

19



OFICINA ESPAÑOLA DE
PATENTES Y MARCAS

ESPAÑA



11 Número de publicación: **2 985 696**

51 Int. Cl.:

G06F 1/32 (2009.01)

H04W 52/02 (2009.01)

12

TRADUCCIÓN DE PATENTE EUROPEA

T3

96 Fecha de presentación y número de la solicitud europea: **23.02.2018** **E 21203463 (1)**

97 Fecha y número de publicación de la concesión europea: **21.02.2024** **EP 3964927**

54 Título: **Gestión de energía de procesador**

30 Prioridad:

01.04.2017 US 201715477029

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

07.11.2024

73 Titular/es:

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

72 Inventor/es:

KOKER, ALTUG;
APPU, ABHISHEK R.;
VEERNAPU, KIRAN C.;
RAY, JOYDEEP;
SEMBU, BALAJI;
SURTI, PRASOONKUMAR;
SINHA, KAMAL;
HOEKSTRA, ERIC J.;
FU, WENYIN;
KABURLASOS, NIKOS;
BOROLE, BHUSHAN M.;
SCHLUESSLER, TRAVIS T.;
SHAH, ANKUR N. y
KENNEDY, JONATHAN

74 Agente/Representante:

LEHMANN NOVO, María Isabel

ES 2 985 696 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

Gestión de energía de procesador

5 SECTOR

La presente invención se refiere, en general, al sector de la electrónica. Más concretamente, algunas realizaciones se refieren a técnicas para implementar la gestión de energía de procesador.

10 ESTADO DE LA TÉCNICA ANTERIOR

A medida que mejora la tecnología de fabricación de circuitos integrados, los fabricantes pueden integrar funciones adicionales en un solo sustrato de silicio. A medida que aumenta el número de funciones, también lo hace el número de componentes en un solo chip de circuito integrado (IC). Los componentes adicionales añaden conmutación de señal adicional y, a su vez, generan más calor y/o consumen más energía. El calor adicional puede dañar los componentes del chip, por ejemplo, por expansión térmica. Además, el consumo de energía adicional puede limitar las ubicaciones de uso y/o los modelos de uso para dichos dispositivos, por ejemplo, especialmente para los dispositivos que dependen de la energía de la batería para funcionar. Por lo tanto, la gestión eficiente de la energía puede tener un impacto directo en la eficiencia, la longevidad y los modelos de uso de los dispositivos electrónicos.

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

Para aumentar adicionalmente el rendimiento, los procesadores de gráficos habitualmente implementan técnicas de procesamiento, tales como canalización, que intenta procesar, en paralelo, tantos datos de gráficos como sea posible a lo largo de todas las diferentes partes de la canalización de gráficos. Los procesadores gráficos en paralelo con arquitecturas de una sola instrucción y múltiples subprocesos (SIMT) están diseñados para maximizar la cantidad de procesamiento en paralelo en el canal de gráficos. En una arquitectura de SIMT, grupos de subprocesos en paralelo intentan ejecutar instrucciones de programa de manera sincrónica conjuntamente con la mayor frecuencia posible para aumentar la eficiencia de procesamiento. Puede encontrarse una vista global general del software y hardware para arquitecturas SIMT en Shane Cook, CUDA Programming, capítulo 3, páginas 37-51 (2013) y/o Nicholas Wilt, CUDA Handbook, A Comprehensive Guide to GPU Programming, secciones 2.6.2 a 3.1.2 (junio de 2013).

La Patente US 2014/125679 A1 da a conocer que los cuellos de botella de rendimiento que surgen en determinados recursos dentro de una unidad de procesador gráfico pueden aliviarse reequilibrando dinámicamente las cargas de trabajo entre los recursos, con el objetivo de eliminar el cuello de botella de rendimiento actual y, al mismo tiempo, mantener la disipación de energía dentro de un presupuesto de energía actualmente asignado.

La Patente US 7,903,116 B1 da a conocer que un sistema de gráficos adapta un nivel de rendimiento para que sea suficiente para mantener un criterio de rendimiento en un rango aceptable. Se monitoriza al menos un parámetro de utilización del dominio de reloj del núcleo y del dominio de reloj de la memoria. En respuesta a la detección de una condición de sobreutilización, el nivel de rendimiento se incrementa para mantener el número mínimo deseado de tramas por segundo. En respuesta a la detección de una condición de infrautilización, el nivel de rendimiento se disminuye para reducir el consumo de energía y aumentar la vida útil del sistema de gráficos.

La Patente US 2016/225348 A1 da a conocer técnicas en las que un procesador determina un nivel de rendimiento esperado de una unidad de procesamiento de gráficos (GPU) basándose en una cantidad de comandos que la GPU debe ejecutar. El procesador emite información que indica el nivel de rendimiento, y la GPU ajusta su velocidad de reloj antes de la ejecución de los comandos que se utilizaron para determinar el nivel de rendimiento.

La invención se define por un procedimiento, un aparato y un medio legible por ordenador según las reivindicaciones independientes.

BREVE DESCRIPCIÓN DE LOS DIBUJOS

Para que la forma en que las características citadas anteriormente de las presentes realizaciones puedan entenderse en detalle, se puede obtener una descripción más particular de las realizaciones, brevemente resumida anteriormente, con referencia a las realizaciones, algunas de las cuales se ilustran en los dibujos adjuntos. Sin embargo, se ha de observar que los dibujos adjuntos ilustran únicamente realizaciones típicas y, por lo tanto, no han de considerarse limitativas de su alcance. La presente invención está dirigida a las figuras 6 y 7 y sus correspondientes párrafos descriptivos. Las figuras restantes y su descripción se proporcionan con fines ilustrativos.

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

Las figuras 2A a 2D ilustran los componentes de un procesador en paralelo, de acuerdo con una realización.

Las figuras 3A a 3B son diagramas de bloques de multiprocesadores de gráficos, de acuerdo con las realizaciones.

Las figuras 4A a 4F ilustran una arquitectura a modo de ejemplo en la que una pluralidad de GPU están acopladas en comunicación a una pluralidad de procesadores de múltiples núcleos.

La figura 5 ilustra una canalización de procesamiento de gráficos, de acuerdo con una realización.

La figura 6A es un diagrama de flujo que ilustra las operaciones de un procedimiento de gestión de energía del procesador, de acuerdo con las realizaciones.

La figura 6B es un diagrama de bloques esquemático de una arquitectura para implementar un procedimiento para la gestión de energía del procesador, de acuerdo con las realizaciones.

La figura 6C es un diagrama de flujo que ilustra las operaciones de un procedimiento de gestión de energía del procesador, de acuerdo con las realizaciones.

La figura 6D es un diagrama de bloques esquemático de una arquitectura para implementar un procedimiento para la gestión de energía del procesador, de acuerdo con las realizaciones.

La figura 7A es un diagrama esquemático de bloques de una arquitectura para implementar un procedimiento de gestión de energía del procesador, de acuerdo con las realizaciones.

Las figuras 7B y 7C son diagramas de flujo que ilustran operaciones en un procedimiento para la gestión de energía del procesador, de acuerdo con las realizaciones.

La figura 8 ilustra un diagrama de bloques de un regulador de conmutación de acuerdo con una realización.

La figura 9 es un diagrama de bloques de un sistema que incluye un multiprocesador de transmisión en continuo, de acuerdo con una o varias realizaciones.

La figura 10 ilustra un diagrama de bloques de un sistema de procesamiento en paralelo, de acuerdo con una realización.

La figura 11 es un diagrama de bloques de un sistema de procesamiento, de acuerdo con una realización.

La figura 12 es un diagrama de bloques de un procesador de acuerdo con una realización.

La figura 13 es un diagrama de bloques de un procesador de gráficos, de acuerdo con una realización.

La figura 14 es un diagrama de bloques de un motor de procesamiento de gráficos de un procesador de gráficos de acuerdo con algunas realizaciones.

La figura 15 es un diagrama de bloques de un procesador de gráficos proporcionado por una realización adicional.

La figura 16 ilustra una lógica de ejecución de subprocesos que incluye una matriz de elementos de procesamiento empleados en algunas realizaciones.

La figura 17 es un diagrama de bloques que ilustra unos formatos de instrucción de procesador de gráficos de acuerdo con algunas realizaciones.

La figura 18 es un diagrama de bloques de un procesador de gráficos de acuerdo con otra realización.

Las figuras 19A y 19B ilustran un formato de comandos de procesador de gráficos y una secuencia de comandos, de acuerdo con algunas realizaciones.

La figura 20 ilustra una arquitectura de software de gráficos a modo de ejemplo para un sistema de procesamiento de datos de acuerdo con algunas realizaciones.

La figura 21 es un diagrama de bloques que ilustra un sistema de desarrollo de núcleo IP, de acuerdo con una realización.

La figura 22 es un diagrama de bloques que ilustra un circuito integrado de sistema en chip a modo de ejemplo, de acuerdo con una realización.

La figura 23 es un diagrama de bloques que ilustra un procesador de gráficos a modo de ejemplo adicional.

La figura 24 es un diagrama de bloques que ilustra un procesador de gráficos adicional a modo de ejemplo de un circuito integrado de sistema en chip, de acuerdo con una realización.

DESCRIPCIÓN DETALLADA

En la siguiente descripción, se exponen numerosos detalles específicos para proporcionar un entendimiento completo de diversas realizaciones. Sin embargo, se pueden poner en práctica diversas realizaciones sin los detalles específicos. En otros casos, los métodos, procedimientos, componentes y circuitos bien conocidos no se han descrito en detalle para no oscurecer las realizaciones particulares. Además, diversos aspectos de las realizaciones se pueden realizar usando diversos medios, tales como circuitos semiconductores integrados ("hardware"), instrucciones legibles por ordenador organizadas en uno o varios programas ("software") o alguna combinación de hardware y software. A los efectos de esta invención, la referencia a "lógica" significará hardware, software, firmware o alguna combinación de los mismos.

Algunas realizaciones analizadas en el presente documento pueden aplicarse en cualquier procesador (tal como GPCPU, CPU, GPU, etc.), controladores de gráficos, etc. También se dan a conocer y reivindican otras realizaciones. Además, algunas realizaciones se pueden aplicar en sistemas informáticos que incluyen uno o varios procesadores (por ejemplo, con uno o varios núcleos de procesador), tales como los que se analizan en el presente documento, incluidos, por ejemplo, dispositivos informáticos móviles, por ejemplo, un teléfono inteligente, tableta, UMPC (ordenador personal ultramóvil), ordenador portátil, dispositivo informático Ultrabook™, dispositivos ponibles (tales como un reloj inteligente o gafas inteligentes), etc.

En algunas realizaciones, una unidad de procesamiento gráfico (GPU) se acopla en comunicación con núcleos del sistema central/procesador para acelerar las operaciones de gráficos, operaciones de aprendizaje automatizado, operaciones de análisis de patrones y diversas funciones de GPU de propósito general (GPGPU). La GPU se puede acoplar en comunicación con el procesador/núcleos del sistema central a través de un bus u otra interconexión (por ejemplo, una interconexión de alta velocidad tal como por ejemplo PCIe o NVLink). En otras realizaciones, la GPU se puede integrar en el mismo paquete o chip que los núcleos y se puede acoplar en comunicación con los núcleos a través de un bus/interconexión interna del procesador (es decir, interna al paquete o chip). Independientemente de la manera en la que esté conectada la GPU, los núcleos del procesador pueden asignar trabajo a la GPU en forma de secuencias de comandos/instrucciones contenidas en un descriptor de trabajo. La GPU utiliza entonces circuitos/lógica dedicados para procesar eficientemente 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 la materia que las realizaciones descritas en el presente documento pueden llevarse a la práctica sin uno o varios de estos detalles específicos. En otros casos, no se han descrito características bien conocidas para evitar oscurecer los detalles de las presentes realizaciones.

Vista general del sistema

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

En una realización, el subsistema de procesamiento 101 incluye uno o varios procesadores en paralelo 112 acoplados al concentrador de memoria 105 a través de un bus u otro enlace de comunicación 113. El enlace de comunicación 113 puede ser uno de cualquier número de tecnologías o protocolos de enlace de comunicación basados en estándares, tales como, pero de forma no limitativa, PCI Express, o puede ser una interfaz de comunicaciones o tejido de comunicaciones específicos del proveedor. En una realización, el uno o varios procesadores en paralelo 112 forman

un sistema de procesamiento vectorial o paralelo enfocado informáticamente que incluye un gran número de núcleos de procesamiento y/o agrupamientos de procesamiento, tales como un procesador de muchos núcleos integrados (MIC). En una realización, el uno o varios procesadores en paralelo 112 forman un subsistema de procesamiento de gráficos que puede emitir píxeles a uno del uno o varios dispositivos de visualización 110A acoplados a través del concentrador de E/S 107. El uno o varios procesadores en paralelo 112 también pueden incluir un controlador de visualización y una interfaz de visualización (no mostrada) para permitir una conexión directa a uno o varios dispositivos de visualización 110B.

Dentro del subsistema de E/S 111, una unidad de almacenamiento de sistema 114 puede conectarse al concentrador de E/S 107 para proporcionar un mecanismo de almacenamiento para el sistema informático 100. Un conmutador de E/S 116 puede utilizarse para proporcionar un mecanismo de interfaz que permita conexiones entre el concentrador de E/S 107 y otros componentes, tales como un adaptador de red 118 y/o un adaptador de red inalámbrico 119 que pueden integrarse en la plataforma, y otros diversos dispositivos que pueden añadirse a través de uno o varios dispositivos complementarios 120. El adaptador de red 118 puede ser un adaptador de Ethernet u otro adaptador de red por cable. El adaptador de red inalámbrico 119 puede incluir uno o varios de un dispositivo de red de Wi-Fi, de Bluetooth, de comunicación de campo cercano (NFC) o de otro tipo que incluye una o varias radios inalámbricas.

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

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

Se apreciará que el sistema informático 100 mostrado en el presente documento es ilustrativo y que son posibles variaciones y modificaciones. La topología de conexión, incluyendo el número y disposición de los puentes, el número de procesador o procesadores 102, y el número de procesador o procesadores en paralelo 112 puede modificarse según se desee. Por ejemplo, en algunas realizaciones, la memoria del sistema 104 está conectada al procesador o procesadores 102 directamente en lugar de a través de un puente, mientras que otros dispositivos se comunican con la memoria del sistema 104 a través del concentrador de memoria 105 y el procesador o procesadores 102. En otras topologías alternativas, el procesador o procesadores en paralelo 112 están conectados al concentrador de E/S 107 o directamente a uno del uno o varios procesadores 102, en lugar de al concentrador de memoria 105. En otras realizaciones, el concentrador de E/S 107 y el concentrador de memoria 105 pueden estar integrados en un único chip. Algunas realizaciones pueden incluir dos o más conjuntos de procesador o procesadores 102 adjuntos mediante múltiples zócalos, que pueden acoplarse con dos o más instancias del procesador o 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 implementaciones del sistema informático 100. Por ejemplo, puede admitirse cualquier número de tarjetas o periféricos complementarios, o pueden eliminarse algunos componentes. Además, algunas arquitecturas pueden utilizar una terminología diferente para componentes similares a los ilustrados en la figura 1. Por ejemplo, el concentrador de memoria 105 puede denominarse Northbridge en algunas arquitecturas, mientras que el concentrador de E/S 107 puede denominarse Southbridge.

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

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

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

La matriz de agrupamientos de procesamiento 212 puede incluir hasta "N" agrupamientos de procesamiento (por ejemplo, el agrupamiento 214A, el agrupamiento 214B al agrupamiento 214N). Cada agrupamiento 214A a 214N de la matriz de agrupamientos de procesamiento 212 puede ejecutar un gran número de subprocesos concurrentes. El planificador 210 puede asignar trabajo a los agrupamientos 214A a 214N del conjunto de agrupamientos de procesamiento 212 usando varios algoritmos de planificación y/o distribución de trabajo, que pueden variar dependiendo de la carga de trabajo que surja para cada tipo de programa o cálculo. La planificación puede ser manejada dinámicamente por el planificador 210, o puede ser asistida en parte por la lógica del compilador durante la compilación de la lógica del programa configurado para su ejecución por el conjunto de agrupamientos de procesamiento 212. En una realización, se pueden asignar diferentes agrupamientos 214A a 214N de la matriz de agrupamientos de procesamiento 212 para procesar diferentes tipos de programas o para realizar diferentes tipos de cálculos.

La matriz de agrupamientos de procesamiento 212 se puede configurar para realizar diversos tipos de operaciones de procesamiento en paralelo. En una realización, la matriz de agrupamientos de procesamiento 212 está configurada para realizar operaciones informáticas en paralelo de propósito general. Por ejemplo, la matriz de agrupamientos de procesamiento 212 puede incluir lógica para ejecutar tareas de procesamiento, incluido el filtrado de datos de vídeo y/o audio, la realización de operaciones de modelado, incluidas operaciones de física, y la realización de transformaciones de datos.

En una realización, la matriz de agrupamientos de procesamiento 212 está configurada para realizar operaciones de procesamiento de gráficos en paralelo. En las realizaciones en las que el procesador en paralelo 200 está configurado para realizar operaciones de procesamiento de gráficos, la matriz de agrupamientos de procesamiento 212 puede incluir lógica adicional para admitir la ejecución de dichas operaciones de procesamiento de gráficos, incluyendo, de forma no limitativa, lógica de muestreo de textura para realizar operaciones de textura, así como lógica de teselado y otra lógica de procesamiento de vértices. Adicionalmente, la matriz de agrupamientos de procesamiento 212 puede configurarse para ejecutar programas de sombreado relacionados con el procesamiento de gráficos tales como, de forma no limitativa, 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 a través de la unidad de E/S 204 para su procesamiento. Durante el procesamiento, los datos transferidos pueden almacenarse en una memoria en chip (por ejemplo, la memoria de procesador en paralelo 222) durante el procesamiento y, a continuación, escribirse de nuevo en la memoria del sistema.

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

Durante la operación, la matriz de agrupamientos de procesamiento 212 puede recibir tareas de procesamiento para ser ejecutadas a través del planificador 210, que recibe comandos que definen las tareas de procesamiento desde el extremo frontal 208. Para operaciones de procesamiento de gráficos, las tareas de procesamiento pueden incluir índices de datos que hay que procesar, por ejemplo, datos de superficie (parche), datos de primitiva, datos de vértice y/o datos de píxel, así como parámetros de estado y comandos que definen cómo han de procesarse los datos (por ejemplo, qué programa ha de ejecutarse). El planificador 210 puede configurarse para extraer los índices que corresponden a las tareas o puede recibir los índices desde el extremo frontal 208. El extremo frontal 208 puede estar

configurado para garantizar que la matriz de agrupamientos de procesamiento 212 esté configurada en un estado válido antes de que se inicie la carga de trabajo especificada en memorias intermedias de comandos entrantes (por ejemplo, memorias intermedias de lotes, memorias intermedias de carga, etc.).

Cada una de la una o varias instancias de la unidad de procesamiento en paralelo 202 puede acoplarse con la memoria de procesador en paralelo 222. Se puede acceder a la memoria de procesador en paralelo 222 a través de la barra transversal de memoria 216, que puede recibir peticiones de memoria de la matriz de agrupamientos de procesamiento 212 así como de la unidad de E/S 204. La barra transversal de memoria 216 puede acceder a la memoria de procesador en paralelo 222 por medio de una interfaz de memoria 218. La interfaz de memoria 218 puede incluir múltiples unidades de partición (por ejemplo, unidad de partición 220A, unidad de partición 220B, hasta la unidad de partición 220N), donde cada una puede acoplarse a una porción (por ejemplo, unidad de memoria) de la memoria de procesador en paralelo 222. En una implementación, el número de unidades de partición 220A a 220N está configurado para que sea igual al número de unidades de memoria, de tal manera que una primera unidad de partición 220A tiene una primera unidad de memoria 224A correspondiente, una segunda unidad de partición 220B tiene una unidad de memoria 224B correspondiente y una N-ésima unidad de partición 220N tiene una N-ésima unidad de memoria 224N correspondiente. En otras realizaciones, el número de unidades de partición 220A a 220N puede no ser igual al número de dispositivos de memoria.

En varias realizaciones, las unidades de memoria 224A a 224N pueden incluir varios tipos de dispositivos de memoria, incluyendo memoria de acceso aleatorio dinámico (DRAM) o memoria de acceso aleatorio de gráficos, tal como memoria de acceso aleatorio de gráficos sincrónica (SGRAM), incluyendo memoria de doble velocidad de datos de gráficos (GDDR). En una realización, las unidades de memoria 224A a 224N también pueden incluir memoria 3D apilada, que incluye, pero sin limitación, memoria de ancho de banda alto (HBM). Los expertos en la materia apreciarán que la implementación específica de las unidades de memoria 224A a 224N puede variar y que puede seleccionarse de entre uno de varios diseños convencionales. Se pueden almacenar objetivos de renderización, tales como memorias intermedias de tramas o correlaciones de textura, en las unidades de memoria 224A a 224N, lo que permite que las unidades de partición 220A a 220N escriban porciones de cada objetivo de renderización en paralelo para usar de manera eficaz el ancho de banda disponible de la memoria de procesador en paralelo 222. En algunas realizaciones, se puede excluir una instancia local de la memoria de procesador en paralelo 222 en favor de un diseño de memoria unificada que utiliza memoria de sistema junto con memoria caché local.

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

Aunque se ilustra una única instancia de la unidad de procesamiento paralelo 202 dentro del procesador en paralelo 200, se puede incluir cualquier número de instancias de la unidad de procesamiento paralelo 202. Por ejemplo, se pueden proporcionar múltiples instancias de la unidad de procesamiento en paralelo 202 en una única tarjeta adicional o se pueden interconectar múltiples tarjetas adicionales. Las diferentes instancias de la unidad de procesamiento en paralelo 202 pueden configurarse para interoperar incluso si las diferentes instancias tienen diferentes números de núcleos de procesamiento, diferentes cantidades de memoria de procesador en paralelo local y/u otras diferencias de configuración. Por ejemplo, y en una realización, algunas instancias de la unidad de procesamiento en paralelo 202 pueden incluir unidades de coma flotante de precisión más alta con relación a otras instancias. Los sistemas que incorporan una o varias instancias de la unidad de procesamiento en paralelo 202 o el procesador en paralelo 200 se pueden implementar en una diversidad de configuraciones y factores de forma, incluyendo, de forma no limitativa, ordenadores personales de sobremesa, portátiles o de mano, servidores, estaciones de trabajo, consolas de juegos y/o sistemas integrados.

La **figura 2B** es un diagrama de bloques de una unidad de partición 220, de acuerdo con una realización. En una realización, la unidad de partición 220 es una instancia de una de las unidades de partición 220A a 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 rasterización). La memoria caché L2 221 es una memoria caché de lectura/escritura que está configurada para realizar operaciones de carga y almacenamiento recibidas desde la barra transversal de memoria 216 y la ROP 226. Los errores de lectura y las peticiones urgentes de escritura son enviadas por la memoria caché L2 221 a la interfaz de memoria intermedia de tramas 225 para su procesamiento. Las actualizaciones incorrectas también pueden enviarse a la memoria intermedia de tramas a través

de la interfaz de memoria intermedia de tramas 225 para su procesamiento oportunista. En una realización, la interfaz de memoria intermedia de tramas 225 interactúa con una de las unidades de memoria de la memoria del procesador en paralelo, tal como las unidades de memoria 224A a 224N de la figura 2 (por ejemplo, dentro de la memoria 222 del procesador en paralelo).

En las aplicaciones de gráficos, la ROP 226 es una unidad de procesamiento que realiza operaciones de rasterización tales como estarcido, prueba z, mezcla y similares. La ROP 226 a continuación emite los datos de gráficos procesados que se almacenan en la memoria de gráficos. En algunas realizaciones, la ROP 226 incluye lógica de compresión para comprimir los datos z o de color que se escriben en la memoria y descomprimir los datos z o de color que se leen de la memoria. En algunas realizaciones, la ROP 226 está incluida dentro de cada agrupamiento de procesamiento (por ejemplo, agrupamiento 214A a 214N de la figura 2) en lugar de dentro de la unidad de partición 220. En tal realización, las solicitudes de lectura y escritura de datos de píxeles se transmiten a través de la barra transversal de memoria 216 en lugar de datos de fragmentos de píxeles. Los datos de gráficos procesados pueden ser mostrados en un dispositivo de visualización, tal como uno de uno o varios dispositivos de visualización 110 de la figura 1, encaminados para su posterior procesamiento por el procesador(es) 102, o encaminados 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 agrupamiento de procesamiento 214 dentro de una unidad de procesamiento en paralelo, de acuerdo con una realización. En una realización, el agrupamiento de procesamiento es una instancia de uno de los agrupamientos de procesamiento 214A a 214N de la figura 2. El agrupamiento de procesamiento 214 puede configurarse para ejecutar muchos subprocesos en paralelo, donde el término "subproceso" se refiere a una instancia de un programa particular que se ejecuta en un conjunto particular de datos de entrada. En algunas realizaciones, se usan técnicas de emisión de instrucciones de única instrucción, de múltiples datos (SIMD) para admitir la ejecución paralela de un gran número de subprocesos sin proporcionar múltiples unidades de instrucción independientes. En otras realizaciones, se utilizan técnicas de una sola instrucción, múltiples subprocesos (SIMT) para admitir la ejecución paralela de un gran número de subprocesos generalmente sincronizados, utilizando una unidad de instrucción común configurada para emitir instrucciones a un conjunto de motores de procesamiento dentro de cada uno de los agrupamientos de procesamiento. A diferencia del régimen de ejecución de SIMD, donde todos los motores de procesamiento ejecutan habitualmente instrucciones idénticas, la ejecución de SIMT permite que diferentes subprocesos sigan más fácilmente rutas de ejecución divergentes a través de un programa de subprocesos dado. Los expertos en la materia entenderán que un régimen de procesamiento SIMD representa un subconjunto funcional de un régimen de procesamiento SIMT.

El funcionamiento del agrupamiento de procesamiento 214 puede controlarse a través de 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 a través de un multiprocesador de gráficos 234 y/o una unidad de texturas 236. El multiprocesador de gráficos 234 ilustrado es una instancia a modo de ejemplo de un procesador en paralelo de SIMT. Sin embargo, varios tipos de procesadores en paralelo SIMT de diferentes arquitecturas pueden incluirse dentro del agrupamiento de procesamiento 214. Una o varias instancias del multiprocesador de gráficos 234 pueden incluirse dentro de un agrupamiento de procesamiento 214. El multiprocesador de gráficos 234 puede procesar datos y una barra transversal de datos 240 se puede utilizar para distribuir los datos procesados a uno de los múltiples destinos posibles, incluyendo otras unidades de sombreado. El gestor de canalización 232 puede facilitar la distribución de los datos procesados mediante la especificación de destinos para los datos procesados que se distribuirán a través de la barra transversal de datos 240.

Cada multiprocesador de gráficos 234 dentro del agrupamiento de procesamiento 214 puede incluir un conjunto idéntico de lógica de ejecución funcional (por ejemplo, unidades aritmético-lógicas, unidades de carga-almacenamiento, etc.). La lógica de ejecución funcional puede configurarse de una manera canalizada en la que pueden emitirse nuevas instrucciones antes de que se hayan completado instrucciones previas. 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 de cálculo de diversas funciones algebraicas. En una realización, el mismo hardware de unidad funcional puede aprovecharse para realizar diferentes operaciones y puede estar presente cualquier combinación de unidades funcionales.

Las instrucciones transmitidas al agrupamiento de procesamiento 214 constituyen un subproceso. Un conjunto de subprocesos ejecutándose a través del 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 puede ser asignado 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 varios de los motores de procesamiento pueden estar inactivos durante los ciclos en los que se procesa ese grupo de subprocesos. Un grupo de subprocesos puede incluir también 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 puede realizarse durante ciclos de reloj consecutivos.

En una realización, múltiples grupos de subprocesos pueden ejecutarse concurrentemente en un multiprocesador de gráficos 234.

En una realización, el multiprocesador de gráficos 234 incluye una memoria caché interna para realizar operaciones de carga y almacenamiento. En una realización, el multiprocesador de gráficos 234 puede renunciar a una memoria caché interna y utilizar una memoria caché (por ejemplo, la memoria caché L1 308) dentro del agrupamiento de procesamiento 214. Cada multiprocesador de gráficos 234 también tiene acceso a memorias caché L2 dentro de las unidades de partición (por ejemplo, las unidades de partición 220A a 220N de la figura 2) que se comparten entre todos los agrupamientos de procesamiento 214 y se pueden usar para transferir datos entre subprocesos. El multiprocesador de gráficos 234 puede acceder también a memoria global fuera de chip, que puede incluir uno o varios de memoria de procesador en paralelo local y/o memoria de sistema. Cualquier memoria externa a la unidad de procesamiento en paralelo 202 puede utilizarse como memoria global. Las realizaciones en las que el agrupamiento 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 agrupamiento de procesamiento 214 puede incluir una MMU 245 (unidad de gestión de memoria) que está configurada para correlacionar direcciones virtuales en direcciones físicas. En otras realizaciones, una o varias instancias de la MMU 245 pueden residir dentro de la interfaz de memoria 218 de la figura 2. La MMU 245 incluye un conjunto de entradas de tabla de página (PTE) usadas para correlacionar una dirección virtual con una dirección física de una tesela (más información sobre la generación de teselas) y, opcionalmente, un índice de línea de memoria caché. La MMU 245 puede incluir memorias intermedias de conversión adelantada (TLB) de dirección o memorias caché que pueden residir dentro del multiprocesador de gráficos 234 o la memoria caché L1 o el agrupamiento de procesamiento 214. La dirección física se procesa para distribuir la localidad de acceso a los datos de superficie para permitir un entrelazado eficiente de solicitudes entre unidades de partición. El índice de línea de memoria caché se puede usar para determinar si una solicitud para una línea de memoria caché es un acierto o un fallo.

En aplicaciones de gráficos y cálculo, se puede configurar un agrupamiento de procesamiento 214 de tal manera que cada multiprocesador de gráficos 234 esté acoplado a una unidad de textura 236 para realizar operaciones de correlaciones de textura, por ejemplo, determinar posiciones de muestras de textura, leer datos de textura y filtrar los datos de textura. Los datos de textura se leen desde una memoria caché L1 de textura interna (no mostrada) o, en algunas realizaciones, desde la memoria caché L1 dentro del multiprocesador de gráficos 234 y se extraen desde una memoria caché L2, memoria de procesador en paralelo local o memoria de sistema, según sea necesario. Cada multiprocesador de gráficos 234 envía las tareas procesadas a la barra transversal de datos 240 para proporcionar la tarea procesada a otro agrupamiento 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 a través de la barra transversal de memoria 216. Una preROP 242 (unidad de operaciones previas a la rasterización) está configurada para recibir datos desde el multiprocesador de gráficos 234, dirigir datos a las unidades de ROP, que pueden estar ubicadas con unidades de partición como se describe en el presente documento (por ejemplo, las unidades de partición 220A a 220N de la figura 2). La unidad preROP 242 puede realizar optimizaciones para la mezcla de color, organizar datos de color de píxel y realizar conversiones de dirección.

Se apreciará que la arquitectura de núcleo descrita en el presente documento es ilustrativa y que son posibles variaciones y modificaciones. Cualquier número de unidades de procesamiento, por ejemplo, multiprocesador de gráficos 234, unidades de textura 236, preROP 242, etc., pueden incluirse dentro de un agrupamiento de procesamiento 214. Además, si bien solo se muestra un agrupamiento de procesamiento 214, una unidad de procesamiento en paralelo como la descrita en el presente documento puede incluir cualquier número de instancias del agrupamiento de procesamiento 214. En una realización, cada agrupamiento de procesamiento 214 puede ser configurado para operar independientemente de otros agrupamientos de procesamiento 214 usando unidades de procesamiento separadas y distintas, memorias caché L1, etc.

La figura 2D muestra un multiprocesador de gráficos 234, de acuerdo con una realización. En tal realización, el multiprocesador de gráficos 234 se acopla con el gestor de canalización 232 del agrupamiento de procesamiento 214. El multiprocesador de gráficos 234 tiene una canalización de ejecución que incluye, de forma no limitativa, una memoria caché de instrucciones 252, una unidad de instrucciones 254, una unidad de asignación de direcciones 256, un archivo de registro 258, uno o varios núcleos de unidad de procesamiento de gráficos de propósito general (GPGPU) 262 y una o varias unidades de carga/almacenamiento 266. Los núcleos de GPGPU 262 y las unidades de carga/almacenamiento 266 están acoplados con la memoria caché 272 y a la memoria compartida 270 mediante una interconexión de memoria y caché 268.

En una realización, la memoria caché de instrucciones 252 recibe un flujo de instrucciones a ejecutar desde el gestor de canalización 232. Las instrucciones se almacenan en memoria caché en la memoria caché de instrucciones 252 y son despachadas para su ejecución por la unidad de instrucciones 254. La unidad de instrucción 254 puede despachar instrucciones como grupos de subprocesos (por ejemplo, *warps*), con cada subproceso del grupo de subprocesos asignado a una unidad de ejecución diferente dentro del núcleo GPGPU 262. Una instrucción puede acceder a cualquiera del espacio de direcciones local, compartido o global, especificando una dirección dentro de un espacio de direcciones unificado. La unidad de correlación de direcciones 256 puede utilizarse para traducir direcciones en el

espacio de direcciones unificado a una dirección de memoria distinta que pueda ser accedida por las unidades de carga/almacenamiento 266.

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

Cada núcleo de GPGPU 262 puede incluir unidades de coma flotante (FPU) y/o unidades aritmético-lógicas (ALU) de números enteros que se usan para ejecutar instrucciones del multiprocesador de gráficos 324. Los núcleos de GPGPU 262 pueden ser similares en arquitectura o pueden diferir en arquitectura, según las realizaciones. Por ejemplo, y en una realización, una primera porción de los núcleos de GPGPU 262 incluye una FPU de precisión sencilla y una ALU de números enteros, mientras que una segunda porción de los núcleos de GPGPU incluye una FPU de precisión doble. En una realización, las FPU pueden implementar la norma IEEE 754-2008 para aritmética de coma flotante o posibilitar aritmética de coma flotante de precisión variable. El multiprocesador de gráficos 324 puede incluir adicionalmente una o varias unidades de función fija o de función especial para realizar funciones específicas, tales como operaciones de copia de rectángulo o de mezcla de píxeles. En una realización, uno o varios de los núcleos de GPGPU puede incluir también lógica de función fija o especial.

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

Las figuras 3A y 3B ilustran multiprocesadores de gráficos adicionales, de acuerdo con las realizaciones. Los multiprocesadores de gráficos 325, 350 ilustrados son variantes del multiprocesador de gráficos 324 de la figura 2C. Los multiprocesadores de gráficos 325, 350 ilustrados pueden configurarse como un multiprocesador de transmisión en continuo (SM) capaz de ejecutar simultáneamente un gran número de subprocesos de ejecución.

La figura 3A muestra un multiprocesador de gráficos 325 de acuerdo con una realización adicional. El multiprocesador de gráficos 325 incluye múltiples instancias adicionales de unidades de recurso de ejecución relativas al multiprocesador de gráficos 324 de la figura 2D. Por ejemplo, el multiprocesador de gráficos 325 puede incluir múltiples instancias de la unidad de instrucción 332A-332B, archivo de registro 334A-334B, y unidad(es) de textura 344A-344B. El multiprocesador de gráficos 325 también incluye múltiples conjuntos de unidades de ejecución de gráficos o de cálculo (por ejemplo, núcleo de GPGPU 336A-336B, núcleo de GPGPU 337A-337B, núcleo de GPGPU 338A-338B) y múltiples conjuntos de unidades de carga/almacenamiento 340A-340B. En una realización, las unidades de recurso de ejecución tienen una memoria caché de instrucciones común 330, memoria caché de textura y/o datos 342 y memoria compartida 346. Los diversos componentes pueden comunicarse mediante una estructura de interconexión 327. En una realización, la estructura de interconexión 327 incluye uno o varios conmutadores de barra transversal para permitir la comunicación entre los diversos componentes del multiprocesador de gráficos 325.

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

Los expertos en la materia entenderán que la arquitectura descrita en las figuras 1, 2A a 2D y 3A-3B es descriptiva y no limitativa en cuanto al alcance de las presentes realizaciones. Por lo tanto, las técnicas descritas en el presente documento se pueden implementar en cualquier unidad de procesamiento configurada correctamente, incluyendo, sin limitación, uno o varios procesadores de aplicaciones móviles, una o varias unidades centrales de procesamiento (CPU) de escritorio o de servidor, incluyendo CPU de múltiples núcleos, una o varias unidades de procesamiento en paralelo, tales como la unidad de procesamiento en paralelo 202 de la figura 2, así como uno o varios procesadores gráficos o unidades de procesamiento para propósitos especiales, sin apartarse del alcance de las realizaciones descritas en el presente documento.

En algunas realizaciones, un procesador en paralelo o GPGPU, como se describe en el presente documento, está acoplado de manera comunicativa a núcleos de anfitrión/procesador para acelerar operaciones de gráficos, operaciones de aprendizaje automático, operaciones de análisis de patrones y diversas funciones de GPU de propósito general (GPGPU). La GPU puede estar acoplada en comunicación al procesador/núcleos anfitrión a través de un bus u otro tipo de interconexión (por ejemplo, una interconexión de alta velocidad como PCIe o NVLink). En otras realizaciones, la GPU se puede integrar en el mismo paquete o chip que los núcleos y se puede acoplar en comunicación con los núcleos a través de un bus/interconexión interna del procesador (es decir, interna al paquete o chip). Independientemente de la manera en la que esté conectada la GPU, los núcleos del procesador pueden asignar trabajo a la GPU en forma de secuencias de comandos/instrucciones contenidas en un descriptor de trabajo. La GPU utiliza entonces circuitos/lógica dedicados para procesar eficientemente estos comandos/instrucciones.

