República Federativa do Brasil
Ministério da Economia
Instituto Nacional da Propriedade Industrial

(21) BR 102020019666-9 A2

(22) Data do Depósito: 26/09/2020

(43) Data da Publicação Nacional: 18/05/2021

(54) Título: APARELHO E MÉTODO PARA FACILITAR OPERAÇÕES DE MULTIPLICAÇÃO DE MATRIZ; ACELERADOR DE HARDWARE

(51) Int. Cl.: G06T 1/20; G06F 9/30.

(52) CPC: G06T 1/20; G06F 9/30.

(30) Prioridade Unionista: 13/11/2019 US 16/682,225.

(71) Depositante(es): INTEL CORPORATION.

(72) Inventor(es): NEVIN MATHEW; ASHUTOSH GARG; SHUBRA MARWAHA.

(57) Resumo: APARELHO E MÉTODO PARA FACILITAR OPERAÇÕES DE MULTIPLICAÇÃO DE MATRIZ; ACELERADOR DE HARDWARE A presente invenção refere-se a um aparelho para facilitar operações de multiplicação de matriz. O aparelho compreende hardware de multiplicação para operar em um modo de produto escalar, em que um estágio de multiplicação incluído no hardware de multiplicação é configurado como um produto escalar de diversos vetores de bit (N) para realizar NxN operações de multiplicação em uma pluralidade de multiplicandos e realizar operações de adição em resultados das NxN operações de multiplicação.

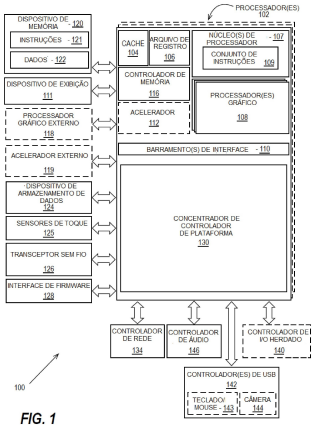


FIG. 1

"APARELHO E MÉTODO PARA FACILITAR OPERAÇÕES DE MULTIPLICAÇÃO DE MATRIZ; ACELERADOR DE HARDWARE"

CAMPO DA INVENÇÃO

[0001] As modalidades se referem, em geral, a processamento de dados e, mais particularmente, a processamento de dados por meio de uma unidade de processamento de gráficos de propósito geral.

HISTÓRICO

[0002] Os algoritmos de aprendizado profundo estão sendo atualmente implantados em diversos aplicativos de aprendizado por máquina, como reconhecimento de áudio/vídeo, resumo de vídeo, etc. As redes neurais de diversas formas (por exemplo, Redes Neurais Convolucionais (CNNs), *Redes Neurais Recorrentes* (*RNNs*), *Memórias Longas de Curto Prazo* (*LSTMs*), etc.) são aplicadas para realizar tais cargas de trabalho devido a sua natureza altamente paralela. Os pedidos de aprendizado por máquina tipicamente implantam cargas de trabalho de multiplicação de matriz por meio de multiplicadores.

BREVE DESCRIÇÃO DOS DESENHOS

[0003] Desse modo, a maneira na qual os recursos citados acima das presentes modalidades podem ser entendidos em detalhes, uma descrição mais particular das modalidades, brevemente resumida acima, pode ser obtida em referência às modalidades, algumas das quais são ilustradas nos desenhos anexos. No entanto, deve ser observado que os desenhos anexos ilustram apenas

modalidades típicas e, portanto, não devem ser considerados limitantes ao seu escopo.

[0004] A **Figura 1** é um diagrama de blocos de um sistema de processamento, de acordo com uma modalidade;

[0005] As **Figuras 2A a 2D** ilustram sistemas de computação e processadores de gráficos fornecidos pelas modalidades descritas no presente documento;

[0006] As **Figuras 3A a 3C** ilustram diagramas de blocos de processador de gráficos adicional e arquiteturas de acelerador de computação fornecidos por modalidades;

[0007] A **Figura 4** é um diagrama de blocos de um mecanismo de processamento de gráficos 410 de um processador de gráficos de acordo com algumas modalidades;

[0008] As **Figuras 5A a 5B** ilustram lógica de execução de encadeamento 500, incluindo uma matriz de elementos de processamento empregada em um núcleo de processador de gráficos de acordo com as modalidades;

[0009] A **Figura 6** ilustra uma unidade de execução adicional 600, de acordo com uma modalidade;

[0010] A **Figura 7** é um diagrama de blocos que ilustra formatos de instrução de processador de gráficos 700 de acordo com algumas modalidades;

[0011] A **Figura 8** é um diagrama de blocos de um processador de gráficos de acordo com outra modalidade;

[0012] As **Figuras 9A e 9B** ilustram um formato de comando de processador de gráficos e sequência de comandos, de acordo com algumas modalidades;

[0013] A **Figura 10** ilustra arquitetura de software de gráficos exemplificativa para um sistema de

processamento de dados de acordo com algumas modalidades;

[0014] As **Figuras 11A a 11D** ilustram um conjunto de pacote de circuito integrado, de acordo com uma modalidade;

[0015] A **Figura 12** é um diagrama de blocos que ilustra um sistema exemplificativo em um circuito integrado de chip, de acordo com uma modalidade;

[0016] As **Figuras 13A e 13B** são um diagrama de blocos que ilustra um processador de gráficos exemplificativo adicional;

[0017] A **Figura 14** ilustra uma pilha de software de aprendizado por máquina, de acordo com uma modalidade;

[0018] As **Figuras 15A e 15B** ilustram camadas de redes neurais profundas exemplificativas;

[0019] A **Figura 16** ilustra uma rede neural recorrente exemplificativa;

[0020] A **Figura 17** ilustra treinamento e implantação de uma rede neural profunda;

[0021] A **Figura 18** é um diagrama de blocos que ilustra aprendizado distribuído;

[0022] A **Figura 19** ilustra uma modalidade de um dispositivo de computação que emprega um acelerador;

[0023] As **Figuras 20A e 20B** ilustram um multiplicador convencional;

[0024] A **Figura 21** ilustra uma modalidade de um multiplicador de produto escalar;

[0025] A **Figura 22** ilustra outra modalidade de um multiplicador de produto escalar;

[0026] A **Figura 23** é um fluxograma que ilustra uma modalidade de um processo para realizar uma operação de multiplicação; e

[0027] A **Figura 24** é um fluxograma que ilustra uma modalidade de um processo para realizar uma operação de multiplicação de produto escalar.

DESCRIÇÃO DETALHADA

[0028] Nas modalidades, um acelerador compreende hardware de multiplicação para operar ou em um modo convencional ou um modo de produto escalar no qual um estágio de multiplicação no hardware de multiplicação é configurado como um produto escalar de diversos N vetores de bit para realizar operações de adição em resultados de NxN operações de multiplicação.

[0029] Na descrição a seguir, inúmeros detalhes específicos são apresentados para fornecer um entendimento mais completo. No entanto, será evidente para um indivíduo versado na técnica que as modalidades descritas no presente documento podem ser praticadas sem um ou mais desses detalhes específicos. Em outros exemplos, recursos bem conhecidos não foram descritos para evitar obscurecer os detalhes das presentes modalidades.

VISÃO GERAL DO SISTEMA

[0030] A **Figura 1** é um diagrama de blocos de um sistema de processamento 100, de acordo com uma modalidade. O sistema 100 pode ser usado em um sistema de área de trabalho de processador único, um sistema de estação de trabalho de múltiplos processadores, ou um sistema de servidor que tem um grande número de

processadores 102 ou núcleos de processador 107. Em uma modalidade, o sistema 100 é uma plataforma de processamento incorporada dentro de um circuito integrado de sistema em um chip (SoC) para uso em dispositivos móveis, portáteis ou embutidos, como dentro de dispositivos de Internet das Coisas (IoT) com conectividade com fio ou sem fio a uma rede de área ampla ou local.

[0031] Em uma modalidade, o sistema 100 pode incluir, se acoplar ou ser integrado a: uma plataforma de jogos baseada em servidor; um console de jogos, incluindo um console de jogos e mídia; um console de jogos móvel, um console de jogos portátil, ou um console de jogos online. Em algumas modalidades, o sistema 100 é parte de um telefone móvel, telefone inteligente, dispositivo de computação do tipo tablet ou dispositivo móvel conectado à Internet, como um computador do tipo laptop com baixa capacidade de armazenamento interno. O sistema de processamento 100 também pode incluir, se acoplar ou ser integrado a: um dispositivo utilizável junto ao corpo, como um dispositivo utilizável junto ao corpo tipo relógio inteligente; roupa ou óculos inteligente aprimorado com recursos de realidade aumentada (AR) ou realidade virtual (VR) para fornecer saídas visuais, de áudio ou táteis para complementar experiências visuais, de áudio ou táteis do mundo real ou, de outra forma, fornecer texto, áudio, gráficos, vídeo, imagens ou vídeo holográfico, ou retroalimentação tátil; outro dispositivo de realidade aumentada (AR); ou outro dispositivo de realidade virtual (VR). Em algumas

modalidades, o sistema de processamento 100 inclui ou é parte de uma televisão ou dispositivo decodificador de sinais. Em uma modalidade, o sistema 100 pode incluir, se acoplar ou ser integrado a um veículo de autocondução como um ônibus, trator-reboque, carro, motor ou ciclo de energia elétrica, avião ou planador (ou qualquer combinação dos mesmos). O veículo de autocondução pode usar o sistema 100 para processar o ambiente detectado ao redor do veículo.

[0032] Em algumas modalidades, o um ou mais processadores 102 cada um, incluem um ou mais núcleos de processador 107 para processar instruções que, quando executadas, realizam operações para software de sistema ou usuário. Em algumas modalidades, pelo menos um dentre o um ou mais núcleos de processador 107 é configurado para processar um conjunto de instruções específicas 109. Em algumas modalidades, o conjunto de instruções 109 pode facilitar Computação de Conjunto de Instruções Complexas (CISC), Computação de Conjunto de Instruções Reduzidas (RISC), ou computação por meio de uma Palavra de Instrução Muito Longa (VLIW). Um ou mais núcleos de processador 107 podem processar um conjunto de instruções 109 diferente, o que pode incluir instruções para facilitar a emulação de outros conjuntos de instruções. O núcleo de processador 107 também pode incluir outros dispositivos de processamento, como um Processador de Sinal Digital (DSP).

[0033] Em algumas modalidades, o processador 102 inclui memória cache 104. Dependendo da arquitetura, o processador 102 pode ter um cache interno único ou

múltiplos níveis de cache interno. Em algumas modalidades, a memória cache é compartilhada entre diversos componentes do processador 102. Em algumas modalidades, o processador 102 também usa um cache externo (por exemplo, um cache Nível 3 (L3) ou Cache de Último Nível (LLC)) (não mostrado), que pode ser compartilhado entre núcleos de processador 107 com o uso de técnicas de coerência de cache conhecidas. Um arquivo de registro 106 pode ser adicionalmente incluído no processador 102 e pode incluir tipos diferentes de registros para armazenar tipos diferentes de dados (por exemplo, registros inteiros, registros de ponto flutuante, registros de situação e um registro de indicador de instrução). Alguns registros podem ser registros de propósito geral, enquanto outros registros podem ser específicos para o projeto do processador 102.

[0034] Em algumas modalidades, um ou mais processadores 102 são acoplados a um ou mais barramentos de interface 110 para transmitir sinais de comunicação como endereço, dados, ou sinais de controle entre processador 102 e outros componentes no sistema 100. O barramento de interface 110, em uma modalidade, pode ser um barramento de processador, como uma versão do barramento de Interface de Mídia Direta (DMI). No entanto, barramentos de processador não são limitados ao barramento de DMI, e podem incluir um ou mais barramentos de Interconexão de Componente Periférico (por exemplo, PCI, PCI expressa), barramentos de memória, ou outros tipos de barramentos de interface. Em uma modalidade, o processador (ou processadores) 102

incluem um controlador de memória integrada 116 e um concentrador de controlador de plataforma 130. O controlador de memória 116 facilita comunicação entre um dispositivo de memória e outros componentes do sistema 100, enquanto o concentrador de controlador de plataforma (PCH) 130 fornece conexões para dispositivos de I/O por meio de um barramento de I/O local.

[0035] O dispositivo de memória 120 pode ser um dispositivo de memória de acesso aleatório dinâmico (DRAM), um dispositivo de memória de acesso aleatório estático (SRAM), dispositivo de memória flash, dispositivo de memória de alteração de fase, ou algum outro dispositivo de memória que tem desempenho adequado para servir como memória de processo. Em uma modalidade, o dispositivo de memória 120 pode operar como memória de sistema para o sistema 100, para armazenar dados 122 e instruções 121 para uso quando o um ou mais processadores 102 executam um aplicativo ou processo. O controlador de memória 116 também se acopla a um processador de gráficos externo opcional 118, que pode se comunicar com o um ou mais processadores de gráficos 108 em processadores 102 para executar operações de gráfico e mídia. Em algumas modalidades, operações de gráfico, mídia ou computação podem ser auxiliadas por um acelerador 112 que é um coprocessador que pode ser configurado para executar um conjunto especializado de operações de gráficos, mídia ou computação. Por exemplo, em uma modalidade, o acelerador 112 é um acelerador de multiplicação de matriz usado para otimizar aprendizado por máquina ou operações de computação. Em uma

modalidade, o acelerador 112 é um acelerador de traçado de raios que pode ser usado para executar operações de traçado de raios em combinação com o processador de gráficos 108. Em uma modalidade, um acelerador externo 119 pode ser usado no lugar de ou em combinação com o acelerador 112.

[0036] Em algumas modalidades, um dispositivo de exibição 111 pode se conectar ao processador (ou processadores) 102. O dispositivo de exibição 111 pode ser um ou mais dentre um dispositivo de exibição interno, como em um dispositivo eletrônico móvel ou um dispositivo do tipo laptop ou um dispositivo de exibição externo afixado por meio de uma interface de exibição (por exemplo, DisplayPort, etc.). Em uma modalidade, o dispositivo de exibição 111 pode ser um visor montado de cabeça (HMD), como um dispositivo de exibição estereoscópico para uso em aplicativos de realidade virtual (VR) ou aplicativos de realidade aumentada (AR).

[0037] Em algumas modalidades, o concentrador de controlador de plataforma 130 possibilita que periféricos se conectem ao dispositivo de memória 120 e ao processador 102 por meio de um barramento de I/O de alta velocidade. Os periféricos de I/O incluem, mas não são limitados a, um controlador de áudio 146, um controlador de rede 134, uma interface de firmware 128, um transceptor sem fio 126, sensores sensíveis ao toque 125, um dispositivo de armazenamento de dados 124 (por exemplo, memória não volátil, memória volátil, unidade de disco rígido, memória flash, NAND, NAND 3D, Xpoint 3D, etc.). O dispositivo de armazenamento de dados 124

pode se conectar por meio de uma interface de armazenamento (por exemplo, SATA) ou por meio de um barramento periférico, como um barramento de Interconexão de Componente Periférico (por exemplo, PCI, PCI expressa). Os sensores sensíveis ao toque 125 podem incluir sensores de tela sensível ao toque, sensores de pressão ou sensores de impressão digital. O transceptor sem fio 126 pode ser um transceptor Wi-Fi, um transceptor Bluetooth, ou um transceptor de rede móvel como um transceptor 3G, 4G ou 5G de Evolução a Longo Prazo (LTE). A interface de firmware 128 possibilita comunicação com firmware de sistema, e pode ser, por exemplo, uma interface de firmware extensível unificada (UEFI). O controlador de rede 134 pode possibilitar uma conexão de rede a uma rede com fio. Em algumas modalidades, um controlador de rede de alto desempenho (não mostrado) se acopla ao barramento de interface 110. O controlador de áudio 146, em uma modalidade, é um controlador de áudio de alta definição de múltiplos canais. Em uma modalidade, o sistema 100 inclui um controlador de I/O herdado opcional 140 para acoplar dispositivos herdados (por exemplo, sistema pessoal 2 (PS/2)) ao sistema. O concentrador de controlador de plataforma 130 também pode se conectar a um ou mais dispositivos de entrada de conexão de controladores de Barramento em Série Universal (USB) 142, como combinações de teclado e mouse 143, uma câmera 144, ou outros dispositivos de entrada de USB.

[0038] Será observado que o sistema 100 mostrado é exemplificativo e não limitante, uma vez que outros

tipos de sistemas de processamento de dados que são configurados de modo diferente também podem ser usados. Por exemplo, um exemplo do controlador de memória 116 e concentrador de controlador de plataforma 130 pode ser integrado a um processador de gráficos externo distinto, como o processador de gráficos externo 118. Em uma modalidade, o concentrador de controlador de plataforma 130 e/ou controlador de memória 116 pode ser externo ao um ou mais processadores 102. Por exemplo, o sistema 100 pode incluir um controlador de memória externo 116 e concentrador de controlador de plataforma 130, que pode ser configurado como um concentrador de controlador de memória e concentrador de controlador periférico dentro de um conjunto de chips de sistema que está em comunicação com o processador (ou processadores) 102.

[0039] Por exemplo, as placas de circuito (“deslizadores”) podem ser usadas nesses componentes, como CPUs, memória e outros componentes são projetados para desempenho térmico aumentado. Em alguns exemplos, componentes de processamento, como os processadores, estão localizados em um lado superior de um deslizador, enquanto memória próxima, como DIMMs, está localizada em um lado inferior do deslizador. Como resultado do fluxo de ar aprimorado fornecido por esse projeto, os componentes podem operar em frequências e níveis de potência superiores aos de sistemas típicos, aumentando, desse modo, o desempenho. Adicionalmente, os deslizadores são configurados para se encaixar cegamente nos cabos de força e comunicação de dados em um rack, aprimorando, desse modo, sua habilidade em ser

rapidamente removido, atualizado, reinstalado e/ou substituído. De maneira similar, componentes individuais localizados nos deslizadores, como processadores, aceleradores, memória e unidades de armazenamento de dados, são configurados para serem facilmente atualizados devido a seu espaçamento aumentado um do outro. Na modalidade ilustrativa, os componentes incluem adicionalmente recursos de certificação de hardware para provar sua autenticidade.

[0040] Um centro de dados pode utilizar uma arquitetura de rede única (“malha”) que sustenta múltiplas arquiteturas de rede diferentes que incluem Ethernet e Omni-Path. Os deslizadores podem ser acoplados a comutadores por meio de fibras ópticas, que fornecem largura de banda superior e latência inferior ao cabeamento de par torcido típico (por exemplo, Categoria 5, Categoria 5e, Categoria 6, etc.). Devido à alta largura de banda, interconexões de baixa latência e arquitetura de rede, o centro de dados pode, durante o uso, agrupar recursos, tal como memória, aceleradores (por exemplo, GPUs, aceleradores de gráficos, FPGAs, ASICs, rede neural e/ou aceleradores de inteligência artificial, etc.), e unidades de armazenamento de dados que são fisicamente desagregadas, e fornecer os mesmos para computar recursos (por exemplo, processadores) conforme necessário, o que permite que os recursos de computação acessem os recursos agrupados como se fossem locais.

[0041] Uma fonte de alimentação ou abastecimento de energia pode fornecer tensão e/ou corrente ao sistema

100 ou qualquer componente ou sistema descrito no presente documento. Em um exemplo, a fonte de alimentação inclui um adaptador de CA para CC (corrente alternada para corrente contínua) para se ligar a uma saída de parede. Tal potência de CA pode ser fonte de alimentação de energia renovável (por exemplo, energia solar). Em um exemplo, a fonte de alimentação inclui uma fonte de alimentação de CC, como um conversor externo de CA para CC. Em um exemplo, abastecimento de energia ou fonte de alimentação inclui hardware de carregamento sem fio para carregamento por meio de proximidade a um campo de carregamento. Em um exemplo, a fonte de alimentação pode incluir uma bateria interna, fornecimento de corrente alternada, abastecimento de energia com base em movimento, abastecimento de energia solar ou fonte de célula de combustível.

[0042] As **Figuras 2A a 2D** ilustram sistemas de computação e processadores de gráficos fornecidos pelas modalidades descritas no presente documento. Os elementos das Figuras 2A a 2D que têm os mesmos números de referência (ou nomes) que os elementos de qualquer outra Figura no presente documento podem operar ou funcionar de qualquer maneira similar àquela descrita em outro lugar no presente documento, porém, sem limitação à mesma.

[0043] A **Figura 2A** é um diagrama de blocos de uma modalidade de um processador 200 que tem um ou mais núcleos de processador 202A-202N, um controlador de memória integrado 214 e um processador de gráficos integrado 208. O processador 200 pode incluir núcleos

adicionais até e que incluem núcleo adicional 202N representado pelas caixas de linha tracejada. Cada um dos núcleos de processador 202A-202N inclui uma ou mais unidades de cache interno 204A-204N. Em algumas modalidades, cada núcleo de processador também tem acesso a uma ou mais unidades de cache compartilhado 206. As unidades de cache interno 204A-204N e unidades de cache compartilhado 206 representam uma hierarquia de memória cache dentro do processador 200. A hierarquia de memória cache pode incluir pelo menos um nível de instrução e cache de dados dentro de cada núcleo de processador e um ou mais níveis de cache de nível intermediário compartilhado, como um Nível 2 (L2), Nível 3 (L3), Nível 4 (L4), ou outros níveis de cache, em que o nível mais alto de cache antes da memória externa é classificado como o LLC. Em algumas modalidades, lógica de coerência de cache mantém coerência entre as diversas unidades de cache 206 e 204A-204N.

[0044] Em algumas modalidades, o processador 200 também pode incluir um conjunto de uma ou mais unidades de controle de barramento 216 e um núcleo de agente de sistema 210. A um ou mais unidades de controle de barramento 216 gerenciam um conjunto de barramentos periféricos, como um ou mais barramentos de PCI ou PCI expressa. O núcleo de agente de sistema 210 fornece gerenciamento de funcionalidade para os vários componentes de processador. Em algumas modalidades, o núcleo de agente de sistema 210 inclui um ou mais controladores de memória integrada 214 para gerenciar

acesso a diversos dispositivos de memória externa (não mostrados).

[0045] Em algumas modalidades, um ou mais dos núcleos de processador 202A-202N incluem suporte para múltiplos encadeamentos simultâneos. Em tal modalidade, o núcleo de agente de sistema 210 inclui componentes para coordenar e operar núcleos 202A-202N durante processamento de múltiplos encadeamentos. O núcleo de agente de sistema 210 pode incluir adicionalmente uma unidade de controle de potência (PCU), que inclui lógica e componentes para regular o estado de potência de núcleos de processador 202A-202N e processador de gráficos 208.

[0046] Em algumas modalidades, o processador 200 inclui adicionalmente processador de gráficos 208 para executar operações de processamento de gráficos. Em algumas modalidades, o processador de gráficos 208 se acopla ao conjunto de unidades de cache compartilhado 206, e ao núcleo de agente de sistema 210, incluindo o um ou mais controladores de memória integrada 214. Em algumas modalidades, o núcleo de agente de sistema 210 também inclui um controlador de exibição 211 para acionar saída de processador de gráficos a um ou mais visores acoplados. Em algumas modalidades, o controlador de exibição 211 também pode ser um módulo separado acoplado ao processador de gráficos por meio de pelo menos uma interconexão, ou pode ser integrado dentro do processador de gráficos 208.

[0047] Em algumas modalidades, uma unidade de interconexão baseada em anel 212 é usada para acoplar os

componentes internos do processador 200. No entanto, uma unidade de interconexão alternativa pode ser usada, como uma interconexão de ponto a ponto, uma interconexão comutada ou outras técnicas, que incluem técnicas bem conhecidas na arte. Em algumas modalidades, o processador de gráficos 208 se acopla à interconexão de anel 212 por meio de um enlace de I/O 213.

[0048] O enlace de I/O exemplificativo 213 representa pelo menos uma dentre múltiplas variedades de interconexões de I/O, incluindo uma interconexão de I/O em pacote, o que facilita comunicação entre diversos componentes de processador e um módulo de memória embutida de alto desempenho 218, como um módulo de eDRAM. Em algumas modalidades, cada um dos núcleos de processador 202A–202N e processador de gráficos 208 pode usar módulos de memória embutida 218 como um Cache de Último Nível compartilhado.

[0049] Em algumas modalidades, núcleos de processador 202A–202N são núcleos homogêneos que executam o mesmo conjunto de instruções arquitetura. Em outra modalidade, núcleos de processador 202A–202N são heterogêneos em termos de conjunto de instruções arquitetura (ISA), em que um ou mais de núcleos de processador 202A–202N executam um primeiro conjunto de instruções, enquanto pelo menos um dos outros núcleos executa um subconjunto do primeiro conjunto de instruções ou um conjunto de instruções diferente. Em uma modalidade, núcleos de processador 202A–202N são heterogêneos em termos de microarquitetura, em que um ou mais núcleos que têm um consumo de potência relativamente maior se acoplam a um

ou mais núcleos de potência que têm um consumo de potência inferior. Em uma modalidade, núcleos de processador 202A-202N são heterogêneos em termos de capacidade computacional. Adicionalmente, o processador 200 pode ser implantado em um ou mais chips ou como um circuito integrado de SoC que tem os componentes ilustrados, além de outros componentes.

[0050] A **Figura 2B** é um diagrama de blocos de lógica de hardware de um núcleo de processador de gráficos 219, de acordo com algumas modalidades descritas no presente documento. Elementos da Figura 2B que têm os mesmos números de referência (ou nomes) que os elementos de qualquer outra Figura no presente documento podem operar ou funcionar de qualquer maneira similar àquela descrita em outro lugar no presente documento, porém, sem limitação a tal. O núcleo de processador de gráficos 219, algumas vezes, chamado de uma fatia de núcleo, pode ser um ou múltiplos núcleos de gráficos dentro de um processador de gráficos modular. O núcleo de processador de gráficos 219 é exemplificativo de uma fatia de núcleo de gráficos, e um processador de gráficos, como descrito no presente documento, pode incluir múltiplas fatias de núcleo de gráficos com base em envelopes-alvo de potência e desempenho. Cada núcleo de processador de gráficos 219 pode incluir um bloco de função fixo 230 acoplado a múltiplos subnúcleos 221A-221F, também chamados de subfatias, que incluem blocos modulares de lógica de função fixa e propósito geral.

[0051] Em algumas modalidades, o bloco de função fixo 230 inclui uma geometria/pipeline de função fixa 231 que

pode ser compartilhada por todos os subnúcleos no núcleo de processador de gráficos 219, por exemplo, em implantações de processador de gráficos de desempenho inferior e/ou potência inferior. Em diversas modalidades, a geometria/pipeline de função fixa 231 inclui um pipeline 3D de função fixa (por exemplo, pipeline 3D 312 como na **Figura 3** e na **Figura 4**, descritas abaixo), uma unidade de front-end de vídeo, um gerador de encadeamento e despachante de encadeamento e um gerenciador de armazenamento temporário de retorno unificado, que gerencia armazenamentos temporários de retorno unificados (por exemplo, armazenamento temporário de retorno unificado 418 na **Figura 4**, como descrito abaixo).

[0052] Em uma modalidade, o bloco de função fixa 230 também inclui uma interface de SoC de gráficos 232, um microcontrolador de gráficos 233, e um pipeline de mídia 234. A interface de SoC de gráficos 232 fornece uma interface entre o núcleo de processador de gráficos 219 e outros núcleos de processador dentro de um sistema em um circuito integrado de chip. O microcontrolador de gráficos 233 é um subprocessador programável que é configurável para gerenciar diversas funções do núcleo de processador de gráficos 219, incluindo envio de encadeamento, programação e preferência. O pipeline de mídia 234 (por exemplo, pipeline de mídia 316 da **Figura 3** e da **Figura 4**) inclui lógica para facilitar a decodificação, codificação, pré-processamento e/ou pós-processamento de dados de multimídia, incluindo dados de imagem e vídeo. O pipeline de mídia 234 implanta

operações de mídia por meio de solicitações para computar ou amostrar lógica dentro dos subnúcleos 221-221F.

[0053] Em uma modalidade, a interface de SoC 232 possibilita que o núcleo de processador de gráficos 219 se comunique com núcleos de processador de aplicativo de propósito geral (por exemplo, CPUs) e/ou outros componentes dentro de um SoC, incluindo elementos de hierarquia de memória, como uma memória cache de último nível compartilhada, a RAM de sistema, e/ou DRAM em chio ou em pacote embutida. A interface de SoC 232 também pode possibilitar comunicação com dispositivos de função fixa dentro do SoC, como pipelines de imageamento de câmera, e possibilita o uso de e/ou implanta atômicas de memória global que podem ser compartilhadas entre o núcleo de processador de gráficos 219 e CPUs dentro do SoC. A interface de SoC 232 também pode implantar controles de gerenciamento de potência para o núcleo de processador de gráficos 219 e possibilitar uma interface entre um domínio de relógio do núcleo de gráfico 219 e outros domínios de relógio dentro do SoC. Em uma modalidade, a interface de SoC 232 possibilita o recebimento de armazenamentos temporários de comando de um gerador de fluxo contínuo de comando e despachante de encadeamento global que são configurados para fornecer comandos e instruções para cada um dentre um ou mais núcleos de gráficos dentro de um processador de gráficos. Os comandos e instruções podem ser despachados para o pipeline de mídia 234, quando operações de mídia devem ser realizadas, ou uma geometria e pipeline de

função fixa (por exemplo, geometria e pipeline de função fixa 231, geometria e pipeline de função fixa 237) quando operações de processamento de gráficos devem ser realizadas.

[0054] O microcontrolador de gráficos 233 pode ser configurado para executar diversas tarefas de programação e gerenciamento para o núcleo de processador de gráficos 219. Em uma modalidade, o microcontrolador de gráficos 233 pode executar programação de carga de trabalho de computação e/ou gráficos nos diversos mecanismos paralelos de gráficos dentro de matrizes de unidade de execução (EU) 222A-222F, 224A-224F dentro dos subnúcleos 221A-221F. Nesse modelo de programação, software hospedeiro que é executado em um núcleo de CPU de um SoC, incluindo o núcleo de processador de gráficos 219 pode submeter cargas de trabalho, uma dentre múltiplas campainhas de processador de gráfico, o que invoca uma operação de programação no mecanismo de gráficos apropriado. Operações de programação incluem determinar qual carga de trabalho executar a seguir, submeter uma carga de trabalho a um gerador de fluxo contínuo de comando, pré-esvaziar cargas de trabalho existentes que são executadas em um mecanismo, monitorar o progresso de uma carga de trabalho e notificar o software hospedeiro quando uma carga de trabalho está concluída. Em uma modalidade, o microcontrolador de gráficos 233 também pode facilitar estados ociosos ou de baixa potência para o núcleo de processador de gráficos 219, fornecendo o núcleo de processador de gráficos 219 com a capacidade de salvar e restaurar registros dentro

do núcleo de processador de gráficos 219 através de transições de estado de baixa potência independentemente do sistema operacional e/ou software acionador de gráficos no sistema.

[0055] O núcleo de processador de gráficos 219 pode ter mais ou menos que os subnúcleos ilustrados 221A-221F, até *N* subnúcleos modulares. Para cada conjunto de *N* subnúcleos, o núcleo de processador de gráficos 219 também pode incluir lógica de função compartilhada 235, cache de memória compartilhada 236, uma geometria/pipeline de função fixa 237 237, bem como lógica de função fixa adicional 238 para acelerar diversos gráficos e computar operações de processamento. A lógica de função compartilhada 235 pode incluir unidades de lógica associadas à lógica de função compartilhada 420 da **Figura 4** (por exemplo, amostrador, matemática e/ou lógica de comunicação ente encadeamentos) que podem ser compartilhados por cada *N* subnúcleos dentro do núcleo de processador de gráficos 219. A memória cache e/ou compartilhada 236 pode ser um cache de último nível para o conjunto de *N* subnúcleos 221A-221F dentro do núcleo de processador de gráficos 219, e também pode servir como memória compartilhada que é acessível por múltiplos subnúcleos. A geometria/pipeline de função fixa 237 pode ser incluído em vez da geometria/pipeline de função fixa 231 dentro do bloco de função fixa 230 e pode incluir as mesmas unidades de lógica ou unidades de lógica similares.

[0056] Em uma modalidade, o núcleo de processador de gráficos 219 inclui lógica de função fixa adicional 238

que pode incluir diversas lógicas de aceleração de função fixa para uso pelo núcleo de processador de gráficos 219. Em uma modalidade, a lógica de função fixa adicional 238 inclui um pipeline de geometria adicional para uso apenas em posição de sombreamento. Apenas em posição de sombreamento, dois pipelines de geometria existem, o pipeline de geometria completo dentro de geometria/pipeline de função fixa238, 231, e um pipeline de eliminação, que é um pipeline de geometria adicional que pode ser incluído dentro da lógica de função fixa adicional 238. Em uma modalidade, o pipeline de eliminação é uma versão reduzida do pipeline de geometria completo. O pipeline completo e o pipeline de eliminação podem executar diferentes exemplos da mesma aplicação, sendo que cada exemplo tem um contexto separado. Sombreamento apenas em posição pode esconder execuções de eliminação longas de triângulos descartados, que permite que o sombreamento seja concluído mais cedo em alguns casos. Por exemplo e em uma modalidade, a lógica de pipeline de eliminação dentro da lógica de função fixa adicional 238 pode executar sombreadores de posição em paralelo com o aplicativo principal e, em geral, gera resultados críticos mais rápidos que o pipeline completo, uma vez que o pipeline de eliminação coleta e sombreia apenas o atributo de posição dos vértices, sem realizar rasterização e renderização dos pixels para o armazenamento temporário de quadro. O pipeline de eliminação pode usar os resultados cruciais gerados para computar informações de visibilidade para todos os

triângulos sem relação a possibilidade desses triângulos serem selecionados. O pipeline completo (que, nesse caso, pode ser denominado um pipeline repetido) pode consumir as informações de visibilidade para pular os triângulos selecionados para sombrear apenas os triângulos visíveis que são finalmente passados para a fase de rasterização.

[0057] Em uma modalidade, a lógica de função fixa adicional 238 também pode incluir lógica de aceleração de aprendizado por máquina, como lógica de multiplicação de matriz de função fixa, para implantações, incluindo otimizações para treinamento ou inferência de aprendizado por máquina.

[0058] Dentro de cada subnúcleo de gráficos 221A a 221F inclui um conjunto de recursos de execução que podem ser usados para realizar operações de gráficos, mídia e computação em resposta às solicitações pelo pipeline de gráficos, pipeline de mídia ou programas de sombreador. Os subnúcleos de gráficos 221A–221F incluem múltiplas matrizes de EU 222A–222F, 224A–224F, envio de encadeamento e lógica de comunicação entre encadeamentos (TD/IC) 223A–223F, um amostrador 3D (por exemplo, textura) 225A–225F, um amostrador de mídia 206A–206F, um processador de sombreador 227A–227F, e memória local compartilhada (SLM) 228A–228F. As matrizes de EU 222A–222F, 224A–224F cada uma, incluem múltiplas unidades de execução, que são unidades de processamento de gráficos de propósito geral com capacidade de realizar operações de lógica de ponto fixo/número inteiro e ponto de flutuação em serviço de uma operação de gráficos, mídia

ou computação, incluindo programas de sombreador de gráficos, mídia ou computação. A lógica de TD/IC 223A-223F executa operações de envio de encadeamento e controle de encadeamento locais para as unidades de execução dentro de um subnúcleo e facilita comunicação entre encadeamentos que são executados nas unidades de execução do subnúcleo. O amostrador 3D 225A-225F pode ler textura ou outros dados relacionados a gráficos na em memória. O amostrador 3D pode ler dados de textura diferentemente com base em um estado de amostra configurada e no formato de textura associado a uma dada textura. O amostrador de mídia 206A-206F pode executar operações de leitura similares com base no tipo e formato associados a dados de mídia. Em uma modalidade, cada subnúcleo de gráficos 221A-221F pode incluir alternativamente um amostrador de mídia e 3D unificado. Os encadeamentos que são executados nas unidades de execução dentro de cada um dos subnúcleos 221A-221F pode fazer uso de memória local compartilhada 228A-228F dentro de cada subnúcleo, para possibilitar encadeamentos que são executados dentro de um grupo de encadeamento para executar com o uso de um agrupamento comum de memória em chip.

[0059] A **Figura 2C** ilustra uma unidade de processamento de gráficos (GPU) 239 que inclui conjuntos dedicados de recursos de processamento de gráficos dispostos em grupos de múltiplos núcleos 240A-240N. Embora os detalhes de apenas um único grupo de múltiplos núcleos 240A sejam fornecidos, será observado que os outros grupos de múltiplos núcleos 240B-240N podem ser

equipados com os mesmos conjuntos ou conjuntos similares de recursos de processamento de gráficos.

[0060] Como ilustrado, um grupo de múltiplos núcleos 240A pode incluir um conjunto de núcleos de gráficos 243, um conjunto de núcleos de tensor 244, e um conjunto de núcleos de traçamento de raios 245. Um agendador/despachante 241 agenda e despacha os encadeamentos de gráficos para execução nos diversos núcleos 243, 244, 245. Um conjunto de arquivos de registrador 242 armazena valores operacionais usados pelos núcleos 243, 244, 245 quando executam os encadeamentos de gráficos. Esses podem incluir, por exemplo, registros de número inteiro para armazenar valores de número inteiro, registros de ponto de flutuação para armazenar valores de ponto de flutuação, registros de vetor para armazenar elementos de dados empacotados (elementos de dados de número inteiro e/ou ponto de flutuação) e registros de bloco de lado a lado para armazenar valores de tensor/matriz. Em uma modalidade, os registros de bloco de lado a lado são implantados como conjuntos combinados de registros de vetor.

[0061] Uma ou mais unidades de caches de nível 1 (L1) e memória compartilhada 247 armazenam dados de gráficos, como dados de textura, dados de vértice, dados de pixel, dados de raio, dados de volume limitante, etc., localmente dentro de cada grupo de múltiplos núcleos 240A. Uma ou mais unidades de textura 247 também podem ser usadas para executar operações de textura, como mapeamento de textura e amostragem. Um cache de Nível 2

(L2) 253 compartilhado por todos ou um subconjunto dos grupos de múltiplos núcleos 240A-240N armazena dados de gráficos e/ou instruções para múltiplos encadeamentos de gráficos concorrentes. Como ilustrado, o cache L2 253 pode ser compartilhado através de uma pluralidade de grupos de múltiplos núcleos 240A-240N. Um ou mais controladores de memória 248 acoplam a GPU 239 a uma memória 249 que pode ser uma memória de sistema (por exemplo, DRAM) e/ou uma memória de gráficos dedicada (por exemplo, memória GDDR6).

[0062] O conjunto de circuitos de entrada/saída (I/O) 250 acopla a GPU 239 a um ou mais dispositivos de I/O 252, como processadores de sinal digital (DSPs), controladores de rede ou dispositivos de entrada de usuário. Uma interconexão em chip pode ser usada para acoplar os dispositivos de I/O 252 à GPU 239 e à memória 249. Uma ou mais unidades de gerenciamento de memória de I/O (IOMMUs) 251 do conjunto de circuitos de I/O 250 acopla os dispositivos de I/O 252 diretamente à memória de sistema 249. Em uma modalidade, a IOMMU 251 gerencia múltiplos conjuntos de tabelas de página para mapear endereços virtuais para endereços físicos na memória de sistema 249. Nessa modalidade, os dispositivos de I/O 252, CPU(s) 246, e GPU(s) 239 podem compartilhar o mesmo espaço de endereço virtual.

[0063] Em uma implantação, a IOMMU 251 suporta virtualização. Nesse caso, a mesma pode gerenciar um primeiro conjunto de tabelas de página para mapear endereços virtuais de visitante/gráficos para endereços físicos de visitante/gráficos e um segundo conjunto de

tabelas de página para mapear os endereços físicos de visitante/gráficos para endereços físicos de sistema/hospedeiro (por exemplo, dentro de memória de sistema 249). Os endereços de base de cada um dentre o primeiro e o segundo conjuntos de tabelas de páginas podem ser armazenados em registros de controle e desativados em uma comutação de contexto (por exemplo, de modo que o novo contexto seja dotado de acesso ao conjunto relevante de tabelas de páginas). Embora não ilustrado na **Figura 2C**, cada um dos núcleos 243, 244, 245 e/ou grupos de múltiplos núcleos 240A-240N pode incluir armazenamentos temporários à parte de tradução (TLBs) para traduções físicas de visitante para virtual de visitante de cache, traduções físicas de hospedeiro para físicas de visitante, e traduções físicas de hospedeiro para virtual de visitante.

[0064] Em uma modalidade, as CPUs 246, GPUs 239, e dispositivos de I/O 252 são integrados em um único chip semicondutor e/ou pacote de chips. A memória ilustrada 249 pode ser integrada no mesmo chip ou pode ser acoplada aos controladores de memória 248 por meio de uma interface fora de chip. Em uma implantação, a memória 249 compreende memória GDDR6 que compartilha o mesmo espaço de endereço virtual que outras memórias de nível de sistema físico, embora os princípios subjacentes da invenção não sejam limitados a essa implantação específica.

[0065] Em uma modalidade, os núcleos de tensor 244 incluem uma pluralidade de unidades de execução especificamente projetadas para executar operações de

matriz, que são a operação de computação fundamental usada para executar operações de aprendizado profundo. Por exemplo, operações de multiplicação matricial simultâneas podem ser usadas para treinamento e inferência de rede neural. Os núcleos de tensor 244 podem executar processamento de matriz com o uso de uma variedade de precisões de operando, incluindo ponto de flutuação de precisão único (por exemplo, 32 bits), meio ponto de flutuação de precisão (por exemplo, 16 bits), palavras inteiras (16 bits), bytes (8 bits) e meio byte (4 bits). Em uma modalidade, uma implantação de rede neural extrai recursos de cada cena renderizada, que combina, potencialmente, detalhes dos múltiplos quadros, para construir uma imagem final de alta qualidade.

[0066] Em implantações de aprendizado profundo, trabalho de multiplicação matricial paralelo pode ser programado para execução nos núcleos de tensor 244. O treinamento de redes neurais, em particular, exige operações de produto escalar matricial de número significativo. A fim de processar uma formulação de produto interno de uma multiplicação de matriz N x N x N, os núcleos de tensor 244 podem incluir pelo menos N elementos de processamento de produto escalar. Antes da multiplicação matricial começar, uma matriz inteira é carregada em registros de bloco de lado a lado e pelo menos uma coluna de uma segunda matriz é carregada em cada ciclo para N ciclos. Em cada ciclo, há N produtos escalares que são processados.

[0067] Elementos matriciais podem ser armazenados em diferentes precisões dependendo da implantação

particular que inclui palavras de 16 bits, bytes de 8 bits (por exemplo, INT8) e meio byte de 4 bits (por exemplo, INT4). Modos de precisão diferentes podem ser especificados para os núcleos de tensor 244 para garantir que a precisão mais eficaz seja usada para diferentes cargas de trabalho (por exemplo, como cargas de trabalho de inferência que podem tolerar quantização para bytes e meio byte).

[0068] Em uma modalidade, os núcleos de traçado de raios 245 aceleram operações de traçado de raios tanto para implantações de traçado de raios em tempo não real quanto traçado de raios em tempo real. Em particular, os núcleos de traçado de raios 245 incluem conjunto de circuitos de interseção/travessia de raio para executar travessia de raio com o uso de hierarquias de volume limitantes (BVHs) e interseções de identificação entre raios e primitivos envolvidos dentro dos volumes de BVH. Os núcleos de traçado de raios 245 também podem incluir conjuntos de circuito para executar teste e seleção de profundidade (por exemplo, com o uso de um armazenamento temporário Z ou disposição similar). Em uma implantação, os núcleos de traçado de raios 245 realizam operações transversais e de interseção em consonância com as técnicas de eliminação de ruído de imagem descritas no presente documento, pelo menos uma porção da qual pode ser executada nos núcleos de tensor 244. Por exemplo, em uma modalidade, os núcleos de tensor 244 implantam uma rede neural de aprendizado profundo para executar eliminação de ruído de quadros gerados pelos núcleos de traçado de raios 245. No entanto, a CPU(s) 246, núcleos

de gráficos 243, e/ou núcleos de traçado de raios 245 também podem implantar todos ou uma porção dos algoritmos de eliminação de ruído e/ou aprendizado profundo.

