(19)日本国特許庁(JP)

(12)特許公報(B2)

(11)特許番号
特許第7364564号
(P7364564)

(45)発行日　令和5年10月18日(2023.10.18)

(24)登録日　令和5年10月10日(2023.10.10)

(51)国際特許分類		FI		
G06T 1/20 (2006.01)		G06T	1/20	C
G06F 9/38 (2018.01)		G06F	9/38	310J
		G06F	9/38	370C

請求項の数　15　（全20頁）

(21)出願番号	特願2020-528895(P2020-528895)
(86)(22)出願日	平成30年9月19日(2018.9.19)
(65)公表番号	特表2021-504828(P2021-504828A)
(43)公表日	令和3年2月15日(2021.2.15)
(86)国際出願番号	PCT/US2018/051735
(87)国際公開番号	WO2019/108292
(87)国際公開日	令和1年6月6日(2019.6.6)
審査請求日	令和3年9月16日(2021.9.16)
(31)優先権主張番号	15/828,059
(32)優先日	平成29年11月30日(2017.11.30)
(33)優先権主張国・地域又は機関	米国(US)

(73)特許権者　591016172
アドバンスト・マイクロ・ディバイシズ・インコーポレイテッド
ＡＤＶＡＮＣＥＤ ＭＩＣＲＯ ＤＥＶＩＣＥＳ ＩＮＣＯＲＰＯＲＡＴＥＤ
アメリカ合衆国 ９５０５４ カリフォルニア州、 サンタ クララ、オーガスティン ドライブ ２４８５

