejecución realiza una operación de adición simultánea en cada canal de color que representa un elemento de textura o un elemento de imagen. De forma predeterminada, la unidad de ejecución ejecuta cada instrucción en todos los canales de datos de los operandos. En algunas formas de realización, el campo de control de instrucciones 2314 permite el control sobre ciertas opciones de ejecución, tales como la selección de canales (p. ej., predicación) y el orden de los canales de datos (p. ej., *swizzle*). Para instrucciones en el formato de instrucción de 128 bits 2310, un campo de tamaño de ejecución 2316 limita el número de canales de datos que se ejecutarán en paralelo. En algunas formas de realización, el campo de tamaño de ejecución 2316 no está disponible para su uso en el formato de instrucción compacta de 64 bits 2330.

Algunas instrucciones de la unidad de ejecución tienen hasta tres operandos, incluyendo dos operandos de origen, src0 2320, src1 2322 y un destino 2318. En algunas formas de realización, las unidades de ejecución admiten instrucciones de destino dual, en donde uno de los destinos está implícito. Las instrucciones de manipulación de datos pueden tener un tercer operando de origen (por ejemplo, SRC2 2234), en donde el código de opcode de instrucción 2312 determina el número de operandos de origen. El último operando fuente de una instrucción puede ser un valor inmediato (por ejemplo, codificado de forma rígida) que se pasa con la instrucción.

En algunas formas de realización, el formato de instrucción de 128 bits 2310 incluye un campo de modo de acceso/dirección 2326 que especifica, por ejemplo, si se utiliza el modo de direccionamiento de registro directo o el modo de direccionamiento de registro indirecto. Cuando se utiliza el modo de direccionamiento de registro directo, la dirección de registro de uno o más operandos se proporciona directamente mediante bits en la instrucción.

En algunas formas de realización, el formato de instrucción de 128 bits 2310 incluye un campo de modo de acceso/dirección 2326, que especifica un modo de dirección y/o un modo de acceso para la instrucción. En una forma de realización, el modo de acceso se utiliza para definir una alineación de acceso a datos para la instrucción. Algunas formas de realización admiten modos de acceso que incluyen un modo de acceso alineado de 16 bytes y un modo de acceso alineado de 1 byte, en 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 utilizar direccionamiento alineado por bytes para los operandos de origen y destino y cuando está en un segundo modo, la instrucción puede utilizar direccionamiento alineado por 16 bytes para todos los operandos de origen y de destino.

En una forma de realización, la parte del modo de dirección del campo de modo de acceso/dirección 2326 determina si la instrucción es para utilizar el direccionamiento directo o indirecto. Cuando se utiliza el modo de direccionamiento de registro directo, los bits en la instrucción proporcionan directamente la dirección de registro de uno o más operandos. Cuando se utiliza el modo de direccionamiento de registro indirecto, la dirección de registro de uno o más operandos se puede calcular en función de un valor de registro de dirección y de un campo inmediato de dirección en la instrucción.

En algunas formas de realización, las instrucciones se agrupan en función de los campos de bits del código de operación 2312 para simplificar la decodificación del código de operación 2340. Para un código de opcode de 8 bits, los bits 4, 5 y 6 permiten que la unidad de ejecución determine el tipo de código opcode. La agrupación precisa de códigos de opcode que se muestra es simplemente un ejemplo. En algunas formas de realización, un grupo de códigos opcode de lógica y movimiento 2342 incluye instrucciones lógicas y de movimiento de datos (por ejemplo, mover (mov), comparar (cmp)). En algunas formas de realización, el grupo de movimiento y lógica 2342 comparte los cinco bits más significativos (MSB), en donde las instrucciones de movimiento (mov) tienen la forma de 0000xxxxb y las instrucciones lógicas tienen la forma de 0001xxxxb. Un grupo de instrucciones de control de flujo 2344 (por ejemplo, denominada, salto (jmp)) incluye instrucciones en forma de 0010xxxxb (por ejemplo, 0x20). Un grupo de instrucciones varias 2346 incluye una combinación de instrucciones, incluyendo instrucciones de sincronización (por ejemplo, esperar, enviar) en forma de 0011xxxxb (por ejemplo, 0x30). Un grupo de instrucciones matemáticas paralelas 2348 incluye instrucciones aritméticas por componentes (por ejemplo, sumar, multiplicar (mul)) en forma de 0100xxxxb (por ejemplo, 0x40). El grupo matemático en paralelo 2348 realiza las operaciones aritméticas en paralelo mediante los canales de datos. El grupo matemático vectorial 2350 incluye instrucciones aritméticas (p. ej., dp4) en la forma de 0101xxxxb (p. ej., 0x50). El grupo de matemática vectorial realiza operaciones aritméticas, tales como cálculos de productos escalar en operandos vectoriales.

Canalización de gráficos

La Figura 24 es un diagrama de bloques de otra forma de realización de un procesador de gráficos 2400. Los elementos de la Figura 24, que tienen los mismos números de referencia (o nombres) que los elementos de cualquier otra figura, en este documento, pueden operar o funcionar de cualquier manera similar a la que se describen en otra parte del presente documento, pero no se limitan a dicha circunstancia operativa.

En algunas formas de realización, el procesador de gráficos 2400 incluye una canalización de gráficos 2420, una canalización de medios 2430, un motor de visualización 2440, una lógica de ejecución de subprocesos 2450 y una canalización de salida de renderizado 2470. En algunas formas de realización, el procesador de gráficos 2400 es un procesador de gráficos dentro de un sistema de procesamiento multinúcleo que incluye uno o más núcleos de procesamiento de uso general. El procesador de gráficos se controla mediante escrituras de registro en uno o más registros de control (no ilustrados) o mediante comandos emitidos al procesador de gráficos 2400 mediante una interconexión de anillo 2402. En algunas formas de realización, la interconexión de anillo 2402 acopla el procesador de gráficos 2400 a otros componentes de procesamiento, tales como otros procesadores de gráficos o procesadores de uso general. Los comandos de la interconexión de anillo 2402 son interpretados por un transmisor de comandos 2403, que proporciona instrucciones a los componentes individuales de la canalización de gráficos 2420 o la canalización de medios 2430.

En algunas formas de realización, el transmisor de comandos 2403 dirige la operación de un buscador de vértices 2405 que realiza la lectura de datos de vértices de la memoria y ejecuta comandos de procesamiento de vértices proporcionados por el transmisor de comandos 2403. En algunas formas de realización, el buscador de vértices 2405 proporciona datos de vértices a un sombreador de vértices 2407, que realiza operaciones de transformación e iluminación del espacio coordinado en cada vértice. En algunas formas de realización, el buscador de vértices 2405 y el sombreador de vértices 2407 ejecutan instrucciones de procesamiento de vértices enviando subprocesos de ejecución a las unidades de ejecución 2452A-2452B mediante un expedidor de subprocesos 2431.

En algunas formas de realización, las unidades de ejecución 2452A-2452B son una matriz de procesadores vectoriales que tienen un conjunto de instrucciones para realizar operaciones de gráficos y medios. En algunas formas de realización, las unidades de ejecución 2452A-2452B tienen una memoria caché L1 2451 adjunta que es específica para cada matriz o compartida entre las matrices. La memoria caché se puede configurar como una memoria caché de datos, una memoria caché de instrucciones o una sola memoria caché que está dividida para contener datos e instrucciones en diferentes particiones.

En algunas formas de realización, la canalización de gráficos 2420 incluye componentes de teselado para realizar un teselado acelerado por hardware de objetos en 3D. En algunas formas de realización, un sombreador de casco programable 2411 configura las operaciones de teselado. Un sombreador de dominio programable 2417 proporciona una evaluación de extremo posterior de la salida del teselado. Un teselador 2413 opera en la dirección del sombreador de casco 2411 y contiene una lógica de uso especial para generar un conjunto de objetos geométricos detallados basados en un modelo geométrico aproximado que se proporciona como entrada a la canalización de gráficos 2420. En algunas formas de realización, si no se utiliza el teselado, los componentes de teselado (p. ej., el sombreador de casco 2411, el teselador 2413 y el sombreador de dominio 2417) se pueden omitir.

- En algunas formas de realización, un sombreador de geometría 2419 puede procesar objetos geométricos completos mediante uno o más subprocesos enviados a las unidades de ejecución 2452A-2452B, o puede proceder directamente al recortador 2429. En algunas formas de realización, el sombreador de geometría opera en la totalidad de objetos geométricos, en lugar de vértices o parches de vértices como en etapas anteriores de la canalización de gráficos. Si el teselado está deshabilitado, el sombreador de geometría 2419 recibe una entrada del sombreador de vértices 2407. En algunas formas de realización, el sombreador de geometría 2419 puede programarse mediante un programa de sombreado de geometría para realizar el teselado de geometría si las unidades de teselado están deshabilitadas.
- Antes de la rasterización, un recortador 2429 procesa datos de vértice. El recortador 2429 puede ser un recortador de función fija o un recortador programable que tiene funciones de recorte y de sombreador de geometría. En algunas formas de realización, un rasterizador y un componente de prueba de profundidad 2473, en la canalización de salida de renderizado 2470, envía sombreadores de píxeles para convertir los objetos geométricos en sus representaciones por píxel. En algunas formas de realización, la lógica del sombreador de píxeles se incluye en la lógica de ejecución de subprocesos 2450. En algunas formas de realización, una aplicación puede omitir el rasterizador y el componente de prueba de profundidad 2473 y acceder a los datos de vértice sin rasterizar mediante una unidad de flujo de salida 2423.
- El procesador de gráficos 2400 tiene un bus de interconexión, un tejido funcional de interconexión o algún otro mecanismo de interconexión que permite la etapa de datos y mensajes entre los componentes principales del procesador. En algunas formas de realización, las unidades de ejecución 2452A-2452B y memorias caché(s) asociada(s) 2451, muestreador de textura y medios 2454, y memoria caché de textura/muestreador 2458 se interconectan mediante un puerto de datos 2456 para realizar un acceso a la memoria y comunicarse con los componentes de canalización de salida de procesamiento del procesador. En algunas formas de realización, el muestreador 2454, las memorias caché 2451, 2458 y las unidades de ejecución 2452A-2452B tienen rutas de acceso a la memoria separadas.
- En algunas formas de realización, la canalización de salida de renderizado 2470 contiene un rasterizador y un componente de prueba de profundidad 2473 que convierte objetos basados en vértices en una representación asociada basada en píxeles. En algunas formas de realización, la lógica del rasterizador incluye una unidad de ventana/enmascarador para realizar una función fija de rasterización de líneas y triángulos. Una memoria caché de procesamiento 2478 asociada y una memoria caché de profundidad 2479 también están disponibles en algunas formas de realización. Un componente de operaciones de píxeles 2477 realiza operaciones basadas en píxeles en los datos, aunque en algunos casos, las operaciones de píxeles asociadas con operaciones en 2D (por ejemplo, transferencias de imágenes de bloques de bits con combinación) son realizadas por el motor 2D 2441, o sustituidas en el momento de la visualización por el controlador de pantalla 2443 utilizando planos de visualización superpuestos. En algunas formas de realización, una memoria caché L3 compartida 2475 está disponible para todos los componentes gráficos, lo que permite compartir datos sin el uso de la memoria principal del sistema.
- En algunas formas de realización, la canalización de medios del procesador de gráficos 2430 incluye un motor de medios 2437 y una interfaz de vídeo 2434. En algunas formas de realización, la interfaz de vídeo 2434 recibe comandos de canalización del transmisor de comandos 2403. En algunas formas de realización, la canalización de medios 2430 incluye un transmisor de comandos separado. En otras formas de realización, el extremo frontal de vídeo 2434 procesa los comandos de medios antes de enviar el comando al motor de medios 2437. En algunas formas de realización, el motor de medios 2437 incluye la funcionalidad de generación de subprocesos para generar subprocesos para enviarlos a la lógica de ejecución de subprocesos 2450 a través del expedidor de subprocesos 2431.
- En algunas formas de realización, el procesador de gráficos 2400 incluye un motor de visualización 2440. En algunas formas de realización, el motor de visualización 2440 es externo al procesador 2400 y se acopla con el procesador de gráficos mediante la interconexión en anillo 2402, o algún otro bus o tejido funcional de interconexión. En algunas formas de realización, el motor de visualización 2440 incluye un motor 2D 2441 y un controlador de pantalla 2443. En algunas formas de realización, el motor de visualización 2440 contiene lógica de uso especial capaz de operar, de manera independiente, de la canalización en 3D. En algunas formas de realización, el controlador de pantalla 2443 se acopla con un dispositivo de visualización (no ilustrado), que puede ser un dispositivo de visualización integrado en el sistema, tal como en un ordenador portátil, o en un dispositivo de visualización externo conectado mediante un conector de dispositivo de visualización.
- En algunas formas de realización, la canalización de gráficos 2420 y la canalización de medios 2430 se pueden configurar para realizar operaciones basadas en múltiples interfaces de programación de gráficos y medios y no son específicas de ninguna interfaz de programación de aplicaciones (API). En algunas formas de realización, el software del controlador para el procesador de gráficos traslada las denominadas API que son específicas de una biblioteca multimedia o de gráficos particular en comandos que pueden ser procesados por el procesador de gráficos. En algunas formas de realización, se proporciona soporte para Open Graphics Library (OpenGL), Open Computing Language (OpenCL) y/o Vulkan Graphics and Compute API, todos del Grupo Khronos. En algunas formas de realización, también se puede proporcionar soporte para la biblioteca Direct3D de Microsoft Corporation. En algunas formas de realización, se puede admitir una combinación de estas bibliotecas. También se puede proporcionar soporte para la Biblioteca de

visión por ordenador de código abierto (OpenCV). También se admitiría una futura API con una canalización 3D compatible si se puede realizar una asignación desde la canalización de la futura API a la canalización del procesador de gráficos.

5 Programación de canalización de gráficos

La Figura 25A es un diagrama de bloques que ilustra un formato de comando de procesador de gráficos 2500 según algunas formas de realización. La Figura 25B es un diagrama de bloques que ilustra una secuencia de comandos del procesador de gráficos 2510 según una forma de realización. Las casillas con líneas continuas en la Figura 25A ilustran los componentes que se suelen incluir en un comando de gráficos, mientras que las líneas discontinuas incluyen componentes que son opcionales o que solamente se incluyen en un subconjunto de los comandos de gráficos. El formato de comando de procesador de gráficos a modo de ejemplo 2500 de la Figura 25A incluye campos de datos para identificar un cliente objetivo 2502 del comando, un código de operación (opcode) 2504 y los datos pertinentes 2506 para dicho comando. Un sub-opcode 2505 y un tamaño de comando 2508 también se incluyen en algunos comandos.

En algunas formas de realización, el cliente 2502 especifica la unidad cliente del dispositivo gráfico que procesa los datos de comando. En algunas formas de realización, un analizador de comandos del procesador de gráficos examina el campo de cliente de cada comando para condicionar el procesamiento posterior del comando y enrutar los datos del comando a la unidad de 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 de medios. Cada unidad de cliente tiene una canalización de procesamiento correspondiente que procesa los comandos. Una vez que la unidad de cliente recibe el comando, la unidad de cliente realiza la lectura del código opcode 2504 y, si está presente, el sub-opcode 2505 para determinar la operación a realizar. La unidad cliente ejecuta el comando utilizando la información del campo de datos 2506. Para algunos comandos, se espera que un tamaño de comando explícito 2508 especifique el tamaño del comando. En algunas formas de realización, el analizador de comandos determina de manera automática el tamaño de al menos algunos de los comandos basándose en el código opcode del comando. En algunas formas de realización, los comandos se alinean mediante múltiplos de una palabra doble.

El diagrama de flujo en la Figura 25B muestra una secuencia de comando de procesador de gráficos a modo de ejemplo 2510. En algunas formas de realización, el software o firmware de un sistema de procesamiento de datos que presenta una forma de realización de un procesador de gráficos utiliza una versión de la secuencia de comando que se muestra para configurar, ejecutar y finalizar un conjunto de operaciones gráficas. Se ilustra y describe una secuencia de comandos de muestra solamente con fines a modo de ejemplo, ya que las formas de realización no se limitan a estos comandos específicos o a esta secuencia de comandos. Además, los comandos pueden emitirse como un lote de comandos en una secuencia de comandos, de modo que el procesador de gráficos procesará la secuencia de comandos al menos parcialmente concurrente.

En algunas formas de realización, la secuencia de comandos del procesador de gráficos 2510 puede comenzar con un comando de vaciado de canalización 2512 para hacer que cualquier canalización de gráficos activa complete los comandos actualmente pendientes para la canalización. En algunas formas de realización, la canalización 3D 2522 y la canalización de medios 2524 no funcionan de manera simultánea. El vaciado de la canalización se realiza para que la canalización de gráficos activa complete los comandos pendientes. En respuesta a un vaciado de canalización, el analizador de comandos para el procesador de gráficos detendrá el procesamiento de comandos hasta que los motores de dibujo activos completen las operaciones pendientes y se invaliden las memorias caché de lectura pertinentes. De manera opcional, cualquier dato en la memoria caché de renderizado que esté marcada como 'sucia' se puede vaciar en la memoria. En algunas formas de realización, el comando de vaciado de canalización 2512 se puede utilizar para la sincronización de la canalización o antes de colocar el procesador de gráficos en un estado de baja potencia.

En algunas formas de realización, se utiliza un comando de selección de canalización 2513 cuando una secuencia de comandos requiere que el procesador de gráficos conmute explícitamente entre canalizaciones. En algunas formas de realización, se requiere un comando de selección de canalización 2513 solamente 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 formas de realización, se requiere un comando de vaciado de canalización 2512 inmediatamente antes de una conmutación de canalización mediante el comando de selección de canalización 2513.

En algunas formas de realización, un comando de control de canalización 2514 configura una canalización de gráficos para su funcionamiento y se utiliza para programar la canalización en 3D 2522 y la canalización de medios 2524. En algunas formas de realización, el comando de control de canalización 2514 configura el estado de canalización para la canalización activa. En una forma de realización, el comando de control de canalización 2514 se utiliza 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 formas de realización, los comandos para el estado de la memoria intermedia de retorno 2516 se utilizan para configurar un conjunto de memorias intermedias de retorno para que las canalizaciones respectivas realicen la escritura de datos. Algunas operaciones de canalización requieren la asignación, selección o configuración de una o más memorias intermedias de retorno en las que las operaciones realizan la escritura de datos intermedios durante el procesamiento. En algunas formas de realización, el procesador de gráficos también utiliza una o más memorias intermedias de retorno para almacenar datos de salida y realizar una comunicación entre subprocesos. En algunas formas de realización, configurar el estado de la memoria intermedia de retorno 2516 incluye seleccionar el tamaño y el número de memorias intermedias de retorno a utilizar para un conjunto de operaciones de canalización.

Los comandos restantes en la secuencia de comandos difieren en función de la canalización activa para las operaciones. Basándose en una determinación de canalización 2520, la secuencia de comandos se adapta a la canalización 3D 2522 comenzando con el estado de canalización 3D 2530 o la canalización de medios 2524 que comienza en el estado de canalización de medios 2540.

Los comandos para configurar el estado de canalización 3D 2530 incluyen comandos de configuración de estado 3D para estado de memoria intermedia de vértice, estado de elemento de vértice, estado de color constante, estado de memoria intermedia de profundidad y otras variables de estado que deben configurarse antes de que se procesen los comandos de primitivos 3D. Los valores de estos comandos se determinan, al menos en parte, en función de la API 3D particular en uso. En algunas formas de realización, los comandos 2530 de estado de canalización 3D también pueden deshabilitar u omitir de manera selectiva ciertos elementos de canalización si dichos elementos no se utilizarán.

En algunas formas de realización, el comando de primitivo 3D 2532 se utiliza para enviar primitivos 3D para que sean procesados por la canalización 3D. Los comandos y los parámetros asociados que se pasan al procesador de gráficos mediante el comando de primitivo 3D 2532 se reenvían a la función de búsqueda de vértices en la canalización de gráficos. La función de búsqueda de vértice utiliza los datos de comando de primitivo 3D 2532 para generar estructuras de datos de vértice. Las estructuras de datos de vértice se almacenan en una o más memorias intermedias de retorno. En algunas formas de realización, el comando de primitivo 3D 2532 se utiliza para realizar operaciones de vértice en primitivos 3D mediante sombreadores de vértice. Para procesar sombreadores de vértices, la canalización 3D 2522 envía subprocesos de ejecución de sombreadores a las unidades de ejecución del procesador de gráficos.

En algunas formas de realización, la canalización 3D 2522 se activa mediante un comando o evento de ejecución 2534. En algunas formas de realización, una escritura de registro inicia la ejecución de un comando. En algunas formas de realización, la ejecución se inicia mediante un comando "go" o "kick" en la secuencia de comandos. En una forma de realización, la ejecución del comando se inicia utilizando un comando de sincronización de canalización para vaciar la secuencia de comandos a través de la canalización de gráficos. La canalización 3D realizará el procesamiento de geometría para los primitivos 3D. Una vez que se completan las operaciones, los objetos geométricos resultantes se rasterizan y el motor de píxeles colorea los píxeles resultantes. También se pueden incluir comandos adicionales para controlar el sombreado de píxeles y las operaciones de extremo posterior de píxeles para dichas operaciones.

En otras formas de realización, la secuencia de comandos del procesador de gráficos 2510 sigue la ruta de canalización de medios 2524 cuando se realizan operaciones de medios. En general, el uso específico y la forma de programación para la canalización de medios 2524 depende de las operaciones de medios o de cálculo a realizar. Las operaciones específicas de decodificación de medios pueden descargarse a la canalización de medios durante la decodificación de medios. En algunas formas de realización, la canalización de medios también se puede omitir y la decodificación de medios se puede realizar, en su totalidad o en parte, utilizando los recursos proporcionados por uno o más núcleos de procesamiento de uso general. En una forma de realización, la canalización de medios también incluye elementos para operaciones de unidades de procesador de gráficos de uso general (GPGPU), en donde el procesador de gráficos se utiliza para realizar operaciones de vector SIMD utilizando programas de sombreador computacional que no están relacionados explícitamente con la representación de primitivos de gráficos.