[0069] Além disso, como descrito acima, uma abordagem distribuída para eliminação de ruído pode ser empregada, na qual a GPU 239 está em um dispositivo de computação acoplado a outros dispositivos de computação em uma rede ou interconexão de alta velocidade. Nessa modalidade, os dispositivos de computação interconectados compartilham dados de aprendizado/treinamento de rede neural para melhorar a velocidade com a qual o sistema geral aprende a realizar eliminação de ruído para diferentes tipos de quadros de imagem e/ou diferentes aplicações de gráficos.

[0070] Em uma modalidade, os núcleos de traçado de raios 245 processam todas as interseções primitivas de raio e travessia de BVH, evitando que os núcleos de gráficos 243 sejam sobrecarregados com milhares de instruções por raio. Em uma modalidade, cada núcleo de traçado de raios 245 inclui um primeiro conjunto de conjunto de circuitos especializado para executar testes de caixa delimitadora (por exemplo, para operações de travessia) e um segundo conjunto de conjunto de circuitos especializado para executar os testes de interseção de raio e triângulo (por exemplo, raios de interseção que foram atravessados). Desse modo, em uma modalidade, o grupo de múltiplos núcleos 240A pode simplesmente lançar uma sonda de raio, e os núcleos de traçado de raios 245 realizam de modo independente

travessia e interseção de raio e retornam dados de ocorrência (por exemplo, uma ocorrência, nenhuma ocorrência, múltiplas ocorrências, etc.) para o contexto de encadeamento. Os outros núcleos 243, 244 são livres para executar outro trabalho de computação ou gráficos enquanto os núcleos de traçado de raios 245 realizam as operações de travessia e interseção.

[0071] Em uma modalidade, cada núcleo de traçado de raios 245 inclui uma unidade de travessia para executar operações de teste de BVH e uma unidade de interseção que executa testes de interseção de raio primitivo. A unidade de interseção gera uma resposta de “ocorrência”, “nenhuma ocorrência”, ou “múltiplas ocorrências”, que é fornecida ao encadeamento apropriado. Durante as operações de interseção e travessia, os recursos de execução dos outros núcleos (por exemplo, núcleos de gráficos 243 e núcleos de tensor 244) são livres para realizar outras formas de trabalho de gráficos.

[0072] Em uma modalidade particular descrita abaixo, uma abordagem de traçado de raios/rasterização híbrida é usada quando trabalho é distribuído entre os núcleos de gráficos 243 e núcleos de traçado de raios 245.

[0073] Em uma modalidade, os núcleos de traçado de raios 245 (e/ou outros núcleos 243, 244) incluem suporte de hardware para um conjunto de instruções de traçado de raios como Traçado de raios DirectX da Microsoft (DXR) que inclui um comando DispatchRays, bem como geração de raio, ocorrência mais próxima, qualquer ocorrência e sombreadores perdidos, que possibilitam a designação de conjuntos únicos de sombreadores e texturas para cada

objeto. Outra plataforma de traçado de raios que pode ser suportada pelos núcleos de traçado de raios 245, núcleos de gráficos 243 e núcleos de tensor 244 é Vulkan 1.1.85. No entanto, nota-se que os princípios subjacentes da invenção não se limitam a nenhuma ISA de traçado de raios particular.

[0074] Em geral, os diversos núcleos 245, 244, 243 podem suportar um conjunto de instruções de traçado de raios que inclui instruções/funções para geração de raio, ocorrência mais próxima, qualquer ocorrência, interseção de raio primitivo, construção de caixa delimitadora hierárquica e por primitiva, perda, visita e exceções. Mais especificamente, uma modalidade inclui instruções de traçado de raios para realizar as seguintes funções:

[0075] Geração de Raio – Instruções de geração de raio podem ser executadas para cada pixel, amostra, ou outra atribuição de trabalho definida pelo usuário.

[0076] Ocorrência Mais Próxima – Uma instrução de ocorrência mais próxima pode ser executada para localizar o ponto de interseção mais próximo de um raio com primitivos dentro de uma cena.

[0077] Qualquer Ocorrência – Uma instrução de qualquer ocorrência identifica múltiplas interseções entre um raio e primitivos dentro de uma cena, potencialmente para identificar um novo ponto de interseção mais próximo.

[0078] Interseção – Uma instrução de interseção realiza um teste de interseção de raio primitivo e emite um resultado.

[0079] Construção de Caixa Delimitadora Por Primitivo – Essa instrução constrói uma caixa delimitadora ao redor de um dado primitivo ou grupo de primitivos (por exemplo, ao construir uma nova BVH ou outra estrutura de dados de aceleração).

[0080] Perda – Indica que um raio erra toda a geometria dentro de uma cena, ou região especificada de uma cena.

[0081] Visita – Indica os volumes filhos que um raio atravessará.

[0082] Exceções – Inclui vários tipos de manipuladores de exceção (por exemplo, invocados para várias condições de erro).

[0083] A **Figura 2D** é um diagrama de blocos de unidade de processamento de gráficos de propósito geral (GPGPU) 270 que pode ser configurada como um processador de gráficos e/ou acelerador de computação, de acordo com modalidades descrita no presente documento. A GPGPU 270 pode se interconectar com processadores hospedeiros (por exemplo, uma ou mais CPUs 246) e memória 271, 272 por meio de um ou mais barramentos de sistema e/ou memória. Em uma modalidade, a memória 271 é memória de sistema que pode ser compartilhada com a uma ou mais CPUs 246, enquanto a memória 272 é memória de dispositivo que é dedicada para a GPGPU 270. Em uma modalidade, componentes dentro da GPGPU 270 e memória de dispositivo 272 podem ser mapeados em endereços de memória que são acessíveis à uma ou mais CPUs 246. Acesso à memória 271 e 272 pode ser facilitada por meio de um controlador de memória 268. Em uma modalidade, o controlador de memória

268 inclui um controlador de acesso de memória direta interna (DMA) 269 ou pode incluir lógica para executar operações que seriam, de outro modo, realizadas por um controlador de DMA.

[0084] A GPGPU 270 inclui múltiplas memórias de cache, incluindo um cache L2 253, cache L1 254, um cache de instrução 255, e memória compartilhada 256, pelo menos uma porção a qual também pode ser particionada como uma memória cache. A GPGPU 270 também inclui múltiplas unidades de computação 260A-260N. Cada unidade de computação 260A-260N inclui um conjunto de registros de vetor 261, registros escalares 262, unidades de lógica de vetor 263, e unidades de lógica escalar 264. As unidades de computação 260A-260N também podem incluir memória compartilhada local 265 e um contador de programa 266. As unidades de computação 260A-260N podem se acoplar a um cache constante 267, que pode ser usado para armazenar dados constantes, que são dados que não serão alterados durante a execução de programa kernel ou de sombreador que é executado na GPGPU 270. Em uma modalidade, o cache constante 267 é um cache de dados escalar e dados em cache podem ser coletados diretamente nos registros escalares 262.

[0085] Durante a operação, a uma ou mais CPUs 246 podem escrever comandos em registros ou memória na GPGPU 270 que foram mapeados em um espaço de endereço acessível. Os processadores de comando 257 podem ler os comandos de registros ou memória e determinar como esses comandos serão processados dentro da GPGPU 270. Um despachante de encadeamento 258 pode, então, ser usado

para despachar encadeamentos para as unidades de computação 260A-260N para executar esses comandos. Cada unidade de computação 260A-260N pode executar encadeamentos independentemente das outras unidades de computação. Adicionalmente, cada unidade de computação 260A-260N pode ser independentemente configurada para computação condicional e pode emitir condicionalmente os resultados de computação para a memória. Os processadores de comando 257 podem interromper a uma ou mais CPUs 246 quando os comandos submetidos são concluídos.

[0086] As **Figuras 3A a 3C** ilustram diagramas de blocos de arquiteturas de processador de gráficos e acelerador de computação adicionais fornecidas através de modalidades descritas no presente documento. Os elementos das Figuras 3A a 3C que têm os mesmos números de referência (ou nomes) que os elementos de qualquer outra Figura no presente documento podem operar ou funcionar de qualquer maneira similar àquela descrita em outro lugar no presente documento, porém, sem limitação a tal.

[0087] A **Figura 3A** é um diagrama de blocos de um processador de gráficos 300, que pode ser uma unidade de processamento de gráficos distinta, ou pode ser um processador de gráficos integrado a uma pluralidade de núcleos de processamento, ou outros dispositivos semicondutores como, mas não limitados a, dispositivos de memória ou interfaces de rede. Em algumas modalidades, o processador de gráficos se comunica por meio de uma interface de I/O mapeada de memória para

registros no processador de gráficos e com comandos colocados na memória de processador. Em algumas modalidades, processador de gráficos 300 inclui uma interface de memória 314 para acessar a memória. A interface de memória 314 pode ser uma interface para memória local, um ou mais caches internos, um ou mais caches externos compartilhados e/ou para memória de sistema.

[0088] Em algumas modalidades, o processador de gráficos 300 também inclui um controlador de exibição 302 para acionar dados de saída de exibição para um dispositivo de exibição 318. O controlador de exibição 302 inclui hardware para um ou mais planos de sobreposição para a exibição e a composição de múltiplas camadas de elementos de interface de vídeo ou de usuário. O dispositivo de exibição 318 pode ser um dispositivo de exibição interno ou externo. Em uma modalidade, o dispositivo de exibição 318 é um dispositivo de exibição montado de cabeça, como um dispositivo de exibição de realidade virtual (VR) ou um dispositivo de exibição de realidade aumentada (AR). Em algumas modalidades, o processador de gráficos 300 inclui um mecanismo de codec de vídeo 306 para codificar, decodificar ou transcodificar mídia para, de, ou entre um ou mais formatos de codificação de mídia, incluindo, mas não limitado a formatos de Grupo de Especialistas de Figuração em Movimento (MPEG) como MPEG-2, formatos de Codificação de Vídeo Avançada (AVC) como H.264/MPEG-4 AVC, H.265/HEVC, Aliança para Mídia Aberta (AOMedia) VP8, VP9, bem como a Sociedade de

Engenheiros de Televisão e Figuração em Movimento (SMPTE) 421M/VC-1, e formatos de Grupo de Especialistas de Fotografia Conjunto (JPEG), como JPEG, e formatos de JPEG em Movimento (MJPEG).

[0089] Em algumas modalidades, o processador de gráficos 300 inclui um mecanismo de transferência de imagem em bloco (BLIT) 304 para executar operações de rasterizador bidimensionais (2D), incluindo, por exemplo, transferências de bloco limitado por bits. No entanto, em uma modalidade, as operações de gráficos 2D são realizadas com o uso de um ou mais componentes de mecanismo de processamento de gráficos (GPE) 310. Em algumas modalidades, GPE 310 é um mecanismo de computação para executar operações de gráficos, incluindo operações de gráficos tridimensionais (3D) e operações de mídia.

[0090] Em algumas modalidades, GPE 310 inclui um pipeline 3D 312 para realizar operações 3D, como renderizar cenas e imagens tridimensionais com o uso de funções de processamento que atuam sobre formatos primitivos 3D (por exemplo, retângulo, triângulo, etc.). O pipeline 3D 312 inclui elementos de função fixa e programável que realizam diversas tarefas dentro do elemento e/ou geram encadeamentos de execução para um subsistema de Mídia/3D 315. Embora o pipeline 3D 312 possa ser usado para executar operações de mídia, uma modalidade de GPE 310 também inclui um pipeline de mídia 316 que é especificamente usado para executar operações de mídia, como aperfeiçoamento de imagem e pós-processamento de vídeo.

[0091] Em algumas modalidades, o pipeline de mídia 316 inclui unidades de lógica programáveis ou de função fixa para executar uma ou mais operações de mídia especializadas, como aceleração de decodificação de vídeo, desentrelaçamento de vídeo, e aceleração de codificação de vídeo no lugar de, ou em prol do mecanismo de codec de vídeo 306. Em algumas modalidades, o pipeline de mídia 316 inclui adicionalmente uma unidade de geração de encadeamento para gerar encadeamentos para execução no subsistema de Mídia/3D 315. Os encadeamentos gerados realizam cálculos para as operações de mídia em uma ou mais unidades de execução de gráficos incluídas no subsistema de Mídia/3D 315.

[0092] Em algumas modalidades, o subsistema de Mídia/3D 315 inclui lógica para executar encadeamentos gerados por pipeline 3D 312 e pipeline de mídia 316. Em uma modalidade, os pipelines enviam solicitações de execução de encadeamento para o subsistema de Mídia/3D 315, que inclui envio de encadeamento lógica para arbitrar e despachar as diversas solicitações para fontes de execução de encadeamento disponíveis. Os recursos de execução incluem uma matriz de unidades de execução de gráficos para processar os encadeamentos 3D e de mídia. Em algumas modalidades, o subsistema de Mídia/3D 315 inclui um ou mais caches internos para instruções e dados de encadeamento. Em algumas modalidades, o subsistema também inclui memória compartilhada, inclusive registros e memória endereçável, para compartilhar dados entre encadeamentos e para armazenar dados de saída.

[0093] A **Figura 3B** ilustra um processador de gráficos 320 que tem uma arquitetura lado a lado, de acordo com modalidades descritas no presente documento. Em uma modalidade, o processador de gráficos 320 inclui um agrupamento de mecanismo de processamento de gráficos 322 que tem múltiplos exemplos do mecanismo de processamento de gráficos 310 da **Figura 3A** dentro de um bloco de mecanismo de gráficos 310A-310D. Cada bloco de mecanismo de gráficos 310A-310D pode ser interconectado por meio de um conjunto de interconexões de bloco 323A-323F. Cada bloco de mecanismo de gráficos 310A-310D também pode ser conectado a um módulo de memória ou dispositivo de memória 326A-326D por meio de interconexões de memória 325A-325D. Os dispositivos de memória 326A-326D podem usar qualquer tecnologia de memória de gráficos. Por exemplo, os dispositivos de memória 326A-326D podem ser memória de taxa de dados duplos de gráficos (GDDR). Os dispositivos de memória 326A-326D, em uma modalidade, são módulos de memória de largura de banda alta (HBM) que podem estar em molde com seu respectivo bloco de mecanismo de gráficos 310A-310D. Em uma modalidade, os dispositivos de memória 326A-326D são dispositivos de memória empilhados que podem ser empilhados no topo de seu respectivo bloco de mecanismo de gráficos 310A-310D. Em uma modalidade, cada bloco de mecanismo de gráficos 310A-310D e memória associada 326A-326D reside em chiplets separados, que são ligados a um molde de base ou substrato de base, como descrito em detalhes adicionais nas **Figuras 11B a 11D**.

[0094] O agrupamento de mecanismo de processamento de gráficos 322 pode se conectar a uma interconexão de malha em chip ou em pacote 324. A interconexão de malha 324 pode possibilitar comunicação entre blocos de mecanismo de gráficos 310A-310D e componentes como o codec de vídeo 306 e um ou mais mecanismos de cópia 304. Os mecanismos de cópia 304 podem ser usados para mover dados para fora de, para e entre os dispositivos de memória 326A-326D e memória que é externa ao processador de gráficos 320 (por exemplo, memória de sistema). A interconexão de malha 324 também pode ser usada para interconectar os blocos de mecanismo de gráficos 310A-310D. O processador de gráficos 320 pode incluir opcionalmente um controlador de exibição 302 para possibilitar uma conexão com um dispositivo de exibição externo 318. O processador de gráficos também pode ser configurado como um acelerador de computação ou gráficos. Na configuração de acelerador, o controlador de exibição 302 e o dispositivo de exibição 318 podem ser omitidos.

[0095] O processador de gráficos 320 pode se conectar a um sistema de hospedeiro por meio de uma interface de hospedeiro 328. A interface de hospedeiro 328 pode possibilitar comunicação entre o processador de gráficos 320, memória de sistema, e/ou outros componentes de sistema. A interface de hospedeiro 328 pode ser, por exemplo, um barramento de PCI expressa ou outro tipo de interface de sistema de hospedeiro.

[0096] A **Figura 3C** ilustra um acelerador de computação 330, de acordo com modalidades descritas no presente

documento. O acelerador de computação 330 pode incluir similaridades de arquitetura com o processador de gráficos 320 da Figura 3B e é otimizado para aceleração de computação. Um agrupamento de mecanismo de computação 332 pode incluir um conjunto de blocos de mecanismo de computador 340A-340D que incluem lógica de execução que é otimizada para operações de computação de propósito feral com base em vetor ou paralelas. Em algumas modalidades, os blocos de mecanismo de computação 340A-340D não incluem lógica de processamento de gráficos de função fixa, embora, em uma modalidade, um ou mais dos blocos de mecanismo de computação 340A-340D possam incluir lógica para executar aceleração de mídia. O mecanismo de computação tiles 340A-340D pode se conectar à memória 326A-326D por meio de interconexões de memória 325A-325D. A memória 326A-326D e interconexões de memória 325A-325D podem ter tecnologia similar, como no processador de gráficos 320, ou podem ser diferentes. Os blocos de mecanismo de computação de gráficos 340A-340D também podem ser interconectados por meio de um conjunto de interconexões de bloco 323A-323F e podem ser conectados com e/ou interconectados por uma interconexão de malha 324. Em uma modalidade, o acelerador de computação 330 inclui um cache L3 grande 336 que pode ser configurado como um cache amplo de dispositivo. O acelerador de computação 330 também pode ser conectar a um processador de hospedeiro e memória por meio de uma interface de hospedeiro 328 de uma maneira similar ao processador de gráficos 320 da **Figura 3B.**

MECANISMO DE PROCESSAMENTO DE GRÁFICOS

[0097] A **Figura 4** é um diagrama de blocos de um mecanismo de processamento de gráficos 410 de um processador de gráficos de acordo com algumas modalidades. Em uma modalidade, o mecanismo de processamento de gráficos (GPE) 410 é uma versão do GPE 310 mostrado na **Figura 3A**, e também pode representar um bloco de mecanismo de gráficos 310A-310D da **Figura 3B**. Elementos da **Figura 4** que têm os mesmos números de referência (ou nomes) que os elementos de qualquer outra Figura no presente documento podem operar ou funcionar de qualquer maneira similar àquela descrita em outro lugar no presente documento, porém, sem limitação a tal. Por exemplo, o pipeline 3D 312 e o pipeline de mídia 316 da **Figura 3A** são ilustrados. O pipeline de mídia 316 é opcional em algumas modalidades do GPE 410 e pode não ser explicitamente incluído dentro do GPE 410. Por exemplo e em pelo menos uma modalidade, um processador separado de mídia e/ou imagem é acoplado ao GPE 410.

[0098] Em algumas modalidades, o GPE 410 se acopla a ou inclui um gerador de fluxo contínuo de comando 403, que fornece um fluxo contínuo comando para o pipeline 3D 312 e/ou pipelines de mídia 316. Em algumas modalidades, gerador de fluxo contínuo de comando 403 é acoplado à memória, que pode ser memória de sistema, ou um ou mais dentre cache de memória interna e cache de memória compartilhada. Em algumas modalidades, o gerador de fluxo contínuo de comando 403 recebe comandos da memória e envia os comandos para o pipeline 3D 312 e/ou pipeline de mídia 316. Os comandos são diretrizes coletadas a partir de um armazenamento temporário de anel, que

armazena comandos para o pipeline 3D 312 e o pipeline de mídia 316. Em uma modalidade, o armazenamento temporário de anel pode incluir adicionalmente armazenamentos temporários de comando de lote que armazenam lotes de múltiplos comandos. Os comandos para o pipeline 3D 312 também podem incluir referências para dados armazenados na memória, como, mas não limitados a, dados de vértice e geometria para o pipeline 3D 312 e/ou dados de imagem e objetos de memória para o pipeline de mídia 316. O pipeline 3D 312 e o pipeline de mídia 316 processam os comandos e dados realizando-se operações por meio de lógica dentro dos respectivos pipelines ou despachando-se um ou mais encadeamentos de execução para uma matriz de núcleo de gráficos 414. Em uma modalidade, a matriz de núcleo de gráficos 414 inclui um ou mais blocos de núcleos de gráficos (por exemplo, núcleo (ou núcleos) de gráficos 415A, núcleo (ou núcleos) de gráficos 415B), em que cada bloco inclui um ou mais núcleos de gráficos. Cada núcleo de gráficos inclui um conjunto de recursos de execução de gráficos que inclui lógica de execução de propósito geral e específica de gráficos para realizar operações de gráficos e computação, assim como processamento de textura de função fixada e/ou lógica de aceleração de aprendizado de máquina e inteligência artificial.

[0099] Em diversas modalidades, o pipeline 3D 312 pode incluir lógica programável e função fixa para processar um ou mais programas de sombreador, como sombreadores de vértice, sombreadores de geometria, sombreadores de pixel, sombreadores de fragmento, sombreadores de

computação, ou outros programas de sombreador, processando-se as instruções e despachando-se encadeamentos de execução para a matriz de núcleo de gráficos 414. A matriz de núcleo de gráficos 414 fornece um bloco unificado de recursos de execução para uso no processamento desses programas de sombreador. A lógica de execução de múltiplos propósitos (por exemplo, unidades de execução) dentro do núcleo (ou núcleos) de gráficos 415A-414B da matriz de núcleo de gráfico 414 inclui suportar diversas linguagens de sombreador de API 3D e pode executar múltiplos encadeamentos de execução simultâneos associados a múltiplos sombreadores.

[00100] Em algumas modalidades, a matriz de núcleo de gráficos 414 inclui lógica de execução para executar funções de mídia, como processamento de vídeo e/ou imagem. Em uma modalidade, as unidades de execução incluem lógica de propósito geral que é programável para realizar operações computacionais de propósito geral paralelas, adicionalmente às operações de processamento de gráficos. A lógica de propósito geral pode executar operações de processamento em paralelo ou em combinação com lógica de propósito geral dentro do núcleo (ou núcleos) de processador 107 da **Figura 1** ou núcleo 202A-202N, como na **Figura 2A**.

[00101] Os dados de saída gerados por encadeamentos que são executados na matriz de núcleo de gráficos 414 podem emitir dados para a memória em um armazenamento temporário de retorno unificado (URB) 418. O URB 418 pode armazenar dados para múltiplos encadeamentos. Em algumas modalidades, o URB 418 pode ser usado para

enviar dados entre diferentes encadeamentos que são executados na matriz de núcleo de gráficos 414. Em algumas modalidades, o URB 418 pode ser adicionalmente usado para sincronização entre encadeamentos na matriz de núcleo de gráficos e lógica de função fixa dentro da lógica de função compartilhada 420.

[00102] Em algumas modalidades, matriz de núcleo de gráficos 414 é escalonável, de modo que a matriz inclua um número variável de núcleos de gráficos, em que cada um tem um número variável de unidades de execução com base no nível de desempenho e potência alvo de GPE 410. Em uma modalidade, os recursos de execução são dinamicamente escalonáveis, de modo que os recursos de execução possam ser ativados ou desativados, conforme necessário.

[00103] A matriz de núcleo de gráficos 414 se acopla à lógica de função compartilhada 420 que inclui múltiplos recursos que são compartilhados entre os núcleos de gráficos na matriz de núcleo de gráficos. As funções compartilhadas dentro da lógica de função compartilhada 420 são unidades de lógica de hardware que fornecem funcionalidade complementar especializada para a matriz de núcleo de gráficos 414. Em diversas modalidades, a lógica de função compartilhada 420 inclui, mas não é limitada a amostrador 421, matemática 422, e lógica de comunicação entre encadeamentos (ITC) 423. Adicionalmente, algumas modalidades implantam um ou mais caches 425 dentro da lógica de função compartilhada 420.

[00104] Uma função compartilhada é implantada pelo menos em um caso em que a demanda por uma determinada

função especializada é insuficiente para inclusão dentro da matriz de núcleo de gráficos 414. Em vez de uma única instantização dessa função especializada ser implantada como uma entidade independente na lógica de função compartilhada 420 e compartilhada entre os recursos de execução dentro da matriz de núcleo de gráficos 414. O conjunto preciso de funções que são compartilhadas entre a matriz de núcleo de gráficos 414 e incluídos dentro da matriz de núcleo de gráficos 414 varia entre as modalidades. Em algumas modalidades, funções compartilhadas específicas dentro da lógica de função compartilhada 420 que são usadas de modo extensivo pela matriz de núcleo de gráficos 414 podem ser incluídas dentro da lógica de função compartilhada 416 dentro da matriz de núcleo de gráficos 414. Em diversas modalidades, a lógica de função compartilhada 416 dentro da matriz de núcleo de gráficos 414 pode incluir algumas ou todas as lógicas dentro da lógica de função compartilhada 420. Em uma modalidade, todos os elementos de lógica dentro da lógica de função compartilhada 420 podem ser duplicados dentro da lógica de função compartilhada 416 da matriz de núcleo de gráficos 414. Em uma modalidade, a lógica de função compartilhada 420 é excluída em favor da lógica de função compartilhada 416 dentro da matriz de núcleo de gráficos 414.

UNIDADES DE EXECUÇÃO

[00105] As **Figuras 5A a 5B** ilustram lógica de execução de encadeamento 500 que inclui uma matriz de elementos de processamento empregados em um núcleo de processador de gráficos de acordo com modalidades descritas no

presente documento. Elementos das **Figuras 5A a 5B** que têm os mesmos números de referência (ou nomes) que os elementos de qualquer outra Figura no presente documento podem operar ou funcionar de qualquer maneira similar àquela descrita em outro lugar no presente documento, porém, sem limitação a tal. As **Figuras 5A a 5B** ilustram uma visão geral de lógica de execução de encadeamento 500, que pode ser representativa de lógica de hardware ilustrada com cada subnúcleo 221A-221F da **Figura 2B**. A **Figura 5A** é representativa de uma unidade de execução dentro de um processador de gráficos de propósito geral, enquanto a **Figura 5B** é representativa de uma unidade de execução que pode ser usada dentro de um acelerador de computação.

[00106] Como ilustrado na **Figura 5A**, em algumas modalidades, logica de execução de encadeamento 500 inclui um processador de sombreador 502, um despachante de encadeamento 504, cache de instrução 506, uma matriz de unidade de execução escalonável que inclui uma pluralidade de unidades de execução 508A-508N, um amostrador 510, memória local compartilhada 511, um cache de dados 512, e uma porta de dados 514. Em uma modalidade, a matriz de unidade de execução escalonável pode escalonar dinamicamente habilitando ou desabilitando uma ou mais unidades de execução (por exemplo, qualquer uma das unidades de execução 508A, 508B, 508C, 508D, através de 508N-1 e 508N) com base nas exigências computacionais de uma carga de trabalho. Em uma modalidade, os componentes incluídos são interconectados por meio de uma malha de interconexão

que se liga a cada um dos componentes. Em algumas modalidades, lógica de execução de encadeamento 500 inclui uma ou mais conexões à memória, como memória de sistema ou cache de memória, através de um ou mais dentre cache de instrução 506, porta de dados 514, amostrador 510, e unidades de execução 508A-508N. Em algumas modalidades, cada unidade de execução (por exemplo, 508A) é uma unidade computacional de propósito geral programável independente que tem capacidade para executar múltiplos encadeamentos de hardware simultâneos enquanto se processam múltiplos elementos de dados em paralelo para cada encadeamento. Em diversas modalidades, a matriz de unidades de execução 508A-508N é escalonável para incluir qualquer número unidades de execução individuais.

[00107] Em algumas modalidades, as unidades de execução 508A-508N são principalmente usadas para executar programas de sombreador. Um processador de sombreador 502 pode processar os diversos programas de sombreador e despachar encadeamentos de execução associados aos programas de sombreador por meio de um despachante de encadeamento 504. Em uma modalidade, o despachante de encadeamento inclui lógica para arbitrar solicitações de iniciação de encadeamento dos pipelines de mídia e gráficos e instanciar os encadeamentos solicitados em uma ou mais unidades de execução nas unidades de execução 508A-508N. Por exemplo, um pipeline de geometria pode enviar sombreadores de vértice, tesselação ou geometria para a lógica de execução de encadeamento para processamento. Em algumas modalidades,

despachante de encadeamento 504 também pode processar solicitações de geração de encadeamento de tempo de execução dos programas de sombreador de execução.

[00108] Em algumas modalidades, as unidades de execução 508A-508N suportam um conjunto de instruções que incluem suporte nativo para muitas instruções de sombreador de gráficos 3D padrão, de modo que os programas de sombreador de bibliotecas de gráficos (por exemplo, Direct 3D e OpenGL) sejam executados com uma tradução mínima. As unidades de execução suportam processamento de vértice e geometria (por exemplo, programas de vértice, programas de geometria, sombreadores de vértice), processamento de pixel (por exemplo, sombreadores de pixel, sombreadores de fragmento) e processamento de propósito geral (por exemplo, sombreadores de computação e de mídia). Cada uma das unidades de execução 508A-508N tem capacidade de execução de múltiplos dados de instrução única de múltiplas questões (SIMD) e operação com múltiplos encadeamentos possibilita um ambiente de execução eficaz na face de acessos de memória de latência superior. Cada encadeamento de hardware dentro de cada unidade de execução tem um arquivo de registro de largura de banda elevada dedicado e estado de encadeamento independente e associado. Execução tem múltiplas questões por tempo em pipelines que têm capacidade para operações de ponto de flutuação de precisão de número inteiro, única e dupla, capacidade de ramificação de SIMD, operações lógicas, operações transcendentais e outras operações diversas. Enquanto se aguarda por dados de memória ou uma das

funções compartilhadas, lógica de dependência dentro das unidades de execução 508A-508N ocasiona um encadeamento de espera para suspender até que os dados solicitados tenham sido retornados. Enquanto o encadeamento de espera dorme, os recursos de hardware podem ser devotados ao processamento de outros encadeamentos. Por exemplo, durante um atraso associado a uma operação de sombreador de vértice, uma unidade de execução pode realizar operações para um sombreador de pixel, sombreador de fragmento ou outro tipo de programa de sombreador que inclui um diferente sombreador de vértice. Várias modalidades podem se aplicar para usar a execução através do Múltiplo Encadeamento de Instrução Única (SIMT) como uma alternativa ao uso de SIMD ou adicionalmente ao uso de SIMD. Referência a um núcleo ou operação de SIMD também pode se aplicar ao SIMT ou se aplicar aos SIMD em combinação com SIMT.

[00109] Cada unidade de execução em unidades de execução 508A-508N opera em matrizes de elementos de dados. A quantidade de elementos de dados é o “tamanho de execução”, ou a quantidade de canais para a instrução. Um canal de execução é uma unidade lógica de execução para acesso de elemento de dados, mascaramento e controle de fluxo dentro de instruções. O número de canais pode ser independente do número de Unidades de Lógica Aritmética (ALUs) físicas ou Unidades de Ponto de Flutuação (FPUs) para um processador de gráficos particular. Em algumas modalidades, unidades de execução 508A-508N suportam tipos de dados de número inteiro e ponto de flutuação.

[00110] O conjunto de instruções de unidade de execução inclui instruções de SIMD. Os vários elementos de dados podem ser armazenados como um tipo de dados de pacote em um registro e a unidade de execução processará os vários elementos com base no tamanho de dados dos elementos. Por exemplo, quando se opera em um vetor amplo de 256 bits, os 256 bits do vetor são armazenados em um registrador e a unidade de execução opera no vetor como quatro elementos de dados de pacote de 54-bit separados (elementos de dados de tamanho de Palavra Dupla (QW)), oito elementos de dados de pacote de 32-bit separados (elementos de dados de tamanho de Palavra Dupla (DW)), dezesseis elementos de dados de pacote de 16-bit separados (elementos de dados de tamanho de Palavra (W)), ou trinta e dois elementos de dados de 8-bit separados (elementos de dados de tamanho de byte (B)). No entanto, larguras de vetor e tamanhos de registros diferentes são possíveis.

[00111] Em uma modalidade, uma ou mais unidades de execução podem ser combinadas em uma unidade de execução fundida 509A-509N que tem lógica de controle de encadeamento (507A-507N) que é comum para as EUs fundidas. Múltiplas EUs podem ser fundidas em um grupo de EU. Cada EU no grupo de EU fundida pode ser configurada para executar um encadeamento de hardware de SIMD separado. O número de EUs em um grupo de EU fundida pode variar de acordo com modalidades. Adicionalmente, várias larguras de SIMD podem ser reveladas por EU, que incluem, porém, sem limitação, SIMD8, SIMD16 e SIMD32. Cada unidade de execução de gráficos fundida 509A-509N

inclui pelo menos duas unidades de execução. Por exemplo, a unidade de execução fundida 509A inclui uma primeira EU 508A, segunda EU 508B, e lógica de controle de encadeamento 507A que é comum para a primeira EU 508A e a segunda EU 508B. A lógica de controle de encadeamento 507A controla encadeamentos executados na unidade de execução de gráficos fundida 509A, permitindo que cada EU dentro das unidades de execução fundidas 509A–509N seja executada com o uso de um registro de indicador de instrução comum.

[00112] Um ou mais caches de instrução internos (por exemplo, 506) são incluídos na lógica de execução de encadeamento 500 para instruções de encadeamento de cache para as unidades de execução. Em algumas modalidades, um ou mais caches de dados (por exemplo, 512) são incluídos para armazenar em cache dados de encadeamento durante a execução de encadeamento. A execução de encadeamentos na lógica de execução 500 também pode armazenar explicitamente dados gerenciados na memória local compartilhada 511. Em algumas modalidades, um amostrador 510 é incluído para fornecer amostragem de textura para operações 3D e amostragem de mídia para operações de mídia. Em algumas modalidades, o amostrador 510 inclui funcionalidade de amostragem de textura ou mídia para processar textura ou dados de mídia durante o processo de amostragem antes de fornecer os dados amostrados para uma unidade de execução.

[00113] Durante a execução, os pipelines de gráficos e mídia enviam solicitações de iniciação de encadeamento para lógica de execução de encadeamento 500 por meio de

lógica de geração e envio de encadeamento. Uma vez que um grupo de objetos geométricos foi processado e rasterizado em dados de pixel, a lógica de processador de pixel (por exemplo, lógica de sombreador de pixel, lógica de sombreador de fragmento, etc.) dentro do processador de sombreador 502 é invocada para computar, adicionalmente, informações de saída e fazem os resultados serem gravados em superfícies de saída (por exemplo, armazenamentos temporários de cor, armazenamentos temporários de profundidade, armazenamentos temporários de estêncil, etc.). Em algumas modalidades, um sombreador de pixel ou sombreador de fragmento calcula os valores dos vários atributos de vértice que devem ser interpolados através do objeto rasterizado. Em algumas modalidades, a lógica de processador de pixel dentro do processador de sombreador 502, então, executa um programa de sombreador de pixel ou fragmento abastecido pela interface de programação de aplicativo (API). Para executar o programa de sombreador, o processador de sombreador 502 despacha encadeamentos para uma unidade de execução (por exemplo, 508A) por meio de despachante de encadeamento 504. Em algumas modalidades, processador de sombreador 502 usa lógica de amostragem de textura no amostrador 510 para acessar dados de textura em mapas de textura armazenados na memória. Operações aritméticas nos dados de textura e nos dados de geometria de entrada computam dados de cor em pixel para cada fragmento geométrico, ou descarta um ou mais pixels de processamento adicional.

[00114] Em algumas modalidades, a porta de dados 514 fornece um mecanismo de acesso de memória para a lógica de execução de encadeamento 500 para emitir dados processados para memória para processamento adicional em um pipeline de saída de processador de gráficos. Em algumas modalidades, a porta de dados 514 inclui ou se acopla a uma ou mais memórias de cache (por exemplo, cache de dados 512) para dados de cache para acesso de memória por meio da porta de dados.

[00115] Em uma modalidade, a lógica de execução 500 também pode incluir um traçador de raios 505 que pode fornecer funcionalidade de aceleração de traçado de raios. O traçador de raios 505 pode suportar um conjunto de instruções de traçado de raios que inclui instruções/funções para geração de raio. O conjunto de instruções de traçado de raios pode ser similar a ou diferente do conjunto de instruções de traçado de raios suportado pelos núcleos de traçado de raios 245 na **Figura 2C**.

[00116] A **Figura 5B** ilustra detalhes internos exemplificativos de uma unidade de execução 508, de acordo com modalidades. Uma unidade de execução de gráficos 508 pode incluir uma unidade de coletor de instrução 537, uma matriz de arquivo de registro geral (GRF) 524, uma matriz de arquivo de registro arquitetônico (ARF) 526, um árbitro de encadeamento 522, uma unidade de envio 530, uma unidade de ramificação 532, um conjunto de unidades de ponto de flutuação de SIMD (FPUs) 534, e em uma modalidade um conjunto de ALUs de SIMD de número inteiro dedicado 535. A GRF 524 e a

ARF 526 incluem o conjunto de arquivos de registro geral e arquivos de registro de arquitetura associados a cada encadeamento de hardware simultâneo que podem estar ativos na unidade de execução de gráficos 508. Em uma modalidade, estado arquitetônico por encadeamento é mantido na ARF 526, enquanto dados usados durante execução de armazenamento são armazenados na GRF 524. O estado de execução de cada encadeamento, incluindo os indicadores de instrução para cada encadeamento, pode ser mantido em registros específicos de encadeamento na ARF 526.

[00117] Em uma modalidade, a unidade de execução de gráficos 508 tem uma arquitetura que é uma combinação de Múltiplos Encadeamentos Simultâneos (SMT) e Múltiplos Encadeamentos Intervalados (IMT) otimizados. A arquitetura tem uma configuração modular que pode ser ajustada no tempo de projeto com base em um número-alvo de encadeamentos simultâneos e número de registros por unidade de execução, em que recursos de unidade de execução são divididos através da lógica usada para executar múltiplos encadeamentos simultâneos. O número de encadeamentos lógicos que podem ser executados pela unidade de execução de gráficos 508 não é limitado ao número de encadeamentos de hardware, e múltiplos encadeamentos lógicos podem ser designados para cada encadeamento de hardware.

[00118] Em uma modalidade, a unidade de execução de gráficos 508 pode coemitir múltiplas instruções, que podem, cada uma, ser instruções diferentes. O árbitro de encadeamento 522 do encadeamento de unidade de execução

de gráficos 508 pode despachar as instruções para uma dentre a unidade de envio 530, unidade de ramificação 532, ou FPU(s) de SIMD 534 para execução. Cada encadeamento de execução pode acessar 128 registros de propósito geral dentro da GRF 524, em que cada registrador pode armazenar 32 bytes, acessível como um vetor de 8 elementos de SIMD de elementos de dados de 32-bits. Em uma modalidade, cada encadeamento de unidade de execução tem acesso a 4 Kbytes dentro da GRF 524, embora as modalidades não sejam limitadas, e mais ou menos recursos de registrador podem ser fornecidos em outras modalidades. Em uma modalidade, a unidade de execução de gráficos 508 é particionada em sete encadeamentos de hardware que podem realizar independentemente operações computacionais, embora o número de encadeamentos por unidade de execução também possa variar de acordo com as modalidades. Por exemplo, em uma modalidade, até 16 encadeamentos de hardware são suportados. Em uma modalidade na qual sete encadeamentos podem acessar 4 Kbytes, a GRF 524 pode armazenar um total de 28 Kbytes. Quando 16 encadeamentos podem acessar 4 Kbytes, a GRF 524 pode armazenar um total de 64 Kbytes. Modos de endereçamento flexíveis podem permitir que registros sejam endereçados juntos para construir registros efetivamente mais amplos ou para representar estruturas de dados de bloco retangulares distanciadas.

[00119] Em uma modalidade, operações de memória, operações de amostrador e outras comunicações de sistema de latência mais longa são despachadas por meio de

instruções de “envio” que são executadas pela unidade de envio de passagem de mensagem 530. Em uma modalidade, instruções de ramificação são despachadas para uma unidade de ramificação dedicada 532 para facilitar divergência de SIMD e convergência eventual.

[00120] Em uma modalidade, a unidade de execução de gráficos 508 inclui uma ou mais unidades de ponto de flutuação de SIMD (FPU(s)) 534 para executar operações de ponto de flutuação. Em uma modalidade, a FPU(s) 534 também suporta computação de número inteiro. Em uma modalidade, a FPU(s) 534 pode executar SIMD até *M* número de operações de ponto de flutuação de 32-bits (ou número inteiro), ou execução de SIMD até *2M* número inteiro de 16-bits ou operações de ponto de flutuação de 16-bits. Em uma modalidade, pelo menos uma das FPUs fornece capacidade matemática estendida para suportar funções de matemática transcendental de alta produtividade e ponto de flutuação de 54-bit de precisão dupla. Em algumas modalidades, um conjunto de ALUs de SIMD de número inteiro de 8 bits 535 também estão presentes, e podem ser especificamente otimizadas para realizar operações associadas às computações de aprendizado de máquina.

[00121] Em uma modalidade, matrizes de múltiplos exemplos da unidade de execução de gráficos 508 podem ser instanciados em um agrupamento de subnúcleo de gráficos (por exemplo, uma subfatia). Para escalonabilidade, arquitetos de produto podem escolher o número exato de unidades de execução por agrupamento de subnúcleo. Em uma modalidade, a unidade de execução 508 pode executar instruções em uma pluralidade de canais de

execução. Em uma modalidade adicional, cada encadeamento executado na unidade de execução de gráficos 508 é executada em um canal diferente.

[00122] **A Figura 6** ilustra uma unidade de execução adicional 600, de acordo com uma modalidade. A unidade de execução 600 pode ser uma unidade de execução otimizada por computação para uso, por exemplo, em um bloco de lado a lado de mecanismo de computação 340A a 340D como na **Figura 3C**, porém, sem limitação a isso. Variantes da unidade de execução 600 também podem ser usadas em um bloco de mecanismo de gráficos 310A-310D como na **Figura 3B**. Em uma modalidade, a unidade de execução 600 inclui uma unidade de controle de encadeamento 601, uma unidade de estado de encadeamento 602, uma unidade de coleta/pré-coleta de instrução 603, e uma unidade de decodificação de instrução 604. A unidade de execução 600 inclui adicionalmente um arquivo de registro 606 que armazena registros que podem ser designados para encadeamentos de hardware dentro da unidade de execução. A unidade de execução 600 inclui adicionalmente uma unidade de envio 607 e uma unidade de ramificação 608. Em uma modalidade, a unidade de envio 607 e a unidade de ramificação 608 podem operar de modo similar à unidade de envio 530 e uma unidade de ramificação 532 da unidade de execução de gráficos 508 da **Figura 5B**.

[00123] A unidade de execução 600 também inclui uma unidade de computação 610 que inclui múltiplos tipos diferentes de unidades funcionais. Em uma modalidade, a unidade de computação 610 inclui uma unidade ALU 611 que

inclui uma matriz de unidades de lógica aritmética. A unidade ALU 611 pode ser configurada para realizar operações de ponto de flutuação e número inteiro de 64-bits, 32-bits e 16-bits. Operações de número inteiro e ponto de flutuação podem ser reveladas simultaneamente. A unidade de computação 610 também pode incluir uma matriz sistólica 612, e uma unidade de matemática 613. A matriz sistólica 612 inclui uma rede ampla *W* e rede profunda *D* de unidades de processamento de dados que podem ser usadas para executar operações de vetor ou outras operações paralelas de dados de uma maneira sistólica. Em uma modalidade, a matriz sistólica 612 pode ser configurada para executar operações de matriz, como operações de produto escalar de matriz. Em uma modalidade, a matriz sistólica 612 suporta operações de ponto de flutuação de 16-bits, bem como operações de número inteiro de 8-bits e 4-bits. Em uma modalidade, a matriz sistólica 612 pode ser configurada para acelerar operações de aprendizado por máquina. Em tais modalidades, a matriz sistólica 612 pode ser configurada com suporte para o formato de ponto de flutuação de 16-bits bfloat. Em uma modalidade, uma unidade de matemática 613 pode ser incluída para executar um subconjunto específico de operações matemáticas de uma maneira eficaz e menos potente que uma unidade ALU 611. A unidade de matemática 613 pode incluir uma variante de lógica de matemática que pode ser constatada em lógica de função compartilhada de um mecanismo de processamento de gráficos fornecido por outras modalidades (por exemplo, lógica de matemática 422 da lógica de função

compartilhada 420 da **Figura 4**). Em uma modalidade, a unidade de matemática 613 pode ser configurada para realizar operações de ponto de flutuação de 32-bits e 64-bits.