(74)代理人　100108833
弁理士　早川 裕司

(74)代理人　100111615
弁理士　佐野 良太

(74)代理人　100162156
弁理士　村雨 圭介

(72)発明者　アニルーダ アール． アチャリャ

最終頁に続く

(54)【発明の名称】　処理ユニットにおけるワークロードの正確な停止及び再開

(57)【特許請求の範囲】

【請求項１】

処理ユニットの複数のパイプラインのうち第１グループのパイプラインにおいて第１ワークロードを実行することと、

前記処理ユニットの前記複数のパイプラインのうち第２グループのパイプラインにおいて、前記第１ワークロードに依存する第２ワークロードを実行することと、

依存関係に基づいて前記第１ワークロード及び前記第２ワークロードを同時に停止することと、

前記第１ワークロード及び前記第２ワークロードを停止したことに応じて、前記第１ワークロード及び前記第２ワークロードの状態情報を第１メモリに記憶することと、を含む、

方法。

【請求項２】

前記第１ワークロードを実行することは、前記処理ユニットの計算パイプラインにおいて計算ワークロードを実行することを含み、

前記第２ワークロードを実行することは、前記処理ユニットのグラフィックスパイプラインにおいてグラフィックスワークロードを実行することを含む、

請求項１の方法。

【請求項３】

前記第１ワークロード及び前記第２ワークロードを実行することと同時に、前記処理ユニットの前記複数のパイプラインのうち第３グループのパイプラインにおいて第３ワーク

ロードを実行することをさらに含み、
前記第１グループ、前記第２グループ及び前記第３グループのパイプラインは、相互に排他的である、
請求項１又は２の方法。
【請求項４】
前記第１ワークロード及び前記第２ワークロードを停止することは、前記第３グループのパイプラインにおいて前記第３ワークロードの実行を継続している間、前記第１ワークロード及び前記第２ワークロードを停止することを含む、
請求項３の方法。
【請求項５】
前記第１ワークロード及び前記第２ワークロードを停止したことに応じて、前記第１グループ及び前記第２グループのパイプラインにおいて少なくとも１つの第４ワークロードを実行することをさらに含む、
請求項１～４の何れかの方法。
【請求項６】
前記少なくとも１つの第４ワークロードの完了又は停止に応じて、前記第１メモリに記憶された前記状態情報に基づいて前記第１グループ及び前記第２グループのパイプラインを構成することと、
前記第１グループ及び前記第２グループのパイプラインを構成した後に、前記第１ワークロード及び前記第２ワークロードの実行を前記第１グループ及び前記第２グループのパイプラインにおいてそれぞれ再開することと、をさらに含む、
請求項５の方法。
【請求項７】
前記第１ワークロード及び前記第２ワークロードの実行を再開することは、停止する前に前記第１ワークロード及び前記第２ワークロードを実行した計算ユニットのセットと同じ計算ユニットのセットにおいて前記第１ワークロード及び前記第２ワークロードの実行を再開することを含む、
請求項６の方法。
【請求項８】
前記処理ユニットの電源を切断したことに応じて、状態情報を前記第１メモリから第２メモリに移動させることであって、前記第２メモリは、前記処理ユニットの電源が切断されている間、前記状態情報を保持する、ことをさらに含む、
請求項１～７の何れかの方法。
【請求項９】
前記処理ユニットの電源を投入したことに応じて、状態情報を前記第２メモリから前記第１メモリに移動させることをさらに含む、
請求項８の方法。
【請求項１０】
複数のパイプラインを実装するように構成された複数の計算ユニットを含む処理ユニットを備える装置であって、
前記処理ユニットは、
前記複数のパイプラインのうち第１グループのパイプラインにおいて第１ワークロードを実行することと、
前記複数のパイプラインのうち第２グループのパイプラインにおいて、前記第１ワークロードに依存する第２ワークロードを実行することと、
依存関係に基づいて前記第１ワークロード及び前記第２ワークロードを同時に停止することと、
前記第１ワークロード及び前記第２ワークロードを停止したことに応じて、前記第１ワークロード及び前記第２ワークロードの状態情報を第１メモリに記憶することと、
を行うように構成されている、

装置。
【請求項１１】
前記複数のパイプラインのうち前記第１グループのパイプラインは、計算ワークロードを実行するように構成された計算パイプラインを含み、
前記複数のパイプラインのうち前記第２グループのパイプラインは、グラフィックスワークロードを実行するように構成されたグラフィックスパイプラインを含む、
請求項１０の装置。
【請求項１２】
前記処理ユニットは、
前記第１ワークロード及び前記第２ワークロードを実行することと同時に、前記複数のパイプラインのうち第３グループのパイプラインにおいて第３ワークロードを実行するように構成されており、
前記複数のパイプラインの前記第１グループ、前記第２グループ及び前記第３グループのパイプラインは、相互に排他的である、
請求項１０又は１１の装置。
【請求項１３】
前記処理ユニットは、前記第３グループのパイプラインにおいて前記第３ワークロードの実行を継続している間、前記第１ワークロード及び前記第２ワークロードを停止するように構成されている、
請求項１２の装置。
【請求項１４】
前記処理ユニットは、前記第１ワークロード及び前記第２ワークロードを停止したことに応じて、前記第１グループ及び前記第２グループのパイプラインにおいて少なくとも１つの第４ワークロードを実行するように構成されている、
請求項１０～１３の何れかの装置。
【請求項１５】
前記第１グループ及び前記第２グループのパイプラインは、前記少なくとも１つの第４ワークロードの完了又は停止に応じて、前記第１メモリに記憶された前記状態情報に基づいて構成され、
前記処理ユニットは、前記第１グループ及び前記第２グループのパイプラインを構成した後に、前記第１グループ及び前記第２グループのパイプラインにおいて前記第１ワークロード及び前記第２ワークロードの実行をそれぞれ再開する、
請求項１４の装置。
【発明の詳細な説明】
【背景技術】
【０００１】
グラフィックス処理ユニット（ＧＰＵ）等の処理ユニットは、通常、１つ以上の命令をパイプライン化したり、１つ以上のワークロードの命令を同時に実行したりすることの可能な複数のプロセッサコアを実装する。ＧＰＵのプロセッサコアは、一連のプログラム可能なシェーダと固定機能ハードウェアブロックとで構成されたグラフィックスパイプラインを用いて、三次元（３Ｄ）グラフィックスを処理する。例えば、フレーム内で可視であるオブジェクトの３Ｄモデルは、例えば、三角形、他のポリゴン又はパッチ等のプリミティブのセットによって表現され、これらは、ユーザに表示するためのピクセル値を生成するために、グラフィックスパイプラインにおいて処理される。ＧＰＵのパイプラインは、グラフィックスのワークロードに関連し得る計算ワークロード又は関連し得ない計算ワークロードを処理することも可能である。場合によっては、計算ワークロードとグラフィックスワークロードとの間に依存関係が存在する。例えば、ＧＰＵの１つのパイプラインで実行される計算ワークロードは、ＧＰＵの１つ以上の他のパイプラインで実行されるグラフィックスワークロードに対して、入力（例えば、レンダリングされる移動オブジェクトの位置等）を提供することができる。