En algunas formas de realización, la canalización de medios 2524 se configura de manera similar a la canalización 3D 2522. Un conjunto de comandos para configurar el estado de la canalización de medios 2540 se envían o se colocan en una cola de comandos antes de los comandos de objeto de medios 2542. En algunas formas de realización, los comandos para el estado de canalización de medios 2540 incluyen datos para configurar los elementos de canalización de medios que se utilizarán para procesar los objetos de medios. Lo que antecede incluye datos para configurar la lógica de decodificación de vídeo y de codificación de vídeo dentro de la canalización de medios, tal como el formato de codificación o de decodificación. En algunas formas de realización, los comandos para el estado de canalización de medios 2540 también admiten el uso de uno o más punteros a elementos de estado "indirectos" que contienen un lote de configuraciones de estado.

En algunas formas de realización, los comandos de objetos de medios 2542 proporcionan punteros a objetos de medios para que los procese la canalización de medios. Los objetos de medios incluyen memorias intermedias que contienen datos de vídeo para ser procesados. En algunas formas de realización, todos los estados de la canalización de medios deben ser válidos antes de emitir un comando de objeto de medios 2542. Una vez que se configura el estado de la canalización y los comandos de objetos de medios 2542 se encuentran en cola de espera, la canalización

de medios 2524 se activa mediante un comando de ejecución 2544 o un evento de ejecución equivalente (p. ej., escritura de registro). La salida de la canalización de medios 2524 puede a continuación procesarse posteriormente mediante operaciones proporcionadas por la canalización 3D 2522 o la canalización de medios 2524. En algunas formas de realización, las operaciones GPGPU se configuran y ejecutan de manera similar a las operaciones de medios.

Arquitectura de software de gráficos

La Figura 26 ilustra una arquitectura de software de gráficos a modo de ejemplo para un sistema de procesamiento de datos 2600 según algunas formas de realización. En algunas formas de realización, la arquitectura de software incluye una aplicación de gráficos 3D 2610, un sistema operativo 2620 y al menos un procesador 2630. En algunas formas de realización, el procesador 2630 incluye un procesador de gráficos 2632 y uno o más núcleos de procesador de uso general 2634. La aplicación de gráficos 2610 y el sistema operativo 2620 se ejecutan cada uno en la memoria del sistema 2650 del sistema de procesamiento de datos.

En algunas formas de realización, la aplicación de gráficos 3D 2610 contiene uno o más programas de sombreador que incluyen instrucciones de sombreador 2612. Las instrucciones del 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 2614 en un lenguaje de máquina adecuado para su ejecución por el núcleo del procesador de uso general 2634. La aplicación también incluye objetos gráficos 2616 definidos por datos de vértice.

En algunas formas de realización, el sistema operativo 2620 es un sistema operativo de Microsoft® Windows® de Microsoft Corporation, un sistema operativo patentado similar a UNIX o un sistema operativo similar a UNIX de código abierto que utiliza una variante del kernel de Linux. El sistema operativo 2620 puede soportar una API de gráficos 2622 tal como la API de Direct3D, la API de OpenGL o la API de Vulkan. Cuando la API de Direct3D está en uso, el sistema operativo 2620 utiliza un compilador de sombreador de extremo frontal 2624 para compilar cualquier instrucción de sombreador 2612 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 sombreador. En algunas formas de realización, los sombreadores de alto nivel se compilan en sombreadores de bajo nivel durante la compilación de la aplicación de gráficos 3D 2610. En algunas formas de realización, las instrucciones de sombreador 2612 se proporcionan en una forma intermedia, tal como una versión de la Representación Intermedia Portátil Estándar (SPIR) utilizada por la API de Vulkan.

En algunas formas de realización, el controlador de gráficos de modo usuario 2626 contiene un compilador de sombreador de extremo posterior 2627 para convertir las instrucciones de sombreador 2612 en una representación específica de hardware. Cuando la API de OpenGL está en uso, las instrucciones de sombreador 2612 en el lenguaje de alto nivel GLSL se pasan a un controlador de gráficos de modo de usuario 2626 para su compilación. En algunas formas de realización, el controlador de gráficos, en modo de usuario 2626, utiliza las funciones de modo kernel del sistema operativo 2628 para comunicarse con un controlador de gráficos en modo kernel 2629. En algunas formas de realización, el controlador de gráficos en modo kernel 2629 se comunica con el procesador de gráficos 2632 para enviar comandos e instrucciones.

Puestas en práctica del Núcleo IP

Uno o más aspectos de al menos una forma de realización pueden ponerse en práctica 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 diversas lógicas dentro del procesador. Cuando se realiza la lectura por una máquina, las instrucciones pueden hacer que la máquina obtenga la lógica para realizar las técnicas descritas en este documento. Dichas representaciones, conocidas como "núcleos de IP", son unidades de lógica reutilizables para un circuito integrado que pueden almacenarse en un medio tangible legible por máquina tal como un modelo de hardware que describe la estructura del circuito integrado. El modelo de hardware se puede suministrar a varios clientes o instalaciones de fabricación, que cargan el modelo de hardware en máquinas de fabricación que obtiene el circuito integrado. El circuito integrado se puede fabricar de manera que el circuito realice las operaciones descritas en asociación con cualquiera de las formas de realización descritas en el presente documento.

La Figura 27 es un diagrama de bloques que ilustra un sistema de desarrollo de núcleo de IP 2700 que puede utilizarse para fabricar un circuito integrado para realizar operaciones de conformidad con una forma de realización. El sistema de desarrollo de núcleo IP 2700 puede utilizarse para generar diseños modulares reutilizables que pueden incorporarse a un diseño más grande o utilizarse para obtener un circuito integrado completo (por ejemplo, un circuito integrado SOC). Una instalación de diseño 2730 puede generar una simulación de software 2710 de un diseño de núcleo de IP en un lenguaje de programación de alto nivel (por ejemplo, C/C++). La simulación de software 2710 se puede utilizar para diseñar, probar y verificar el comportamiento del núcleo de IP utilizando un modelo de simulación 2712. El modelo de simulación 2712 puede incluir simulaciones funcionales, de comportamiento y/o de temporización. A continuación, se puede crear o sintetizar un diseño de nivel de transferencia de registro (RTL) 2715 a partir del

modelo de simulación 2712. El diseño RTL 2715 es una abstracción del comportamiento del circuito integrado que modela el flujo de señales digitales entre registros de hardware, incluyendo la lógica asociada realizada utilizando las señales digitales modeladas. Además de un diseño RTL 2715, también se pueden crear, diseñar o sintetizar diseños de nivel inferior a nivel de lógica o a nivel de transistor. Por lo tanto, los datos particulares del diseño inicial y de la simulación pueden variar.

El diseño RTL 2715 o equivalente puede sintetizarse aún más mediante la instalación de diseño en un modelo de hardware 2720, que puede estar en un lenguaje de descripción de hardware (HDL), o en alguna otra representación de datos de diseño físico. El HDL puede simularse o probarse más para verificar el diseño del núcleo de IP. El diseño del núcleo de IP se puede almacenar para entregarlo a una instalación de fabricación de terceros 2765 utilizando una memoria no volátil 2740 (por ejemplo, un disco duro, una memoria instantánea o cualquier medio de almacenamiento no volátil). De manera alternativa, el diseño del núcleo de IP puede transmitirse (por ejemplo, mediante Internet) a través de una conexión por cable 2750 o de una conexión inalámbrica 2760. La instalación de fabricación 2765 puede entonces obtener un circuito integrado que se basa, al menos en parte, en el diseño del núcleo de IP. El circuito integrado obtenido puede configurarse para realizar operaciones de conformidad con al menos una forma de realización descrita en este documento.

Sistema, a modo de ejemplo, en un circuito integrado físico

Las Figuras 28-30 ilustran ejemplos de circuitos integrados y de procesadores de gráficos asociados que pueden obtenerse utilizando uno o más núcleos de IP, de conformidad con varias formas de realización descritas en este documento. Además de lo que se ilustra, se pueden incluir otras lógicas y circuitos, incluyendo procesadores/núcleos de gráficos adicionales, controladores de interfaz periféricos o núcleos de procesador de uso general.

La Figura 28 es un diagrama de bloques que ilustra un sistema, a modo de ejemplo, en un circuito integrado 2800 que puede fabricarse utilizando uno o más núcleos de IP, según una forma de realización. El circuito integrado, a modo de ejemplo 2800, incluye uno o más procesadores de aplicaciones 2805 (por ejemplo, CPUs), al menos un procesador de gráficos 2810, y puede incluir de manera adicional un procesador de imágenes 2815 y/o un procesador de vídeo 2820, cualquiera de los cuales puede ser un Núcleo IP modular procedente de la misma o de múltiples instalaciones de diseño diferentes. El circuito integrado 2800 incluye un periférico o lógica de bus que incluye un controlador USB 2825, un controlador UART 2830, un controlador SPI/SDIO 2835 y un controlador I2S/I2C 2840. Además, el circuito integrado puede incluir un dispositivo de visualización 2845 acoplado a uno o más de entre un controlador de interfaz multimedia de alta definición (HDMI) 2850 y una interfaz de pantalla de Interfaz de Procesador de Aplicaciones Móviles (MIPI) 2855. El almacenamiento puede ser proporcionado por un subsistema de memoria instantánea 2860 que incluya memoria instantánea y un controlador de memoria instantánea. La interfaz de memoria se puede proporcionar mediante un controlador de memoria 2865 para acceder a dispositivos de memoria SDRAM o SRAM. Algunos circuitos integrados incluyen, además, un motor de seguridad incorporado 2870.

La Figura 29 es un diagrama de bloques que ilustra un procesador gráfico 2910 a modo de ejemplo de un sistema en un circuito integrado que puede fabricarse utilizando uno o más núcleos de IP, según una forma de realización. El procesador de gráficos 2910 puede ser una variante del procesador de gráficos 2810 de la Figura 28. El procesador de gráficos 2910 incluye un procesador de vértice 2905 y uno o más procesadores de fragmentos 2915A-2915N (por ejemplo, 2915A, 2915B, 2915C, 2915D, hasta 2915N-1 y 2915N). El procesador de gráficos 2910 puede ejecutar diferentes programas de sombreador mediante una lógica separada, de modo que el procesador de vértices 2905 esté optimizado para ejecutar operaciones para programas de sombreador de vértices, mientras que los uno o más procesadores de fragmentos 2915A-2915N ejecutan operaciones de sombreador de fragmentos (por ejemplo, píxeles) para programas de sombreador de fragmentos o píxeles. El procesador de vértices 2905 realiza la etapa de procesamiento de vértices de la canalización de gráficos 3D y genera primitivos y datos de vértices. Los procesadores de fragmentos 2915A-2915N utilizan los datos de primitivos y de vértice generados por el procesador de vértice 2905 para obtener una memoria intermedia de tramas que se muestra en un dispositivo de visualización. En una forma de realización, los procesadores de fragmentos 2915A-2915N están optimizados para ejecutar programas de sombreador de fragmentos según lo dispuesto en la API de OpenGL, que se puede utilizar para realizar operaciones similares tal como un programa de sombreador de píxeles según lo dispuesto en la API Direct 3D.

El procesador de gráficos 2910 incluye, además, una o más unidades de gestión de memoria (26Us) 2920A-2920B, la memoria(s) caché 2925A-2925B y la interconexión(es) de circuito 2930A-2930B. Las una o más 26U(s) 2920A-2920B proporcionan la asignación de direcciones virtuales a físicas para el procesador de gráficos 2910, incluyendo el procesador de vértices 2905 y/o el(los) procesador(es) de fragmentos 2915A-2915N, que pueden hacer referencia a datos de vértice o imagen/textura almacenados en la memoria, además, de los datos de vértice o de imagen/textura almacenados en una o más memorias caché 2925A-2925B. En una forma de realización, las una o más 26U 2920A-2920B pueden sincronizarse con otras 26Us dentro del sistema, incluyendo una o más 26Us asociadas con uno o más procesadores de aplicaciones 2805, procesadores de imagen 2815 y/o procesadores de vídeo 2820 de la Figura 28, de manera que cada procesador 2805-2820 pueda participar en un sistema de memoria virtual compartida o unificada. La(s) interconexión(es) de uno o más circuitos 2930A-2930B permiten que el procesador de gráficos 2910 interactúe con otros núcleos IP dentro del circuito integrado SoC, ya sea mediante un bus interno del SoC o mediante una conexión directa, según las formas de realización.

La Figura 30 es un diagrama de bloques que ilustra un procesador de gráficos a modo de ejemplo adicional 3010 de un sistema en un circuito integrado que puede obtenerse utilizando uno o más núcleos de IP, según una forma de realización. El procesador de gráficos 3010 puede ser una variante del procesador de gráficos 2810 de la Figura 28. El procesador de gráficos 3010 incluye una o más unidades 26U(s) 2920A-2920B, la memoria(s) caché 2925A-2925B y la interconexión(es) de circuito 2930A-2930B del circuito integrado 2900 de la Figura 29.

El procesador de gráficos 3010 incluye uno o más núcleos de sombreador 3015A-3015N (por ejemplo, 3015A, 3015B, 3015C, 3015D, 3015E, 3015F, hasta 3015N-1 y 3015N), que proporciona una arquitectura de núcleo de sombreador unificada en donde un único núcleo o tipo de núcleo puede ejecutar todos los tipos de código de sombreador programable, incluyendo el código de programa de sombreador para poner en práctica sombreadores de vértices, sombreadores de fragmentos y/o sombreadores de cálculo. El número exacto de núcleos de sombreador presentes puede variar entre formas de realización y puestas en práctica. Además, el procesador de gráficos 3010 incluye un gestor de tareas entre núcleos 3005, que actúa como un expedidor de subprocesos para enviar subprocesos de ejecución a uno o más núcleos de sombreador 3015A-3015N y una unidad de mosaico 3018 para acelerar las operaciones de mosaico para el renderizado basado en mosaico, en donde las operaciones de renderizado para un escenario operativo se subdividen en el espacio de la imagen, por ejemplo, para utilizar la coherencia espacial local dentro de un escenario operativo o para optimizar el uso de las memorias caché internas.

La descripción y los dibujos que anteceden han de considerarse en un sentido ilustrativo y no restrictivo. Los expertos en esta técnica comprenderán que se pueden realizar diversas modificaciones y cambios en las formas de realización descritas en el presente documento sin desviarse por ello del alcance más amplio de la invención tal como se establece en las reivindicaciones adjuntas.

REIVINDICACIONES

1. Un aparato que comprende:

5 un circuito generador de números aleatorios uniformes, URNG, (710) para generar números aleatorios uniformes; y
un circuito sumador (750) que incluye una pluralidad de sumadores, estando el circuito sumador (750) acoplado al
circuito URNG (710), en donde los números aleatorios uniformes generados por el circuito URNG (710) se transmiten
con comunicaciones (711-713) a cualquier sumador de entre la pluralidad de sumadores del circuito sumador (750),
10 acelerando el circuito sumador (750), por hardware, la generación de números aleatorios gaussianos para el
aprendizaje máquina, en donde el circuito sumador (750) comprende:

un primer sumador de entre la pluralidad de sumadores, estando el primer sumador configurado para sumar los
primeros y segundos números aleatorios uniformes procedentes del circuito URNG (710) para generar una primera
15 salida del primer sumador, y para sumar otros dos números aleatorios uniformes para generar una segunda salida del
primer sumador después de que el primer sumador haya generado la primera salida;

un segundo sumador de entre la pluralidad de sumadores, estando el segundo sumador configurado para sumar un
tercer número aleatorio uniforme procedente del circuito URNG (710) con la primera salida para generar una primera
20 salida del segundo sumador; y

un N-ésimo sumador de entre la pluralidad de sumadores, estando el N-ésimo sumador configurado para sumar otro
número aleatorio uniforme procedente del circuito URNG (710) con una primera salida de un sumador anterior de entre
la pluralidad de sumadores para generar un primer número aleatorio gaussiano.

2. El aparato según la reivindicación 1, que comprende, además, un multiprocesador de gráficos y/o un núcleo
GPGPU, en donde el circuito sumador (750) se pone en práctica en el multiprocesador de gráficos y/o en el núcleo
GPGPU.

3. El aparato según la reivindicación 1, en donde el primer número aleatorio gaussiano se genera durante un período
de tiempo que es aproximadamente igual a un retardo de tiempo para cada sumador multiplicado por N sumadores.

4. El aparato según la reivindicación 1, en donde el segundo sumador está destinado a sumar otros dos números
aleatorios uniformes para generar una segunda salida del segundo sumador después de que el segundo sumador
35 haya generado la primera salida.

5. El aparato según la reivindicación 4, en donde el N-ésimo sumador está destinado a sumar otro número aleatorio
uniforme procedente del circuito URNG (710) con una segunda salida del sumador anterior para generar un segundo
número aleatorio gaussiano.

6. Un método para generar números aleatorios gaussianos que comprende:

utilizar un circuito generador de números aleatorios uniformes, URNG, para generar (602) números aleatorios
uniformes, en donde los números aleatorios uniformes generados por el circuito URNG se transmiten con
45 comunicaciones a cualquier sumador de entre una pluralidad de sumadores de un circuito sumador;

sumar (604), con un primer sumador del circuito sumador, números aleatorios uniformes primero y segundo,
procedentes del circuito URNG, para generar una primera salida del primer sumador;

50 sumar (622), con el primer sumador, otros dos números aleatorios uniformes para generar una segunda salida del
primer sumador después de que el primer sumador haya generado la primera salida;

sumar (606), con un segundo sumador del circuito sumador, un tercer número aleatorio uniforme, procedente del
circuito URNG, con la primera salida para generar una primera salida del segundo sumador; y

55 sumar, con un N-ésimo sumador del circuito sumador, otro número aleatorio uniforme, procedente del circuito URNG,
con una primera salida de un sumador anterior para generar un primer número aleatorio gaussiano para un aprendizaje
máquina basado en la teoría bayesiana.

7. El método según la reivindicación 6, en donde el primer número aleatorio gaussiano se genera durante un período
de tiempo que es aproximadamente igual a un retardo de tiempo de cada sumador multiplicado por N sumadores.

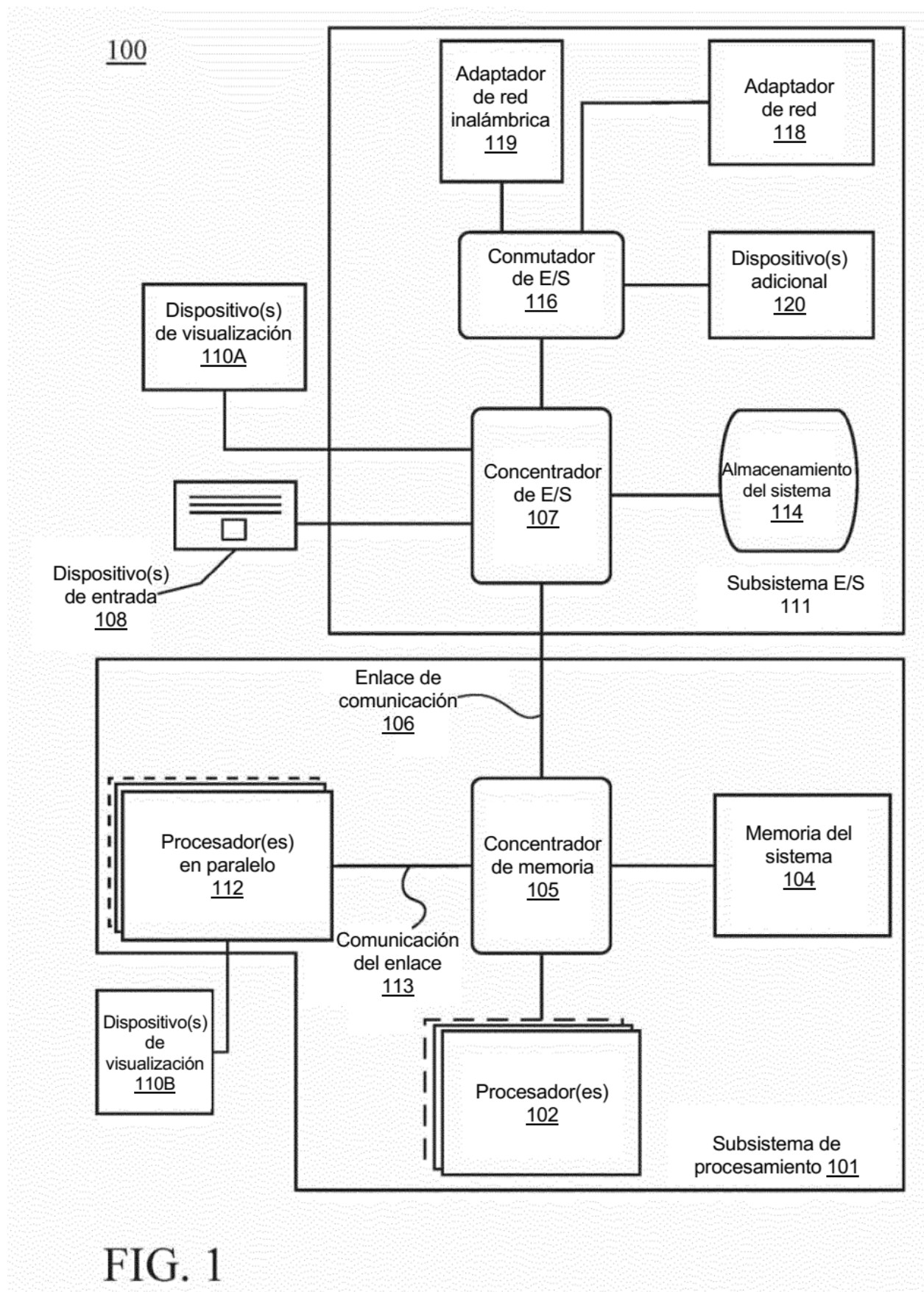
8. El método según la reivindicación 6, que comprende, además:

65 sumar, con el segundo sumador, otros dos números aleatorios uniformes para generar una segunda salida del
segundo sumador después de que el segundo sumador haya generado la primera salida.

9. El método (600) según la reivindicación 6, que comprende, además:

5 sumar (624), con el N-ésimo sumador, otro número aleatorio uniforme, procedente del circuito URNG, con una segunda salida del sumador anterior para generar un segundo número aleatorio gaussiano.

10. Un aparato que comprende una pluralidad de medios para realizar un método según cualquiera de las reivindicaciones 6 a 9.