[00124] A unidade de controle de encadeamento 601 inclui lógica para controlar a execução de encadeamentos dentro da unidade de execução. A unidade de controle de encadeamento 601 pode incluir lógica de arbitragem de encadeamento para iniciar, parar e ocupar por preempção a execução de encadeamentos dentro da unidade de execução 600. A unidade de estado de encadeamento 602 pode ser usada para armazenar estado de encadeamento para encadeamentos designados para execução na unidade de execução 600. Armazenar o estado de encadeamento dentro da unidade de execução 600 possibilita a preemptividade rápida de encadeamentos quando esses encadeamentos se tornam bloqueados ou ociosos. A unidade de coleta/pré-coleta de instrução 603 pode coletar instruções de um cache de instrução de lógica de execução de nível superior (por exemplo, cache de instrução 506 como na **Figura 5A**). A unidade de coleta/pré-coleta de instrução 603 também pode emitir solicitações de pré-coleta para instruções a serem carregadas no cache de instrução com base em uma análise de encadeamentos atualmente executados. A unidade de decodificação de instrução 604 pode ser usada para decodificar instruções a serem executadas pelas unidades de computação. Em uma modalidade, a unidade de decodificação de instrução 604 pode ser usada como um

decodificador secundário para decodificar instruções complexas em micro-operações constituintes.

[00125] A unidade de execução 600 inclui adicionalmente um arquivo de registro 606 que pode ser usado por encadeamentos de hardware que são executados na unidade de execução 600. Os registros no arquivo de registro 606 podem ser divididos na lógica usada para executar múltiplos encadeamentos simultâneos dentro da unidade de computação 610 da unidade de execução 600. O número de encadeamentos de lógica que pode ser executado pela unidade de execução de gráficos 600 não é limitado ao número de encadeamentos de hardware, e múltiplos encadeamentos de lógica podem ser designados para cada encadeamento de hardware. O tamanho do arquivo de registro 606 pode variar nas modalidades com base no número de encadeamentos de hardware suportados. Em uma modalidade, a renomeação de registro pode ser usada para alocar dinamicamente registros em encadeamentos de hardware.

[00126] A **Figura 7** é um diagrama de blocos que ilustra formatos de instrução de processador de gráficos 700 de acordo com algumas modalidades. Em uma ou mais modalidades, as unidades de execução de processador de gráficos suportam um conjunto de instruções que tem instruções em múltiplos formatos. As caixas com linha contínua ilustram os componentes que são geralmente incluídos em uma instrução de unidade de execução, enquanto as de linhas tracejadas incluem componentes que são opcionais ou que são somente incluídos em um subconjunto das instruções. Em algumas modalidades, o

formato de instrução 700 descrito e ilustrado são macroinstruções, uma vez que as mesmas são instruções fornecidas para a unidade de execução, em oposição às micro-operações que resultam da decodificação de instrução uma vez que a instrução é processada.

[00127] Em algumas modalidades, as unidades de execução de processador de gráficos suportam nativamente instruções em um formato de instrução de 128-bits 710. Um formato de instrução de 64-bits compactado 730 está disponível para algumas instruções com base na instrução selecionada, opções de instrução, e número de operandos. O formato de instrução de 128-bits nativo 710 fornece acesso a todas as opções de instrução, enquanto algumas opções e operações são restringidas no formato de 64-bits 730. As instruções nativas disponíveis no formato de 64-bits 730 variam de acordo com a modalidade. Em algumas modalidades, a instrução é compactada, em parte, com o uso de um conjunto de valores de índice em um campo de índice 713. O hardware de unidade de execução faz referência a um conjunto de tabelas de compactação com base nos valores de índice e usa as saídas de tabela de compactação para reconstruir uma instrução nativa no formato de instrução de 128-bits 710. Outros tamanhos e formatos de instrução podem ser usados.

[00128] Para cada formato, código operacional de instrução 712 define a operação que a unidade de execução deve executar. As unidades de execução executam cada instrução em paralelo através dos múltiplos elementos de dados de cada operando. Por exemplo, em resposta a uma instrução de adição, a unidade de

execução realiza uma operação de adição simultânea através de cada canal de cor que representa um elemento de textura ou elemento de figuração. Por padrão, a unidade de execução realiza cada instrução através de todos os canais de dados dos operandos. Em algumas modalidades, o campo de controle de instrução 714 possibilita controlar determinadas opções de execução, como seleção de canais (por exemplo, predicação) e ordem de canal de dados (por exemplo, efetuar swizzling). Para instruções no formato de instrução de 128-bits 710 um campo de tamanho de execução 716 limita o número de canais de dados que serão executados em paralelo. Em algumas modalidades, o campo de tamanho de execução 716 não está disponível para uso no formato de instrução compacto de 64-bits 730.

[00129] Algumas instruções de unidade de execução têm até três operandos, incluindo dois operandos de fonte, src0 720, src1 722, e um destino 718. Em algumas modalidades, as unidades de execução suportam instruções de destino duplo, onde um dentre os destinos está implicado. As instruções de manipulação de dados podem ter um operando de terceira fonte (por exemplo, SRC2 724), em que o código operacional de instrução 712 determina o número de operandos de fonte. Um último operando-fonte da instrução pode ser um valor imediato (por exemplo, codificado permanentemente) passado com a instrução.

[00130] Em algumas modalidades, o formato de instrução de 128-bits 710 inclui um campo de modo de acesso/endereço 726 que especifica, por exemplo, se modo

de endereçamento de registro direto ou modo de endereçamento de registro indireto é usado. Quando o modo de endereçamento de registro direto é usado, o endereço de registro de um ou mais operandos é diretamente fornecido em bits na instrução.

[00131] Em algumas modalidades, o formato de instrução de 128-bit 710 inclui um campo de modo de acesso/endereço 726, que especifica um modo de endereço e/ou um modo de acesso para a instrução. Em uma modalidade, o modo de acesso é usado para definir um alinhamento de acesso de dados para a instrução. Algumas modalidades suportam modos de acesso que incluem um modo de acesso alinhado de 16 bytes e um modo de acesso alinhado de 1 byte, em que o alinhamento de byte do modo de acesso determina o alinhamento de acesso dos operandos de instrução. Por exemplo, quando em um primeiro modo, a instrução pode usar endereçamento alinhado por byte para operandos de fonte e destino e, quando em um segundo modo, a instrução pode usar endereçamento alinhado de 16-bytes para todos os operandos de fonte e destino.

[00132] Em uma modalidade, a porção de modo de endereço do campo de modo de acesso/endereço 726 determina se a instrução é usar endereçamento direto ou indireto. Quando o modo de endereçamento de registro direto é usado bits na instrução fornecem diretamente o endereço de registro de um ou mais operandos. Quando o modo de endereçamento de registro indireto é usado, o endereço de registro de um ou mais operandos pode ser computado

com base em um valor de registro de endereço e um campo imediato de endereço na instrução.

[00133] Em algumas modalidades, as instruções são agrupadas com base em campos de código operacional 712 para simplificar decodificação de Código Operacional 740. Para um código operacional de 8 bits, bits 4, 5, e 6 permitem que a unidade de execução determine o tipo de código operacional. O agrupamento de código operacional preciso mostrado é meramente um exemplo. Em algumas modalidades, um grupo de código operacional de movimentação e lógica 742 inclui movimento de dados e instruções de lógica (por exemplo, movimentação (mov), comparação (cmp)). Em algumas modalidades, grupo de movimentação e lógica 742 compartilha os cinco bits mais significativos (MSB), em que instruções de movimentação (mov) são na forma de 0000xxxxb e instruções de lógica estão na forma de 0001xxxxb. Um grupo de instrução de controle de fluxo 744 (por exemplo, chamada, salto (jmp)) inclui instruções na forma de 0010xxxxb (por exemplo, 0x20). Um grupo de instruções diversas 746 inclui uma mistura de instruções, incluindo instruções de sincronização (por exemplo, esperar, enviar) na forma de 0011xxxxb (por exemplo, 0x30). Um grupo de instrução de matemática paralelo 748 inclui instruções de aritmética em termos de componente (por exemplo, adicionar, multiplicar (mul)) na forma de 0100xxxxb (por exemplo, 0x40). O grupo de matemática paralelo 748 executa as operações aritméticas em paralelo nos canais de dados. O grupo de matemática de vetor 750 inclui instruções aritméticas (por exemplo, dp4) na forma de

0101xxxxb (por exemplo, 0x50). O grupo de matemática de vetor realiza a aritmética tal como cálculos de produto escalar em operandos de vetor. A decodificação de código operacional ilustrada 740, em uma modalidade, pode ser usada para determinar qual porção de uma unidade de execução será usada para executar uma instrução decodificada. Por exemplo, algumas instruções podem ser projetadas como instruções sistólicas que serão reveladas por uma matriz sistólica. Outras instruções, como instruções de traçado de raios (não mostradas) podem ser encaminhadas para um núcleo de traçado de raios ou lógica de traçado de raios dentro de uma fatia ou divisão de lógica de execução.

PIPELINE DE GRÁFICOS

[00134] A **Figura 8** é um diagrama de blocos de outra modalidade de um processador de gráficos 800. Elementos da **Figura 8** que têm os mesmos números de referência (ou nomes) que os elementos de qualquer outra Figura no presente documento podem operar ou funcionar de qualquer maneira similar àquela descrita em outro lugar no presente documento, porém, sem limitação a tal.

[00135] Em algumas modalidades, o processador de gráficos 800 inclui um pipeline de geometria 820, um pipeline de mídia 830, um mecanismo de exibição 840, lógica de execução de encadeamento 850, e um pipeline de saída de renderizador 870. Em algumas modalidades, o processador de gráficos 800 é um processador de gráficos dentro de um sistema de processamento de múltiplos núcleos que inclui um ou mais núcleos de processamento de propósito geral. O processador de gráficos é

controlado por gravações de registrador para um ou mais registradores de controle (não mostrado) ou por meio de comandos emitidos para o processador de gráficos 800 por meio de uma interconexão de anel 802. Em algumas modalidades, interconexão de anel 802 acopla o processador de gráficos 800 a outros componentes de processamento, como outros processadores de gráficos ou processadores de propósito geral. Os comandos da interconexão de anel 802 são interpretados por um gerador de fluxo contínuo de comando 803, que fornece instruções para componentes individuais do pipeline de geometria 820 ou para o pipeline de mídia 830.

[00136] Em algumas modalidades, o gerador de fluxo contínuo de comando 803 direciona a operação de um coletor de vértice 805 que lê dados de vértice de memória e executa comandos de processamento de vértice fornecidos por gerador de fluxo contínuo de comando 803. Em algumas modalidades, o coletor de vértice 805 fornece dados de vértice para um sombreador de vértice 807, que executa operações de iluminação e transformação de espaço coordenada para cada vértice. Em algumas modalidades, o coletor de vértice 805 e o sombreador de vértice 807 executam instruções de processamento de vértice ao despachar encadeamentos de execução para unidades de execução 852A-852B por meio de um despachante de encadeamento 831.

[00137] Em algumas modalidades, as unidades de execução 852A-852B são uma matriz de processadores de vetor que têm um conjunto de instruções para executar operações de gráfico e mídia. Em algumas modalidades, as unidades de

execução 852A-852B têm um cache L1 afixado 851 que é específico para cada matriz ou compartilhado entre as matrizes. O cache pode ser configurado como um cache de dados, um cache de instrução ou um cache único que é particionado para conter dados e instruções em partições diferentes.

[00138] Em algumas modalidades, o pipeline de geometria 820 inclui componentes de tesselação para executar tesselação acelerada de hardware de objetos 3D. Em algumas modalidades, um sombreador hull programável 811 configura as operações de tesselação. Um sombreador de domínio programável 817 fornece avaliação de back-end de saída de tesselação. Um tesselador 813 opera na direção de sombreador hull 811 e contém lógica de propósito especial para gerar um conjunto de objetos geométricos detalhados com base em um modelo geométrico bruto que é fornecido como entrada para o pipeline de geometria 820. Em algumas modalidades, se a tesselação não for usada, componentes de tesselação (por exemplo, sombreador hull 811, o tesselador 813, e o sombreador de domínio 817) podem ser contornados.

[00139] Em algumas modalidades, objetos geométricos completos podem ser processados por um sombreador de geometria 819 por meio de um ou mais encadeamentos despachados para as unidades de execução 852A-852B, ou podem prosseguir diretamente para o limitador 829. Em algumas modalidades, o sombreador de geometria opera em objetos geométricos inteiros, em vez de em vértices ou remendos de vértices como em estágios anteriores do pipeline de gráficos. Se a tesselação for desabilitada,

o sombreador de geometria 819 recebe entrada do sombreador de vértice 807. Em algumas modalidades, o sombreador de geometria 819 é programável por um programa de sombreador de geometria para executar tesselação de geometria se as unidades de tesselação forem desabilitadas.

[00140] Antes da rasterização, um limitador 829 processa dados de vértice. O limitador 829 pode ser um limitador de função fixa ou um limitador programável que tem funções de sombreador de geometria e limitação. Em algumas modalidades, um rasterizador e componente de teste de profundidade 873 no pipeline de saída de renderização 870 despacha sombreadores de pixel para converter os objetos geométricos em representações por pixel. Em algumas modalidades, lógica de sombreador de pixel é incluída na lógica de execução de encadeamento 850. Em algumas modalidades, um aplicativo pode contornar o rasterizador e o componente de teste de profundidade 873 e acessar dados de vértice não rasterizados por meio de uma unidade de transmissão 823.

[00141] O processador de gráficos 800 tem um barramento de interconexão, uma malha de interconexão ou algum outro mecanismo de interconexão que permite que os dados e as mensagens passem entre os componentes principais do processador. Em algumas modalidades, as unidades de execução 852A-852B e unidades de lógica associadas (por exemplo, cache L1 851, amostrador 854, cache de textura 858, etc.) se interconectam por meio de uma porta de dados 856 para executar acesso de memória e se comunicar com componentes de pipeline de saída de renderizador do

processador. Em algumas modalidades, o amostrador 854, os caches 851, 858 e as unidades de execução 852A-852B cada um, têm, trajetórias separadas de acesso de memória. Em uma modalidade, o cache de textura 858 também pode ser configurado como um cache de amostrador.

[00142] Em algumas modalidades, o pipeline de saída de renderizador 870 contém um rasterizador e o componente de teste de profundidade 873 que converte objetos baseados em vértice em uma representação baseada em pixel associada. Em algumas modalidades, a lógica de rasterizador inclui uma unidade para colocar em janela/mascaradora para realizar rasterização de linha e triângulo de função fixa. Um cache de renderizador associado 878 e cache de profundidade 879 também estão disponíveis em algumas modalidades. Um componente de operações de pixel 877 executa operações baseadas em pixel nos dados, embora em alguns exemplos, operações de pixel associadas a operações 2D (por exemplo, transferências de imagem de bloco de bit com mescla) são realizadas pelo mecanismo 2D 841, ou substituídas em tempo de exibição pelo controlador de exibição 843 com o uso de planos de exibição sobrepostos. Em algumas modalidades, um cache L3 compartilhado 875 está disponível para todos os componentes de gráficos, permitindo o compartilhamento de dados sem o uso de memória de sistema principal.

[00143] Em algumas modalidades, o processador de pipeline de gráficos de mídia 830 inclui um mecanismo de mídia 837 e um front-end de vídeo 834. Em algumas modalidades, o front-end de vídeo 834 recebe comandos de

pipeline do gerador de fluxo contínuo de comando 803. Em algumas modalidades, o pipeline de mídia 830 inclui um gerador de fluxo contínuo de comando separado. Em algumas modalidades, o front-end de vídeo 834 processa comandos de mídia antes de enviar o comando para o mecanismo de mídia 837. Em algumas modalidades, mecanismo de mídia 837 o inclui funcionalidade de geração de encadeamento para gerar encadeamentos para despache para a lógica de execução de encadeamento 850 por meio de despachante de encadeamento 831.

[00144] Em algumas modalidades, o processador de gráficos 800 inclui um mecanismo de exibição 840. Em algumas modalidades, o mecanismo de exibição 840 é externo ao processador 800 e se acopla ao processador de gráficos por meio da interconexão de anel 802, ou algum outro barramento ou malha de interconexão. Em algumas modalidades, o mecanismo de exibição 840 inclui um mecanismo 2D 841 e um controlador de exibição 843. Em algumas modalidades, o mecanismo de exibição 840 contém lógica de propósito especial com capacidade de operar independentemente do pipeline 3D. Em algumas modalidades, o controlador de exibição 843 se acopla a um dispositivo de exibição (não mostrado), que pode ser um dispositivo de exibição integrado de sistema, como em um computador do tipo laptop, ou um dispositivo de exibição externo afixado por meio de um conector de dispositivo de exibição.

[00145] Em algumas modalidades, o pipeline de geometria 820 e o pipeline de mídia 830 são configuráveis para executar operações com base em múltiplos gráficos e

interface de programação de mídia e não são específicos a qualquer uma interface de programação de aplicativo (API). Em algumas modalidades, o software acionador para o processador de gráficos traduz chamadas de API que são específicas para uma biblioteca de gráficos ou de mídia particular em comandos que podem ser processados pelo processador de gráficos. Em algumas modalidades, o suporte é fornecido pela Biblioteca de Gráficos Aberta (OpenGL), Linguagem de Computação Aberta (OpenCL), e/ou gráficos Vulkan e API de computação, todos do grupo Khronos. Em algumas modalidades, o suporte também pode ser fornecido pela biblioteca Direct3D da Microsoft Corporation. Em algumas modalidades, uma combinação dessas bibliotecas pode ser suportada. O suporte também pode ser fornecido para a Biblioteca de Visão Computadorizada de Fonte Aberta (OpenCV). Uma API futura como um pipeline 3D compatível também seria suportada se um mapeamento pudesse ser realizado a partir do pipeline da API futura para o pipeline do processador de gráficos.

PROGRAMAÇÃO DE PIPELINE DE GRÁFICOS

[00146] A **Figura 9A** é um diagrama de blocos que ilustra um formato de comando de processador de gráficos 900 de acordo com algumas modalidades. A **Figura 9B** é um diagrama de blocos que ilustra um processador de gráficos sequência de comandos 910 de acordo com uma modalidade. As caixas com linha contínua na **Figura 9A** ilustram os componentes que são geralmente incluídos em um comando de gráficos enquanto as de linhas tracejadas incluem componentes que são opcionais ou que são somente

incluídos em um subconjunto dos gráficos comandos. O formato de comando de processador de gráficos exemplificativo 900 da **Figura 9A** inclui campos de dados para identificar um cliente 902, um código de operação de comando (código operacional) 904, e dados 906 para o comando. Um subcódigo operacional 905 e um tamanho de comando 908 também são incluídos em alguns comandos.

[00147] Em algumas modalidades, o cliente 902 especifica a unidade de cliente do dispositivo de gráficos que processa os dados de comando. Em algumas modalidades, um analisador de processador de comando de gráficos examina o campo de cliente de cada comando para condicionar o processamento adicional do comando e rotear os dados de comando para a unidade de cliente apropriada. Em algumas modalidades, as unidades de cliente de processador de gráficos incluem uma unidade de interface de memória, uma unidade de renderização, uma unidade 2D, uma unidade 3D e uma unidade de mídia. Cada unidade de cliente tem um pipeline de processamento correspondente que processa os comandos. Uma vez que o comando é recebido pela unidade de cliente, a unidade de cliente lê o código operacional 904 e, se presente, subcódigo operacional 905 para determinar a operação para se executar. A unidade de cliente executa o comando com o uso de informações em campo de dados 906. Para alguns comandos, um tamanho de comando explícito 908 é esperado para especificar o tamanho do comando. Em algumas modalidades, o analisador de comando determina automaticamente o tamanho de pelo menos alguns dos comandos com base no código operacional de comando. Em

algumas modalidades, os comandos são alinhados por meio de múltiplos de uma palavra dupla. Outros formatos de comando podem ser usados.

[00148] O fluxograma na **Figura 9B** ilustra um processador de gráficos sequência de comandos exemplificativo 910. Em algumas modalidades, o software ou o firmware de um sistema de processamento de dados que caracteriza uma modalidade do processador de gráficos usa uma versão da sequência de comando mostrada para estabelecer, executar e concluir um conjunto de operações de gráficos. Uma sequência de comando de amostra é mostrada e descrita apenas a título de exemplo, na medida em que modalidades não se limitam a esses comandos específicos ou a essa sequência de comando. Além disso, os comandos podem ser emitidos como lote de comandos em uma sequência de comando, de modo que o processador de gráficos processará a sequência de comandos em, pelo menos parcialmente, concomitância.

[00149] Em algumas modalidades, o processador de gráficos sequência de comandos 910 pode começar com um comando de liberação de pipeline 912 para fazer com que qualquer pipeline de gráficos ativo complete os comandos atualmente pendentes para o pipeline. Em algumas modalidades, o pipeline 3D 922 e o pipeline de mídia 924 não operam de modo concomitante. A liberação de pipeline é realizada para fazer com que o pipeline de gráficos ativo conclua quaisquer comandos pendentes. Em resposta a uma liberação de pipeline, o analisador de comando para o processador de gráficos interromperá o processamento de comando até que os mecanismos de

desenho ativos concluam as operações pendentes e os caches de leitura relevantes sejam invalidados. Opcionalmente, quaisquer dados no cache de renderização que sejam marcados “sujo” podem ser liberados para a memória. Em algumas modalidades, o comando de liberação de pipeline 912 pode ser usado para sincronização de pipeline ou antes de colocar o processador de gráficos em um estado de baixa potência.

[00150] Em algumas modalidades, um comando de seleção de pipeline 913 é usado quando uma sequência de comandos necessita do processador de gráficos para comutar explicitamente entre pipelines. Em algumas modalidades, um comando de seleção de pipeline 913 é necessário apenas uma vez dentro de um contexto de execução antes de emitir comandos de pipeline a mesmos que o contexto seja emitir comandos para ambos os pipelines. Em algumas modalidades, um comando de liberação de pipeline 912 é necessário imediatamente antes de uma comutação de pipeline por meio do comando de seleção de pipeline 913.

[00151] Em algumas modalidades, um comando de controle de pipeline 914 configura um pipeline de gráficos para operação e é usado para programar o pipeline 3D 922 e o pipeline de mídia 924. Em algumas modalidades, o comando de controle de pipeline 914 configura o estado de pipeline para o pipeline ativo. Em uma modalidade, o comando de controle de pipeline 914 é usado para sincronização de pipeline e para limpar dados de uma ou mais memórias de cache dentro do pipeline ativo antes do processamento de um lote de comandos.

[00152] Em algumas modalidades, comandos de estado de armazenamento temporário de retorno 916 são usados para configurar um conjunto de armazenamentos temporários de retorno para os respectivos pipelines para escrever dados. Algumas operações de pipeline exigem a alocação, seleção ou configuração de um ou mais armazenamentos temporários de retorno em que as operações gravam dados intermediário durante o processamento. Em algumas modalidades, o processador de gráficos também usa um ou mais armazenamentos temporários de retorno para armazenar dados de saída e para realizar comunicação de encadeamento cruzada. Em algumas modalidades, o estado de armazenamento temporário de retorno 916 inclui selecionar o tamanho e número de armazenamentos temporários de retorno para uso para um conjunto de operações de pipeline.

[00153] Os comandos remanescentes na sequência de comando diferem com base no pipeline ativo para operações. Com base em uma determinação de pipeline 920, a sequência de comandos é personalizada para o pipeline 3D 922 que começa com o estado de pipeline 3D 930 ou o pipeline de mídia 924 que começa no estado de pipeline de mídia 940.

[00154] Os comandos para configurar o estado de pipeline 3D 930 incluem comandos de definição de estado 3D para estado de armazenamento temporário de vértice, estado de elemento de vértice, estado de cor constante, estado de armazenamento temporário de profundidade e outras variáveis de estado que devem ser configuradas antes de comandos primitivos 3D serem processados. Os

valores desses comandos são determinados, pelo menos em parte, com base na API 3D particular em uso. Em algumas modalidades, os comandos de estado de pipeline 3D 930 também têm capacidade para desabilitar ou desviar seletivamente certos elementos de pipeline, caso esses elementos não sejam usados.

[00155] Em algumas modalidades, comando primitivo 3D 932 é usado para submeter primitivos 3D a serem processados pelo pipeline 3D. Os comandos e parâmetros associados que são passados para o processador de gráficos por meio do comando primitivo 3D 932 são encaminhados para a função de coleta de vértice no pipeline de gráficos. A função de coleta de vértice usa os dados de comando primitivo 3D 932 para gerar estruturas de dados de vértice. As estruturas de dados de vértice são armazenadas em um ou mais armazenamentos temporários de retorno. Em algumas modalidades, comando primitivo 3D 932 é usado para executar operações de vértice em primitivos 3D por meio de sombreadores de vértice. Para processar sombreadores de vértice, pipeline 3D 922 despacha encadeamentos de execução de sombreador para unidades de execução de processador de gráficos.

[00156] Em algumas modalidades, o pipeline 3D 922 é disparado por meio de um comando ou evento de execução 934. Em algumas modalidades, uma gravação de registro dispara a execução de comando. Em algumas modalidades, a execução é disparada por meio de um comando “ir” ou “pontapé” na sequência de comando. Em uma modalidade, a execução de comando é disparada com o uso de um comando

de sincronização de pipeline para liberar a sequência de comando através do pipeline de gráficos. O pipeline 3D realizará processamento de geometria para os primitivos 3D. Uma vez que operações sejam concluídas, os objetos geométricos resultantes são rasterizados e o mecanismo de pixel colore os pixels resultantes. Comandos adicionais para controlar operações de sombreamento de pixel e de extremidade posterior de pixel também podem ser incluídos para essas operações.

[00157] Em algumas modalidades, o processador de gráficos sequência de comandos 910 segue a trajetória de pipeline de mídia 924 quando realiza operações de mídia. Em geral, o uso específico e maneira de programar para o pipeline de mídia 924 dependem das operações de computação e de mídia para serem realizados. Operações de decodificação de mídia específicas podem ser descarregadas para o pipeline de mídia durante a decodificação de mídia. Em algumas modalidades, o pipeline de mídia também pode ser desviado e a decodificação de mídia pode ser realizada como um todo ou em parte com o uso de recursos fornecidos por um ou mais núcleos de processamento de propósito geral. Em uma modalidade, o pipeline de mídia também inclui elementos para operações de unidade de processador de propósito geral de gráficos (GPGPU), em que o processador de gráficos é usado para realizar operações de vetor de SIMD com o uso de programas de sombreador computacionais que não são explicitamente relacionados à renderização de primitivos de gráficos.

[00158] Em algumas modalidades, pipeline de mídia 924 é configurado de uma maneira similar ao pipeline 3D 922. Um conjunto de comandos para configurar o estado de pipeline de mídia 940 é despachado ou colocado em uma fila de comando antes dos comandos de objeto de mídia 942. Em algumas modalidades, comandos para o estado de pipeline de mídia 940 incluem dados para configurar os elementos de pipeline de mídia que serão usados para processar os objetos de mídia. Isso inclui dados para configurar a lógica de decodificação de vídeo e de codificação de vídeo dentro do pipeline de mídia, tal como formato de codificação ou de decodificação. Em algumas modalidades, comandos para o estado de pipeline de mídia 940 também suportam o uso de um ou mais indicadores para elementos de estado “indireto” que contêm um lote de configurações de estado.

[00159] Em algumas modalidades, os comandos de objeto de mídia 942 fornecem indicadores para objetos de mídia para processamento pelo pipeline de mídia. Os objetos de mídia incluem armazenamentos temporários de memória que contêm dados de vídeo a serem processados. Em algumas modalidades, todos os estados de pipeline de mídia devem ser válidos antes de emitir um comando de objeto de mídia 942. Uma vez que o estado de pipeline é configurado e comandos de objeto de mídia 942 são listados, o pipeline de mídia 924 é disparado por meio de um comando de execução 944 ou um evento de execução equivalente (por exemplo, escrita de registrador). A saída do pipeline de mídia 924 pode, então, ser pós-processada por operações fornecidas pelo pipeline 3D 922

ou pelo pipeline de mídia 924. Em algumas modalidades, operações de GPGPU são configuradas e executadas de uma maneira similar às operações de mídia.

ARQUITETURA DE SOFTWARE DE GRÁFICOS

[00160] A **Figura 10** ilustra uma arquitetura de software de gráficos exemplificativa para um sistema de processamento de dados 1000 de acordo com algumas modalidades. Em algumas modalidades, a arquitetura de software inclui um aplicativo de gráficos 3D 1010, um sistema operacional 1020, e pelo menos um processador 1030. Em algumas modalidades, o processador 1030 inclui um processador de gráficos 1032 e um ou mais núcleos de processador de propósito geral 1034. O aplicativo de gráficos 1010 e o sistema operacional 1020, cada um, são executados na memória de sistema 1050 do sistema de processamento de dados.

[00161] Em algumas modalidades, o aplicativo de gráficos 3D 1010 contém um ou mais programas de sombreador incluindo instruções de sombreador 1012. As instruções de linguagem de sombreador podem estar em uma linguagem de sombreador de alto nível, tal como a Linguagem de Sombreador de Alto Nível (HLSL) de Direct3D, a Linguagem de Sombreador OpenGL (GLSL), e assim por diante. O aplicativo também inclui instruções executáveis 1014 em uma linguagem de máquina adequada para execução pelo núcleo de processador de propósito geral 1034. O aplicativo também inclui objetos de gráficos 1016 definidos por dados de vértice.

[00162] Em algumas modalidades, o sistema operacional 1020 é um sistema operacional Microsoft® Windows®

disponível junto à Microsoft Corporation, um sistema operacional como UNIX proprietário, ou um sistema operacional como UNIX de fonte aberta com o uso de uma variante do kernel da Linux. O sistema operacional 1020 pode suportar uma API de gráficos 1022 como a API Direct3D, a API OpenGL, ou a API Vulkan. Quando a API Direct3D está em uso, o sistema operacional 1020 usa um compilador de sombreador de front-end 1024 para compilar quaisquer instruções de sombreador 1012 em HLSL em uma linguagem de sombreador de nível inferior. A compilação pode ser uma compilação no tempo certo ou o aplicativo pode realizar pré-compilação de sombreador. Em algumas modalidades, sombreadores de alto nível são compilados em sombreadores de baixo nível durante a compilação do aplicativo de gráficos 3D 1010. Em algumas modalidades, as instruções de sombreador 1012 são fornecidas em uma forma intermediária, como uma versão da Representação Intermediária Portátil Padrão (SPIR) usada pela API Vulkan.

[00163] Em algumas modalidades, acionador de gráficos de modo de usuário 1026 contém a compilador de sombreador de back-end 1027 para converter as instruções de sombreador 1012 em uma representação específica de hardware. Quando a API OpenGL está em uso, instruções de sombreador 1012 na linguagem de alto nível de GLSL são passadas para um acionador de gráficos de modo de usuário 1026 para compilação. Em algumas modalidades, o acionador de gráficos de modo de usuário 1026 usa funções de modo de kernel de sistema operacional 1028 para se comunicar com um acionador de gráficos de modo

de kernel 1029. Em algumas modalidades, o acionador de gráficos de modo de kernel 1029 se comunica com processador de gráficos 1032 para despachar comandos e instruções.

IMPLANTAÇÕES DE NÚCLEO DE IP

[00164] Um ou mais aspectos da pelo menos uma modalidade podem ser implantados por código representativo armazenado em uma mídia legível por máquina que representa e/ou define lógica dentro de um circuito integrado, tal como um processador. Por exemplo, a mídia legível por máquina pode incluir instruções que representam várias lógicas dentro do processador. Quando lidas por uma máquina, as instruções podem fazer a máquina fabricar a lógica para realizar as técnicas descritas no presente documento. Tais representações, conhecidas como “núcleos de IP”, são unidades reutilizáveis da lógica para um circuito integrado que pode ser armazenado em uma mídia legível por máquina tangível como um modelo de hardware que descreve a estrutura do circuito integrado. O modelo de hardware pode ser fornecido aos vários clientes ou instalações de fabricação, que carregam o modelo de hardware em máquinas de fabricação que fabricam o circuito integrado. O circuito integrado pode ser fabricado de modo que o circuito realize operações descritas em associação com qualquer uma das modalidades descritas no presente documento.

[00165] A **Figura 11A** é um diagrama de blocos que ilustra um sistema de desenvolvimento de núcleo de IP 1100 que pode ser usado para fabricar um circuito

integrado para executar operações de acordo com uma modalidade. O sistema de desenvolvimento de núcleo de IP 1100 pode ser usado para gerar projetos reutilizáveis, modulares que podem ser incorporados a um projeto maior ou usados para construir um circuito integrado completo (por exemplo, um circuito integrado de SOC). Uma instalação de projeto 1130 pode gerar uma simulação de software 1110 de um projeto de núcleo de IP em uma linguagem de programação de alto nível (por exemplo, C/C++). A simulação de software 1110 pode ser usada para projetar, testar e verificar o comportamento do núcleo de IP com o uso de um modelo de simulação 1112. O modelo de simulação 1112 pode incluir simulações funcionais, comportamentais e/ou temporais. Um projeto de nível de transferência de registro (RTL) 1115 pode, então, ser criado ou sintetizado a partir do modelo de simulação 1112. O projeto de RTL 1115 é uma abstração do comportamento do circuito integrado que modela o fluxo de sinais digitais entre registros de hardware, incluindo a lógica associada realizada com o uso dos sinais digitais modelados. Além de um projeto de RTL 1115, projetos de nível inferior no nível de lógica ou nível de transistor também podem ser criados, projetados ou sintetizados. Assim, os detalhes particulares do projeto inicial e simulação podem variar.

[00166] O projeto de RTL 1115 ou equivalente pode ser ainda sintetizado pela instalação de projeto em um modelo de hardware 1120, que pode estar em uma linguagem de descrição de hardware (HDL), ou alguma outra representação de dados de projeto físico. A HDL pode ser

adicionalmente simulado ou testado para verificar o projeto de núcleo de IP. O projeto de núcleo de IP pode ser armazenado para entrega para uma instalação de fabricação terceirizada 1165 com o uso de memória não volátil 1140 (por exemplo, disco rígido, memória flash, ou qualquer mídia de armazenamento não volátil). De modo alternativo, o projeto de núcleo de IP pode ser transmitido (por exemplo, por meio da Internet) em uma conexão com fio 1150 ou conexão sem fio 1160. A instalação de fabricação 1165 pode, então, fabricar um circuito integrado que tem como base, pelo menos em parte, o projeto de núcleo de IP. O circuito integrado fabricado pode ser configurado para realizar operações de acordo com pelo menos uma modalidade descrita no presente documento.

[00167] A **Figura 11B** ilustra uma vista lateral em corte transversal de uma montagem de pacote de circuito integrado 1170, de acordo com algumas modalidades descritas no presente documento. A montagem de pacote de circuito integrado 1170 ilustra uma implantação de um ou mais dispositivos de processador ou acelerador, conforme descrito no presente documento. A montagem de pacote 1170 inclui múltiplas unidades de lógica de hardware 1172, 1174 conectada a um substrato 1180. A lógica 1172, 1174 pode ser implantada pelo menos parcialmente em hardware de lógica configurável ou lógica de funcionalidade fixa, e pode incluir uma ou mais porções de qualquer um dentre o núcleo (ou núcleos) de processador, processador (ou processadores) de gráficos, ou outros dispositivos de acelerador descritos no

presente documento. Cada unidade de lógica 1172, 1174 pode ser implantada dentro de um molde semicondutor e acoplada ao substrato 1180 por meio de uma estrutura de interconexão 1173. A estrutura de interconexão 1173 pode ser configurada para rotear sinais elétricos entre a lógica 1172, 1174 e o substrato 1180, e pode incluir interconexões como, mas não limitadas a saliências ou pilares. Em algumas modalidades, a estrutura de interconexão 1173 pode ser configurada para rotear sinais elétricos como, por exemplo, sinais de entrada/saída (I/O) e/ou sinais de solo ou potência associados à operação da lógica 1172, 1174. Em algumas modalidades, o substrato 1180 é um substrato de laminado à base de epóxi. O substrato 1180 pode incluir outros tipos adequados de substratos em outras modalidades. A montagem de pacote 1170 pode ser conectada a outros dispositivos elétricos por meio de uma interconexão de pacote 1183. A interconexão de pacote 1183 pode ser acoplada a uma superfície do substrato 1180 para rotear sinais elétricos para outros dispositivos elétricos, como uma placa-mãe, outros conjuntos de chips, ou módulo de múltiplos chips.

[00168] Em algumas modalidades, as unidades de lógica 1172, 1174 são eletricamente acopladas a uma ponte 1182 que é configurada para rotear sinais elétricos entre a lógica 1172, 1174. A ponte 1182 pode ser uma estrutura de interconexão densa que fornece uma rota para sinais elétricos. A ponte 1182 pode incluir um substrato de ponte composto de vidro ou um material semicondutor adequado. Os recursos de roteamento elétrico podem ser

formados no substrato de ponte para fornecer uma conexão chip para chip entre a lógica 1172, 1174.

[00169] Embora duas unidades de lógica 1172, 1174 e uma ponte 1182 sejam ilustradas, modalidades descritas no presente documento podem incluir mais ou menos unidades de lógica em um ou mais moldes. O um ou mais moldem podem ser conectados por zero ou mais pontes, uma vez que a ponte 1182 pode ser excluída quando a lógica é incluída em um único molde. Alternativamente, múltiplos moldes ou unidades de lógica podem ser conectados por uma ou mais pontes. Adicionalmente, múltiplas unidades de lógica, moldes e pontes podem ser conectados juntos em outras configurações possíveis que incluem configurações tridimensionais.

[00170] A **Figura 11C** ilustra uma montagem de pacote 1190 que inclui múltiplas unidades de chiplets de lógica de hardware conectados a um substrato 1180 (por exemplo, molde de base). Uma unidade de processamento de gráficos, processador paralelo e/ou acelerador de computação, conforme descrito no presente documento, pode ser composto de chiplets de silício diversos que são separadamente fabricados. Nesse contexto, um chiplet é um circuito integrado pelo menos parcialmente empacotado que inclui unidades distintas de lógica que podem ser montadas com outros chiplets em um pacote maior. Um diverso conjunto de chiplets com diferente lógica de núcleo de IP pode ser montado em um único dispositivo. Adicionalmente, os chiplets podem ser integrados em um molde de base ou chiplet de base com o uso de tecnologia de interposição ativa. Os conceitos

descritos no presente documento permitem a interconexão e comunicação entre as diferentes formas de IP dentro da GPU. Núcleos de IP podem ser fabricados com o uso de diferentes tecnologias de processo e compostos durante a fabricação, o que evita a complexidade de convergir múltiplos IPs, especialmente em um grande SoC com diversos tipos de IPs, no mesmo processo de fabricação. Permitir o uso de múltiplas tecnologias de processo aumenta o tempo de comercialização e fornece uma maneira rentável de criar múltiplas SKUs de produto. Adicionalmente, os IPs desagregados são mais passíveis de serem acionados por potência independentemente, componentes que não estão em uso em uma dada carga de trabalho podem ser desligados, o que reduz o consumo de potência geral.

[00171] Os chiplets de lógica de hardware podem incluir chiplets de lógica de hardware de propósito geral 1172, chiplets de lógica ou I/O 1174, e/ou chiplets de memória 1175. Os chiplets de lógica de hardware 1172 e chiplets de lógica ou I/O 1174 podem ser implantados pelo menos parcialmente em hardware de lógica configurável ou lógica de funcionalidade fixa e podem incluir uma ou mais porções de qualquer um dentre o núcleo (ou núcleos) de processador, processador (ou processadores) de gráficos, processadores paralelos, ou outros dispositivos de acelerador descritos no presente documento. Os chiplets de memória 1175 podem ser memória DRAM (por exemplo, GDDR, HBM) ou memória cache (SRAM).

[00172] Cada chiplet pode ser fabricado como molde semicondutor separado e acoplado ao substrato 1180 por

meio de uma estrutura de interconexão 1173. A estrutura de interconexão 1173 pode ser configurada para rotear sinais elétricos entre os diversos chiplets e lógica dentro do substrato 1180. A estrutura de interconexão 1173 pode incluir interconexões como, mas não limitadas a saliências ou pilares. Em algumas modalidades, a estrutura de interconexão 1173 pode ser configurada para rotear sinais elétricos como, por exemplo, sinais de entrada/saída (I/O) e/ou sinais de solo ou potência associados à operação da lógica, I/O e chiplets de memória.

[00173] Em algumas modalidades, o substrato 1180 é um substrato de laminado à base de epóxi. O substrato 1180 pode incluir outros tipos adequados de substratos em outras modalidades. A montagem de pacote 1190 pode ser conectada a outros dispositivos elétricos por meio de uma interconexão de pacote 1183. A interconexão de pacote 1183 pode ser acoplada a uma superfície do substrato 1180 para rotear sinais elétricos para outros dispositivos elétricos, como uma placa-mãe, outros conjuntos de chips, ou módulo de múltiplos chips.

[00174] Em algumas modalidades, um chiplet de lógica ou I/O 1174 e um chiplet de memória 1175 podem ser eletricamente acoplados por meio de uma ponte 1187 que é configurada para rotear sinais elétricos entre o chiplet de lógica ou I/O 1174 e um chiplet de memória 1175. A ponte 1187 pode ser uma estrutura de interconexão densa que fornece uma rota para sinais elétricos. A ponte 1187 pode incluir um substrato de ponte composto de vidro ou um material semicondutor adequado. Os recursos de

roteamento elétrico podem ser formados no substrato de ponte para fornecer uma conexão chip para chip entre o chiplet de lógica ou I/O 1174 e um chiplet de memória 1175. A ponte 1187 também pode ser chamada de uma ponte de silício ou uma ponte de interconexão. Por exemplo, a ponte 1187, em algumas modalidades, é uma Ponte de Interconexão de Múltiplos Moldes Embutidos (EMIB). Em algumas modalidades, a ponte 1187 pode ser simplesmente uma conexão direta de um chiplet para outro chiplet.

[00175] O substrato 1180 pode incluir componentes de hardware para I/O 1191, memória cache 1192, e outra lógica de hardware 1193. Uma malha 1185 pode ser embutida no substrato 1180 para possibilitar comunicação entre os diversos chiplets de lógica e a lógica 1191, 1193 dentro do substrato 1180. Em uma modalidade, o I/O 1191, a malha 1185, cache, ponte, e outra lógica de hardware 1193 podem ser integrados em um molde de base que é estratificada no topo do substrato 1180.

[00176] Em diversas modalidades, uma montagem de pacote 1190 pode incluir mais ou menos número de componentes e chiplets que são interconectados por uma malha 1185 ou uma ou mais pontes 1187. Os chiplets dentro da montagem de pacote 1190 podem ser dispostos em uma disposição 3D ou 2,5D. Em geral, estruturas de ponte 1187 podem ser usadas para facilitar uma interconexão ponto a ponto entre, por exemplo, chiplets de lógica ou I/O e chiplets de memória. A malha 1185 pode ser usada para interconectar os diversos chiplets de lógica e/ou I/O (por exemplo, chiplets 1172, 1174, 1191, 1193), com outros chiplets de lógica e/ou I/O. Em uma modalidade, a

memória cache 1192 dentro do substrato pode atuar como um cache global para a montagem de pacote 1190, parte de um cache global distribuído, ou como um cache dedicado para a malha 1185.

[00177] A **Figura 11D** ilustra uma montagem de pacote 1194, incluindo chiplets intercambiáveis 1195, de acordo com uma modalidade. Os chiplets intercambiáveis 1195 podem ser montados em fendas padronizadas em um ou mais chiplets de base 1196, 1198. Os chiplets de base 1196, 1198 podem ser acoplados por meio de uma interconexão de ponte 1197, que pode ser similar às outras interconexões de ponte descritas no presente documento e pode ser, por exemplo, uma EMIB. Chiplets de memória também podem estar conectados aos chiplets de lógica ou I/O por meio de uma interconexão de ponte. Chiplets de I/O e lógica podem se comunicar por meio de uma malha de interconexão. Os chiplets de base podem, cada um, suportar uma ou mais fendas em um formato padronizado para um dentre lógica ou I/O ou memória/cache.