Técnicas de interconexión de GPU y procesador anfitrión

La **figura 4A** ilustra una arquitectura a modo de ejemplo en la que una pluralidad de GPU 410 a 413 están acopladas de manera comunicativa a una pluralidad de procesadores de múltiples núcleos 405-406 a través de enlaces de alta velocidad 440 a 443 (por ejemplo, buses, interconexiones de punto a punto, etc.). En una realización, los enlaces de alta velocidad 440 a 443 admiten un caudal de comunicación de 4 GB/s, 30 GB/s, 80 GB/s o mayor, dependiendo de la implementación. Se pueden utilizar varios protocolos de interconexión, incluyendo, entre otros, PCIe 4.0 o 5.0 y NVLink 2.0. Sin embargo, los principios subyacentes de la invención no están limitados a ningún protocolo o caudal de comunicación particular.

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

En una realización, cada procesador multinúcleo 405-406 se acopla en comunicación a una memoria de procesador 401-402, por medio de interconexiones de memoria 430-431, respectivamente, y cada GPU 410 a 413 se acopla en comunicación a la memoria GPU 420 a 423 a través de interconexiones de memoria GPU 450 a 453, respectivamente. Las interconexiones de memoria 430-431 y 450 a 453 pueden utilizar las mismas tecnologías de acceso de memoria, o unas diferentes. A modo de ejemplo, y no de limitación, las memorias de procesador 401-402 y las memorias de GPU 420 a 423 pueden ser memorias volátiles tales como memorias de acceso aleatorio dinámico (DRAM) (incluyendo DRAM apiladas), SDRAM DDR gráfica (GDDR) (por ejemplo, GDDR5, GDDR6), o memoria de alto ancho de banda (HBM) y/o pueden ser memorias no volátiles tales como 3D XPoint o Nano-Ram. En una realización, una parte de las memorias puede ser memoria volátil y otra parte puede ser memoria no volátil (por ejemplo, utilizando una jerarquía de memoria de dos niveles (2LM)).

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

La **figura 4B** ilustra detalles adicionales para una interconexión entre un procesador de múltiples núcleos 407 y un módulo de aceleración de gráficos 446 de acuerdo con una realización. El módulo de aceleración de gráficos 446 puede incluir uno o varios chips de GPU integrados en una tarjeta de línea que está acoplada al procesador 407 mediante el enlace de alta velocidad 440. Alternativamente, el módulo de aceleración de gráficos 446 puede estar integrado en el mismo paquete o chip que el procesador 407.

El procesador 407 ilustrado incluye una pluralidad de núcleos 460A a 460D, cada uno con una memoria intermedia de conversión adelantada de direcciones 461A a 461D y una o varias memorias caché 462A a 462D. Los núcleos pueden

incluir varios otros componentes para ejecutar instrucciones y procesar datos que no se ilustran para evitar oscurecer los principios subyacentes de la invención (por ejemplo, unidades de búsqueda de instrucciones, unidades de predicción de bifurcación, decodificadores, unidades de ejecución, memorias intermedias de reorden, etc.). Las memorias caché 462A a 462D pueden comprender memorias caché de nivel 1 (L1) y de nivel 2 (L2). Además, una o varias memorias caché compartidas 426 pueden incluirse en la jerarquía de almacenamiento en memoria caché y ser compartidas por conjuntos de núcleos 460A a 460D. Por ejemplo, una realización del procesador 407 incluye 24 núcleos, cada uno con su propia memoria caché L1, doce memorias caché L2 compartidas y doce memorias caché L3 compartidas. En esta realización, una de las memorias caché L2 y L3 están compartidas por dos núcleos adyacentes. El procesador 407 y el módulo de integración de acelerador de gráficos 446 se conectan con la memoria de sistema 441, que puede incluir memorias de procesador 401-402

La coherencia se mantiene para los datos y las instrucciones almacenadas en las diversas memorias caché 462A a 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/circuitería de coherencia de memoria caché asociada a la misma para comunicarse a través del bus de coherencia 464 en respuesta a lecturas o escrituras detectadas en líneas de memoria caché particulares. En una implementación, se implementa un protocolo de monitorización de memoria caché a través del bus de coherencia 464 para monitorizar los accesos a memoria caché. Las técnicas de monitorización/coherencia de memoria caché son bien entendidas por los expertos en la materia y no se describirán en detalle aquí para evitar oscurecer los principios subyacentes de la invención.

En una realización, un circuito proxy 425 acopla en comunicación el módulo de aceleración de gráficos 446 al bus de coherencia 464, permitiendo al módulo de aceleración de gráficos 446 participar en el protocolo de coherencia de memoria caché como un par de los núcleos. En particular, una interfaz 435 proporciona conectividad al circuito proxy 425 a través del enlace de alta velocidad 440 (por ejemplo, un bus PCIe, NVLink, etc.) y una interfaz 437 conecta el módulo de aceleración de gráficos 446 al enlace 440.

En una implementación, un circuito de integración de acelerador 436 proporciona servicios de gestión de memoria caché, de acceso a memoria, de gestión de contexto y de gestión de interrupciones en beneficio de una pluralidad de motores de procesamiento de gráficos 431, 432, N del módulo de aceleración de gráficos 446. Los motores de procesamiento de gráficos 431, 432, N pueden comprender cada uno una unidad de procesamiento de gráficos (GPU) independiente. Alternativamente, los motores de procesamiento de gráficos 431, 432, N pueden comprender diferentes tipos de motores de procesamiento de gráficos dentro de una GPU, tales como unidades de ejecución de gráficos, motores de procesamiento de medios (por ejemplo, codificadores/decodificadores de vídeo), muestreadores y motores blit. En otras palabras, el módulo de aceleración de gráficos puede ser una GPU con una pluralidad de motores de procesamiento de gráficos 431-432, N, o los motores de procesamiento de gráficos 431-432, N pueden ser GPU individuales integradas en un paquete, tarjeta de línea o chip común.

En una realización, el circuito de integración de acelerador 436 incluye una unidad de gestión de memoria (MMU) 439 para realizar diversas funciones de gestión de memoria tales como traducciones de memoria virtual a física (también denominadas traducciones de memoria efectiva a real) y protocolos de acceso de memoria para acceder a la memoria de sistema 441. La MMU 439 también puede incluir una memoria intermedia de conversión adelantada (TLB) (no mostrada) para almacenar en memoria caché las conversiones de direcciones virtuales/efectivas a físicas/reales. En una implementación, una memoria caché 438 almacena comandos y datos para un acceso eficaz por los motores de procesamiento de gráficos 431-432, N. En una realización, los datos almacenados en la memoria caché 438 y en las memorias de gráficos 433-434, N se mantienen coherentes con las memorias caché de núcleo 462A a 462D, 456 y la memoria de sistema 411. Tal como se ha mencionado, esto puede lograrse mediante el circuito proxy 425 que participa en el mecanismo de coherencia de memoria caché en nombre de la memoria caché 438 y las memorias 433-434, N (por ejemplo, enviando actualizaciones a la memoria caché 438 relacionadas con modificaciones/accesos de líneas de memoria caché en las memorias caché de procesador 462A a 462D, 456 y recibiendo actualizaciones de la memoria caché 438).

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

En una implementación, las direcciones virtuales/efectivas de un motor de procesamiento de gráficos 431 son convertidas a direcciones reales/físicas en la memoria del sistema 411 por la MMU 439. Una 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 ser dedicado a una única aplicación

ejecutada en el procesador 407 o puede ser compartido entre múltiples aplicaciones. En una realización, se presenta un entorno de ejecución de gráficos virtualizado en el que los recursos de los motores de procesamiento de gráficos 431-432, N se comparten con múltiples aplicaciones o máquinas virtuales (VM). Los recursos pueden subdividirse en "segmentos" que se asignan a diferentes máquinas virtuales y/o aplicaciones en función de los requisitos y prioridades de procesamiento asociados a las máquinas virtuales y/o aplicaciones.

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

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 anfitrión 407, cualquier procesador anfitrión puede dirigirse a estos recursos directamente utilizando un valor de dirección efectivo. Una función del circuito de integración de acelerador 436, en una realización, es la separación física de los motores de procesamiento de gráficos 431-432, N para que aparezcan ante el sistema como unidades independientes.

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

En una realización, para reducir el tráfico de datos a través del enlace 440, se utilizan técnicas de desvío para asegurar que los datos almacenados en las memorias de gráficos 433-434, M son 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 a 460D (al menos no con frecuencia). De manera similar, el mecanismo de desvío intenta mantener los datos que necesitan los núcleos (y, preferentemente, no los motores de procesamiento de gráficos 431-432, N) dentro de las memorias caché 462A a 462D, 456 de los núcleos y de la memoria de sistema 411.

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

Una realización admite diferentes modelos de programación, incluyendo un modelo de programación de proceso dedicado (sin virtualización del módulo de aceleración de gráficos) y modelos de programación compartida (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 realización del modelo de proceso dedicado, los motores de procesamiento de gráficos 431-432, N están dedicados a una única aplicación o proceso bajo un único sistema operativo. La aplicación única puede canalizar otras solicitudes de aplicación a los motores 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 VM/aplicación. Los modelos compartidos requieren un hipervisor de sistema para virtualizar los motores de procesamiento de gráficos 431-432, N para permitir el acceso por parte de cada sistema operativo. En sistemas de subdivisión única sin un hipervisor, los motores de procesamiento de gráficos 431-432, N pertenecen al sistema operativo. En ambos casos, el sistema operativo puede virtualizar los motores de procesamiento de gráficos 431-432, N para proporcionar acceso a cada proceso o aplicación.

Para el modelo de programación 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 realización, los elementos de proceso se almacenan en la memoria de sistema 411 y se pueden direccionar utilizando las técnicas de conversión de dirección efectiva a dirección real descritas en el presente documento. El identificador de proceso puede ser un valor específico de la implementación proporcionado al proceso de anfitrión cuando se registra su contexto con el motor de procesamiento de gráficos 431-432, N (es decir, llamando al software de sistema para añadir el elemento de proceso a la lista vinculada de elementos de proceso). Los 16 bits inferiores del 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. Como se usa en el presente documento, un "segmento" comprende una porción especificada de los recursos de procesamiento del circuito de integración de acelerador 436. El espacio de direcciones efectivas de aplicación 482 dentro de la memoria de sistema 411 almacena los elementos de proceso 483. En una realización, los elementos de proceso 483 se almacenan en respuesta a invocaciones de GPU 481 desde las aplicaciones 480 ejecutadas en el procesador 407. Un elemento de proceso 483 contiene el estado de proceso para la aplicación 480 correspondiente. Un descriptor de trabajo (WD) 484 contenido en el elemento de proceso 483 puede ser un único trabajo solicitado por una aplicación o puede contener un puntero a una cola de trabajos. En este último caso, el WD 484 es un puntero a la cola de solicitudes de trabajos en el espacio de direcciones 482 de la aplicación.

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

En una implementación, el modelo de programación de proceso dedicado es específico de la implementación. En este modelo, un único proceso es propietario del módulo de aceleración de gráficos 446 o de un motor de procesamiento de gráficos 431 individual. Debido a que el módulo de aceleración de gráficos 446 es propiedad de un único proceso, el hipervisor inicializa el circuito de integración de acelerador 436 para la partición propietaria y el sistema operativo inicializa el circuito de integración de 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 obtención de WD 491 en el segmento de integración de acelerador 490 extrae el siguiente WD 484, que incluye una indicación del trabajo que va a hacer uno de los motores de procesamiento de gráficos del módulo de aceleración de gráficos 446. Los datos del WD 484 se pueden almacenar en registros 445 y ser usados por la MMU 439, el circuito de gestión de interrupciones 447 y/o el circuito de gestión de contexto 446, como se ilustra. Por ejemplo, una realización de la MMU 439 incluye circuitos de recorrido de páginas/segmentos para acceder a tablas de segmentos/páginas 486 dentro del espacio de direcciones virtuales 485 del sistema operativo. El circuito de gestión de interrupciones 447 puede procesar eventos de interrupción 492 recibidos desde el módulo de aceleración de gráficos 446. Cuando se realizan operaciones de gráficos, una dirección efectiva 493 generada por un motor de procesamiento de gráficos 431-432, N se convierte en una dirección real por medio de la MMU 439.

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

Tabla 1 - Registros inicializados por hipervisor

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

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

Tabla 2 - Registros inicializados por el sistema operativo

1	Identificación de procesos y subprocesos
2	Puntero de almacenamiento/restauración de contexto de dirección efectiva (EA)
3	Puntero de registro de utilización de acelerador de dirección virtual (VA)
4	Puntero de tabla de segmentos de almacenamiento de dirección virtual (VA)
5	Máscara de autoridad

6 Descriptor de trabajo

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

La **figura 4E** ilustra detalles adicionales para una realización de un modelo compartido. Esta realización incluye un espacio de direcciones reales de hipervisor 498 en el que se almacena una lista de elementos de proceso 499. El espacio de direcciones reales del hipervisor 498 es accesible a través de 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 todas o un subconjunto de particiones en el sistema usen un módulo de aceleración de gráficos 446. Hay dos modelos de programación donde el módulo de aceleración de gráficos 446 está compartido por múltiples procesos y particiones: compartido en segmentos de tiempo y compartido dirigido a gráficos.

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

En una realización, para el modelo compartido, la aplicación 480 debe realizar una llamada al sistema 495 del sistema operativo con un tipo de módulo de aceleración de gráficos 446, un descriptor de trabajo (WD), un valor de registro de máscara de autoridad (AMR) y un puntero de área de almacenamiento/restauración de contexto (CSRP). El tipo del módulo de aceleración de gráficos 446 describe la función de aceleración dirigida como objetivo para la llamada de sistema. El tipo del módulo de aceleración de gráficos 446 puede ser un valor específico del sistema. El WD está formateado específicamente para el módulo de aceleración de gráficos 446 y puede tener la forma de un comando del módulo de aceleración de gráficos 446, un puntero de dirección efectiva a una estructura definida por el usuario, un puntero de dirección efectiva a una cola de comandos, o cualquier otra estructura de datos para describir el trabajo a realizar por el módulo de aceleración de gráficos 446. En una realización, el valor de AMR es el estado de AMR que se debe usar para el proceso actual. El valor pasado al sistema operativo es similar a que una aplicación establezca el AMR. Si las implementaciones del circuito de integración del acelerador 436 y del módulo de aceleración de gráficos 446 no admiten un Registro de Anulación de Máscara de Autoridad de Usuario (UAMOR), el sistema operativo puede aplicar el valor UAMOR actual al valor AMR antes de pasar el AMR en la llamada al hipervisor. El hipervisor 496 puede aplicar opcionalmente el valor actual del registro de anulación de máscara de autoridad (AMOR) antes de colocar el AMR en el elemento de proceso 483. En una realización, el CSRP es uno de los registros 445 que contienen la dirección efectiva de un área en el espacio de direcciones 482 de la aplicación para que el módulo de aceleración de gráficos 446 guarde y restaure el estado de contexto. Este puntero es opcional si no es necesario guardar un estado entre trabajos o cuando se adelanta un trabajo. El área de almacenamiento/restauración de contexto puede estar fijada en la memoria de sistema.

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

Tabla 3 - Parámetros de llamada de SO a hipervisor

1	Un descriptor de trabajo (WD)
2	Un valor de registro de máscara de autoridad (AMR) (potencialmente enmascarado).
3	Puntero de área de almacena(CSRP) de dirección efectiva (EA)
4	Un ID de proceso (PID) y un ID de subproceso (TID) opcional
5	Un puntero de registro de utilización de acelerador (AURP) de dirección virtual (VA)
6	La dirección virtual del puntero de la tabla de segmentos de almacenamiento (SSTP)
7	Un número de servicio de interrupción lógica (LISN)

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

Tabla 4 - Información de elemento de proceso

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

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

Tal como se ilustra en la **figura 4F**, una realización de la invención emplea una memoria unificada direccionable mediante un espacio de direcciones virtual de memoria común usado para acceder a las memorias físicas de procesador 401-402 y a las memorias de GPU 420 a 423. En esta implementación, las operaciones ejecutadas en las GPU 410 a 413 utilizan el mismo espacio de direcciones de memoria virtual/efectiva para acceder a las memorias de los procesadores 401-402 y viceversa, simplificando de este modo la programabilidad. En una realización, una primera porción del espacio de direcciones virtual/efectiva se asigna a la memoria del procesador 401, una segunda porción a la segunda memoria del procesador 402, una tercera porción a la memoria de la GPU 420, y así sucesivamente. Todo el espacio de memoria virtual/efectiva (a veces denominado espacio de direcciones efectivas) se distribuye de este modo a través de cada una de las memorias de procesador 401-402 y las memorias de GPU 420 a 423, lo que permite que cualquier procesador o GPU acceda a cualquier memoria física con una dirección virtual correlacionada con esa memoria.

En una realización, la circuitería de gestión de desvío/coherencia 494A a 494E dentro de una o varias de las MMU 439A a 439E garantiza la coherencia de la memoria caché entre las memorias caché de los procesadores anfitriones (por ejemplo, 405) y las GPU 410 a 413 e implementa técnicas de desvío que indican las memorias físicas en las que deben almacenarse determinados tipos de datos. Si bien en la **figura 4F** se muestran múltiples instancias de la circuitería de gestión de desvío/coherencia 494A a 494E, la circuitería de desvío/coherencia se puede implementar dentro de la MMU de uno o varios procesadores anfitriones 405 y/o dentro del circuito de integración del acelerador 436.

Una realización permite que la memoria conectada a la GPU 420 a 423 se correlacione como parte de la memoria del sistema y se acceda a ella utilizando tecnología de memoria virtual compartida (SVM), pero sin sufrir los típicos inconvenientes de rendimiento asociados con la coherencia total de la memoria caché del sistema. La capacidad para acceder a la memoria anexada a GPU 420 a 423 como memoria de sistema sin una onerosa sobrecarga de coherencia de la memoria caché proporciona un entorno de operación beneficioso para la descarga de GPU. Esta disposición permite que el software del procesador anfitrión 405 establezca operandos y acceda a resultados de cálculo, sin la sobrecarga de las copias de datos de DMA de E/S tradicionales. Estas copias tradicionales implican llamadas a controladores, interrupciones y accesos de E/S correlacionados a memoria (MMIO) que son todos ineficientes en relación con los accesos simples a la memoria. Al mismo tiempo, la capacidad de acceder a la memoria conectada a la GPU 420 a 423 sin sobrecargas de coherencia de memoria caché puede ser fundamental para el tiempo de ejecución de un cálculo descargado. En caso de tráfico de memoria sustancial de escritura de transmisión en continuo, por ejemplo, la sobrecarga de coherencia de memoria caché puede reducir significativamente el ancho de banda de escritura efectivo visto por una GPU 410 a 413. La eficacia de la configuración de los operandos, la eficacia del acceso

a los resultados y la eficiencia del cálculo de la GPU juegan un papel en la determinación de la efectividad de la descarga de la GPU.

En una implementación, la selección entre el desvío de la GPU y el desvío del procesador anfitrión está impulsada por una estructura de datos de seguimiento de desvío. Se puede usar una tabla de desvíos que puede ser, por ejemplo, una estructura granular de página (es decir, controlada en la granularidad de una página de memoria) que incluye 1 o 2 bits por página de memoria conectada a GPU. La tabla de desvíos puede implementarse en un intervalo de memoria robado de una o varias memorias conectadas a GPU 420 a 423, con o sin una memoria caché de desvío en la GPU 410 a 413 (por ejemplo, para almacenar en memoria caché entradas usadas de manera frecuente/reciente de la tabla de desvíos). Como alternativa, toda la tabla de desvíos puede mantenerse dentro de la GPU.

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

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

Un mecanismo para cambiar el estado de desvío emplea una llamada API (por ejemplo, OpenCL), que, a su vez, llama al controlador de dispositivo de la GPU que, a su vez, envía un mensaje (o pone en cola un descriptor de comando) a la GPU indicándole que cambie el estado de desvío y, para algunas transiciones, realizar una operación de vaciado de memoria caché en el anfitrión. Se requiere la operación de vaciado de memoria caché para una transición desde el procesador anfitrión 405 a un desvío de GPU, pero no se requiere para la transición opuesta.

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

Canalización de procesamiento de gráficos

La **figura 5** ilustra una canalización de procesamiento de gráficos 500, de acuerdo con una realización. En una realización, un procesador de gráficos puede implementar la canalización de procesamiento de gráficos 500 ilustrada. El procesador de gráficos puede incluirse dentro de los subsistemas de procesamiento en paralelo como se describe en el presente documento, tal como el procesador en paralelo 200 de la figura 2, que, en una realización, es una variante del procesador o procesadores en paralelo 112 de la figura 1. Los diversos sistemas de procesamiento en paralelo pueden implementar la canalización de procesamiento de gráficos 500 a través de una o varias instancias de la unidad de procesamiento en paralelo (por ejemplo, la unidad de procesamiento en paralelo 202 de la figura 2) como se describe en el presente documento. Por ejemplo, una unidad de sombreado (por ejemplo, el multiprocesador de gráficos 234 de la figura 3) puede estar configurada para realizar las funciones de una o varias de 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, los ensambladores de primitivas 506, 514, 518, la unidad de teselado 510, el rasterizador 522 y la unidad de operaciones de rasterización 526 también las pueden realizar otros motores de procesamiento dentro de un agrupamiento de procesamiento (por ejemplo, el agrupamiento de procesamiento 214 de la figura 3) y una correspondiente unidad de partición (por ejemplo, la unidad de partición 220A a 220N de la figura 2). La canalización de procesamiento de gráficos 500 también se puede implementar usando unidades de procesamiento dedicadas para una o varias funciones. En una realización, una o varias porciones de la canalización de procesamiento de gráficos 500 se pueden realizar mediante lógica de procesamiento en paralelo dentro de un procesador de propósito general (por ejemplo, CPU). En una realización, una o varias porciones de la canalización de procesamiento de gráficos 500 pueden acceder a una memoria en chip (por ejemplo, la memoria de procesador en paralelo 222 como en la figura 2) a través de una interfaz de memoria 528, que puede ser una instancia de la interfaz de memoria 218 de la figura 2.

En una realización, el ensamblador de datos 502 es una unidad de procesamiento que recopila datos de vértices para superficies y primitivas. El ensamblador de datos 502 emite entonces los datos de vértice, incluyendo los atributos de

vértice, a la unidad de procesamiento de vértices 504. La unidad de procesamiento de vértices 504 es una unidad de ejecución programable que ejecuta programas de sombreado de vértices, iluminando y transformando datos de vértices según lo especificado por los programas de sombreado de vértices. La unidad de procesamiento de vértices 504 lee datos que se almacenan en memoria caché, local o de sistema para su uso en el procesamiento de los datos de vértices y puede programarse para transformar los datos de vértices desde una representación de coordenadas basada en objetos hasta un espacio de coordenadas de espacio mundial o un espacio de coordenadas de dispositivo normalizado.

Una primera instancia de un ensamblador de primitivas 506 recibe atributos de vértices desde la unidad de procesamiento de vértices 50. El ensamblador de primitivas 506 lee atributos de vértices almacenados según sea necesario y construye primitivas de gráficos para su procesamiento por la unidad de procesamiento de control de teselado 508. Las primitivas de gráficos incluyen triángulos, segmentos de línea, puntos, parches, etc., según lo admitido por varias interfaces de programación de aplicaciones (API) de procesamiento de gráficos.

La unidad de procesamiento de control de teselado 508 trata los vértices de entrada como puntos de control para un parche geométrico. Los puntos de control se transforman de una representación de entrada del parche (por ejemplo, las bases del parche) a una representación que es adecuada para su uso en la evaluación de superficie por 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 solo 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 primitivas geométricas tales como primitivas 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 opera en coordenadas parametrizadas del parche subdividido para generar una representación de superficie y atributos de vértice para cada vértice asociado con las primitivas geométricas.

Una segunda instancia de un ensamblador de primitivas 514 recibe atributos de vértices de la unidad de procesamiento de evaluación de teselado 512, lee atributos de vértice almacenados según sea necesario y construye primitivas de gráficos para su procesamiento por parte de la unidad de procesamiento de geometría 516. La unidad de procesamiento de geometría 516 es una unidad de ejecución programable que ejecuta programas de sombreado de geometría para transformar primitivas de gráficos recibidas del ensamblador de primitivas 514 según lo especificado por los programas de sombreado de geometría. En una realización, la unidad de procesamiento de geometría 516 está programada para subdividir las primitivas de gráficos en una o varias primitivas de gráficos nuevas y calcular parámetros usados para rasterizar las nuevas primitivas de gráficos.

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

El rasterizador 522 puede realizar selección de profundidad y otras optimizaciones basadas en profundidad. El rasterizador 522 también realiza conversión de escaneo en las nuevas primitivas de gráficos para generar fragmentos y enviar esos fragmentos y datos de cobertura asociados a la unidad de procesamiento de fragmentos/píxeles 524. La unidad de procesamiento de fragmentos/píxeles 524 es una unidad de ejecución programable que está configurada para ejecutar programas de sombreado de fragmentos o programas de sombreado de píxeles. La unidad de procesamiento de fragmentos/píxeles 524 transforma fragmentos o píxeles recibidos del rasterizador 522, según lo especificado por los programas de sombreado de fragmentos o píxeles. Por ejemplo, la unidad de procesamiento de fragmentos/píxeles 524 puede programarse para realizar operaciones que incluyen, de forma no limitativa, correlación de textura, sombreado, mezcla, corrección de textura y corrección de perspectiva para producir fragmentos o píxeles sombreados que se emiten a una unidad de operaciones de rasterización 526. La unidad de procesamiento de fragmentos/píxeles 524 puede leer datos que están almacenados en la memoria del procesador en paralelo o en la memoria del sistema para su uso cuando se procesan los datos de fragmentos. Los programas de sombreado de fragmentos o píxeles se pueden configurar para sombrear muestras, píxeles, teselas u otras granularidades dependiendo de la velocidad de muestreo configurada para las unidades de procesamiento.

La unidad de operaciones de rasterización 526 es una unidad de procesamiento que realiza operaciones de rasterización incluyendo, de forma no limitativa, estarcido, prueba z, mezcla, y similares, y emite datos de píxel como datos de gráficos procesados para ser almacenados en memoria de gráficos (por ejemplo, memoria de procesador en paralelo 222 como en la figura 2, y/o memoria de sistema 104 como en la figura 1, para ser mostrados en el uno o varios dispositivos de visualización 110 o para procesamiento posterior por uno del uno o varios procesadores 102 o procesadores en paralelos 112. En algunas realizaciones, la unidad de operaciones de rasterización 526 está

configurada para comprimir datos z o de color que se escriben en memoria y descomprimir datos z o de color que se leen desde la memoria.

5 Como se describió anteriormente, muchos procesadores modernos, por ejemplo, unidades de procesamiento de gráficos (GPU), admiten cargas de trabajo muy variables dependiendo de diversos factores, como la cantidad y la naturaleza de las aplicaciones que se ejecutan en el procesador o procesadores, los procesos en segundo plano, las características del usuario y similares.

10 La materia objeto aquí descrita aborda estas y otras cuestiones proporcionando diversas técnicas para la gestión de la energía del procesador para buses de comunicación. En una primera técnica de ejemplo descrita con referencia a las figuras 6A y 6B, la lógica que opera en un controlador permite la gestión de energía del procesador basándose en los atributos de un usuario, los hábitos de carga de trabajo de un usuario y/o el perfil de consumo de energía. Se puede establecer un nivel de consumo de energía de un procesador utilizando información en el perfil de usuario.

15 Haciendo referencia a las figuras 6A y 6B, en la operación 610 la lógica que opera en un controlador recopila información del usuario. En algunos ejemplos, la lógica puede recopilar estadísticas basadas en la máquina asociadas con el usuario, tales como niveles de carga de trabajo consumidos por el usuario, consumo de energía del procesador consumido por el usuario, o similares. Alternativamente, o además, la lógica puede recopilar información personal asociada con el usuario, tal como la edad aproximada del usuario, su nivel de visión y similares. Alternativamente, o
20 además, la lógica puede recopilar preferencias del usuario tales como aplicaciones utilizadas habitualmente, niveles de brillo de la pantalla y similares.

En la operación 615 la información del usuario puede ser utilizada para construir un perfil de usuario. Refiriéndose a la figura 6B, uno o varios perfiles de usuario 630, 632, 636 pueden ser generados y almacenados en una memoria. En
25 algunos ejemplos, el perfil de usuario puede basarse en uno o varios patrones de prueba de diagnóstico con la entrada del usuario para seleccionar una preferencia. Cuando un usuario inicia sesión en un sistema de procesamiento de datos, el perfil del usuario puede recuperarse de la memoria y actualizarse durante la sesión del usuario.

Opcionalmente, en la operación 620 la lógica puede categorizar a un usuario basándose al menos en parte en el
30 contenido de un perfil de usuario. Por ejemplo, un usuario que no ejecuta rutinariamente aplicaciones de uso intensivo de gráficos, tales como juegos tridimensionales (3D) de escenarios complejos de uso de medios, puede ser categorizado como un Usuario Ligero. Un usuario que utilice ocasionalmente juegos 3D u otros medios con uso intensivo de gráficos puede ser categorizado como Usuario Medio. Un usuario que sea creador de juegos o contenidos o que haga un uso habitual de juegos en 3D u otros medios con uso intensivo de gráficos puede clasificarse como
35 Usuario Intensivo.

En la operación 625, la lógica puede establecer un perfil de potencia de un procesador, por ejemplo, un procesador de gráficos, basándose al menos en parte en el contenido del perfil y/o la categorización del usuario, solo o en combinación con la retroalimentación de un algoritmo de aprendizaje automático. Tal como se ilustra en la figura 13,
40 el contenido del perfil de usuario puede ser introducido en la lógica que se ejecuta en un procesador, por ejemplo, una unidad de procesamiento de gráficos (GPU) 640, que puede ejecutar la lógica para establecer los niveles de potencia para una o varias GPU 640. Alternativamente, la lógica puede residir en un microcontrolador acoplado a la una o varias GPU 640.

45 Por ejemplo, la lógica puede designar un perfil de ahorro de energía para un usuario ligero limitando el empuje de frecuencia o la habilitación de agrupamientos de cálculo o agrupamientos de cálculo de ciclo en el tiempo de vida de una o varias GPU para proporcionar un uso prolongado de la vida. Para un usuario medio, la lógica puede habilitar más capacidades y puede ser más sensible a los escenarios justo a tiempo (JIT). En el caso de usuarios intensivos, la lógica puede llevar a una o varias GPU directamente a un modo de alto rendimiento y reducir o eliminar los esfuerzos
50 por prolongar la vida útil de la batería.

Tal como se ilustra en la figura 6B, en algunos ejemplos las estadísticas de usuario pueden ser pasadas a un controlador de modo kernel 642.

55 En un ejemplo descrito con referencia a las figuras 6C y 6D, la lógica que opera en un controlador recopila métricas de rendimiento para varias etapas de una canalización de procesamiento de gráficos y ajusta al menos una de un voltaje o una frecuencia de las etapas de canalización de procesamiento de gráficos utilizando las métricas de rendimiento.

60 En referencia a la figura 6C, en la operación 650, la lógica recopila métricas de rendimiento de la canalización. Haciendo referencia a la figura 6D, dependiendo de las características de la carga de trabajo, algunas partes de una canalización de gráficos pueden ser más activas que otras partes. Por ejemplo, una carga de trabajo de cálculo puede mantener ocupada la unidad o unidades de ejecución 664 donde se ejecutan las operaciones matemáticas, mientras que los puertos de datos 666 y las unidades de memoria caché 666, 668, pueden estar menos ocupadas. Por el
65 contrario, una carga de trabajo que implemente un gran número de instrucciones de memoria (por ejemplo, golpeando

la memoria caché L2 668) mantendrá el puerto de datos 666 y la memoria caché L2 668 activos, pero la unidad o unidades de ejecución 662 estarán menos activas ya que están esperando los datos de memoria.

De forma similar, una carga de trabajo en la que las operaciones de memoria son intensas pero los accesos a la memoria no llegan a las memorias caché de gráficos y van a la memoria del sistema resultará en un bloque de gráficos que está mayormente inactivo y esperando memoria ya que la memoria del sistema habitualmente tiene una latencia larga. En este caso solo el controlador de memoria de sistema 670 estará ocupado.

En la operación 655, el controlador está configurado para ajustar el voltaje y/o la frecuencia de las etapas de la canalización utilizando las métricas de rendimiento recopiladas en la operación 650. La lógica está configurada para aumentar un voltaje y/o frecuencia operativos en etapas de canalización que están ocupadas, y reducir un voltaje y/o frecuencia operativos en etapas de canalización que no están ocupadas.

Los videojuegos dependen de una versión de un controlador de gráficos, ya sea que operen en un entorno independiente o en un entorno de red. Una versión determinada de un controlador de gráficos puede implementar algunas heurísticas basadas, por ejemplo, en llamadas a la API y/o en estados de canalización. En función de esto, existe una configuración de bucle abierto para compilar sombreadores y programar registros específicos del hardware con el fin de ofrecer un rendimiento optimizado. Algunos proveedores de hardware independientes (IHV) implementan una configuración local de bucle cerrado basada en la creación de perfiles de hardware mediante heurísticas. Estos algoritmos no tienen en cuenta varios factores relacionados con los contadores de rendimiento y la correlación de otros registros de hardware programables y las elecciones del compilador. Por tanto, el espacio de estados no se explora en la máquina local y, además, puede que no haya suficiente capacidad de cálculo en una máquina de juego típica para explorar este espacio de estados con el fin de ofrecer una experiencia óptima.

En un ejemplo descrito con referencia a la figura 7A, los usuarios finales 710 pueden ofrecer voluntariamente información relacionada con una aplicación, por ejemplo, un juego, que se ejecuta en un dispositivo de procesamiento. En algunos ejemplos, el conjunto de información puede incluir información relativa a la unidad de procesamiento de gráficos ((GPU), por ejemplo, las etapas de silicio, memoria, capacidad de refrigeración, y otros parámetros relacionados con OEM), la versión del controlador, compilador y la función *hash* de configuración de registro de hardware (que describió un punto, las opciones de compilación del sombreador del espacio de estado y los valores de registro de hardware), la temperatura de la GPU, frecuencia de funcionamiento, voltaje, la CPU anfitrión, velocidad de la memoria y el tamaño y la carga en la máquina. Aparte de esta información básica, se exponen contadores de rendimiento específicos del hardware para recopilar estadísticas sobre la utilización de varios bloques y los cuellos de botella.

Algunos equipos tienen contadores de rendimiento que pueden ser consultados periódicamente por un servidor 715. Se pueden crear conjuntos de entrenamiento basados en los puntos de datos y el perfil por GPU y configuración de sistema en base a cada juego. Los conjuntos de entrenamiento pueden utilizarse para entrenar una red neuronal artificial (ANN) y/o una red neuronal celular (CNN) para obtener la máxima velocidad de tramas o una velocidad de tramas sostenida a una determinada temperatura o potencia. En algunos ejemplos, la optimización explora el espacio de estados y elige un punto de estado que define la programación de hardware de los registros y los compiladores de sombreador para controlar el rendimiento.

Cada combinación de programación de estados y opciones de compilador define una función *hash*. Para elegir la función *hash* pueden utilizarse algoritmos de descenso de gradiente o de clasificación. De este modo, el ajuste por juego y por sistema del proveedor de hardware independiente (IHV) puede convertirse en una experiencia personalizada.

La descripción y los dibujos anteriores deben considerarse en un sentido ilustrativo más que restrictivo. Los expertos en la materia comprenderán que pueden realizarse diversas modificaciones y cambios en las realizaciones descritas en el presente documento sin apartarse del alcance de la invención, tal como se establece en las reivindicaciones adjuntas.

En otro ejemplo, puede implementarse una red neuronal para configurar una canalización de procesamiento. Las figuras 7B y 7C son diagramas de flujo que ilustran operaciones en un procedimiento para la gestión de energía del procesador, de acuerdo con las realizaciones. Refiriéndose a la figura 7B, en un ejemplo una aplicación puede ser ejecutada repetidamente durante una sesión de entrenamiento para la canalización de procesamiento bajo parámetros operativos variables y configuraciones de hardware variables (operación 730). Por ejemplo, refiriéndose a la figura 6D, durante la sesión de entrenamiento la aplicación puede ser ejecutada a diferentes velocidades de tramas y a diferentes configuraciones para la memoria caché L1 666 y/o la memoria caché L2 668, o una memoria caché de nivel superior, por ejemplo, la memoria caché L3. Además, durante la sesión de entrenamiento, la aplicación puede ser ejecutada a diferentes voltajes y/o frecuencias para las unidades de ejecución 664.

En la operación 735, los datos de rendimiento de la aplicación recopilados en la operación 730 se utilizan para construir un perfil de rendimiento para la aplicación en la canalización de procesamiento. En la operación 740, los datos del perfil de rendimiento se utilizan para entrenar una red neuronal.

En tiempo de ejecución, la red neuronal configurada en la operación 740 puede ser utilizada para configurar la canalización de procesamiento para ejecutar la aplicación. Refiriéndose a la figura 7C, en la operación 760 una nueva instancia de la aplicación puede ser iniciada en tiempo de ejecución. En la operación 765 la red neuronal entrenada en la operación 730 puede ser usada para configurar la canalización de procesamiento para ejecutar la aplicación.