【０００２】
添付の図面を参照することによって、本開示をより良く理解することができ、この多くの特徴及び利点が当業者に明らかになる。異なる図面において同じ符号を使用する場合、類似又は同一のアイテムを示す。
【図面の簡単な説明】
【０００３】
【図１】いくつかの実施形態による、ディスプレイへの出力を意図した視覚イメージを生成するためのグラフィックス処理ユニット（ＧＰＵ）を含む処理システムのブロック図である。
【図２】いくつかの実施形態による、三次元（３Ｄ）シーンのラスタライズ画像を生成するために高次ジオメトリプリミティブを処理することができるグラフィックスパイプラインを示す図である。
【図３】いくつかの実施形態による、異なるパイプライン上で実行される依存するワークロードの停止／再開動作をサポートする処理システムのブロック図である。
【図４】いくつかの実施形態による、パイプラインのサブセット上で実行される依存するワークロードの停止／再開動作をサポートする処理システムのブロック図である。
【図５】いくつかの実施形態による、ＧＰＵのパイプラインのサブセット上で実行される第１アプリケーションを停止して、当該パイプラインのサブセットを第２アプリケーションに利用させる方法のフロー図である。
【図６】いくつかの実施形態による、１つ以上のサスペンドメモリからパーシステントメモリに状態情報を移動させる方法のフロー図である。
【発明を実施するための形態】
【０００４】
通常、グラフィックス処理ユニット（ＧＰＵ）等の処理ユニットのパイプラインにおいて実行されるワークロードは、当該パイプラインで他の優先順位の高いワークロードを実行できるようにプリエンプトされる。処理ユニットのパイプラインで実行されるワークロードの作業状態は、ベクトル汎用レジスタ（ＶＧＰＲ）、ローカルデータシェア（ＬＤＳ）等のオンチップメモリに記憶される。処理ユニットのパイプラインで実行されているワークロードをプリエンプトして別のワークロードを実行させるには、プリエンプトしたワークロードの実行を開始する前に、現在実行されているワークロードのデータ（及び、対応する状態情報）をパイプラインから外部のオフチップメモリに保存する必要があり、これにより、新たなワークロードの処理に遅延が生じる。さらに、典型的なプリエンプションスキームでは、処理ユニットに実装された異なるパイプライン上で実行されるワークロード間に依存関係が存在するかどうかが考慮されない。したがって、パイプラインのプリエンプトは、プリエンプトされたパイプライン上で実行されているプロセスと、他のパイプライン上で実行されているプロセスとの間に依存関係が存在する場合、他のパイプライン上で実行されているプロセスに悪影響を及ぼす可能性がある。プリエンプション時のワークロードの相互依存性を考慮しないと、プロセスが再開されるときのパイプラインの状態にも影響を及ぼす。この場合、プロセスは、実行機能ユニットレベルにおいて全く同じ状態で再開されない場合がある。
【０００５】
図１～図６は、プリエンプションの前に処理ユニット内のパイプラインを空にすることを必要とせずに、正確な位置での異なるワークロード間の低遅延スイッチングをサポートする「ギャングプリエンプション（gang preemption）」技術を示す図である。いくつかの実施形態では、処理ユニット内のパイプラインの第１サブセットは、第１ワークロードを実行しており、処理ユニット内のパイプラインの第２サブセットは、第１ワークロードに依存してもよいし依存しなくてもよい第２ワークロードを実行している。例えば、第１ワークロードが計算ワークロードであり、第２ワークロードが計算ワークロードから入力を受信するグラフィックスワークロードである場合、第２ワークロードは第１ワークロードに依存する。いくつかの実施形態では、例えば、オペレーティングシステムがパイプ

ラインの第３サブセット上で実行されている場合、第３ワークロードが、第１ワークロード及び第２ワークロードと同時にパイプラインの第３サブセット上で実行されている。第１ワークロード及び第２ワークロードの実行が停止され、パイプラインの第１サブセット及び第２サブセットの状態情報が第１メモリに記憶される。第１ワークロード及び第２ワークロードがプリエンプトされると、パイプラインの第１サブセット及び第２サブセットは、他のワークロードを実行するために利用可能になる。いくつかの実施形態では、第４ワークロードが、パイプラインの第１サブセット及び第２サブセットを用いて、場合によっては他のワークロードと組み合わせて実行される。第３ワークロードは、第４ワークロード及び他のワークロード（存在する場合）と同時に実行され続ける。