[00178] Em uma modalidade, SRAM e circuitos de entrega de potência podem ser fabricados em um ou mais dos chiplets de base 1196, 1198, que podem ser fabricados com o uso de uma tecnologia de processo diferente em relação aos chiplets intercambiáveis 1195 que são empilhados no topo dos chiplets de base. Por exemplo, os chiplets de base 1196, 1198 podem ser fabricados com o uso de uma tecnologia de processo maior, enquanto os chiplets intercambiáveis podem ser fabricados com o uso de uma tecnologia de processo menor. Um ou mais dos chiplets intercambiáveis 1195 podem ser chiplets de

memória (por exemplo, DRAM). As densidades diferentes de memória podem ser selecionadas para a montagem de pacote 1194 com base na potência, e/ou desempenho alvejado para o produto que usa a montagem de pacote 1194. Adicionalmente, chiplets de lógica com um diferente número de tipos de unidades funcionais podem ser selecionados no momento de montagem com base na potência e/ou desempenho alvejado para o produto. Adicionalmente, chiplets que contêm núcleos de lógica de IP de diferentes tipos podem ser inseridos nas fendas de chiplet intercambiável, o que permite projetos de processador híbridos que podem misturar e corresponder diferentes blocos de IP de tecnologia.

SISTEMA EXEMPLIFICATIVO EM UM CIRCUITO INTEGRADO DE CHIP

[00179] As **Figuras 12 a 13** ilustram circuitos integrados exemplificativos e processadores de gráficos associados que podem ser fabricados com o uso de um ou mais núcleos de IP, de acordo com várias modalidades descritas no presente documento. Adicionalmente ao que é ilustrado, outra lógica e circuitos podem ser incluídos, que incluem processadores de gráficos/núcleos adicionais, controladores de interface periféricos ou núcleos de processador de propósito geral.

[00180] A **Figura 12** é um diagrama de blocos que ilustra um sistema exemplificativo em um circuito integrado de chip 1200 que pode ser fabricado com o uso de um ou mais núcleos de IP, de acordo com uma modalidade. O circuito integrado exemplificativo 1200 inclui um ou mais processadores de aplicativo 1205 (por exemplo, CPUs), pelo menos um processador de gráficos 1210, e pode

incluir adicionalmente um processador de imagem 1215 e/ou um processador de vídeo 1220, qualquer um dos quais pode ser um núcleo de IP modular das mesmas ou múltiplas instalações de processo diferentes. O circuito integrado 1200 inclui lógica de barramento ou periférica, incluindo um controlador de USB 1225, controlador de UART 1230, um controlador de SPI/SDIO 1235, e um controlador de I^2S/I^2C 1240. Adicionalmente, o circuito integrado pode incluir um dispositivo de exibição 1245 acoplado a um ou mais de um controlador de interface de multimídia de alta definição (HDMI) 1250 e uma interface de exibição de interface de processador de indústria móvel (MIPI) 1255. O armazenamento pode ser fornecido por um subsistema de memória flash 1260, incluindo memória flash e um controlador de memória flash. A interface de memória pode ser fornecida por meio de um controlador de memória 1265 para acesso a dispositivos de memória SDRAM ou SRAM. Alguns circuitos integrados adicionalmente incluem um mecanismo de segurança embutido 1270.

[00181] As **Figuras 13A a 13B** são diagramas de blocos que ilustram processadores de gráficos exemplificativos para uso dentro de um SoC, de acordo com modalidades descritas no presente documento. A **Figura 13A** ilustra um processador de gráficos exemplificativo 1310 de um sistema em um circuito integrado de chip que pode ser fabricado com o uso de um ou mais núcleos de IP, de acordo com uma modalidade. A **Figura 13B** ilustra um processador de gráficos exemplificativo adicional 1340 de um sistema em um circuito integrado de chip que pode

ser fabricado com o uso de um ou mais núcleos de IP, de acordo com uma modalidade. O processador de gráficos 1310 da **Figura 13A** é um exemplo de um núcleo de processador de gráficos de baixa potência. O processador de gráficos 1340 da **Figura 13B** é um exemplo de um núcleo de processador de gráficos de desempenho superior. Cada um dos processadores de gráficos 1310, 1340 pode ser variantes do processador de gráficos 1210 da **Figura 12.**

[00182] Como mostrado na **Figura 13A**, o processador de gráficos 1310 inclui um processador de vértice 1305 e um ou mais processadores de fragmento 1315A–1315N (por exemplo, 1315A, 1315B, 1315C, 1315D, a 1315N–1, e 1315N). O processador de gráficos 1310 pode executar diferentes programas de sombreador por meio de lógica separada, de modo que o processador de vértice 1305 seja otimizado para executar operações para programas de sombreador de vértice, enquanto o um ou mais processadores de fragmento 1315A–1315N executam operações de sombreamento de fragmento (por exemplo, pixel) para programas de sombreador de fragmento ou pixel. O processador de vértice 1305 executa o estágio de processamento de vértice do pipeline de gráficos 3D e gera dados de vértice e primitivos. O processador (ou processadores) de fragmento 1315A–1315N usa os dados de vértice e primitivos gerados pelo processador de vértice 1305 para produzir um armazenamento temporário de quadro que é exibido em um dispositivo de exibição. Em uma modalidade, o processador (ou processadores) de fragmento 1315A–1315N é otimizado para executar programas de sombreador de fragmento, como fornecido na

API OpenGL, que pode ser usada para executar operações similares como um programa de sombreador de pixel, como fornecido na API Direct 3D.

[00183] O processador de gráficos 1310 inclui adicionalmente uma ou mais unidades de gerenciamento de memória (MMUs) 1320A-1320B, cache (ou caches) 1325A-1325B, e interconexão (ou interconexões) de circuito) 1330A-1330B. A um ou mais MMUs 1320A-1320B fornecem mapeamento de endereço virtual para físico para o processador de gráficos 1310, incluindo para o processador de vértice 1305 e/ou processador (ou processadores) de fragmento 1315A-1315N, que podem fazer referência a dados de vértice ou imagem/textura armazenados na memória, além de dados de vértice ou imagem/textura armazenados no um ou mais caches 1325A-1325B. Em uma modalidade, a uma ou mais MMUs 1320A-1320B podem ser sincronizadas com outras MMUs dentro do sistema, incluindo uma ou mais MMUs associadas ao um ou mais processadores de aplicativo 1205, processador de imagem 1215, e/ou processador de vídeo 1220 da **Figura 12**, de modo que cada processador 1205-1220 possa participar em um sistema de memória virtual compartilhado ou unificado. A um ou mais interconexões de circuito 1330A-1330B possibilitam que o processador de gráficos 1310 faça interface com outros núcleos de IP dentro do SoC, ou por meio de um barramento interno do SoC ou por meio de uma conexão direta, de acordo com modalidades.

[00184] Como mostrado na **Figura 13B**, o processador de gráficos 1340 inclui a uma ou mais MMUs 1320A-1320B,

cache (ou caches) 1325A-1325B, e interconexão (ou interconexões) de circuito 1330A-1330B do processador de gráficos 1310 da **Figura 13A**. O processador de gráficos 1340 inclui um ou mais núcleos de sombreador 1355A-1355N (por exemplo, 1455A, 1355B, 1355C, 1355D, 1355E, 1355F, através de 1355N-1, e 1355N), que fornecem uma arquitetura de núcleo de sombreador unificada na qual um único núcleo ou tipo ou núcleo pode executar todos os tipos de código de sombreador programável, incluindo código de programa de sombreador para implantar sombreadores de vértice, sombreadores de fragmento, e/ou sombreadores de computação. O número exato de núcleos de sombreador presentes pode variar entre as modalidades e implantações. Adicionalmente, o processador de gráficos 1340 inclui um gerenciador de tarefas entre núcleos 1345, que atua como a despachante de encadeamento para despachar encadeamentos de execução para um ou mais núcleos de sombreador 1355A-1355N e uma unidade de agrupamento 1358 para acelerar operações de agrupamento para renderização baseada em agrupamento, nas quais operações de renderização para uma cena são subdivididas em espaço de imagem, por exemplo, para explorar coerência espacial local dentro de uma cena ou para otimizar uso de caches internos.

VISÃO GERAL DE APRENDIZADO POR MÁQUINA

[00185] Um algoritmo de aprendizado por máquina é um algoritmo que pode aprender com base em um conjunto de dados. As modalidades de algoritmos de aprendizado por máquina podem ser projetadas para modelar abstrações de alto nível dentro de um conjunto de dados. Por exemplo,

algoritmos de reconhecimento de imagem podem ser usados para determinar a qual de diversas categorias uma determinada entrada pertence; algoritmos de regressão podem emitir um valor numérico determinado de uma entrada; e algoritmos de reconhecimento de padrão podem ser usados para gerar texto traduzido ou realizar reconhecimento de discurso e/ou texto para discurso.

[00186] Um tipo exemplificativo de algoritmo de aprendizado por máquina é uma rede neural. Há muitos tipos de redes neurais; um tipo simples de rede neural é uma rede de encaminhamento. Uma rede de encaminhamento pode ser implantada como um gráfico acíclico no qual os nós são dispostos em camadas. Tipicamente, uma topologia de rede de encaminhamento inclui uma camada de entrada e uma camada de saída que são separadas por pelo menos uma camada ocultada. A camada ocultada transforma entrada recebida pela camada de entrada em uma representação que é útil para gerar saída na camada de saída. Os nós de rede são completamente conectados por meio de bordas aos nós em camadas adjacentes, mas não há bordas entre nós dentro de cada camada. Os dados recebidos nos nós de uma camada de entrada de uma rede de encaminhamento são propagados (isto é, “encaminhados”) para os nós da camada de saída por meio de uma função de ativação que calcula os estados dos nós de cada camada sucessiva na rede com base em coeficientes (“pesos”) respectivamente associados a cada uma das bordas que conectam as camadas. Dependendo do modelo específico que é representado pelo algoritmo que é executado, a saída do algoritmo de rede neural pode assumir diversas formas.

[00187] Antes de um algoritmo de aprendizado por máquina poder ser usado para modelar um problema particular, o algoritmo é treinado com o uso de um conjunto de dados de treinamento. O treinamento de uma rede neural envolve selecionar uma topologia de rede, com o uso de um conjunto de dados de treinamento que representam um problema que é modelado pela rede, e ajustar os pesos até que o modelo de rede execute com um erro mínimo todos os exemplos do conjunto de dados de treinamento. Por exemplo, durante um processo de treinamento de aprendizado supervisionado para uma rede neural, a saída produzida pela rede em resposta à entrada que representa um exemplo em um conjunto de dados de treinamento é comparada à saída etiquetada como "correta" para esse exemplo, um sinal de erro que representa a diferença entre a saída e a saída etiquetada é calculado, e os pesos associados às conexões são ajustados para minimizar esse erro conforme o sinal de erro é propagado para trás através das camadas da rede. A rede é considerada “treinada” quando os erros para cada uma das saídas geradas dos exemplos do conjunto de dados de treinamento são minimizados.

[00188] A precisão de um algoritmo de aprendizado por máquina pode ser afetada significativamente pela qualidade do conjunto de dados usado para treinar o algoritmo. O processo de treinamento pode ser computacionalmente intensivo e pode necessitar de uma quantidade significativa de tempo em um processador de propósito geral convencional. Consequentemente, hardware de processamento paralelo é usado para treinar muitos

tipos de algoritmos de aprendizado por máquina. Isso é particularmente útil para otimizar o treinamento de redes neurais, uma vez que as computações realizadas no ajuste dos coeficientes em redes neurais realizam por si só naturalmente as implantações paralelas. Especificamente, muitos algoritmos de aprendizado por máquina e aplicativos de software foram adaptados para fazer uso do hardware de processamento paralelo dentro de dispositivos de processamento de gráficos de propósito geral.

[00189] A **Figura 14** é um diagrama generalizado de uma pilha de software de aprendizado por máquina 1400. Um aplicativo de aprendizado por máquina 1402 pode ser configurado para treinar uma rede neural com o uso de um conjunto de dados de treinamento ou para usar uma rede neural profunda treinada para implantar inteligência de máquina. O aplicativo de aprendizado por máquina 1402 pode incluir treinamento e funcionalidade de inferência para uma rede neural e/ou software especializado que pode ser usado para treinar uma rede neural antes da implantação. O aplicativo de aprendizado por máquina 1402 pode implantar qualquer tipo de inteligência de máquina, incluindo, mas não limitado a reconhecimento de imagem, mapeamento e localização, navegação autônoma, síntese de discurso, imageamento médico ou tradução de linguagem.

[00190] A aceleração de hardware para o aplicativo de aprendizado por máquina 1402 pode ser possibilitada por meio de uma estrutura de aprendizado por máquina 1404. A estrutura de aprendizado por máquina 1404 pode fornecer

uma biblioteca de primitivos de aprendizado por máquina. Os primitivos de aprendizado por máquina são operações básicas que são comumente realizadas por algoritmos de aprendizado por máquina. Sem a estrutura de aprendizado por máquina 1404, desenvolvedores de algoritmos de aprendizado por máquina seria necessário criar e otimizar a lógica computacional principal associada ao algoritmo de aprendizado por máquina, então, reotimizar a lógica computacional conforme novos processadores paralelos são desenvolvidos. Em vez disso, o aplicativo de aprendizado por máquina pode ser configurado para executar as computações necessárias com o uso dos primitivos fornecidos pela estrutura de aprendizado por máquina 1404. Os primitivos exemplificativos incluem convolações de tensor, funções de ativação, e agrupamento, que são operações computacionais que são realizadas enquanto ocorre treinamento de uma rede neural convolucional (CNN). A estrutura de aprendizado por máquina 1404 também pode fornecer primitivos para implantar subprogramas de álgebra linear básica realizados por muitos algoritmos de aprendizado por máquina, como operações de vetor e matriz.

[00191] A estrutura de aprendizado por máquina 1404 pode processar dados de entrada recebidos do aplicativo de aprendizado por máquina 1402 e gerar a entrada apropriada para uma estrutura de computação 1406. A estrutura de computação 1406 pode resumir as instruções subjacentes fornecidas para o acionador de GPGPU 1408 para possibilitar que a estrutura de aprendizado por máquina 1404 tenha a vantagem de aceleração de hardware

por meio do hardware de GPGPU 1410 sem precisar que a estrutura de aprendizado por máquina 1404 tenha conhecimento profundo da arquitetura do hardware de GPGPU 1410. Adicionalmente, a estrutura de computação 1406 pode possibilitar aceleração de hardware para a estrutura de aprendizado por máquina 1404 em uma variedade de tipos e gerações do hardware de GPGPU 1410.

IMPLANTAÇÕES DE REDE NEURAL DE APRENDIZADO POR MÁQUINA

[00192] A arquitetura de computação fornecida por modalidades descritas no presente documento pode ser configurada para executar os tipos de processamento paralelo que são particularmente adequados para treinamento e implantação de redes neurais para aprendizado por máquina. Uma rede neural pode ser generalizada como uma rede de funções que tem uma relação de gráfico. Sabe-se na técnica que há uma variedade de tipos de implantações de rede neural usadas em aprendizado por máquina. Um tipo exemplificativo de rede neural é a rede de encaminhamento, como anteriormente descrito.

[00193] Um segundo tipo exemplificativo de rede neural é a Rede Neural Convolucional (CNN). Uma CNN é uma rede de encaminhamento neural especializada para processar dados que têm uma topologia similar à grade, conhecida, como dados de imagem. Consequentemente, CNNs são comumente usadas para aplicativos de visão de computação e reconhecimento de imagem, mas as mesmas também podem ser usadas para outros tipos de reconhecimento de padrão, como processamento de linguagem e discurso. Os nós na camada de entrada de CNN são organizados em um

conjunto de “filtros” (detectores de recurso inspirados pelos campos receptores encontrados na retina), e a saída de cada conjunto de filtros é propagada para os nós em camadas sucessivas da rede. As computações para uma CNN incluem aplicar a operação matemática de convolução a cada filtro para produzir a saída desse filtro. A convolução é um tipo especializado de operação matemática realizada por duas funções para produzir uma terceira função que é uma versão modificada de uma das duas funções originais. Em terminologia de rede convolucional, a primeira função para a convolução pode ser chamada de a entrada, enquanto a segunda função pode ser chamada de o kernel de convolução. A saída pode ser chamada de o mapa de recurso. Por exemplo, a entrada para uma camada de convolução pode ser uma matriz multidimensional de dados que define os diversos componentes de cor de uma imagem de entrada. O kernel de convolução pode ser uma matriz multidimensional de parâmetros, em que os parâmetros são adaptados pelo processo de treinamento para a rede neural.

[00194] As redes neurais recorrentes (RNNs) são uma família de redes neurais encaminhadas que incluem conexões de retroalimentação entre as camadas. RNNs possibilitam modelagem de dados sequenciais por dados de parâmetro de compartilhamento em diferentes partes da rede neural. A arquitetura for uma RNN inclui ciclos. Os ciclos representam a influência de um valor presente de uma variável em seu próprio valor em um tempo futuro, uma vez que pelo menos uma porção dos dados de saída da RNN são usados como retroalimentação para processar

entrada subsequente em uma sequência. Esse recurso torna RNNs particularmente úteis ou processamento de linguagem devido à natureza variável na qual dados de linguagem podem ser compostos.

[00195] As figuras descritas abaixo apresentam redes de encaminhamento, CNN e RNN exemplificativas, bem como descrevem um processo geral para, respectivamente, treinamento e implantação de cada um desses tipos de redes. Será entendido que essas descrições são exemplificativas e não limitantes de modo que qualquer modalidade específica descrita no presente documento e os conceitos ilustrados possam ser aplicados, em geral, a redes neurais profundas e técnicas de aprendizado por máquina em geral.

[00196] As redes neurais exemplificativas descritas acima podem ser usadas para executar aprendizado profundo. O aprendizado profundo é aprendizado por máquina com o uso de redes neurais profundas. As redes neurais profundas usadas em aprendizado profundo são redes neurais artificiais compostas de múltiplas camadas ocultadas, em oposição a redes neurais superficiais que incluem apenas uma única camada ocultada. As redes neurais mais profundas são, em geral, mais computacionalmente intensivas para treinar. No entanto, as camadas ocultadas adicionais da rede possibilitam reconhecimento de padrão de múltiplas etapas que resulta em erro de saída reduzido em relação a técnicas de aprendizado por máquina superficiais.

[00197] As redes neurais profundas usadas em aprendizado profundo tipicamente incluem uma rede de

front-end para executar reconhecimento de recurso acoplada a uma rede de back-end que representa um modelo matemático que pode executar operações (por exemplo, classificação de objeto, reconhecimento de discurso, etc.) com base na representação de recurso fornecida par ao modelo. O aprendizado profundo possibilita que o aprendizado por máquina seja realizado sem necessitar de engenharia de recurso manualmente realizado a ser realizada para o modelo. Em vez disso, as redes neurais profundas podem aprender recursos com base em correlação ou estrutura estatística dentro dos dados de entrada. Os recursos aprendidos podem ser fornecidos para um modelo matemático que pode mapear recursos detectados para uma saída. O modelo matemático usado pela rede é, em geral, especializado para a tarefa específica a ser realizada, e modelos diferentes serão usados para executar tarefa diferente.

[00198] Uma vez que a rede neural é estruturada, um modelo de aprendizado pode ser aplicado à rede para treinar a rede para executar tarefas específicas. O modelo de aprendizado descreve como ajustar os pesos dentro do modelo para reduzir o erro de saída da rede. A propagação reversa de erros é um método comum usado para treinar redes neurais. Um vetor de entrada é apresentado para a rede para processamento. A saída da rede é comparada à saída desejada com o uso de uma função de perda e um valor de erro é calculado para cada um dos neurônios na camada de saída. Os valores de erro são, então, propagados para trás até que cada neurônio tenha um valor de erro associado que representa sensivelmente

sua contribuição para a saída original. A rede pode, então, aprender a partir desses erros com o uso de um algoritmo, como o algoritmo descendente de gradiente estocástico, para atualizar os pesos da rede neural.

[00199] As **Figuras 15A a 15B** ilustram uma rede neural convolucional exemplificativa. A **Figura 15A** ilustra diversas camadas dentro de uma CNN. Como mostrado na **Figura 15A**, uma CNN exemplificativa usada para modelar processamento de imagem pode receber entrada 1502 que descreve os componentes vermelho, verde e azul (RGB) de uma imagem de entrada. A entrada 1502 pode ser processada por múltiplas camadas convolucionais (por exemplo, primeira camada convolucional 1504, segunda camada convolucional 1506). A saída das múltiplas camadas convolucionais pode ser opcionalmente processada por um conjunto de camadas completamente conectadas 1508. Os neurônios em uma camada completamente conectada têm conexões completas a todas as ativações na camada anterior, como anteriormente descrito para uma rede de encaminhamento. A saída das camadas completamente conectadas 1508 pode ser usada para gerar um resultado de saída da rede. As ativações dentro das camadas completamente conectadas 1508 podem ser computadas com o uso de multiplicação de matriz em vez de convolução. Nem todas as implantações de CNN fazem uso de camadas completamente conectadas 1508. Por exemplo, em algumas implantações, a segunda camada convolucional 1506 pode gerar saída para a CNN.

[00200] As camadas convolucionais são conectadas de modo esparso, o que difere da configuração tradicional

de rede neural constatada nas camadas completamente conectadas 1508. As camadas tradicionais de rede neural são completamente conectadas, de modo que cada unidade da saída interaja com cada unidade de entrada. No entanto, as camadas convolucionais são conectadas de modo esparso devido ao fato de que a saída da convolução de um campo é inserida (em vez do respectivo valor de estado de cada um dos nós no campo) para os nós da camada subsequente, como ilustrado. Os kernels associados às camadas convolucionais realizam operações de convolução, em que a saída é enviada para a próxima camada. A redução de dimensionalidade realizada dentro das camadas convolucionais é um aspecto que possibilita que a CNN escalone para processar imagens grandes.

[00201] A **Figura 15B** ilustra estágios de computação exemplificativos dentro de uma camada convolucional de uma CNN. A entrada para uma camada convolucional 1512 de uma CNN pode ser processada em três estágios de uma camada convolucional 1514. Os três estágios podem incluir um estágio de convolução 1516, um estágio de detector 1518 e um estágio de agrupamento 1520. A camada de convolução 1514 pode, então, emitir dados para uma camada convolucional sucessiva. A camada convolucional final da rede pode gerar dados de mapa de recurso de saída ou fornecer entrada para uma camada completamente conectada, por exemplo, para gerar um valor de classificação para a entrada para a CNN.

[00202] No estágio de convolução 1516, executa diversas convoluções em paralelo para produzir um conjunto de ativações lineares. O estágio de convolução 1516 pode

incluir uma transformação afim, que é qualquer transformação que pode ser especificada como uma transformação linear mais uma tradução. As transformações afins incluem rotações, traduções, escalonamento e combinações dessas transformações. O estágio de convolução computa a saída de funções (por exemplo, neurônios) que são conectadas a regiões específicas na entrada, que podem ser determinadas como a região local associada ao neurônio. Os neurônios computam um produto escalar entre os pesos dos neurônios e a região na entrada de local a qual os neurônios são conectados. A saída do estágio de convolução 1516 define um conjunto de ativações lineares que são processadas por estágios sucessivos da camada convolucional 1514.

[00203] As ativações lineares podem ser processadas por um estágio de detector 1518. No estágio de detector 1518, cada ativação linear é processada por uma função de ativação não linear. A função de ativação não linear aumenta as propriedades não lineares da rede geral sem afetar os campos receptores da camada de convolução. Diversos tipos de funções de ativação não lineares podem ser usados. Um tipo particular é a unidade linear retificada (ReLU), que usa uma função de ativação definida como $f(x) = \max(0, x)$, de modo que a ativação seja delimitada para zero.

[00204] O estágio de agrupamento 1520 usa uma função de agrupamento que substitui a saída da segunda camada convolucional 1506 por uma estatística de resumo das saídas próximas. A função de agrupamento pode ser usada para introduzir invariância de tradução na rede neural,

de modo que traduções pequenas para a entrada não alterem as saídas agrupadas. A invariância para tradução local pode ser útil em cenários em que a presença de um recurso nos dados de entrada é mais importante que a localização precisa do recurso. Diversos tipos de agrupamento funções podem ser usadas durante o estágio de agrupamento 1520, incluindo agrupamento máximo, agrupamento médio e agrupamento normal 12. Adicionalmente, algumas implantações de CNN não incluem um estágio de agrupamento. Em vez disso, tais implantações substituem e estágio de convolução adicional que têm um avanço aumentado em relação aos estágios de convolução anteriores.

[00205] A saída da camada convolucional 1514 pode, então, ser processada pela próxima camada 1522. A próxima camada 1522 pode ser uma camada convolucional adicional ou uma das camadas completamente conectadas 1508. Por exemplo, a primeira camada convolucional 1504 da Figura 15A pode emitir para a segunda camada convolucional 1506, enquanto a segunda camada convolucional pode emitir para uma primeira camada das camadas completamente conectadas 1508.

[00206] A **Figura 16** ilustra uma rede neural recorrente exemplificativa. Em uma rede neural recorrente (RNN), o estado anterior da rede influencia a saída do estado atual da rede. As RNNs podem ser construídas em uma variedade de maneiras com o uso de uma variedade de funções. O uso de RNNs, em geral, gira em torno do uso de modelos matemáticos para prever o futuro com base em uma sequência de entradas anterior. Por exemplo, uma RNN

pode ser usada para executar modelagem de linguagem estatística para prever uma palavra iminente devido a uma sequência anterior de palavras. A RNN ilustrada 1600 pode ser descrita como tendo uma camada de entrada 1602 que recebe um vetor de entrada, camadas ocultadas 1604 para implantar uma função recorrente, um mecanismo de retroalimentação 1605 para possibilitar uma "memória" de estados anteriores, e uma camada de saída 1606 para emitir um resultado. A RNN 1600 opera com base em etapas de tempo. O estado da RNN em uma determinada etapa de tempo é influenciado com base na etapa de tempo anterior por meio do mecanismo de retroalimentação 1605. Para uma determinada etapa de tempo, o estado das camadas ocultadas 1604 é definido pelo estado anterior e a entrada na etapa de tempo atual. Uma entrada inicial (x_1) em uma primeira etapa de tempo pode ser processada pela camada ocultada 1604. Uma segunda entrada (x_2) pode ser processada pela camada ocultada 1604 com o uso de informações de estado que são determinadas durante o processamento da entrada inicial (x_1). Um determinado estado pode ser computado como $s_t = f(Ux_t+Ws_{t-1})$, em que U e W são matrizes de parâmetro. A função f é geralmente uma não linearidade, como a função de tangente hiperbólica (Tanh) ou uma variante da função de retificador $f(x)=\max(0,x)$. No entanto, a função matemática específica usada nas camadas ocultadas 1604 pode variar dependendo dos detalhes de implantação específicos da RNN 1600.

[00207] Além das redes básicas de CNN e RNN descritas, variações nessas redes podem ser possibilitadas. Uma variante de RNN exemplificativa é a memória de curto

prazo longa (LSTM) RNN. RNNs de LSTM têm capacidade de aprender dependência de longo prazo que podem ser necessárias para processar sequências mais longas de linguagem. Uma variante na CNN é uma rede de crença profunda convolucional, que tem uma estrutura similar a uma CNN e é treinada de uma maneira similar a uma rede de crença profunda. Uma rede de crença profunda (DBN) é uma rede neural generativa que é composta de múltiplas camadas de variáveis estocásticas (aleatórias). DBNs podem ser treinadas camada-por-camada com o uso de aprendizado não supervisionado ávido. Os pesos aprendidos da DBN podem, então, ser usados para fornecer redes neurais pré-treino determinando-se um conjunto inicial ideal de pesos para a rede neural.

[00208] **A Figura 17** ilustra treinamento e implantação de uma rede neural profunda. Uma vez que uma determinada rede foi estruturada para uma tarefa, a rede neural é treinada com o uso de um conjunto de dados de treinamento 1702. Diversas estruturas de treinamento foram desenvolvidas para possibilitar aceleração de hardware do processo de treinamento. Por exemplo, a estrutura de aprendizado por máquina 1404 da Figura 14 pode ser configurada como uma estrutura de treinamento 1704. A estrutura de treinamento 1704 pode se prender a uma rede neural não treinada 1706 e possibilitar a rede neural não treinada de ser treinada com o uso dos recursos de processamento paralelo descritos no presente documento para gerar uma rede neural treinada 1708. Para iniciar o processo de treinamento, os pesos iniciais podem ser escolhidos aleatoriamente ou por pré-

treinamento com o uso de uma rede de crença profunda. O ciclo de treinamento pode, então, ser realizado ou de uma maneira supervisionada ou não supervisionada.

[00209] O aprendizado não supervisionado é um método de aprendizado no qual treinamento é realizado como uma operação mediada, como quando o conjunto de dados de treinamento 1702 inclui entrada emparelhada com a saída desejada para a entrada, ou em que o conjunto de dados de treinamento inclui entrada que tem saída conhecida e a saída da rede neural é manualmente classificada. A rede processa as entradas e compara as saídas resultantes com um conjunto de saídas esperadas ou desejadas. Os erros são, então, propagados de volta através do sistema. A estrutura de treinamento 1704 pode ajustar os pesos que controlam a rede neural não treinada 1706. A estrutura de treinamento 1704 pode fornecer ferramentas para monitorar quão bem a rede neural não treinada 1706 está convergindo para um modelo adequado para gerar respostas corretas com base em dados de entrada conhecidos. O processo de treinamento ocorre repetidamente conforme os pesos da rede são ajustados para refinar a saída gerada pela rede neural. O processo de treinamento pode continuar até que a rede neural alcance uma precisão estatisticamente desejada associada a uma rede neural treinada 1708. A rede neural treinada 1708 pode, então, ser empregada para implantar qualquer número de operações de aprendizado por máquina para gerar um resultado de inferência 814 com base em entrada de novos dados 812.

[00210] O aprendizado não supervisionado é um método de aprendizado no qual a rede tenta treinar a si mesma com o uso de dados não rotulados. Desse modo, para aprendizado não supervisionado, o conjunto de dados de treinamento 1702 incluirá dados de entrada sem quaisquer dados de saída associados. A rede neural não treinada 1706 pode aprender agrupamentos dentro da entrada não rotulada e pode determinar como entradas individuais estão relacionadas ao conjunto de dados geral. O treinamento não supervisionado pode ser usado para gerar um mapa de auto-organização, que é um tipo de rede neural treinada 1708 com capacidade de realizar operações útil na redução da dimensionalidade de dados. O treinamento não supervisionado também pode ser usado para executar detecção de anomalia, o que permite a identificação de pontos de dados em um conjunto de dados de entrada que desviam dos padrões normais dos dados.

[00211] As variações em treinamento supervisionado e não supervisionado também podem ser empregadas. O aprendizado semissupervisionado é uma técnica na qual o conjunto de dados de treinamento 1702 inclui uma mistura de dados rotulados e não rotulados da mesma distribuição. O aprendizado incremental é uma variante de aprendizado supervisionado na qual dados de entrada são continuamente usados para treinar adicionalmente o modelo. O aprendizado incremental possibilita que a rede neural treinada 1708 se adapte aos novos dados 1712 sem esquecer do conhecimento incutido dentro da rede durante o treinamento inicial.

[00212] Se supervisionado ou não supervisionado, o processo de treinamento para redes neurais particularmente profundas pode ser muito computacionalmente intensivo para um único nó de computação. Em vez de usar um único nó de computação, uma rede distribuída de nós computacionais pode ser usada para acelerar o processo de treinamento.

[00213] A **Figura 18** é um diagrama de blocos que ilustra aprendizado distribuído. O aprendizado distribuído é um modelo de treinamento que usa múltiplos nós de computação distribuídos para executar treinamento supervisionado ou não supervisionado de uma rede neural. Os nós computacionais distribuídos podem, cada um, incluir um ou mais processadores de hospedeiro e um ou mais dos nós de processamento de propósito geral. Como ilustrado, em aprendizado distribuído pode ser realizado paralelismo de modelo 1802, paralelismo de dados 1804, ou uma combinação de paralelismo de dados e de modelo 1804.

[00214] Em paralelismo de modelo 1802, diferentes nós computacionais em um sistema distribuído podem executar computações de treinamento para diferentes partes de uma única rede. Por exemplo, cada camada de uma rede neural pode ser treinada por um nó de processamento diferente do sistema distribuído. Os benefícios de paralelismo de modelo incluem a capacidade de escalonar para modelos particularmente grandes. A divisão das computações associadas a diferentes camadas da rede neural possibilita o treinamento de redes neurais muito grandes nas quais os pesos de todas camadas não se ajustariam na

memória de um único nó computacional. Em alguns casos, paralelismo de modelo pode ser particularmente útil ao realizar treinamento não supervisionado de redes neurais grandes.

[00215] Em paralelismo de dados 1804, os diferentes nós da rede distribuída têm um exemplo completo do modelo e cada nó recebe uma porção diferente dos dados. Os resultados dos nós diferentes são, então, combinados. Embora abordagens diferentes para paralelismo de dados sejam possíveis, todas as abordagens de treinamento paralelo de dados necessitam de uma técnica de combinar resultados e sincronizar os parâmetros de modelo entre cada nó. As abordagens exemplificativas para combinar dados incluem ponderação de parâmetro e paralelismo de dados baseado em atualização. A ponderação de parâmetro treina cada nó em um subconjunto dos dados de treinamento e ajusta os parâmetros globais (por exemplo, pesos, inclinações) para a média dos parâmetros de cada nó. A ponderação de parâmetro usa um servidor de parâmetro central que mantém os dados de parâmetro. O paralelismo de dados baseados em atualização é similar à ponderação de parâmetro, exceto pelo fato de que, em vez de transferir parâmetros dos nós para o servidor de parâmetro, as atualizações para o modelo são transferidas. Adicionalmente, paralelismo de dados baseado em atualização pode ser realizado de uma maneira descentralizada, em que as atualizações são compactadas e transferidas entre nós.

[00216] O paralelismo de dados e de modelo combinados 1806 podem ser implantados, por exemplo, em um sistema

distribuído no qual cada nó computacional inclui múltiplas GPUs. Cada nó pode ter um exemplo completo do modelo com GPUs separadas dentro de cada nó que são usadas para treinar diferentes porções do modelo.

[00217] O treinamento distribuído tem sobrecarga aumentada em relação ao treinamento em uma única máquina. No entanto, os processadores paralelos e GPGPUs descritos no presente documento podem, cada um, implantar diversas técnicas para reduzir a sobrecarga de treinamento distribuído, incluindo técnicas para possibilitar transferência de dados de GPU para GPU de largura de banda alta e sincronização de dados remotos acelerada.

APLICATIVOS DE APRENDIZADO POR MÁQUINA EXEMPLIFICATIVOS

[00218] O aprendizado por máquina pode ser aplicado para solucionar uma variedade de problemas tecnológicos, incluindo, mas não limitados a, visão computacional, navegação e acionamento autônomo, reconhecimento de discurso e processamento de linguagem. A visão computacional foi tradicionalmente uma das áreas de pesquisa mais ativas para aplicativos de aprendizado por máquina. Os aplicativos de visão computacional estão na faixa de habilidades visuais de reprodução de ser humano, como reconhecer faces, a criar novas categorias de habilidades visuais. Por exemplo, aplicativos de visão computacional podem ser configurados para reconhecer ondas de som das vibrações induzidas em objetos visíveis em um vídeo. O aprendizado por máquina acelerado por processador paralelo possibilita que os aplicativos de visão computacional sejam treinados com o

uso de conjunto de dados de treinamento significativamente maiores que anteriormente viáveis e possibilita que sistemas de inferência sejam empregados com o uso de processadores paralelos de baixa potência.

[00219] O aprendizado por máquina acelerado por processador paralelo tem aplicativos de acionamento autônomo, incluindo reconhecimento de sinalização de pista e estrada, evasão de obstáculo, navegação e controle de acionamento. As técnicas de aprendizado por máquina acelerado podem ser usadas para treinar modelos de acionamento com base em conjuntos de dados que definem as respostas apropriadas para entrada de treinamento específico. Os processadores paralelos descritos no presente documento podem possibilitar treinamento rápido das redes neurais crescentemente complexas usadas para soluções de acionamento autônomas e possibilitam a implantação de processadores de inferência de baixa potência em uma plataforma móvel adequada para integração em veículos autônomos.

[00220] As redes neurais profundas aceleradas por processador paralelo possibilitaram abordagens de aprendizado por máquina para reconhecimento automático de discurso (ASR). ASR inclui a criação de uma função que computa a sequência linguística mais provável devido a uma sequência acústica de entrada. O aprendizado por máquina acelerado com o uso de redes neurais profundas possibilitou a substituição dos modelos ocultos de Markov (HMMs) e modelos de mistura gaussiana (GMMs) anteriormente usados para ASR.

[00221] O aprendizado por máquina acelerado por processador paralelo também pode ser usado para acelerar processamento de linguagem natural. Os procedimentos de aprendizado automático podem fazer uso de algoritmos de inferência estatística para produzir modelos que são robustos para entrada errada ou não familiar. Os aplicativos de processador de linguagem natural exemplificativos incluem tradução automática de máquina entre linguagens humanas.

[00222] As plataformas de processamento paralelo usadas para aprendizado por máquina podem ser divididas em plataformas de treinamento e plataformas de implantação. As plataformas de treinamento são, em geral, altamente paralelas e incluem otimizações para acelerar treinamento de nó único de múltiplas GPUs e treinamento de múltiplos nós, de múltiplas GPUs, enquanto plataformas de aprendizado por máquina (por exemplo, inferência) integradas, em geral, incluem processadores paralelos de potência inferior adequados para uso em produtos, como câmeras, robôs autônomos e veículos autônomos.

[00223] A **Figura 19** ilustra uma modalidade de um dispositivo de computação 1900 que emprega um acelerador 1910. O dispositivo de computação 1900 (por exemplo, dispositivos inteligentes utilizáveis junto ao corpo, dispositivos de realidade virtual (VR), visor montado na cabeça (HMDs), computadores móveis, dispositivos de Internet de Todas as Coisas (IoT), computadores do tipo laptop, computadores do tipo desktop, computadores de servidor, etc.) pode ser o mesmo que o sistema de

processamento de dados 100 da **Figura 1** e, consequentemente, por questões de brevidade, clareza e facilidade de entendimento, muitos dos detalhes indicados acima em referência às **Figuras 1 a 18** não são adicionalmente discutidos ou repetidos doravante. Como ilustrado, em uma modalidade, o dispositivo de computação 1900 é mostrado como hospedando um acelerador 1910. Embora mostrado como um componente independente, outras modalidades podem incluir acelerador 1910 que é incluído dentro de GPU 1914. Em ainda outras modalidades, o acelerador 1910 pode ser incluído dentro de CPU 1912.

[00224] Por todo o documento, termos como “domínio de gráficos” podem ser referenciados de modo intercambiável com “unidade de processamento de gráficos”, “processador de gráficos”, ou simplesmente “GPU” e, de modo similar, “domínio de CPU” ou “domínio de hospedeiro” podem ser referenciados de modo intercambiável com “unidade de processamento de computador”, “processador de aplicativo”, ou simplesmente “CPU”.

[00225] O dispositivo de computação 1900 pode incluir qualquer número e tipo de dispositivos de comunicação, como sistemas de computação grandes, como computadores de servidor, computadores do tipo desktop, etc., e pode incluir ainda decodificadores de sinal (por exemplo, decodificadores de sinal de televisão a cabo baseados em Internet, etc.), dispositivos baseados em sistema de posicionamento global (GPS), etc. O dispositivo de computação 1900 pode incluir dispositivos móveis de computação que servem como dispositivos de comunicação,

como telefones celulares, incluindo telefones inteligentes, assistentes pessoais digitais (PDAs), computadores do tipo tablet, computadores do tipo laptop, leitores eletrônicos, televisões inteligentes, plataformas de televisão, dispositivos utilizáveis junto ao corpo (por exemplo, óculos, relógios, braceletes, cartões inteligentes, joias, itens de vestuário, etc.), reprodutores de mídia, etc. Por exemplo, em uma modalidade, o dispositivo de computação 1900 pode incluir um dispositivo móvel de computação que emprega uma plataforma de computador que hospeda um circuito integrado (“IC”), como sistema em um chip (“SoC” ou “SOC”), integrando diversos componentes de hardware e/ou software de dispositivo de computação 1900 em um único chip.

[00226] Como ilustrado, em uma modalidade, o dispositivo de computação 1900 pode incluir qualquer número e tipo de componentes de hardware e/ou software, como (sem limitação) GPU 1914, acionador de gráficos (também chamado de “acionador de GPU”, “lógica de acionador de gráficos”, “lógica de acionador”, acionador em modo de usuário (UMD), UMD, estrutura de acionador em modo de usuário (UMDF), UMDF, ou simplesmente “acionador”) 1916, CPU 1912, memória 1908, dispositivos de rede, acionadores, ou similares, bem como fontes de entrada/saída (I/O) 1904, como telas sensíveis ao toque, painéis sensíveis ao toque, monitores sensíveis ao toque, teclados virtuais ou regulares, mouse virtual ou regular, portas, conectores, etc.

[00227] O dispositivo de computação 1900 pode incluir sistema operacional (OS) 1906 que serve como uma interface entre hardware e/ou recursos físicos do dispositivo de computador 1900 e um usuário. É contemplado que CPU 1912 pode incluir um ou mais processadores, como processador (ou processadores) 102 da **Figura 1**, enquanto GPU 1914 pode incluir um ou mais processadores de gráficos (ou múltiplos processadores).

[00228] Deve ser observado que termos como “nó”, “nó de computação”, “servidor”, “dispositivo de servidor”, “computador em nuvem”, “servidor em nuvem”, “computador de servidor em nuvem”, “máquina”, “máquina de hospedeiro”, “dispositivo”, “dispositivo de computação”, “computador”, “sistema de computação” e similares, podem ser usados de modo intercambiável por todo este documento. Também deve ser observado que termos como “aplicativo”, “aplicativo de software”, “programa”, “programa de software”, “pacote”, “pacote de software” e similares, podem ser usados de modo intercambiável por todo este documento. Além disso, termos como “trabalho”, “entrada”, “solicitação”, “mensagem” e similares, podem ser usados de modo intercambiável por todo este documento.

[00229] É contemplado e como ainda descrito em referência às **Figuras 1 a 13**, alguns processos do pipeline de gráficos, como descrito acima, são implantados em software, enquanto o restante é implantado em hardware. Um pipeline de gráficos pode ser implantado em um projeto de coprocessador de gráficos, em que CPU 1912 é projetada para trabalhar com GPU 1914,

que pode ser incluída em ou estar colocalizada com CPU 1912. Em uma modalidade, GPU 1914 pode empregar qualquer número e tipo de lógica de software e hardware convencional para realizar as funções convencionais relacionadas a renderização de gráficos, bem como lógica de software e hardware inovadora para executar qualquer número e tipo de instruções.

[00230] Como anteriormente mencionado, a memória 1908 pode incluir uma memória de acesso aleatório (RAM) que compreende banco de dados de aplicativo que tem informações de objeto. Um concentrador de controlador de memória, como controlador de memória 105 da **Figura 1**, pode acessar dados na RAM e encaminhar os mesmos para a GPU 1914 para processamento de pipeline de gráficos. RAM pode incluir RAM de taxa de dados dupla (DDR RAM), RAM de saída de dados estendida (EDO RAM), etc. A CPU 1912 interage com um pipeline de gráficos de hardware para compartilhar funcionalidade de pipeline de gráficos.

[00231] Os dados processados são armazenados em um armazenamento temporário no pipeline de gráficos de hardware, e informações de estado são armazenadas na memória 1908. A imagem resultante é, então, transferida para fontes de I/O 1904, como um componente de exibição para exibição da imagem. É contemplado que o dispositivo de exibição pode ser de diversos tipos, como Tubo de Raios Catódicos (CRT), Transistor de Película Fina (TFT), Visor de Cristal Líquido (LCD), matriz de Diodo Emissor de Luz Orgânica (OLED), etc., para exibir informações para um usuário.

[00232] A memória 1908 pode compreender uma região pré-alocada de um armazenamento temporário (por exemplo, armazenamento temporário de quadro); no entanto, deve ser entendido por um indivíduo versado na técnica que as modalidades não são tão limitadas, e que qualquer memória acessível para o pipeline de gráficos inferior pode ser usada. O dispositivo de computação 1900 pode incluir ainda concentrador de controle de entrada/saída (I/O) (ICH) 107, como referenciado na **Figura 1**, como uma ou mais fontes de I/O 1904, etc.