Componentes de energía

La **figura 8** ilustra un diagrama de bloques de un regulador de conmutación de acuerdo con una realización. Uno o más reguladores de conmutación que se muestran en la figura 8 pueden incorporarse en diversos sistemas analizados en el presente documento para proporcionar energía a uno o varios microprocesadores de circuito integrado (IC). Mientras que una sola fase del regulador de conmutación de estacionamiento de corriente con un solo inductor puede analizarse con referencia a la figura 8, una o varias de las múltiples fases del regulador de conmutación de estacionamiento de corriente pueden implementarse con un inductor dividido. Además, se puede usar una combinación de uno o varios reguladores de conmutación de estacionamiento de corriente (con o sin un inductor dividido) con uno o varios dispositivos convencionales de conversión de energía eléctrica para proporcionar energía a la carga (por ejemplo, la circuitería lógica 814).

Más particularmente, la figura 8 ilustra un sistema 800 que incluye un regulador de conmutación (a veces denominado regulador de conmutación de estacionamiento de corriente). El regulador de conmutación de estacionamiento de corriente puede ser un regulador de conmutación multifase en diversas realizaciones. La unidad de control multifase 802 está acoplada a múltiples fases, donde cada fase puede incluir una o varias fases aguas arriba 804 y una o varias fases aguas abajo 806. Como se muestra, una fuente de energía eléctrica 808 está acoplada a la lógica de control aguas arriba 810 (que proporciona mecanismos de control de corriente en cada fase aguas arriba). Se puede usar más de una lógica de control aguas arriba en diversas implementaciones. Cada fase aguas arriba puede incluir un inductor (no mostrado) que está acoplado a una fase aguas abajo respectiva. En una realización, cada una de las fases aguas arriba puede incluir uno o varios inductores. La unidad de control multifase 802 puede configurar cualquier lógica de control aguas arriba activa 810, por ejemplo, para generar una corriente a través de un inductor acoplado entre las fases aguas arriba y las fases aguas abajo. La lógica de control aguas abajo 812 puede ser configurada por la unidad de control multifase 802 para estar ENCENDIDA, APAGADA o conmutar para regular el nivel de voltaje en la carga (por ejemplo, la circuitería lógica 814). A su vez, la lógica de control aguas abajo 812 puede ser configurada por la unidad de control multifase 802 para mantener el nivel de voltaje en la carga dentro de un intervalo basado al menos en parte en los valores $V_{\min}$ (voltaje mínimo) y $V_{\max}$ (voltaje máximo).

En una realización, un inductor (acoplado entre una fase aguas abajo y una fase aguas arriba respectiva) puede situarse fuera de un paquete de semiconductores 816 que incluye la carga 814. Se puede situar otro inductor (no mostrado) dentro del paquete 816, por ejemplo, para reducir la capacitancia parásita. En una realización, el inductor dentro del paquete 816 puede ser un inductor de núcleo de aire plano que está acoplado a la circuitería lógica 814 por medio de una o varias lógicas de conmutación que incluyen transistores de efecto de campo de semiconductores de óxido metálico (MOSFET) planos. Además, uno o varios de los componentes analizados en el presente documento (por ejemplo, con referencia a las figuras 8, 9 y/o 10, que incluyen, por ejemplo, la memoria caché L3, lógica de control aguas arriba y/o lógica de control aguas abajo) se pueden proporcionar en la capa o capas del sustrato (por ejemplo, entre paquetes de semiconductores), en una matriz de circuito integrado, o fuera de un paquete de semiconductores (por ejemplo, en una placa de circuito impreso (PCB)) en diversas realizaciones.

La **figura 9** es un diagrama de bloques de un sistema 900 que incluye un multiprocesador de transmisión en continuo 902, de acuerdo con una o varias realizaciones. El multiprocesador de transmisión en continuo puede incluir 32 carriles 904 de múltiples subprocesos de instrucción única (SIMT) que son capaces de emitir colectivamente hasta 32 instrucciones por ciclo de reloj, por ejemplo, uno a partir de cada uno de 32 subprocesos. Dependiendo de la implementación, pueden estar presentes más o menos carriles, tales como 64, 128, 256, etc. Los carriles SIMT 904 pueden, a su vez, incluir una o más: unidades aritméticas lógicas (ALU) 906, unidades de funciones especiales (SFU) 908, unidades de memoria (MEM) 910 y/o unidades de textura (TEX) 912.

En algunas realizaciones, una o varias ALU 906 y/o unidad o unidades TEX 912 pueden ser de baja energía o alta capacidad, por ejemplo, tal como se describe con referencia a los elementos 920 y 922. Por ejemplo, el sistema puede correlacionar el 100 % de las direcciones de registro para los subprocesos 0 a 30 a la porción de baja energía y el 100 % de las direcciones de registro para los subprocesos 31 a 127 a la porción de alta capacidad. Como otro ejemplo, el sistema puede correlacionar el 20 % de los registros de cada subproceso a la porción de baja energía y correlacionar el 80 % de los registros de cada subproceso a la porción de alta capacidad. Además, el sistema puede determinar el número de entradas asignadas por subproceso basándose en la información de tiempo de ejecución.

Tal como se ilustra en la figura 9, el multiprocesador de transmisión en continuo 902 también incluye un archivo de registro 914, una lógica de planificador 916 (por ejemplo, para planificar subprocesos o grupos de subprocesos, o ambos) y memoria compartida 918, por ejemplo, almacenamiento temporal local. Como se describe en el presente documento, un "grupo de subprocesos" se refiere a una pluralidad de subprocesos que se agrupan con índices de subprocesos ordenados (por ejemplo, secuenciales o consecutivos). En general, un archivo de registro se refiere a

una matriz de registros a los que acceden los componentes de un procesador (incluyendo un procesador de gráficos) como los que se analizan en el presente documento. El archivo de registro 914 incluye una porción o estructura de baja energía 920 y una porción o estructura de alta capacidad 922. El multiprocesador de transmisión en continuo 902 puede configurarse para direccionar el archivo de registro 914 usando un único espacio de nombres lógico tanto para la porción de baja energía como para la porción de alta capacidad.

En algunas realizaciones, el sistema puede incluir varios registros físicos que pueden ser compartidos por los subprocesos que se ejecutan simultáneamente en el sistema. Esto permite que el sistema use un solo espacio de nombres para implementar un esquema de correlación de registro flexible. A continuación, un compilador puede asignar intervalos activos de registros a direcciones de registros, y el compilador puede usar un mecanismo de asignación de registros para minimizar o reducir la cantidad de registros usados por subproceso. Se pueden asignar múltiples intervalos activos a la misma dirección de registro siempre que los intervalos activos no se superpongan en una realización. Esto permite determinar, por ejemplo, en tiempo de ejecución y después de que se hayan compilado las instrucciones, cuántas entradas por subproceso se asignarán en la porción de baja energía frente a la porción de alta capacidad. Por ejemplo, el sistema puede correlacionar el 100 % de las direcciones de registro para los subprocesos 0 a 30 a la porción de baja energía y el 100 % de las direcciones de registro para los subprocesos 31 a 127 a la porción de alta capacidad. Como otro ejemplo, el sistema puede correlacionar el 20 % de los registros de cada subproceso a la porción de baja energía y correlacionar el 80 % de los registros de cada subproceso a la porción de alta capacidad. El sistema puede determinar la cantidad de entradas asignadas por subproceso basándose en la información de tiempo de ejecución, por ejemplo, con respecto a la cantidad de grupos de subprocesos que se ejecutan y el beneficio marginal de iniciar más grupos de subprocesos o asignar a un número menor de grupos de subprocesos más espacio en la porción de baja energía.

La **figura 10** ilustra un diagrama de bloques de un sistema de procesamiento en paralelo 1000, de acuerdo con una realización. El Sistema 1000 incluye un subsistema de procesamiento en paralelo 1002 (presentado previamente) que a su vez incluye una o varias unidades de procesamiento en paralelo (PPU) PPU-0 a PPU-P. Cada PPU está acoplada a una memoria de procesamiento en paralelo (PP) local (por ejemplo, Mem-0 a MEM-P, respectivamente). En algunas realizaciones, el sistema de subsistema de PP 1002 puede incluir un número P de PPU. La PPU-0 1004 y las memorias de procesamiento en paralelo 1006 pueden implementarse usando uno o varios dispositivos de circuito integrado, tales como procesadores programables, circuitos integrados específicos de aplicación (ASIC) o dispositivos de memoria.

Haciendo referencia a la figura 10, se muestran varios conmutadores o conexiones opcionales 1007 que pueden usarse en el sistema 1000 para gestionar la energía. Si bien se muestran varios conmutadores 1007, las realizaciones no se limitan a los conmutadores mostrados específicamente y se pueden utilizar más o menos conmutadores dependiendo de la implementación. Estas conexiones/conmutadores 1007 se pueden utilizar para activación de reloj o activación de energía general. Por lo tanto, los elementos 1007 pueden incluir uno o varios de un transistor de energía, un conmutador integrado en el circuito integrado, conexiones del plano de energía o similares. En una realización, antes de cortar la alimentación a una porción del sistema 1000 por medio de los conmutadores/conexiones 1007, la lógica (por ejemplo, un microcontrolador, un procesador de señales digitales, firmware, etc.) puede garantizar que los resultados de la operación se comprometan (por ejemplo, a la memoria) o se finalicen para mantener la corrección.

Además, en algunas realizaciones, una o varias de las PPU en el subsistema de procesamiento en paralelo 1002 son procesadores de gráficos con canalizaciones de renderización que pueden configurarse para realizar diversas tareas, tales como las que se analizan en el presente documento con respecto a otras figuras. Los datos/información de gráficos pueden comunicarse a través del puente de memoria 1008 con otros componentes de un sistema informático (incluidos los componentes del sistema 1000). Los datos pueden comunicarse a través de un bus compartido y/o una o varias interconexiones 1010 (incluyendo, por ejemplo, uno o varios enlaces directos o punto a punto). La PPU-0 1004 puede acceder a su memoria de procesamiento en paralelo local 1014 (que puede utilizarse como memoria de gráficos, incluyendo, por ejemplo, una memoria intermedia de trama) para almacenar y actualizar datos de píxeles, entregar datos de píxeles a un dispositivo de visualización (como los que se tratan en el presente documento), etc. En algunas realizaciones, el subsistema de procesamiento en paralelo 1002 puede incluir una o varias PPU que operan como procesadores de gráficos y una o varias PPU que operan para realizar cálculos de propósito general. Las PPU pueden ser idénticas o diferentes, y cada PPU puede tener acceso a su propio dispositivo o dispositivos de memoria de procesamiento en paralelo dedicados, ningún dispositivo o dispositivos de memoria de procesamiento en paralelo dedicados, o un dispositivo de memoria compartida o memoria caché.

En una realización, las operaciones realizadas por las PPU pueden ser controladas por otro procesador (o una de las PPU) generalmente denominado procesador maestro o núcleo de procesador. En una realización, el procesador/núcleo maestro puede escribir un flujo de comandos para cada PPU en una memoria intermedia de inserción en varias ubicaciones, tal como una memoria de sistema principal, una memoria caché u otra memoria, tales como las que se describen en el presente documento con referencia a otras figuras. A continuación, cada PPU puede leer los comandos escritos y ejecutarlos de forma asíncrona en relación con la operación del procesador/núcleo maestro.

Además, tal como se muestra en la figura 10, la PPU-0 incluye una lógica de extremo frontal 1020 que puede incluir una unidad de entrada/salida (E/S o ES) (por ejemplo, para comunicarse con otros componentes del sistema 1000 a través del puente de memoria 1008) y/o una interfaz de anfitrión (por ejemplo, que recibe comandos relacionados con tareas de procesamiento). El extremo frontal 1020 puede recibir comandos leídos por la interfaz de anfitrión (por ejemplo, desde la memoria intermedia de inserción)). El extremo frontal 1020 a su vez proporciona los comandos a una unidad de planificación de trabajo 1022 que planifica y asigna operación(es)/tarea(s) asociada(s) con los comandos a una matriz de agrupamientos de procesamiento o subsistema aritmético 1024 para su ejecución.

Tal como se muestra en la figura 10, la matriz de agrupamientos de procesamiento 1024 puede incluir una o varias unidades de agrupamiento de procesamiento general (GPC) (por ejemplo, GPC-0 1026, GPC-1 1028, hasta GPC-M 1030). Cada GPC puede ser capaz de ejecutar una gran cantidad (por ejemplo, cientos o miles) de subprocesos al mismo tiempo, donde cada subproceso es una instancia de un programa. En diversas aplicaciones, pueden asignarse diferentes GPC para procesar diferentes tipos de programas o para realizar diferentes tipos de cálculos. Por ejemplo, en una aplicación de gráficos, se puede asignar un primer conjunto de GPC (por ejemplo, que incluya una o varias unidades de GPC) para realizar operaciones de teselado y producir topologías de primitivas para parches, y un segundo conjunto de GPC (por ejemplo, que incluya una o varias unidades de GPC) pueden asignarse para realizar sombreado de teselado para evaluar los parámetros de parche para las topologías de primitivas y para determinar las posiciones de los vértices y otros atributos por vértice. La asignación de GPC puede variar dependiendo de la carga de trabajo que surja para cada tipo de programa o cálculo.

Adicionalmente, las tareas de procesamiento asignadas por la unidad de planificación de trabajo 1022 pueden incluir índices de datos que van a procesarse, tales como datos de superficie/parche, datos de primitivas, datos de vértices, datos de píxeles y/o parámetros de estado y comandos que definen cómo han de procesarse los datos (por ejemplo, qué programa ha de ejecutarse). La unidad de planificación de trabajo 1022 puede estar configurada para extraer los índices que corresponden a las tareas o puede recibir los índices desde el extremo frontal 1020. El extremo frontal 1020 también puede garantizar que las GPC estén configuradas en un estado válido antes de que se inicie el procesamiento especificado por las memorias intermedias de inserción.

En una realización, la ruta de comunicación 1012 es un enlace exprés (o PCI-e) de interfaz de componentes periféricos (PCI), en el que se pueden asignar carriles dedicados a cada PPU. También se pueden usar otras rutas de comunicación. Por ejemplo, los comandos relacionados con las tareas de procesamiento pueden dirigirse a la interfaz de anfitrión 1018, mientras que los comandos relacionados con las operaciones de memoria (por ejemplo, leer de, o escribir en la memoria de procesamiento en paralelo 1014) pueden dirigirse a una unidad de barra transversal de memoria 1032.

En algunas realizaciones, el subsistema de procesamiento en paralelo 1002 puede implementarse como una tarjeta adicional que se inserta en una ranura de expansión del sistema informático o servidor (tal como un servidor Blade). En otras realizaciones, una PPU puede integrarse en un solo chip con un puente de bus, tal como un puente de memoria 1008, un puente de E/S, etc. Aún en otras realizaciones, algunos o todos los componentes de la PPU pueden integrarse en un solo chip de circuito integrado con uno o varios núcleos de procesador, dispositivos de memoria, memorias caché, etc.

Además, uno de los principales problemas de los procesadores de los módems actuales es que han alcanzado un límite de velocidad de reloj de alrededor de 4 GHz. En este punto, simplemente generan demasiado calor para la tecnología actual y requieren soluciones de enfriamiento especiales y costosas. Esto se debe a que a medida que aumentamos la frecuencia del reloj, aumenta el consumo de energía. De hecho, el consumo de energía de una CPU, si fijas el voltaje, es aproximadamente el cubo de su frecuencia de reloj. Para empeorar esto, a medida que aumenta el calor generado por la CPU, para la misma frecuencia de reloj, el consumo de energía también aumenta debido a las propiedades del silicio. Esta conversión de energía en calor es una completa pérdida de energía. Este uso cada vez más ineficiente de la energía finalmente significa que no puede alimentar o enfriar el procesador lo suficiente y alcanza los límites térmicos del dispositivo o su carcasa, el llamado muro de energía.

Ante la imposibilidad de aumentar la velocidad del reloj y fabricar procesadores cada vez más rápidos, los fabricantes de procesadores tuvieron que idear otro plan de juego. Se han visto obligados a seguir el camino de añadir más núcleos a los procesadores, en lugar de intentar continuamente aumentar las velocidades de reloj de la CPU y/o extraer más instrucciones por reloj a través del paralelismo de nivel de instrucciones.

Además, el uso de energía es una gran consideración cuando se diseñan máquinas que funcionan constantemente. A menudo, los costes operativos de hacer funcionar un superordenador durante unos pocos años pueden equivaler al coste de instalarla en primer lugar. Ciertamente, el coste de hacer funcionar una máquina de este tipo durante su vida útil superará fácilmente los costes de instalación originales. El uso de energía proviene de los componentes mismos, pero también del enfriamiento necesario para permitir que tales ordenadores funcionen. Incluso una estación de trabajo de gama alta con cuatro GPU requiere cierta planificación sobre cómo mantenerla fresca. A menos que el usuario viva en un clima frío y pueda desterrar el ordenador a un lugar frío, hará un buen trabajo calentándole la oficina. Coloque varias de estas máquinas en una habitación y, muy rápidamente, la temperatura del aire en esa habitación comenzará a elevarse a niveles bastante inaceptables.

Por lo tanto, se gasta una cantidad significativa de energía en la instalación de sistemas de aire acondicionado para garantizar que los ordenadores permanezcan fríos y puedan funcionar sin producir errores. Esto es especialmente así donde las temperaturas de verano pueden alcanzar los 85 °F/30 °C o más. El aire acondicionado es caro de mantener. Se debe pensar mucho en la mejor manera de enfriar un sistema de este tipo y si la energía térmica se puede reutilizar de alguna manera. Los sistemas enfriados por líquido son muy eficientes de esta manera, ya que el líquido puede circular a través de un intercambiador de calor y hacia un sistema de calefacción convencional sin ninguna posibilidad de que los dos líquidos se mezclen. Con los costes cada vez mayores de los recursos naturales y las crecientes presiones sobre las compañías para que se las considere ecológicas, simplemente sacar el calor por la ventana ya no es económica o socialmente aceptable.

Los sistemas refrigerados por líquido ofrecen una opción interesante en términos de reciclaje de la energía térmica residual. Si bien un sistema enfriado por aire solo se puede usar para calentar el área inmediata en la que se encuentra, el calor de los refrigerantes líquidos se puede bombear a otra parte. Mediante el uso de un intercambiador de calor, el refrigerante se puede enfriar con agua convencional. A continuación, este puede bombearse a un sistema de calefacción o incluso usarse para calentar una piscina al aire libre u otra gran masa de agua. Cuando se instalan varios de estos sistemas, tal como en un centro informático de una empresa o una universidad, puede tener mucho sentido usar esta energía térmica residual para reducir la factura de calefacción en otras partes de la organización.

Muchas instalaciones de superordenadores se ubican junto a un río importante precisamente porque necesitan un suministro de agua fría. Otras usan grandes torres de refrigeración para disipar la energía térmica residual. Ninguna solución es particularmente ecológica. Habiendo pagado ya por la energía, tiene poco sentido simplemente desperdiciarla cuando podría usarse tan fácilmente para calefacción. Al considerar el uso de energía, también se debe recordar que el diseño del programa en realidad juega un papel muy importante en el consumo de energía. La operación más costosa, desde el punto de vista de la energía, es mover datos dentro y fuera del chip. Por lo tanto, simplemente hacer un uso eficiente de los registros y la memoria compartida dentro del dispositivo reduce enormemente el uso de energía. Si también considera que el tiempo total de ejecución de los programas bien escritos es mucho menor que el de los mal escritos, puede ver que reescribir los programas antiguos para hacer uso de nuevas funciones, tales como una memoria compartida más grande, puede incluso reducir los costes operativos en un gran centro de datos.

Haciendo referencia a la figura 10, la interfaz de memoria 1014 incluye N unidades de partición (por ejemplo, Unidad-0 1034, Unidad-1 1036, Unidad-N 1038) que están directamente acopladas, cada una, a una porción correspondiente de la memoria de procesamiento en paralelo 1006 (tal como Mem-0 1040, Mem-1 1042, hasta Mem-N 1044). La cantidad de unidades de partición generalmente puede ser igual a la cantidad de memoria presentada previamente (o N, como se muestra). La memoria presentada anteriormente puede implementarse con una memoria volátil tal como la memoria dinámica de acceso aleatorio (DRAM) u otros tipos de memoria volátil como las que se analizan en el presente documento. En otras realizaciones, el número de unidades de partición puede no igualar el número de dispositivos de memoria. Los datos gráficos (tales como objetivos de renderización, memorias intermedias de trama o correlaciones de textura) pueden almacenarse a través de dispositivos de memoria presentados previamente, permitiendo que las unidades de partición escriban porciones de datos de gráficos en paralelo para utilizar eficientemente el ancho de banda disponible de la memoria de procesamiento en paralelo 1006.

Además, una cualquiera de las GPC puede procesar datos para escribirlos en cualquiera de las unidades de partición dentro de la memoria de procesamiento en paralelo. La unidad de barra transversal 1032 puede implementarse como una interconexión que está configurada para encaminar la salida de cada GPC a la entrada de cualquier unidad de partición o a otra GPC para su posterior procesamiento. Por lo tanto, las GPC 1026 a 1030 pueden comunicarse con la interfaz de memoria 1014 a través de la unidad de barra transversal 1032 para leer de, o escribir en diversos otros dispositivos de memoria (o externos). Tal como se muestra, la unidad de barra transversal 1032 puede comunicarse directamente con el extremo frontal 1020, además de tener un acoplamiento (directo o indirecto) a la memoria local 1006, para permitir que los núcleos de procesamiento dentro de las diferentes GPC se comuniquen con la memoria de sistema y/u otra memoria que no es local para PPU. Además, la unidad de barra transversal 1032 puede utilizar canales virtuales para organizar flujos de tráfico entre las GPC y las unidades de partición.

Visión general del sistema

La **figura 11** es un diagrama de bloques de un sistema de procesamiento 1100, de acuerdo con una realización. En diversas realizaciones, el sistema 1100 incluye uno o varios procesadores 1102 y uno o varios procesadores de gráficos 1108, y puede ser un sistema de sobremesa de procesador único, un sistema de estación de trabajo de multiprocesador o un sistema de servidor que tiene un gran número de procesadores 1102 o núcleos de procesador 1107. En una realización, el sistema 1100 es una plataforma de procesamiento incorporada dentro de un circuito integrado de sistema en chip (SoC) para su uso en dispositivos móviles, portátiles o integrados.

Una realización del sistema 1100 puede incluir, o incorporarse dentro de una plataforma de juegos basada en servidor, una consola de juegos, incluyendo una consola de juegos y de medios, una consola de juegos móvil, una consola de juegos portátil o una consola de juegos en línea. En algunas realizaciones, el sistema 1100 es un teléfono móvil, un

teléfono inteligente, un dispositivo informático de tipo tableta o un dispositivo de Internet móvil. El sistema de procesamiento de datos 1100 también puede incluir, acoplarse o integrarse dentro de un dispositivo ponible, tal como un dispositivo ponible de reloj inteligente, un dispositivo de gafas inteligentes, un dispositivo de realidad aumentada o un dispositivo de realidad virtual. En algunas realizaciones, el sistema de procesamiento de datos 1100 es un dispositivo de televisión o de descodificador que tiene uno o varios procesadores 1102 y una interfaz gráfica generada por uno o varios procesadores de gráficos 1108.

En algunas realizaciones, el uno o varios procesadores 1102 incluyen, cada uno, uno o varios núcleos de procesador 1107 para procesar instrucciones que, cuando se ejecutan, realizan operaciones para el software del sistema y del usuario. En algunas realizaciones, cada uno del uno o varios núcleos de procesador 1107 está configurado para procesar un conjunto de instrucciones específico 1109. En algunas realizaciones, el conjunto de instrucciones 1109 puede facilitar el cálculo de conjunto de instrucciones complejo (CISC), el cálculo de conjunto de instrucciones reducido (RISC) o el cálculo mediante una palabra de instrucción muy larga (VLIW). Cada uno de múltiples núcleos de procesador 1107 puede procesar un conjunto de instrucciones diferente 1109, que puede incluir instrucciones para facilitar la emulación de otros conjuntos de instrucciones. El núcleo del procesador 1107 también puede incluir otros dispositivos de procesamiento, tales como un procesador de señales digitales (DSP).

En algunas realizaciones, el procesador 1102 incluye memoria caché 1104. Dependiendo de la arquitectura, el procesador 1102 puede tener una única memoria caché interna o múltiples niveles de memoria caché interna. En algunas realizaciones, la memoria caché se comparte entre varios componentes del procesador 1102. En algunas realizaciones, el procesador 1102 también utiliza una memoria caché externa (por ejemplo, una memoria caché de nivel 3 (L3) o una memoria caché de último nivel (LLC)) (no mostrada), que se puede compartir entre los núcleos 1107 del procesador utilizando técnicas conocidas de coherencia de memoria caché. Se incluye adicionalmente un archivo de registro 1106 en el procesador 1102 que puede incluir diferentes tipos de registros para almacenar diferentes tipos de datos (por ejemplo, registros de números enteros, registros de coma flotante, registros de estado y un registro de puntero de instrucción). Algunos registros pueden ser registros de propósito general, mientras que otros registros pueden ser específicos del diseño del procesador 1102.

En algunas realizaciones, el procesador 1102 está acoplado con un bus de procesador 1110 para transmitir señales de comunicación, tales como dirección, datos o señales de control entre el procesador 1102 y otros componentes en el sistema 1100. En una realización, el sistema 1100 utiliza una arquitectura de sistema de 'concentrador' a modo de ejemplo, que incluye un concentrador de controlador de memoria 1116 y un concentrador de controlador de entrada y salida (E/S) 1130. Un concentrador de controlador de memoria 1116 facilita la comunicación entre un dispositivo de memoria y otros componentes del sistema 1100, mientras que un concentrador de controlador de E/S (ICH) 1130 proporciona conexiones a dispositivos de E/S mediante un bus de E/S local. En una realización, la lógica del concentrador de controlador de memoria 1116 está integrada dentro del procesador.

El dispositivo de memoria 1120 puede ser un dispositivo de memoria de acceso aleatorio dinámica (DRAM), un dispositivo de memoria de acceso aleatorio estática (SRAM), un dispositivo de memoria flash, un dispositivo de memoria de cambio de fase o algún otro dispositivo de memoria que tenga un rendimiento adecuado para servir como memoria de proceso. En una realización, el dispositivo de memoria 1120 puede funcionar como una memoria de sistema para el sistema 1100, para almacenar los datos 1122 y las instrucciones 1121 para su uso cuando el uno o varios procesadores 1102 ejecutan una aplicación o proceso. El concentrador de controlador de memoria 1116 también se acopla con un procesador de gráficos externo opcional 1112, que puede comunicarse con el uno o varios procesadores de gráficos 1108 en los procesadores 1102 para realizar operaciones de gráficos y de medios.

En algunas realizaciones, el ICH 1130 posibilita que los periféricos se conecten al dispositivo de memoria 1120 y al procesador 1102 mediante un bus de E/S de alta velocidad. Los periféricos de E/S incluyen, de forma no limitativa, un controlador de audio 1146, una interfaz de firmware 1128, un transceptor inalámbrico 1126 (por ejemplo, Wi-Fi, Bluetooth), un dispositivo de almacenamiento de datos 1124 (por ejemplo, unidad de disco duro, memoria flash, etc.), y un controlador de E/S heredado 1140 para acoplar dispositivos heredados (por ejemplo, de sistema personal 2 (PS/2)) al sistema. Uno o varios controladores 1142 de bus serie universal (USB) conectan dispositivos de entrada, tales como combinaciones de teclado y ratón 1144. Un controlador de red 1134 también puede acoplarse con el ICH 1130. En algunas realizaciones, un controlador de red de alto rendimiento (no mostrado) se acopla con el bus de procesador 1110. Se apreciará que el sistema 1100 mostrado es a modo de ejemplo y no limitativo, como otros tipos de sistemas de procesamiento de datos que estén configurados de manera diferente. Por ejemplo, el concentrador de controlador de E/S 1130 puede integrarse dentro del uno o varios procesadores 1102, o el concentrador de controlador de memoria 1116 y el concentrador de controlador de E/S 1130 pueden integrarse en un procesador de gráficos externo discreto, tal como el procesador de gráficos externo 1112.

La **figura 12** es un diagrama de bloques de una realización de un procesador 1200 que tiene uno o varios núcleos de procesador 1202A a 1202N, un controlador de memoria integrado 1214 y un procesador de gráficos integrado 1208. Aquellos elementos de la **figura 12** que tienen los mismos números (o nombres) de referencia que los elementos de cualquier otra figura en el presente documento pueden operar o funcionar de cualquier manera similar a la descrita en cualquier otra parte en el presente documento, pero sin limitación a esto. El procesador 1200 puede incluir núcleos adicionales hasta e incluyendo el núcleo adicional 1202N representado por los recuadros en líneas discontinuas. Cada

uno de los núcleos de procesador 1202A a 1202N incluye una o varias unidades de memoria caché internas 1204A a 1204N. En algunas realizaciones, cada núcleo de procesador también tiene acceso a una o varias unidades en memoria caché compartidas 1206.

Las unidades de memoria caché internas 1204A a 1204N y las unidades de memoria caché compartidas 1206 representan una jerarquía de memoria caché dentro del procesador 1200. La jerarquía de memoria caché puede incluir al menos un nivel de memoria caché de instrucciones y de datos dentro de cada núcleo de procesador y uno o varios niveles de memoria caché de nivel medio compartida, tal como Nivel 2 (L2), Nivel 3 (L3), Nivel 4 (L4), u otros niveles de memoria caché, donde el nivel más alto de memoria caché antes de la memoria externa se clasifica como LLC. En algunas realizaciones, la lógica de coherencia de memoria caché mantiene la coherencia entre las diversas unidades de caché 1206 y 1204A a 1204N.

En algunas realizaciones, el procesador 1200 también puede incluir un conjunto de una o varias unidades de controlador de bus 1216 y un núcleo de agente del sistema 1210. La una o varias unidades de controlador de bus 1216 gestionan un conjunto de buses periféricos, tales como uno o varios buses de interconexión de componentes periféricos (por ejemplo, PCI, PCI Express). El núcleo de agente de sistema 1210 proporciona funcionalidad de gestión para los diversos componentes de procesador. En algunas realizaciones, el núcleo de agente del sistema 1210 incluye uno o varios controladores de memoria integrados 1214 para gestionar el acceso a varios dispositivos de memoria externos (no mostrados).

En algunas realizaciones, uno o varios de los núcleos de procesador 1202A a 1202N incluyen soporte para múltiples subprocesos simultáneos. En una realización de este tipo, el núcleo de agente de sistema 1210 incluye componentes para coordinar y operar los núcleos 1202A a 1202N durante el procesamiento de múltiples subprocesos. El núcleo de agente del sistema 1210 puede incluir adicionalmente una unidad de control de energía (PCU), que incluye lógica y componentes para regular el estado de energía de los núcleos de procesador 1202A a 1202N y el procesador de gráficos 1208.

En algunas realizaciones, el procesador 1200 incluye adicionalmente el procesador de gráficos 1208 para ejecutar las operaciones de procesamiento de gráficos. En algunas realizaciones, el procesador de gráficos 1208 se acopla con el conjunto de unidades de caché compartidas 1206 y el núcleo de agente del sistema 1210, incluyendo uno o varios controladores de memoria integrados 1214. En algunas realizaciones, un controlador de visualización 1211 está acoplado al procesador de gráficos 1208 para controlar la salida del procesador de gráficos hacia uno o varios dispositivos de visualización acoplados. En algunas realizaciones, el controlador de visualización 1211 puede ser un módulo separado acoplado con el procesador de gráficos a través de al menos una interconexión, o puede estar integrado dentro del procesador de gráficos 1208 o el núcleo de agente de sistema 1210.

En algunas realizaciones, se usa una unidad de interconexión basada en anillo 1212 para acoplar los componentes internos del procesador 1200. Sin embargo, se puede usar una unidad de interconexión alternativa, tal como una interconexión punto a punto, una interconexión conmutada u otras técnicas, incluyendo técnicas bien conocidas en la técnica. En algunas realizaciones, el procesador de gráficos 1208 se acopla con la interconexión de anillo 1212 a través de un enlace de E/S 1213.

El enlace de E/S 1213 a modo de ejemplo representa al menos una de múltiples variedades de interconexiones de E/S, que incluyen una interconexión de E/S de paquete, que facilita la comunicación entre diversos componentes de procesador y un módulo de memoria integrado de alto rendimiento 1218, tal como un módulo de eDRAM. En algunas realizaciones, cada uno de los núcleos de procesador 1202A a 1202N y del procesador de gráficos 1208 usan módulos de memoria integrados 1218 como una memoria caché de último nivel compartida.

En algunas realizaciones, los núcleos de procesador 1202A a 1202N son núcleos homogéneos que ejecutan la misma arquitectura de conjunto de instrucciones. En otra realización, los núcleos de procesador 1202A a 1202N son heterogéneos en términos de arquitectura de conjunto de instrucciones (ISA), donde uno o varios de los núcleos de procesador 1202A a 1202N ejecutan un primer conjunto de instrucciones, mientras que al menos uno de los otros núcleos ejecuta un subconjunto del primer conjunto de instrucciones o un conjunto de instrucciones diferente. En una realización, los núcleos de procesador 1202A a 1202N son heterogéneos en términos de microarquitectura, donde uno o varios núcleos que tienen un consumo de energía relativamente mayor se acoplan con uno o varios núcleos de energía que tienen un consumo de energía menor. Adicionalmente, el procesador 1200 se puede implementar en uno o varios chips o como un circuito integrado de SoC que tiene los componentes ilustrados, además de otros componentes.

La **figura 13** es un diagrama de bloques de un procesador de gráficos 1300, que puede ser una unidad de procesamiento de gráficos discreta, o puede ser un procesador de gráficos integrado con una pluralidad de núcleos de procesamiento. En algunas realizaciones, el procesador de gráficos se comunica, mediante una interfaz de E/S correlacionada con la memoria, con registros en el procesador de gráficos y con comandos colocados en la memoria de procesador. En algunas realizaciones, el procesador de gráficos 1300 incluye una interfaz de memoria 1314 para acceder a la memoria. La interfaz de memoria 1314 puede ser una interfaz para la memoria local, una o varias memorias caché internas, una o varias memorias caché externas compartidas y/o la memoria del sistema.

En algunas realizaciones, el procesador de gráficos 1300 también incluye un controlador de visualización 1302 para enviar datos de salida de visualización a un dispositivo de visualización 1320. El controlador de visualización 1302 incluye hardware para uno o varios planos de superposición para la visualización y composición de múltiples capas de elementos de interfaz de usuario o de vídeo. En algunas realizaciones, el procesador de gráficos 1300 incluye un motor de códec de vídeo 1306 para codificar, decodificar o transcodificar medios hacia, desde o entre uno o varios formatos de codificación de medios, incluidos, entre otros, formatos del Grupo de Expertos en Imágenes en Movimiento (MPEG), tales como MPEG-2, formatos de codificación de vídeo avanzada (AVC) tales como H.264/MPEG-4 AVC, así como formatos de la Sociedad de Ingenieros de Televisión e Imágenes en Movimiento (SMPTE) 421M/VC-1 y del Grupo de Expertos Fotográficos Conjuntos (JPEG) tales como JPEG y formatos Motion JPEG (MJPEG).

En algunas realizaciones, el procesador de gráficos 1300 incluye un motor de transferencia de imágenes en bloques (BLIT) 1304 para realizar operaciones de rasterización bidimensionales (2D) que incluyen, por ejemplo, transferencias de bloques con límites de bits. Sin embargo, en una realización, las operaciones de gráficos 2D se realizan usando uno o varios componentes del motor de procesamiento de gráficos (GPE) 1310. En algunas realizaciones, el GPE 1310 es un motor informático para realizar operaciones de gráficos, incluyendo operaciones de gráficos tridimensionales (3D) y operaciones de medios.

En algunas realizaciones, el GPE 310 incluye una canalización 3D 1312 para llevar a cabo operaciones 3D, tales como por ejemplo renderizar imágenes y escenas tridimensionales utilizando funciones de procesamiento que actúan sobre formas de primitivas 3D (por ejemplo, rectángulo, triángulo, etc.). La canalización 3D 1312 incluye elementos de función programable y fija que llevan a cabo diversas tareas dentro del elemento y/o generan subprocesos de ejecución en un subsistema de 3D/medios 1315. Si bien la canalización 3D 1312 se puede usar para realizar operaciones de medios, una realización de GPE 1310 también incluye una canalización de medios 1316 que se usa específicamente para realizar operaciones de medios, tales como postprocesamiento de vídeo y mejora de imágenes.

En algunas realizaciones, la canalización de medios 1316 incluye funciones fijas o unidades lógicas programables para realizar una o varias operaciones de medios especializadas, tales como aceleración de decodificación de vídeo, desentrelazado de vídeo y aceleración de codificación de vídeo en lugar de, o en nombre del motor de códec de vídeo 1306. En algunas realizaciones, la canalización de medios 1316 incluye adicionalmente una unidad de generación de subprocesos para generar subprocesos para su ejecución en el subsistema 3D/medios 1315. Los subprocesos generados realizan cálculos para las operaciones de medios en una o varias unidades de ejecución de gráficos incluidas en el subsistema 3D/medios 1315.

En algunas realizaciones, el subsistema 3D/medios 1315 incluye una lógica para ejecutar subprocesos generados por la canalización 3D 1312 y la canalización de medios 1316. En una realización, las canalizaciones envían solicitudes de ejecución de subprocesos al subsistema 3D/medios 1315, que incluye lógica de despacho de subproceso para arbitrar y despachar las diversas solicitudes a recursos de ejecución de subproceso disponibles. Los recursos de ejecución incluyen una matriz de unidades de ejecución de gráficos para procesar los subprocesos 3D y de medios. En algunas realizaciones, el subsistema 3D/medios 1315 incluye una o varias memorias caché internas para instrucciones y datos de subproceso. En algunas realizaciones, el subsistema también incluye memoria compartida, que incluye registros y memoria direccionable, para compartir datos entre subprocesos y para almacenar datos de salida.