【０００６】

第４ワークロード及び他のワークロード（存在する場合）の完了又は停止に応じて、第１ワークロード及び第２ワークロードの状態情報が第１メモリから読み出され、パイプラインの第１サブセット及び第２サブセットの各々を構成するために用いられる。次に、第１ワークロード及び第２ワークロードの実行は、パイプラインの第１サブセット及び第２サブセット上で、停止前に存在したときと全く同じ状態で再開される。例えば、単一命令複数データ（ＳＩＭＤ）プロセッサコアのセットで実行されているシェーダの複数のインスタンスは、プリエンプトされる前にシェーダを実行していた同じＳＩＭＤプロセッサコアのセット上で、同じ状態で再開される。プリエンプトされたワークロードの実行を、停止前と全く同じ状態から再開することは、最終結果を生成するために複数のパイプライン上で実行されている相互依存的なワークロード（例えば、ゲームアプリケーションによって生成される計算ワークロード及びグラフィックスワークロード）にとって特に重要である。いくつかの実施形態では、第１メモリに記憶された状態情報は、処理ユニットの電源を切断したことに応じて、パーシステントストレージ（persistent storage）に書き込まれる。次に、状態情報は、処理ユニットの電源を投入したことに応じて、パーシステントストレージから第１メモリに書き込まれる。

【０００７】

図１は、いくつかの実施形態による、ディスプレイ１１０への出力を意図した視覚イメージを生成するためのグラフィックス処理ユニット（ＧＰＵ）１０５を含む処理システム１００のブロック図である。ＧＰＵ１０５は、複数の計算ユニット１１１，１１２，１１３，１１４を含むマルチスレッドプロセッサである。これらの計算ユニットは、本明細書ではまとめて「計算ユニット１１１～１１４」と呼ばれる。計算ユニット１１１～１１４は、命令を同時又は並列に実行するように構成されている。計算ユニット１１１～１１４のいくつかの実施形態は、ビデオレンダリングに必要とされる基本的な数値演算等のように比較的小さい演算セットを実行するように構成されている。分かりやすくするために、図１には４つの計算ユニット１１１～１１４が示されているが、ＧＰＵ１０５のいくつかの実施形態は、数十、数百又は数千の計算ユニットを含む。計算ユニット１１１～１１４のいくつかの実施形態は、例えば、単一命令複数データ（ＳＩＭＤ）のウェーブとして、単一のプログラムの複数のインスタンス（又は、ウェーブ）を複数のデータセットに対して同時に実行する。例えば、計算ユニット１１１～１１４は、データのアレイ又はストリームに対して同じ演算シーケンスを実行するように構成されている。

【０００８】

処理システム１００は、メモリ１１５を含む。メモリ１１５のいくつかの実施形態は、ダイナミックランダムアクセスメモリ（ＤＲＡＭ）として実装される。しかしながら、メモリ１１５は、スタティックランダムアクセスメモリ（ＳＲＡＭ）、不揮発性ＲＡＭ等を含む他のタイプのメモリを用いて実装されてもよい。図示した実施形態では、ＧＰＵ１０５は、バス１２０を介してメモリ１１５と通信する。しかしながら、ＧＰＵ１０５のいくつかの実施形態は、直接接続を介して、又は、他のバス、ブリッジ、スイッチ、ルータ等を介してメモリ１１５と通信する。ＧＰＵ１０５は、メモリ１１５に記憶された命令を実行し、実行された命令の結果等の情報をメモリ１１５に記憶する。例えば、メモリ１１５は、ＧＰＵ１０５内の計算ユニット１１１～１１４によって実行されるプログラムコード