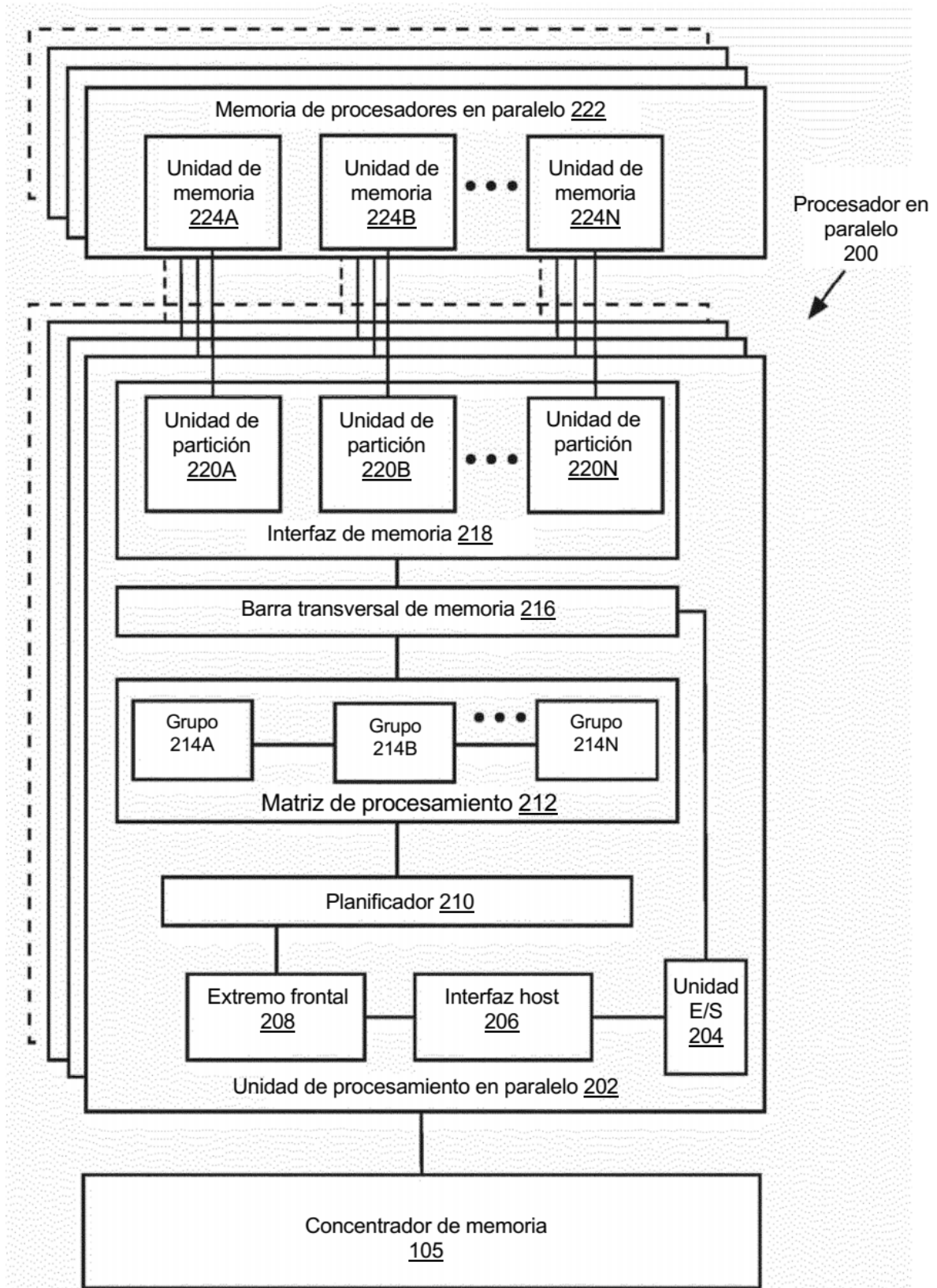
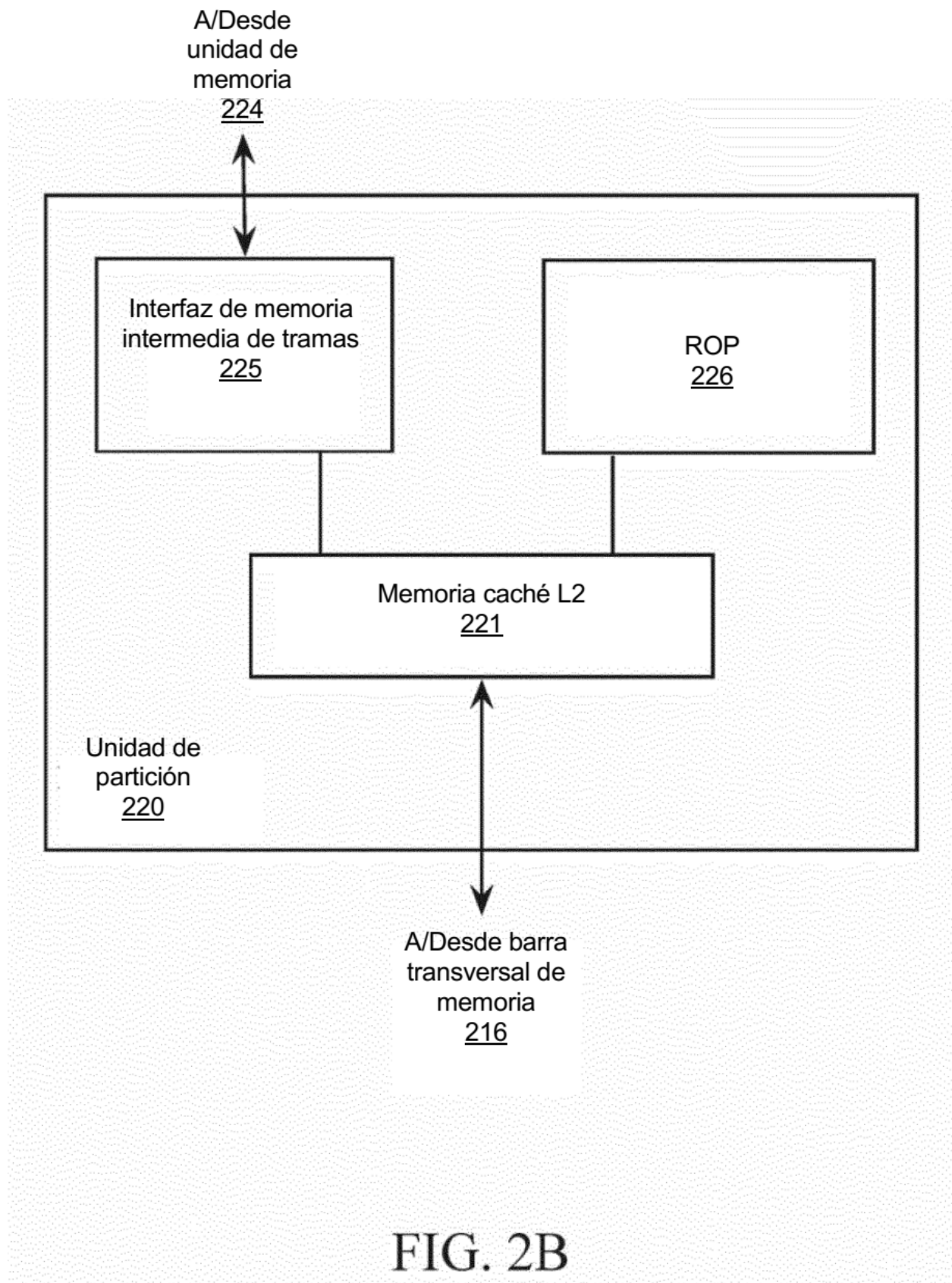