[00233] A CPU 1912 pode incluir um ou mais processadores para executar instruções a fim de realizar quaisquer rotinas de software que o sistema de computação implanta. As instruções frequentemente envolvem algum tipo de operação realizada mediante dados. Tanto os dados quanto as instruções podem ser armazenados em memória de sistema 1908 e qualquer cache associado. O cache é tipicamente projetado para ter tempos de latência mais curtos que a memória de sistema 1908; por exemplo, cache pode ser integrado ao mesmo chip (ou chips) de silício que o processador(s) e/ou construído com células de RAM estática mais rápida (SRAM) enquanto a memória de sistema 1908 pode ser construída com células de RAM estática mais lenta (DRAM). Como tende a armazenar instruções e dados mais frequentemente usados no cache em oposição à memória de sistema 1908, a eficácia de desempenho geral de dispositivo de computação 1900 aprimora. É contemplado que, em algumas modalidades, GPU 1914 pode existir como parte de CPU 1912 (como parte de um pacote de CPU

física), nesse caso, memória 1908 pode ser compartilhada pela CPU 1912 e pela GPU 1914 ou mantida separada.

[00234] A memória de sistema 1908 pode ser tornada disponível para outros componentes dentro do dispositivo de computação 1900. Por exemplo, quaisquer dados (por exemplo, dados de gráficos de entrada) recebidos de diversas interfaces para o dispositivo de computação 1900 (por exemplo, teclado e mouse, porta de impressora, porta de Rede de Área Local (LAN), porta de modem, etc.) ou recuperados de um elemento de armazenamento interno do dispositivo de computador 1900 (por exemplo, unidade de disco rígido) são, em geral, temporariamente listados na memória de sistema 1908 antes de serem operados pelo um ou mais processadores na implantação de um programa de software. De modo similar, dados que um programa de software determina devem ser enviados do dispositivo de computação 1900 para uma entidade externa através de uma das interfaces de sistema de computação, ou armazenados em um elemento de armazenamento interno são, em geral, temporariamente listados na memória de sistema 1908 antes de serem transmitidos ou armazenados.

[00235] Ademais, por exemplo, um ICH pode ser usado para garantir que tais dados sejam apropriadamente passados entre a memória de sistema 1908 e sua interface de sistema de computação correspondente apropriada (e dispositivo de armazenamento interno se o sistema de computação for assim projetado) e pode ter enlaces ponto a ponto bidirecionais entre si mesma e as fontes/dispositivos de I/O observados 1904. De modo similar, o concentrador de controle de plataforma (PCH)

pode ser usado para gerenciar as diversas solicitações conflitantes para acessos de memória de sistema 1908 entre CPU 1912 e GPU 1914, interfaces e elementos de armazenamento internos que podem surgir aproximadamente no tempo uns em relação aos outros.

[00236] As fontes de I/O 1904 podem incluir um ou mais dispositivos de I/O que são implantados para transferir dados para e/ou do dispositivo de computação 1900 (por exemplo, um adaptador de rede); ou, para um armazenamento não volátil em larga escala dentro de dispositivo de computação 1900 (por exemplo, unidade de disco rígido). O dispositivo de entrada de usuário, incluindo chaves alfanuméricas e outras chaves, pode ser usado para comunicar informações e seleções de comando para a GPU 1914. Outro tipo de dispositivo de entrada de usuário é controle de cursor, como um mouse, um trackball, uma tela sensível ao toque, um monitor sensível ao toque, ou chaves de direção de cursor para comunicar informações de direção e seleções de comando para GPU 1914 e para controlar movimento de cursor no dispositivo de exibição. As matrizes de câmera e microfone do dispositivo de computador 1900 podem ser empregadas para observar gestos, registrar áudio e vídeo e para receber e transmitir comandos visuais e de áudio.

[00237] O dispositivo de computação 1900 pode incluir ainda interface (ou interfaces) de rede para fornecer acesso a uma rede, como uma LAN, uma rede de área ampla (WAN), uma rede de área metropolitana (MAN), uma rede de área pessoal (PAN), Bluetooth, uma rede de nuvem, uma rede móvel (por exemplo, 3ª Geração (3G), 4ª Geração

(4G), etc.), uma intranet, a Internet, etc. A interface (ou interfaces) de rede podem incluir, por exemplo, uma interface de rede sem fio que tem antena, que pode representar uma ou mais antenas. A interface (ou interfaces) de rede também pode incluir, por exemplo, uma interface de rede com fio para se comunicar com dispositivos remotos por meio de cabo de rede, que pode ser, por exemplo, um cabo de Ethernet, um cabo coaxial, um cabo de fibra óptica, um cabo em série ou um cabo em paralelo.

[00238] A interface (ou interfaces) de rede pode fornecer acesso a uma LAN, por exemplo, conformando-se aos padrões IEEE 802.11b e/ou IEEE 802.11g, e/ou a interface de rede sem fio pode fornecer acesso a uma rede de área pessoal, por exemplo, conformando-se a padrões de Bluetooth. Outras interfaces e/ou protocolos de rede sem fio, incluindo versões anteriores e subsequentes dos padrões, também podem ser suportadas. Além disso, ou em vez disso, comunicação por meio dos padrões de LAN sem fio, interface (ou interfaces) de rede podem fornecer comunicação sem fio com o uso de, por exemplo, protocolos de Divisão de Tempo, Múltiplos Acessos (TDMA), protocolos de Sistemas Globais para Comunicações Móveis (GSM), protocolos de Divisão de Código, protocolo de Múltiplos Acessos (CDMA) e/ou qualquer outro tipo de protocolos de comunicações sem fio.

[00239] A interface (ou interfaces) de rede pode incluir uma ou mais interfaces de comunicação, como um modem, um cartão de interface de rede, ou outros

dispositivos de interface bem conhecidos, como aqueles usados para se acoplar à Ethernet, token ring, ou outros tipos de fixações com fio ou sem fio físicas para propósitos de fornecer um enlace de comunicação para suportar uma LAN ou uma WAN, por exemplo. Dessa maneira, o sistema de computador também pode ser acoplado a diversos dispositivos periféricos, clientes, superfícies de controle, consoles ou servidores por meio de uma infraestrutura de rede convencional, incluindo uma Intranet ou a Internet, por exemplo.

[00240] Deve ser observado que um sistema mais ou menos equipado que o exemplo descrito acima pode ser preferencial para determinadas implantações. Portanto, a configuração de dispositivo de computação 1900 pode variar de implantação para implantação dependendo de diversos fatores, como limitações de preço, exigências de desempenho, aprimoramentos tecnológicos ou outras circunstâncias. Exemplos do dispositivo eletrônico ou sistema de computador 1900 podem incluir (sem limitação) um dispositivo móvel, um assistente pessoal digital, um dispositivo móvel de computação, um telefone inteligente, um telefone celular, um aparelho telefônico, um pager unidirecional, um pager bidirecional, um dispositivo de troca de mensagens, um computador, um computador pessoal (PC), um computador do tipo desktop, um computador do tipo laptop, um computador do tipo notebook, um computador portátil, um computador do tipo tablet, um servidor, uma matriz de servidor ou parque de servidores, um servidor da web, um servidor de rede, um servidor de Internet, uma estação

de trabalho, um minicomputador, um computador de quadro principal, um supercomputador, um utensílio de rede, um utensílio da web, um sistema de computação distribuído, sistemas de múltiplos processadores, sistemas baseados em processador, aparelhos eletrônicos de consumidor, aparelhos eletrônicos de consumidor programáveis, televisão, televisão digital, decodificador de sinal, ponto de acesso sem fio, estação-base, estação de assinante, centro de assinante móvel, controlador de rede de rádio, roteador, concentrador, porta de comunicação, ponte, comutador, máquina, ou combinações dos mesmos.

[00241] As modalidades podem ser implantadas como qualquer um ou uma combinação de: um ou mais microchips ou circuitos integrados interconectados com o uso de uma placa-mãe, lógica de hardware, software armazenado por um dispositivo de memória e executado por um microprocessador, firmware, um circuito integrado específico de aplicativo (ASIC), e/ou uma matriz de porta de campo programável (FPGA). O termo "lógica" pode incluir, a título de exemplo, software ou hardware e/ou combinações de software e hardware.

[00242] As modalidades podem ser fornecidas, por exemplo, como um produto de programa de computador, o que pode incluir uma ou mais mídias legíveis por máquina que têm armazenadas nas mesmas instruções executáveis por máquina que, quando executadas por uma ou mais máquinas, como um computador, rede de computadores, ou outros dispositivos eletrônicos, podem resultar na uma ou mais máquinas que realizam operações de acordo com as

modalidades descritas no presente documento. Uma mídia legível por máquina pode incluir, mas sem limitação, disquetes, discos ópticos, CD-ROMs (Memórias de Apenas Leitura de Disco Compacto), e discos ópticos-magnéticos, ROMs, RAMs, EPROMs (Memórias de Apenas Leitura Programáveis Apagáveis), EEPROMs (Memórias de Apenas Leitura Programáveis Eletricamente Apagáveis), cartões magnéticos ou ópticos, memória flash, ou outro tipo de mídia legível por mídia/máquina adequada para armazenar instruções executáveis por máquina.

[00243] Ademais, as modalidades podem ser transferidas por download como um produto de programa de computador, em que o programa pode ser transferido de um computado remoto (por exemplo, um servidor) para um computador solicitante (por exemplo, um cliente) por meio de um ou mais sinais de dados embutidos e/ou modulados por uma onda carreadora ou outro meio de propagação por meio de um enlace de comunicação (por exemplo, um modem e/ou conexão de rede).

[00244] De acordo com uma modalidade, acelerador 1910 inclui um hardware de multiplicação de matriz 1913 (por exemplo, matriz sistólica) para realizar operações de multiplicação de matriz de aprendizado por máquina. Em tal modalidade, as operações podem ser realizadas em matrizes grandes (por exemplo, 16x16) ao separar (ou dividir) as matrizes em matrizes menores (por exemplo, 8x8). Desse modo, o processamento das matrizes envolve um número significativo de multiplicações de matriz, incluindo uma grande quantidade de operações de multiplicar-adicionar.

[00245] Muitas vezes, realizar multiplicações de matriz exige trocas entre desempenho e precisão. Por exemplo, hardware pode precisar suportar aplicativos tanto de ponto de flutuação 16 (float16) quanto de ponto de flutuação 8 (float8). Para a mesma largura de banda de memória, um modo float8 pode executar duas vezes o número de operações de multiplicar-adicionar, com precisão reduzida. No entanto, implantar conjunto de circuitos separado para cada modo adiciona custo de potência e área significativo.

[00246] A **Figura 20A** ilustra um multiplicador de bit 2N convencional repartido em um adicionador e multiplicador de bit quatro N. Como mostrado na **Figura 20A**, uma multiplicação dos termos resultados em uma adição de A[N-1:0]*B[N-1:0], A[2N-1:N]*B[N-1:0], A[N-1:0]*B[2N-1:N] e A[2N-1:N]*B[2N-1:N] para igual A[2N-1:0]*B[2N-1:0]. A mesma arquitetura de multiplicador pode ser usada para computar multiplicações de 2k*N bit (ou modo duplex) usando-se o multiplicador 2N por 2N mostrado na **Figura 20A** para realizar duas multiplicações N por N. A **Figura 20B** ilustra tal multiplicação com o uso do multiplicador. Como mostrado na **Figura 20B**, o multiplicador 2Nx2N é modificado em modo duplex para zerar termos intermediários (A2[N-1:0]*B1[N-1:0]*~duplex e A1[N-1:0]*B2[N-1:0]*~duplex) por meio de um sinal indicador para operação duplex (por exemplo, ~ duplex). Os termos diferentes de zero são os produtos de duas multiplicações NxN para A2[N-1:0]*B2[N-1:0] e A1[N-1:0]*B1[N-1:0]. A2[N-1:0]*B2[N-1:0] e A1[N- 1:0]*B1[N-1:0] são adicionados em um estágio subsequente com

hardware adicional. Desse modo, um adicionador adicional é necessário para realizar multiplicações de bit 2k*N, que adiciona lógica significativa, área e potência necessários para alcançar a multiplicação 2Nx2N.

[00247] De acordo com uma modalidade, matriz 1913 inclui um multiplicador de produto escalar otimizado 1915 (**Figura 19**) que suporta múltiplas larguras de bit sem ter que incluir um adicionador adicional. Um produto escalar envolve multiplicação e adição. Por exemplo, determinando A e B como entradas para um multiplicador sistólico, em que cada um é M bits de largura, há um número (k) de 2N elementos de bit de largura, de modo que um primeiro caso (por exemplo, Caso 1) seja representado por:

$$M = k * 2N; \text{ e produto Escalar } = \sum_{i=0}^{k-1} A_i * B_i \qquad (1)$$

em que “i” é um índice através dos k elementos que produzem os vetores de entrada A e B.

[00248] Em um segundo caso (por exemplo, Caso 2) no qual um aplicativo de hardware suporta um produto escalar de elementos 2k N bit de largura:

$$M = 2k * N; \text{ e produto Escalar } = \sum_{i=0}^{2k-1} A_i * B_i \qquad (2)$$

[00249] Como mostrado acima, Caso 2 tem duas vezes o número de multiplicadores em comparação com o Caso 1, em que cada multiplicador individual é significativamente menor. Adicionalmente, Caso 2 tem duas vezes o número de elementos para adicionar em relação ao Caso 1 embora cada elemento no Caso 2 tenha uma menor quantidade de bits.

[00250] A **Figura 21** ilustra uma modalidade de um multiplicador de produto escalar 1915. Como mostrado na **Figura 21**, multiplicador 1915 é um multiplicador 2Nx2N que inclui um estágio de multiplicador 2110 e um estágio de adição 2120. Em uma modalidade, o multiplicador 1915 pode operar em um modo de produto escalar para realizar multiplicações 2k N bit, bem como um modo convencional (por exemplo, para realizar multiplicação 2Nx2N). No modo de produto escalar, o multiplicador 1915 é configurado como um produto escalar de diversos (por exemplo, 2 N) vetores de bit (A2*B2 + A1*B1) para realizar operações de multiplicação NxN e uma operação de adição das multiplicações NxN em estágio de multiplicador 2110, em vez de ter que implantar uma operação de adição em um estágio subsequente. Em tal modalidade, o primeiro e o último termos da operação de multiplicação são reduzidos para zero (ou zerados) por meio de um sinal de indicador para uma multiplicação de produto escalar (por exemplo, ~dotp), de modo que (A1[N-1:0]*B1[N-1:0])*~dotp e (A2[N-1:0]*B2[N-1:0])*~dotp = 0.

[00251] Em uma modalidade adicional, os bits mais significativos e os bits menos significativos de um dos multiplicandos (por exemplo, primeiros N bits e os últimos N bits da entrada B (B1&B2)) são trocados antes da multiplicação por meio de multiplexação. Tal troca possibilita uma multiplicação vertical (por exemplo, A1B1 e A2B2) de elementos, em vez de uma multiplicação diagonal de elementos (por exemplo, A1B2 e A2B1). De modo subsequente, estágio de adição 2120 realiza uma adição de A2*(dotp?B2:B1) e A1*(dotp?B1:B2), resultando

em uma saída de A2*B2 + A1*B1. Em conformidade, não há necessidade de um adicionador separado para realizar essa operação.

[00252] Em outras modalidades, o multiplicador 1915 pode generalizar a otimização de produto escalar na qual entrada recebida é dividida em K elementos. Desse modo, entradas fornecidas A[N*K-1:0] e B[N*K-1:0], em que K = o número de entradas por peça adicionadas e N = largura de bit de cada um dos multiplicandos, as saídas de multiplicador podem ser generalizadas para as seguintes equações para os dois modos de operação:

[00253] Saída de Modo Convencional:

$$\sum_{j=0}^{K-1}\sum_{i=0}^{K-1}(a[N(j+1)-1:Nj]*b[(N(i+1)-1:Ni])\ll(i+j)*N) \quad (3)$$

[00254] Saída de Modo de Produto Escalar:

$$\sum_{i=0}^{K-1}(a[N(K-i)-1:N(K-i-1)]*(dotp?\,b[N(K-i)-1:N(K-i-1)] : b[N(i+1)-1:Ni])\ll(K-1)*N \quad (4)$$

[00255] Com base no descrito acima, todos os outros termos no multiplicador 1915 são zerados. Com base em um exemplo de K =3, as equações acima são expandidas, como mostrado na **Figura 22**, em que A_0 = a[N-1:0], A_1 = a[2N-1:N], A_2 = a[3N-1:2N] e B_0 = b[N-1:0], B_1 = b[2N-1:N], B_2 = b[3N-1:2N]. Como mostrado na **Figura 22**, todos os produtos são definidos como 0 (ou anulados), exceto por A2*(dotp?B_2:B_0); A_1*B_1; e A_0*(dotp?B_0:B_2);

[00256] A **Figura 23** é um fluxograma que ilustra uma modalidade de um processo para realizar uma operação de

multiplicação. No bloco de processamento 2310, entrada é recebida no multiplicador 1915. No bloco de processamento 2320, a entrada é separada em K elementos. No bloco de processamento 2330, um modo de operação (por exemplo, Convencional ou Produto Escalar) é determinado. No bloco de processamento 2340, uma operação de multiplicação é realizada (por exemplo, multiplicação k 2N bit ou multiplicação 2k N bit) como indicado pelo modo de operação. Em uma modalidade, a indicação é dotada da entrada recebida. No bloco de processamento 2350, o resultado da operação de multiplicação é retornado.

[00257] A **Figura 24** é um fluxograma que ilustra uma modalidade de um processo para realizar uma operação de multiplicação de produto escalar 2k N bit. No bloco de processamento 2410, o multiplicador é configurado como um produto escalar. No bloco de processamento 2420, bits de um dos multiplicandos são trocados. No bloco de processamento 2430, uma operação de multiplicação é realizada. No bloco de processamento 2440, o primeiro e o último termos da operação de multiplicação são zerados. No bloco de processamento 2450, uma adição dos termos intermediários é realizada.

[00258] As seguintes cláusulas e/ou exemplos pertencem às modalidades ou exemplos adicionais: As especificações nos exemplos podem ser usadas em qualquer lugar em uma ou mais modalidades. Os diversos recursos das modalidades diferentes podem ser combinados de modo variável com alguns recursos incluídos e outros excluídos para adequar uma variedade de aplicações

diferentes. Exemplos podem incluir matéria como um método, meios para executar atuações do método, pelo menos uma mídia legível por máquina, incluindo instruções que, quando realizadas por uma máquina fazem com que a máquina execute atuações do método, ou de um aparelho ou sistema para facilitar comunicação híbrida de acordo com as modalidades e exemplos descritos no presente documento.

[00259] Algumas modalidades pertencem ao Exemplo 1 que inclui um aparelho para facilitar operações de multiplicação de matriz, que compreende hardware de multiplicação para operar em um modo de produto escalar, em que um estágio de multiplicação incluído no hardware de multiplicação é configurado como um produto escalar de diversos vetores de bit (N) para realizar NxN operações de multiplicação em uma pluralidade de multiplicandos e realizar operações de adição em resultados das NxN operações de multiplicação.

[00260] O Exemplo 2 inclui a matéria de Exemplo 1, em que o estágio de multiplicação realiza ainda uma varredura de bits mais significativos e bits menos significativos de um primeiro dentre a pluralidade de multiplicandos antes de realizar as operações de multiplicação.

[00261] O Exemplo 3 inclui a matéria de Exemplos 1 e 2, em que o estágio de multiplicação reduz primeiro e último termos da operação de multiplicação para zero.

[00262] O Exemplo 4 inclui a matéria de Exemplos 1 a 3, em que o hardware de multiplicação compreende ainda um

estágio de adição para realizar uma operação de adição em termos intermediários da operação de multiplicação.

[00263] O Exemplo 5 inclui uma matéria de Exemplos 1 a 4, em que o hardware de multiplicação é ainda configurado para operar em um modo convencional para realizar 2N operações de adição de multiplicação de matriz.

[00264] O Exemplo 6 inclui a matéria de Exemplos 1 a 5, em que o hardware de multiplicação recebe entrada e separa a entrada em uma pluralidade de elementos.

[00265] O Exemplo 7 inclui a matéria de Exemplos 1 a 6, em que o hardware de multiplicação determina se opera no modo convencional ou no modo de produto escalar.

[00266] Algumas modalidades pertencem ao Exemplo 8 que inclui um método para facilitar operações de multiplicação de matriz que compreende operar hardware de multiplicação em um modo de produto escalar, incluindo configurar um estágio de multiplicação como um produto escalar de diversos vetores de bit (N) para realizar NxN operações de multiplicação em uma pluralidade de multiplicandos e realizar operações de adição em resultados das NxN operações de multiplicação.

[00267] O Exemplo 9 inclui a matéria de Exemplo 8, que compreende ainda realizar uma varredura de bits mais significativos e bits menos significativos de um primeiro dentre a pluralidade de multiplicandos antes de realizar as operações de multiplicação.

[00268] O Exemplo 10 inclui a matéria de Exemplos 8 e 9, que compreende ainda reduzir primeiro e último termos da operação de multiplicação para zero.

[00269] O Exemplo 11 inclui a matéria de Exemplos 8 a 10, que compreende ainda realizar uma operação de adição em termos intermediários da operação de multiplicação em um estágio de adição do hardware de multiplicação.

[00270] O Exemplo 12 inclui a matéria de Exemplos 8 a 11, que compreende ainda configurar o hardware de multiplicação para operar em um modo convencional para realizar 2N operações de adição de multiplicação de matriz.

[00271] O Exemplo 13 inclui a matéria de Exemplos 8 a 12, que compreende ainda receber entrada no hardware de multiplicação e separar a entrada em uma pluralidade de elementos.

[00272] O Exemplo 14 inclui a matéria de Exemplos 8 a 13, que compreende ainda o hardware de multiplicação determina se opera no modo convencional ou no modo de produto escalar.

[00273] Algumas modalidades pertencem ao Exemplo 15 que inclui um acelerador de hardware que compreende uma matriz sistólica, incluindo hardware de multiplicação para operar em um modo de produto escalar, em que um estágio de multiplicação incluído no hardware de multiplicação é configurado como um produto escalar de diversos vetores de bit (N) para realizar NxN operações de multiplicação em uma pluralidade de multiplicandos e realizar operações de adição em resultados das NxN operações de multiplicação.

[00274] O Exemplo 16 inclui a matéria de Exemplo 15, em que o estágio de multiplicação realiza ainda uma varredura de bits mais significativos e bits menos

significativos de um primeiro dentre a pluralidade de multiplicandos antes de realizar as operações de multiplicação.

[00275] O Exemplo 17 inclui a matéria de Exemplos 15 e 16, em que o estágio de multiplicação reduz primeiro e último termos da operação de multiplicação para zero.

[00276] O Exemplo 18 inclui a matéria de Exemplos 15 a 17, em que o hardware de multiplicação compreende ainda um estágio de adição para realizar uma operação de adição em termos intermediários da operação de multiplicação.

[00277] O Exemplo 19 inclui uma matéria de Exemplos 15 a 18, em que o hardware de multiplicação é ainda configurado para operar em um modo convencional para realizar 2N operações de adição de multiplicação de matriz.

[00278] O Exemplo 20 inclui a matéria de Exemplos 15 a 19, em que o hardware de multiplicação determina se opera no modo convencional ou no modo de produto escalar.

[00279] A descrição anterior e os desenhos devem estar relacionados em um sentido ilustrativo, em vez de um sentido restritivo. O indivíduo versado na técnica entenderá que diversas modificações e alterações podem ser realizadas nas modalidades descritas no presente documento sem se afastar do espírito e escopo amplos da invenção como apresentado nas reivindicações anexas.

REIVINDICAÇÕES

1. Aparelho para facilitar operações de multiplicação de matriz, **caracterizado** por compreender:

hardware de multiplicação para operar em um modo de produto escalar, em que um estágio de multiplicação incluído no hardware de multiplicação é configurado como um produto escalar de diversos vetores de bit (N) para realizar NxN operações de multiplicação em uma pluralidade de multiplicandos e realizar operações de adição em resultados das NxN operações de multiplicação.

2. Aparelho, de acordo com a reivindicação 1, **caracterizado** pelo fato de que o estágio de multiplicação realiza ainda uma varredura de bits mais significativos e bits menos significativos de um primeiro dentre a pluralidade de multiplicandos antes de realizar as operações de multiplicação.

3. Aparelho, de acordo com a reivindicação 2, **caracterizado** pelo fato de que o estágio de multiplicação reduz o primeiro e o último termos da operação de multiplicação para zero.

4. Aparelho, de acordo com a reivindicação 3, **caracterizado** pelo fato de que o hardware de multiplicação compreende ainda um estágio de adição para realizar uma operação de adição em termos intermediários da operação de multiplicação.

5. Aparelho, de acordo com a reivindicação 1, **caracterizado** pelo fato de que o hardware de multiplicação é ainda configurado para

operar em um modo convencional para realizar 2N operações de adição de multiplicação de matriz.

6. Aparelho, de acordo com a reivindicação 5, **caracterizado** pelo fato de que o hardware de multiplicação recebe entrada e separa a entrada em uma pluralidade de elementos.

7. Aparelho, de acordo com a reivindicação 6, **caracterizado** pelo fato de que o hardware de multiplicação determina se opera no modo convencional ou no modo de produto escalar.

8. Método para facilitar operações de multiplicação de matriz, **caracterizado** por compreender operar hardware de multiplicação em um modo de produto escalar, incluindo:

configurar um estágio de multiplicação como um produto escalar de diversos vetores de bit (N) para realizar NxN operações de multiplicação em uma pluralidade de multiplicandos; e

realizar operações de adição em resultados das NxN operações de multiplicação.

9. Método, de acordo com a reivindicação 8, **caracterizado** pelo fato de que compreende ainda realizar uma varredura de bits mais significativos e bits menos significativos de um primeiro dentre a pluralidade de multiplicandos antes de realizar as operações de multiplicação.

10. Método, de acordo com a reivindicação 9, **caracterizado** pelo fato de que compreende ainda reduzir o primeiro e o último termos da operação de multiplicação para zero.

11. Método, de acordo com a reivindicação 10, **caracterizado** pelo fato de que compreende ainda realizar uma operação de adição em termos intermediários da operação de multiplicação em um estágio de adição do hardware de multiplicação.

12. Método, de acordo com a reivindicação 8, **caracterizado** pelo fato de que compreende ainda configurar o hardware de multiplicação para operar em um modo convencional para realizar 2N operações de adição de multiplicação de matriz.

13. Método, de acordo com a reivindicação 12, **caracterizado** pelo fato de que compreende ainda:

receber entrada no hardware de multiplicação; e

separar a entrada em uma pluralidade de elementos.

14. Método, de acordo com a reivindicação 13, **caracterizado** pelo fato de que compreende ainda o hardware de multiplicação que determina se opera no modo convencional ou no modo de produto escalar.

15. Acelerador de hardware, **caracterizado** por compreender:

uma matriz sistólica, incluindo hardware de multiplicação para operar em um modo de produto escalar, em que um estágio de multiplicação incluído no hardware de multiplicação é configurado como um produto escalar de diversos vetores de bit (N) para realizar NxN operações de multiplicação em uma pluralidade de multiplicandos e realizar operações de adição em resultados das NxN operações de multiplicação.

16. Acelerador, de acordo com a reivindicação 15, **caracterizado** pelo fato de que o estágio de multiplicação realiza ainda uma varredura de bits mais significativos e bits menos significativos de um primeiro dentre a pluralidade de multiplicandos antes de realizar as operações de multiplicação.

17. Acelerador, de acordo com a reivindicação 16, **caracterizado** pelo fato de que o estágio de multiplicação reduz o primeiro e o último termos da operação de multiplicação para zero.

18. Acelerador, de acordo com a reivindicação 17, **caracterizado** pelo fato de que o hardware de multiplicação compreende ainda um estágio de adição para realizar uma operação de adição em termos intermediários da operação de multiplicação.

19. Acelerador, de acordo com a reivindicação 15, **caracterizado** pelo fato de que o hardware de multiplicação é ainda configurado para operar em um modo convencional para realizar 2N operações de adição de multiplicação de matriz.

20. Acelerador, de acordo com a reivindicação 19, **caracterizado** pelo fato de que o hardware de multiplicação determina se opera no modo convencional ou no modo de produto escalar.

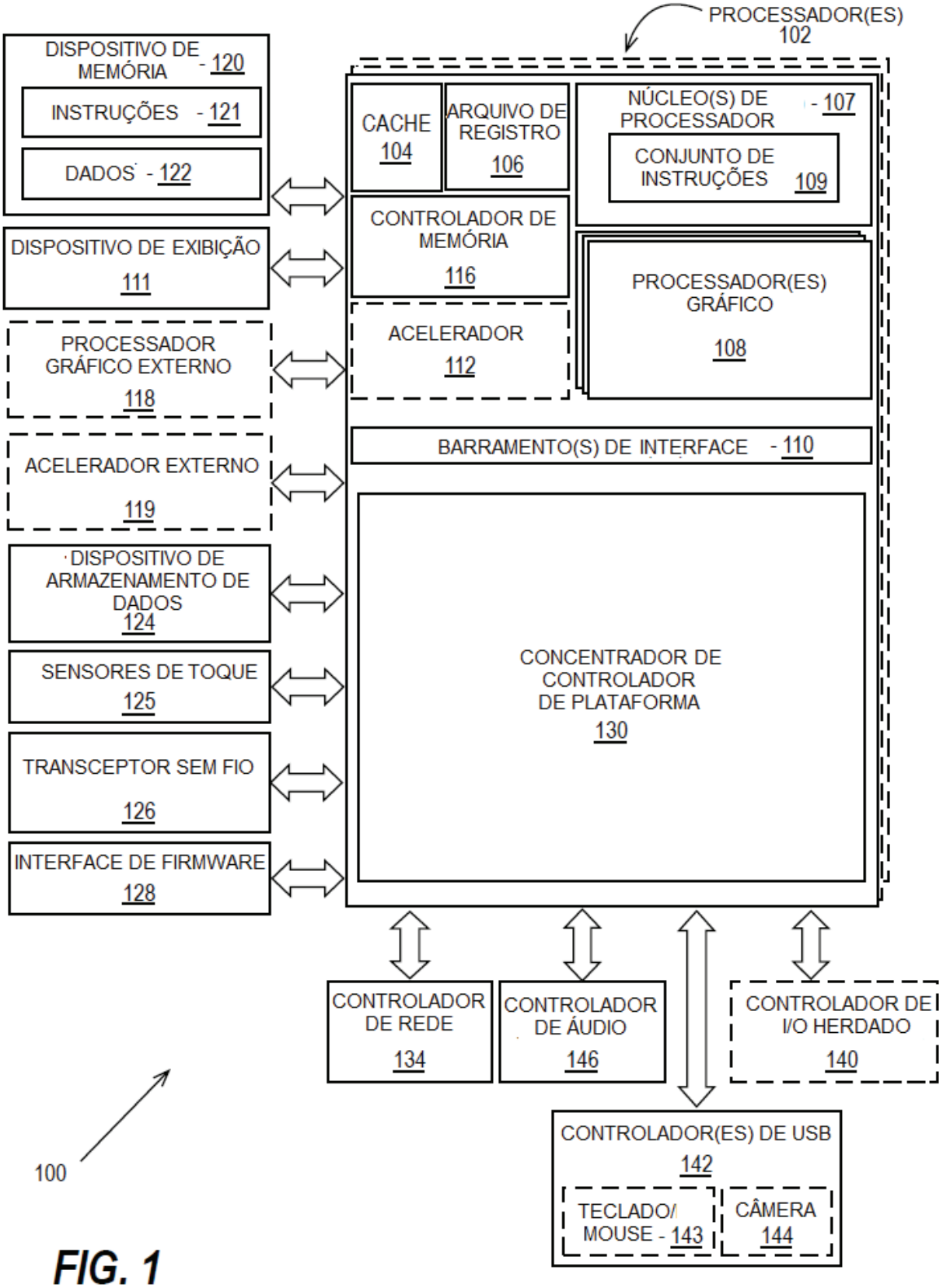

PROCESSADOR(ES)
102
DISPOSITIVO DE MEMÓRIA - 120
INSTRUÇÕES - 121
DADOS - 122
CACHE
104
ARQUIVO DE REGISTRO
106
NÚCLEO(S) DE PROCESSADOR - 107
CONJUNTO DE INSTRUÇÕES
109
CONTROLADOR DE MEMÓRIA
116
PROCESSADOR(ES) GRÁFICO
108
ACELERADOR
112
DISPOSITIVO DE EXIBIÇÃO
111
PROCESSADOR GRÁFICO EXTERNO
118
ACELERADOR EXTERNO
119
BARRAMENTO(S) DE INTERFACE - 110
DISPOSITIVO DE ARMAZENAMENTO DE DADOS
124
SENSORES DE TOQUE
125
TRANSCEPTOR SEM FIO
126
INTERFACE DE FIRMWARE
128
CONCENTRADOR DE CONTROLADOR DE PLATAFORMA
130
CONTROLADOR DE REDE
134
CONTROLADOR DE ÁUDIO
146
CONTROLADOR DE I/O HERDADO
140
CONTROLADOR(ES) DE USB
142
TECLADO/ MOUSE - 143
CÂMERA
144
100

FIG. 1

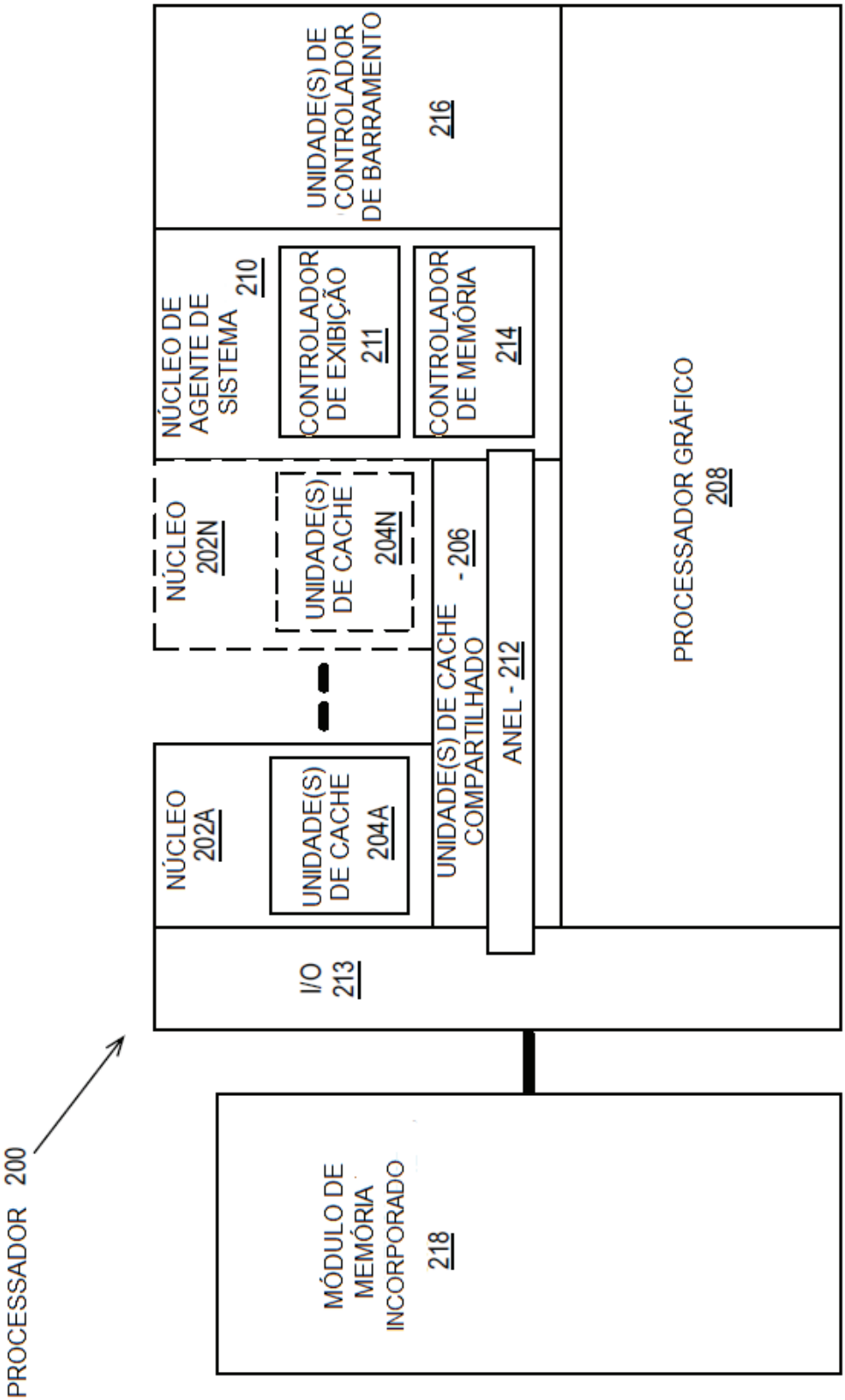

PROCESSADOR 200
MÓDULO DE MEMÓRIA INCORPORADO 218
I/O 213
NÚCLEO 202A
UNIDADE(S) DE CACHE 204A
NÚCLEO 202N
UNIDADE(S) DE CACHE 204N
UNIDADE(S) DE CACHE COMPARTILHADO - 206
ANEL - 212
NÚCLEO DE AGENTE DE SISTEMA 210
CONTROLADOR DE EXIBIÇÃO 211
CONTROLADOR DE MEMÓRIA 214
UNIDADE(S) DE CONTROLADOR DE BARRAMENTO 216
PROCESSADOR GRÁFICO 208

FIG. 2A

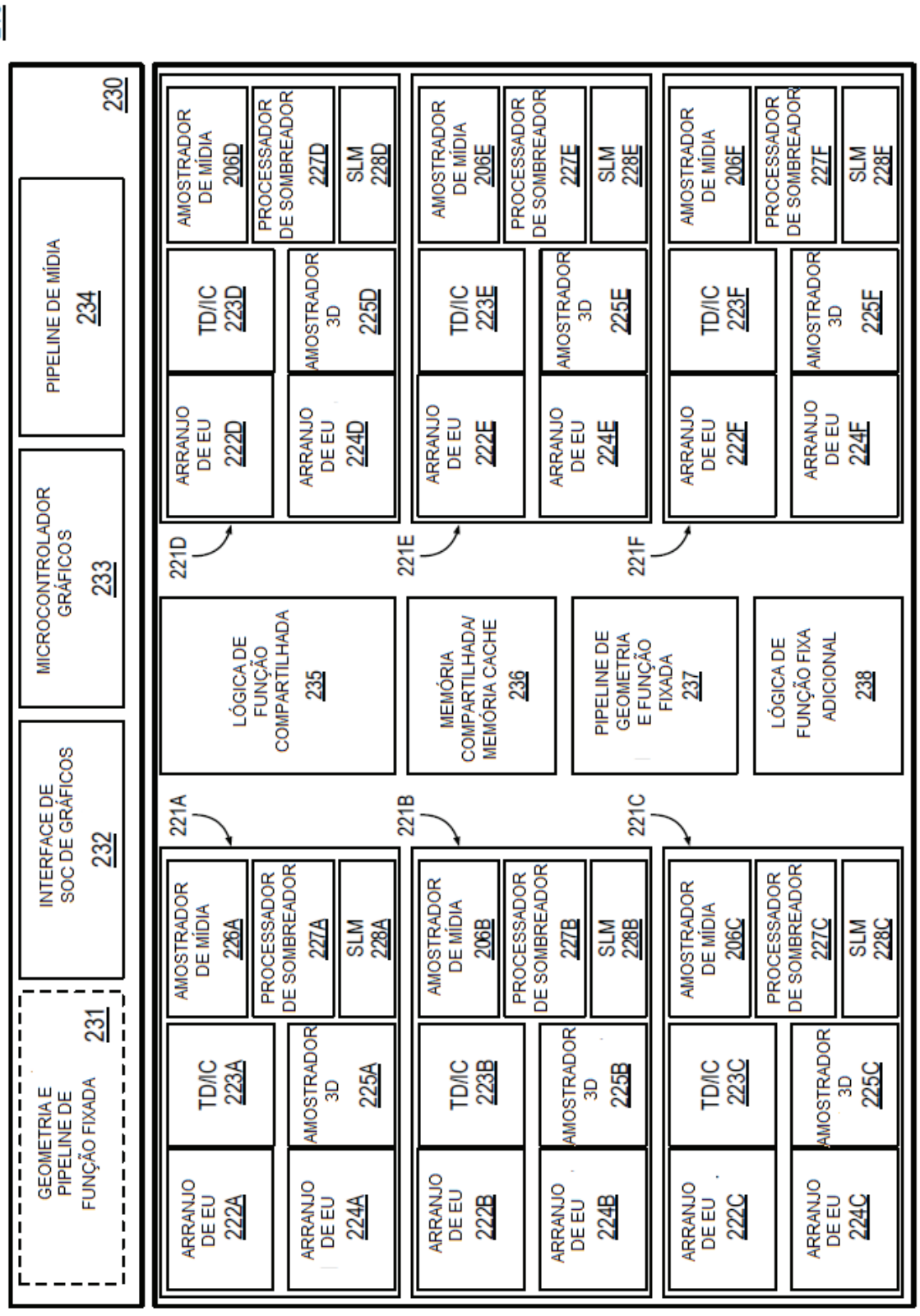

219
GEOMETRIA E PIPELINE DE FUNÇÃO FIXADA 231
INTERFACE DE SOC DE GRÁFICOS 232
MICROCONTROLADOR GRÁFICOS 233
PIPELINE DE MÍDIA 234
230
221A
221B
221C
221D
221E
221F
ARRANJO DE EU 222A
TD/IC 223A
AMOSTRADOR DE MÍDIA 226A
PROCESSADOR DE SOMBREADOR 227A
SLM 228A
ARRANJO DE EU 224A
AMOSTRADOR 3D 225A
ARRANJO DE EU 222B
TD/IC 223B
AMOSTRADOR DE MÍDIA 206B
PROCESSADOR DE SOMBREADOR 227B
SLM 228B
ARRANJO DE EU 224B
AMOSTRADOR 3D 225B
ARRANJO DE EU 222C
TD/IC 223C
AMOSTRADOR DE MÍDIA 206C
PROCESSADOR DE SOMBREADOR 227C
SLM 228C
ARRANJO DE EU 224C
AMOSTRADOR 3D 225C
ARRANJO DE EU 222D
TD/IC 223D
AMOSTRADOR DE MÍDIA 206D
PROCESSADOR DE SOMBREADOR 227D
SLM 228D
ARRANJO DE EU 224D
AMOSTRADOR 3D 225D
ARRANJO DE EU 222E
TD/IC 223E
AMOSTRADOR DE MÍDIA 206E
PROCESSADOR DE SOMBREADOR 227E
SLM 228E
ARRANJO DE EU 224E
AMOSTRADOR 3D 225E
ARRANJO DE EU 222E
TD/IC 223F
AMOSTRADOR DE MÍDIA 206E
PROCESSADOR DE SOMBREADOR 227E
SLM 228E
ARRANJO DE EU 224E
AMOSTRADOR 3D 225F
LÓGICA DE FUNÇÃO COMPARTILHADA 235
MEMÓRIA COMPARTILHADA/ MEMÓRIA CACHE 236
PIPELINE DE GEOMETRIA E FUNÇÃO FIXADA 237
LÓGICA DE FUNÇÃO FIXA ADICIONAL 238