Motor de procesamiento de gráficos

La **figura 14** es un diagrama de bloques de un motor de procesamiento de gráficos 1410 de un procesador de gráficos de acuerdo con algunas realizaciones. En una realización, el motor de procesamiento de gráficos (GPE) 1410 es una versión del GPE 1310 mostrado en la **figura 13**. Los elementos de la **figura 14** que tienen los mismos números de referencia (o nombres) que los elementos de cualquier otra figura en el presente documento pueden operar o funcionar de cualquier manera similar a la descrita en cualquier otra parte del presente documento, pero no se limitan a tal cosa. Por ejemplo, se ilustra la canalización de 3D 1312 y la canalización de medios 1316 de la **figura 13**. La canalización de medios 1316 es opcional en algunas realizaciones del GPE 1410 y puede no incluirse explícitamente dentro del GPE 1410. Por ejemplo, y en al menos una realización, un procesador de medios y/o imágenes separado está acoplado al GPE 1410.

En algunas realizaciones, el GPE 1410 se acopla con o incluye un transmisor en continuo de comandos 1403, que proporciona un flujo de comandos a la canalización 3D 1312 y/o a las canalizaciones de medios 1316. En algunas realizaciones, el transmisor en continuo de comandos 1403 está acoplado con memoria, que puede ser memoria de sistema, o una o varias de memoria de caché interna y memoria de caché compartida. En algunas realizaciones, el transmisor en continuo de comandos 1403 recibe comandos de la memoria y envía los comandos a la canalización 3D 1312 y/o a la canalización de medios 1316. Los comandos son directivas obtenidas de una memoria intermedia en anillo, que almacena comandos para la canalización 3D 1312 y la canalización de medios 1316. En una realización, la memoria intermedia en anillo puede incluir adicionalmente memorias intermedias de comandos por lotes que almacenan lotes de múltiples comandos. Los comandos para la canalización 3D 1312 también pueden incluir referencias a datos almacenados en la memoria, tales como, entre otros, datos de vértices y geometría para la

canalización 3D 1312 y/o datos de imagen y objetos de memoria para la canalización de medios 1316. La canalización de 3D 1312 y la canalización de medios 1316 procesan los comandos y datos realizando operaciones mediante una lógica dentro de las canalizaciones respectivas o despachando uno o varios subprocesos de ejecución a una matriz de núcleo de gráficos 1414.

En diversas realizaciones, la canalización de 3D 1312 puede ejecutar uno o varios programas sombreadores, tales como sombreadores de vértices, sombreadores de geometría, sombreadores de píxeles, sombreadores de fragmentos, sombreadores de cálculo u otros programas sombreadores, procesando las instrucciones y despachando subprocesos de ejecución a la matriz de núcleo de gráficos 1414. La matriz de núcleo de gráficos 1414 proporciona un bloque unificado de recursos de ejecución. La lógica de ejecución multipropósito (por ejemplo, unidades de ejecución) dentro de la matriz de núcleos de gráficos 1414 incluye soporte para varios lenguajes de sombreado API 3D y puede ejecutar múltiples subprocesos de ejecución simultáneos asociados con múltiples sombreadores.

En algunas realizaciones, la matriz de núcleo de gráficos 1414 también incluye una lógica de ejecución para realizar funciones de medios, tales como procesamiento de vídeo y/o de imagen. En una realización, las unidades de ejecución incluyen adicionalmente lógica de propósito general que es programable para realizar operaciones de cálculo paralelas de propósito general, además de operaciones de procesamiento de gráficos. La lógica de propósito general puede realizar operaciones de procesamiento en paralelo o en conjunto con la lógica de propósito general dentro del núcleo o núcleos de procesador 107 de la **figura 1** o el núcleo 1202A a 1202N como en la **figura 12**.

Los datos de salida generados por los subprocesos que se ejecutan en la matriz de núcleos de gráficos 1414 pueden enviar datos a la memoria en una memoria intermedia de retorno unificada (URB) 1418. La URB 1418 puede almacenar datos para múltiples subprocesos. En algunas realizaciones la URB 1418 puede usarse para enviar datos entre diferentes subprocesos que se ejecutan en la matriz de núcleo de gráficos 1414. En algunas realizaciones, la URB 1418 se puede usar adicionalmente para la sincronización entre subprocesos en la matriz de núcleos de gráficos y la lógica de función fija dentro de la lógica de función compartida 1420.

En algunas realizaciones, la matriz de núcleos de gráficos 1414 es escalable, de tal forma que la matriz incluye un número variable de núcleos de gráficos, cada uno de los cuales tiene un número variable de unidades de ejecución basado en el nivel de potencia y rendimiento objetivo del GPE 1410. En una realización, los recursos de ejecución son dinámicamente escalables, de tal forma que los recursos de ejecución pueden habilitarse o deshabilitarse según sea necesario.

La matriz de núcleos de gráficos 1414 está acoplada a lógica de función compartida 1420 que incluye múltiples recursos que se comparten entre los núcleos de gráficos de la matriz de núcleos de gráficos. Las funciones compartidas dentro de la lógica de función compartida 1420 son unidades lógicas de hardware que proporcionan funcionalidad suplementaria especializada al conjunto de núcleos de gráficos 1414. En diversas realizaciones, la lógica de función compartida 1420 incluye, de forma no limitativa, la lógica del muestreador 1421, del cálculo matemático 1422 y de la comunicación entre subprocesos (ITC) 1423. Además, algunas realizaciones implementan una o varias memorias caché 1425 dentro de la lógica de función compartida 1420. Una función compartida se implementa cuando la demanda de una función especializada dada es insuficiente para su inclusión en la matriz de núcleos de gráficos 1414. En su lugar, se implementa una única instanciación de dicha función especializada como una entidad independiente en la lógica de función compartida 1420 y se comparte entre los recursos de ejecución dentro de la matriz de núcleo de gráficos 1414. El conjunto preciso de funciones que se comparten entre la matriz de núcleos de gráficos 1414 y se incluyen dentro de la matriz de núcleos de gráficos 1414 varía entre las realizaciones.

La **figura 15** es un diagrama de bloques de otra realización de un procesador de gráficos 1500. Los elementos de la **figura 15** que tienen los mismos números de referencia (o nombres) que los elementos de cualquier otra figura en el presente documento pueden operar o funcionar de cualquier manera similar a la descrita en cualquier otra parte del presente documento, pero no se limitan a tal cosa.

En algunas realizaciones, el procesador de gráficos 1500 incluye una interconexión en anillo 1502, un extremo frontal de canalización 1504, un motor de medios 1537 y unos núcleos de gráficos 1580A-1580N. En algunas realizaciones, la interconexión en anillo 1502 acopla el procesador de gráficos a otras unidades de procesamiento, incluyendo otros procesadores de gráficos o uno o varios núcleos de procesador de propósito general. En algunas realizaciones, el procesador de gráficos es uno de muchos procesadores integrados dentro de un sistema de procesamiento de múltiples núcleos.

En algunas realizaciones, el procesador de gráficos 1500 recibe lotes de comandos a través de la interconexión en anillo 1502. Los comandos de entrada se interpretan por un transmisor en continuo de comandos 1503 en el extremo frontal de canalización 1504. En algunas realizaciones, el procesador de gráficos 1500 incluye una lógica de ejecución escalable para realizar procesamiento de geometría 3D y procesamiento de medios a través del núcleo o núcleos de gráficos 1580A a 1580N. Para los comandos de procesamiento de geometría 3D, el transmisor en continuo de comandos 1503 suministra comandos a la canalización de geometría 1536. Para al menos algunos comandos de procesamiento de medios, el transmisor en continuo de comandos 1503 suministra los comandos a un extremo frontal de vídeo 1534, que se acopla con un motor de medios 1537. En algunas realizaciones, el motor de medios 1537

incluye un motor de calidad de vídeo (VQE) 1530 para el postprocesamiento de vídeo e imágenes y un motor de codificación/decodificación multiformato (MFX) 1533 para proporcionar codificación y decodificación de datos de medios acelerada por hardware. En algunas realizaciones, la canalización de geometría 1536 y el motor de medios 1537 generan cada uno subprocesos de ejecución para los recursos de ejecución de subprocesos proporcionados por al menos un núcleo de gráficos 1580A.

En algunas realizaciones, el procesador de gráficos 1500 incluye recursos de ejecución de subprocesos escalable que cuentan con los núcleos modulares 1580A a 1580N (denominados, en ocasiones, segmentos de núcleo), teniendo cada uno múltiples subnúcleos 1550A a 1550N, 1560A a 1560N (denominados, en ocasiones, subsegmentos de núcleo). En algunas realizaciones, el procesador de gráficos 1500 puede tener cualquier número de núcleos de gráficos de 1580A a 1580N. En algunas realizaciones, el procesador de gráficos 1500 incluye un núcleo de gráficos 1580A que tiene al menos un primer subnúcleo 1550A y un segundo subnúcleo 1560A. En otras realizaciones, el procesador de gráficos es un procesador de baja potencia con un único subnúcleo (por ejemplo, 1550A). En algunas realizaciones, el procesador de gráficos 1500 incluye múltiples núcleos de gráficos 1580A a 1580N, incluyendo cada uno un conjunto de primeros subnúcleos 1550A a 1550N y un conjunto de segundos subnúcleos 1560A a 1560N. Cada subnúcleo del conjunto de primeros subnúcleos 1550A a 1550N incluye al menos un primer conjunto de unidades de ejecución 1552A a 1552N y muestreadores de medios/texturas 1554A a 1554N. Cada subnúcleo del conjunto de segundos subnúcleos 1560A a 1560N incluye al menos un segundo conjunto de unidades de ejecución 1562A a 1562N y muestreadores 1564A a 1564N. En algunas realizaciones, cada subnúcleo 1550A a 1550N, 1560A a 1560N comparte un conjunto de recursos compartidos 1570A a 1570N. En algunas realizaciones, los recursos compartidos incluyen memoria caché compartida y lógica de operación de píxeles. También pueden incluirse otros recursos compartidos en las diversas realizaciones del procesador de gráficos.

Unidades de ejecución

La **figura 16** ilustra la lógica de ejecución de subprocesos 1600 que incluye una matriz de elementos de procesamiento empleados en algunas realizaciones de un GPE. Los elementos de la **figura 16** que tienen los mismos números de referencia (o nombres) que los elementos de cualquier otra figura en el presente documento pueden operar o funcionar de cualquier manera similar a la descrita en cualquier otra parte del presente documento, pero no se limitan a tal cosa.

En algunas realizaciones, la lógica de ejecución de subprocesos 1600 incluye un procesador de sombreado 1602, un despachador de subprocesos 1604, una memoria caché de instrucciones 1606, una matriz de unidades de ejecución escalable que incluye una pluralidad de unidades de ejecución 1608A a 1608N, un muestreador 1610, una memoria caché de datos 1612 y un puerto de datos 1614. En una realización, la matriz de unidades de ejecución escalable puede realizar un escalado dinámico habilitando o deshabilitando una o varias unidades de ejecución (por ejemplo, cualquiera de las unidades de ejecución 1608A, 1608B, 1608C, 1608D a 1608N-1 y 1608N) basándose en los requisitos de cálculo de una carga de trabajo. En una realización, los componentes incluidos están interconectados mediante una estructura de interconexión que está enlazada a cada uno de los componentes. En algunas realizaciones, la lógica de ejecución de subprocesos 1600 incluye una o varias conexiones a memoria, tales como una memoria de sistema o memoria caché, a través de una o varias memorias caché de instrucciones 1606, el puerto de datos 1614, el muestreador 1610 y las unidades de ejecución 1608A a 1608N. En algunas realizaciones, cada unidad de ejecución (por ejemplo 1608A) es una unidad de cálculo de propósito general programable autónoma que es capaz de ejecutar múltiples subprocesos de hardware simultáneos mientras procesa múltiples elementos de datos en paralelo para cada subproceso. En diversas realizaciones, la matriz de unidades de ejecución 1608A a 1608N es escalable para incluir cualquier número de unidades de ejecución individuales.

En algunas realizaciones, las unidades de ejecución 1608A a 1608N se usan principalmente para ejecutar programas de sombreado. Un procesador de sombreado 1602 puede procesar los diversos programas de sombreado y despachar subprocesos de ejecución asociados con los programas de sombreado a través de un despachador de subprocesos 1604. En una realización, el despachador de subprocesos incluye lógica para arbitrar solicitudes de iniciación de subprocesos desde las canalizaciones de gráficos y medios e instanciar los subprocesos solicitados en una o varias unidades de ejecución en las unidades de ejecución 1608A a 1608N. Por ejemplo, la canalización de geometría (por ejemplo, 1536 de la **figura 15**) puede despachar sombreadores de vértices, de teselado o de geometría a la lógica de ejecución de subprocesos 1600 (la **figura 16**) para su procesamiento. En algunas realizaciones, el despachador de subprocesos 1604 también puede procesar solicitudes de generación de subprocesos en tiempo de ejecución desde los programas sombreadores en ejecución.

En algunas realizaciones, las unidades de ejecución 1608A a 1608N admiten un conjunto de instrucciones que incluye un soporte nativo para muchas instrucciones de sombreado de gráficos 3D convencionales, de modo que los programas de sombreado desde bibliotecas de gráficos (por ejemplo, Direct 3D y OpenGL) se ejecutan con una conversión mínima. Las unidades de ejecución admiten procesamiento de vértices y geometría (por ejemplo, programas de vértices, programas de geometría, sombreadores de vértices), procesamiento de píxeles (por ejemplo, sombreadores de píxeles, sombreadores de fragmentos) y procesamiento de propósito general (por ejemplo, sombreadores de medios y cálculo). Cada una de las unidades de ejecución 1608A a 1608N es capaz de una ejecución de múltiples datos de instrucción única (SIMD) de múltiple emisión y la operación con múltiples subprocesos permite un entorno de ejecución eficaz frente a accesos a memoria de mayor latencia. Cada subproceso de hardware dentro

de cada unidad de ejecución tiene un archivo de registro de alto ancho de banda dedicado y un estado de subproceso independiente asociado. La ejecución es de múltiples emisiones por reloj a canalizaciones que pueden realizar operaciones de números enteros, de coma flotante de precisión sencilla y doble, de capacidad de ramificación de SIMD, operaciones lógicas, operaciones trascendentales y otras operaciones misceláneas. Mientras se esperan datos de la memoria o de una de las funciones compartidas, la lógica de dependencia dentro de las unidades de ejecución 1608A a 1608N hace que un subproceso de espera esté inactivo hasta que los datos solicitados hayan sido devueltos. Mientras el subproceso en espera está inactivo, pueden dedicarse recursos de hardware a procesar otros subprocesos. Por ejemplo, durante un retardo asociado con una operación de sombreado de vértices, una unidad de ejecución puede realizar operaciones para un sombreador de píxeles, un sombreador de fragmentos u otro tipo de programa de sombreado, incluyendo un sombreador de vértices diferente.

Cada unidad de ejecución en las unidades de ejecución 1608A a 1608N opera sobre matrices de elementos de datos. El número de elementos de datos es el "tamaño de ejecución", o el número de canales para la instrucción. Un canal de ejecución es una unidad lógica de ejecución para el acceso a elementos de datos, enmascaramiento y control de flujo dentro de las instrucciones. El número de canales puede ser independiente del número de unidades aritméticas lógicas (ALU) o unidades de coma flotante (FPU) de un procesador de gráficos en particular. En algunas realizaciones, las unidades de ejecución 1608A a 1608N admiten tipos de datos de números enteros y de coma flotante.

El conjunto de instrucciones de la unidad de ejecución incluye instrucciones SIMD. Los diversos elementos de datos se pueden almacenar como un tipo de datos encapsulado en un registro y la unidad de ejecución procesará los diversos elementos en función del tamaño de los datos de los elementos. Por ejemplo, cuando se realizan operaciones con un vector de 256 bits de ancho, los 256 bits del vector se almacenan en un registro y la unidad de ejecución realiza operaciones con el vector como cuatro elementos de datos empaquetados de 64 bits separados (elementos de datos de tamaño de palabra cuádruple (QW)), ocho elementos de datos empaquetados de 32 bits separados (elementos de datos de tamaño de palabra doble (DW)), dieciséis elementos de datos empaquetados de 16 bits separados (elementos de datos de tamaño de palabra (W)) o treinta y dos elementos de datos de 8 bits separados (elementos de datos de tamaño de octeto (B)). Sin embargo, son posibles diferentes anchuras de vector y tamaños de registro.

Una o varias memorias caché de instrucciones internas (por ejemplo, 1606) se incluyen en la lógica de ejecución de subprocesos 1600 para almacenar en memoria caché instrucciones de subprocesos para las unidades de ejecución. En algunas realizaciones, se incluyen una o varias memorias caché de datos (por ejemplo, 1612) para almacenar en memoria caché datos de subproceso durante la ejecución del subproceso. En algunas realizaciones, se incluye un muestreador 1610 para proporcionar un muestreo de textura para operaciones 3D y muestreo de medios para operaciones de medios. En algunas realizaciones, el muestreador 1610 incluye funcionalidad especializada de muestreo de textura o de medios para procesar datos de textura o de medios durante el proceso de muestreo antes de proporcionar los datos muestreados a una unidad de ejecución.

Durante la ejecución, las canalizaciones de gráficos y medios envían solicitudes de iniciación de subprocesos a la lógica de ejecución de subprocesos 1600 a través de la lógica de generación y despacho de subprocesos. Una vez que se ha procesado y rasterizado un grupo de objetos geométricos para obtener datos de píxel, se invoca una lógica de procesador de píxeles (por ejemplo, lógica de sombreado de píxeles, lógica de sombreador de fragmentos, etc.) dentro del procesador de sombreado 1602 para calcular adicionalmente información de salida y hacer que se escriban resultados para emitir superficies (por ejemplo, memorias intermedias de color, memorias intermedias de profundidad, memorias intermedias de estarcido, etc.). En algunas realizaciones, un sombreador de píxeles o un sombreador de fragmentos calcula los valores de los diversos atributos de vértice que se van a interpolar a lo largo del objeto rasterizado. En algunas realizaciones, la lógica del procesador de píxeles dentro del procesador de sombreado 1602 ejecuta un programa de sombreado de píxeles o fragmentos suministrado por una interfaz de programación de aplicaciones (API). Para ejecutar el programa de sombreado, el procesador de sombreado 1602 despacha subprocesos a una unidad de ejecución (por ejemplo, 1608A) mediante el despachador de subprocesos 1604. En algunas realizaciones, el sombreador de píxeles 1602 usa una lógica de muestreo de textura en el muestreador 1610 para acceder a datos de textura en correlaciones de textura almacenadas en memoria. Las operaciones aritméticas en los datos de textura y los datos de geometría de entrada calculan los datos de color de píxel para cada fragmento geométrico, o descartan uno o varios píxeles del procesamiento posterior.

En algunas realizaciones, el puerto de datos 1614 proporciona un mecanismo de acceso a memoria para que la lógica de ejecución de subprocesos 1600 proporcione datos procesados a la memoria para su procesamiento en una canalización de salida de procesador de gráficos. En algunas realizaciones, el puerto de datos 1614 incluye o se acopla a una o varias memorias caché (por ejemplo, la memoria caché de datos 1612) para almacenar en memoria caché datos para un acceso de memoria mediante el puerto de datos.

La **figura 17** es un diagrama de bloques que ilustra unos formatos de instrucción de procesador de gráficos 1700 de acuerdo con algunas realizaciones. En una o varias realizaciones, las unidades de ejecución del procesador de gráficos admiten un conjunto de instrucciones que tiene instrucciones en múltiples formatos. Los recuadros de líneas continuas ilustran los componentes que se incluyen generalmente en una instrucción de unidad de ejecución, mientras que las líneas discontinuas incluyen componentes que son opcionales o que sólo se incluyen en un subconjunto de las instrucciones. En algunas realizaciones, el formato de instrucción 1700 descrito e ilustrado son macroinstrucciones,

en el sentido de que son instrucciones suministradas a la unidad de ejecución, en contraposición a microoperaciones resultantes de la decodificación de la instrucción una vez que la instrucción es procesada.

En algunas realizaciones, las unidades de ejecución del procesador de gráficos admiten de forma nativa instrucciones en un formato de instrucción de 128 bits 1710. Un formato de instrucción compactado de 64 bits 1730 está disponible para algunas instrucciones basándose en la instrucción, las opciones de instrucción y el número de operandos seleccionados. El formato de instrucción de 128 bits nativo 710 proporciona acceso a todas las opciones de instrucción, mientras que algunas opciones y operaciones están restringidas en el formato de 64 bits 1730. Las instrucciones nativas disponibles en el formato de 64 bits 1730 varían según la realización. En algunas realizaciones, la instrucción se compacta en parte usando un conjunto de valores de índice en un campo de índice 1713. 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 1710.

Para cada formato, el código de operación (opcode) de la instrucción 1712 define la operación que la unidad de ejecución debe realizar. Las unidades de ejecución ejecutan cada instrucción en paralelo a lo largo de los múltiples elementos de datos de cada operando. Por ejemplo, en respuesta a una instrucción de suma, la unidad de ejecución realiza una operación de suma simultánea a través de cada canal de color que representa un elemento de textura o un elemento de imagen. Por defecto, la unidad de ejecución realiza cada instrucción a través de todos los canales de datos de los operandos. En algunas realizaciones, el campo de control de instrucción 1714 permite controlar ciertas opciones de ejecución, tales como la selección de canales (por ejemplo, predicación) y el orden de los canales de datos (por ejemplo, swizzle). Para las instrucciones en el formato de instrucción de 128 bits 1710, un campo de tamaño de ejecución 1716 limita el número de canales de datos que se ejecutarán en paralelo. En algunas realizaciones, el campo tamaño de ejecución 1716 no está disponible para el uso en el formato de instrucción compacto de 64 bits 1730.

Algunas instrucciones de unidad de ejecución tienen hasta tres operandos, incluyendo dos operandos de origen, src0 1720, src1 1722 y un destino 1718. En algunas realizaciones, las unidades de ejecución admiten instrucciones de doble destino, donde uno de los destinos está implícito. Las instrucciones de manipulación de datos pueden tener un tercer operando fuente (por ejemplo, SRC2 1724), donde el código de operación de la instrucción 1712 determina el número de operandos fuente. El último operando de origen de una instrucción puede ser un valor inmediato (por ejemplo, codificado de manera fija) pasado con la instrucción.

En algunas realizaciones, el formato de instrucción de 128 bits 1710 incluye un campo de modo de acceso/dirección 1726 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 varios operandos es proporcionada directamente por bits en la instrucción.

En algunas realizaciones, el formato de instrucción de 128 bits 1710 incluye un campo de modo de dirección/acceso 1726, que especifica un modo de dirección y/o un modo de acceso para la instrucción. En una realización, el modo de acceso se utiliza para definir una alineación de acceso a datos para la instrucción. Algunas realizaciones admiten modos de acceso que incluyen un modo de acceso alineado de 16 bytes y un modo de acceso alineado de 1 byte, donde la alineación de bytes del modo de acceso determina la alineación de acceso de los operandos de la instrucción. Por ejemplo, cuando está en un primer modo, la instrucción puede usar direccionamiento alineado con bytes para los operandos de origen y destino y cuando está en un segundo modo, la instrucción puede usar direccionamiento alineado con 16 bytes para todos los operandos de origen y destino.

En una realización, la porción de modo de dirección del campo de modo de acceso/dirección 1726 determina si la instrucción va a utilizar direccionamiento directo o indirecto. Cuando se utiliza el modo de direccionamiento de registro directo, unos bits de la instrucción proporcionan directamente la dirección de registro de uno o varios operandos. Cuando se utiliza un modo de direccionamiento de registro indirecto, la dirección de registro de uno o varios operandos se puede calcular en función de un valor de registro de dirección y un campo inmediato de dirección en la instrucción.

En algunas realizaciones, las instrucciones se agrupan basándose en los campos de bits del código de operación 1712 para simplificar la decodificación de código de operación 1740. Para un código de operación de 8 bits, los bits 4, 5 y 6 permiten que la unidad de ejecución determine el tipo de código de operación. La agrupación precisa de códigos de operación que se muestra es simplemente un ejemplo. En algunas realizaciones, un grupo de código de operación de movimiento y lógica 1742 incluye instrucciones lógicas y de movimiento de datos (por ejemplo, mover (mov), comparar (cmp)). En algunas realizaciones, el grupo de movimiento y lógica 1742 comparte los cinco bits más significativos (MSB), en donde las instrucciones mover (mov) tienen la forma de 0000xxxxb y las instrucciones de lógica tienen la forma de 0001xxxxb. Un grupo de instrucciones de control de flujo 1744 (por ejemplo, llamar, saltar (jmp)) incluye instrucciones en la forma de 0010xxxxb (por ejemplo, 0x20). Un grupo de instrucciones misceláneas 1746 incluye una mezcla de instrucciones, incluyendo instrucciones de sincronización (por ejemplo, espera, envío) en forma de 001 1xxxxb (por ejemplo, 0x30). Un grupo de instrucciones matemáticas en paralelo 1748 incluye instrucciones aritméticas a nivel de componente (por ejemplo, sumar, multiplicar (mul)) en forma de 0100xxxxb (por ejemplo, 0x40). El grupo matemático en paralelo 1748 realiza las operaciones aritméticas en paralelo a través de

los canales de datos. El grupo matemático vectorial 1750 incluye instrucciones aritméticas (por ejemplo, dp4) en forma de 0101xxxxb (por ejemplo, 0x50). El grupo matemático vectorial realiza operaciones aritméticas tales como cálculos de productos escalares en operandos vectoriales.

5 Canalización de gráficos

La **figura 18** es un diagrama de bloques de otra realización de un procesador de gráficos 1800. Los elementos de la **figura 18** que tienen los mismos números de referencia (o nombres) que los elementos de cualquier otra figura en el presente documento pueden operar o funcionar de cualquier manera similar a la descrita en cualquier otra parte del presente documento, pero no se limitan a tal cosa.

En algunas realizaciones, el procesador de gráficos 1800 incluye una canalización de gráficos 1820, una canalización de medios 1830, un motor de visualización 1840, una lógica de ejecución de subprocesos 1850 y una canalización de salida de renderizado 1870. En algunas realizaciones, el procesador de gráficos 1800 es un procesador de gráficos dentro de un sistema de procesamiento de múltiples núcleos que incluye uno o varios núcleos de procesamiento de propósito general. El procesador de gráficos es controlado por escrituras de registro en uno o varios registros de control (no mostrados) o mediante comandos emitidos al procesador de gráficos 1800 a través de una interconexión de anillo 1802. En algunas realizaciones, la interconexión de anillo 1802 acopla el procesador de gráficos 1800 a otros componentes de procesamiento, tales como otros procesadores gráficos o procesadores de propósito general. Los comandos de la interconexión de anillo 1802 son interpretados por un transmisor en continuo de comandos 1803, que proporciona instrucciones a componentes individuales de la canalización de gráficos 1820 o canalización de medios 1830.

En algunas realizaciones, el transmisor en continuo de comandos 1803 dirige el funcionamiento de un extractor de vértices 1805 que lee datos de vértices de una memoria y ejecuta comandos de procesamiento de vértices proporcionados por el transmisor en continuo de comandos 1803. En algunas realizaciones, el extractor de vértices 1805 proporciona datos de vértice a un sombreador de vértices 1807, que realiza operaciones de transformación y de iluminación de espacio de coordenadas en cada vértice. En algunas realizaciones, el extractor de vértices 1805 y el sombreador de vértices 1807 ejecutan instrucciones de procesamiento de vértices despachando subprocesos de ejecución a las unidades de ejecución 1852A-1852B mediante un despachador de subprocesos 1831.

En algunas realizaciones, las unidades de ejecución 1852A-1852B son una matriz de procesadores de vectores que tienen un conjunto de instrucciones para realizar operaciones de gráficos y de medios. En algunas realizaciones, las unidades de ejecución 1852A-1852B tienen una memoria caché de L1 1851 anexada que es específica para cada matriz o que se comparte entre las matrices. La memoria caché puede configurarse como una memoria caché de datos, una memoria caché de instrucciones, o una única memoria caché que está dividida para contener datos e instrucciones en diferentes particiones.

En algunas realizaciones, la canalización de gráficos 1820 incluye componentes de teselado para realizar el teselado acelerado por hardware de objetos 3D. En algunas realizaciones, un sombreador de casco programable 811 configura las operaciones de teselado. Un sombreador de dominio programable 817 proporciona una evaluación de extremo posterior del resultado del teselado. Un teselador 1813 opera en la dirección del sombreador de casco 1811 y contiene una lógica de propósito especial para generar un conjunto de objetos geométricos detallados en función de un modelo geométrico aproximado que se proporciona como entrada a la canalización de gráficos 1820. En algunas realizaciones, si no se usa el teselado, pueden eludirse los componentes de teselado (por ejemplo, el sombreador de casco 1811, el teselador 1813 y el sombreador de dominio 1817).

En algunas realizaciones, los objetos geométricos completos pueden ser procesados por un sombreador de geometría 1819 a través de uno o varios subprocesos despachados a las unidades de ejecución 1852A-1852B, o pueden proceder directamente al recortador 1829. En algunas realizaciones, el sombreador de geometría opera sobre objetos geométricos enteros, 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 1819 recibe la entrada del sombreador de vértices 1807. En algunas realizaciones, el sombreador de geometría 1819 es programable por 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 1829 procesa los datos de vértices. El recortador 1829 puede ser un recortador de función fija o un recortador programable con funciones de recorte y sombreado de geometría. En algunas realizaciones, un rasterizador y componente de prueba de profundidad 1873 en la canalización de salida de renderizado 1870 despacha sombreadores de píxeles para convertir los objetos geométricos en sus representaciones por píxel. En algunas realizaciones, la lógica de sombreador de píxeles se incluye en la lógica de ejecución de subprocesos 1850. En algunas realizaciones, una aplicación puede eludir el rasterizador y el componente de prueba de profundidad 1873 y acceder a datos de vértice sin rasterizar mediante una unidad de salida de flujo 1823.

El procesador de gráficos 1800 tiene un bus de interconexión, una estructura de interconexión o algún otro mecanismo de interconexión que permite el paso de datos y de mensajes entre los componentes principales del procesador. En algunas realizaciones, las unidades de ejecución 1852A-1852B y la memoria o memorias caché 1851 asociadas, el

muestreador de textura y de medios 1854 y la memoria caché de textura/muestreador 1858 se interconectan mediante un puerto de datos 1856 para realizar un acceso de memoria y comunicarse con componentes de canalización de salida de renderización del procesador. En algunas realizaciones, el muestreador 1854, las memorias caché 1851, 1858 y las unidades de ejecución 1852A-1852B tienen, cada uno, rutas de acceso de memoria separadas.

En algunas realizaciones, la canalización de salida de renderizado 1870 contiene un componente de rasterización y prueba de profundidad 1873 que convierte objetos basados en vértices en una representación asociada basada en píxeles. En algunas realizaciones, la lógica del rasterizador incluye una unidad de ventana/máscara para realizar la rasterización de triángulos y líneas de función fija. Una memoria caché de renderización 1878 y una memoria caché de profundidad 1879 asociadas también están disponibles en algunas realizaciones. Un componente de operaciones de píxeles 1877 realiza operaciones basadas en píxeles sobre los datos, aunque en algunos casos, las operaciones de píxeles asociadas con operaciones 2D (por ejemplo, transferencias de imágenes de bloques de bits con mezcla) son realizadas por el motor 2D 1841, o sustituidas en tiempo de visualización por el controlador de visualización 1843 utilizando planos de visualización superpuestos. En algunas realizaciones, una memoria caché L3 compartida 1875 está disponible para todos los componentes gráficos, permitiendo compartir datos sin el uso de la memoria principal del sistema.

En algunas realizaciones, la canalización de medios del procesador de gráficos 1830 incluye un motor de medios 1837 y un extremo frontal de vídeo 1834. En algunas realizaciones, el extremo frontal de vídeo 1834 recibe comandos de canalización del transmisor de comandos 1803. En algunas realizaciones, la canalización de medios 1830 incluye otro transmisor de comandos. En algunas realizaciones, el extremo frontal de vídeo 1834 procesa los comandos de medios antes de enviar el comando al motor de medios 1837. En algunas realizaciones, el motor de medios 1837 incluye funcionalidad de generación de subprocesos para generar subprocesos para su despacho a la lógica de ejecución de subprocesos 1850 a través del despachador de subprocesos 1831.

En algunas realizaciones, el procesador de gráficos 1800 incluye un motor de visualización 1840. En algunas realizaciones, el motor de visualización 1840 es externo al procesador 1800 y se acopla con el procesador de gráficos a través de la interconexión de anillo 1802, o algún otro bus o estructura de interconexión. En algunas realizaciones, el motor de visualización 1840 incluye un motor 2D 1841 y un controlador de visualización 1843. En algunas realizaciones, el motor de visualización 1840 contiene lógica de propósito especial capaz de operar independientemente de la canalización 3D. En algunas realizaciones, el controlador de visualización 1843 se acopla con un dispositivo de visualización (no mostrado), que puede ser un dispositivo de visualización integrado en el sistema, como en un ordenador portátil, o un dispositivo de visualización externo conectado a través de un conector de dispositivo de visualización.

En algunas realizaciones, la canalización de gráficos 1820 y la canalización de medios 1830 son configurables para realizar operaciones basadas en múltiples interfaces de programación de gráficos y medios y no son específicas de ninguna interfaz de programación de aplicaciones (API). En algunas realizaciones, el software controlador para el procesador de gráficos convierte llamadas API que son específicas de una biblioteca de gráficos o medios particular en comandos que pueden ser procesados por el procesador de gráficos. En algunas realizaciones, se proporciona soporte para la Open Graphics Library (OpenGL), Open Computing Language (OpenCL), y/o Vulkan graphics and compute API, todas de Khronos Group. En algunas realizaciones, también se puede facilitar compatibilidad con la biblioteca Direct3D de Microsoft Corporation. En algunas realizaciones, una combinación de estas bibliotecas puede ser compatible. También puede ser compatible con la biblioteca Open Source Computer Vision Library (OpenCV). También se facilitaría una futura API con una canalización 3D compatible si se puede hacer una correlación desde la canalización de la futura API con la canalización del procesador de gráficos.

Programación de la canalización de gráficos

La **figura 19A** es un diagrama de bloques que ilustra un formato de comandos de procesador de gráficos 1900 de acuerdo con algunas realizaciones. La **figura 19B** es un diagrama de bloques que ilustra una secuencia de comandos de procesador de gráficos 1910 de acuerdo con una realización. Los recuadros en línea continua en la **figura 19A** ilustran los componentes que se incluyen, en general, en un comando de gráficos, mientras que las líneas discontinuas incluyen componentes que son opcionales o que solo se incluyen en un subconjunto de los comandos de gráficos. El formato de comando de procesador de gráficos 1900 a modo de ejemplo de la **figura 19A** incluye campos de datos para identificar un cliente objetivo 1902 del comando, un código de operación (opcode) de comando 1904, y los datos relevantes 1906 para el comando. También se incluye un subcódigo de operación (sub-opcode) 1905 y un tamaño de comando 1908 en algunos comandos.

En algunas realizaciones, el cliente 1902 especifica la unidad cliente del dispositivo de gráficos que procesa los datos de comando. En algunas realizaciones, un analizador de comandos del procesador de gráficos examina el campo de cliente de cada comando para condicionar el procesamiento posterior del comando y dirigir los datos de comando a la unidad de cliente apropiada. En algunas realizaciones, las unidades cliente del procesador de gráficos incluyen una unidad de interfaz de memoria, una unidad de renderizado, una unidad 2D, una unidad 3D y una unidad multimedia. Cada unidad cliente tiene una canalización de procesamiento correspondiente que procesa los comandos. Una vez que el comando es recibido por la unidad cliente, la unidad cliente lee el código de operación 1904 y, si está presente,

el subcódigo de operación 1905 para determinar la operación a realizar. La unidad cliente ejecuta el comando utilizando la información del campo de datos 1906. Para algunos comandos se espera un tamaño de comando explícito 1908 para especificar el tamaño del comando. En algunas realizaciones, el analizador de comandos determina automáticamente el tamaño de al menos algunos de los comandos basándose en el código de operación de comando.

En algunas realizaciones, los comandos se alinean mediante múltiplos de una palabra doble.

El diagrama de flujo en la **figura 19B** muestra una secuencia de comandos de procesador de gráficos 1910 a modo de ejemplo. En algunas realizaciones, el software o firmware de un sistema de procesamiento de datos que cuenta con una realización de un procesador de gráficos utiliza una versión de la secuencia de comandos mostrada para configurar, ejecutar y terminar un conjunto de operaciones gráficas. Se muestra y se describe una secuencia de comandos de muestra únicamente con propósitos ilustrativos, ya que las realizaciones no están limitadas a estos comandos específicos o a esta secuencia de comandos. Además, los comandos se pueden emitir como un lote de comandos en una secuencia de comandos, de tal manera que el procesador de gráficos procese la secuencia de comandos al menos parcialmente de forma simultánea.

En algunas realizaciones, la secuencia de comandos del procesador de gráficos 1910 puede comenzar con un comando de vaciado de canalización 1912 para hacer que cualquier canalización de gráficos activa complete los comandos actualmente pendientes para la canalización. En algunas realizaciones, la canalización 3D 1922 y la canalización de medios 1924 no funcionan simultáneamente. El vaciado de canalización se realiza para hacer que la canalización de gráficos activa complete cualquier comando pendiente. En respuesta a un vaciado de canalización, el analizador de comandos para el procesador de gráficos pausará el procesamiento de comandos hasta que los motores de dibujo activos completen las operaciones pendientes y se invaliden las memorias caché de lectura relevantes. Opcionalmente, cualquier dato en la memoria caché de renderizado que esté marcado como "incorrecto" se puede descargar a la memoria. En algunas realizaciones, el comando de vaciado de canalización 1912 se puede utilizar para la sincronización de la canalización o antes de colocar el procesador de gráficos en un estado de bajo consumo.