FIG. 2A



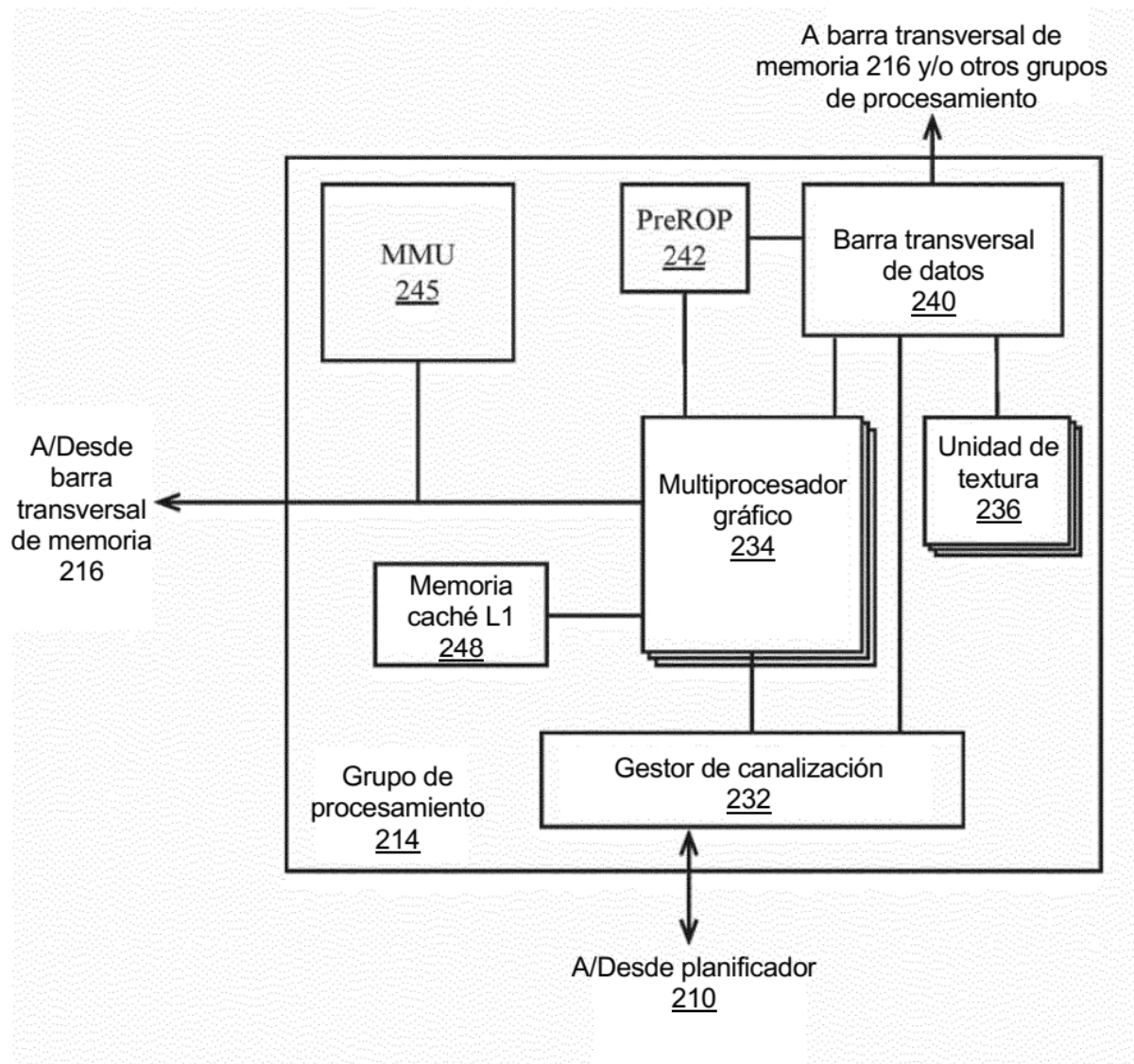


FIG. 2C

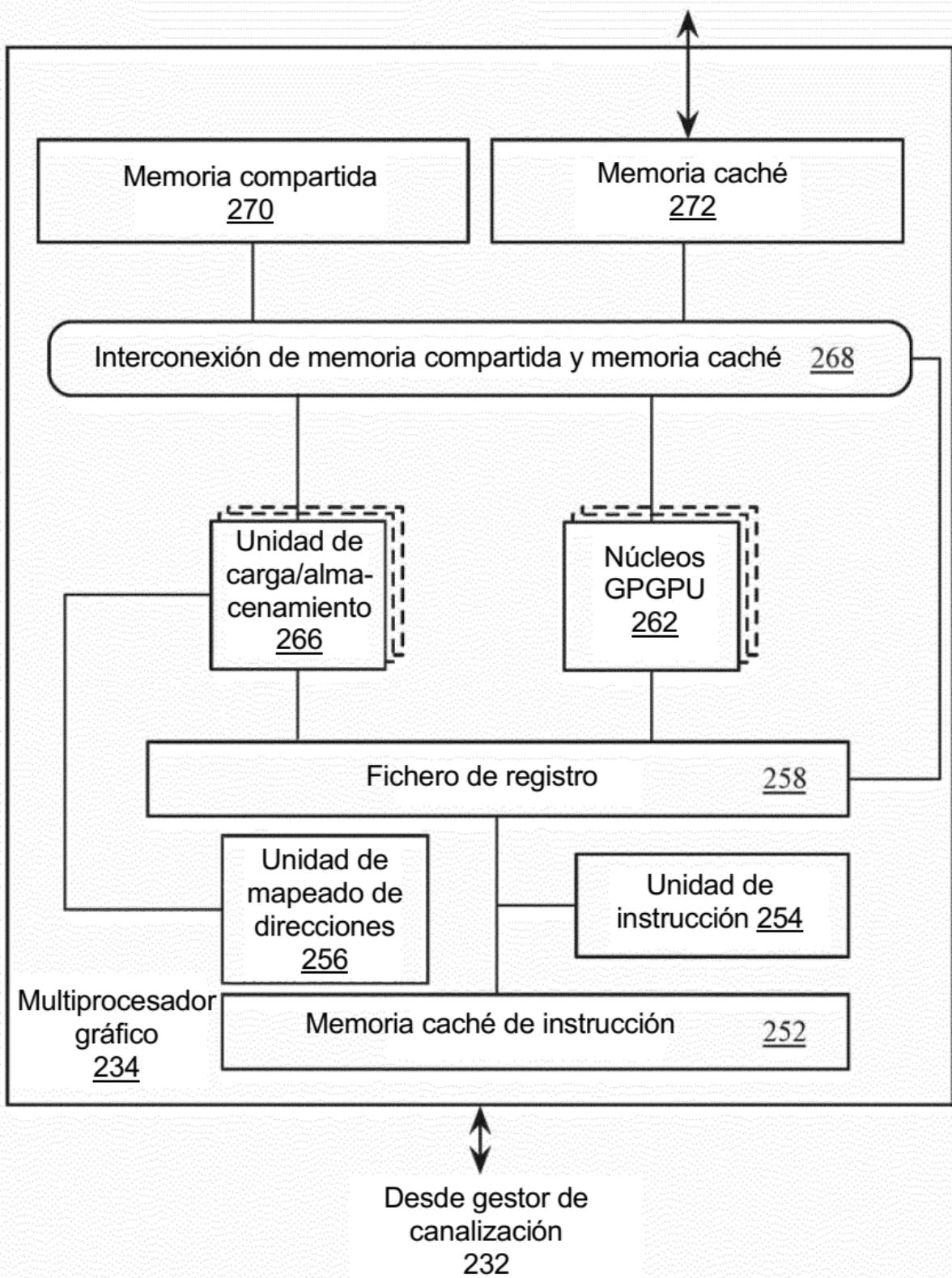


FIG. 2D

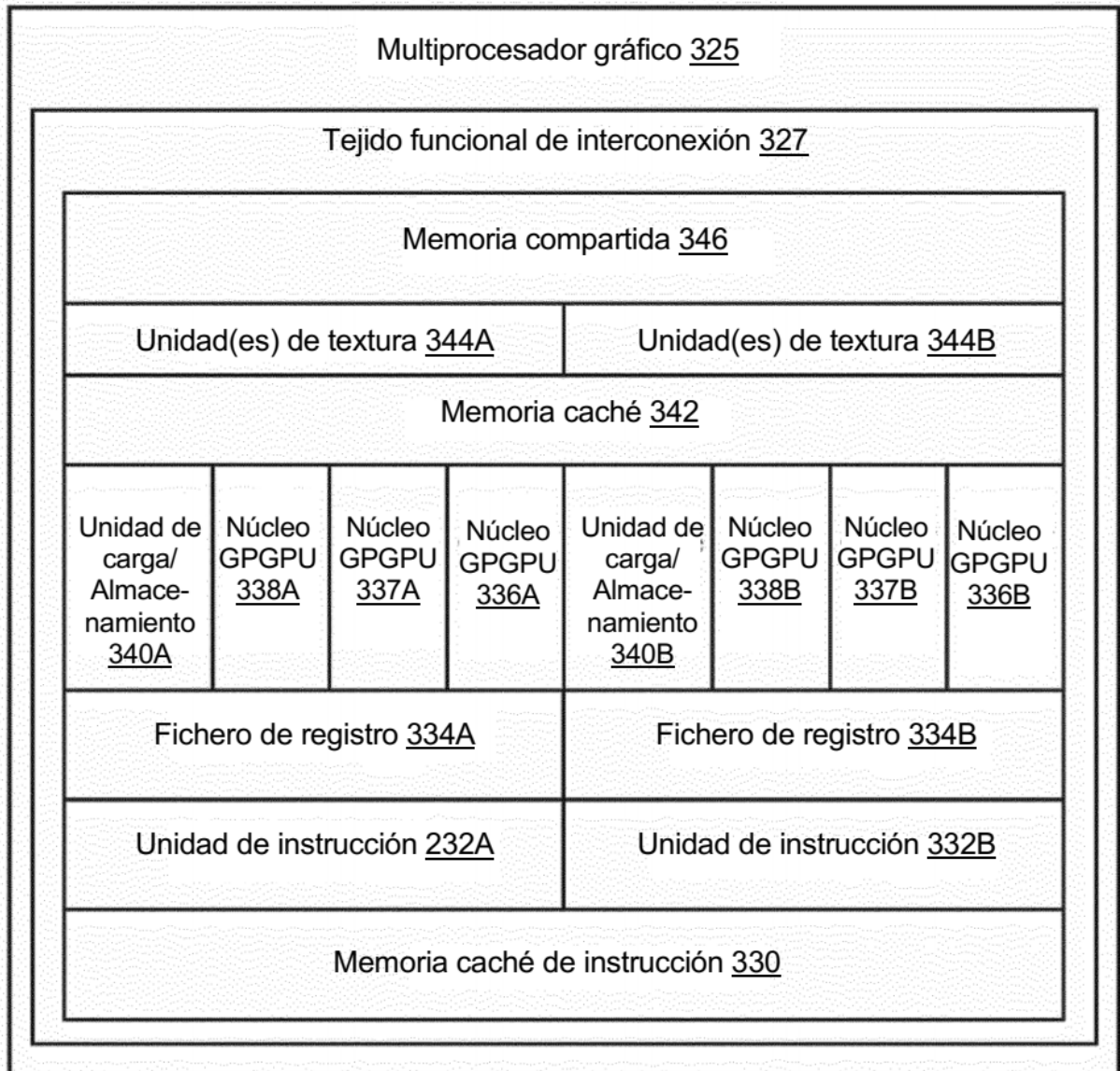


FIG. 3A

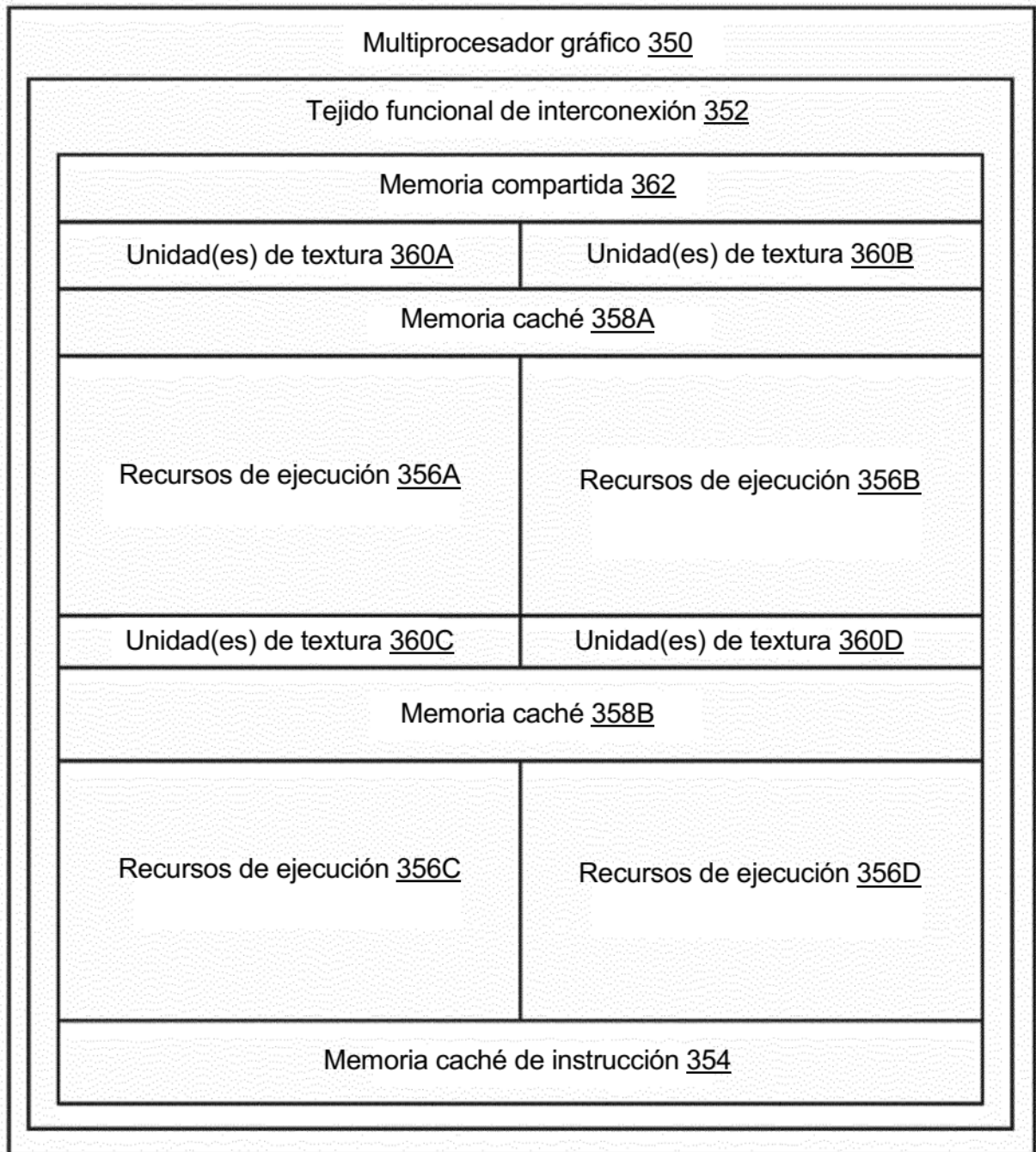


FIG. 3B

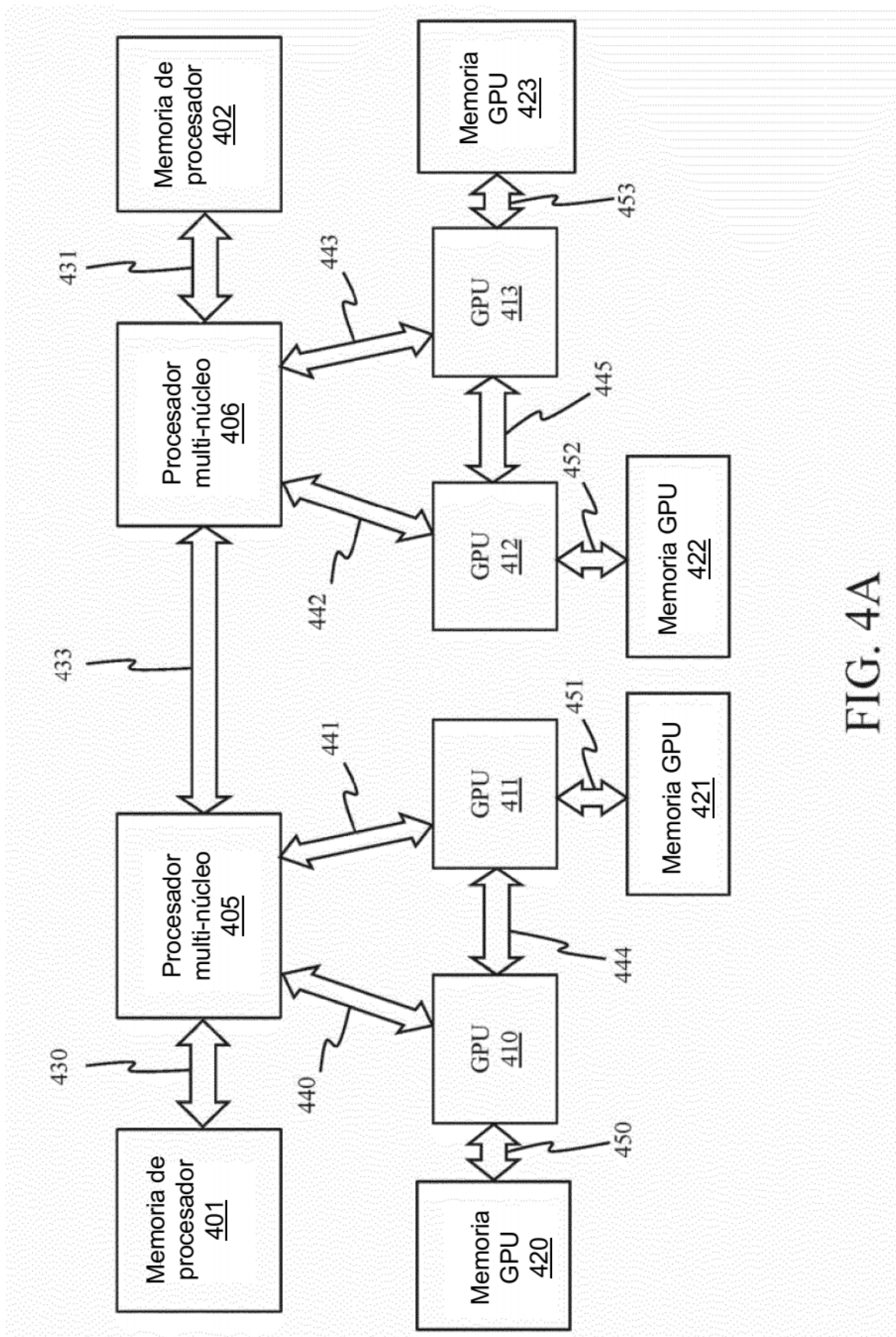


FIG. 4A

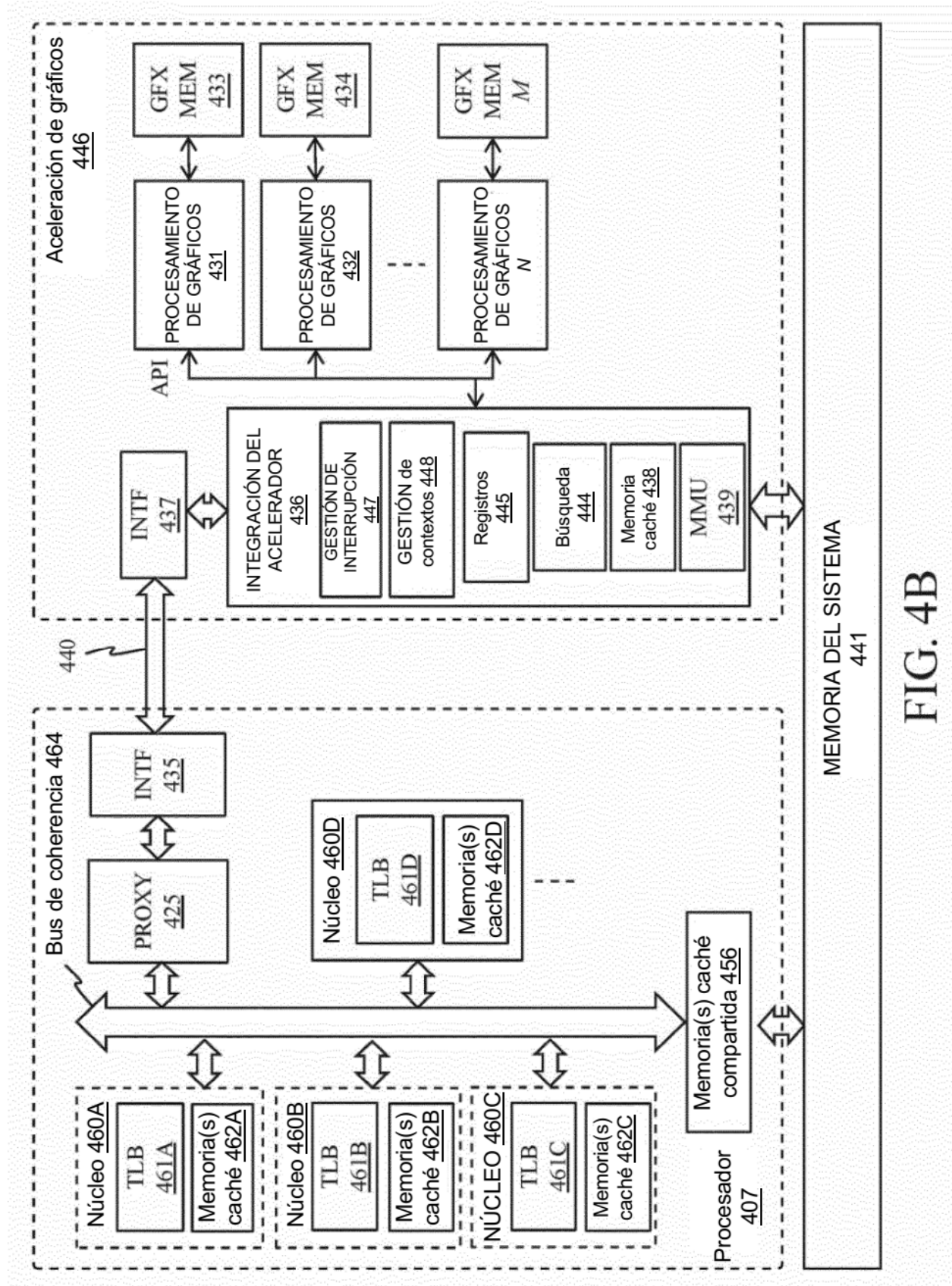


FIG. 4B

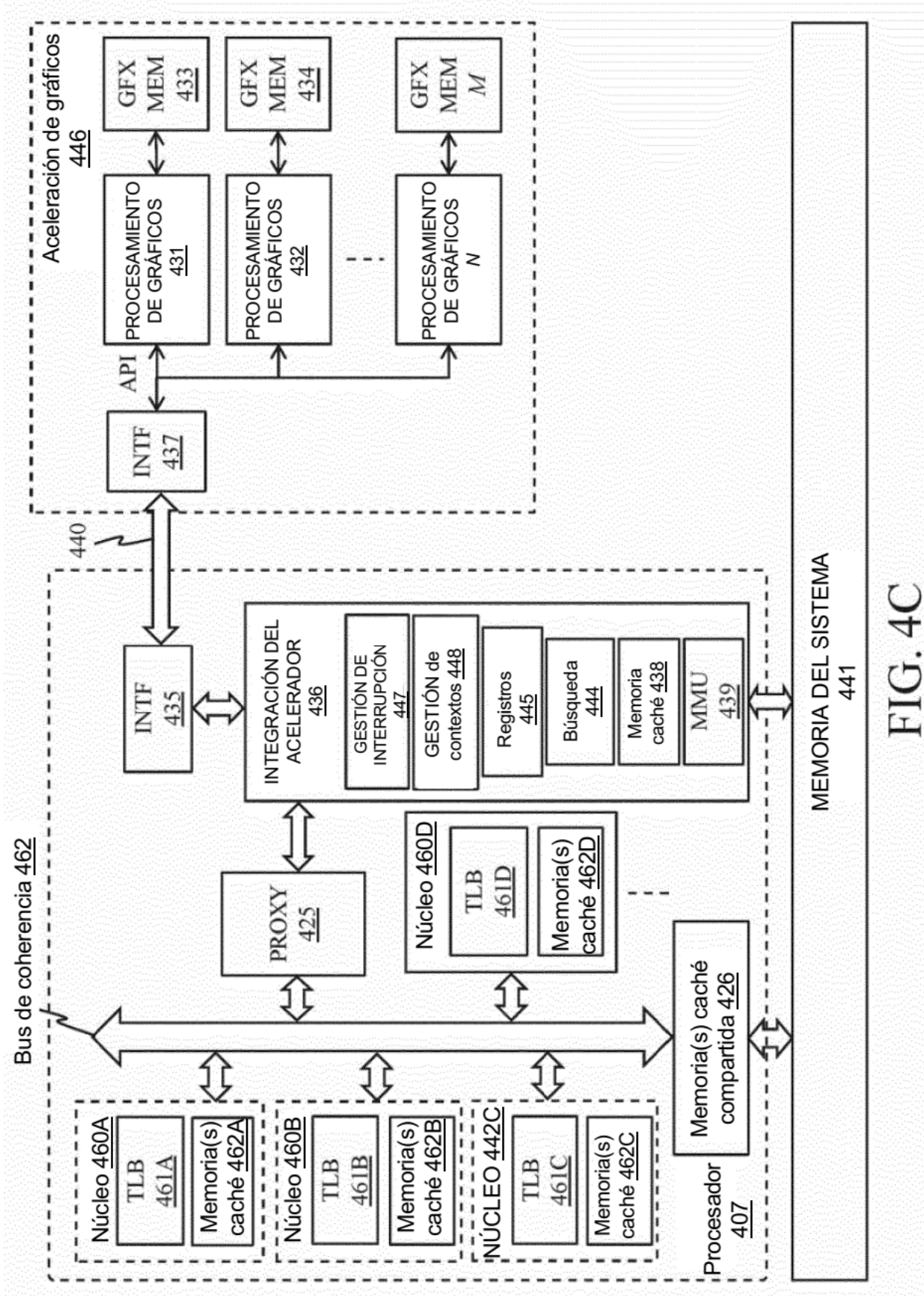