からの命令のコピー１２５を記憶する。

【０００９】

処理システム１００は、命令を実行するための中央処理装置（ＣＰＵ）１３０を含む。ＣＰＵ１３０のいくつかの実施形態は、逐次シリアル処理のために最適化されたマルチプロセッサコア１３１，１３２，１３３，１３４（本明細書ではまとめて「プロセッサコア１３１～１３４」と呼ばれる）を含む。プロセッサコア１３１～１３４は、（例えば、計算ユニット１１１～１１４によって実装された小さい命令セットと比べて）比較的大きい命令セットと、処理システム１００で実行されているアプリケーションによって必要とされる計算の全てレパートリーをＣＰＵ１３０が実行するのを可能にする大きい機能セット（large feature sets）と、を実装する。ＣＰＵ１３０のいくつかの実施形態は、プロセッサコア１３１～１３４を用いて複数のパイプラインを実装することができる。ＣＰＵ１３０は、バス１２０にも接続されており、バス１２０を介してＧＰＵ１０５及びメモリ１１５と通信する。ＣＰＵ１３０は、メモリ１１５に記憶されたプログラムコード１３５等の命令を実行し、ＣＰＵ１３０は、実行された命令の結果等の情報をメモリ１１５に記憶する。ＣＰＵ１３０は、ドローコールをＧＰＵ１０５に発行することによって、グラフィックス処理を開始することができる。ドローコールは、ＣＰＵ１３０によって生成され、フレーム内のオブジェクト（又は、オブジェクトの一部）のレンダリングをＧＰＵ１０５に指示するために、ＧＰＵ１０５に送信されるコマンドである。ＧＰＵ１０５は、ドローコールに応じて、オブジェクトをレンダリングして、ディスプレイ１１０に提供されるピクセル値を生成する。ディスプレイは、ピクセル値を用いて、レンダリングされたオブジェクトを表す画像を表示する。

【００１０】

入出力（Ｉ／Ｏ）エンジン１４０は、キーボード、マウス、プリンタ、外部ディスク等の処理システム１００の他の要素と同様に、ディスプレイ１１０に関連する入出力動作を処理する。Ｉ／Ｏエンジン１４０は、Ｉ／ＯエンジンＩ／Ｏエンジン１４０がＧＰＵ１０５、メモリ１１５又はＣＰＵ１３０と通信できるようにバス１２０に接続されている。図示した実施形態では、Ｉ／Ｏエンジン１４０は、コンパクトディスク（ＣＤ）、デジタル多用途ディスク（ＤＶＤ）、ネットワークサーバ等の外部記憶媒体１４５に記憶された情報を読み込むように構成されている。外部記憶媒体１４５は、ビデオゲーム等のアプリケーションを実現するために用いられるプログラムコードを表す情報を記憶する。外部記憶媒体１４５上のプログラムコードは、メモリ１１５に書き込まれ、ＧＰＵ１０５によって実行される命令のコピー１２５、又は、ＣＰＵ１３０によって実行されるプログラムコード１３５を形成する。

【００１１】

マルチスレッドＧＰＵ１０５内の計算ユニット１１１～１１４は、ＧＰＵ１０５におけるウェーブの実行をサポートするために用いられるリソースを共有する。ＧＰＵ１０５のいくつかの実施形態は、計算ユニット１１１～１１４上で実行されているウェーブの状態情報を記憶するベクトル汎用レジスタ（ＶＧＰＲ、分かりやすくするために図１に示されていない）のセットを実装する。ＶＧＰＲは、計算ユニット１１１～１１４上で同時に実行されているウェーブ間で共有される。例えば、各ウェーブは、ウェーブの状態情報を記憶するために、ＶＧＰＲのサブセットに割り当てられる。ウェーブは、同時に実行されているウェーブ間で分割されるローカルデータシェアや、ローカルキャッシュにアクセスするためにウェーブによって共有されるメモリ帯域幅等を含むＧＰＵ１０５の他のリソースを共有する。マルチスレッドＣＰＵ１３０内のプロセッサコア１３１～１３４もリソースを共有する。

【００１２】

ＧＰＵ１０５は、複数のワークロードを同時又は並列に実行するための複数のパイプラインを実装するように構成されている。計算ユニット１１１～１１４のいくつかの実施形態は、グラフィックスワークロードを実行して、オブジェクトの画像をディスプレイ１１０上で表現できるようにレンダリングするグラフィックスパイプラインを実装するために