En algunas realizaciones, un comando de selección de canalización 1913 se utiliza cuando una secuencia de comandos requiere que el procesador de gráficos conmute explícitamente entre canalizaciones. En algunas realizaciones, un comando de selección de canalización 1913 se requiere sólo una vez dentro de un contexto de ejecución antes de emitir comandos de canalización a menos que el contexto sea para emitir órdenes para ambas canalizaciones. En algunas realizaciones, se requiere un comando de vaciado de canalización 1912 inmediatamente antes de una conmutación de canalización por medio del comando de selección de canalización 1913.

En algunas realizaciones, un comando de control de canalización 1914 configura una canalización de gráficos para su operación y se utiliza para programar la canalización 3D 1922 y la canalización de medios 1924. En algunas realizaciones, el comando de control de canalización 1914 configura el estado de canalización para la canalización activa. En una realización, el comando de control de canalización 1914 se utiliza para la sincronización de canalización y para borrar datos de una o varias memorias caché dentro de la canalización activa antes de procesar un lote de comandos.

En algunas realizaciones, los comandos de estado de la memoria intermedia de retorno 1916 se utilizan para configurar un conjunto de memorias intermedias de retorno para que las respectivas canalizaciones escriban datos. Algunas operaciones de canalización requieren la asignación, selección o configuración de una o varias memorias intermedias de retorno en las que las operaciones escriben datos intermedios durante el procesamiento. En algunas realizaciones, el procesador de gráficos también utiliza una o varias memorias intermedias de retorno para almacenar datos de salida y para realizar la comunicación entre subprocesos. En algunas realizaciones, el estado de memoria intermedia de retorno 1916 incluye seleccionar el tamaño y el número de memorias intermedias de retorno a usar para un conjunto de operaciones de canalización.

Los comandos restantes en la secuencia de comandos difieren basándose en la canalización activa para las operaciones. Basándose en la determinación de la canalización 1920, la secuencia de comandos se adapta a la canalización 3D 1922 comenzando con el estado de canalización 3D 1930 o a la canalización de medios 1924 comenzando con el estado de canalización de medios 1940.

Los comandos para configurar el estado de la canalización 3D de 1930 incluyen comandos de configuración de estado 3D para el estado de la memoria intermedia de vértices, el estado de los elementos de vértices, el estado de color constante, el estado de la memoria intermedia de profundidad y otras variables de estado que deben configurarse antes de que se procesen los comandos de primitivas 3D. Los valores de estos comandos se determinan, al menos en parte, en función del API 3D particular en uso. En algunas realizaciones, los comandos de estado de canalización 3D 1930 también pueden deshabilitar o eludir selectivamente ciertos elementos de canalización si esos elementos no se usan.

En algunas realizaciones, el comando de primitivas 3D 1932 se usa para enviar primitivas 3D que serán procesadas mediante la canalización 3D. Los comandos y parámetros asociados que se pasan al procesador de gráficos a través del comando de primitiva 3D 1932 se reenvían a la función de extracción de vértices en la canalización de gráficos. La función de extracción de vértices usa los datos de comando de primitiva 3D 1932 para generar estructuras de datos

de vértices. Las estructuras de datos de vértices se almacenan en una o varias memorias intermedias de retorno. En algunas realizaciones, el comando de primitiva 3D 1932 se utiliza para realizar operaciones de vértices en primitivas 3D a través de sombreadores de vértices. Para procesar sombreadores de vértices, la canalización 3D 1922 despacha subprocesos de ejecución de sombreador a unidades de ejecución de procesador de gráficos.

En algunas realizaciones, se activa la canalización 3D 1922 mediante un comando o evento de ejecución 1934. En algunas realizaciones, una escritura de registro activa la ejecución de comando. En algunas realizaciones, se activa la ejecución mediante un comando 'ir' o 'disparar' en la secuencia de comandos. En una realización, se activa la ejecución de comando usando un comando de sincronización de canalización para vaciar la secuencia de comandos a través de la canalización de gráficos. La canalización 3D realizará un procesamiento de geometría para las primitivas 3D. Una vez completadas las operaciones, los objetos geométricos resultantes se rasterizan y el motor de píxeles colorea los píxeles resultantes. También se pueden incluir comandos adicionales para controlar el sombreado de píxeles y las operaciones de extremo posterior de píxeles para esas operaciones.

En algunas realizaciones, la secuencia de comandos 1910 del procesador de gráficos sigue la ruta de la canalización de medios 1924 cuando realiza operaciones de medios. En general, el uso específico y la forma de programación de la canalización de medios 1924 depende de las operaciones de medios o de cálculo a realizar. Las operaciones de decodificación de medios específicas pueden descargarse en la canalización de medios durante la decodificación de medios. En algunas realizaciones, la canalización de medios también se puede omitir y la decodificación de medios se puede realizar en su totalidad o en parte utilizando recursos proporcionados por uno o varios núcleos de procesamiento de propósito general. En una realización, la canalización de medios también incluye elementos para operaciones de unidad de procesador de gráficos de propósito general (GPGPU), donde el procesador de gráficos se utiliza para realizar operaciones vectoriales SIMD utilizando programas de sombreado computacional que no están explícitamente relacionados con el renderizado de primitivas de gráficos.

En algunas realizaciones, la canalización de medios 1924 se configura de forma similar a la canalización 3D 1922. Un conjunto de comandos para configurar el estado de la canalización de medios 1940 son enviados o colocados en una cola de comandos antes de los comandos de objeto de medios 1942. En algunas realizaciones, los comandos de estado de canalización de medios 1940 incluyen datos para configurar los elementos de canalización de medios que se usarán para procesar los objetos de medios. Esto incluye datos para configurar la lógica de decodificación y codificación de vídeo dentro de la canalización de medios, tal como el formato de codificación o decodificación. En algunas realizaciones, los comandos de estado de canalización de medios 1940 también admiten el uso de uno o varios punteros a elementos de estado "indirecto" que contienen un lote de ajustes de estado.

En algunas realizaciones, los comandos de objeto de medios 1942 suministran punteros a objetos de medios para su procesamiento por la canalización de medios. Los objetos de medios incluyen memorias intermedias de memoria que contienen datos de vídeo para ser procesados. En algunas realizaciones, todos los estados de canalización de medios deben ser válidos antes de que se emita un comando de objeto de medios 1942. Una vez que el estado de la canalización está configurado y los comandos de objetos de medios 1942 están en cola, la canalización de medios 1924 se activa a través de un comando de ejecución 1944 o un evento de ejecución equivalente (por ejemplo, escritura de registro). La salida de la canalización de medios 1924 puede ser procesada posteriormente por operaciones proporcionadas por la canalización 3D 1922 o la canalización de medios 1924. En algunas realizaciones, las operaciones de GPGPU se configuran y ejecutan de forma similar a las operaciones de medios.

Arquitectura de software de gráficos

La **figura 20** ilustra una arquitectura de software de gráficos a modo de ejemplo para un sistema de procesamiento de datos 2000 de acuerdo con algunas realizaciones. En algunas realizaciones, la arquitectura de software incluye una aplicación de gráficos 3D 2010, un sistema operativo 2020 y al menos un procesador 2030. En algunas realizaciones, el procesador 2030 incluye un procesador de gráficos 2032 y uno o varios núcleos de procesador de propósito general 2034. Cada uno de la aplicación de gráficos 2010 y el sistema operativo 2020 se ejecutan en la memoria de sistema 2050 del sistema de procesamiento de datos.

En algunas realizaciones, la aplicación de gráficos 3D 2010 contiene uno o varios programas de sombreado que incluyen instrucciones de sombreado 2012. Las instrucciones de lenguaje de sombreado pueden estar en un lenguaje de sombreado de alto nivel, como el lenguaje de sombreado de alto nivel (HLSL) o el lenguaje de sombreado OpenGL (GLSL). La aplicación también incluye instrucciones ejecutables 2014 en un lenguaje de máquina adecuado para su ejecución por el núcleo del procesador de propósito general 2034. La aplicación también incluye objetos gráficos 2016 definidos por datos de vértices.

En algunas realizaciones, el sistema operativo 2020 es un sistema operativo Microsoft® Windows® de Microsoft Corporation, un sistema operativo propietario de tipo UNIX o un sistema operativo de tipo UNIX de código abierto que utiliza una variante del kernel Linux. El sistema operativo 2020 puede admitir una API de gráficos 2022 tal como la API Direct3D, la API OpenGL o la API Vulkan. Cuando está en uso la API Direct3D, el sistema operativo 2020 usa un compilador de sombreador de extremo frontal 2024 para compilar cualquier instrucción de sombreado 2012 en HLSL en un lenguaje de sombreado de nivel inferior. La compilación puede ser una compilación justo a tiempo (JIT) o la

aplicación puede realizar una precompilación de sombreadores. En algunas realizaciones, los sombreadores de alto nivel se compilan en sombreadores de bajo nivel durante la compilación de la aplicación de gráficos 3D 2010. En algunas realizaciones, las instrucciones de sombreado 2012 se proporcionan en una forma intermedia, tal como una versión de la Representación Intermedia Portátil Estándar (SPIR) utilizada por la API Vulkan.

En algunas realizaciones, el controlador de gráficos de modo de usuario 2026 contiene un compilador de sombreador de extremo posterior 2027 para convertir las instrucciones de sombreado 2012 en una representación específica de hardware. Cuando la API OpenGL está en uso, las instrucciones de sombreado 2012 en el lenguaje de alto nivel GLSL se pasan al controlador de gráficos de modo de usuario 2026 para su compilación. En algunas realizaciones, el controlador de gráficos de modo de usuario 2026 usa las funciones de modo de núcleo de sistema operativo 2028 para comunicarse con un controlador de gráficos de modo kernel 2029. En algunas realizaciones, el controlador de gráficos de modo kernel 2029 se comunica con el procesador de gráficos 2032 para despachar comandos e instrucciones.

Implementaciones de núcleos IP

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

La **figura 21** es un diagrama de bloques que ilustra un sistema de desarrollo de núcleo IP 2100 que puede usarse para fabricar un circuito integrado para realizar operaciones de acuerdo con una realización. El sistema de desarrollo de núcleo IP 2100 puede usarse para generar diseños reutilizables modulares que pueden incorporarse en un diseño más grande o usarse para construir todo un circuito integrado (por ejemplo, un circuito integrado de SoC). Una instalación de diseño 2130 puede generar una simulación de software 2110 de un diseño de núcleo IP en un lenguaje de programación de alto nivel (por ejemplo, C/C++). La simulación de software 2110 puede ser usada para diseñar, probar y verificar el comportamiento del núcleo IP usando un modelo de simulación 2112. El modelo de simulación 2112 puede incluir simulaciones funcionales, de comportamiento y/o de temporización. Un diseño de nivel de transferencia de registro (RTL) 2115 puede ser creado o sintetizado a partir del modelo de simulación 2112. El diseño RTL 2115 es una abstracción del comportamiento del circuito integrado que modela el flujo de señales digitales entre registros de hardware, incluyendo la lógica asociada realizada usando las señales digitales modeladas. Además de un diseño RTL 2115, también pueden crearse, diseñarse o sintetizarse diseños de nivel inferior a nivel lógico o a nivel de transistor. Por lo tanto, los detalles particulares del diseño inicial y la simulación pueden variar.

El diseño RTL 2115 o equivalente puede ser sintetizado adicionalmente por la instalación de diseño en un modelo de hardware 2120, que puede estar en un lenguaje de descripción de hardware (HDL) o alguna otra representación de datos de diseño físico. El HDL se puede simular o probar más para verificar el diseño del núcleo IP. El diseño del núcleo IP puede ser almacenado para su entrega a una instalación de fabricación de terceros 2165 utilizando memoria no volátil 2140 (por ejemplo, disco duro, memoria flash, o cualquier medio de almacenamiento no volátil). Alternativamente, el diseño del núcleo IP puede ser transmitido (por ejemplo, a través de Internet) a través de una conexión cableada 2150 o inalámbrica 2160. La instalación de fabricación 2165 puede entonces fabricar un circuito integrado que se basa al menos en parte en el diseño del núcleo IP. El circuito integrado fabricado puede estar configurado para realizar operaciones de acuerdo con al menos una realización descrita en el presente documento.

Circuito integrado de sistema en chip, a modo de ejemplo

Las **figuras 22 a 24** ilustran circuitos integrados y procesadores de gráficos asociados, a modo de ejemplo, que pueden fabricarse usando uno o varios núcleos IP, de acuerdo con diversas realizaciones descritas en el presente documento. Además de lo que se ilustra, se pueden incluir otros circuitos y lógica, incluyendo procesadores/núcleos de gráficos adicionales, controladores de interfaz de periféricos o núcleos de procesador de propósito general.

La **figura 22** es un diagrama de bloques que ilustra un circuito integrado de sistema en chip 2200 a modo de ejemplo que puede ser fabricado utilizando uno o varios núcleos IP, de acuerdo con una realización. El circuito integrado 2200 a modo de ejemplo incluye uno o varios procesadores de aplicación 2205 (por ejemplo, CPU), al menos un procesador de gráficos 2210, y puede incluir adicionalmente un procesador de imágenes 2215 y/o un procesador de vídeo 2220, cualquiera de los cuales puede ser un núcleo IP modular del mismo o de múltiples instalaciones de diseño diferentes. El circuito integrado 2200 incluye una lógica de bus o de periféricos que incluye un controlador de USB 2225, un controlador de UART 2230, un controlador de SPI/SDIO 2235 y un controlador de I²S/I²C 2240. Además, el circuito

integrado puede incluir un dispositivo de visualización 2245 acoplado a uno o varios de un controlador de interfaz multimedia de alta definición (HDMI) 2250 y una interfaz de visualización de interfaz de procesador de industria móvil (MIPI) 2255. El almacenamiento puede ser proporcionado por un subsistema de memoria flash 2260 que incluye memoria flash y un controlador de memoria flash. La interfaz de memoria puede proporcionarse a través de un controlador de memoria 2265 para acceder a dispositivos de memoria SDRAM o SRAM. Algunos circuitos integrados incluyen adicionalmente un motor de seguridad integrado 2270.

La **figura 23** es un diagrama de bloques que ilustra un procesador de gráficos 2310 a modo de ejemplo de un circuito integrado de sistema en chip que puede fabricarse usando uno o varios núcleos IP, de acuerdo con una realización.

El procesador de gráficos 2310 puede ser una variante del procesador de gráficos 2210 de la **figura 22**. El procesador de gráficos 2310 incluye un procesador de vértices 2305 y uno o varios procesadores de fragmentos 2315A a 2315N (por ejemplo, 2315A, 2315B, 2315C, 2315D, hasta 2315N-1 y 2315N). El procesador de gráficos 2310 puede ejecutar diferentes programas de sombreado mediante una lógica separada, de modo que el procesador de vértices 2305 está optimizado para ejecutar operaciones para programas de sombreador de vértices, mientras que el uno o varios procesadores de fragmentos 2315A a 2315N ejecutan operaciones de sombreador de fragmentos (por ejemplo, píxeles) para programas sombreado de fragmentos o de píxeles. El procesador de vértices 2305 realiza la etapa de procesamiento de vértices de la canalización de gráficos 3D y genera primitivas y datos de vértices. El procesador o procesadores de fragmentos 2315A a 2315N usan los datos de primitiva y de vértice generados por el procesador de vértices 2305 para producir una memoria intermedia de trama que se visualiza en un dispositivo de visualización. En una realización, el procesador o procesadores de fragmentos 2315A a 2315N están optimizados para ejecutar programas de sombreado de fragmentos según lo previsto en la API de OpenGL, que se pueden usar para realizar operaciones similares como un programa de sombreado de píxeles según lo previsto en la API de Direct 3D.

El procesador de gráficos 2310 incluye adicionalmente una o varias unidades de gestión de memoria (MMU) 2320A-2320B, memoria o memorias caché 2325A-2325B e interconexión o interconexiones de circuito 2330A-2330B. La una o varias MMU 2320A-2320B proporcionan correlación de dirección virtual a física para el circuito integrado 2310, incluyendo para el procesador de vértices 2305 y/o el procesador o procesadores de fragmentos 2315A a 2315N, que pueden hacer referencia a los datos de vértices o de imagen/textura almacenados en memoria, además de los datos de vértices o imagen/textura almacenados en la una o varias memorias caché 2325A-2325B. En una realización, la una o varias MMU 2325A-2325B pueden estar sincronizadas con otras MMU dentro del sistema, que incluyen una o varias MMU asociadas con el uno o varios procesadores de aplicación 2205, el procesador de imagen 2215 y/o el procesador de vídeo 2220 de la **figura 22**, de manera que cada procesador 2205 a 2220 puede participar en un sistema virtual de memoria compartida o unificada. La una o varias interconexiones de circuito 2330A-2330B posibilitan que el procesador de gráficos 2310 interactúe con otros núcleos IP dentro del SoC, o bien mediante un bus interno del SoC o bien mediante una conexión directa, de acuerdo con realizaciones.

La **figura 24** es un diagrama de bloques que ilustra un procesador de gráficos 2410 adicional a modo de ejemplo de un circuito integrado de sistema en chip que puede fabricarse usando uno o varios núcleos IP, de acuerdo con una realización. El procesador de gráficos 2410 puede ser una variante del procesador de gráficos 2210 de la **figura 22**. El procesador de gráficos 2410 incluye la una o varias MMU 2320A-2320B, las memorias caché 2325A-2325B y las interconexiones de circuito 2330A-2330B del circuito integrado 2300 de la **figura 23**.

El procesador de gráficos 2410 incluye uno o varios núcleos de sombreador 2415A a 2415N (por ejemplo, 2415A, 2415B, 2415C, 2415D, 2415E, 2415F a 2415N-1 y 2415N), lo que prevé una arquitectura de núcleo de sombreador unificada en la que un único núcleo o tipo de núcleo puede ejecutar todos los tipos de código de sombreado programable, incluyendo un código de programa de sombreado para implementar sombreadores de vértices, sombreadores de fragmentos y/o sombreadores de cálculo. El número exacto de núcleos de sombreador presentes puede variar entre realizaciones e implementaciones. Además, el procesador de gráficos 2410 incluye un gestor de tareas entre núcleos 2405, que actúa como un despachador de subprocesos para despachar subprocesos de ejecución a uno o varios núcleos de sombreador 2415A a 2415N y una unidad de división en teselas 2418 para acelerar las operaciones de división en teselas para el renderizado basado en teselas, donde las operaciones de renderizado para una escena se subdividen en un espacio de imagen, por ejemplo para aprovechar la coherencia espacial local dentro de una escena o para optimizar el uso de memorias caché internas.

Además, en la descripción y las reivindicaciones, pueden utilizarse los términos "acoplado" y "conectado", junto con sus derivados. En algunas realizaciones, "conectado" puede usarse para indicar que dos o más elementos están en contacto físico o eléctrico directo entre sí. "Acoplado" puede significar que dos o más elementos están en contacto físico o eléctrico directo. Sin embargo, "acoplado" también puede significar que dos o más elementos pueden no estar en contacto directo entre sí, pero aun así pueden cooperar o interactuar entre sí.

Por tanto, aunque las realizaciones se han descrito en un lenguaje específico para rasgos característicos estructurales y/o los actos metodológicos, debe entenderse que la materia objeto reivindicada puede no estar limitada a los rasgos característicos o actos específicos descritos. Más bien, los rasgos característicos y los actos específicos se dan a conocer como formas de ejemplo de implementación de la materia objeto de la reivindicación.

Además, la descripción y los dibujos anteriores deben considerarse en un sentido ilustrativo y no restrictivo. Los expertos en la materia comprenderán que pueden realizarse diversas modificaciones y cambios en las realizaciones descritas en el presente documento sin apartarse del alcance de la invención, tal como se establece en las reivindicaciones adjuntas.

REIVINDICACIONES

1. Procedimiento ejecutado en un aparato que comprende lógica, comprendiendo, al menos parcialmente, lógica de hardware, comprendiendo el procedimiento:
 - 5 recopilar (650) una o varias métricas de rendimiento de canalización asociadas con una o varias etapas de una canalización de procesamiento de gráficos, en el que la una o varias métricas de rendimiento de canalización incluyen características de carga de trabajo y en el que la una o varias etapas de la canalización de gráficos comprende:
al menos uno de un despachador de subprocesos (662), una
10 o varias unidades de ejecución (664), uno o varios puertos de datos (666), una o varias memorias caché (668), y uno o varios controladores de memoria (670); y
ajustar (655) al menos uno de un voltaje operativo o una frecuencia operativa de la una o varias etapas en la canalización de gráficos basándose al menos en parte en la una o varias métricas de rendimiento,
caracterizado por que,
cuando las características de la carga de trabajo indican una carga de trabajo de cálculo, la etapa de ajuste (655)
15 comprende, además, aumentar un voltaje operativo y/o una frecuencia operativa de la una o varias unidades de ejecución (664) que están ocupadas y reducir un voltaje operativo y/o una frecuencia operativa del uno o varios puertos de datos (666) y una o varias memorias caché (668) que no están ocupadas, y
cuando las características de la carga de trabajo indican una carga de trabajo que implementa un gran número de instrucciones de memoria, la etapa de ajustar (655) comprende, además, aumentar un voltaje operativo y/o una
20 frecuencia operativa del uno o varios puertos de datos (666) y de la una o varias memorias caché (668) que están ocupados y reducir un voltaje operativo y/o una frecuencia operativa de la una o varias unidades de ejecución (664) que no están ocupadas.
2. Aparato, que comprende:
 - 25 lógica, que comprende al menos parcialmente lógica de hardware, para realizar un procedimiento según la reivindicación 1.
3. Uno o varios medios legibles por ordenador que tienen almacenadas instrucciones que, cuando se ejecutan en un
30 aparato según la reivindicación 2, configuran el aparato para realizar un procedimiento según la reivindicación 1.