FIG. 4C

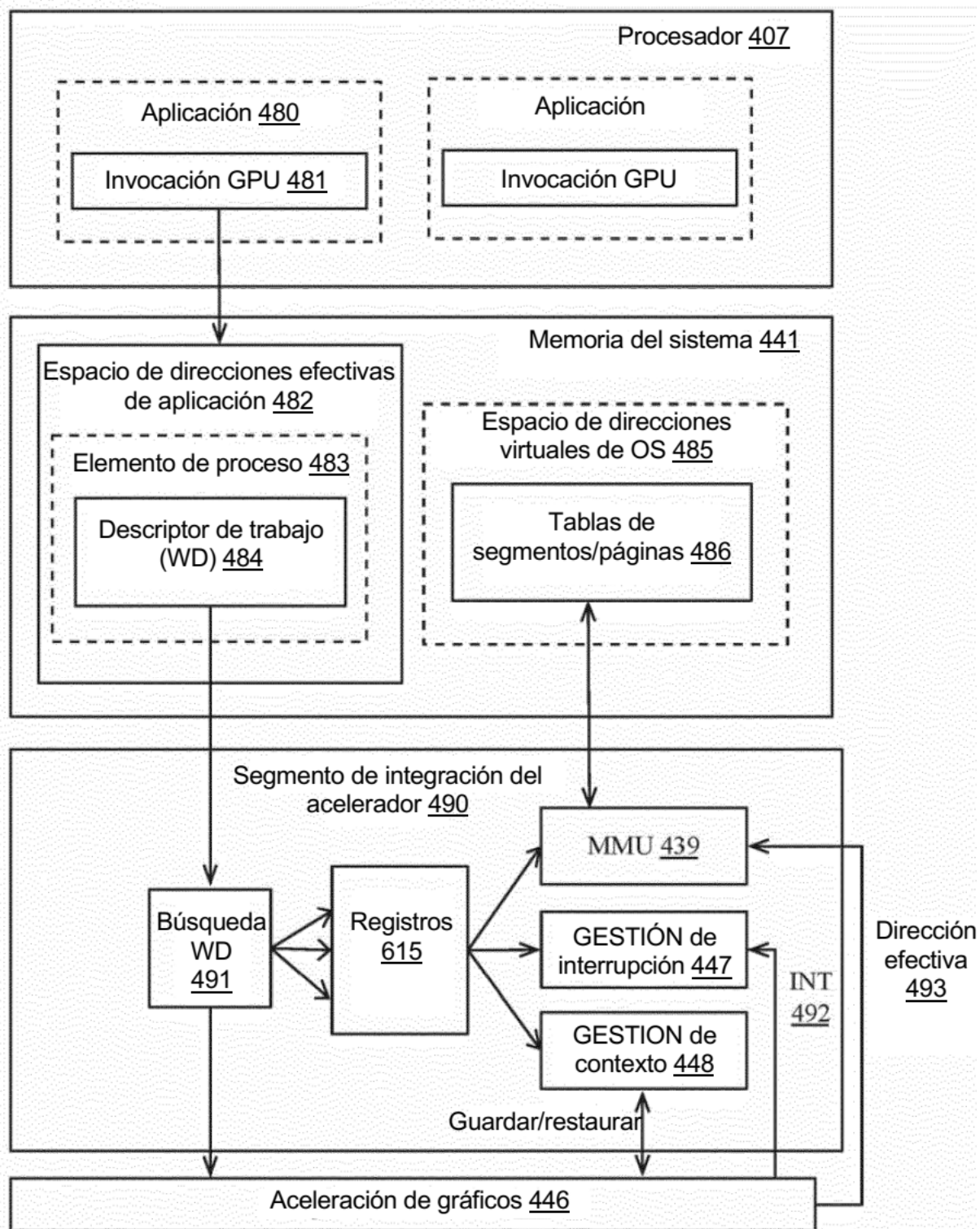


FIG. 4D

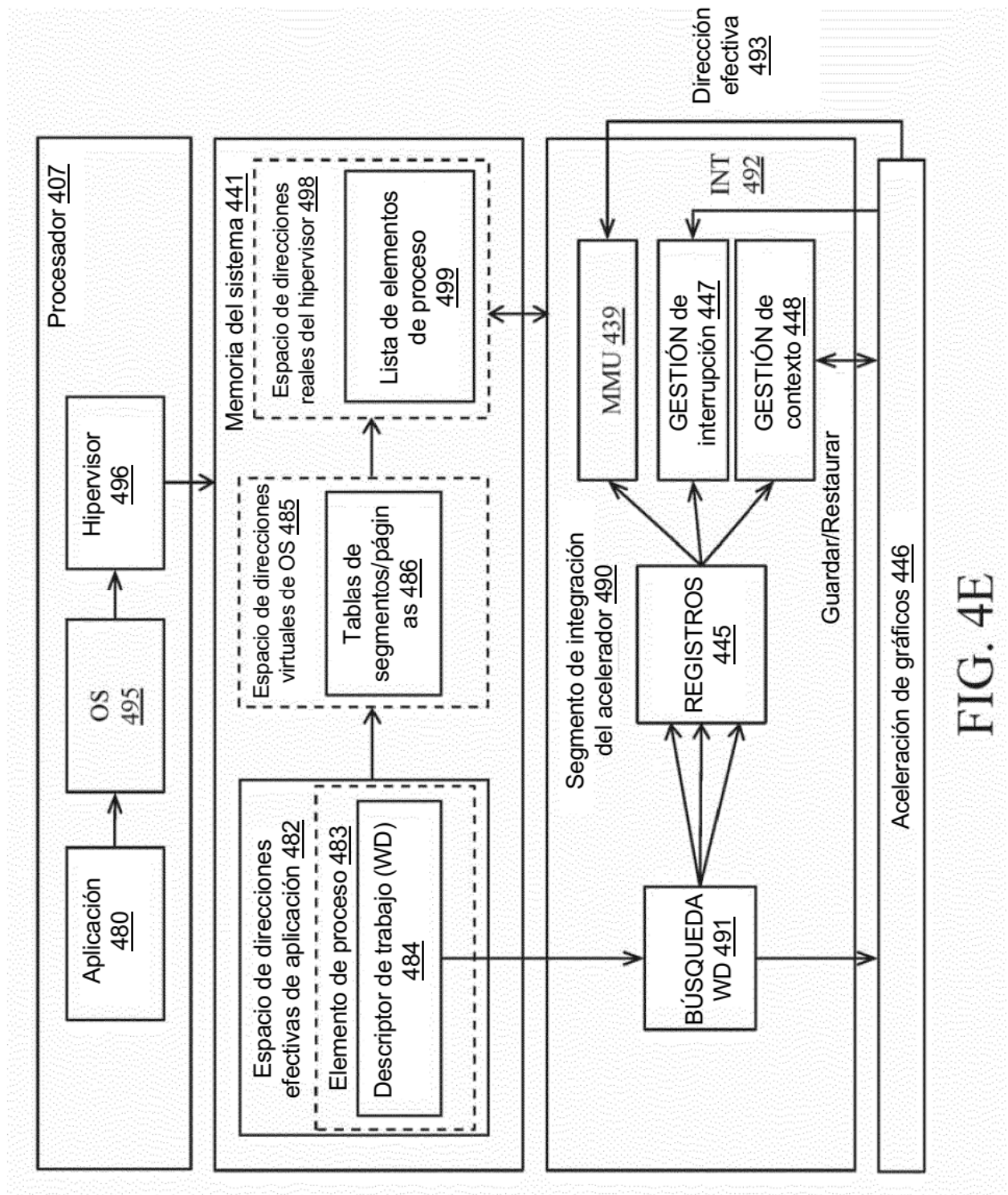


FIG. 4E

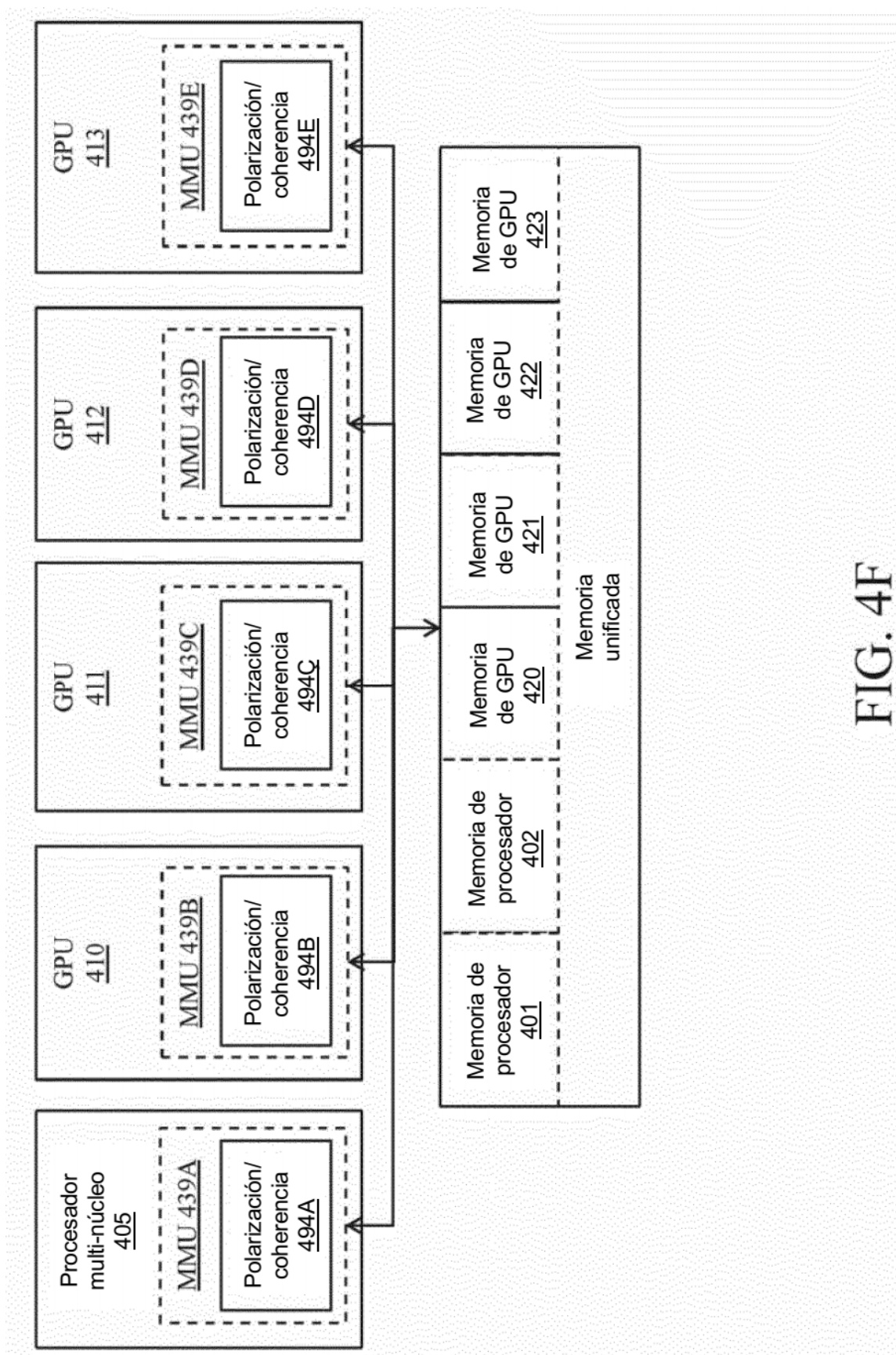


FIG. 4F

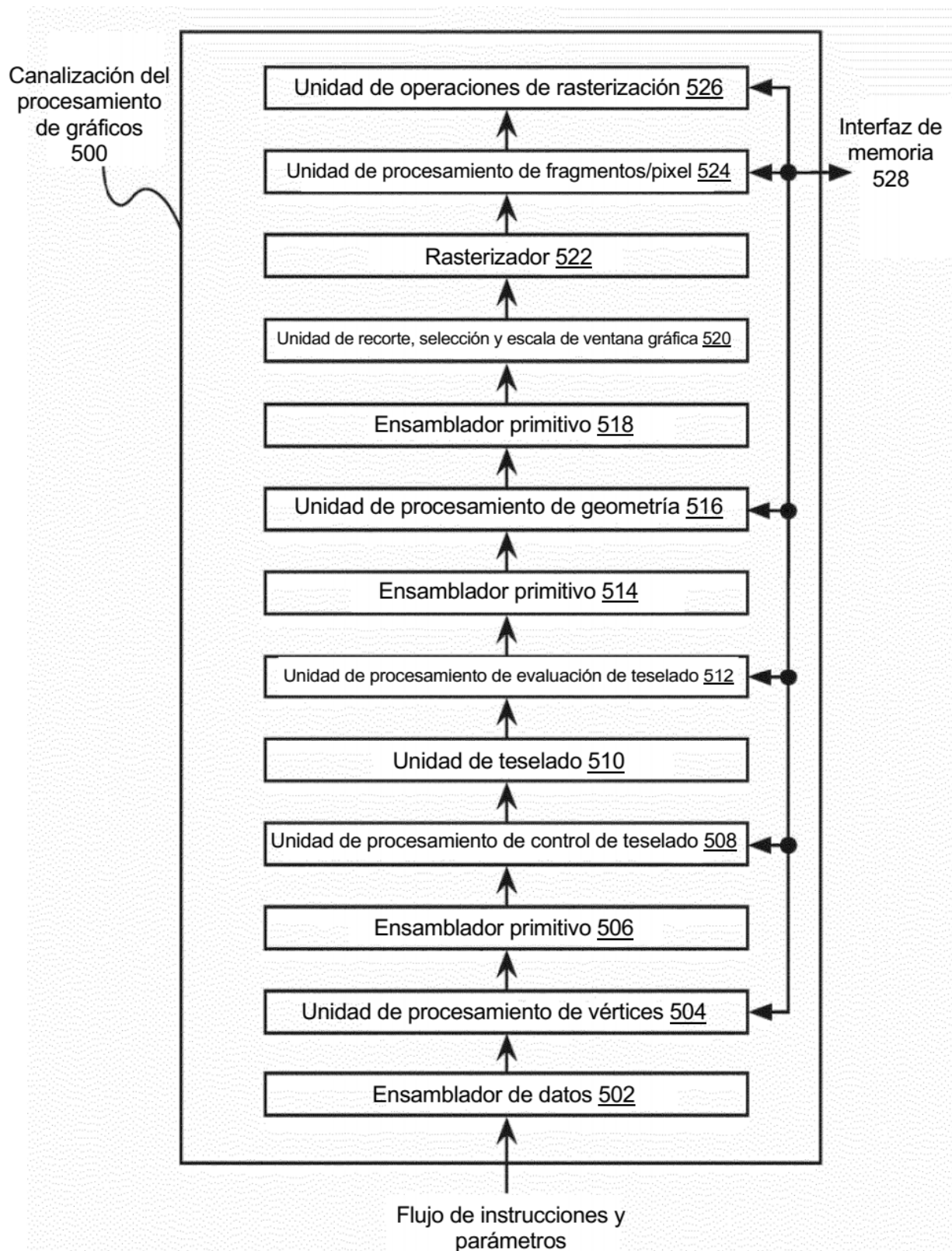


FIG. 5

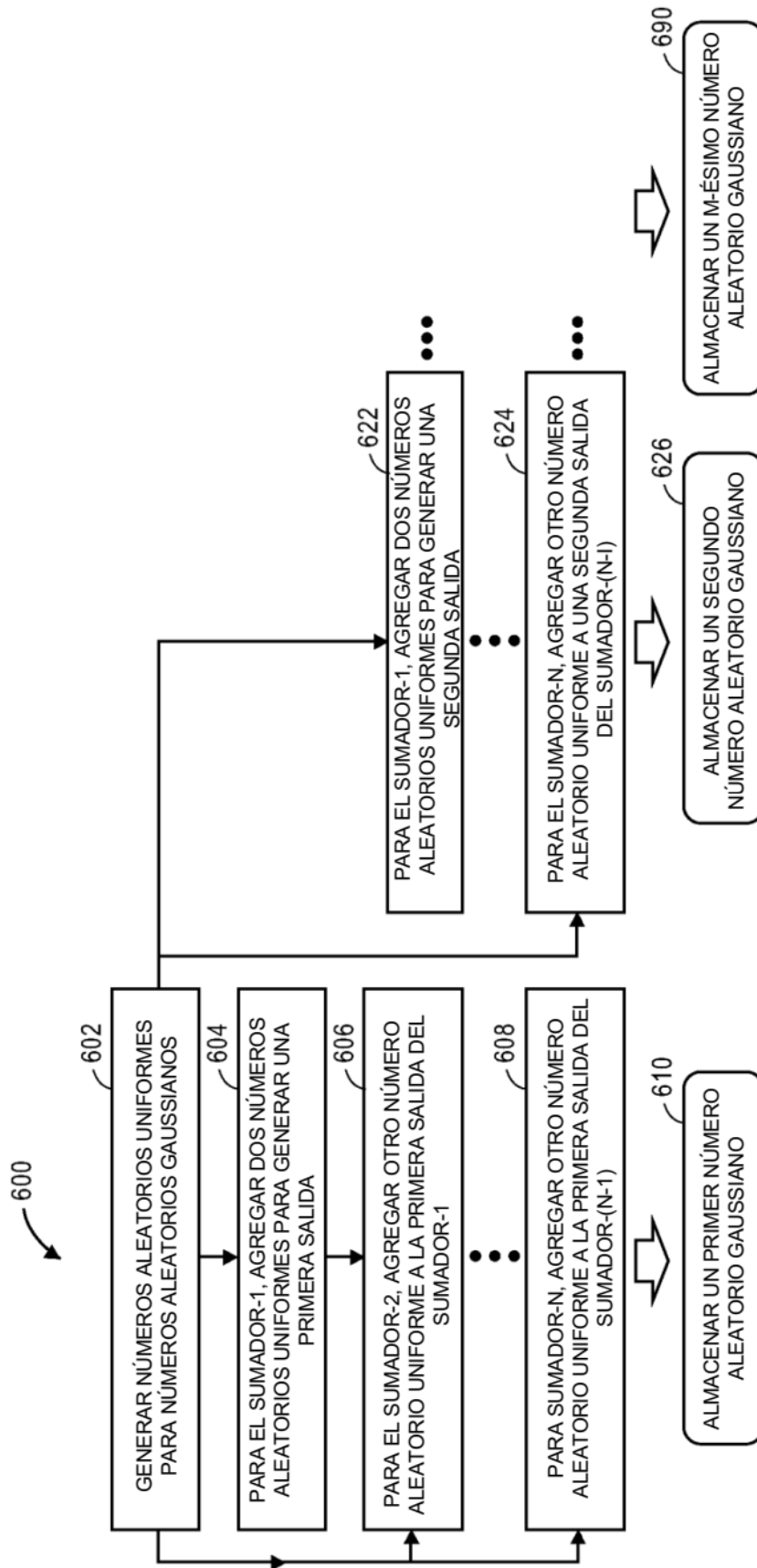
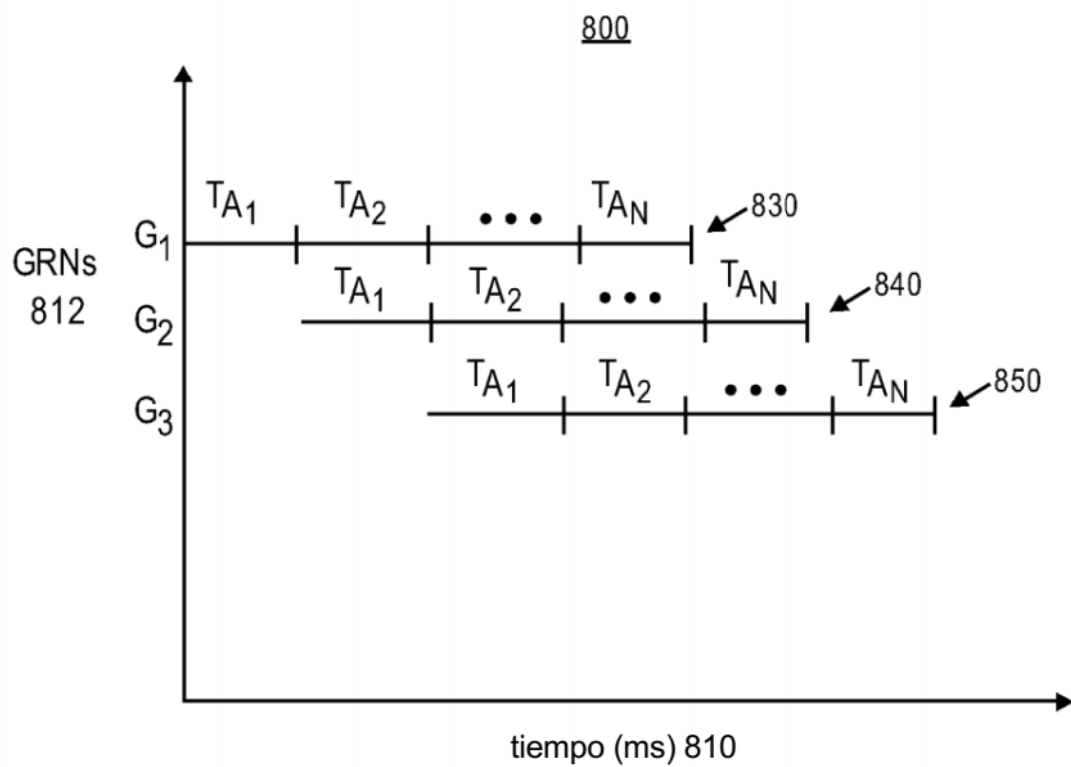
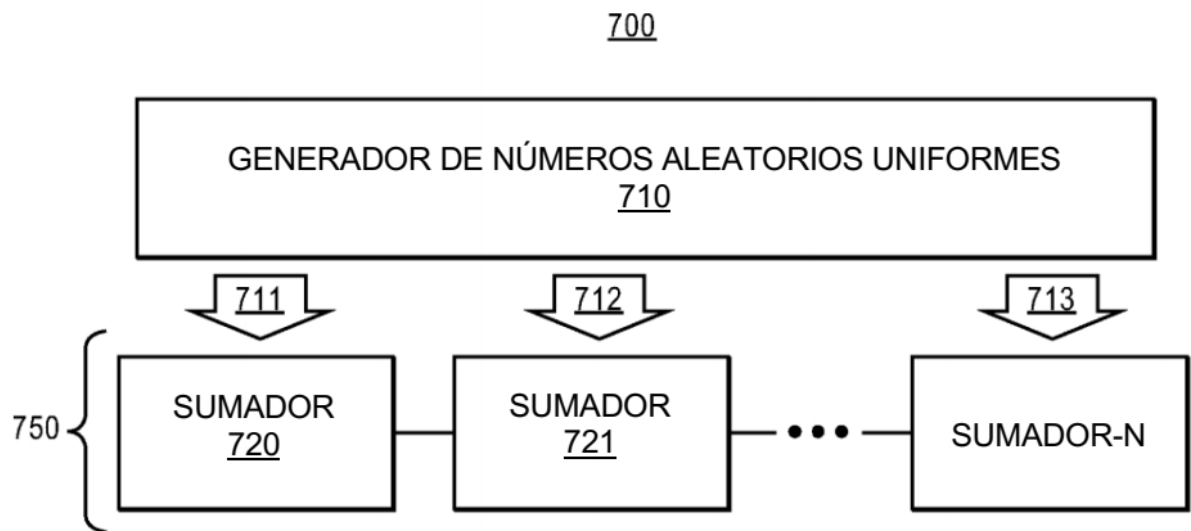