FIG. 2B

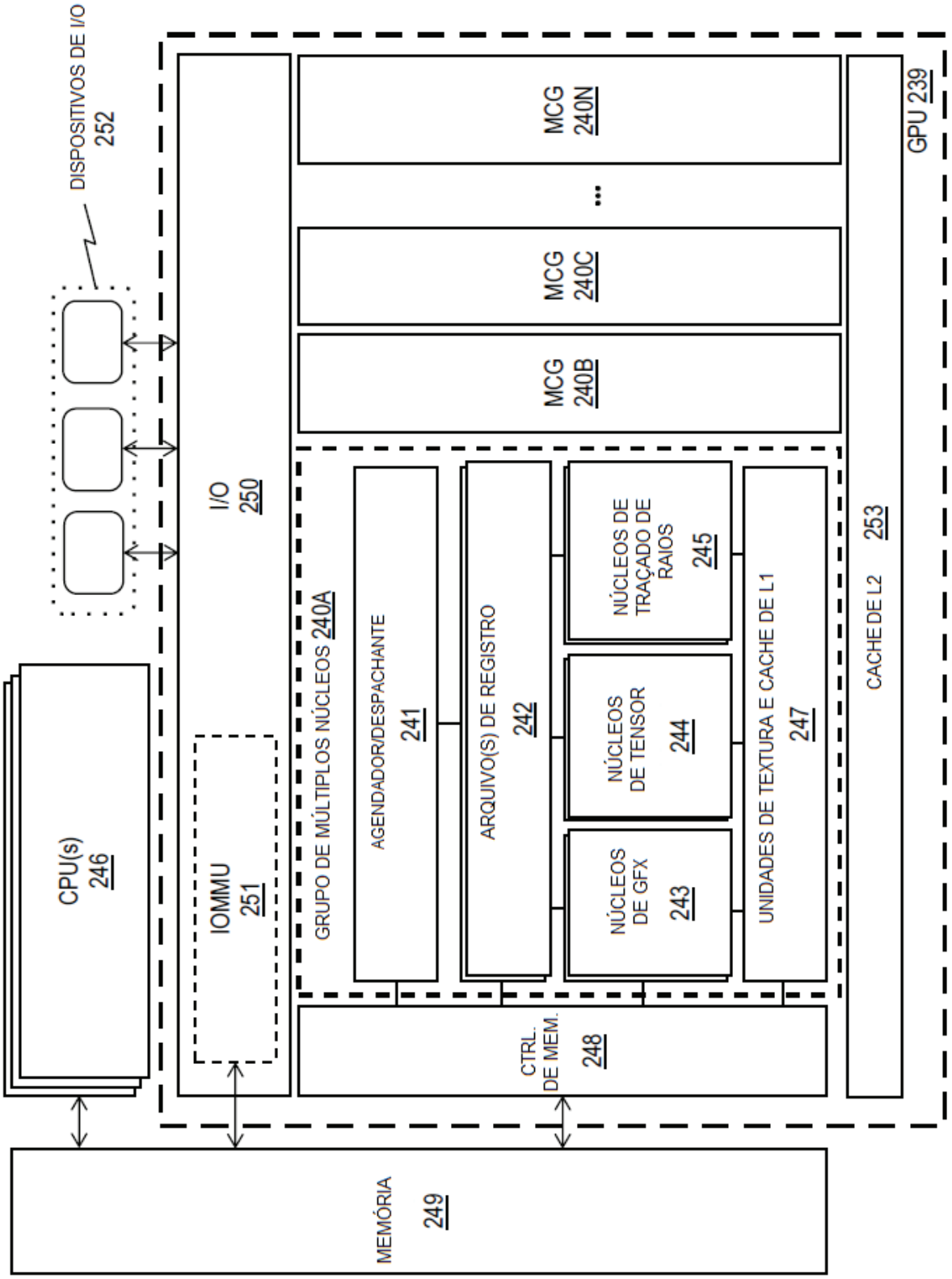

DISPOSITIVOS DE I/O
252
GPU 239
I/O
250
MCG
240N
MCG
240C
MCG
240B
GRUPO DE MÚLTIPLOS NÚCLEOS 240A
AGENDADOR/DESPACHANTE
241
ARQUIVO(S) DE REGISTRO
242
NÚCLEOS DE TRAÇADO DE RAIOS
245
NÚCLEOS DE TENSOR
244
NÚCLEOS DE GFX
243
UNIDADES DE TEXTURA E CACHE DE L1
247
CACHE DE L2
253
CPU(s)
246
IOMMU
251
CTRL. DE MEM.
248
MEMÓRIA
249

FIG. 2C

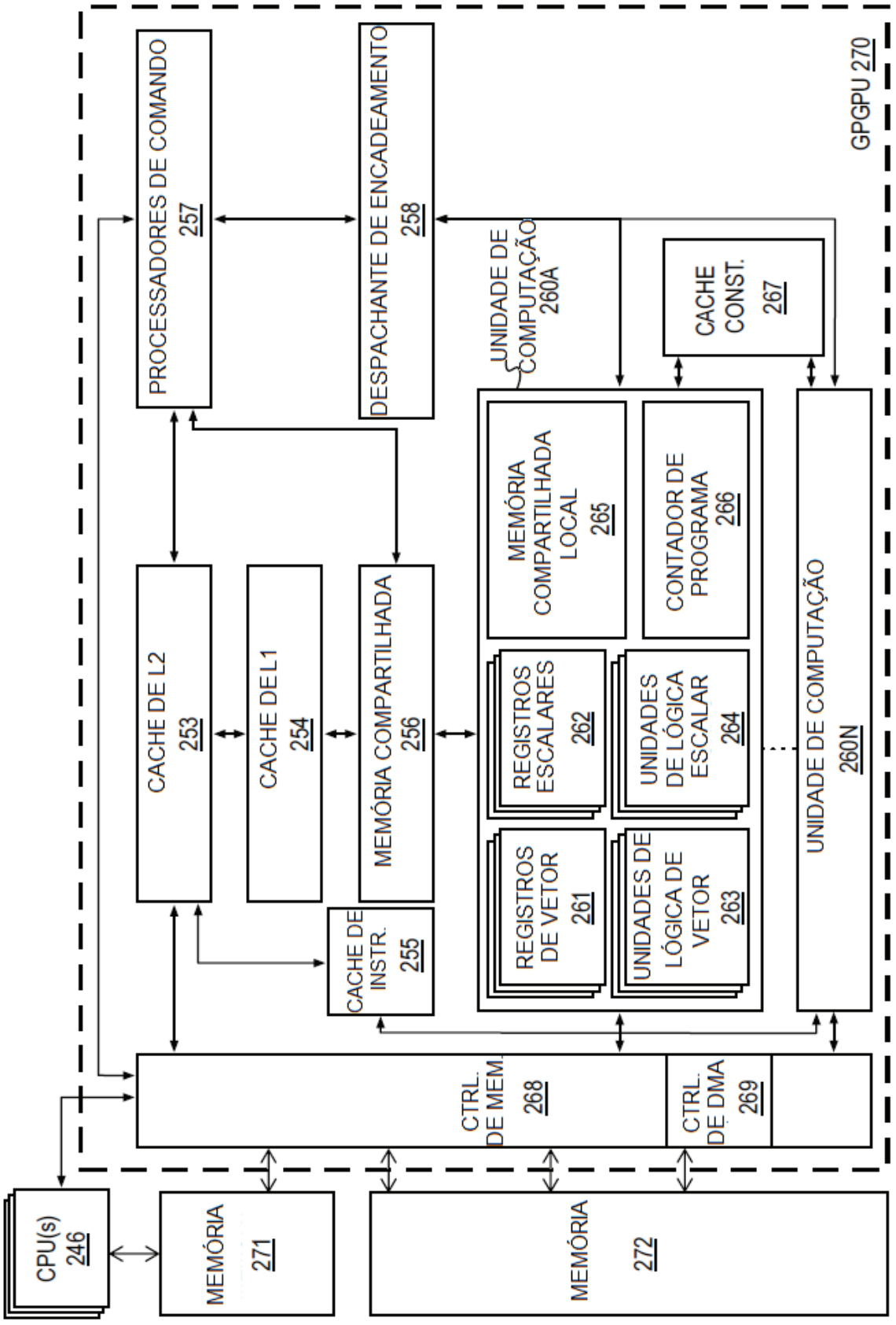

GPGPU 270
PROCESSADORES DE COMANDO 257
DESPACHANTE DE ENCADEAMENTO 258
UNIDADE DE COMPUTAÇÃO 260A
CACHE CONST. 267
MEMÓRIA COMPARTILHADA LOCAL 265
CONTADOR DE PROGRAMA 266
CACHE DE L2 253
CACHE DE L1 254
MEMÓRIA COMPARTILHADA 256
REGISTROS ESCALARES 262
UNIDADES DE LÓGICA ESCALAR 264
UNIDADE DE COMPUTAÇÃO 260N
CACHE DE INSTR. 255
REGISTROS DE VETOR 261
UNIDADES DE LÓGICA DE VETOR 263
CTRL. DE MEM. 268
CTRL. DE DMA 269
CPU(s) 246
MEMÓRIA 271
MEMÓRIA 272

FIG. 2D

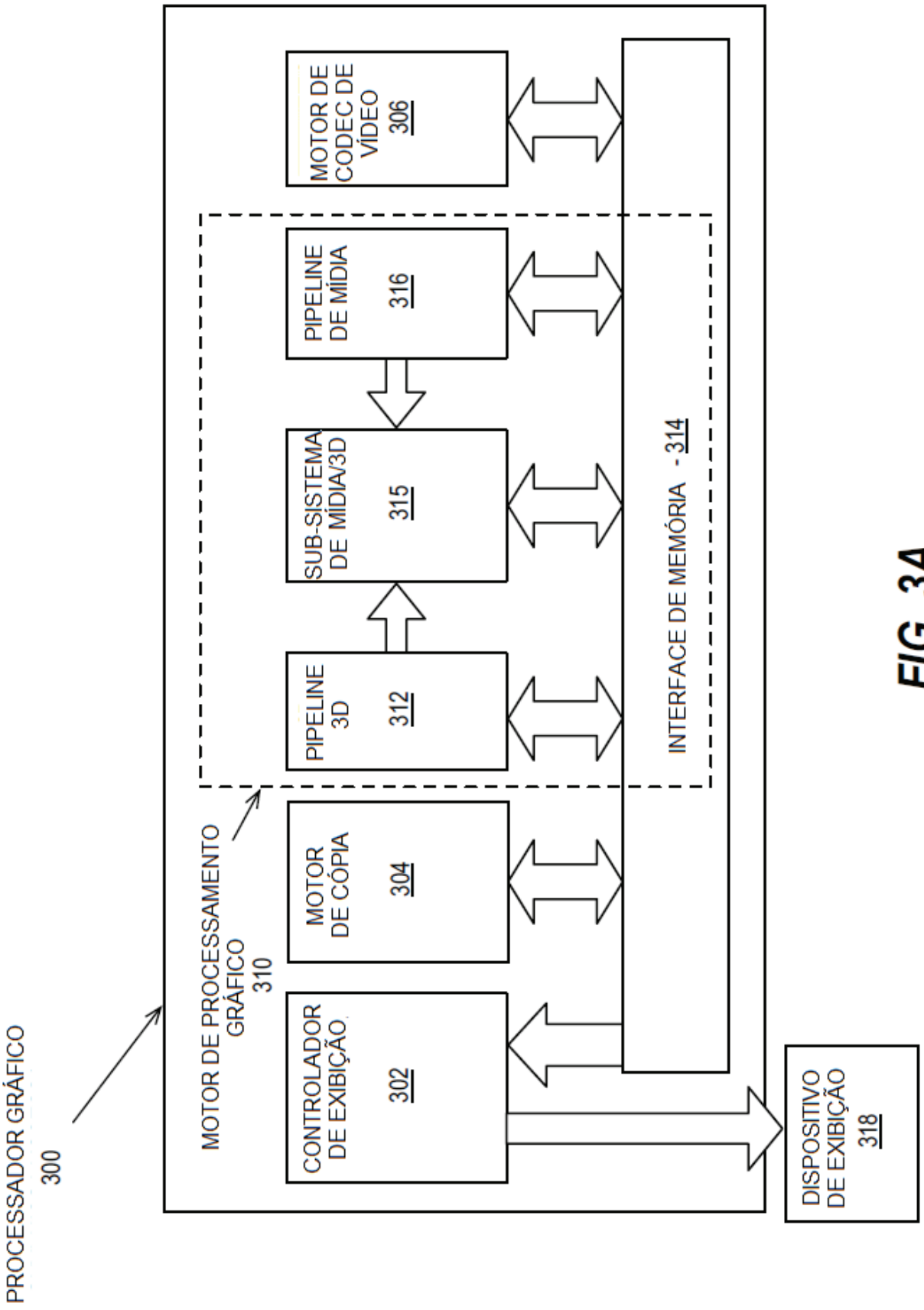

PROCESSADOR GRÁFICO
300
MOTOR DE PROCESSAMENTO GRÁFICO
310
CONTROLADOR DE EXIBIÇÃO
302
MOTOR DE CÓPIA
304
PIPELINE 3D
312
SUB-SISTEMA DE MÍDIA/3D
315
PIPELINE DE MÍDIA
316
MOTOR DE CODEC DE VÍDEO
306
INTERFACE DE MEMÓRIA - 314
DISPOSITIVO DE EXIBIÇÃO
318

FIG. 3A

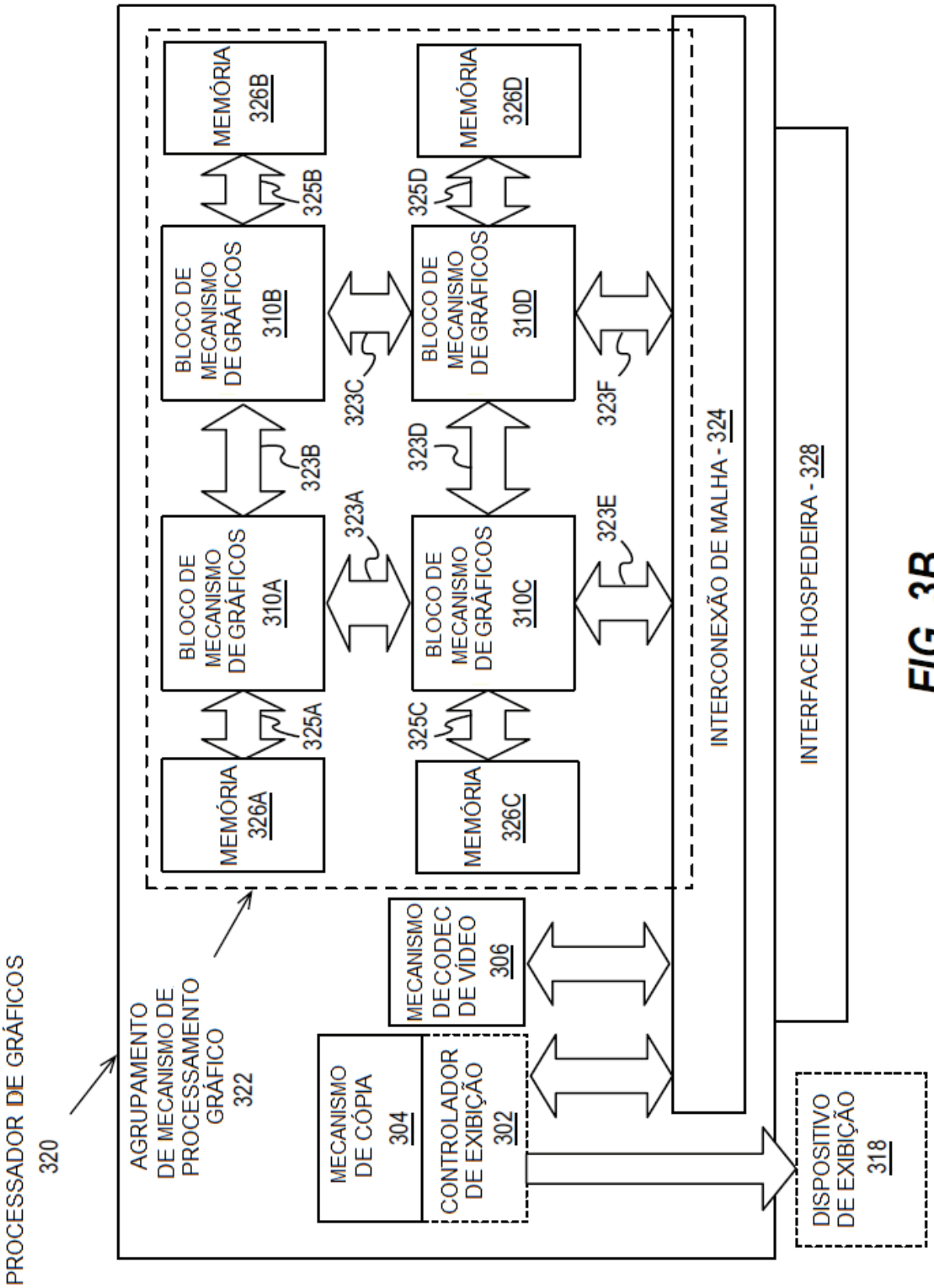

PROCESSADOR DE GRÁFICOS
320
AGRUPAMENTO DE MECANISMO DE PROCESSAMENTO GRÁFICO
322
MECANISMO DE CÓPIA
304
CONTROLADOR DE EXIBIÇÃO
302
MECANISMO DE CODEC DE VÍDEO
306
BLOCO DE MECANISMO DE GRÁFICOS
310A
BLOCO DE MECANISMO DE GRÁFICOS
310B
BLOCO DE MECANISMO DE GRÁFICOS
310C
BLOCO DE MECANISMO DE GRÁFICOS
310D
MEMÓRIA
326A
MEMÓRIA
326B
MEMÓRIA
326C
MEMÓRIA
326D
323A
323B
323C
323D
323E
323F
325A
325B
325C
325D
INTERCONEXÃO DE MALHA - 324
INTERFACE HOSPEDEIRA - 328
DISPOSITIVO DE EXIBIÇÃO
318

FIG. 3B

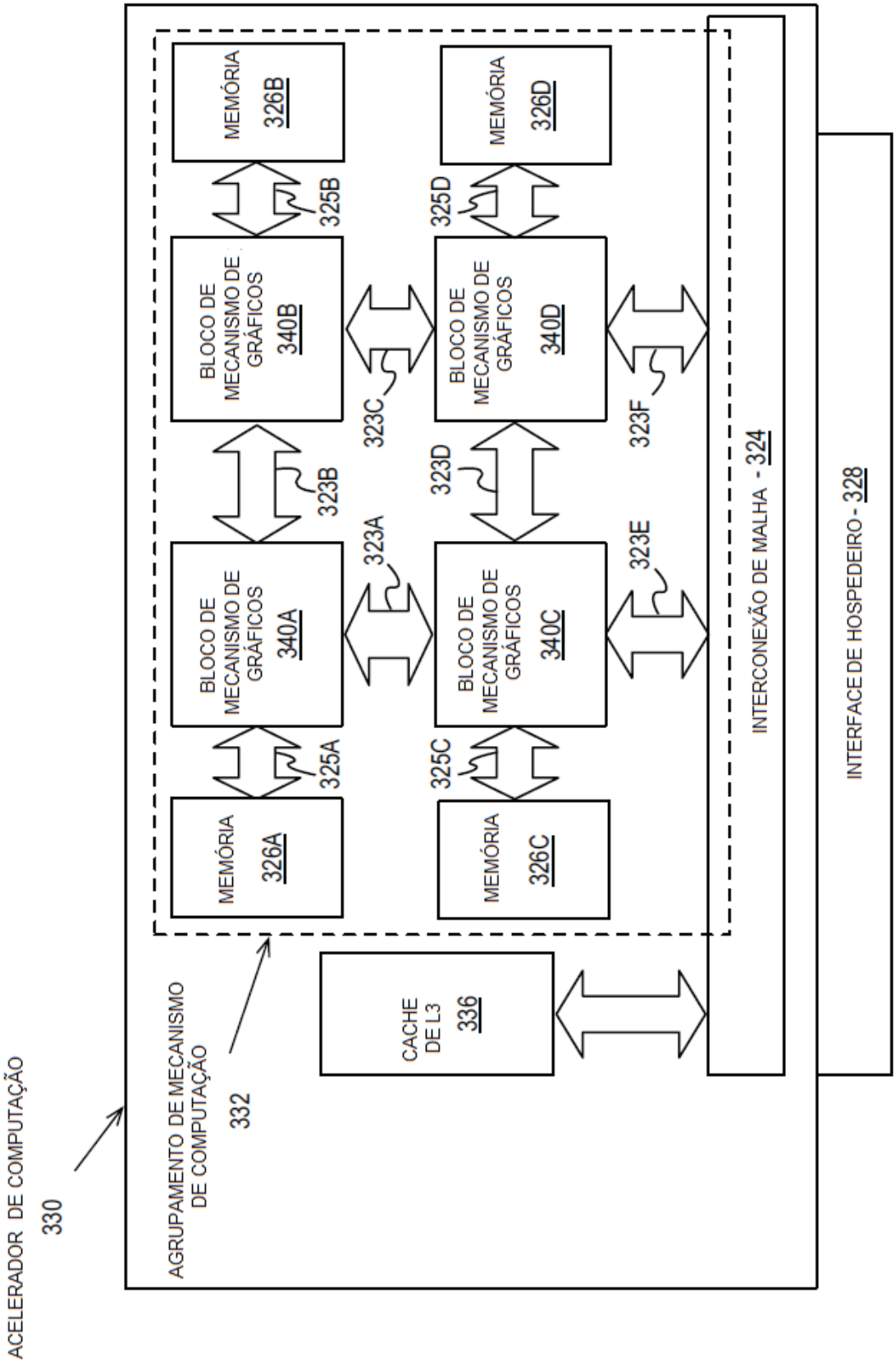

ACELERADOR DE COMPUTAÇÃO
330
AGRUPAMENTO DE MECANISMO DE COMPUTAÇÃO
332
MEMÓRIA 326A
MEMÓRIA 326B
MEMÓRIA 326C
MEMÓRIA 326D
BLOCO DE MECANISMO DE GRÁFICOS 340A
BLOCO DE MECANISMO DE GRÁFICOS 340B
BLOCO DE MECANISMO DE GRÁFICOS 340C
BLOCO DE MECANISMO DE GRÁFICOS 340D
CACHE DE L3 336
INTERCONEXÃO DE MALHA - 324
INTERFACE DE HOSPEDEIRO - 328
323A
323B
323C
323D
323E
323F
325A
325B
325C
325D

FIG. 3C

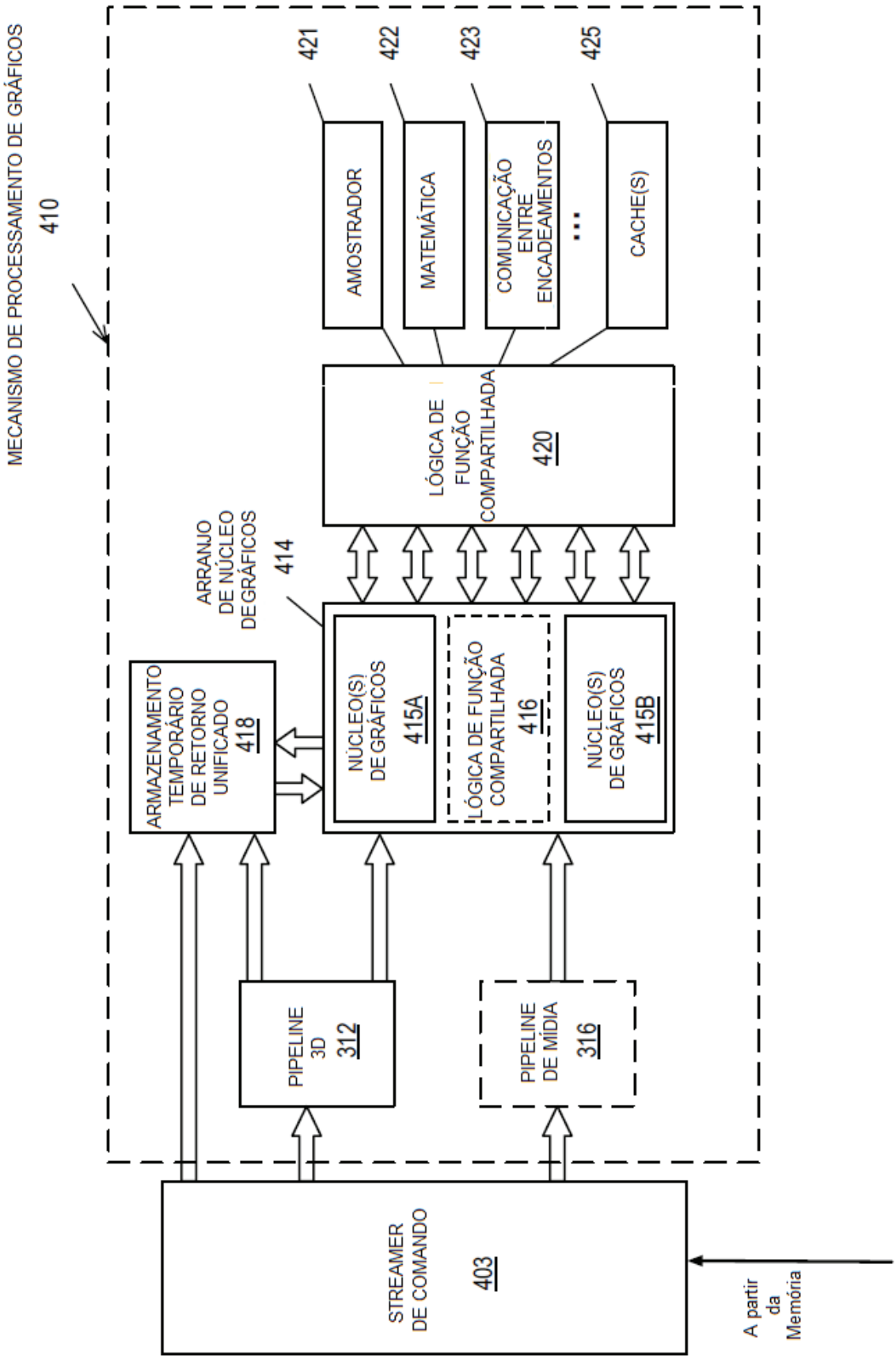

MECANISMO DE PROCESSAMENTO DE GRÁFICOS
410
421
422
423
425
AMOSTRADOR
MATEMÁTICA
COMUNICAÇÃO ENTRE ENCADEAMENTOS
...
CACHE(S)
LÓGICA DE FUNÇÃO COMPARTILHADA
420
ARRANJO DE NÚCLEO DE GRÁFICOS
414
ARMAZENAMENTO TEMPORÁRIO DE RETORNO UNIFICADO
418
NÚCLEO(S) DE GRÁFICOS
415A
LÓGICA DE FUNÇÃO COMPARTILHADA
416
NÚCLEO(S) DE GRÁFICOS
415B
PIPELINE 3D
312
PIPELINE DE MÍDIA
316
STREAMER DE COMANDO
403
A partir da Memória

FIG. 4

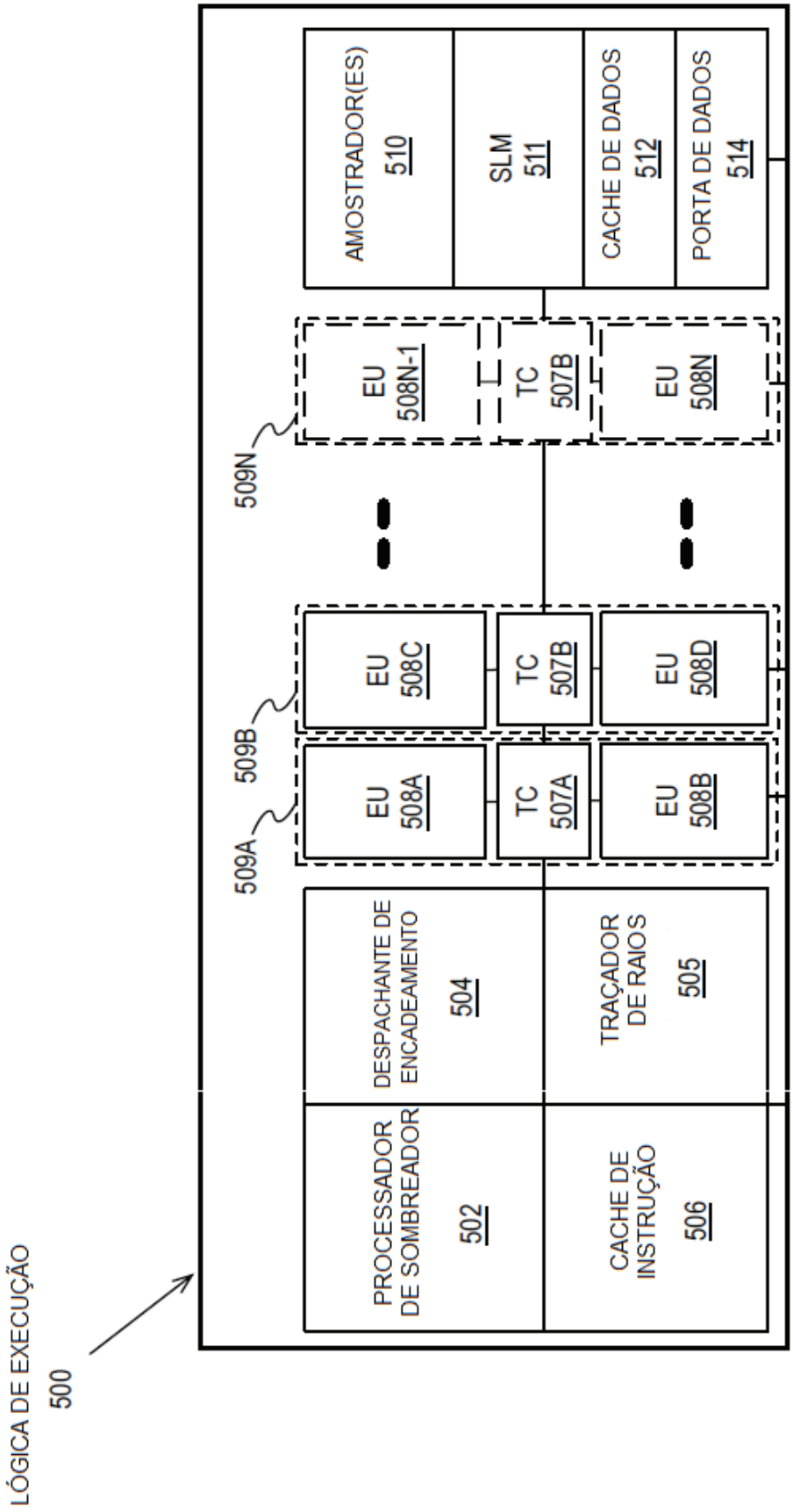

LÓGICA DE EXECUÇÃO
500
PROCESSADOR DE SOMBREADOR
502
DESPACHANTE DE ENCADEAMENTO
504
CACHE DE INSTRUÇÃO
506
TRAÇADOR DE RAIOS
505
AMOSTRADOR(ES)
510
SLM
511
CACHE DE DADOS
512
PORTA DE DADOS
514
509A
EU 508A
TC 507A
EU 508B
509B
EU 508C
TC 507B
EU 508D
509N
EU 508N-1
TC 507B
EU 508N

FIG. 5A

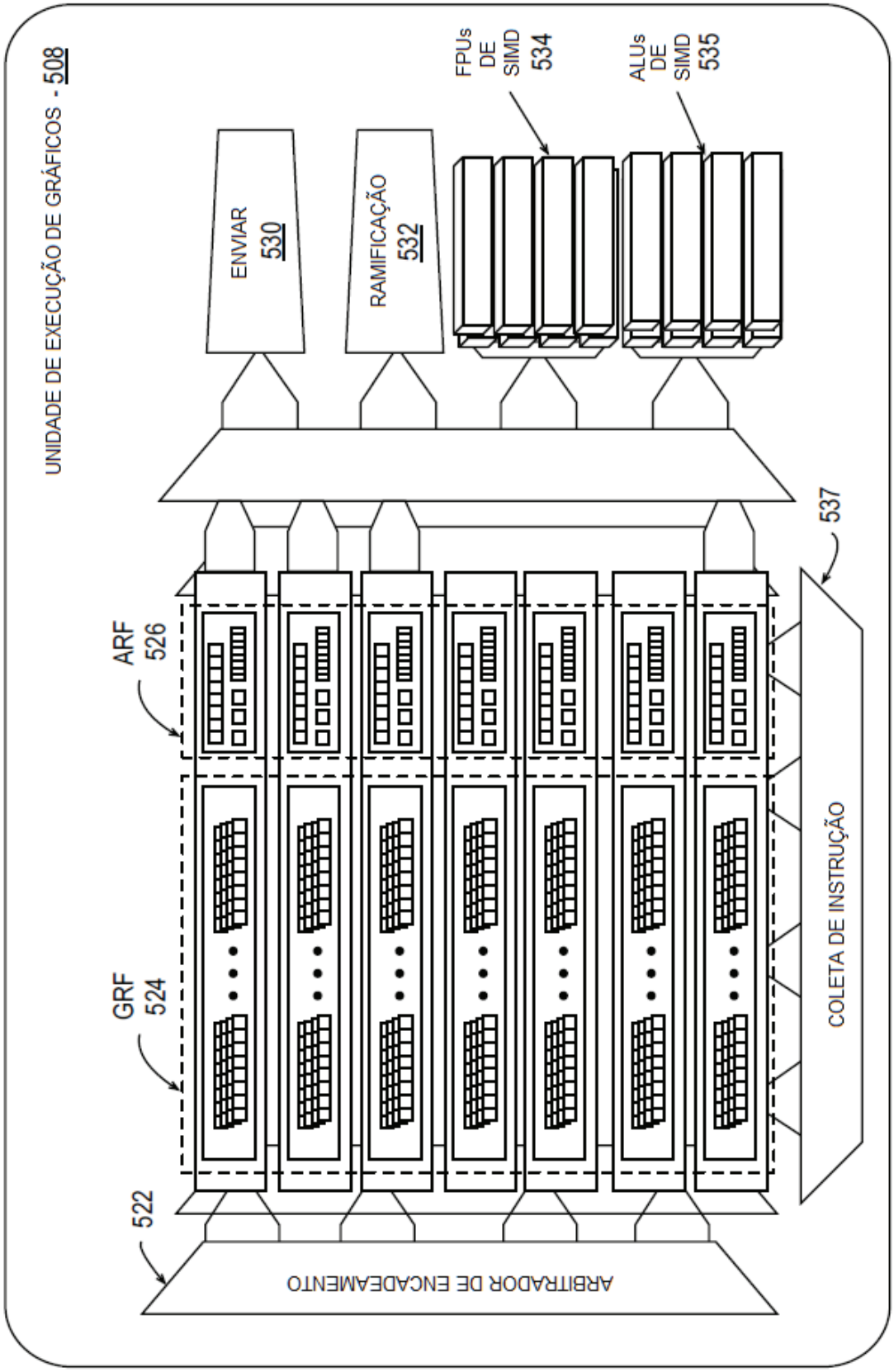

UNIDADE DE EXECUÇÃO DE GRÁFICOS - 508
ENVIAR 530
RAMIFICAÇÃO 532
FPUs DE SIMD 534
ALUs DE SIMD 535
537
ARF 526
GRF 524
522
ARBITRADOR DE ENCADEAMENTO
COLETA DE INSTRUÇÃO

FIG. 5B

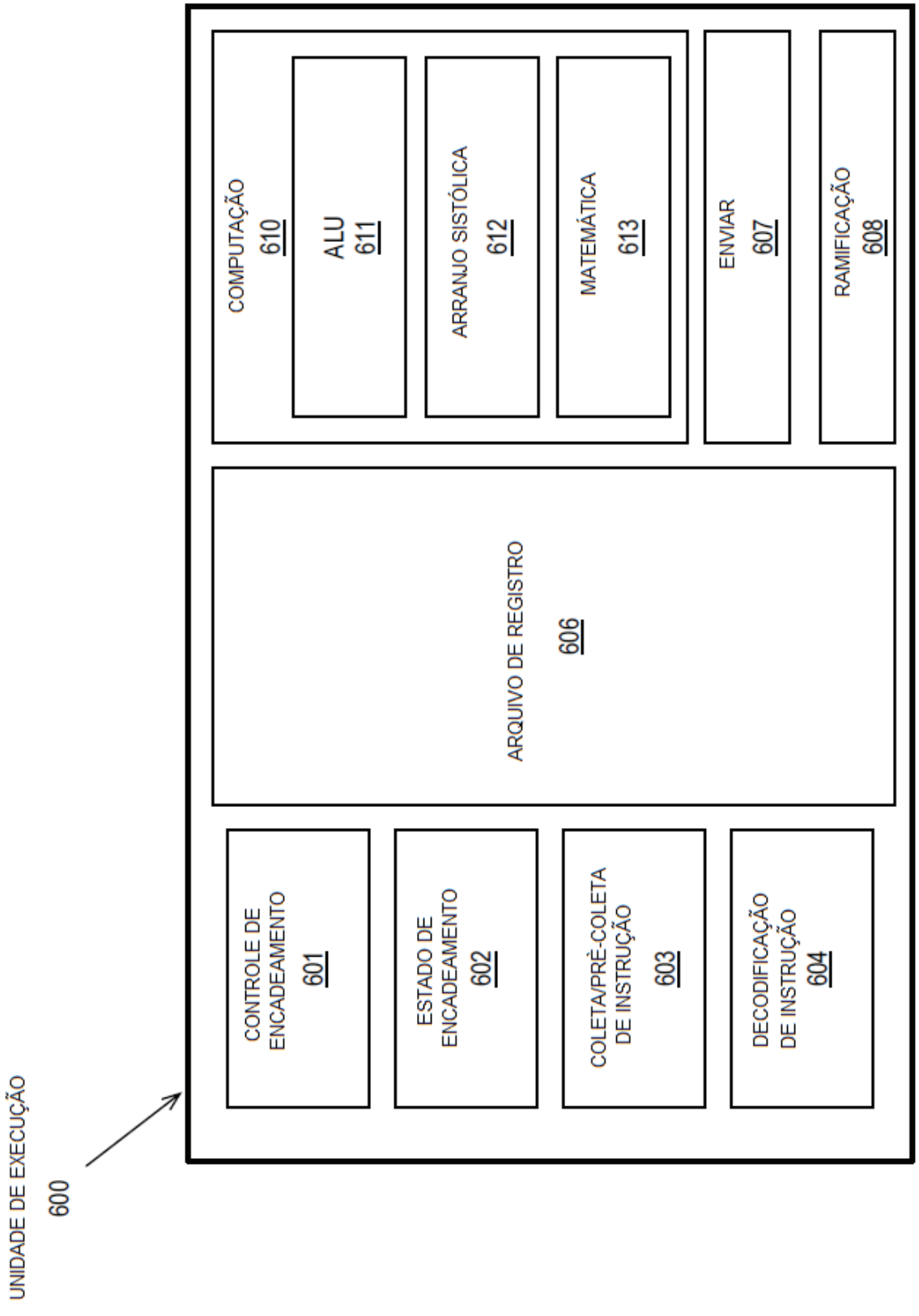

UNIDADE DE EXECUÇÃO
600
CONTROLE DE ENCADEAMENTO
601
ESTADO DE ENCADEAMENTO
602
COLETA/PRÉ-COLETA DE INSTRUÇÃO
603
DECODIFICAÇÃO DE INSTRUÇÃO
604
ARQUIVO DE REGISTRO
606
COMPUTAÇÃO
610
ALU
611
ARRANJO SISTÓLICA
612
MATEMÁTICA
613
ENVIAR
607
RAMIFICAÇÃO
608

FIG. 6

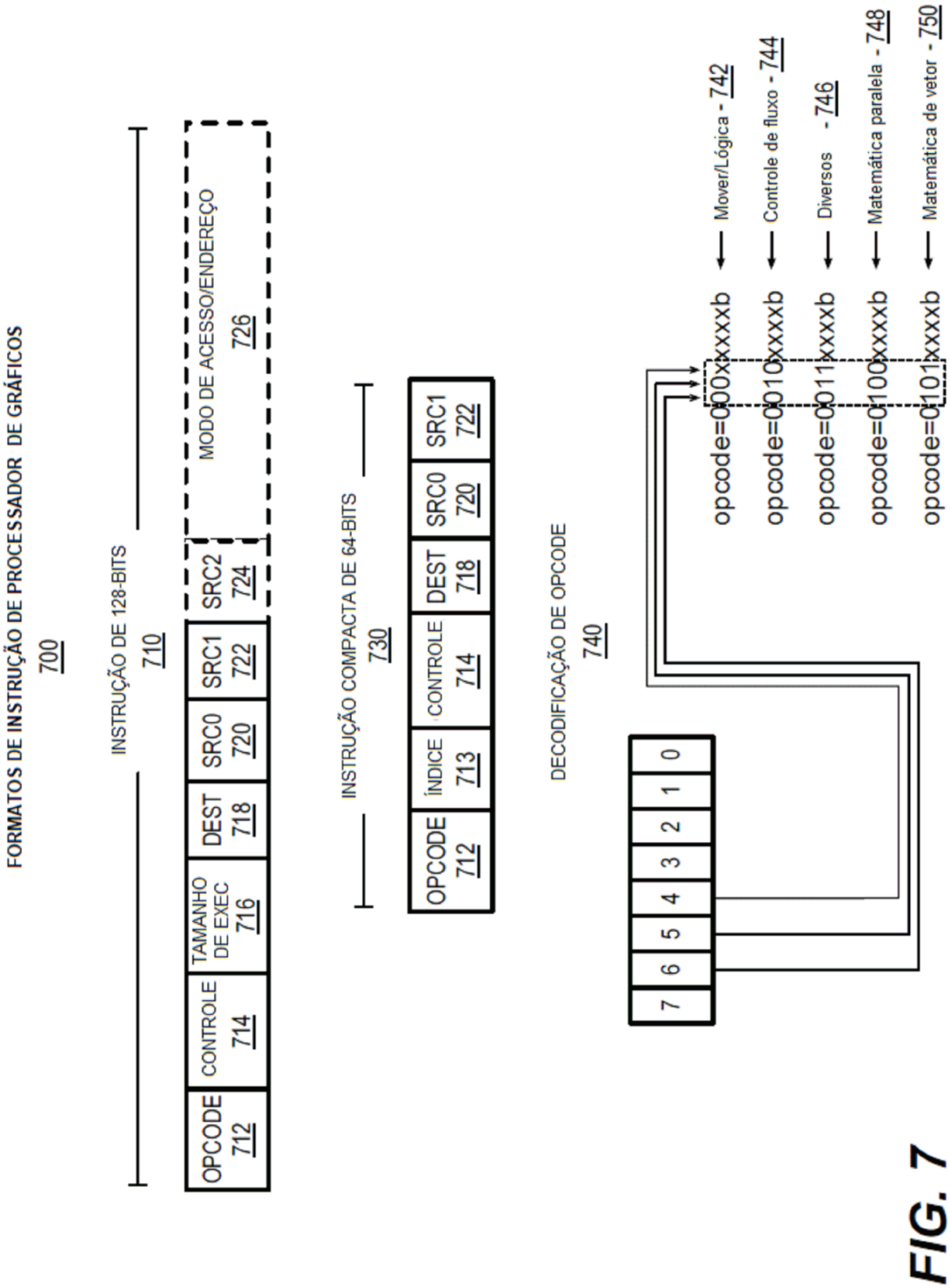

FORMATOS DE INSTRUÇÃO DE PROCESSADOR DE GRÁFICOS
700
INSTRUÇÃO DE 128-BITS
710
OPCODE 712
CONTROLE 714
TAMANHO DE EXEC 716
DEST 718
SRC0 720
SRC1 722
SRC2 724
MODO DE ACESSO/ENDEREÇO 726
INSTRUÇÃO COMPACTA DE 64-BITS
730
OPCODE 712
ÍNDICE 713
CONTROLE 714
DEST 718
SRC0 720
SRC1 722
DECODIFICAÇÃO DE OPCODE
740
7 6 5 4 3 2 1 0
opcode=00xxxxb
opcode=010xxxb
opcode=011xxxb
opcode=0100xxxb
opcode=0101xxxb
Mover/Lógica - 742
Controle de fluxo - 744
Diversos - 746
Matemática paralela - 748
Matemática de vetor - 750