用いられる。計算ユニット１１１～１１４のいくつかの実施形態は、計算ワークロードを実行するための計算パイプラインを実装するためにも用いられる。例えば、計算ユニット１１１～１１４の第１サブセットは、１つ以上のグラフィックスパイプラインを実装し、計算ユニット１１１～１１４の第２サブセットは、１つ以上の計算パイプラインを実装する。ＣＰＵ１３０のプロセッサコア１３１～１３４のいくつかの実施形態も、複数のパイプラインを実装しており、これにより、ＧＰＵ１０５と同様の方法で動作するように構成されている。したがって、分かりやすくするために、以下の説明は、同じ技術がＣＰＵ１３０に適用可能であることを理解した上で、ＧＰＵ１０５に関連して提示される。

【００１３】

ＧＰＵ１０５の異なるパイプライン上で実行されるワークロードは、場合によっては互いに依存する。例えば、計算ワークロードは、グラフィックスワークロードによってレンダリングされるシーンのモデルの計算を実行するために用いられる。計算ワークロードは、シーン内の移動オブジェクトの位置の計算等の計算を実行する。グラフィックスワークロードは、移動オブジェクトを含むシーンの一部を表す画像をレンダリングする。グラフィックスワークロードは、計算ワークロードが、シーンをレンダリングするのに必要なグラフィックスワークロードに対する入力（例えば、移動オブジェクトの位置及び方向等）を提供するので、計算ワークロードに依存する。したがって、計算パイプライン内の計算ワークロードによって生成されたデータは、計算パイプラインから１つ以上のグラフィックスパイプラインに転送され、グラフィックスパイプラインで実行されているグラフィックスワークロードで利用可能になる。

【００１４】

ＧＰＵ１０５を動作させるために実装されたソフトウェア又はドライバ１５０は、例えば、ワークロードを生成しているアプリケーションによって提供された情報を用いて、ワークロード間の依存関係を明示的又は暗黙的に識別することができる。ＧＰＵ１０５内のソフトウェア又はハードウェアによってスケジューリングされたワークロードであって、計算ユニット１１１～１１４を用いて実装された異なるパイプライン上で実行されるワークロード間に依存関係が存在する場合、計算ユニット１１１～１１４を用いて実装された他のパイプライン上で実行されている他のワークロードを停止することなく、依存するワークロードが選択的に停止される。いくつかの実施形態では、第１ワークロードは、ＧＰＵ１０５のパイプラインの第１サブセットで実行されている。第２ワークロードは、ＧＰＵ１０５のパイプラインの第２サブセットで実行されており、第２ワークロードは、第１ワークロードに依存する。ＧＰＵ１０５は、第１ワークロード及び第２ワークロードを停止する。また、ＧＰＵ１０５は、第１ワークロード及び第２ワークロードを停止したことに応じて、第１ワークロード及び第２ワークロードの状態情報１５５を第１メモリに記憶する。例えば、ＧＰＵ１０５は、状態情報１５５をメモリ１１５に記憶する。ワークロードの状態情報の例としては、ドロー状態情報、ディスパッチ状態情報、ＳＩＭＤ識別子、シェーダエンジン識別子、メモリ管理情報、リソース情報等が挙げられる。

【００１５】

停止したワークロードを再開するために、状態情報１５５を用いて、第１パイプライン及び第２パイプラインを、第１ワークロード及び第２ワークロードの停止前の状態に構成する。次に、第１ワークロード及び第２ワークロードの実行が再開される。依存するワークロードは、依存するワークロードが停止したときにこれらが実行されていたのと同じ計算ユニット１１１～１１４上で再開される。その結果、依存するワークロードは、停止前に存在した正確な状態で実行を再開する。

【００１６】

図２は、いくつかの実施形態による、三次元（３Ｄ）シーンのラスタライズ画像を生成するために高次ジオメトリプリミティブを処理することができるグラフィックスパイプライン２００を示す図である。グラフィックスパイプライン２００は、図１に示すＧＰＵ１０５のいくつかの実施形態において実装される。例えば、いくつかの実施形態では、グラフィックスパイプライン２００は、図１に示すＧＰＵ１０５内の計算ユニット１１１～１