FIG. 6



900

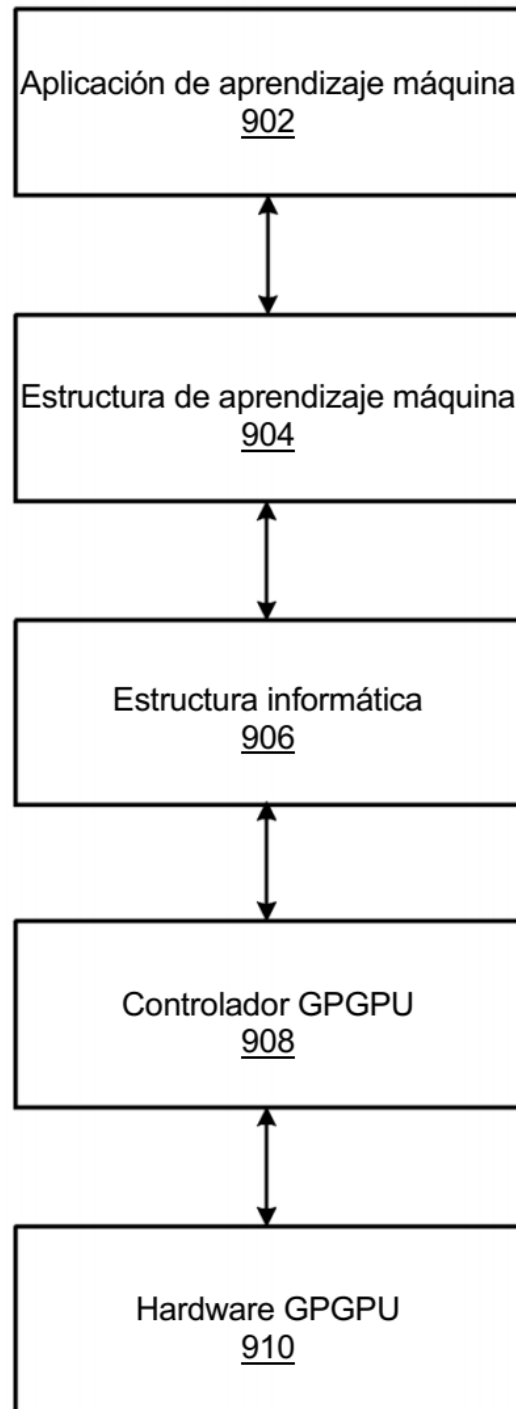


FIG. 9

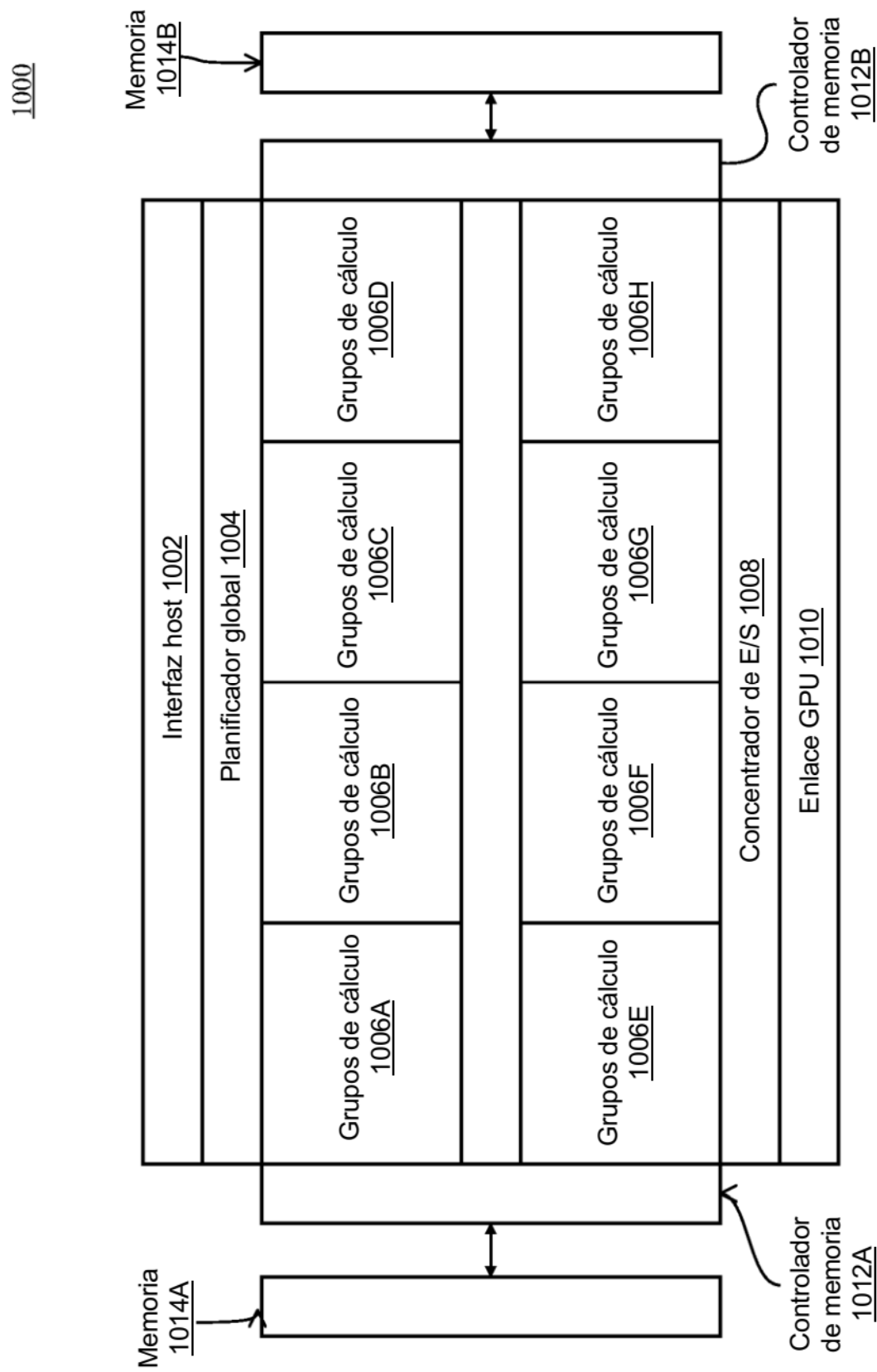


FIG. 10

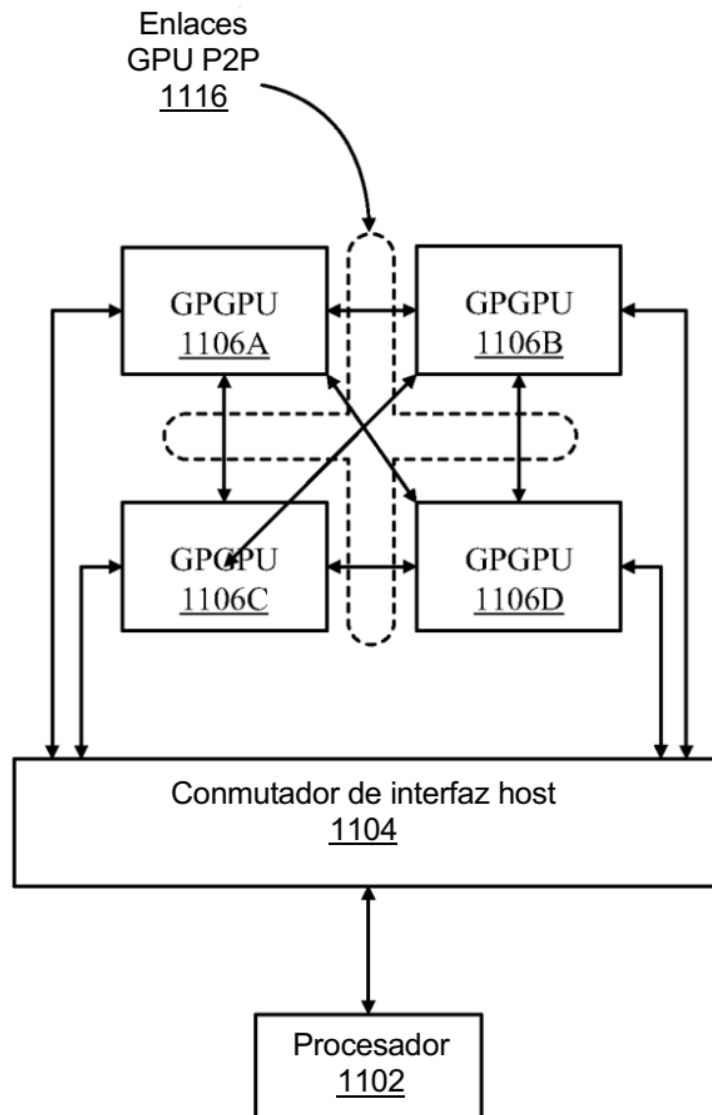


FIG. 11

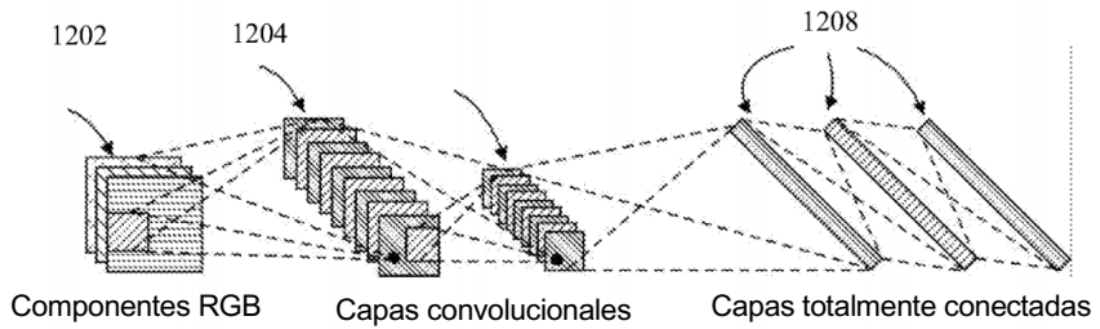


FIG. 12A

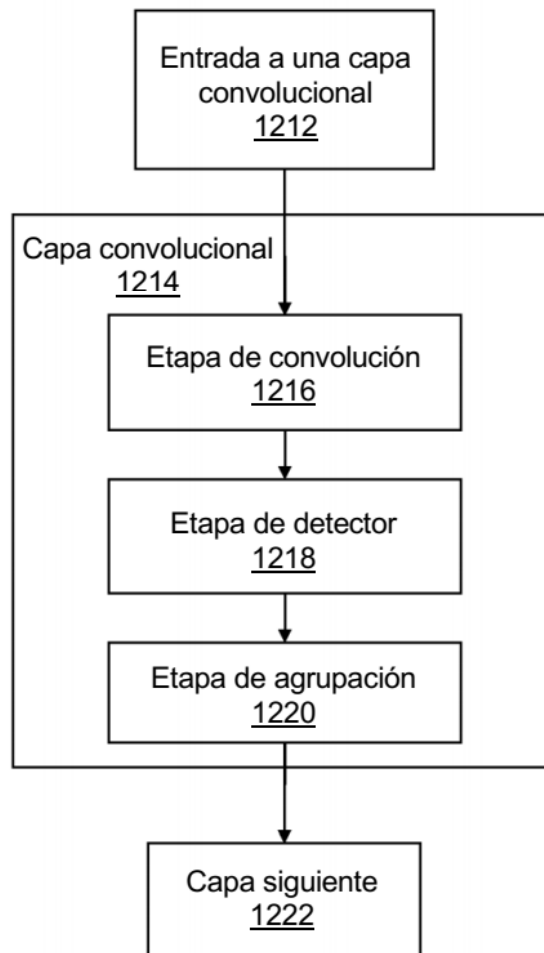


FIG. 12B

1300

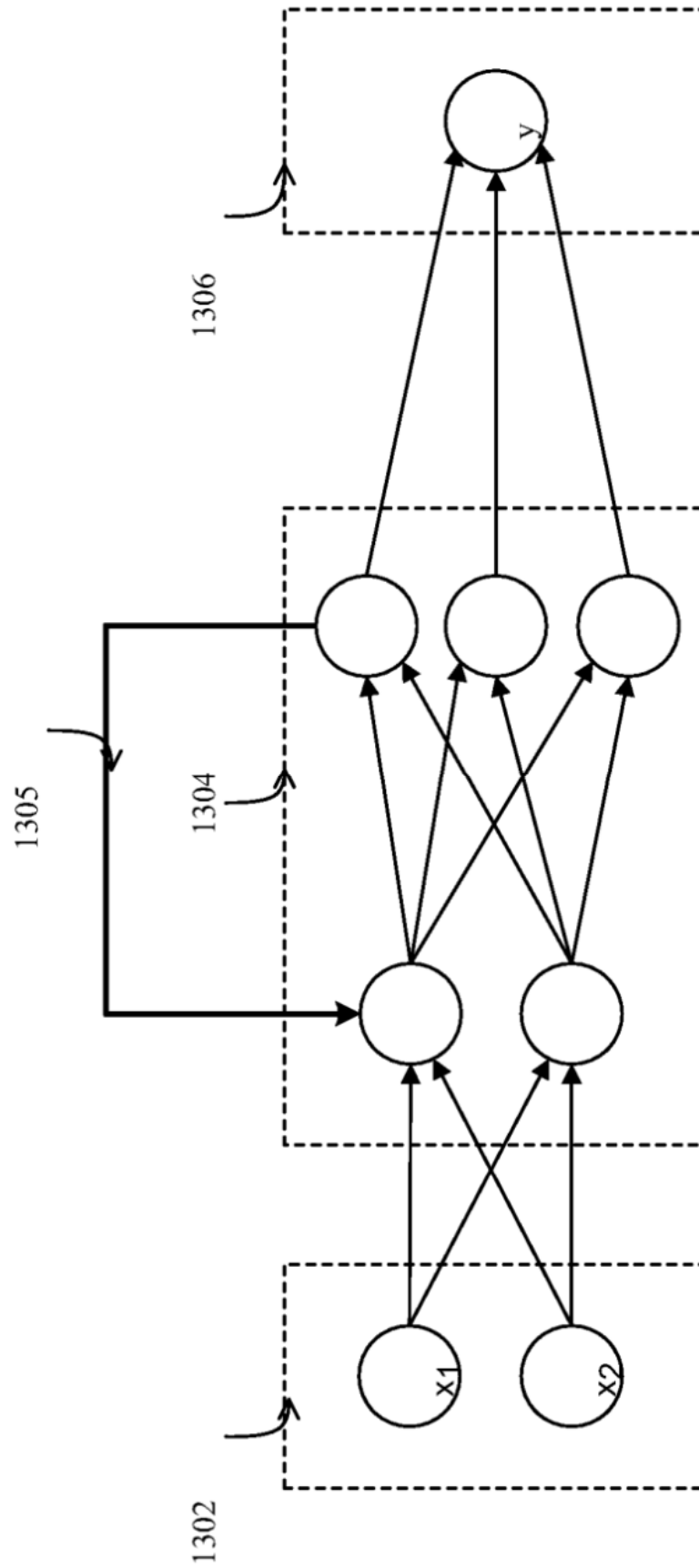


FIG. 13

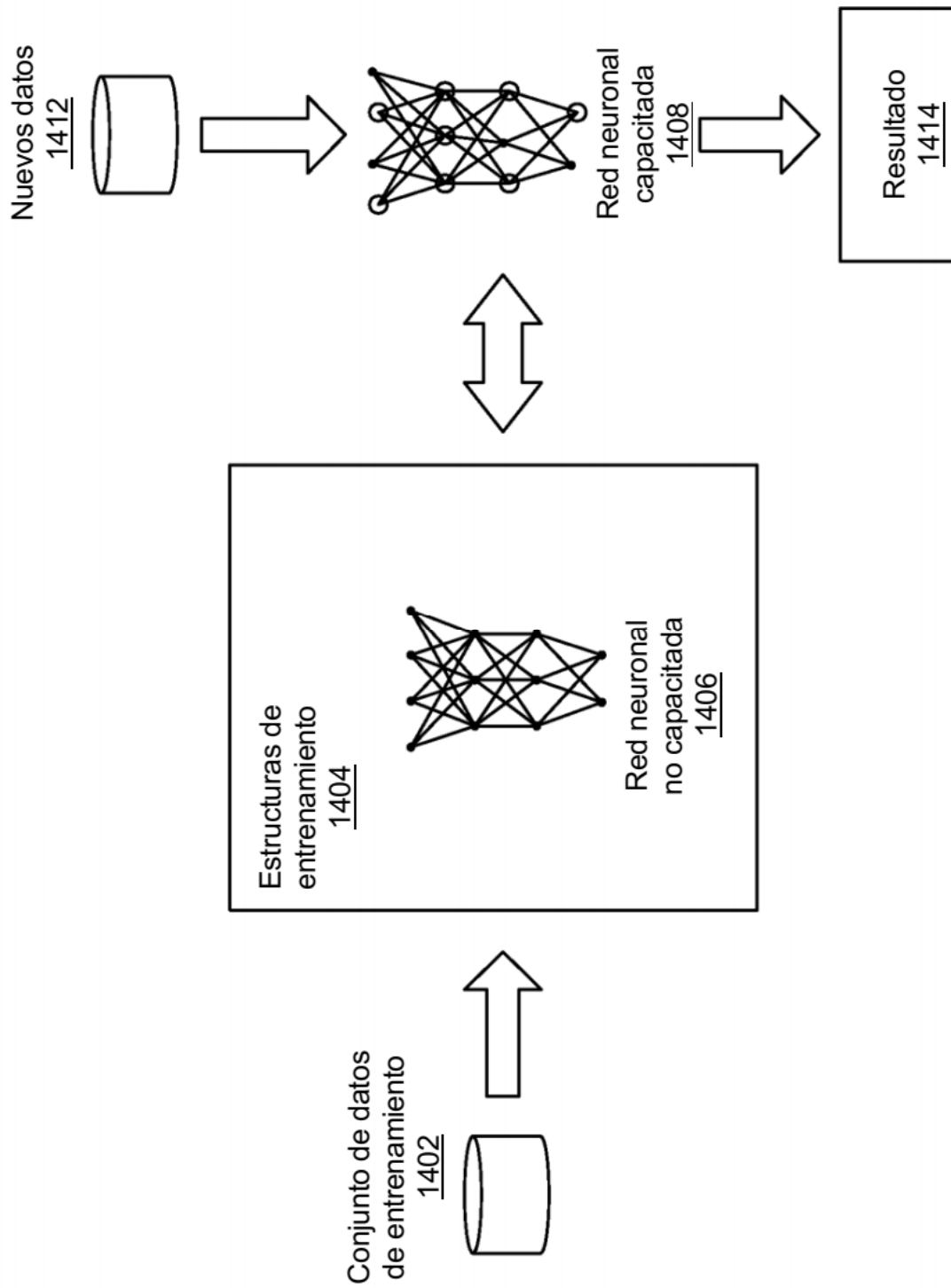


FIG. 14

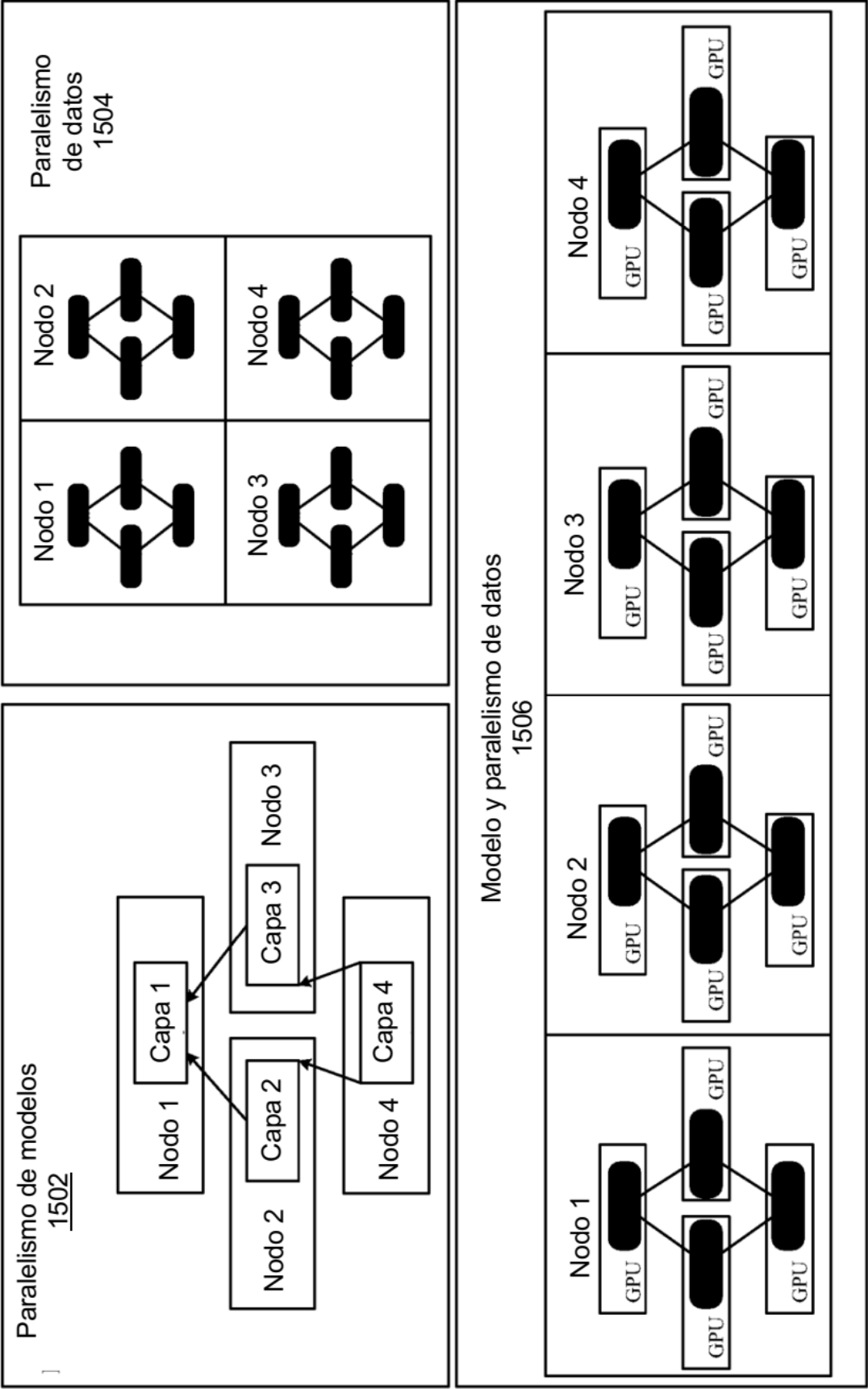


FIG. 15

1600



FIG. 16

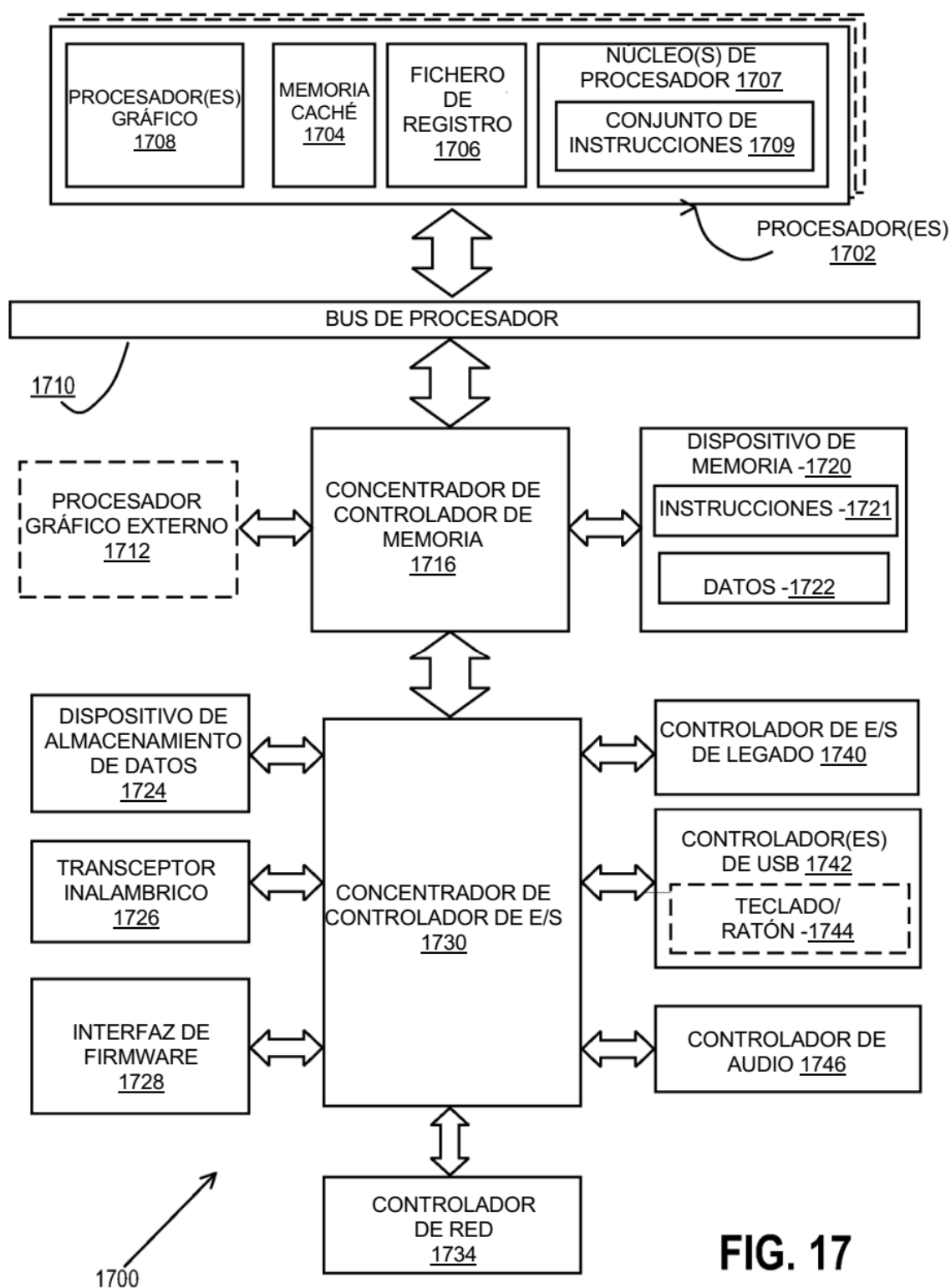