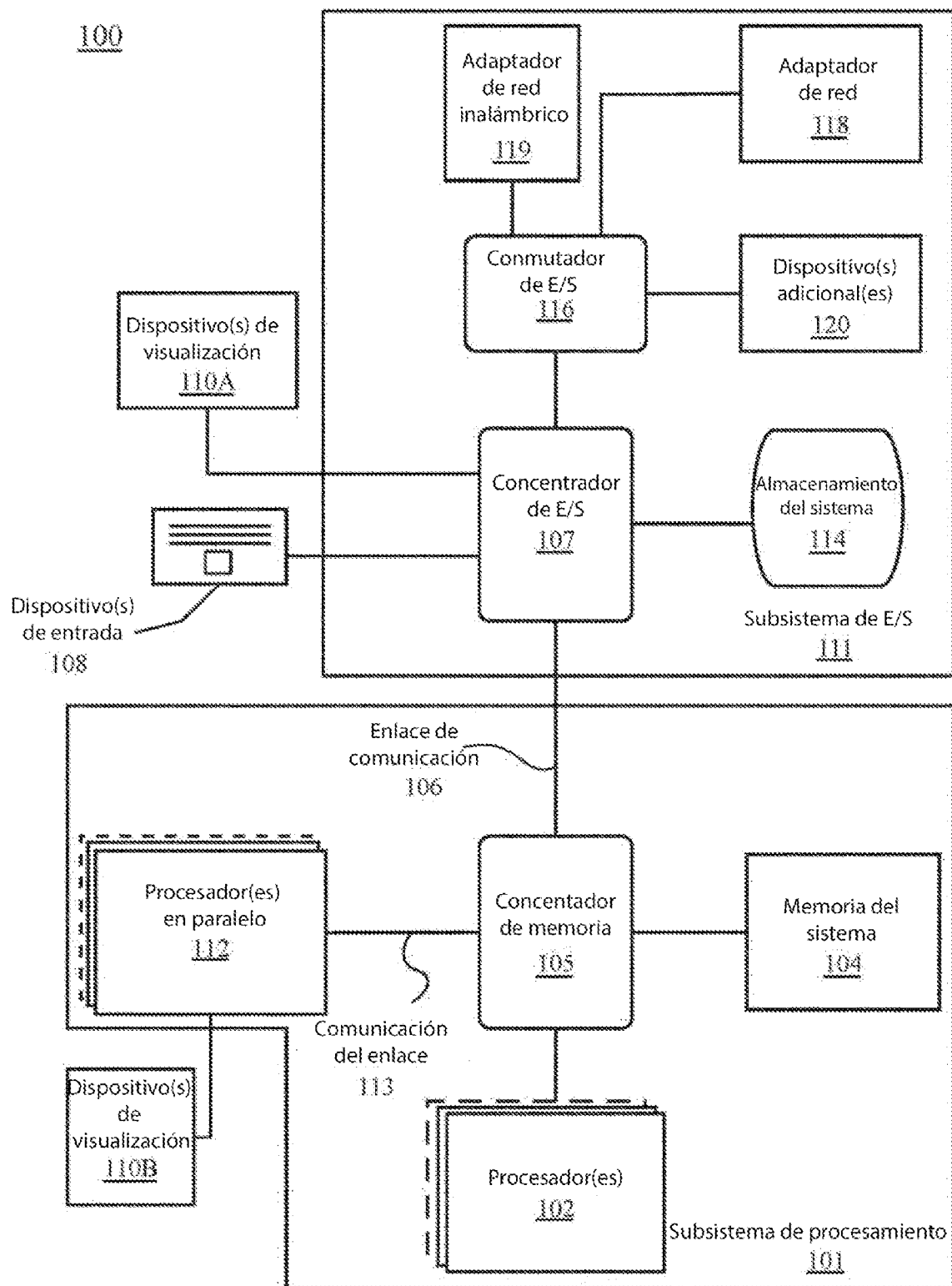


FIG. 1

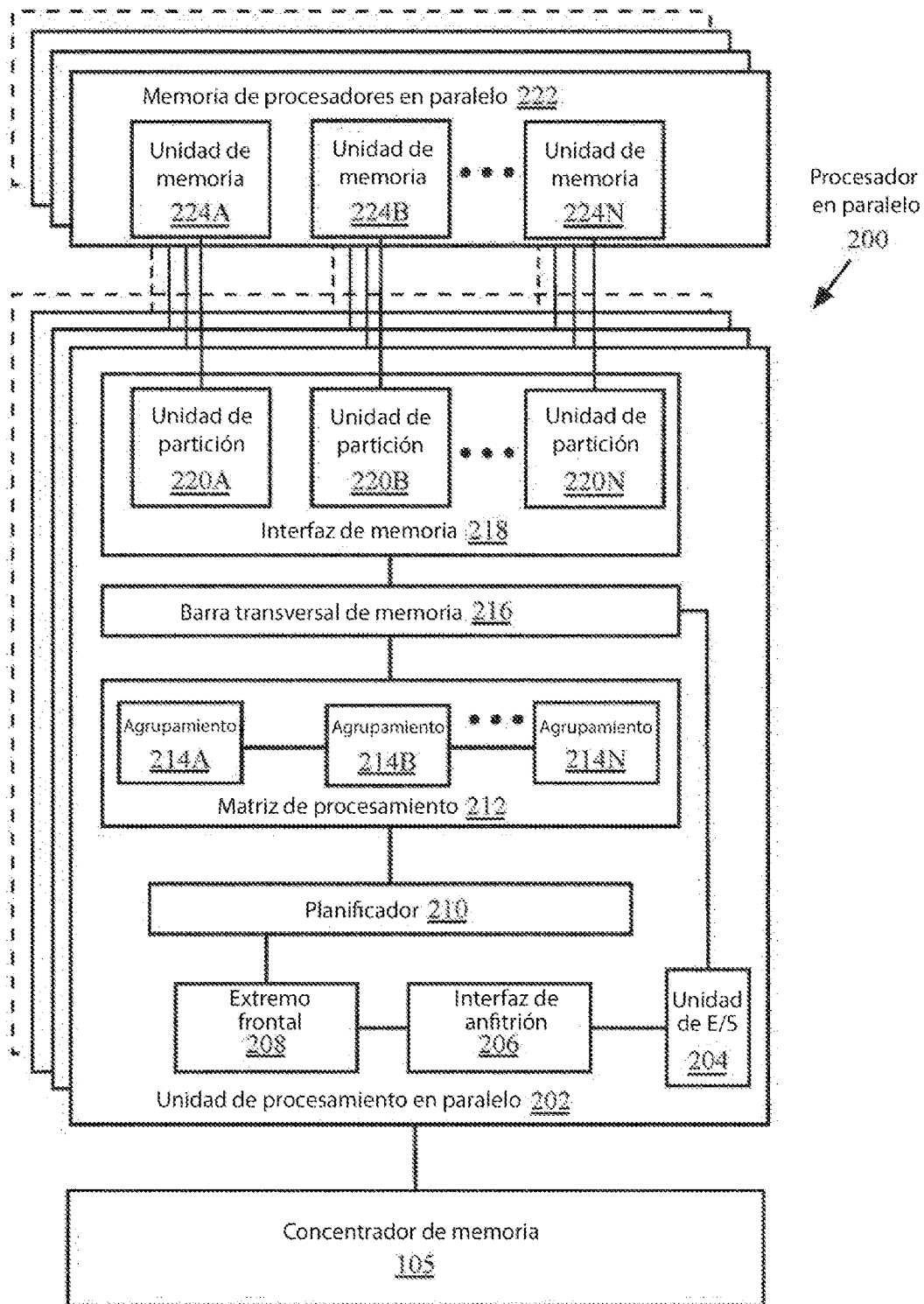


FIG. 2A

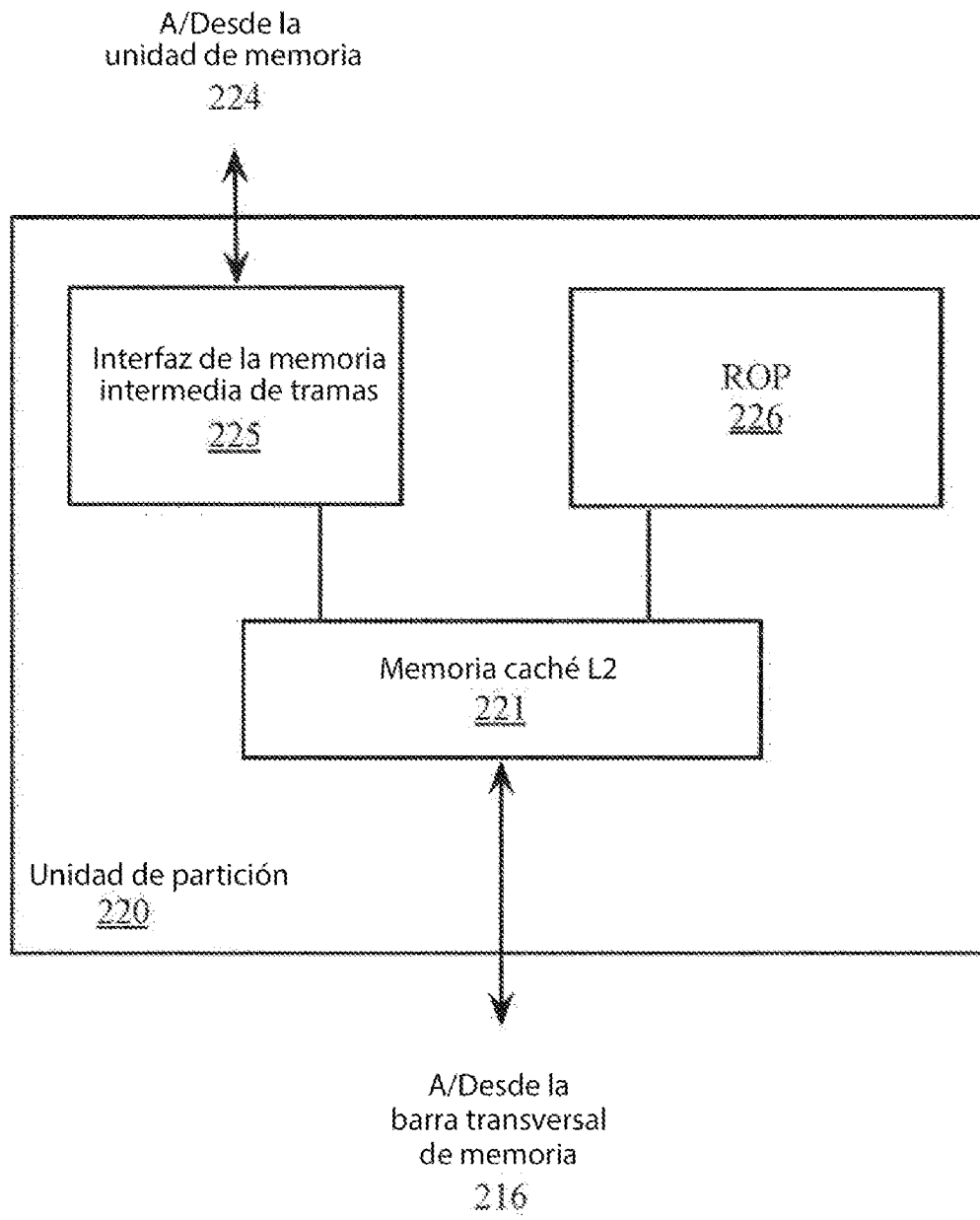


FIG. 2B

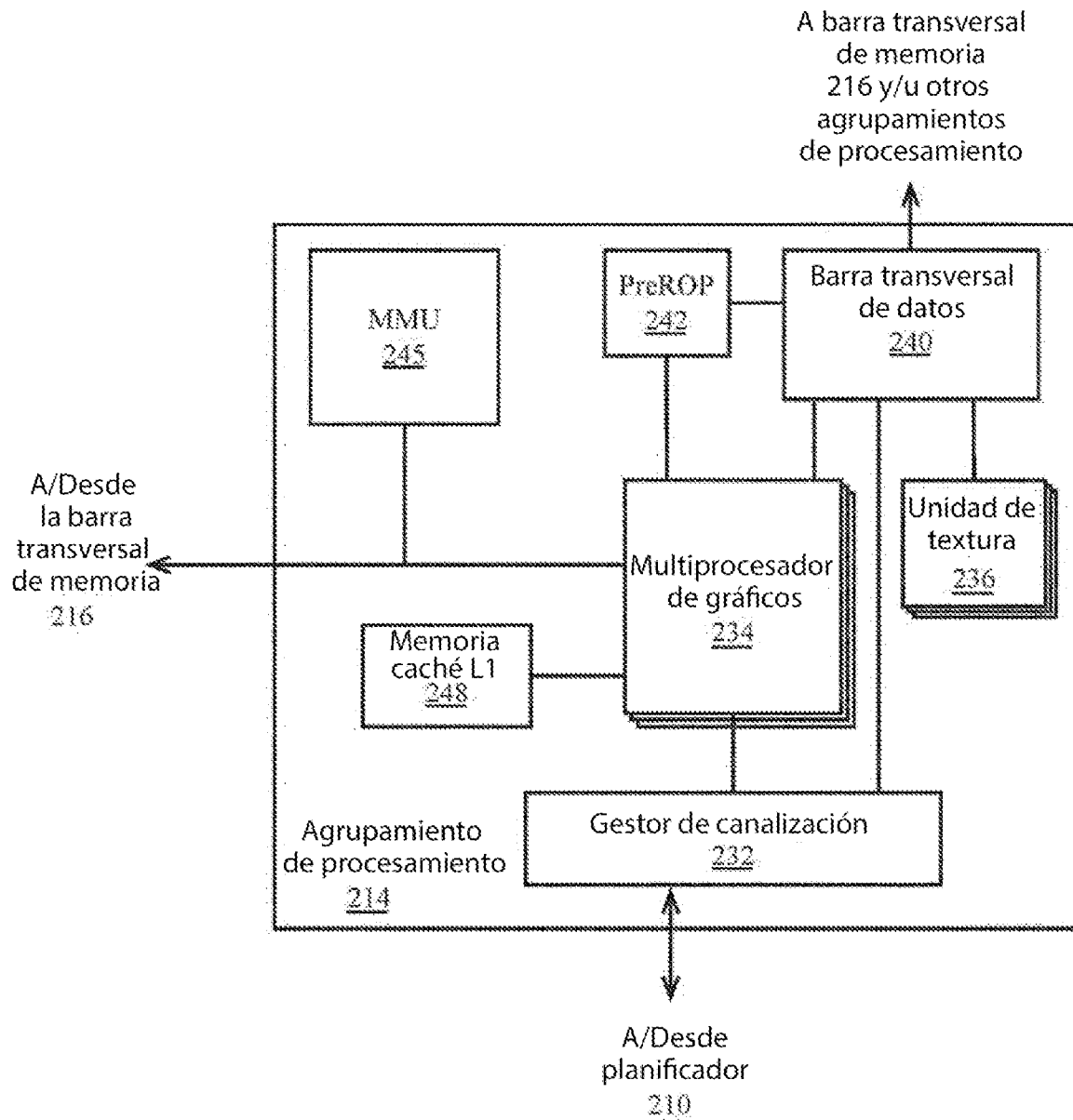


FIG. 2C

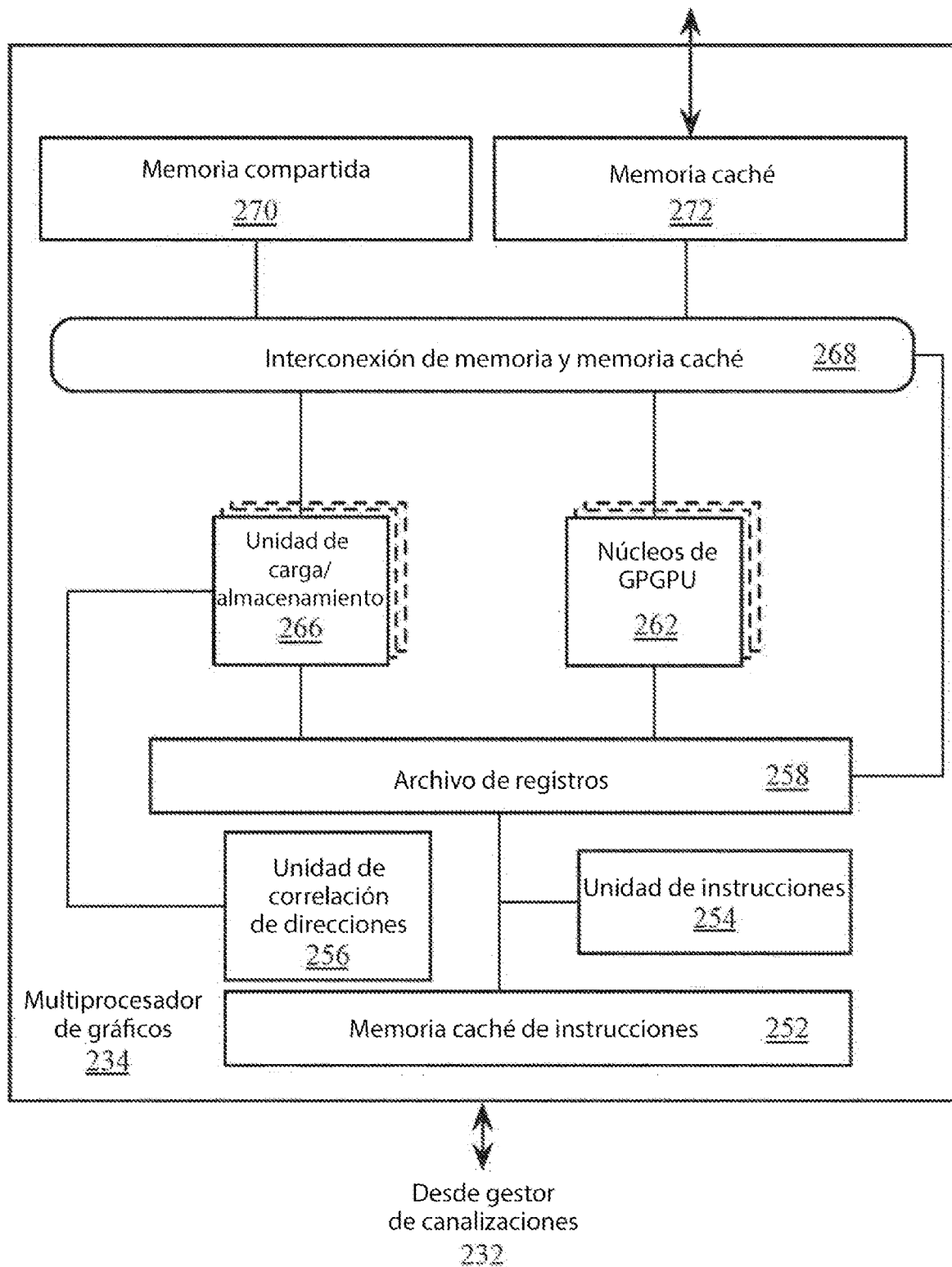


FIG. 2D

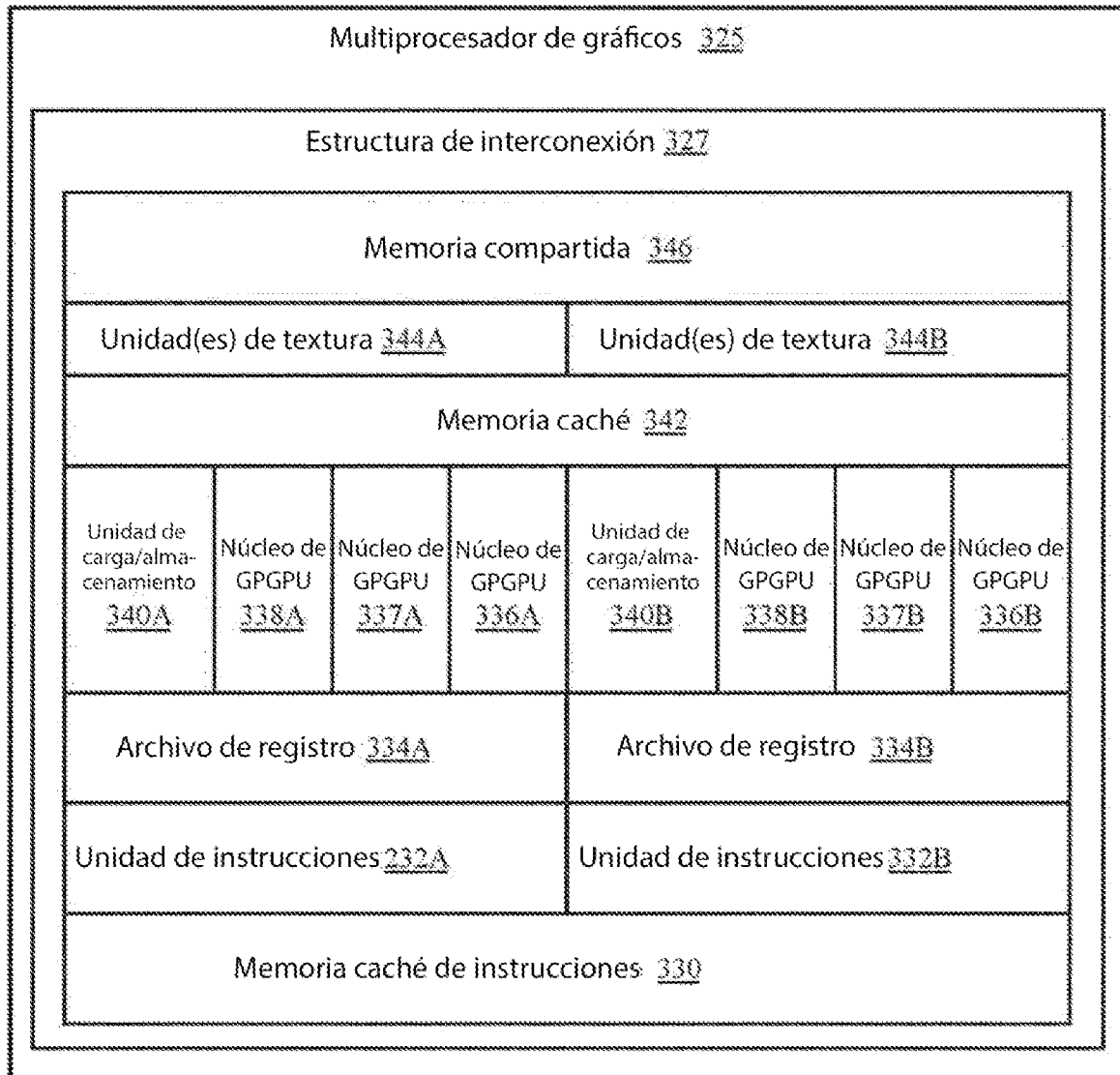


FIG. 3A

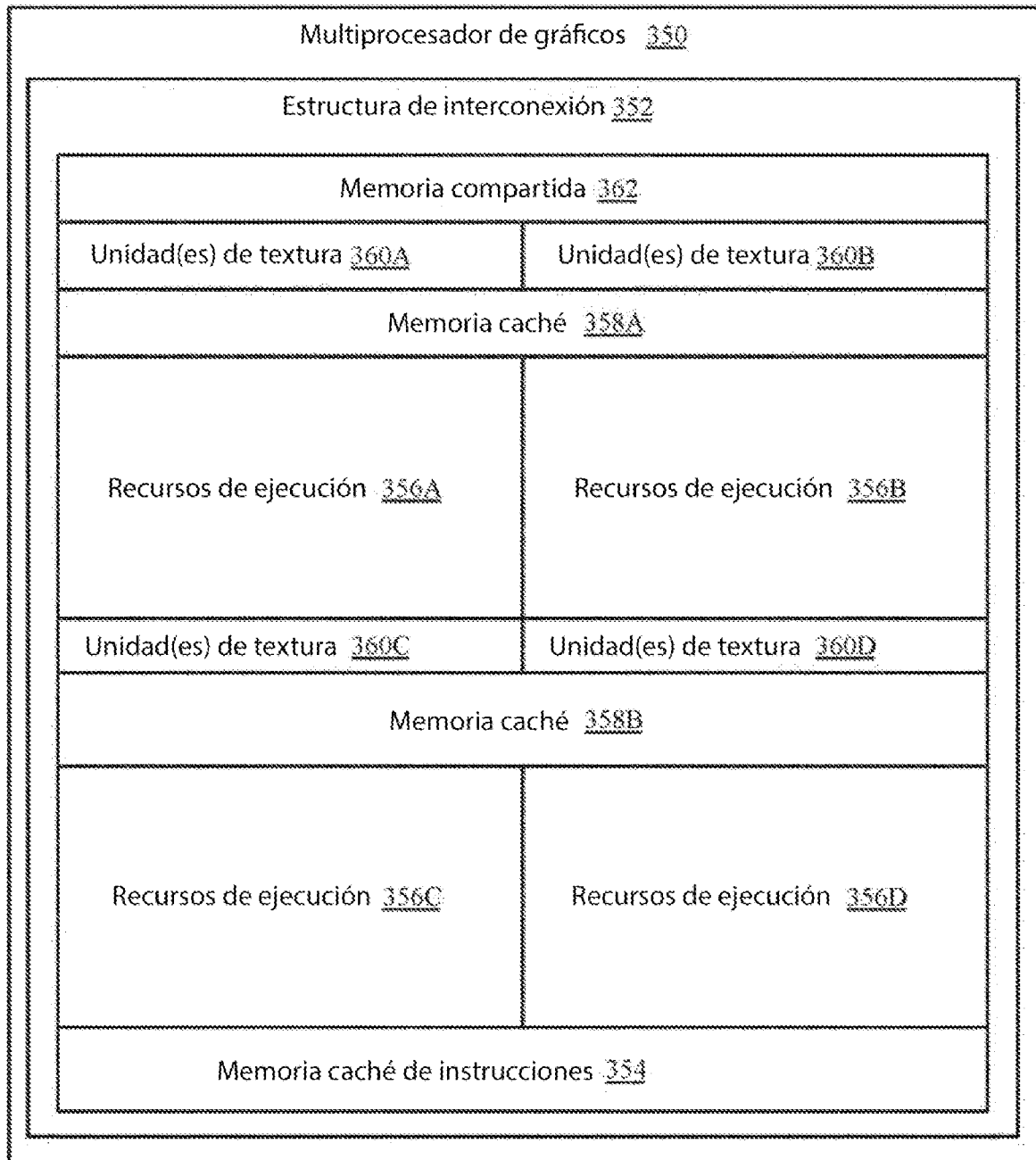


FIG. 3B

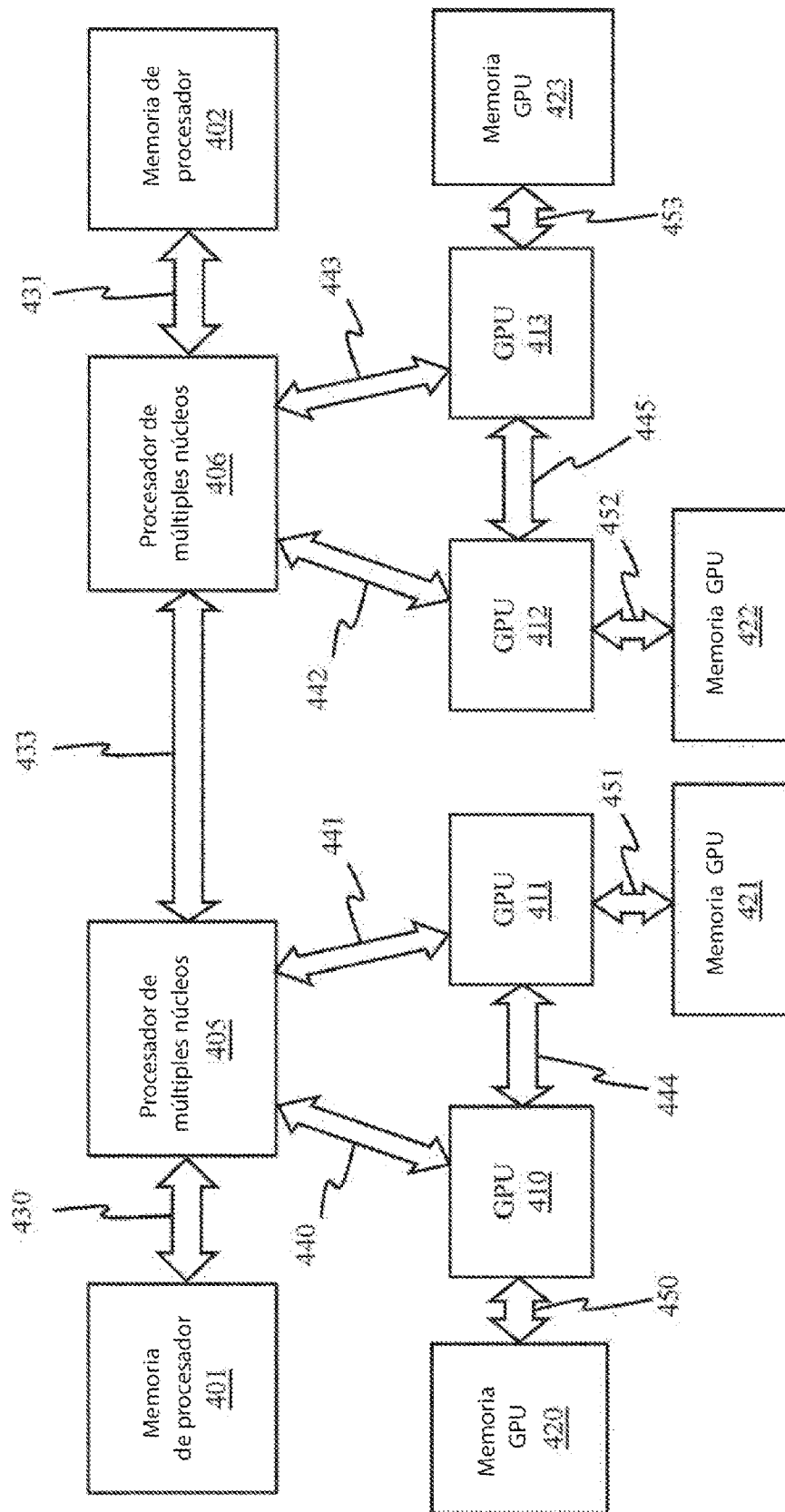


FIG. 4A

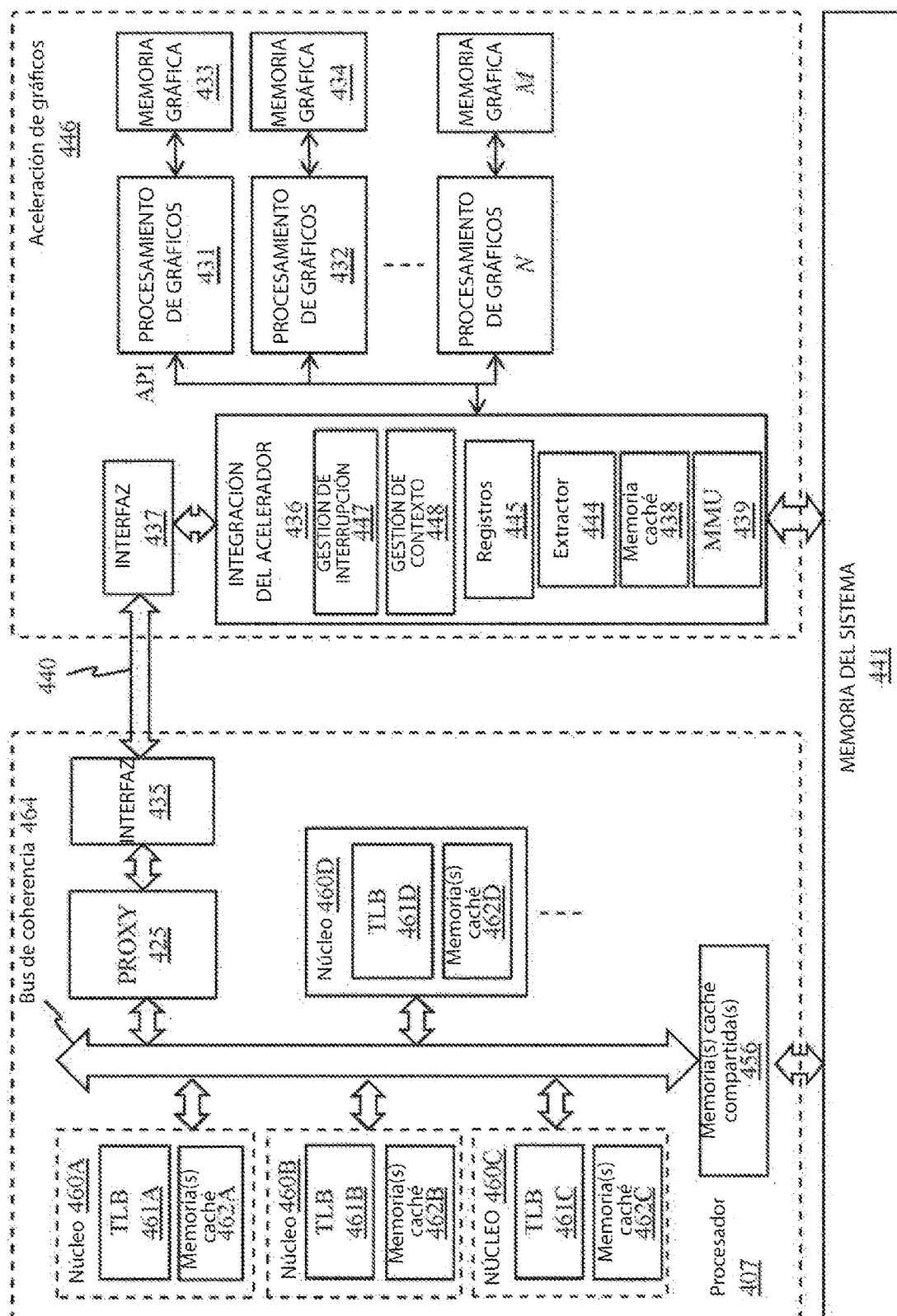


FIG. 4B

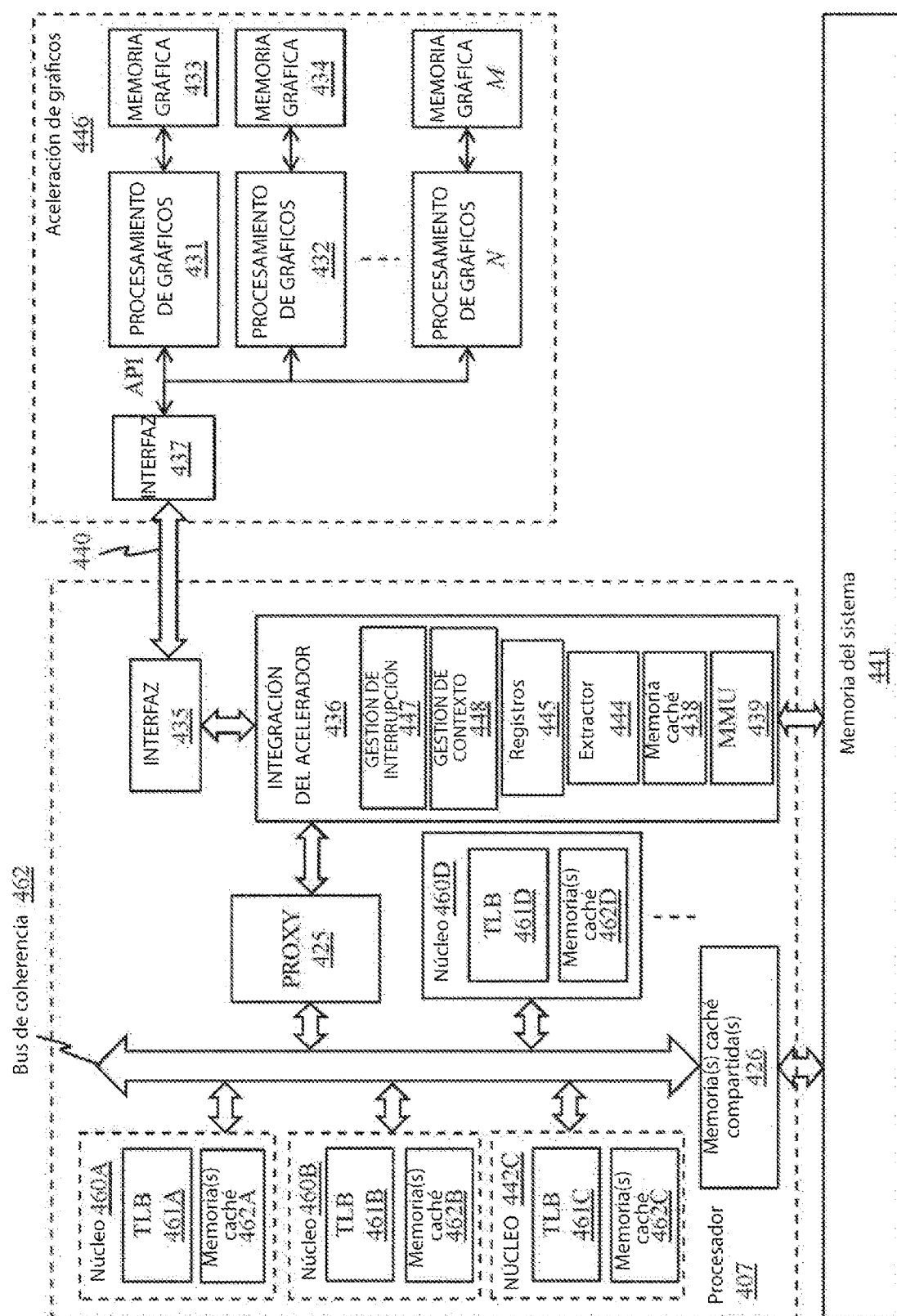


FIG. 4C

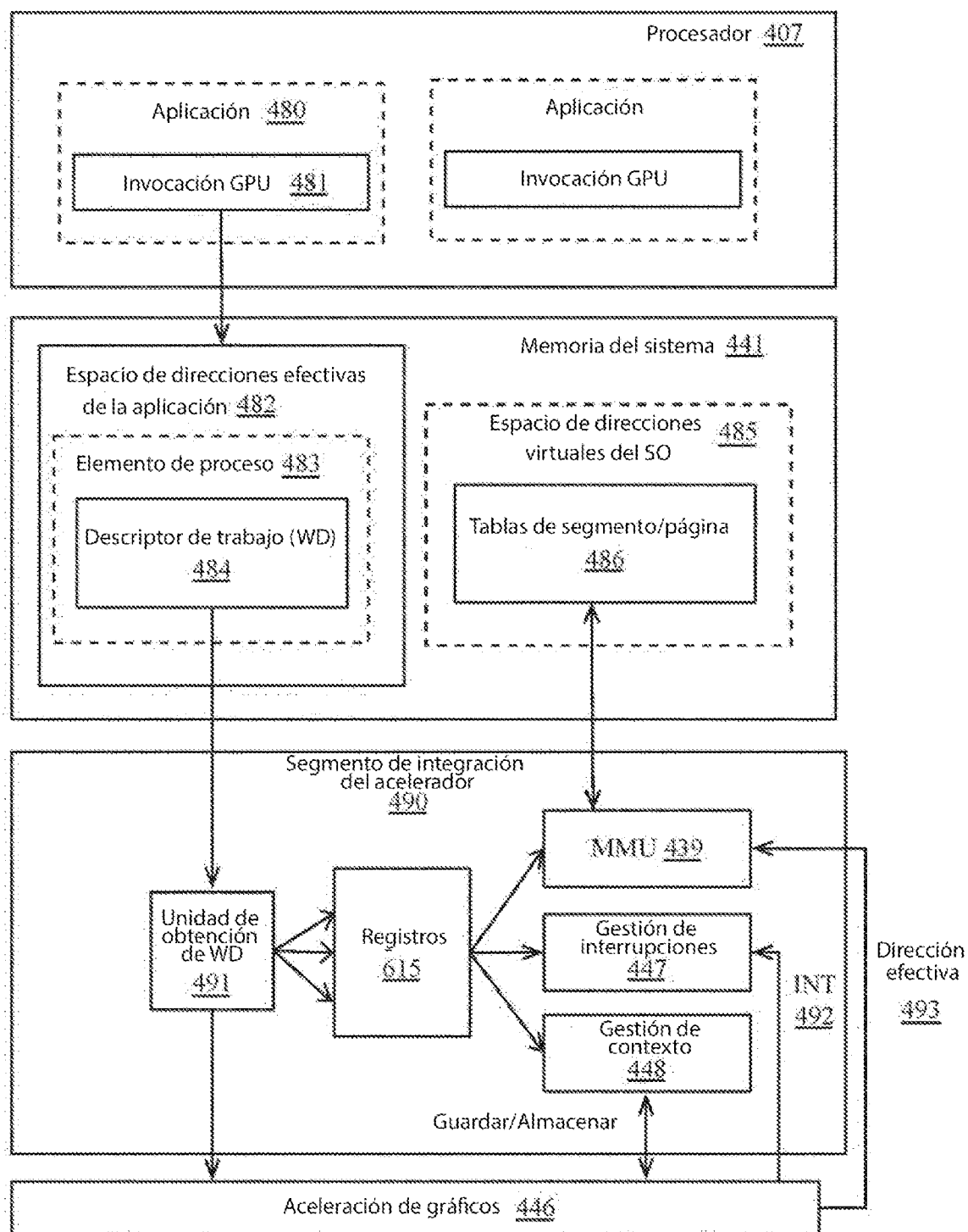


FIG. 4D

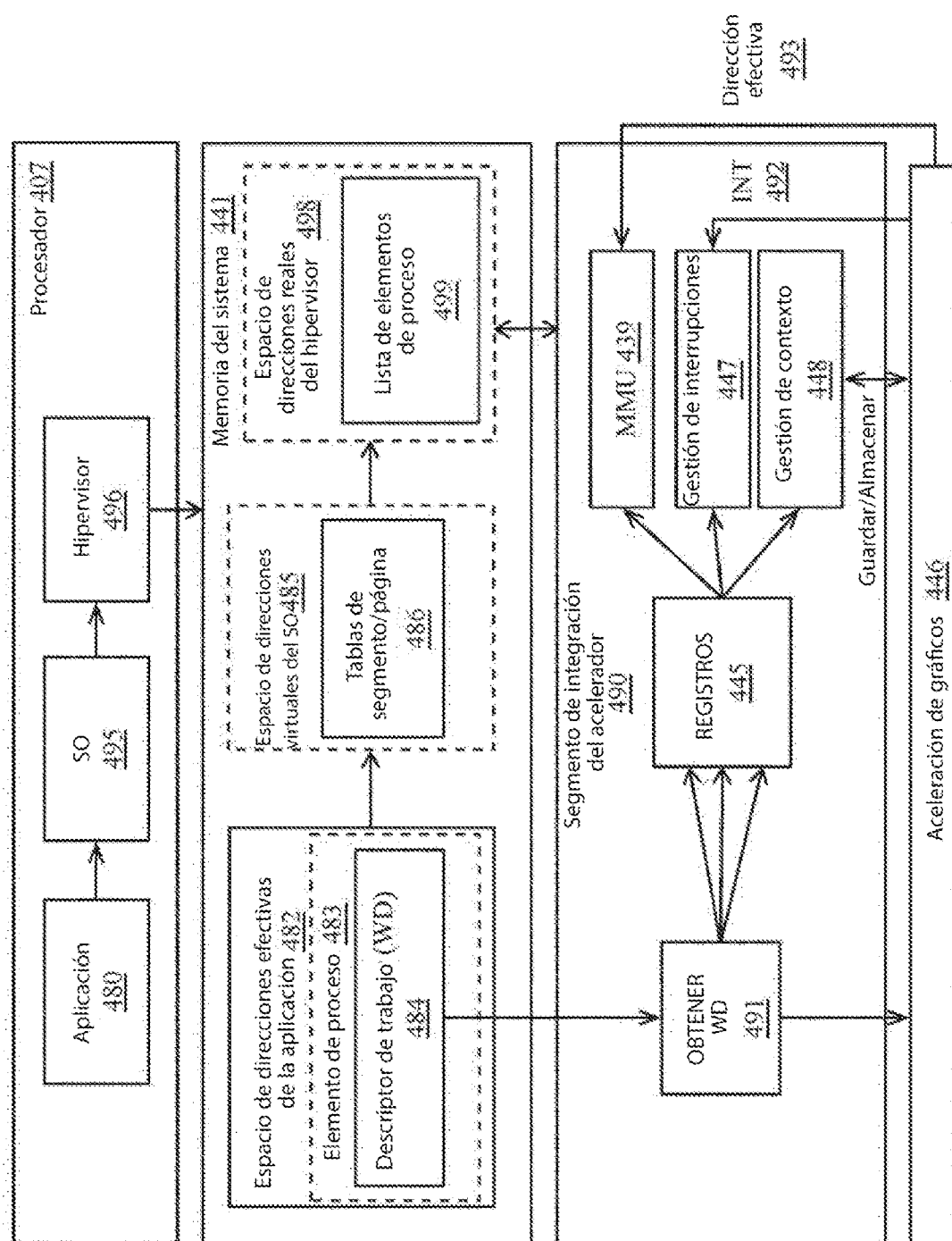


FIG. 4E

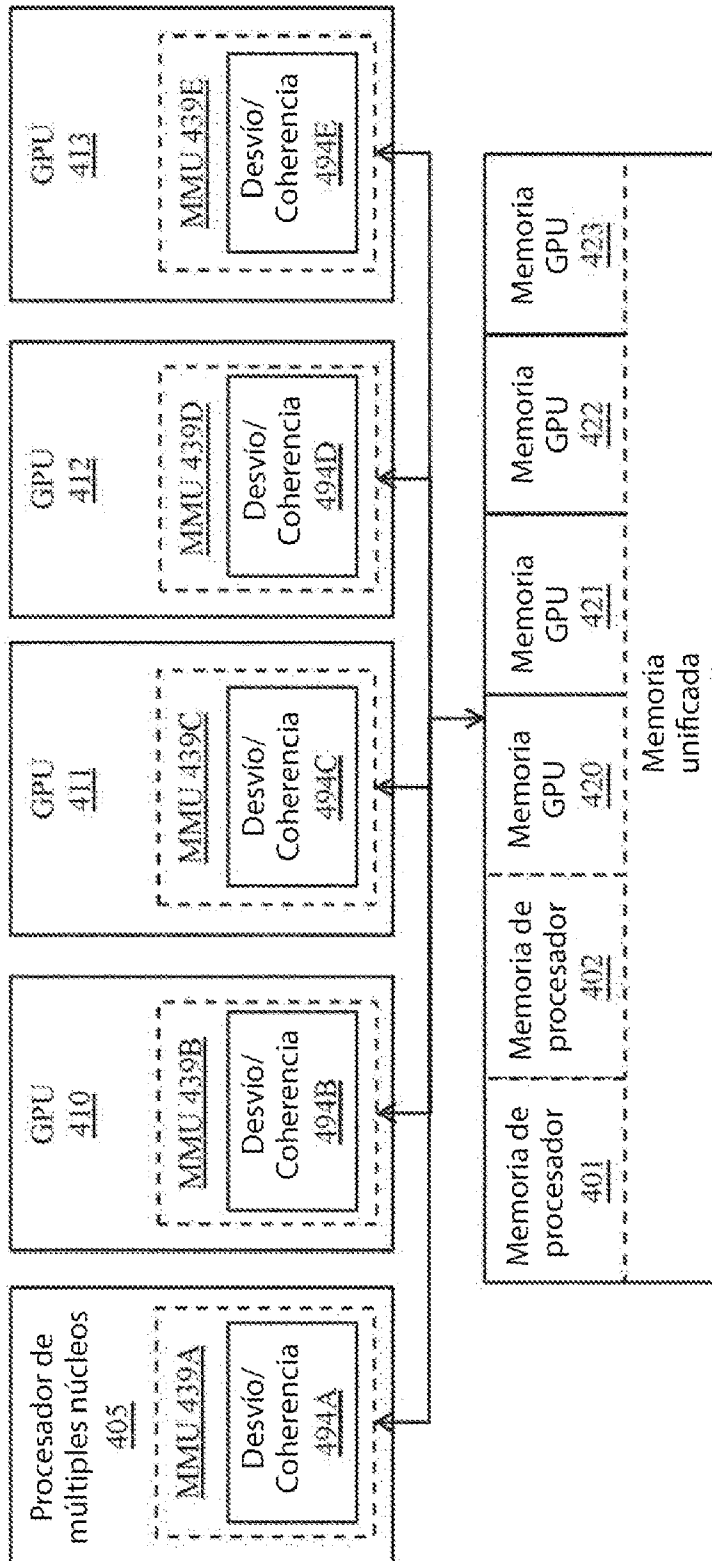


FIG. 4F