FIG. 7

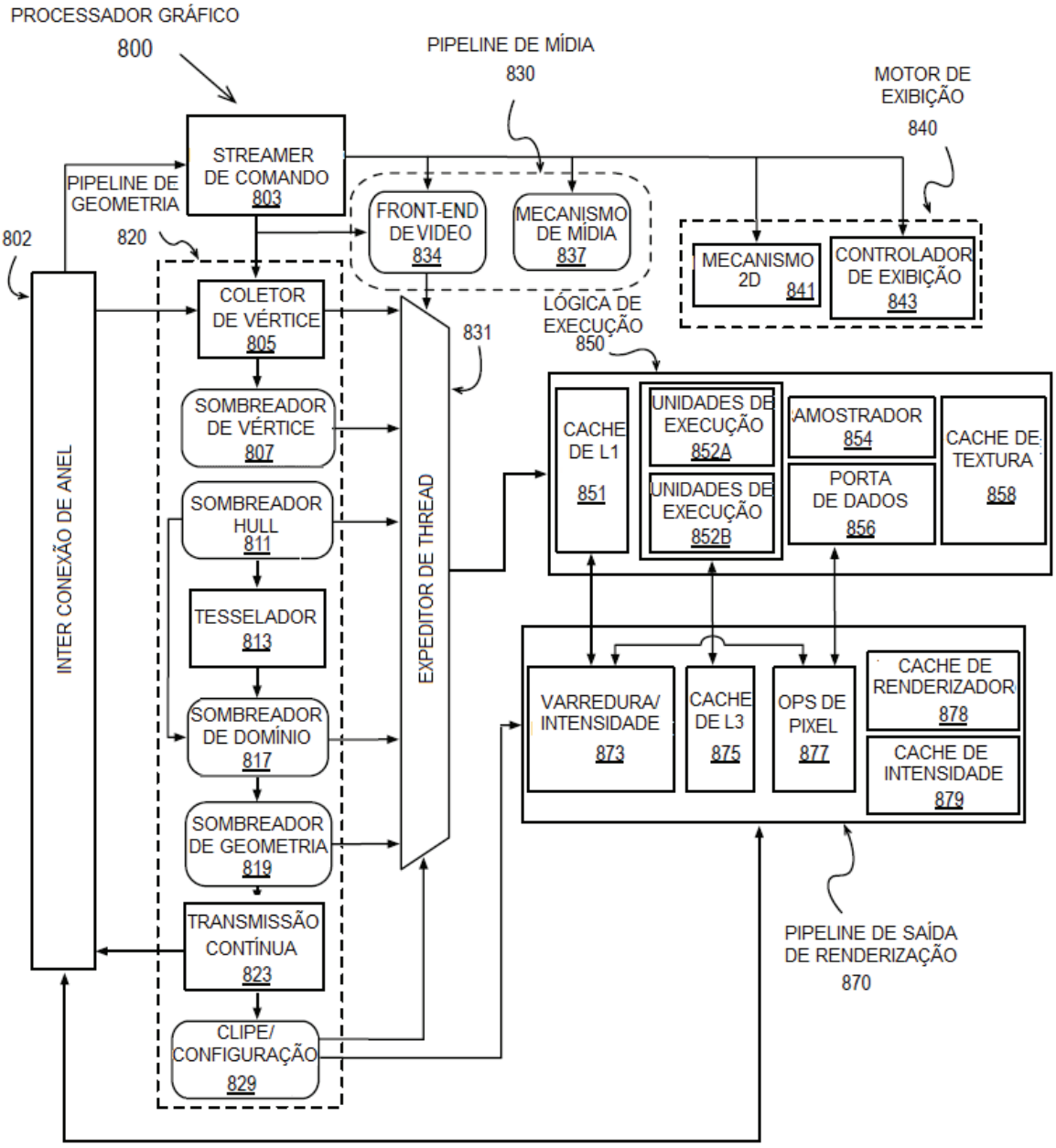

PROCESSADOR GRÁFICO
800
PIPELINE DE MÍDIA
830
MOTOR DE EXIBIÇÃO
840
STREAMER DE COMANDO
803
PIPELINE DE GEOMETRIA
820
802
FRONT-END DE VÍDEO
834
MECANISMO DE MÍDIA
837
MECANISMO 2D
841
CONTROLADOR DE EXIBIÇÃO
843
LÓGICA DE EXECUÇÃO
850
831
COLETOR DE VÉRTICE
805
SOMBREADOR DE VÉRTICE
807
SOMBREADOR HULL
811
TESSELADOR
813
SOMBREADOR DE DOMÍNIO
817
SOMBREADOR DE GEOMETRIA
819
TRANSMISSÃO CONTÍNUA
823
CLIPE/ CONFIGURAÇÃO
829
INTER CONEXÃO DE ANEL
EXPEDITOR DE THREAD
CACHE DE L1
851
UNIDADES DE EXECUÇÃO
852A
UNIDADES DE EXECUÇÃO
852B
AMOSTRADOR
854
PORTA DE DADOS
856
CACHE DE TEXTURA
858
VARREDURA/ INTENSIDADE
873
CACHE DE L3
875
OPS DE PIXEL
877
CACHE DE RENDERIZADOR
878
CACHE DE INTENSIDADE
879
PIPELINE DE SAÍDA DE RENDERIZAÇÃO
870

FIG. 8

FIG. 9A

FORMATO DE COMANDO DE PROCESSADOR DE GRÁFICOS

900

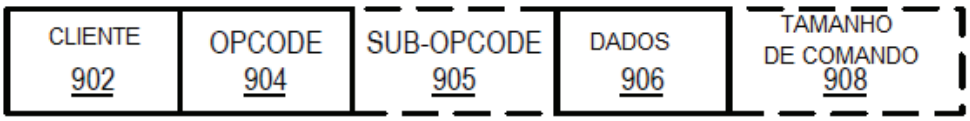
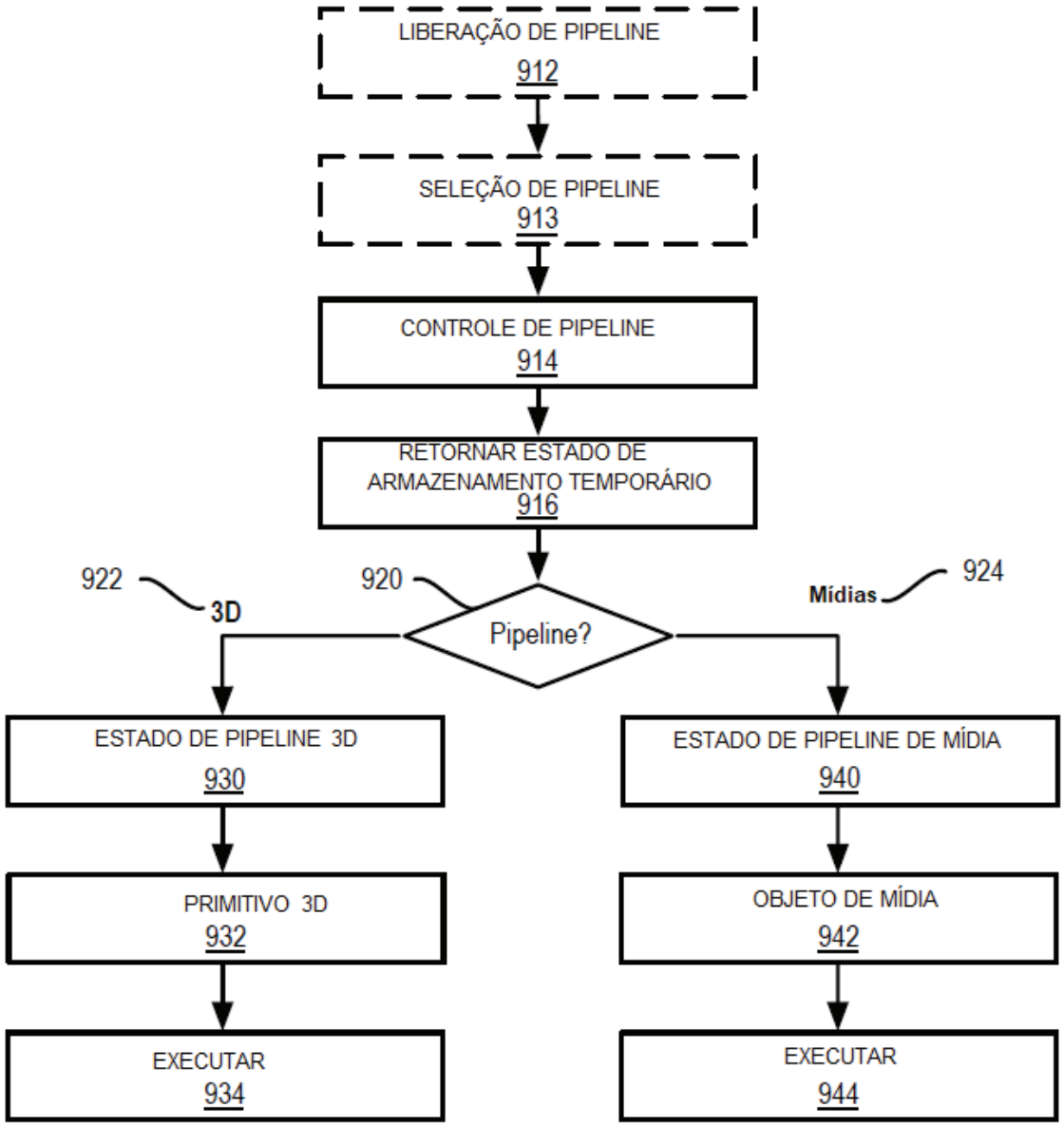


FIG. 9B

SEQUÊNCIA DE COMANDO DE PROCESSADOR DE GRÁFICOS

910



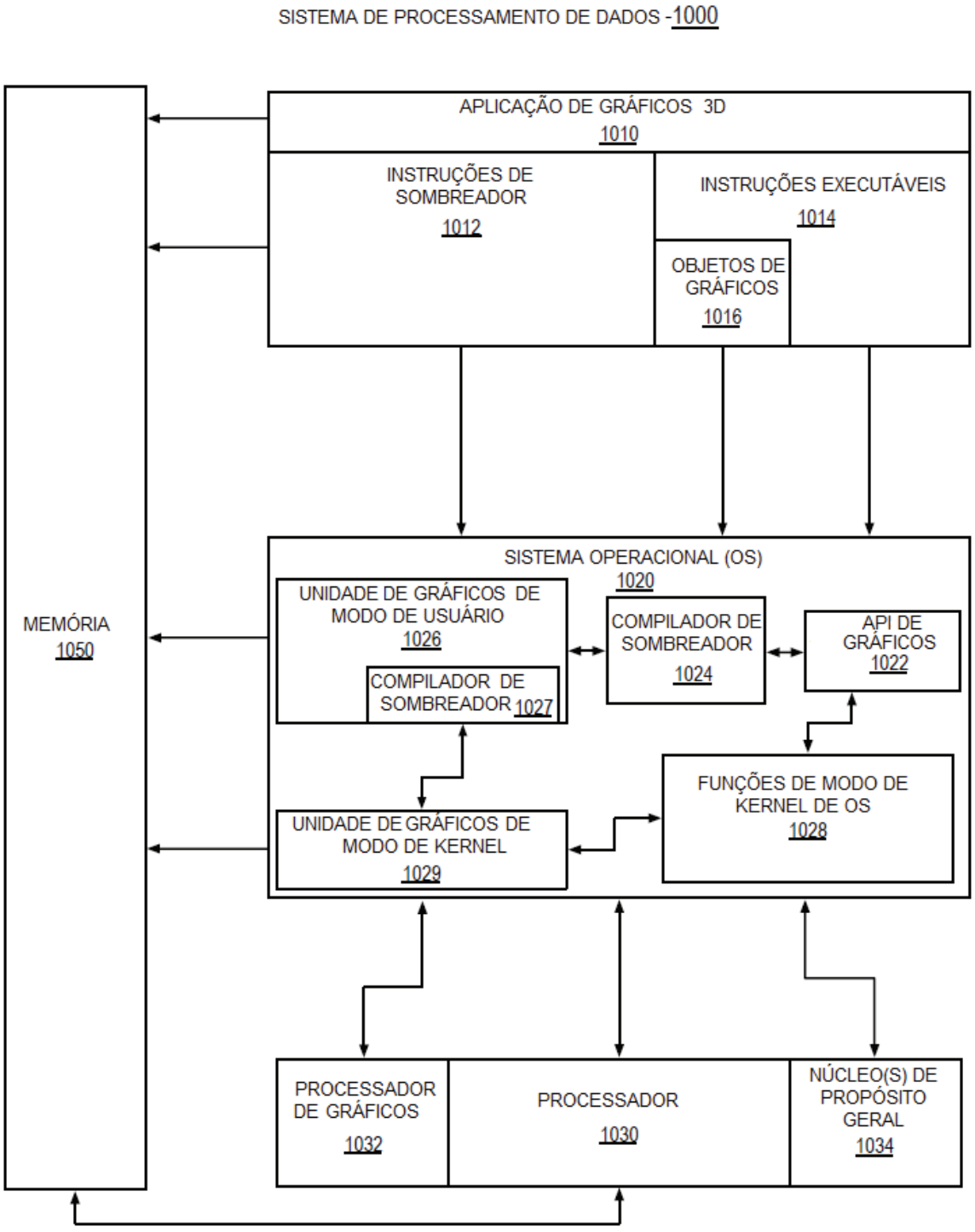

SISTEMA DE PROCESSAMENTO DE DADOS -1000
APLICAÇÃO DE GRÁFICOS 3D
1010
INSTRUÇÕES DE SOMBREADOR
1012
INSTRUÇÕES EXECUTÁVEIS
1014
OBJETOS DE GRÁFICOS
1016
MEMÓRIA
1050
SISTEMA OPERACIONAL (OS)
1020
UNIDADE DE GRÁFICOS DE MODO DE USUÁRIO
1026
COMPILADOR DE SOMBREADOR
1027
COMPILADOR DE SOMBREADOR
1024
API DE GRÁFICOS
1022
FUNÇÕES DE MODO DE KERNEL DE OS
1028
UNIDADE DE GRÁFICOS DE MODO DE KERNEL
1029
PROCESSADOR DE GRÁFICOS
1032
PROCESSADOR
1030
NÚCLEO(S) DE PROPÓSITO GERAL
1034

FIG. 10

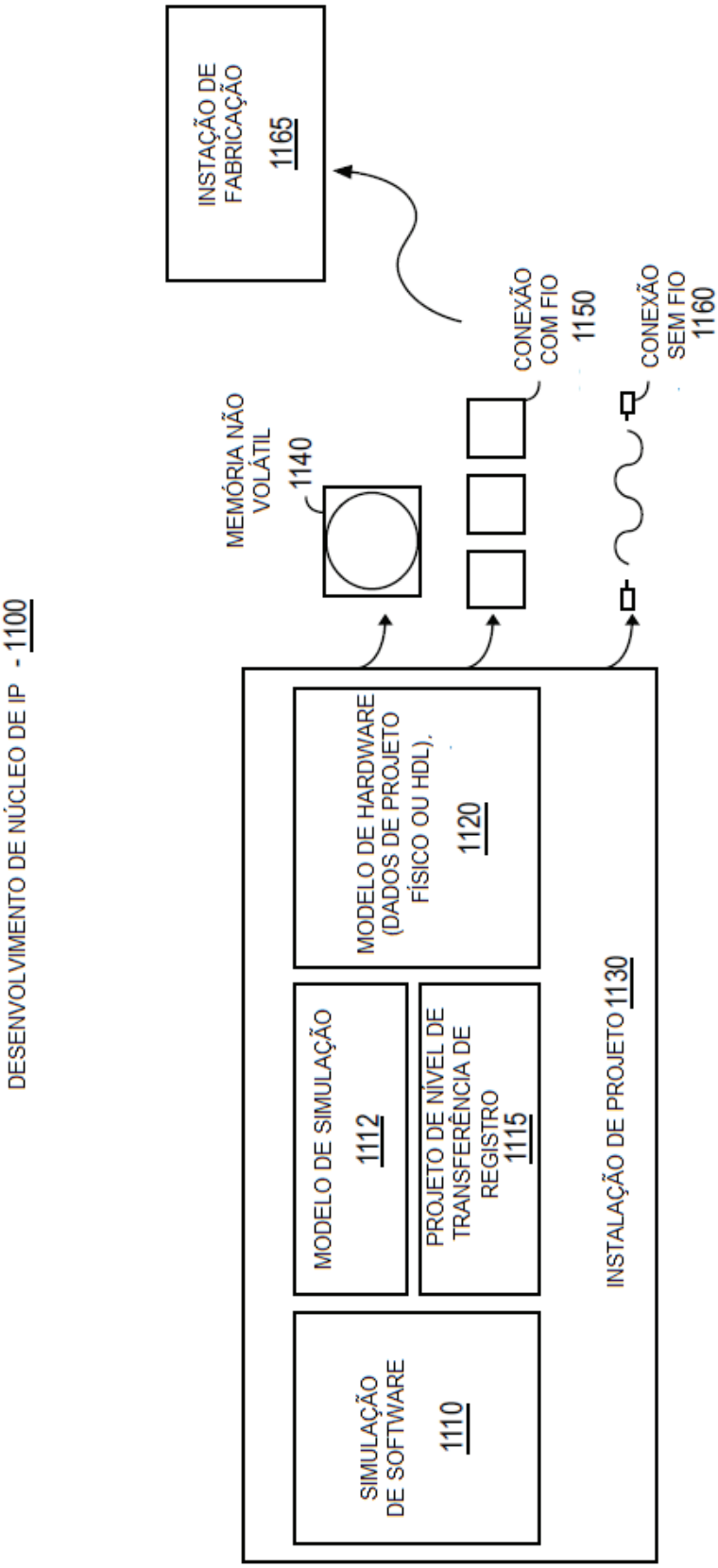

DESENVOLVIMENTO DE NÚCLEO DE IP - 1100
INSTALAÇÃO DE PROJETO 1130
SIMULAÇÃO DE SOFTWARE 1110
MODELO DE SIMULAÇÃO 1112
PROJETO DE NÍVEL DE TRANSFERÊNCIA DE REGISTRO 1115
MODELO DE HARDWARE (DADOS DE PROJETO FÍSICO OU HDL), 1120
MEMÓRIA NÃO VOLÁTIL 1140
CONEXÃO COM FIO 1150
CONEXÃO SEM FIO 1160
INSTAÇÃO DE FABRICAÇÃO 1165

FIG. 11A

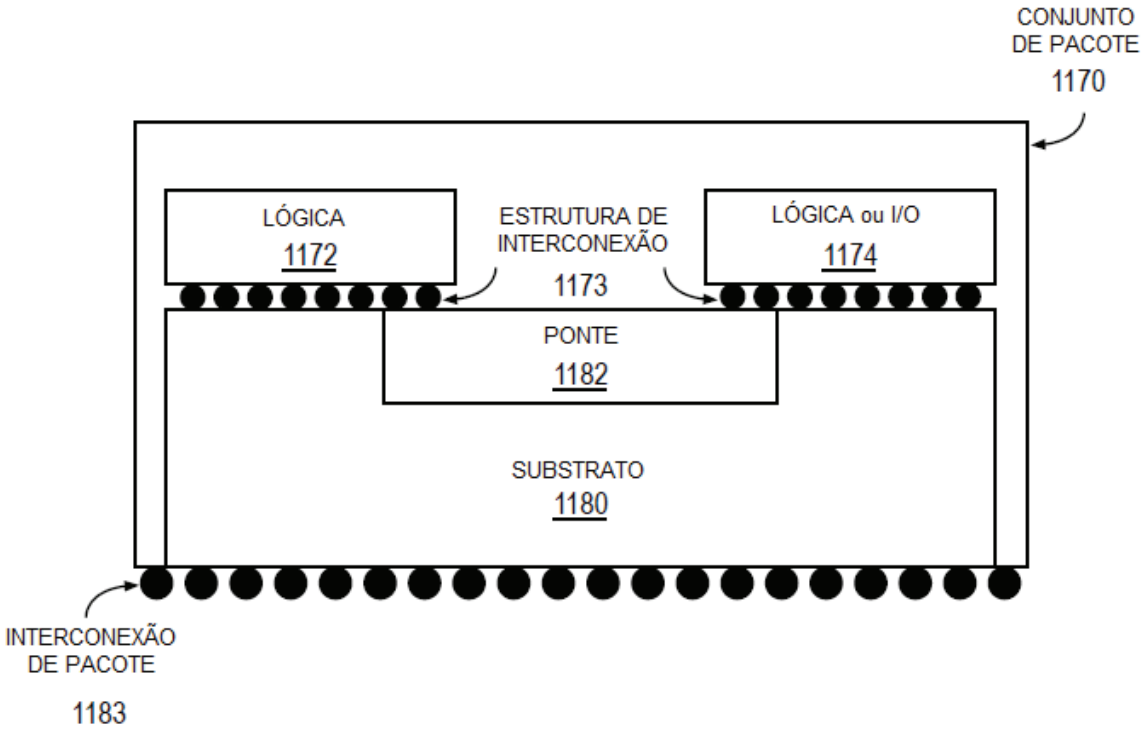

CONJUNTO
DE PACOTE
1170
LÓGICA
1172
ESTRUTURA DE
INTERCONEXÃO
1173
LÓGICA ou I/O
1174
PONTE
1182
SUBSTRATO
1180
INTERCONEXÃO
DE PACOTE
1183

FIG. 11B

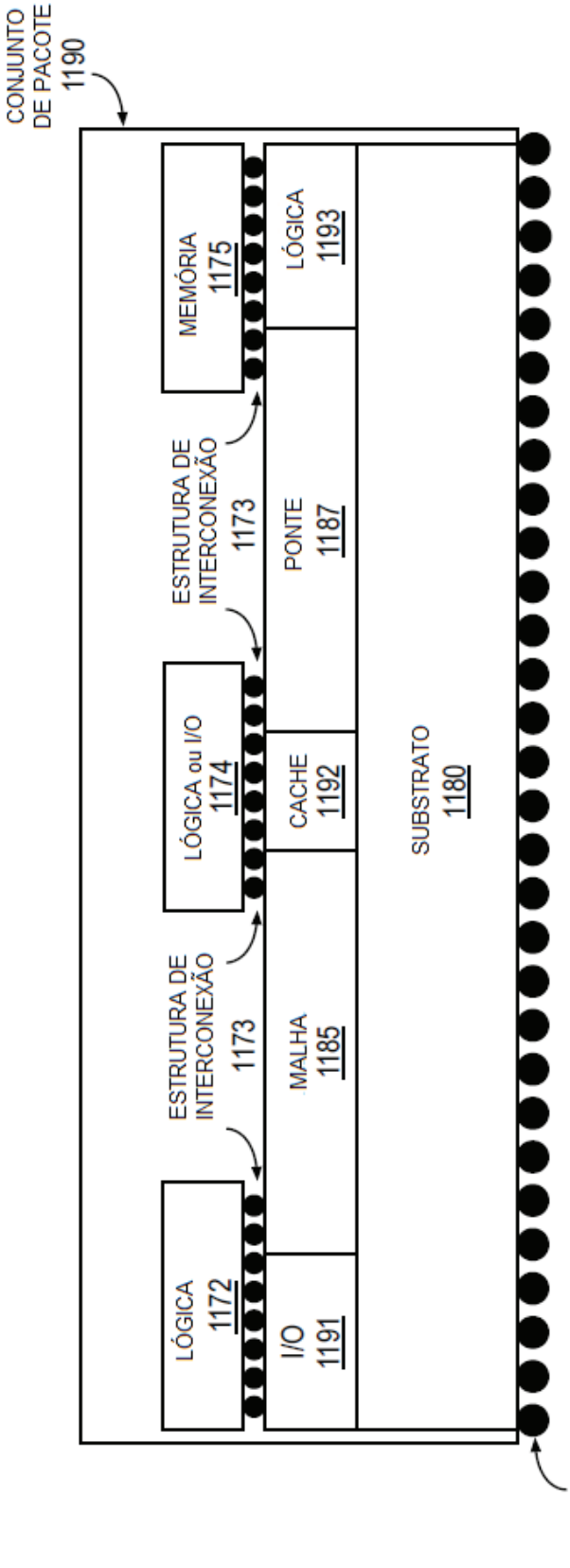

CONJUNTO DE PACOTE 1190
MEMÓRIA 1175
LÓGICA 1193
ESTRUTURA DE INTERCONEXÃO 1173
PONTE 1187
LÓGICA ou I/O 1174
CACHE 1192
SUBSTRATO 1180
ESTRUTURA DE INTERCONEXÃO 1173
MALHA 1185
LÓGICA 1172
I/O 1191
INTERCONEXÃO DE PACOTE 1183

FIG. 11C

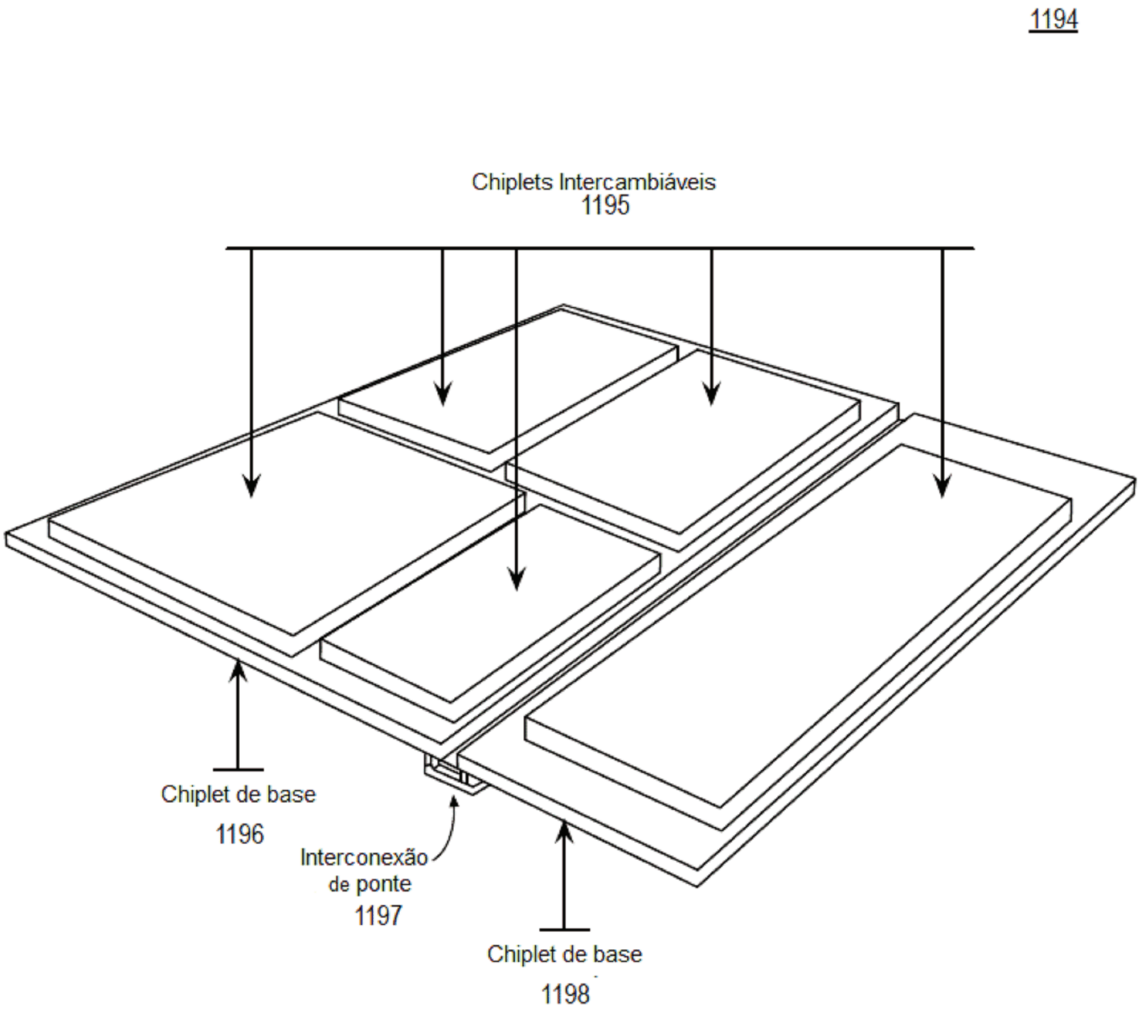

1194
Chiplets Intercambiáveis
1195
Chiplet de base
1196
Interconexão
de ponte
1197
Chiplet de base
1198

FIG. 11D

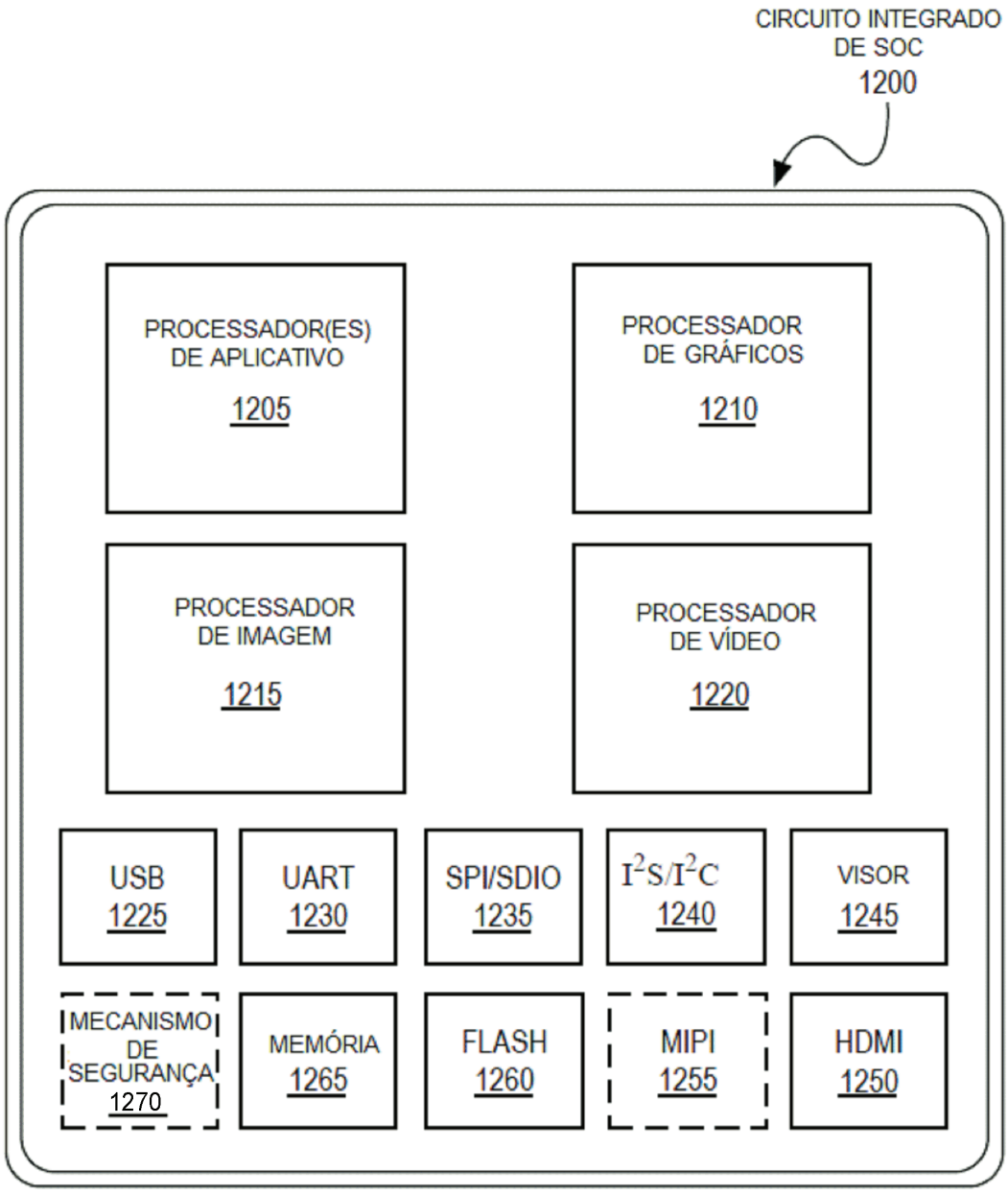

CIRCUITO INTEGRADO
DE SOC
1200
PROCESSADOR(ES)
DE APLICATIVO
1205
PROCESSADOR
DE GRÁFICOS
1210
PROCESSADOR
DE IMAGEM
1215
PROCESSADOR
DE VÍDEO
1220
USB
1225
UART
1230
SPI/SDIO
1235
I2S/I2C
1240
VISOR
1245
MECANISMO
DE
SEGURANÇA
1270
MEMÓRIA
1265
FLASH
1260
MIPI
1255
HDMI
1250

FIG. 12

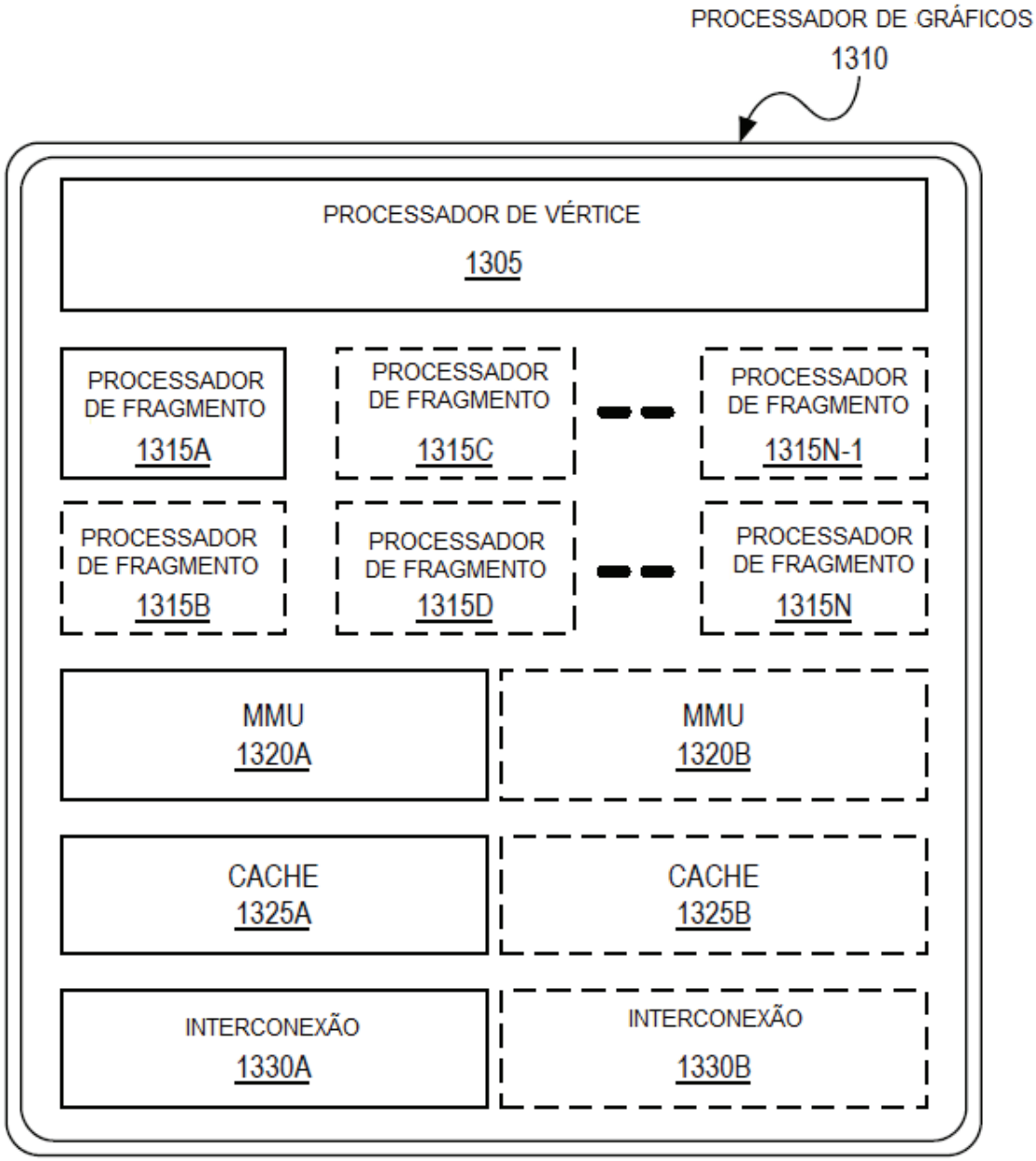

PROCESSADOR DE GRÁFICOS
1310
PROCESSADOR DE VÉRTICE
1305
PROCESSADOR DE FRAGMENTO
1315A
PROCESSADOR DE FRAGMENTO
1315C
PROCESSADOR DE FRAGMENTO
1315N-1
PROCESSADOR DE FRAGMENTO
1315B
PROCESSADOR DE FRAGMENTO
1315D
PROCESSADOR DE FRAGMENTO
1315N
MMU
1320A
MMU
1320B
CACHE
1325A
CACHE
1325B
INTERCONEXÃO
1330A
INTERCONEXÃO
1330B

FIG. 13A

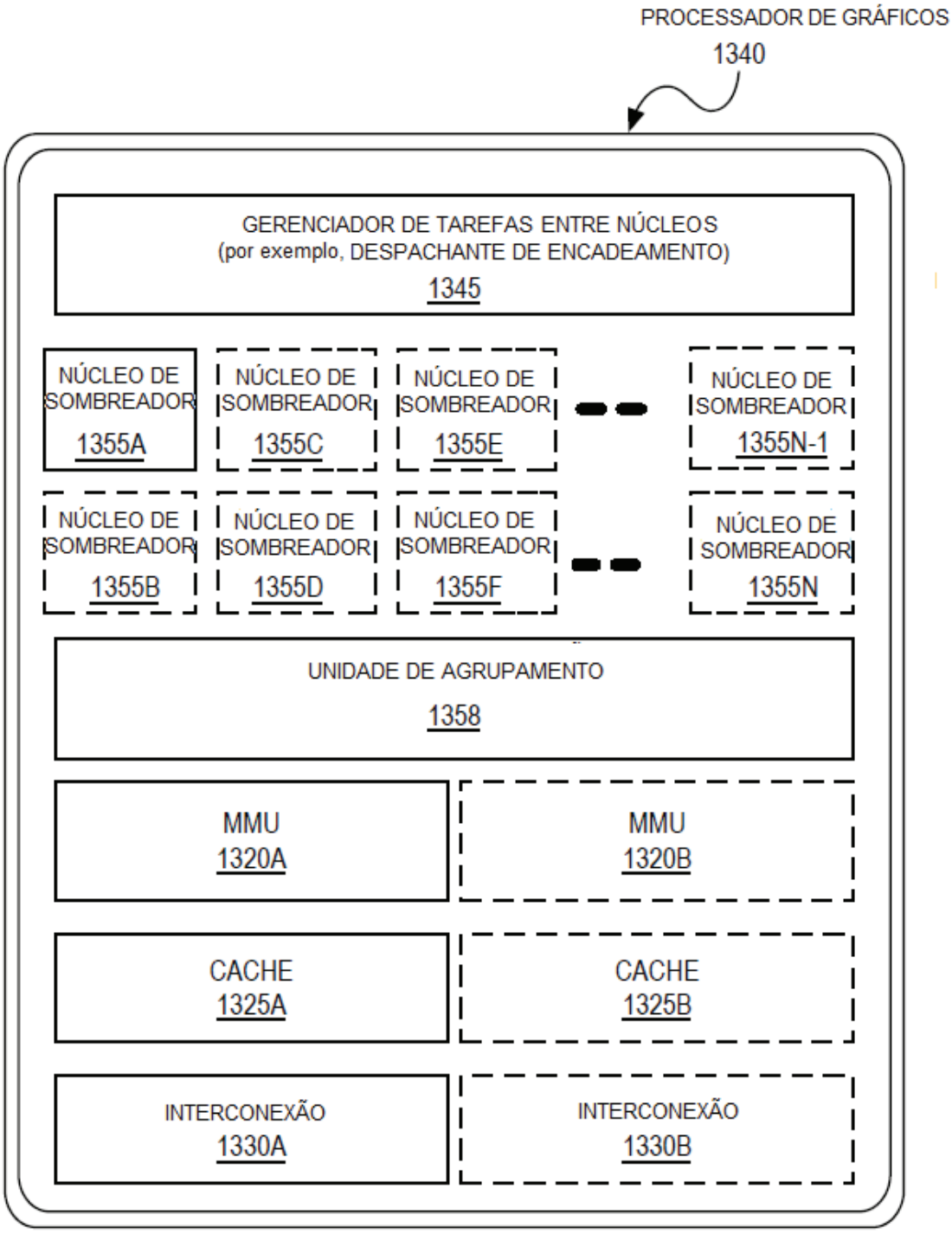

PROCESSADOR DE GRÁFICOS
1340
GERENCIADOR DE TAREFAS ENTRE NÚCLEOS
(por exemplo, DESPACHANTE DE ENCADEAMENTO)
1345
NÚCLEO DE SOMBREADOR
1355A
NÚCLEO DE SOMBREADOR
1355C
NÚCLEO DE SOMBREADOR
1355E
NÚCLEO DE SOMBREADOR
1355N-1
NÚCLEO DE SOMBREADOR
1355B
NÚCLEO DE SOMBREADOR
1355D
NÚCLEO DE SOMBREADOR
1355F
NÚCLEO DE SOMBREADOR
1355N
UNIDADE DE AGRUPAMENTO
1358
MMU
1320A
MMU
1320B
CACHE
1325A
CACHE
1325B
INTERCONEXÃO
1330A
INTERCONEXÃO
1330B

FIG. 13B

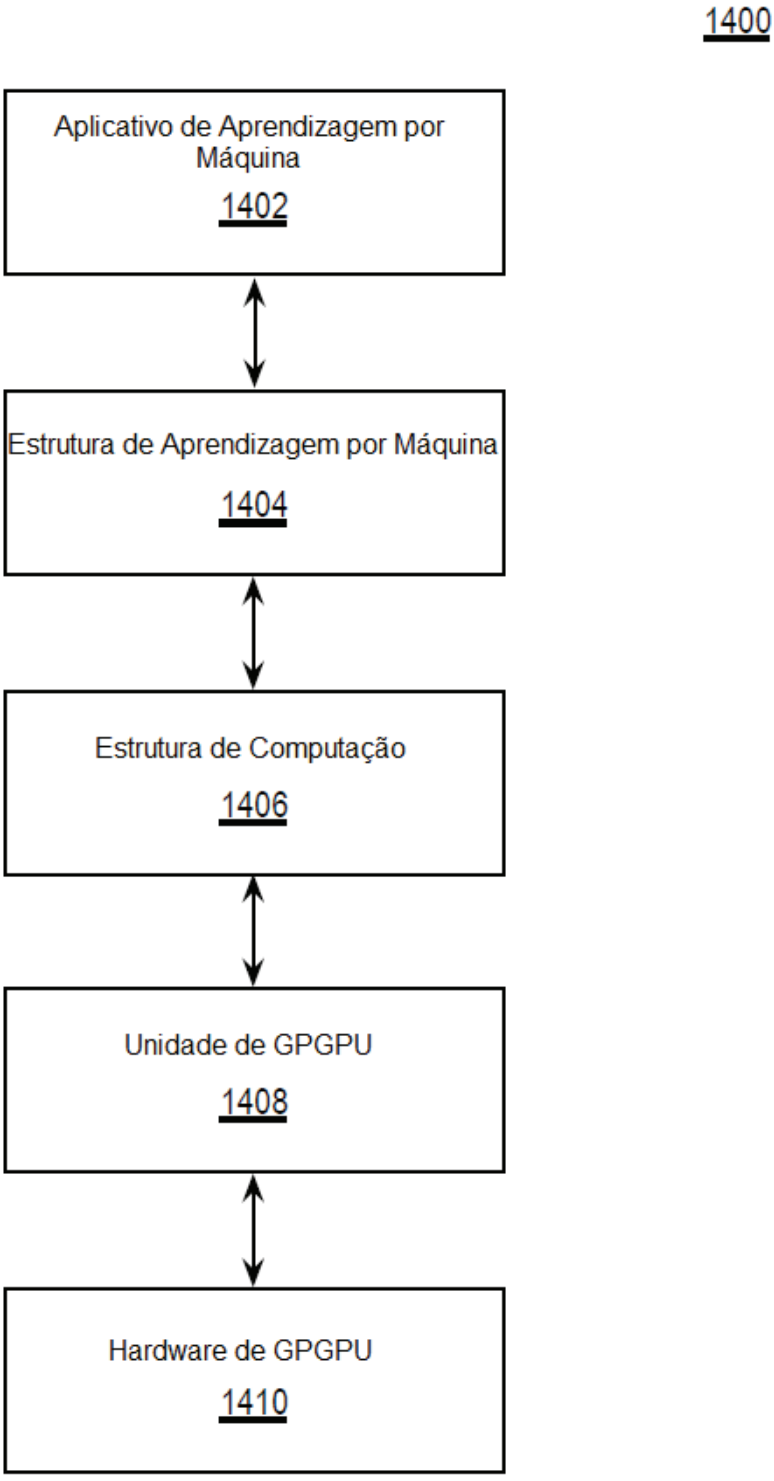

1400
Aplicativo de Aprendizagem por Máquina
1402
Estrutura de Aprendizagem por Máquina
1404
Estrutura de Computação
1406
Unidade de GPGPU
1408
Hardware de GPGPU
1410