FIG. 17

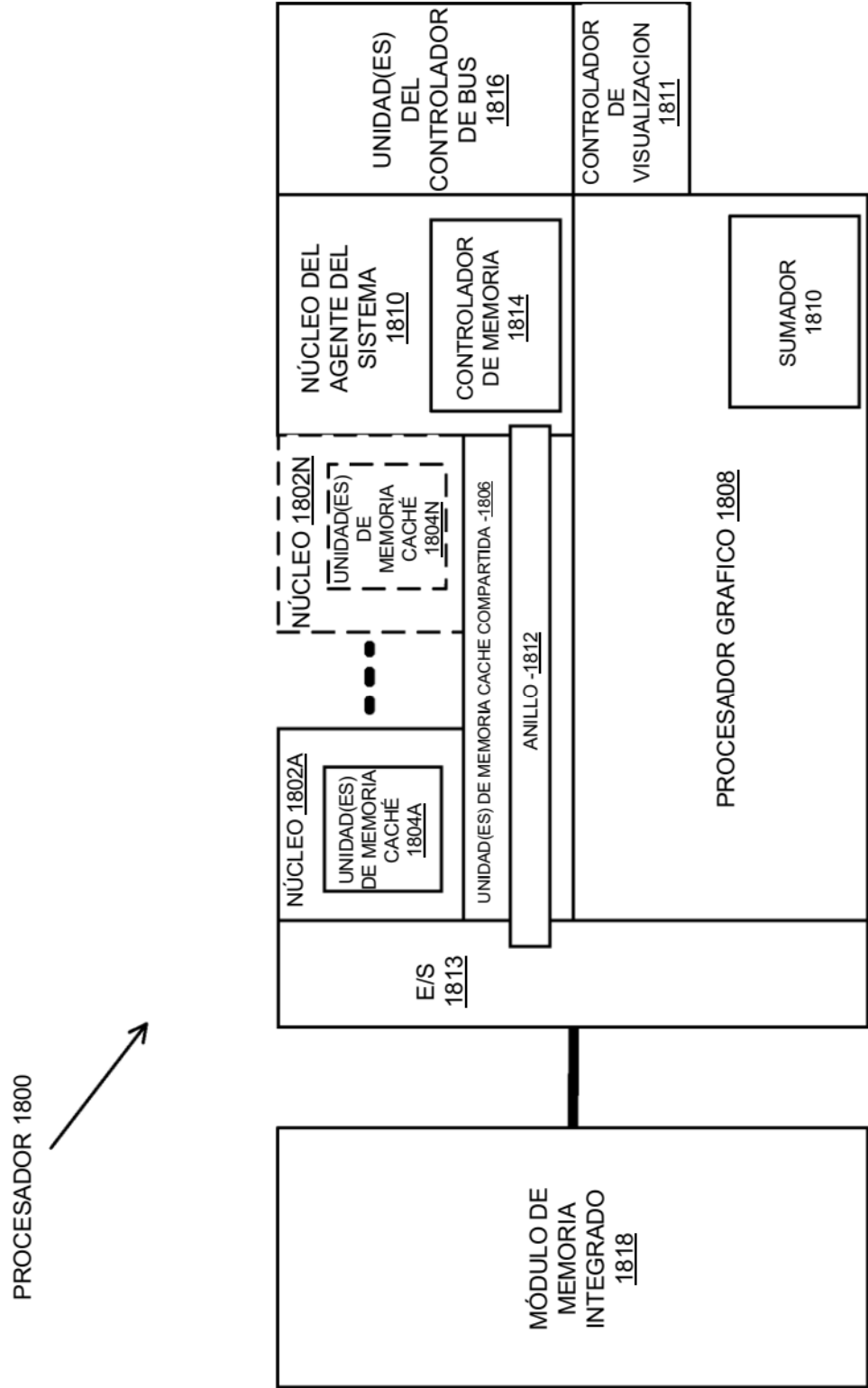


FIG. 18

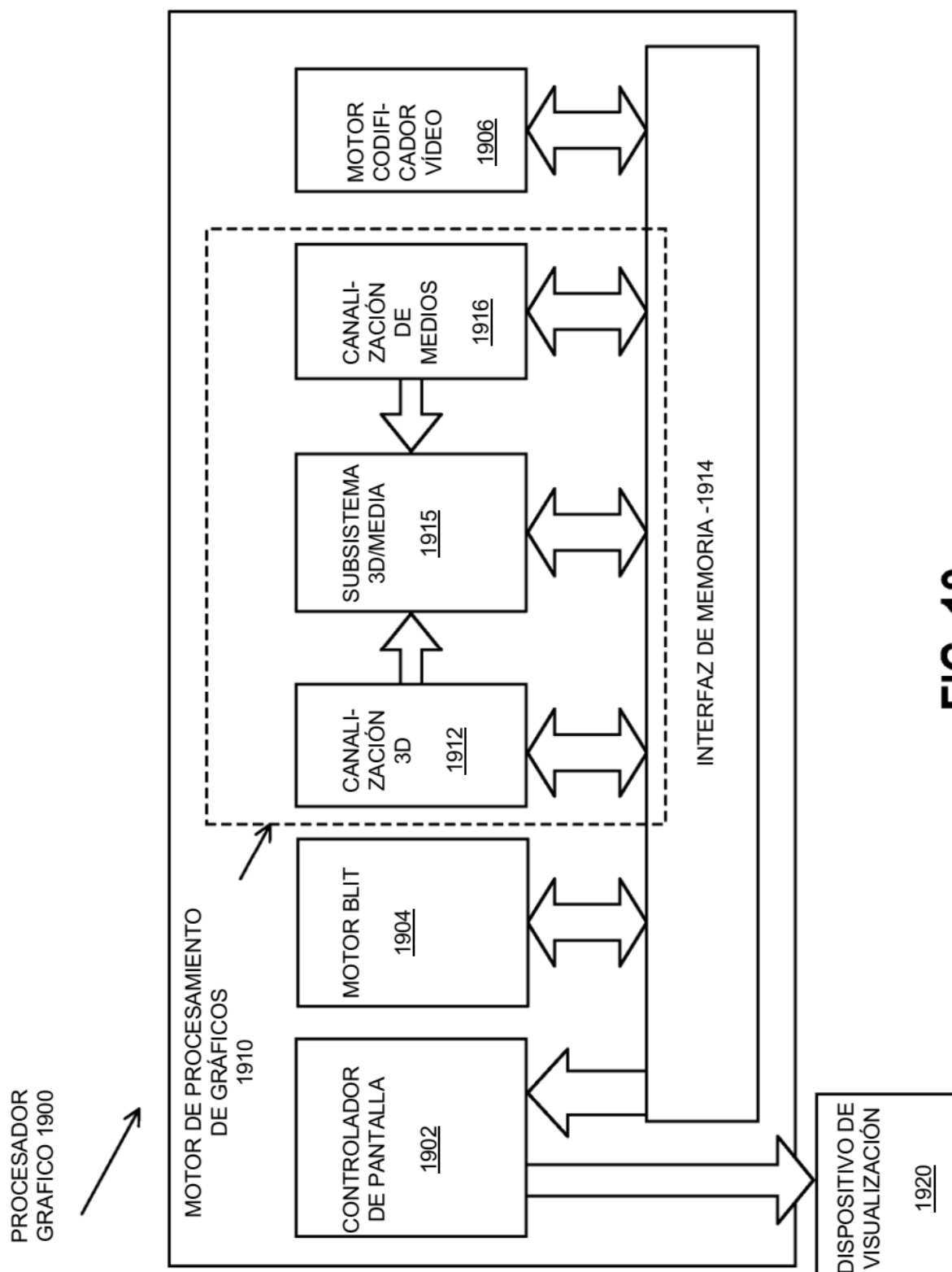


FIG. 19

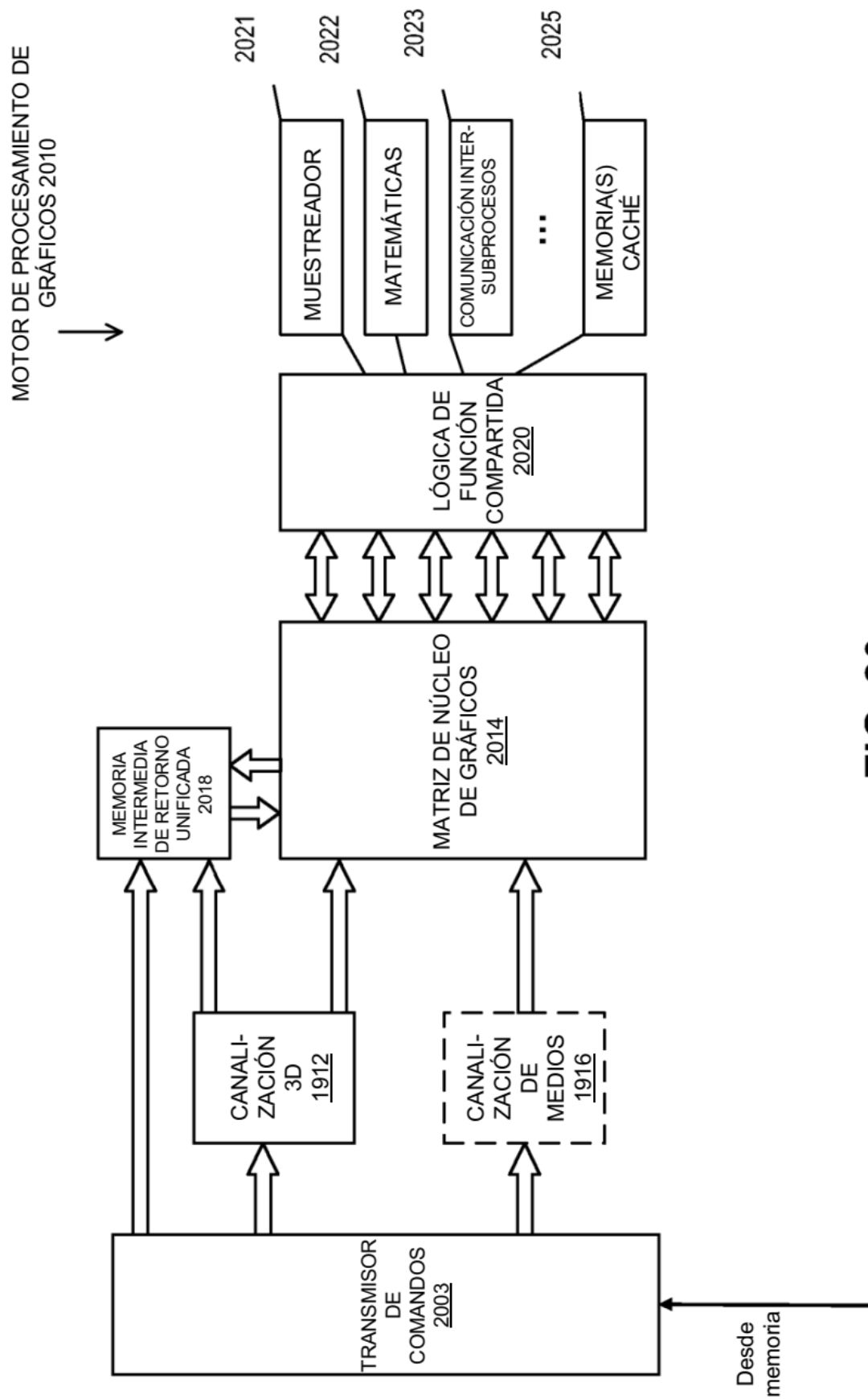


FIG. 20

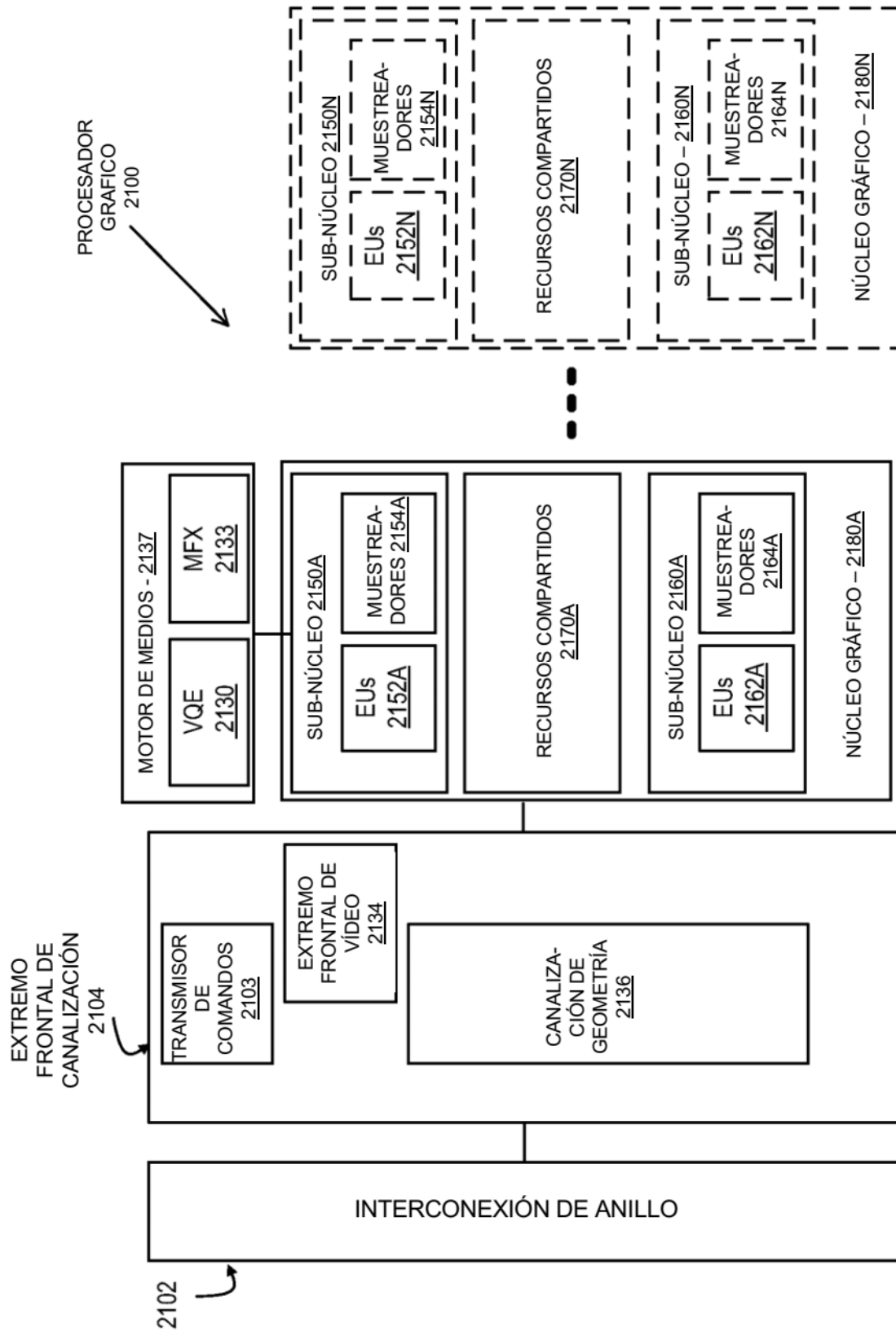


FIG. 21

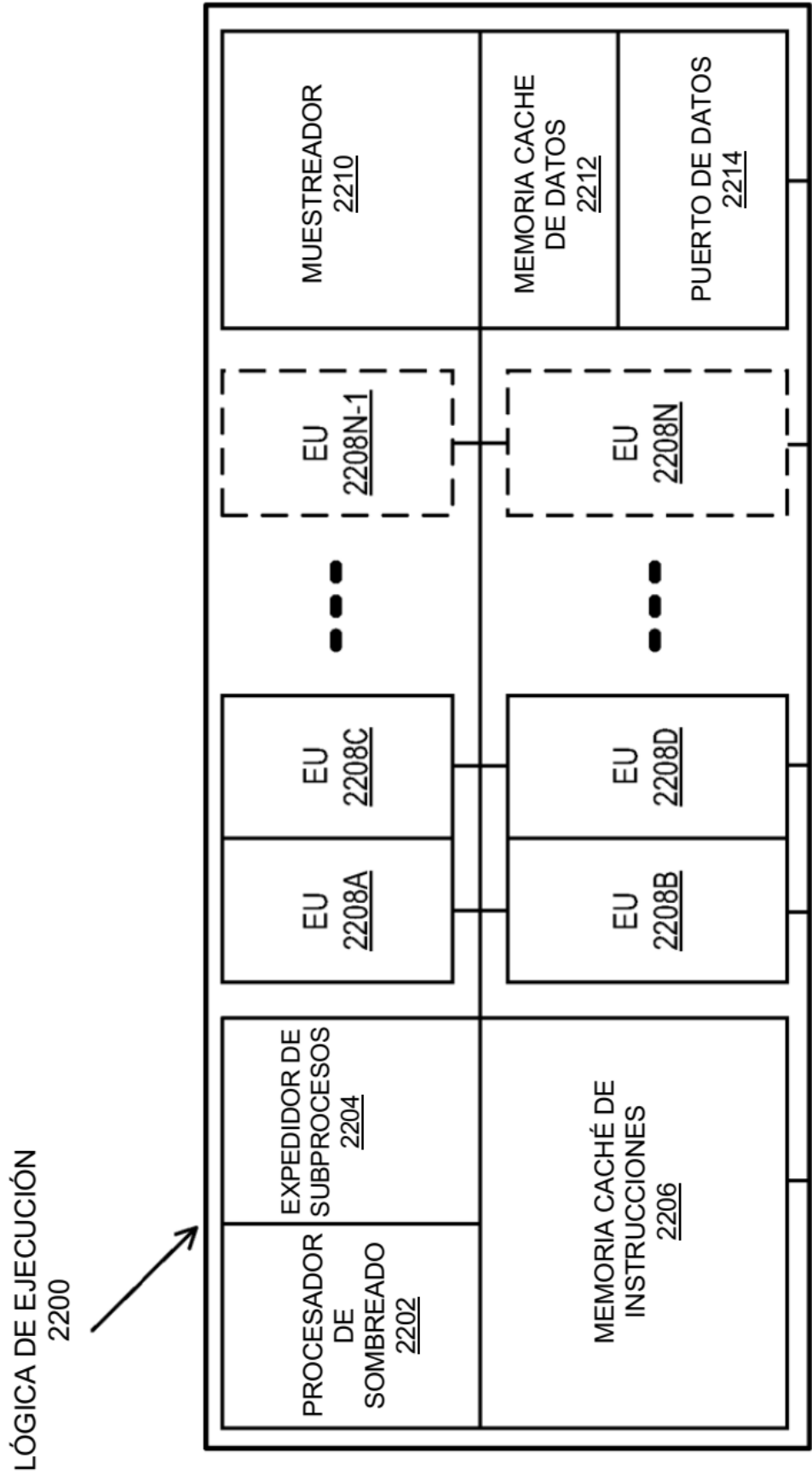


FIG. 22

FORMATOS DE INSTRUCCIONES DEL PROCESADOR GRÁFICO

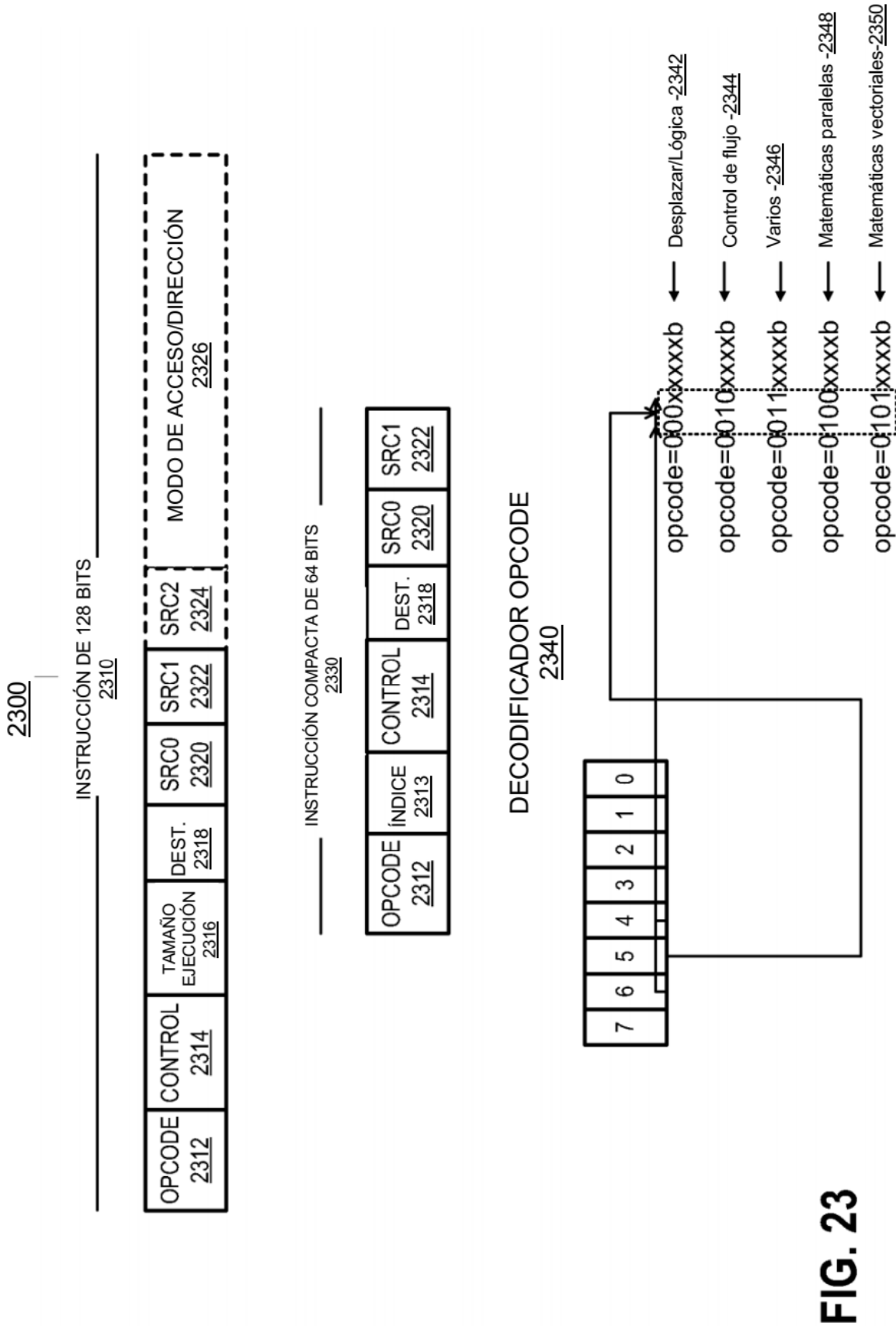


FIG. 23

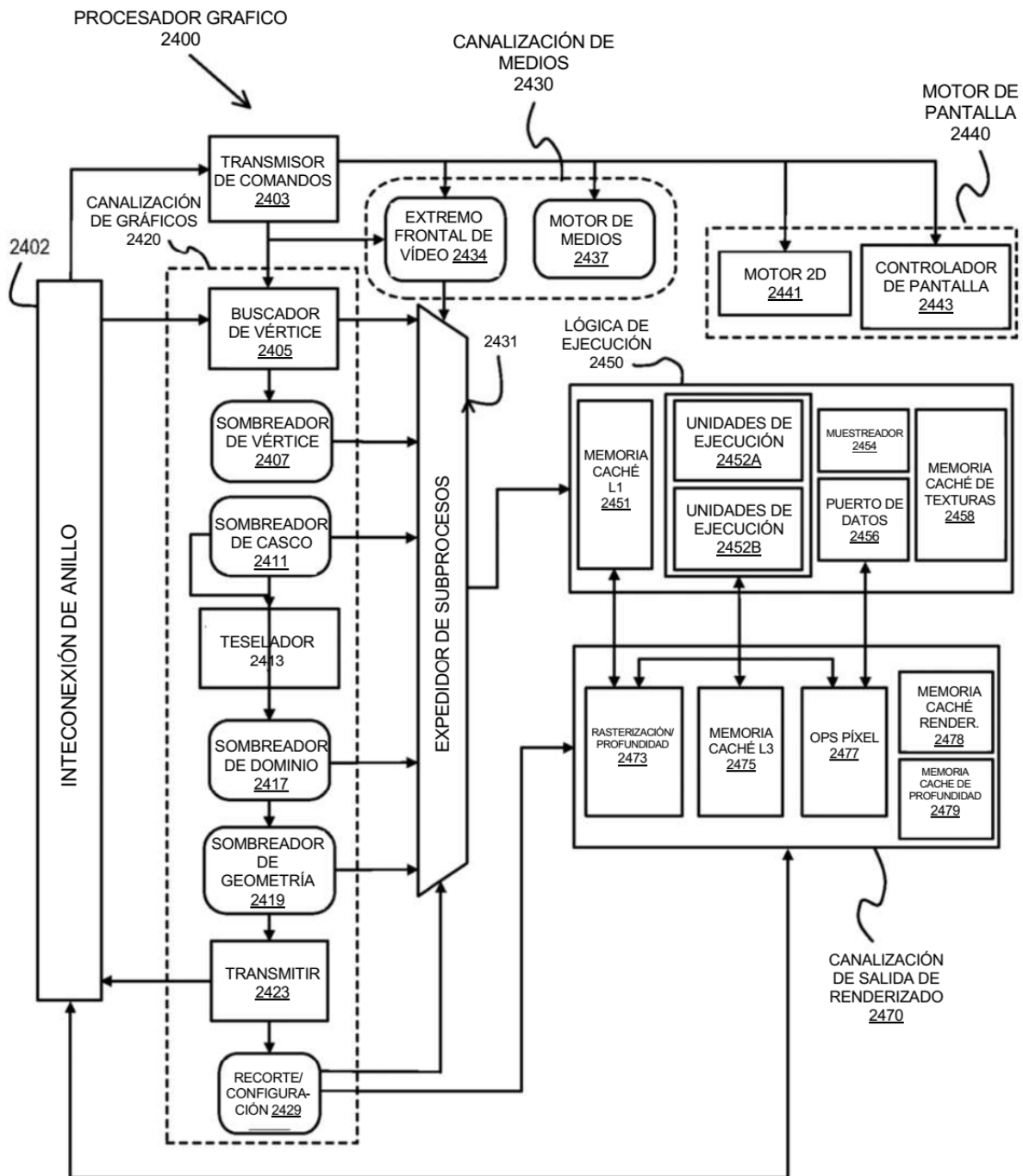


FIG. 24

FIG. 25A FORMATO DE COMANDO DEL PROCESADOR GRÁFICO