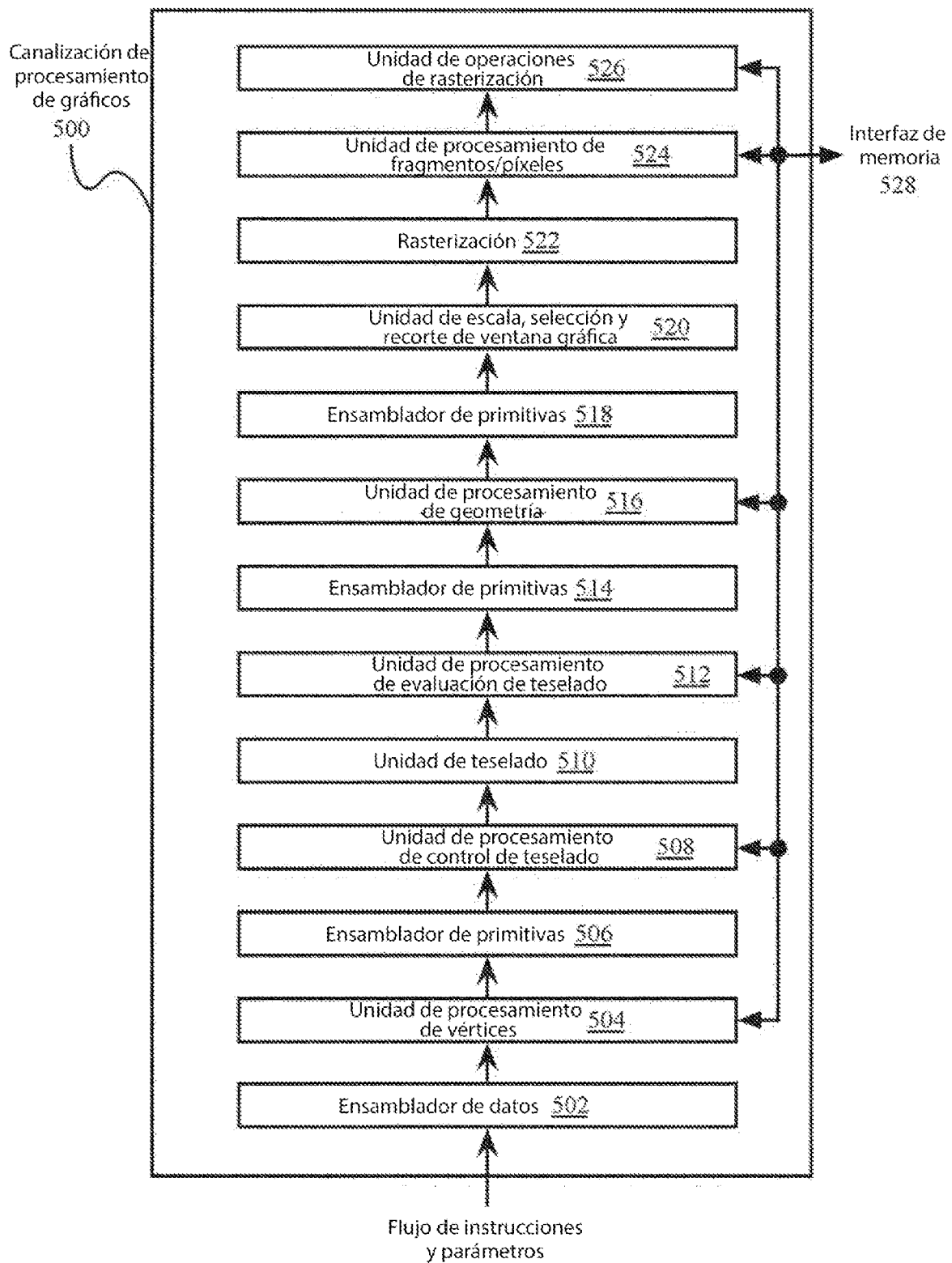


FIG. 5

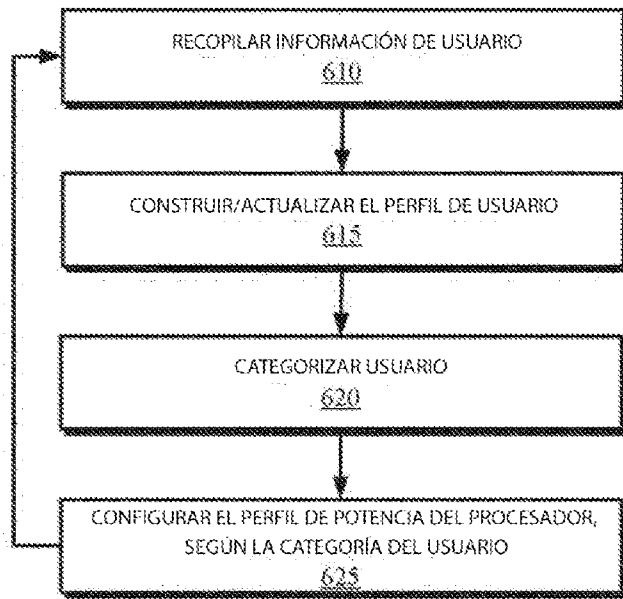


FIG. 6A

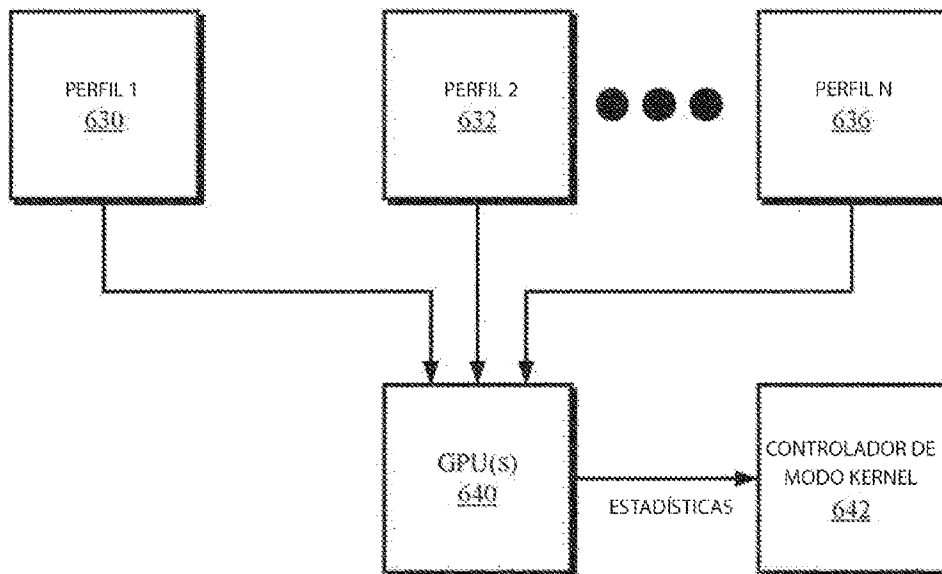


FIG. 6B

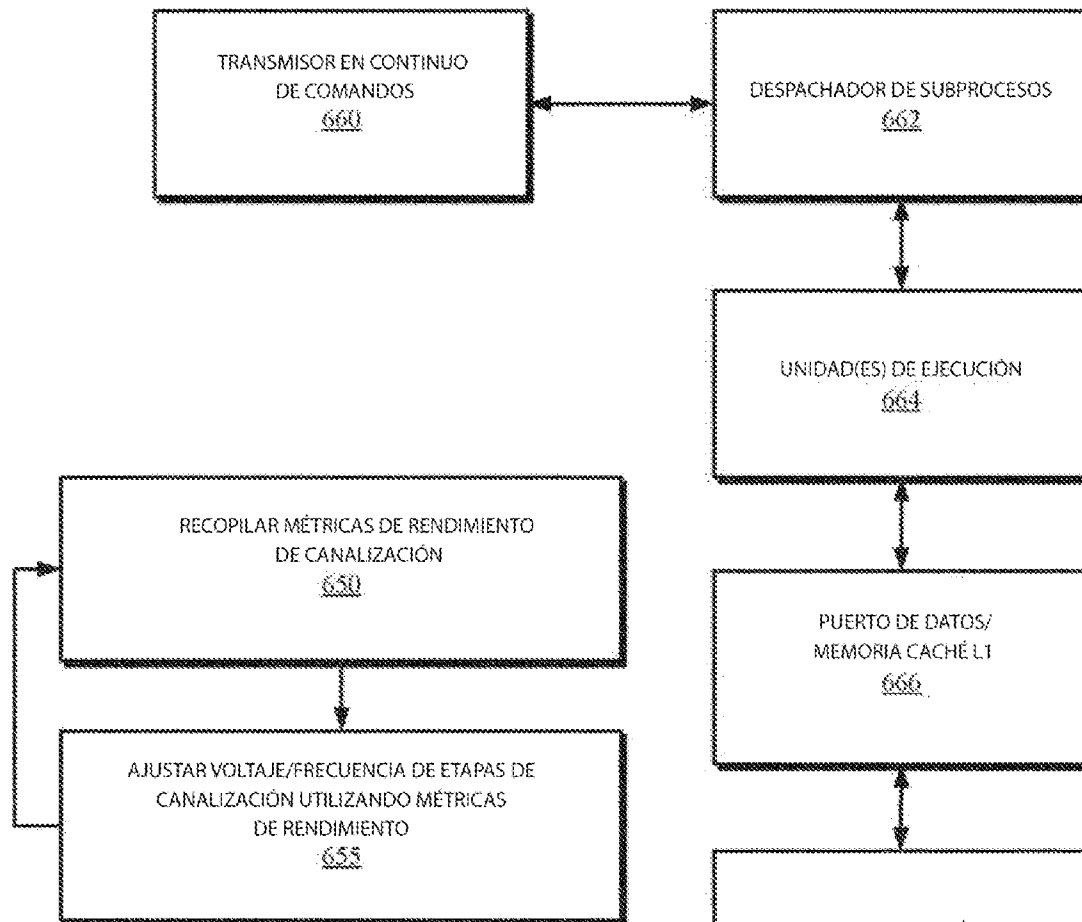


FIG. 6C

FIG. 6D

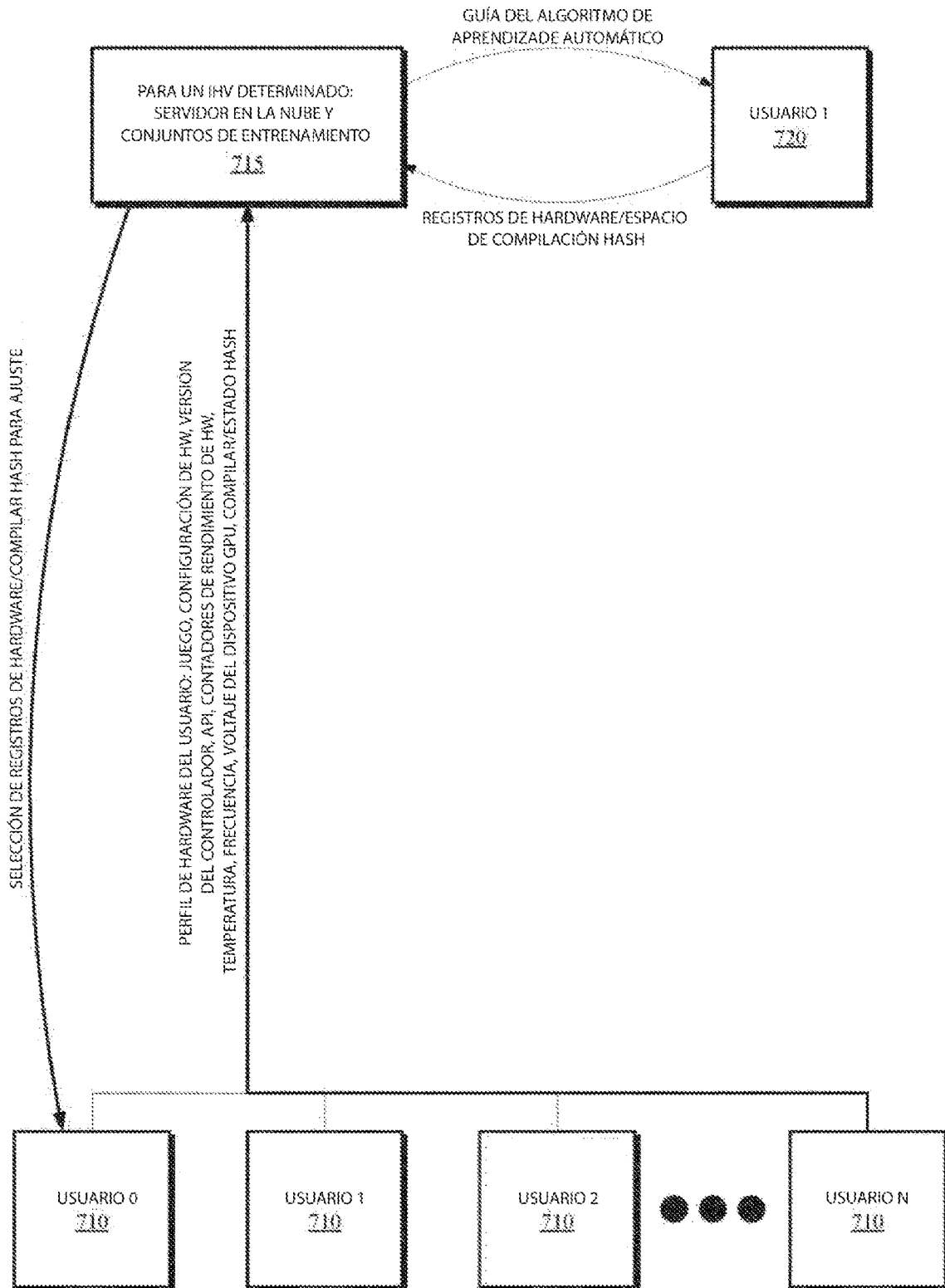


FIG. 7A

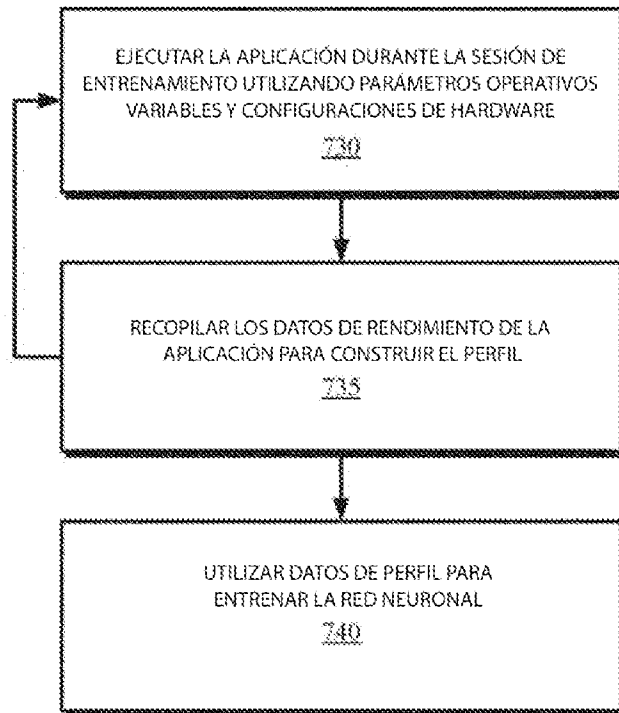


FIG. 7B

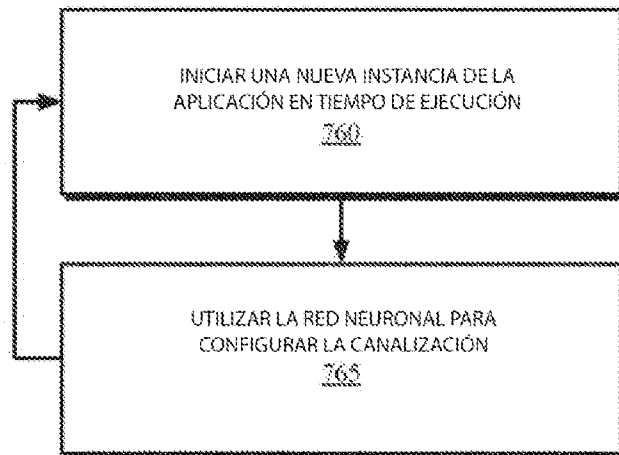


FIG. 7C

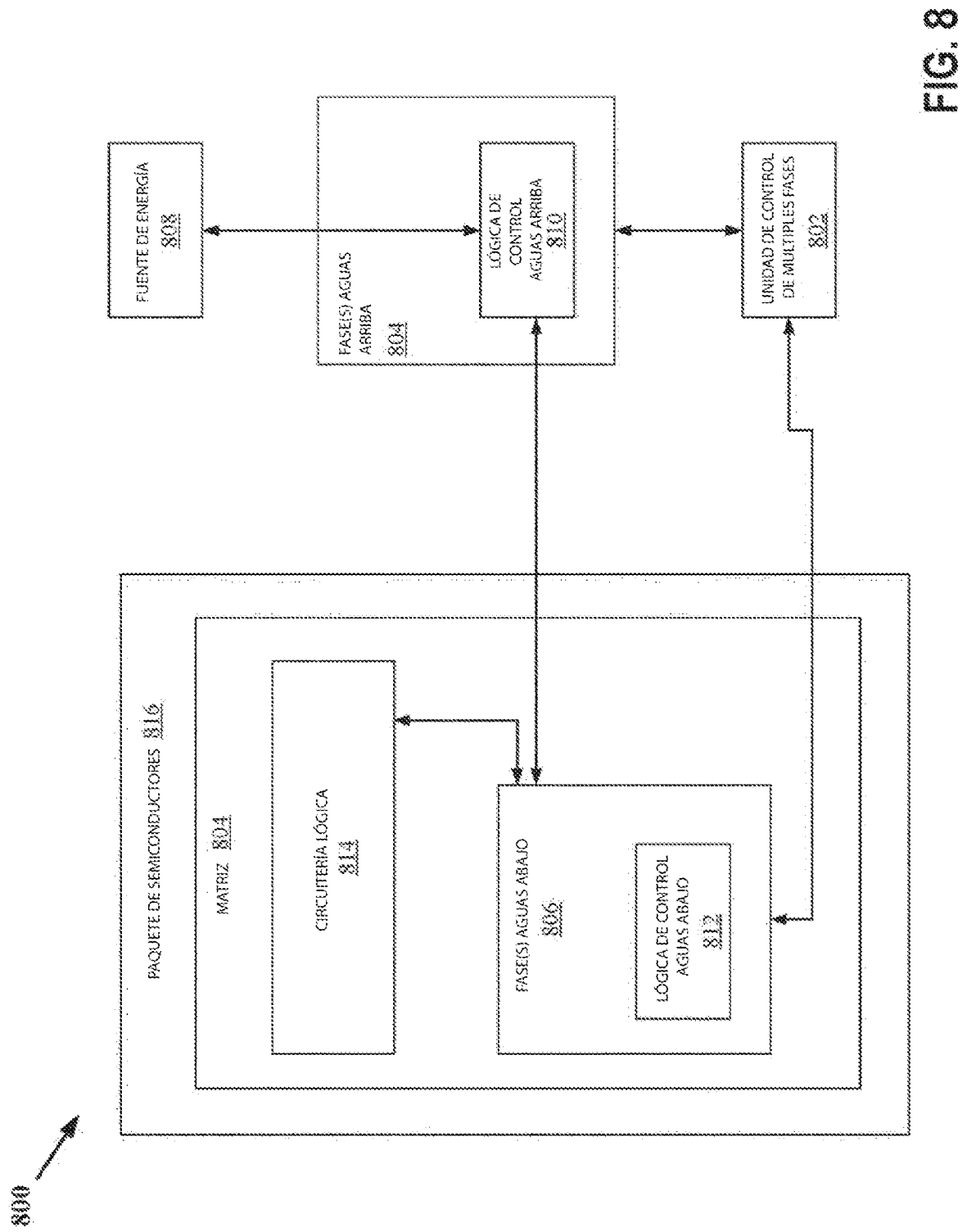


FIG. 8

900

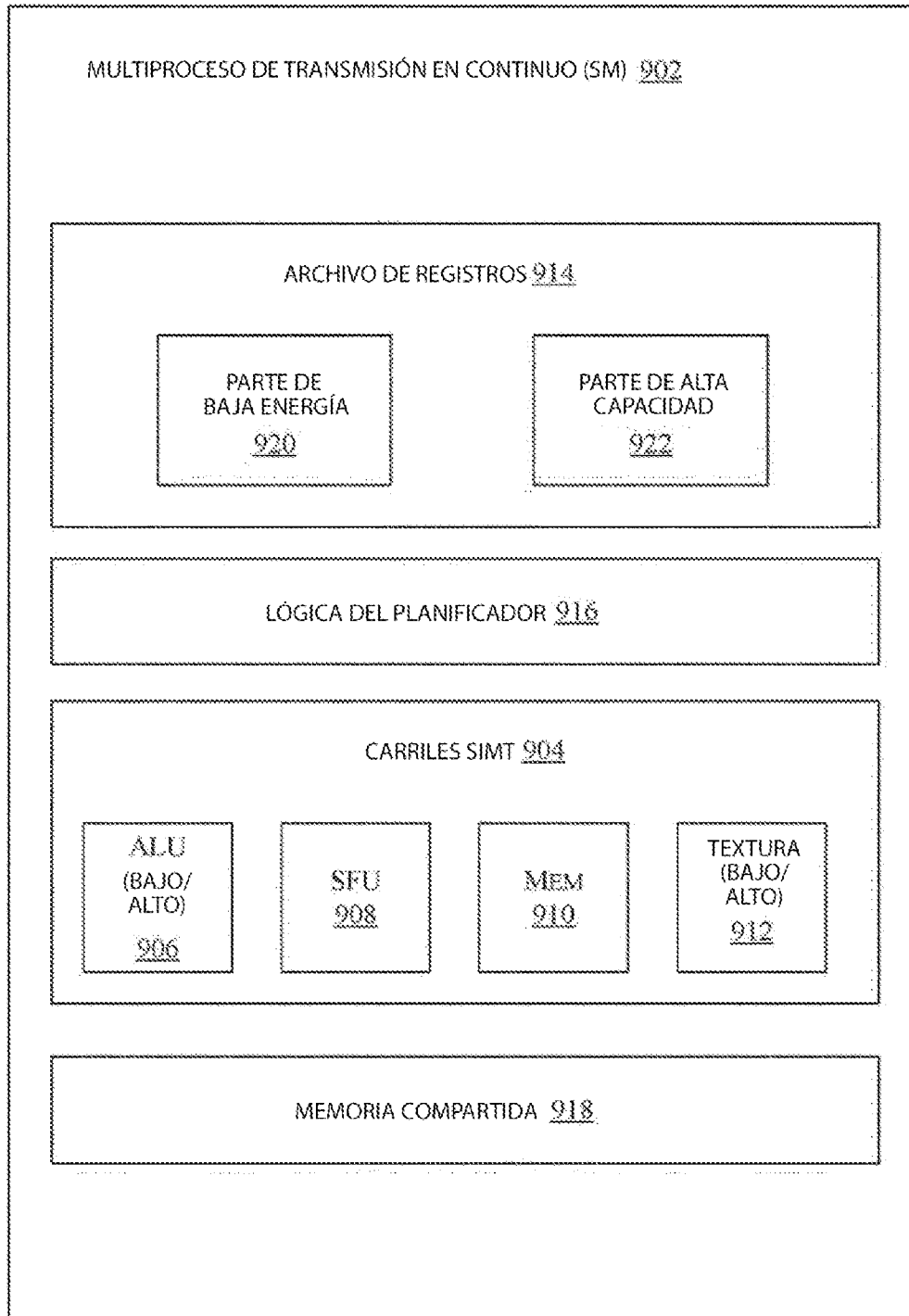
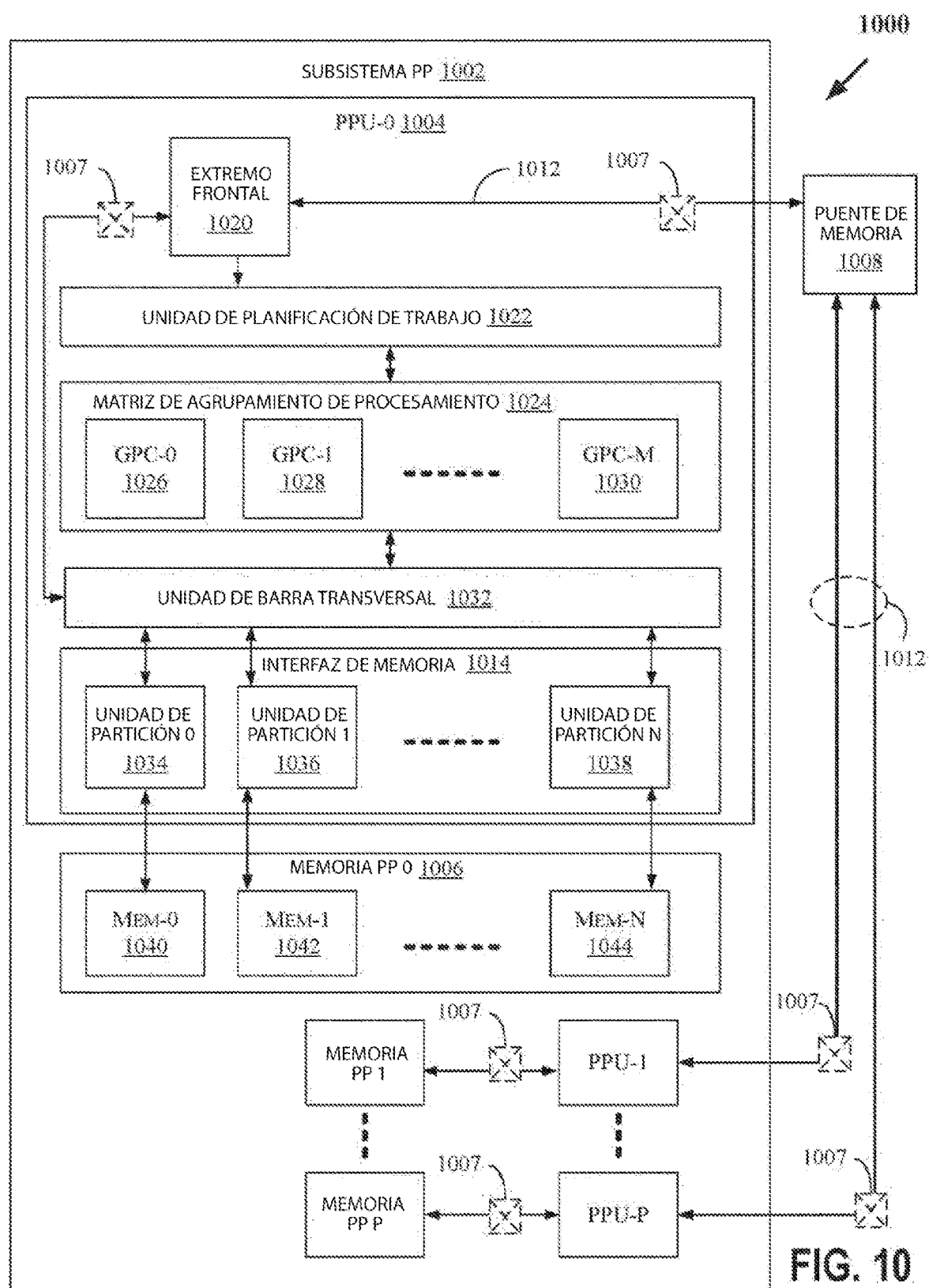


FIG. 9



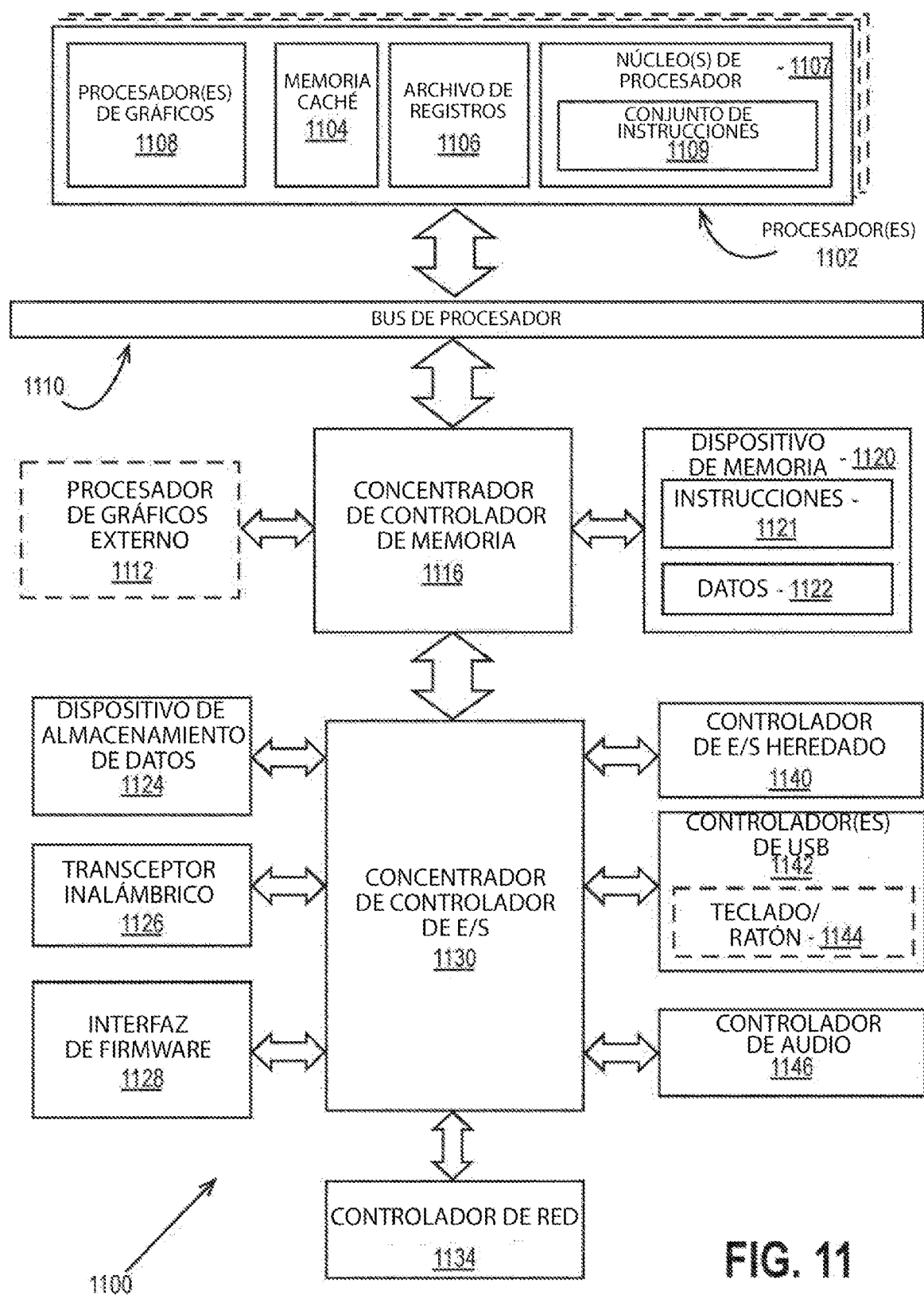


FIG. 11

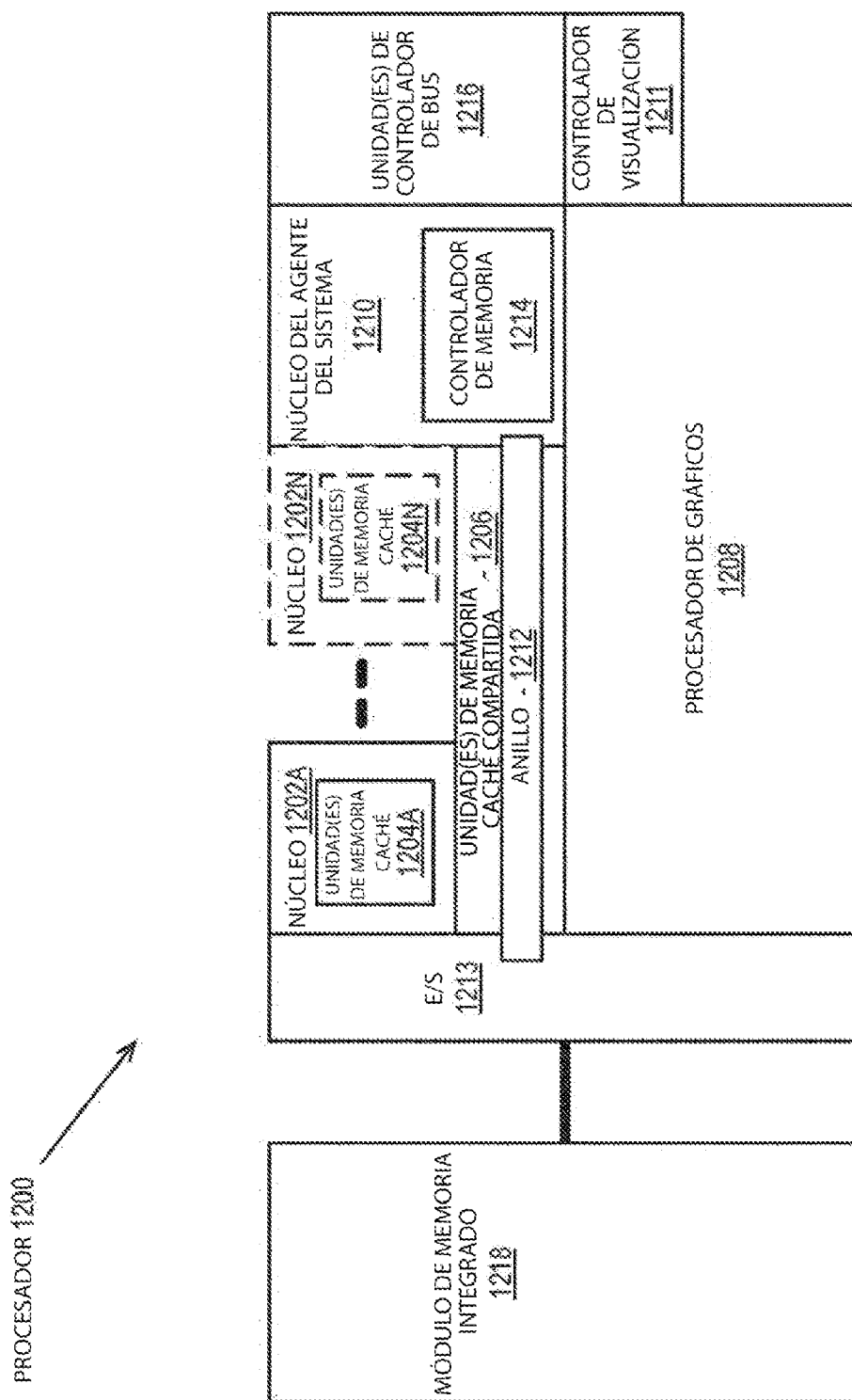


FIG. 12

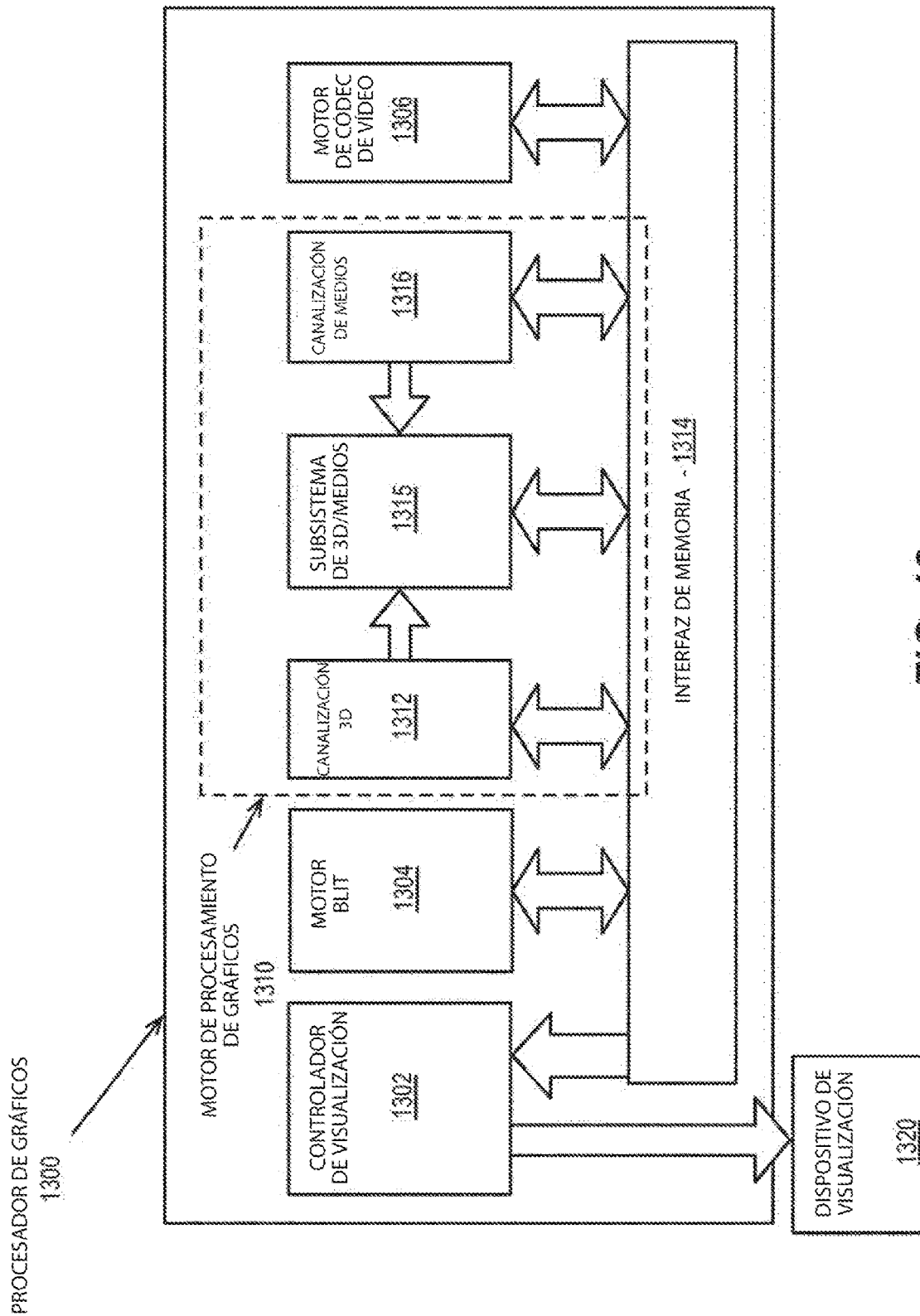
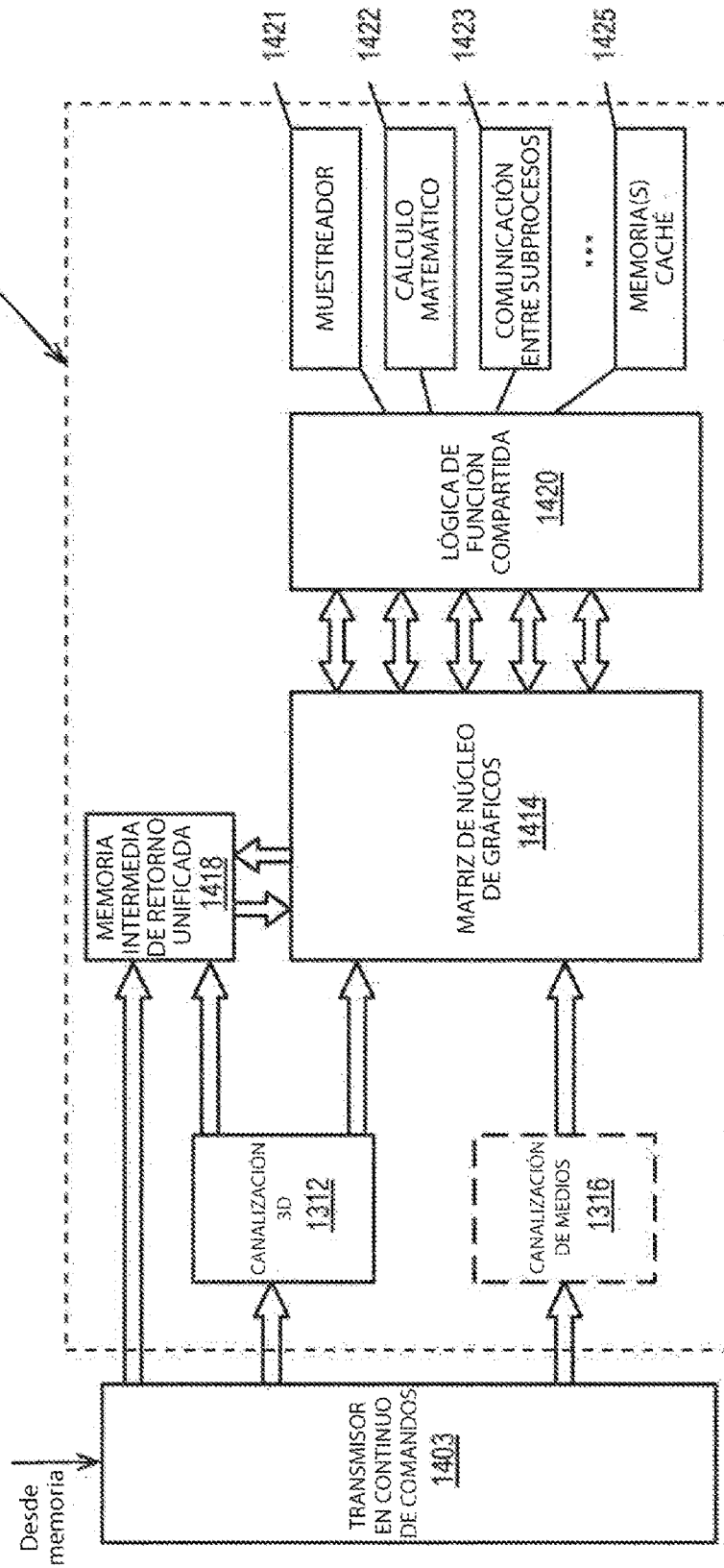