FIG. 14

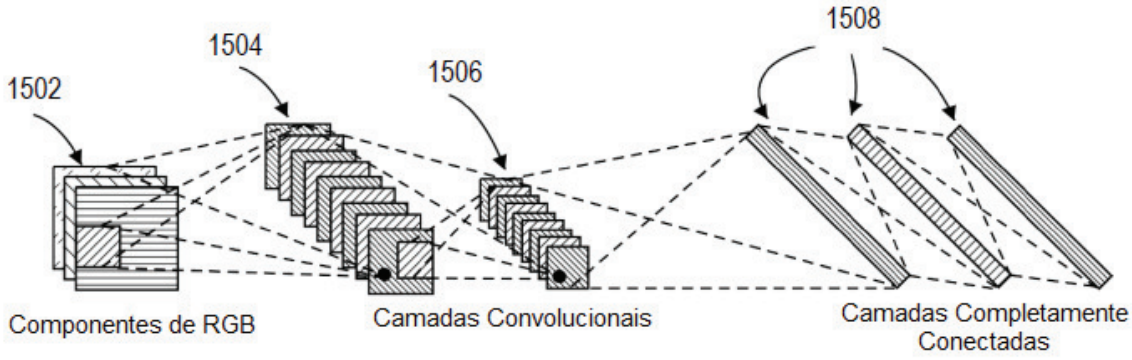


FIG. 15A

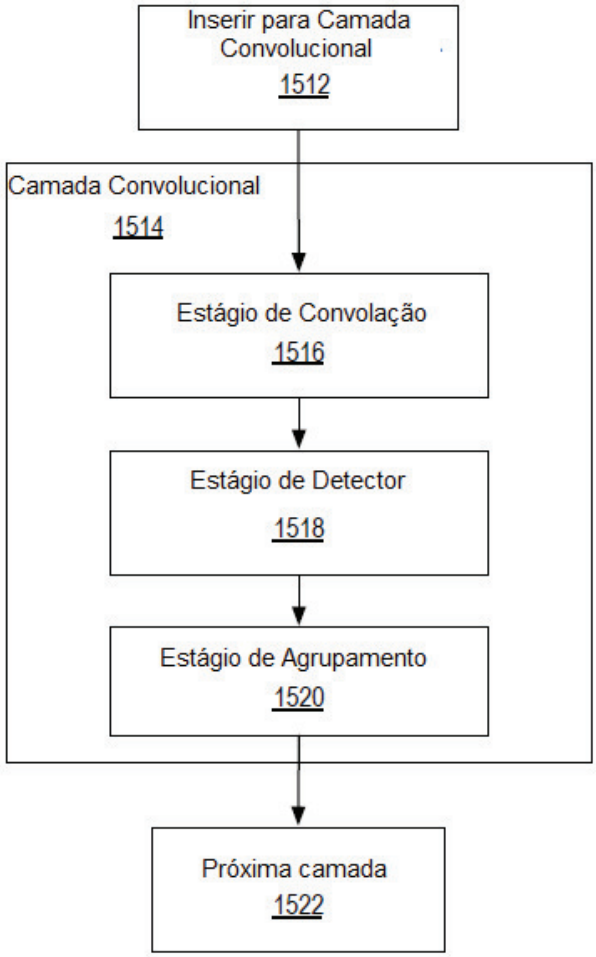


FIG. 15B

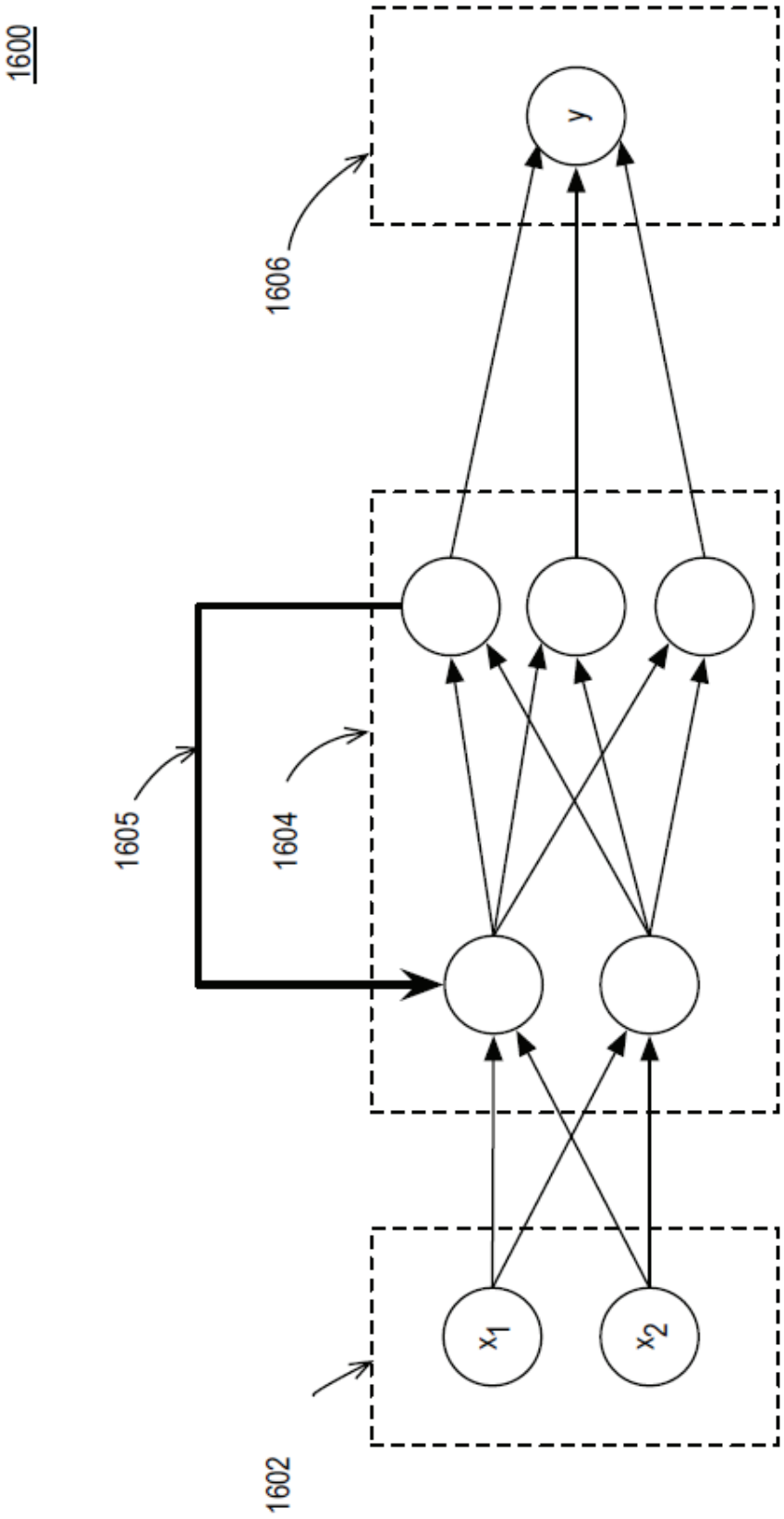


FIG. 16

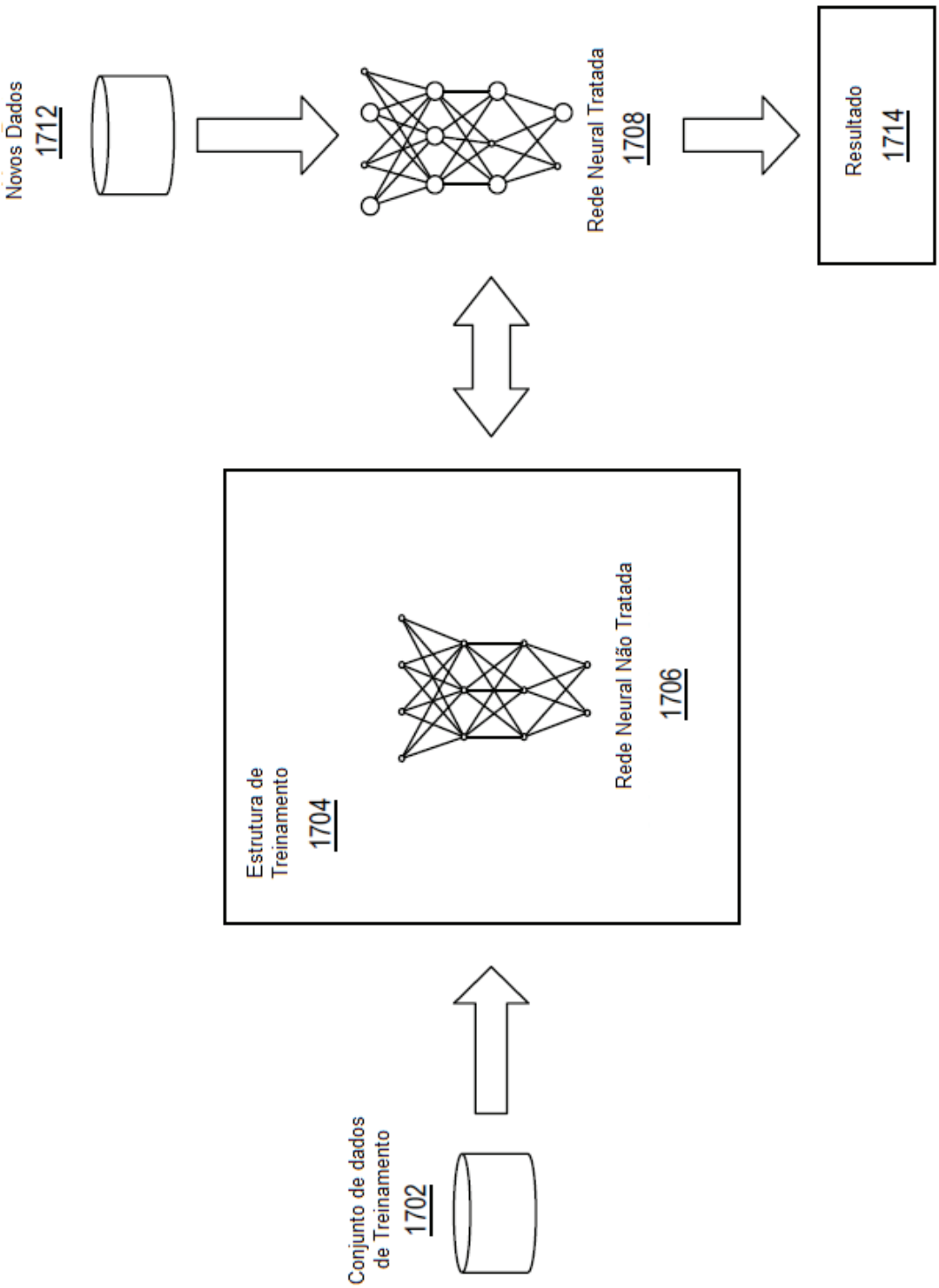

Novos Dados
1712
Rede Neural Tratada
1708
Resultado
1714
Estrutura de Treinamento
1704
Rede Neural Não Tratada
1706
Conjunto de dados de Treinamento
1702

FIG. 17

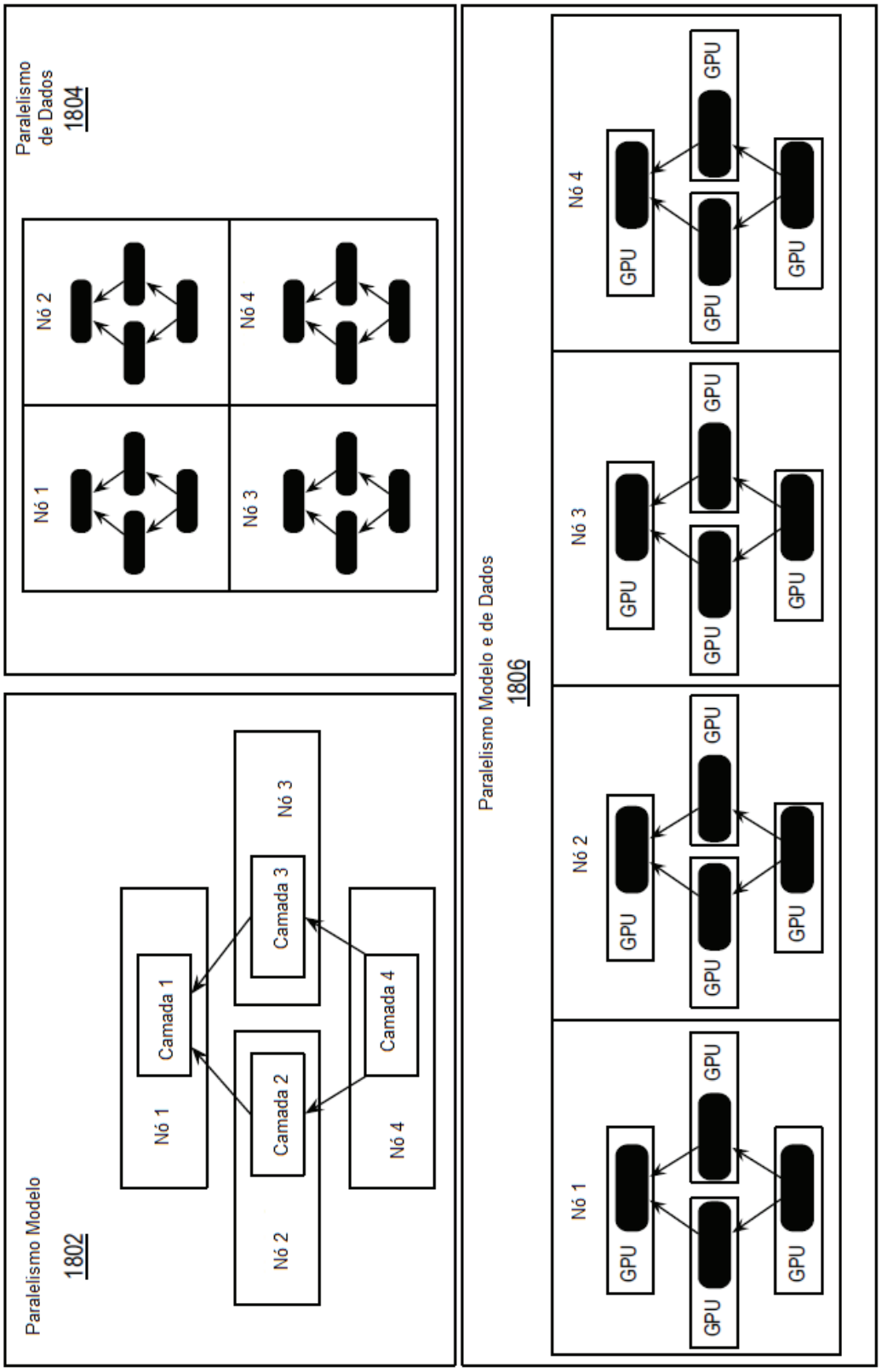

Paralelismo Modelo
1802
Nó 1
Camada 1
Nó 2
Camada 2
Nó 3
Camada 3
Nó 4
Camada 4
Paralelismo de Dados
1804
Nó 1
Nó 2
Nó 3
Nó 4
Paralelismo Modelo e de Dados
1806
Nó 1
Nó 2
Nó 3
Nó 4
GPU

FIG. 18

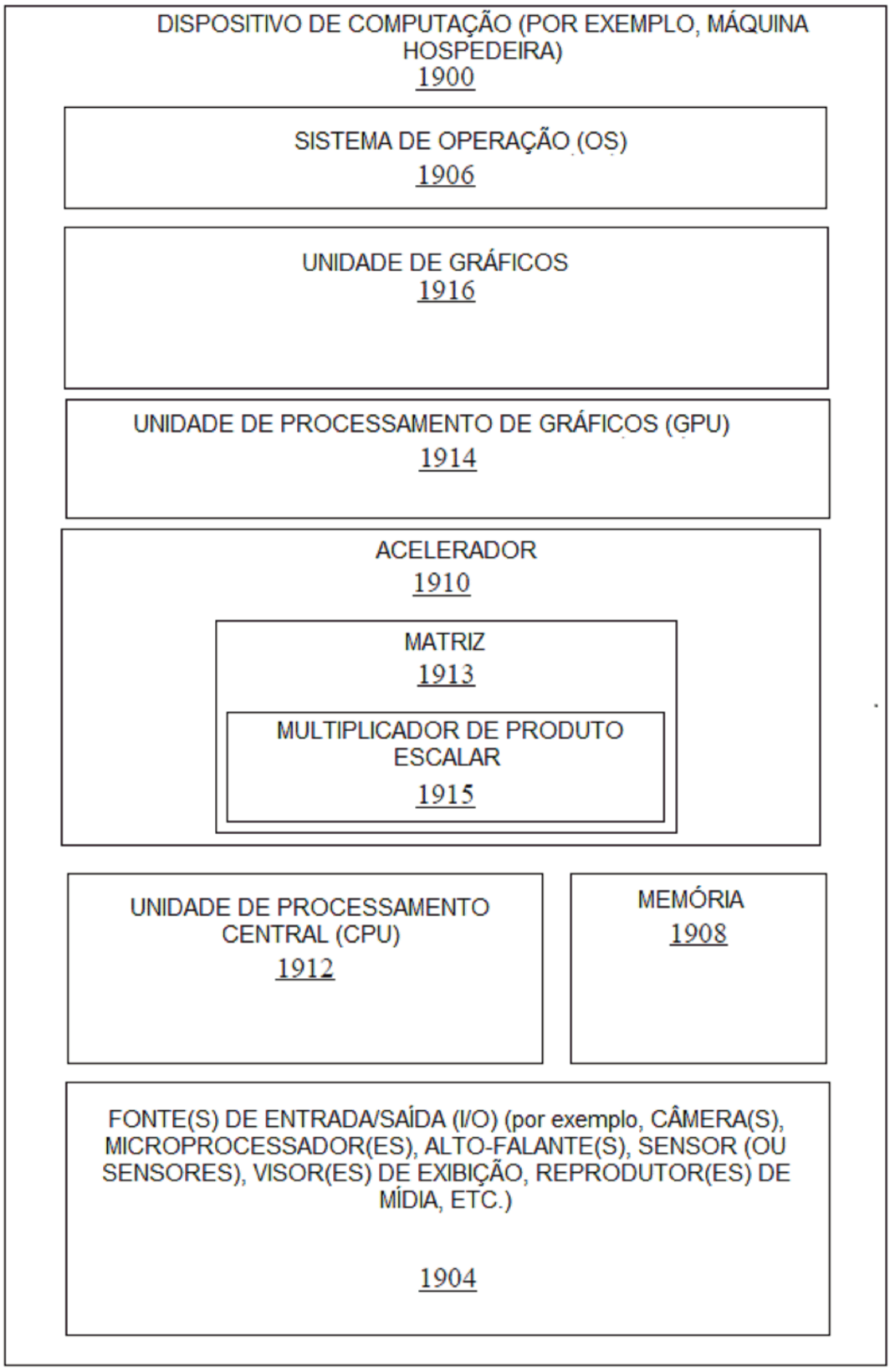

DISPOSITIVO DE COMPUTAÇÃO (POR EXEMPLO, MÁQUINA HOSPEDEIRA)
1900
SISTEMA DE OPERAÇÃO (OS)
1906
UNIDADE DE GRÁFICOS
1916
UNIDADE DE PROCESSAMENTO DE GRÁFICOS (GPU)
1914
ACELERADOR
1910
MATRIZ
1913
MULTIPLICADOR DE PRODUTO ESCALAR
1915
UNIDADE DE PROCESSAMENTO CENTRAL (CPU)
1912
MEMÓRIA
1908
FONTE(S) DE ENTRADA/SAÍDA (I/O) (por exemplo, CÂMERA(S), MICROPROCESSADOR(ES), ALTO-FALANTE(S), SENSOR (OU SENSORES), VISOR(ES) DE EXIBIÇÃO, REPRODUTOR(ES) DE MÍDIA, ETC.)
1904

FIG. 19

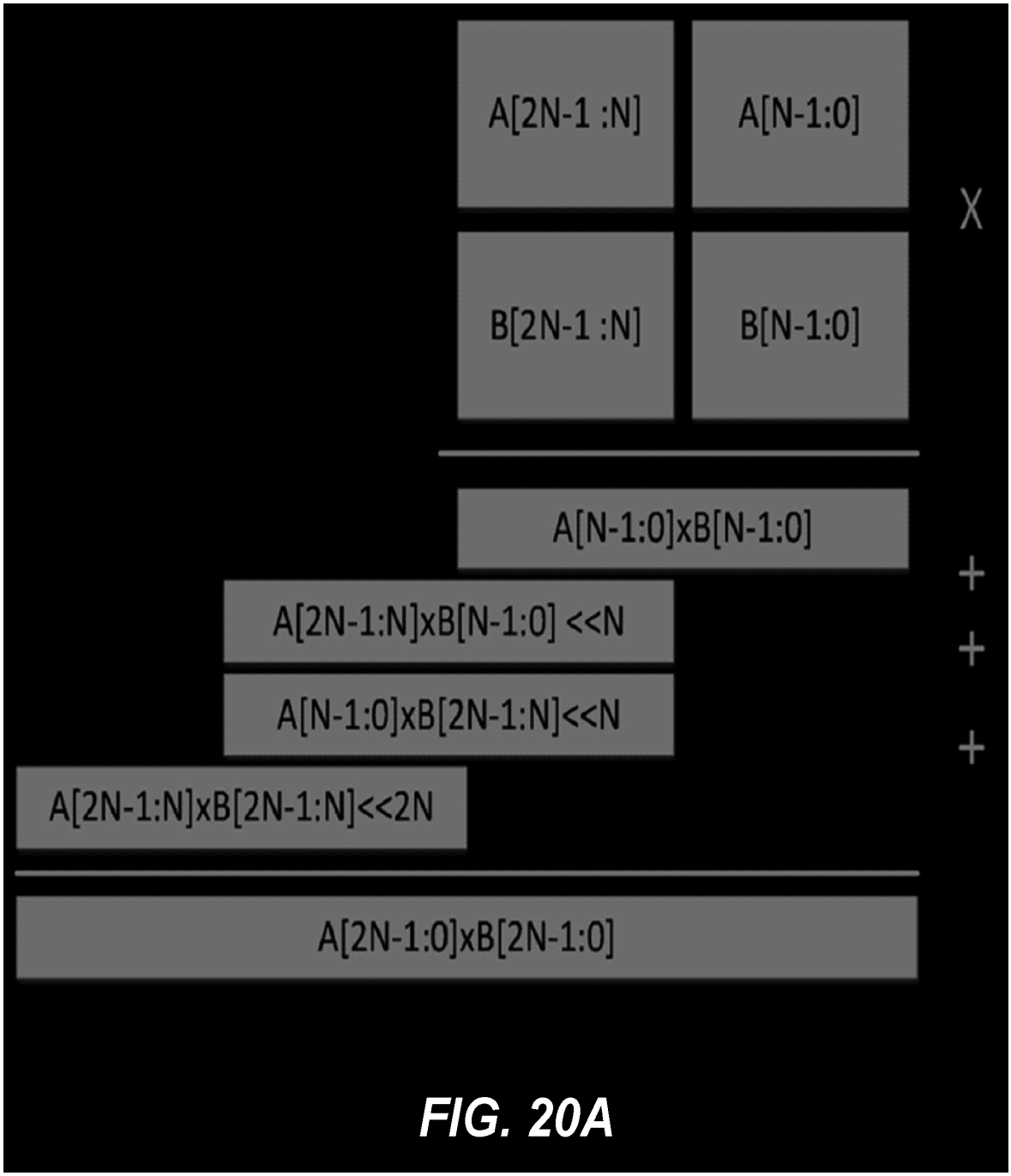

A[2N-1 :N]
A[N-1:0]
X
B[2N-1 :N]
B[N-1:0]
A[N-1:0]xB[N-1:0]
+
A[2N-1:N]xB[N-1:0] <<N
+
A[N-1:0]xB[2N-1:N]<<N
+
A[2N-1:N]xB[2N-1:N]<<2N
A[2N-1:0]xB[2N-1:0]

FIG. 20A

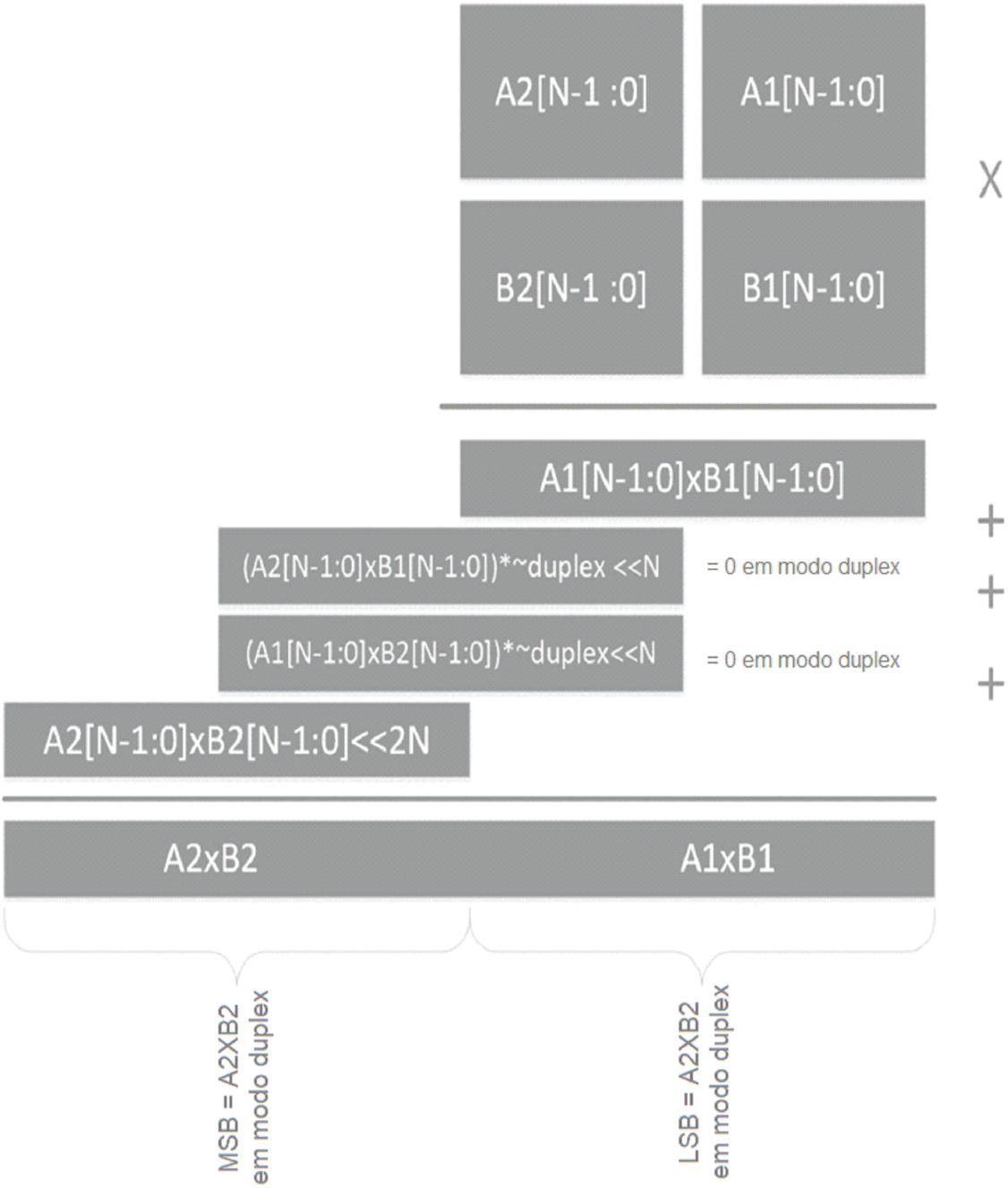

A2[N-1 :0]
A1[N-1:0]
X
B2[N-1 :0]
B1[N-1:0]
A1[N-1:0]xB1[N-1:0]
(A2[N-1:0]xB1[N-1:0])*~duplex <<N
= 0 em modo duplex
(A1[N-1:0]xB2[N-1:0])*~duplex<<N
= 0 em modo duplex
A2[N-1:0]xB2[N-1:0]<<2N
A2xB2
A1xB1
MSB = A2XB2 em modo duplex
LSB = A2XB2 em modo duplex

FIG. 20B

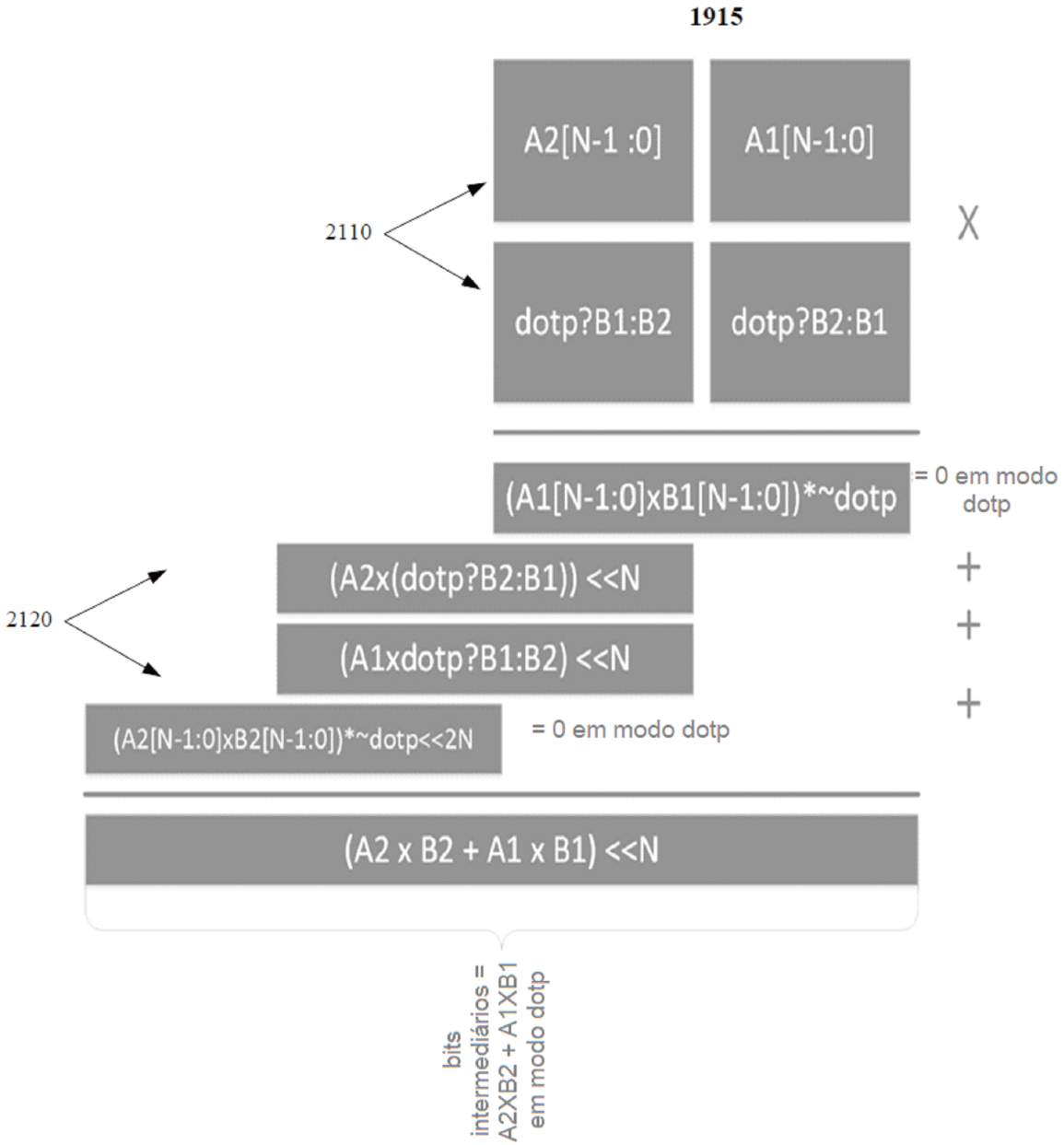

1915
A2[N-1 :0]
A1[N-1:0]
2110
X
dotp?B1:B2
dotp?B2:B1
(A1[N-1:0]xB1[N-1:0])*~dotp
= 0 em modo dotp
(A2x(dotp?B2:B1)) <<N
+
2120
(A1xdotp?B1:B2) <<N
+
(A2[N-1:0]xB2[N-1:0])*~dotp<<2N
= 0 em modo dotp
(A2 x B2 + A1 x B1) <<N
bits intermediários = A2XB2 + A1XB1 em modo dotp

FIG. 21

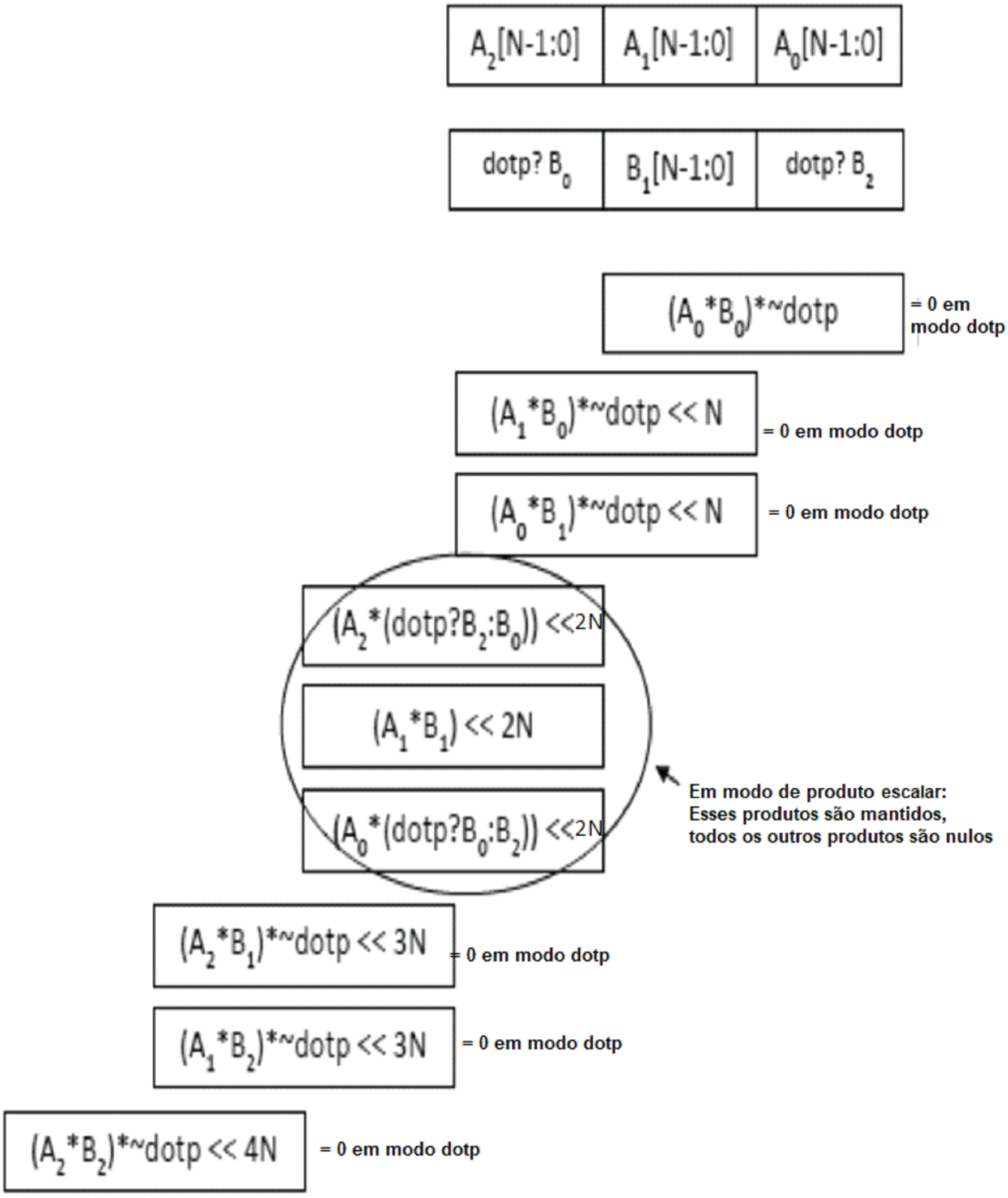

A2[N-1:0]
A1[N-1:0]
A0[N-1:0]
dotp? B0
B1[N-1:0]
dotp? B2
(A0*B0)*~dotp
= 0 em modo dotp
(A1*B0)*~dotp << N
= 0 em modo dotp
(A0*B1)*~dotp << N
= 0 em modo dotp
(A2*(dotp?B2:B0)) <<2N
(A1*B1) << 2N
(A0*(dotp?B0:B2)) <<2N
Em modo de produto escalar:
Esses produtos são mantidos,
todos os outros produtos são nulos
(A2*B1)*~dotp << 3N
= 0 em modo dotp
(A1*B2)*~dotp << 3N
= 0 em modo dotp
(A2*B2)*~dotp << 4N
= 0 em modo dotp

FIG. 22

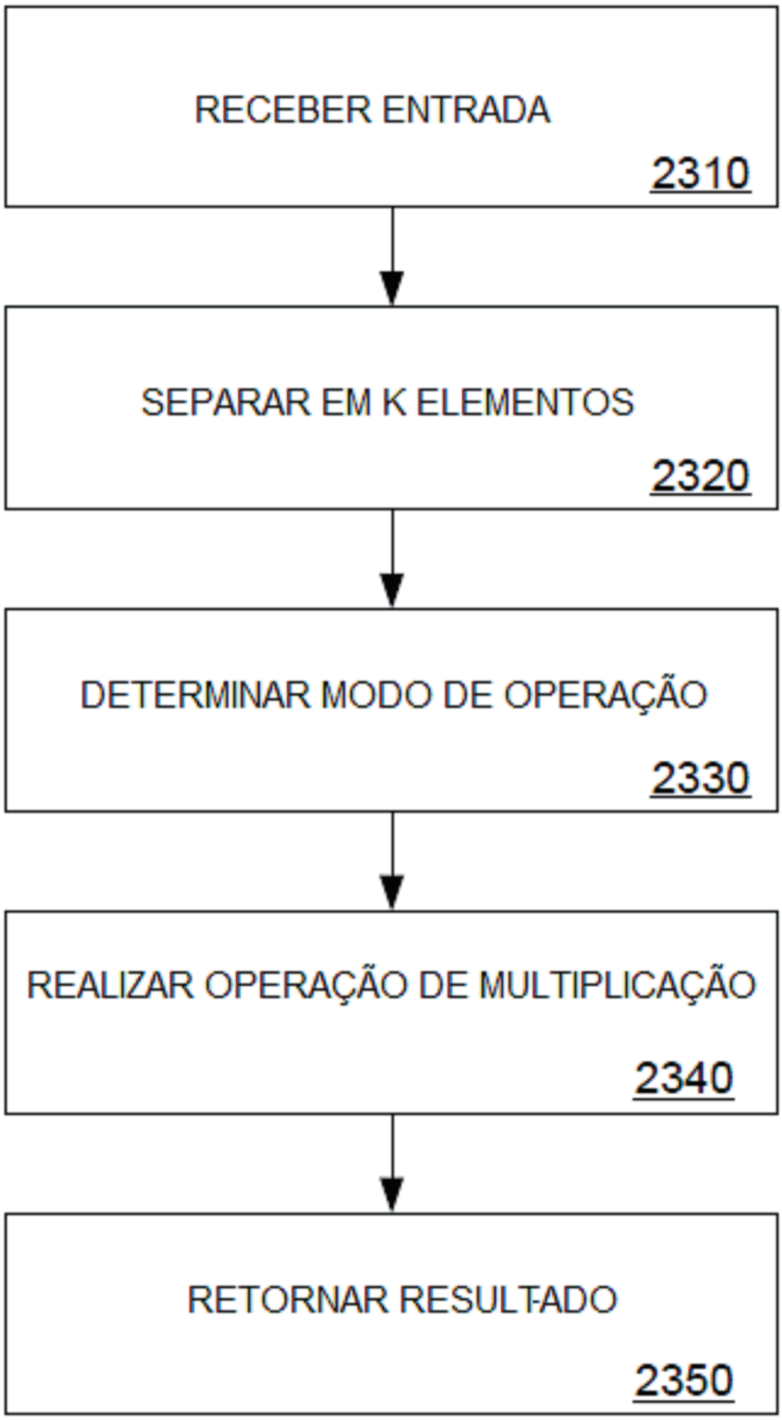

RECEBER ENTRADA
2310
SEPARAR EM K ELEMENTOS
2320
DETERMINAR MODO DE OPERAÇÃO
2330
REALIZAR OPERAÇÃO DE MULTIPLICAÇÃO
2340
RETORNAR RESULTADO
2350

FIG. 23

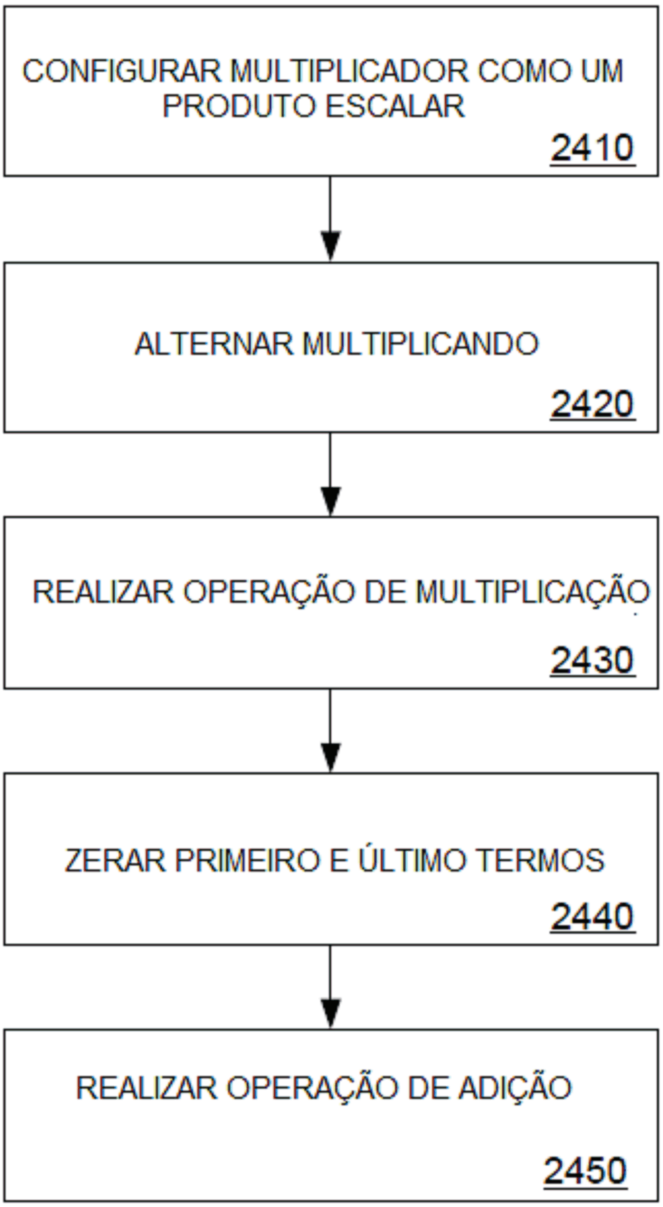

CONFIGURAR MULTIPLICADOR COMO UM PRODUTO ESCALAR
2410
ALTERNAR MULTIPLICANDO
2420
REALIZAR OPERAÇÃO DE MULTIPLICAÇÃO
2430
ZERAR PRIMEIRO E ÚLTIMO TERMOS
2440
REALIZAR OPERAÇÃO DE ADIÇÃO
2450

FIG. 24

RESUMO

"APARELHO E MÉTODO PARA FACILITAR OPERAÇÕES DE MULTIPLICAÇÃO DE MATRIZ; ACELERADOR DE HARDWARE"

A presente invenção refere-se a um aparelho para facilitar operações de multiplicação de matriz. O aparelho compreende hardware de multiplicação para operar em um modo de produto escalar, em que um estágio de multiplicação incluído no hardware de multiplicação é configurado como um produto escalar de diversos vetores de bit (N) para realizar NxN operações de multiplicação em uma pluralidade de multiplicandos e realizar operações de adição em resultados das NxN operações de multiplicação.