2500

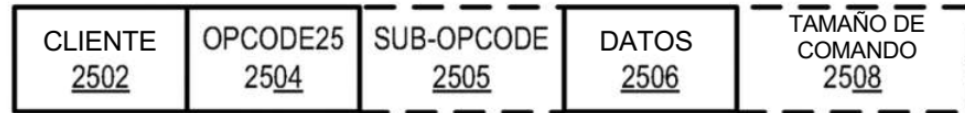
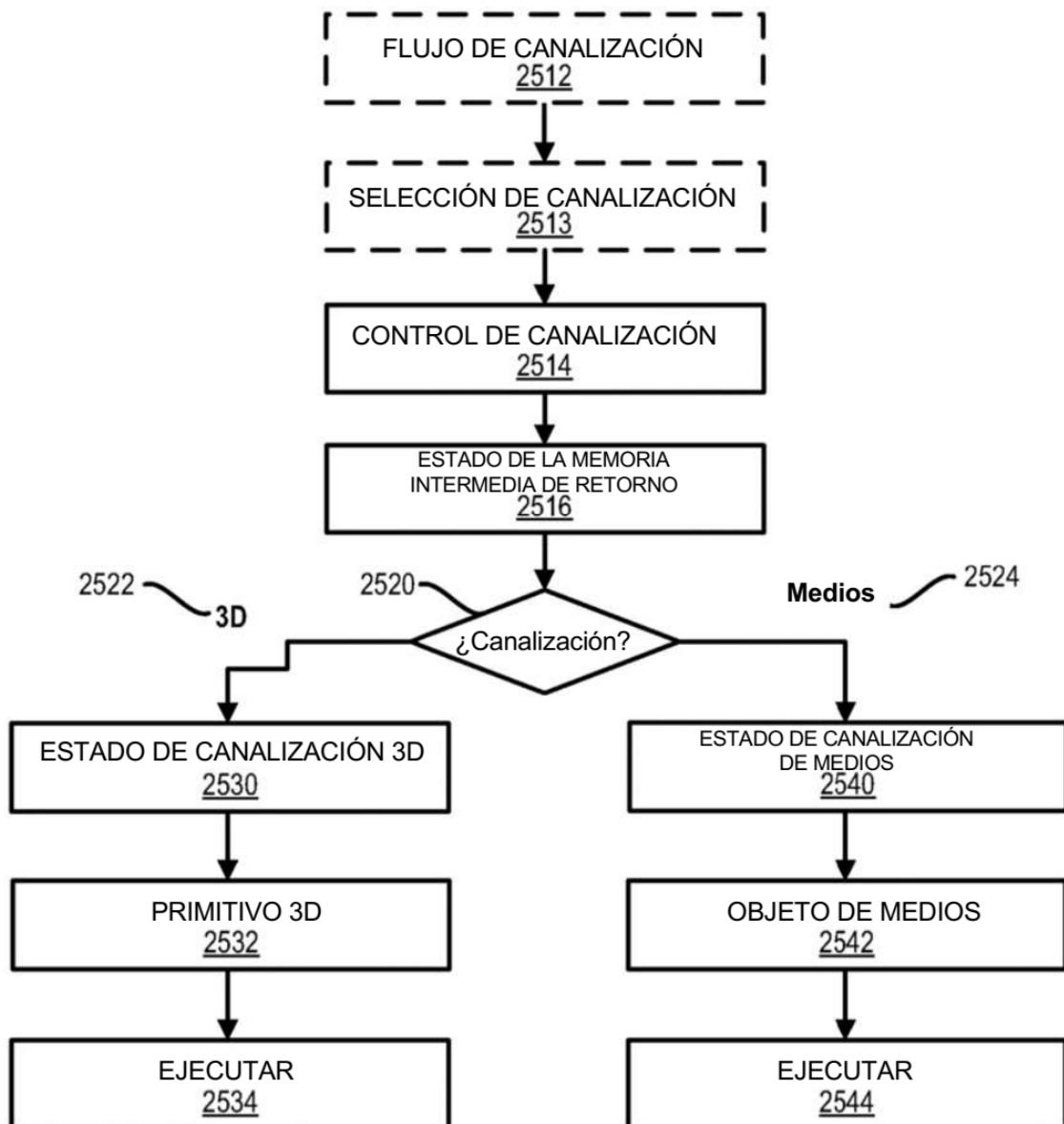


FIG. 25B SECUENCIA DE COMANDO DEL PROCESADOR GRÁFICO
2510



SISTEMA DE PROCESAMIENTO DE DATOS - 2600

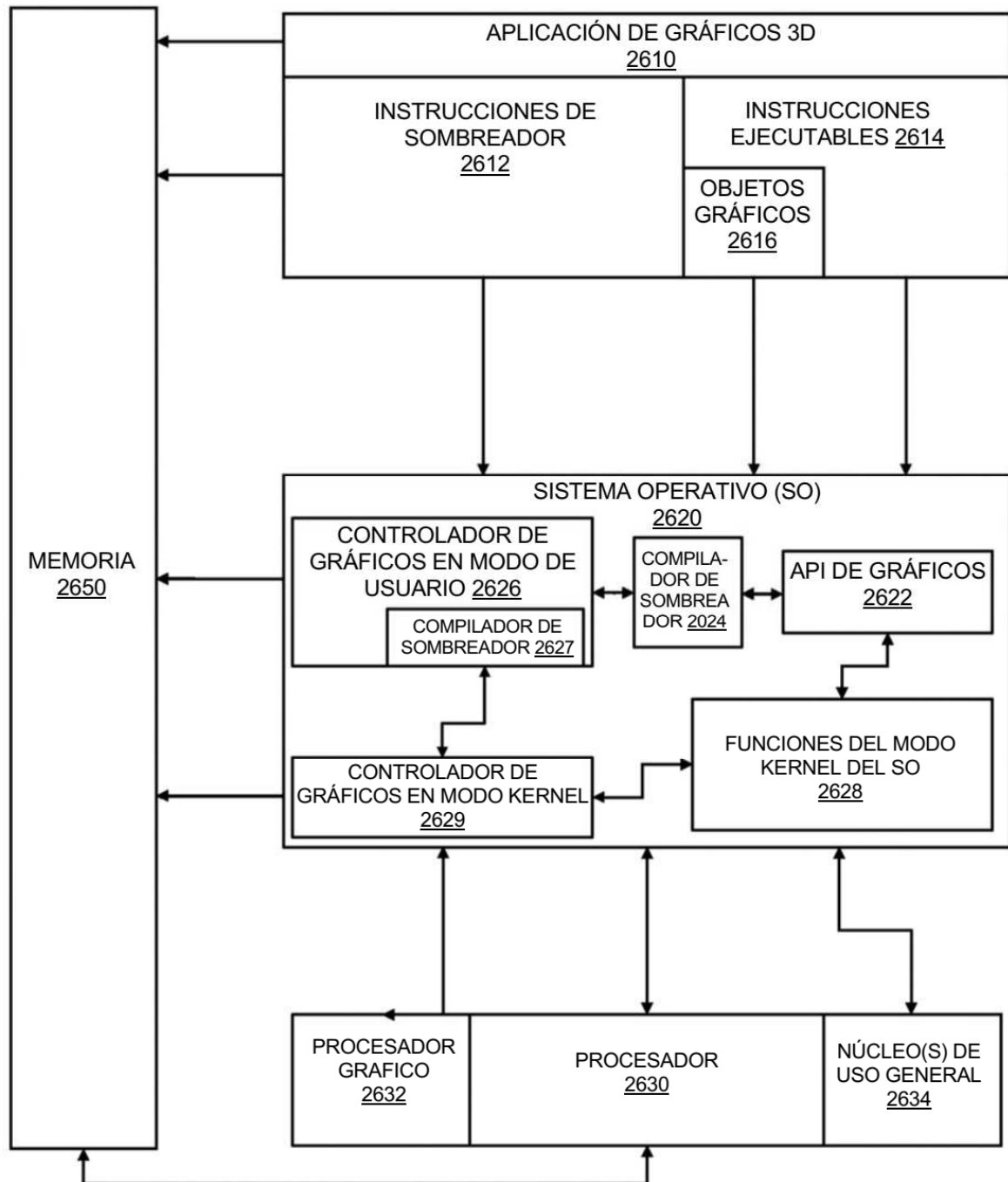


FIG. 26

DESARROLLO DE NÚCLEO IP - 2700

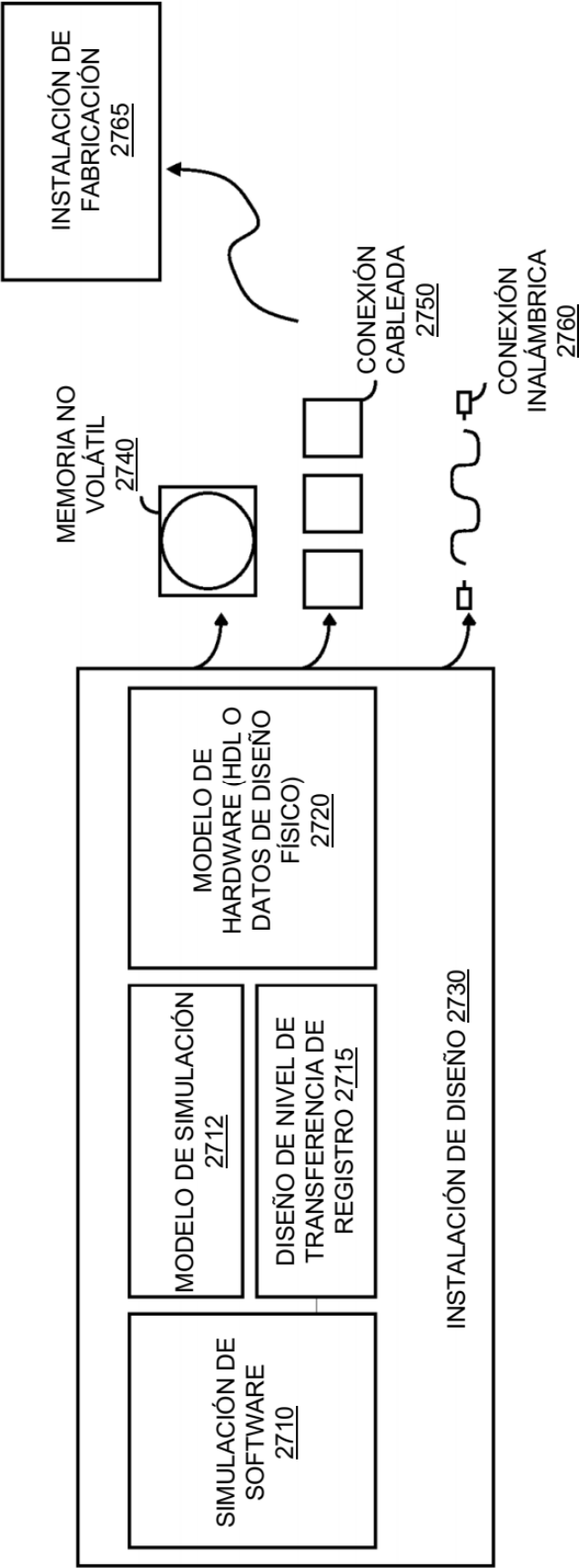


FIG. 27

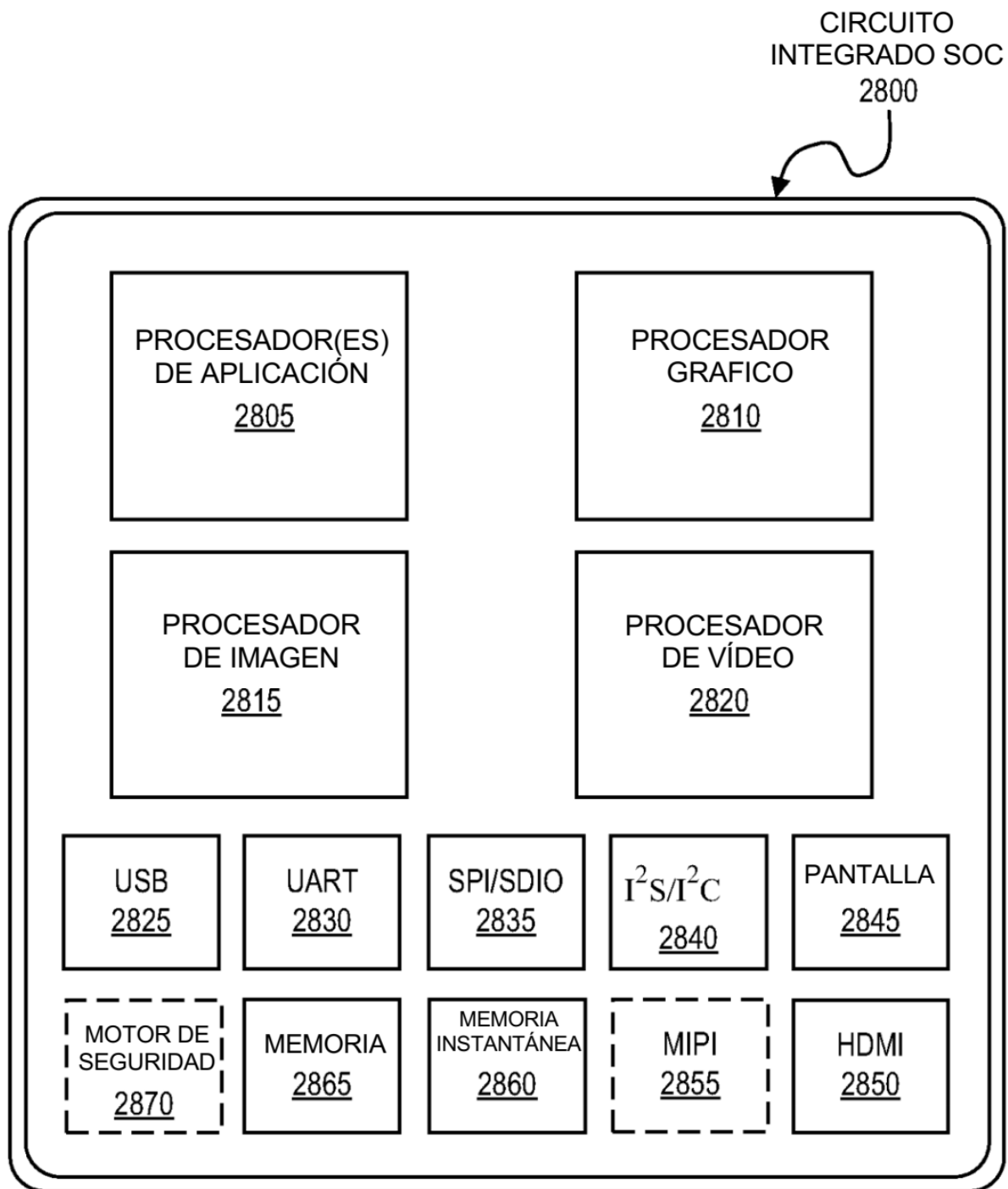


FIG. 28

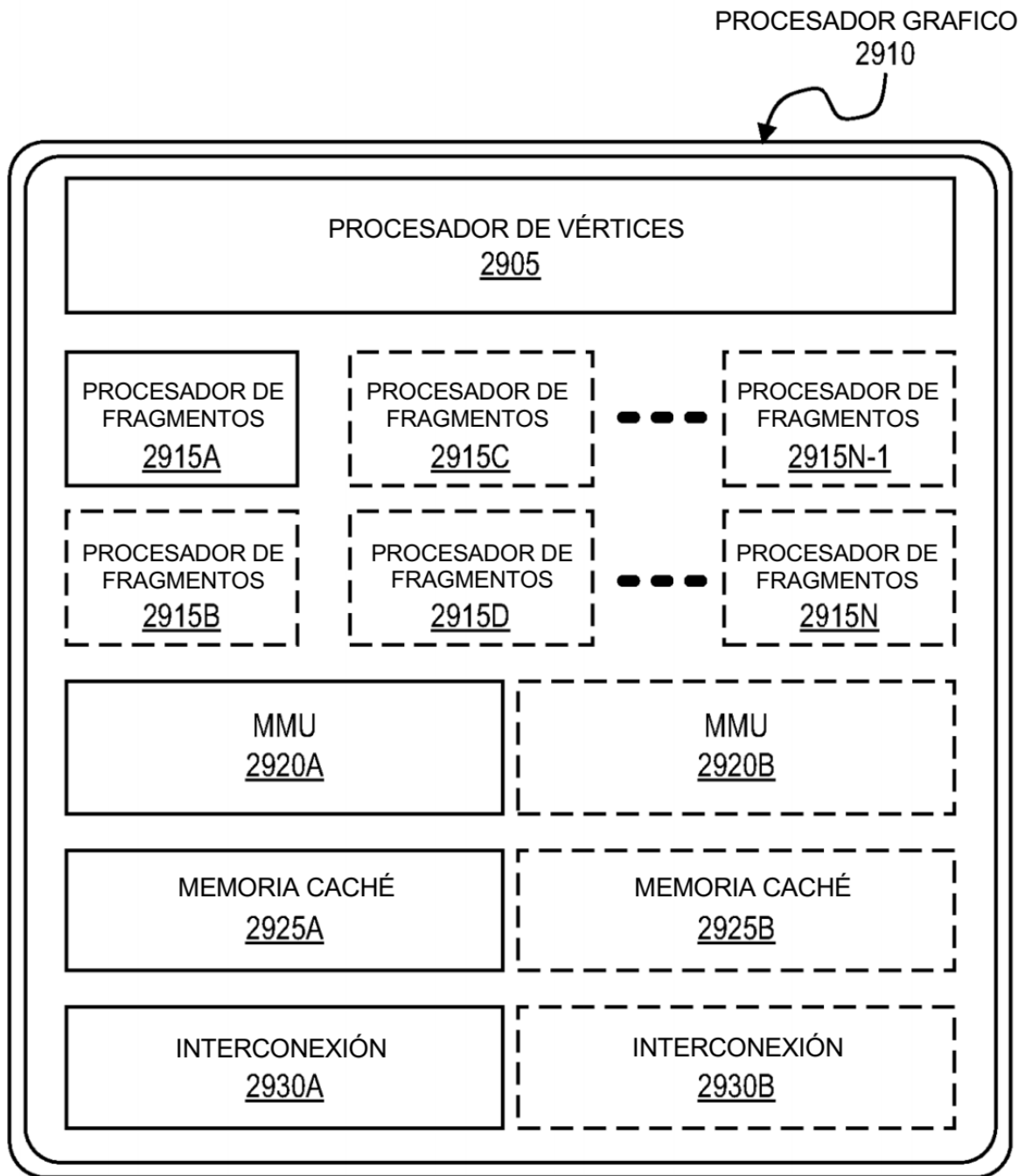


FIG. 29

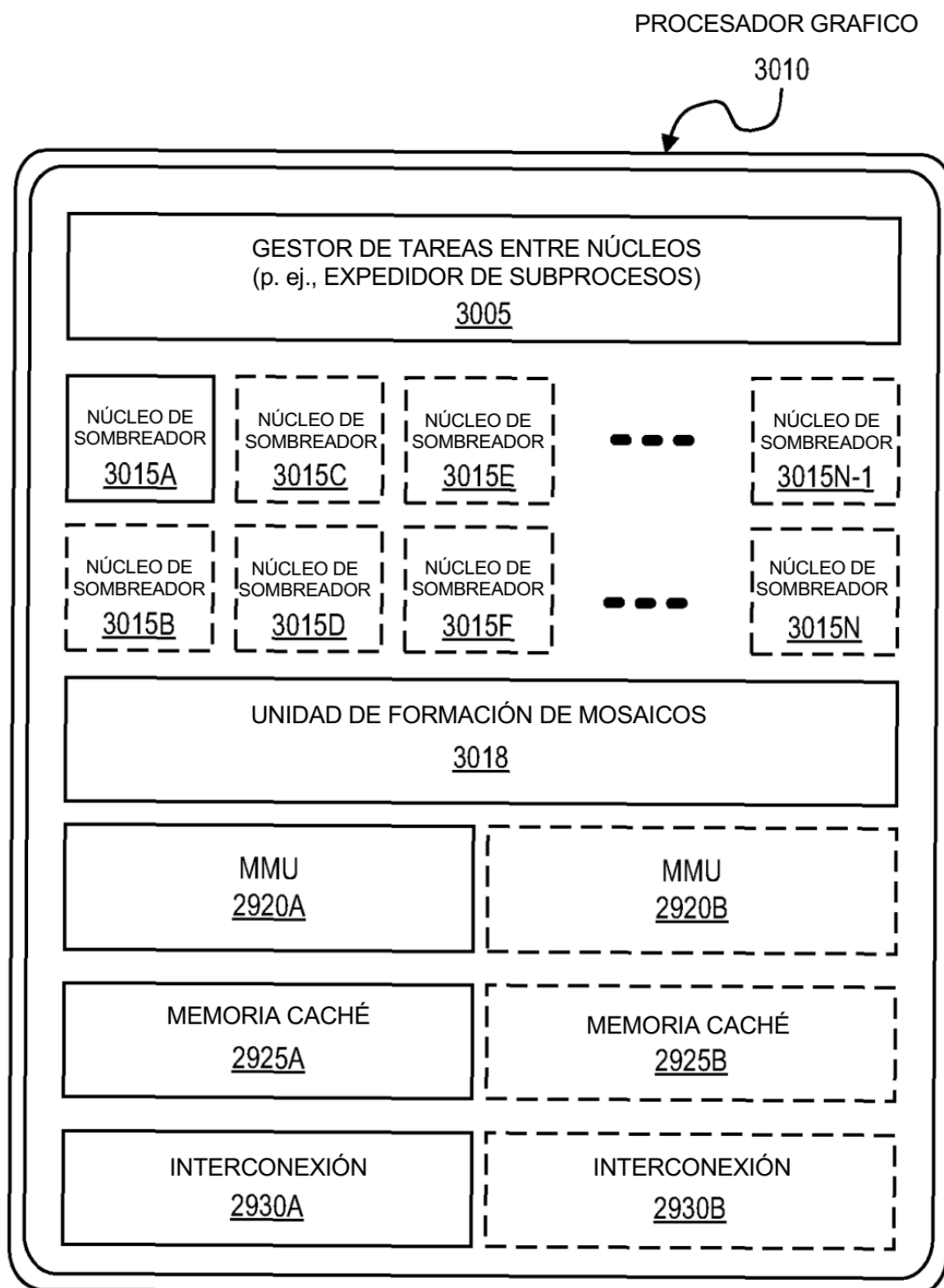


FIG. 30