FIG. 13

FIG. 14



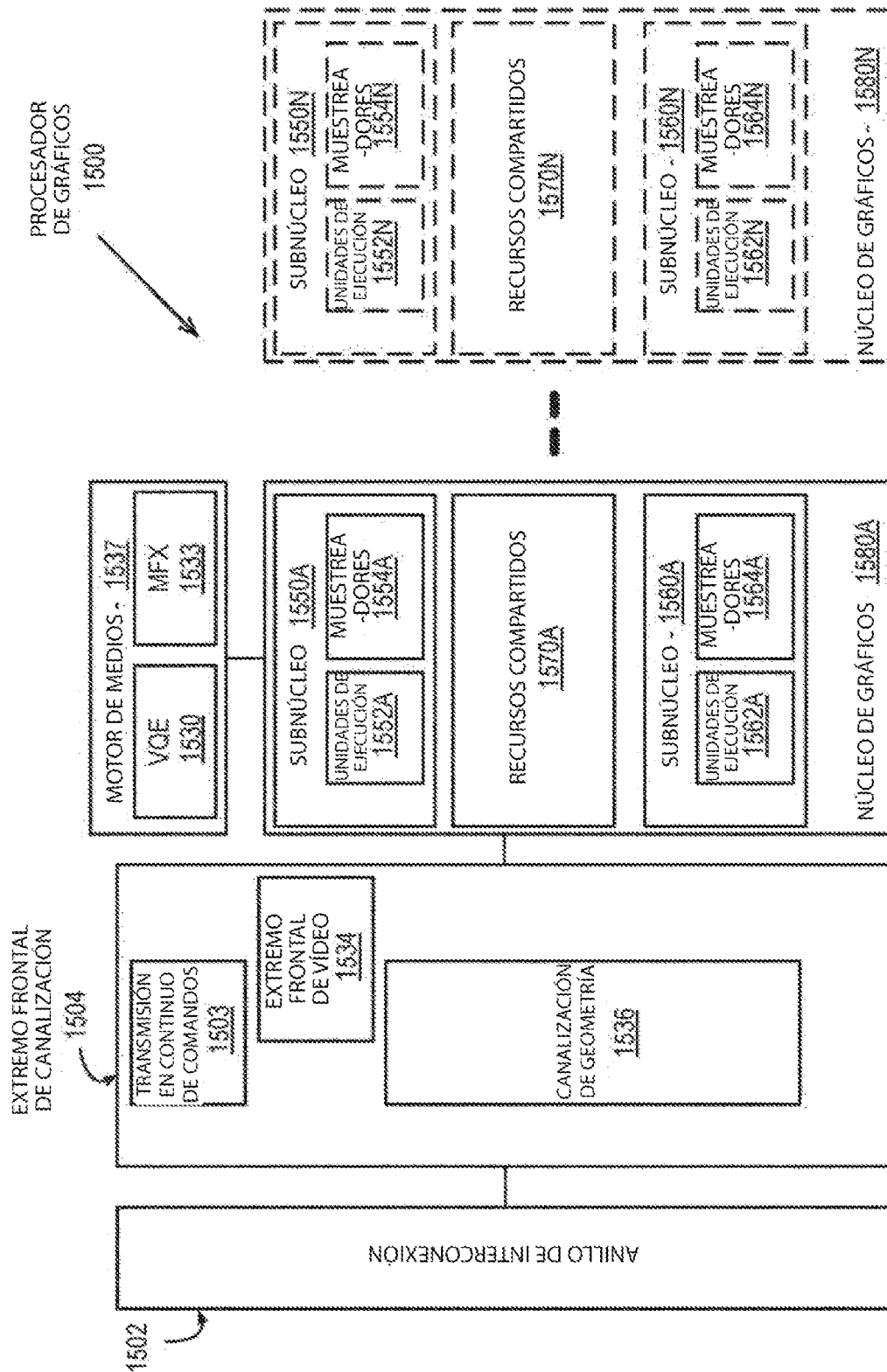


FIG. 15

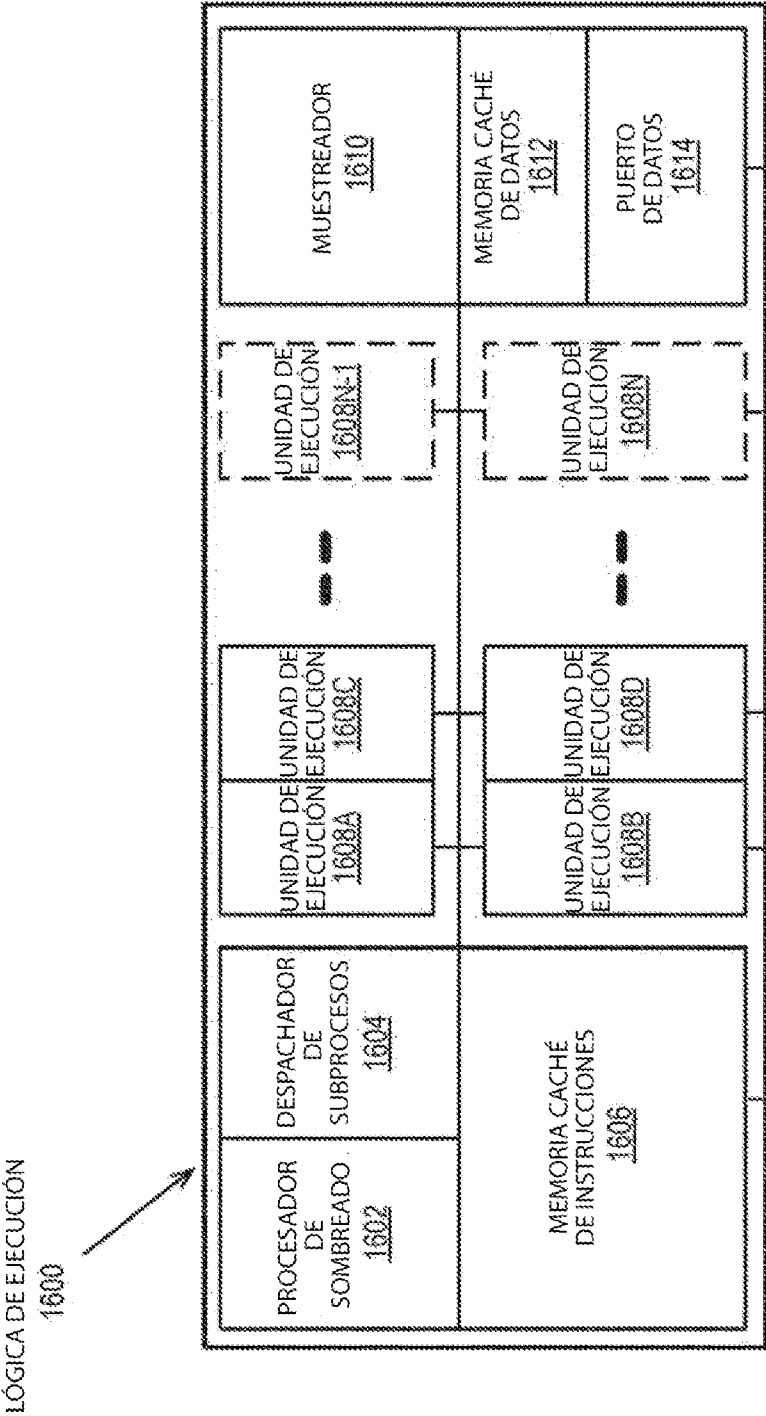


FIG. 16

FORMATOS DE INSTRUCCIONES DEL PROCESADOR DE GRÁFICOS

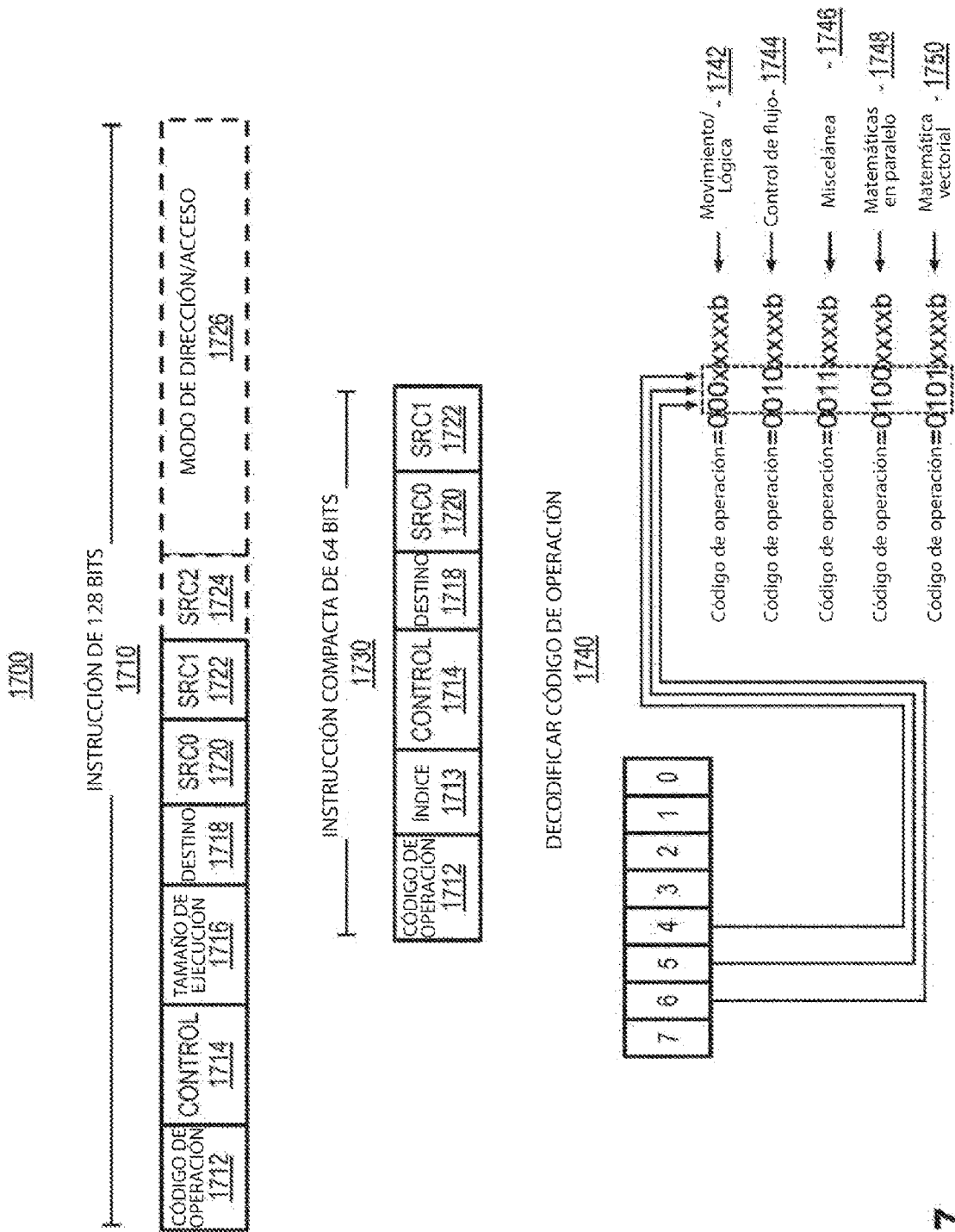


FIG. 17

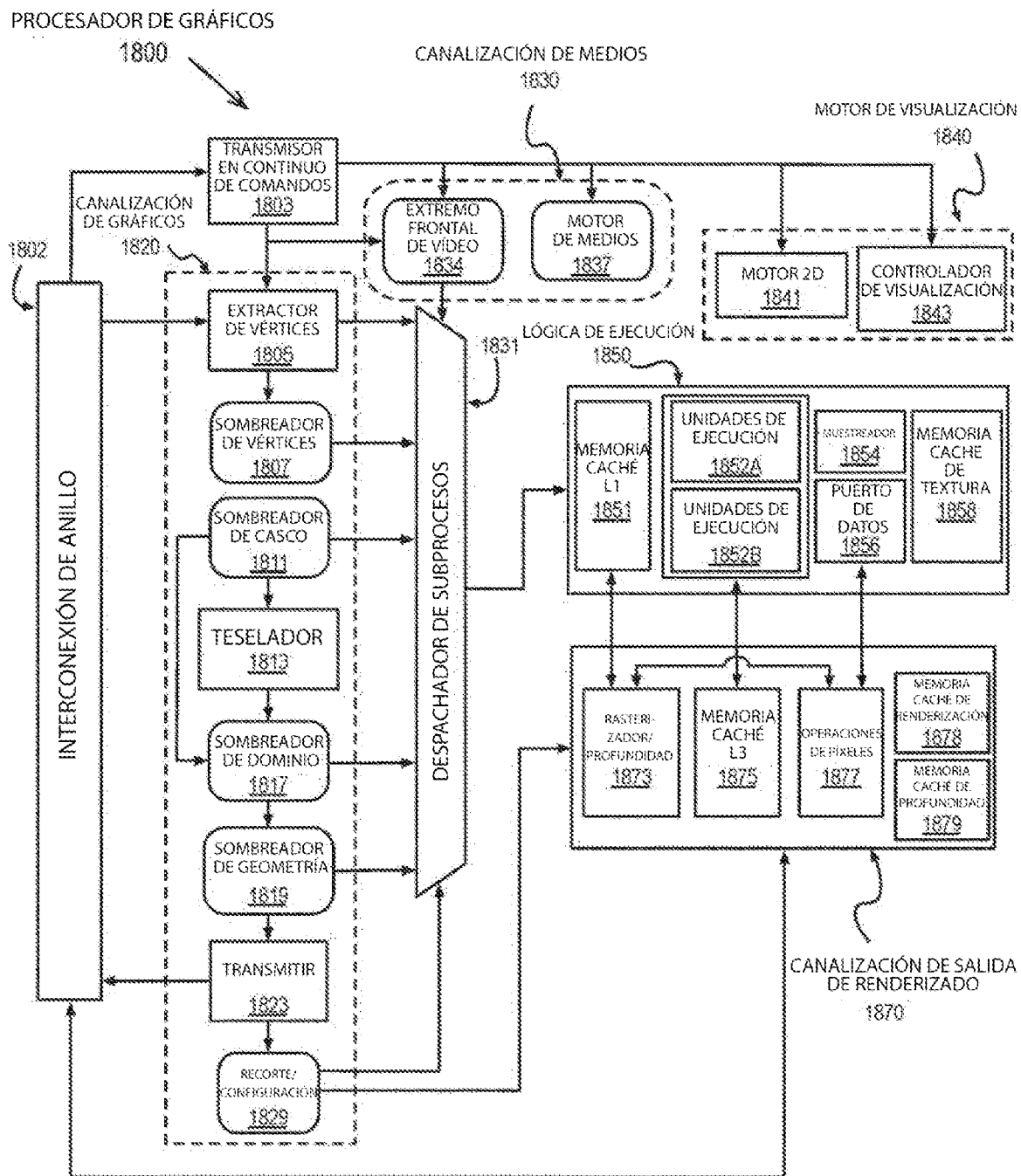


FIG. 18

FIG. 19A FORMATO DE COMANDO DEL PROCESADOR DE GRÁFICOS

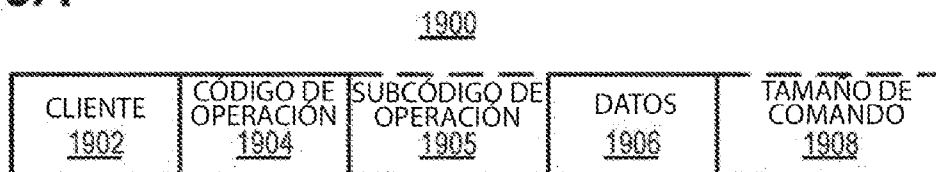
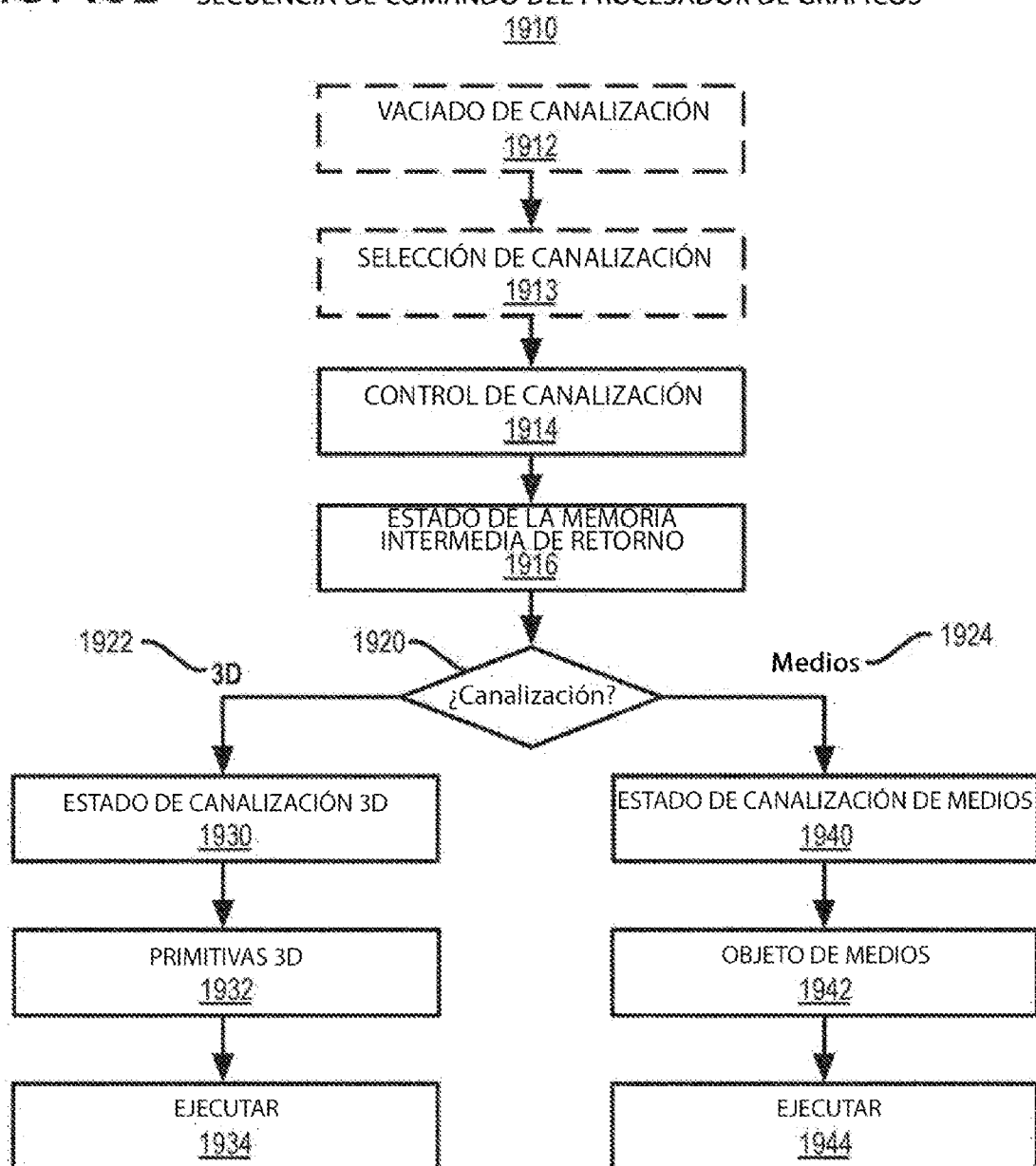


FIG. 19B SECUENCIA DE COMANDO DEL PROCESADOR DE GRÁFICOS



SISTEMA DE PROCESAMIENTO DE DATOS -2000

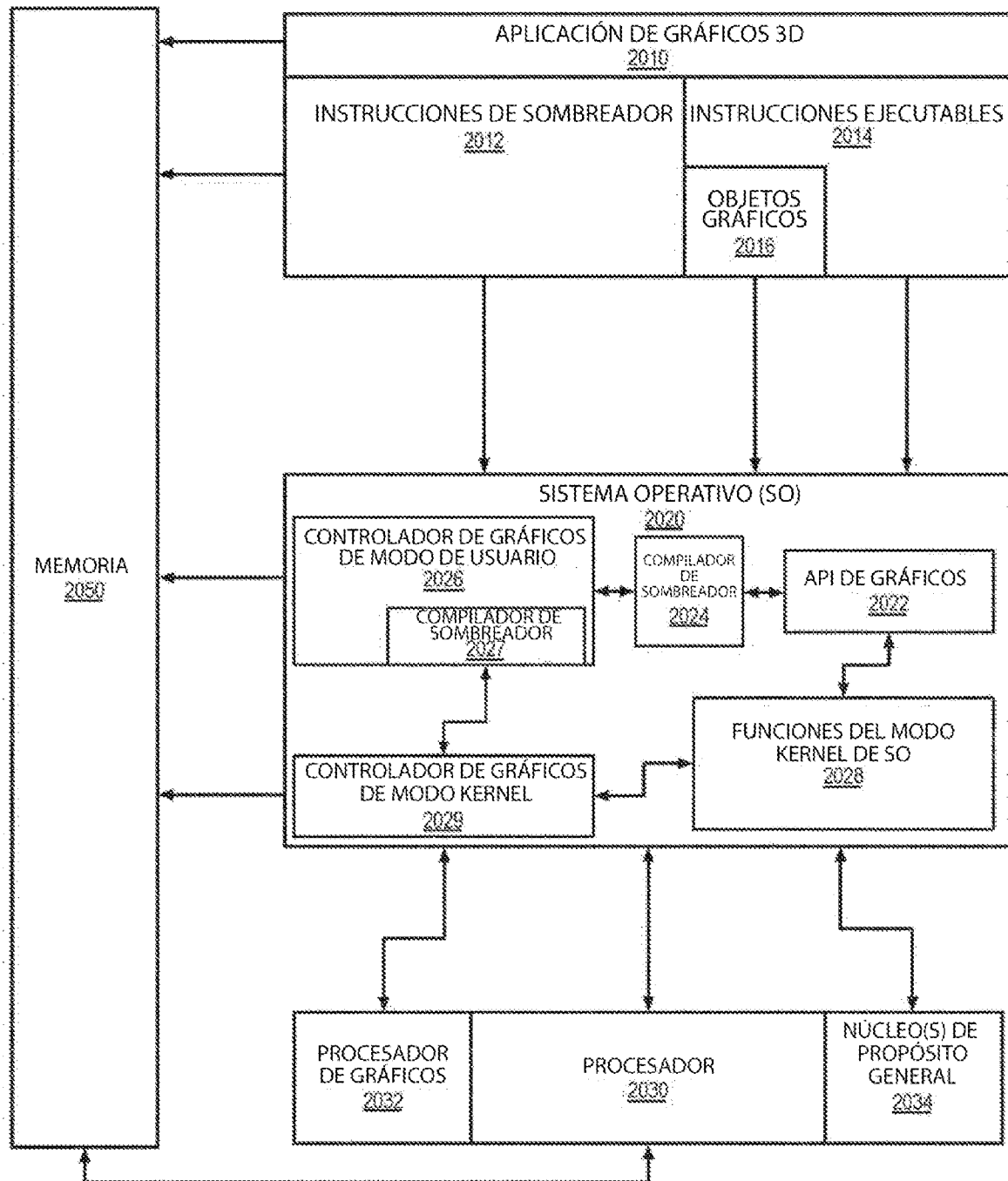


FIG. 20

DESARROLLO DE NÚCLEO IP - 2100

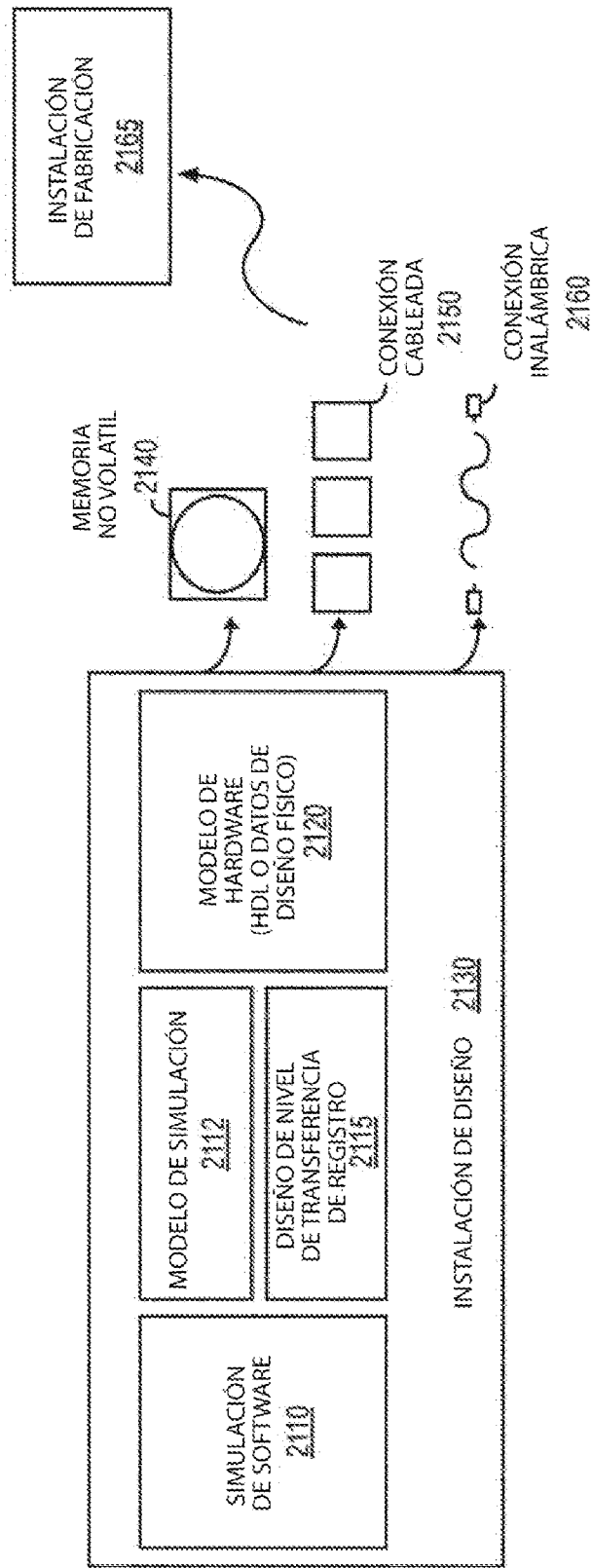


FIG. 21

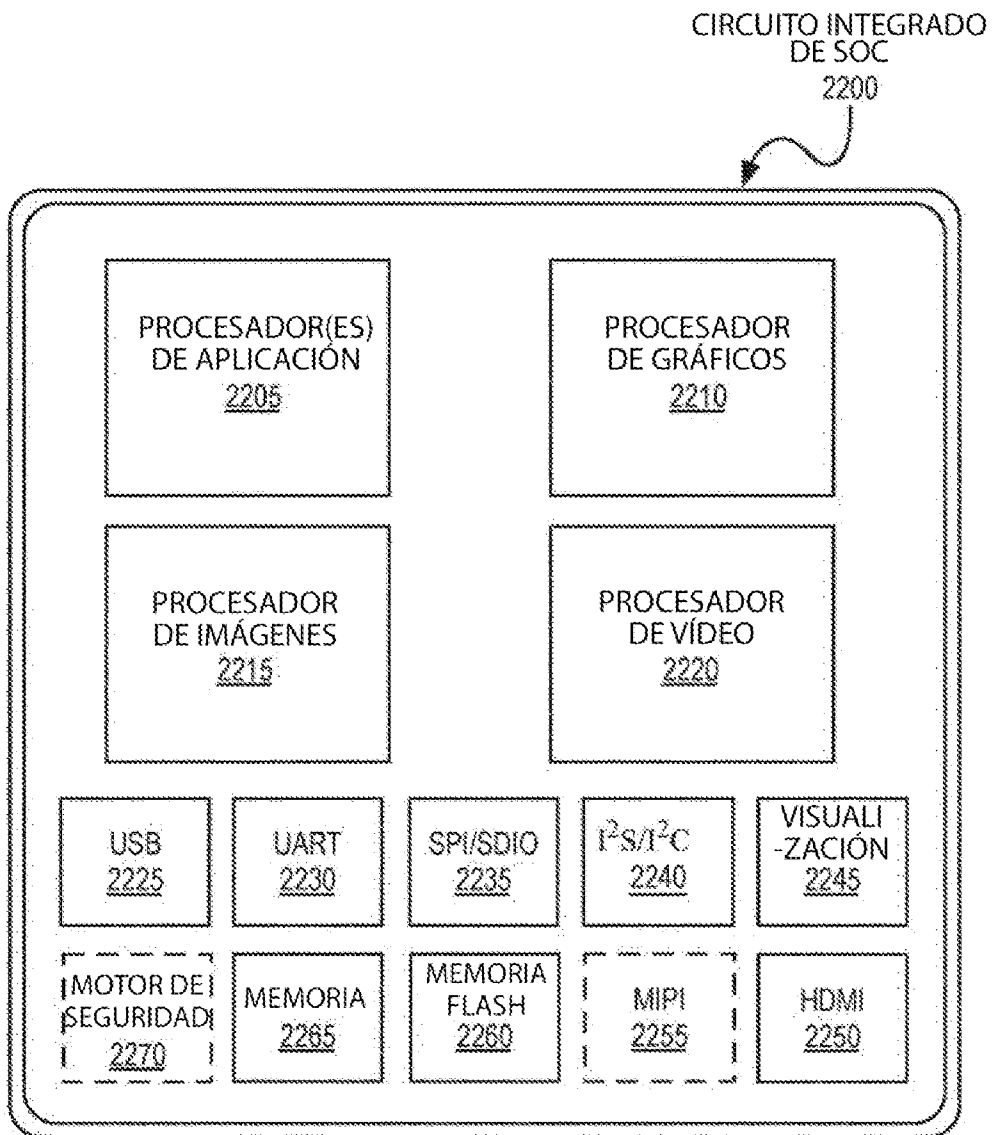


FIG. 22

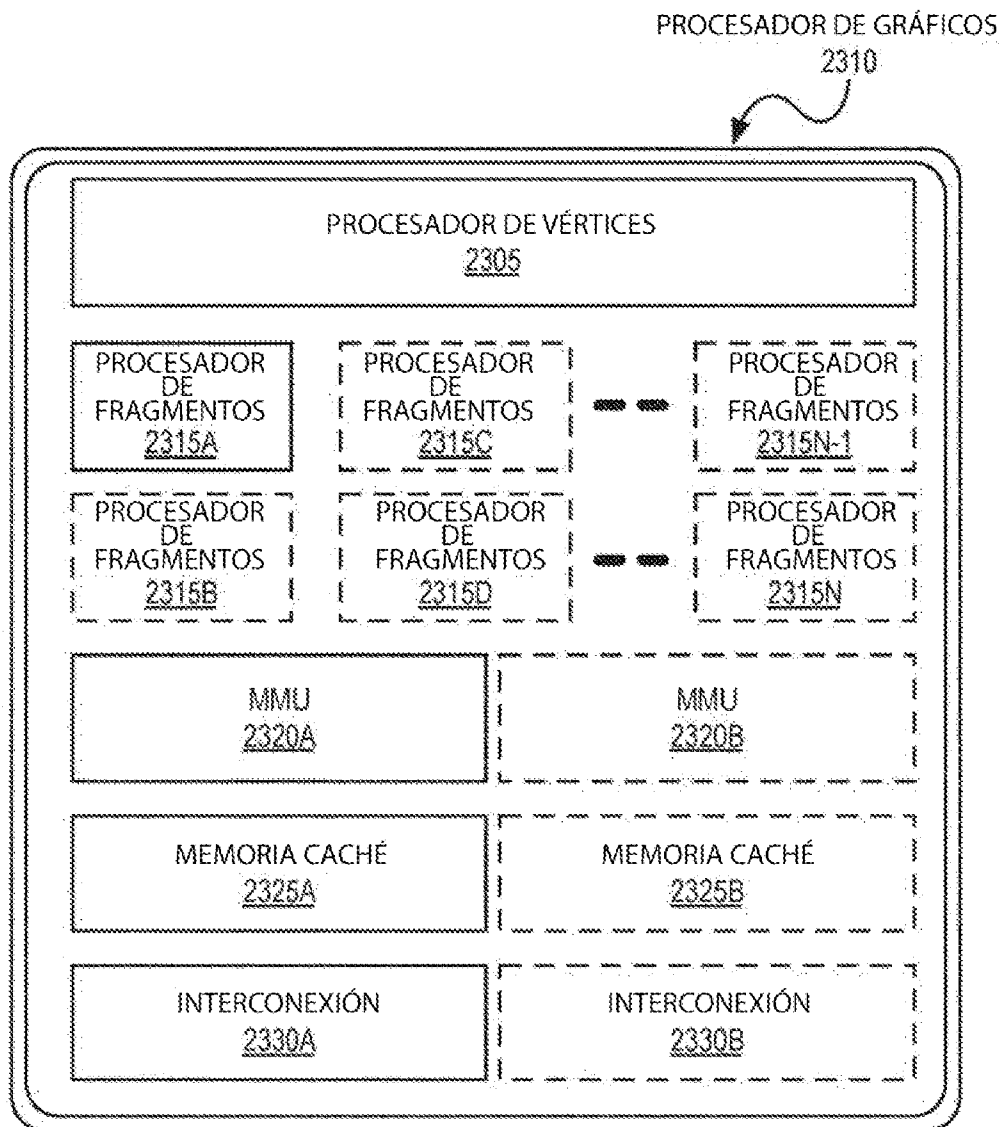


FIG. 23

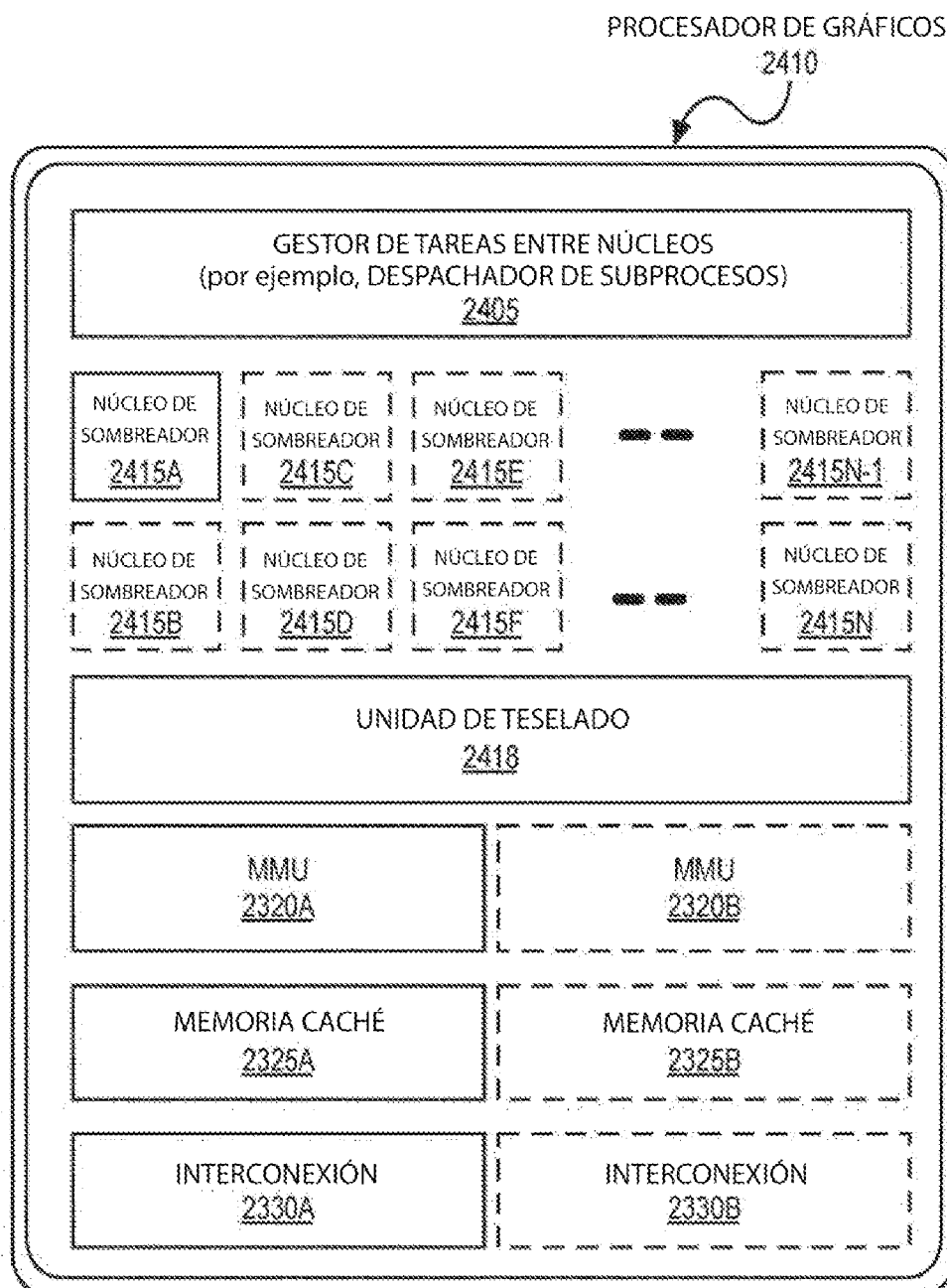


FIG. 24