１４を用いて実装される。
【００１７】
グラフィックスパイプライン２００は、シーンのモデルの一部を表すオブジェクトを定義するために用いられるストレージリソース２０１からの情報にアクセスするように構成された入力アセンブラ２０２を含む。頂点シェーダ２０３は、ソフトウェアで実装することができ、プリミティブの単一の頂点を入力として論理的に受信し、単一の頂点を出力する。頂点シェーダ２０３等のシェーダのいくつかの実施形態は、単一命令複数データ（ＳＩＭＤ）処理を実行し、これにより、複数の頂点が、例えば図１に示す計算ユニット１１１～１１４によって同時に処理される。図２に示すグラフィックスパイプライン２００は、ユニファイドシェーダモデル（unified shader model）を実装し、これにより、グラフィックスパイプライン２００に含まれる全てのシェーダが、共有されたＳＩＭＤ計算ユニット上で同じ実行プラットフォームを有する。したがって、頂点シェーダ２０３を含むシェーダは、本明細書においてユニファイドシェーダプール２０４と呼ばれる共通のリソースのセットを用いて実装される。ユニファイドシェーダプール２０４のいくつかの実施形態は、図１に示すＧＰＵ１０５に実装された計算ユニット１１１～１１４を用いて実装される。
【００１８】
ハルシェーダ２０５は、入力パッチを定義するのに使用される入力高次パッチ又はコントロールポイントに基づいて動作する。ハルシェーダ２０５は、テッセレーション係数及び他のパッチデータを出力する。ハルシェーダ２０５によって生成されたプリミティブは、オプションでテッセレータ２０６に提供されてもよい。テッセレータ２０６は、ハルシェーダ２０５からオブジェクト（パッチ等）を受信し、例えば、ハルシェーダ２０５によってテッセレータ１０６に提供されるテッセレーション係数に基づいて入力オブジェクトをテッセレーションすることによって、入力オブジェクトに対応するプリミティブを識別する情報を生成する。テッセレーションは、例えば、テッセレーション処理によって生成されるプリミティブの粒度を指定するテッセレーション係数によって示すように、例えばパッチ等の入力高次プリミティブを、より詳細なレベルを表すより低次出力プリミティブのセットに分割する。したがって、シーンのモデルを、（メモリ又は帯域幅を節約するために）少数の高次プリミティブで表すことができ、高次プリミティブをテッセレーションすることによってさらなる詳細を追加することができる。
【００１９】
ドメインシェーダ２０７は、ドメイン位置及び（オプションで）他のパッチデータを入力する。ドメインシェーダ２０７は、提供された情報に基づいて動作し、入力ドメイン位置及び他の情報に基づいて、出力用の単一の頂点を生成する。ジオメトリシェーダ２０８は、入力プリミティブを受信し、入力プリミティブに基づいて、ジオメトリシェーダ２０８によって生成される最大４つのプリミティブを出力する。プリミティブの１つのストリームがラスタライザ２０９に提供され、最大４つのプリミティブのストリームがストレージリソース２０１のバッファに連結され得る。ラスタライザ２０９は、シェーディング操作と、他の操作（例えば、クリッピング、パースペクティブ分割、シザリング、ビューポート選択等）と、を実行する。ピクセルシェーダ２１０は、ピクセルフローを入力し、入力されたピクセルフローに応じて、０又は別のピクセルフローを出力する。出力マージャブロック２１１は、ピクセルシェーダ２１０から受信したピクセルに対して、ブレンド、デプス（depth）、ステンシル又は他の操作を実行する。
【００２０】
グラフィックスパイプライン２００のステージは、ユニファイドシェーダプール２０４内の処理リソースを用いて、異なるステージによって実行されるウェーブによって共有されたストレージリソース２１５にアクセスすることができる。ストレージリソース２１５の一部は、図１に示すＧＰＵ１０５の一部としてオンチップで実装されるか、図１に示すメモリ１１５のいくつかの実施形態を用いてオフチップで実装される。図２には、単一のグラフィックスパイプライン２００が示されているが、ストレージリソース２１５（及び

、ユニファイドシェーダプール２０４）のいくつかの実施形態は、複数のグラフィックスパイプラインによって共有される。

【００２１】

ストレージリソース２１５は、複数のウェーブのワークグループ内での読み出し／書き込み通信及び同期化のために用いられるＬＤＳ２２０を含む。ストレージリソース２１５は、ウェーブの現在の状態を定義する状態情報（例えば、ウェーブによって実行された動作の中間結果等）を記憶するＶＧＰＲ２２５も含む。ストレージリソース２１５は、頂点データ、テクスチャデータ、及び、グラフィックスパイプライン２００の１つ以上のステージによって頻繁に用いられる他のデータ等の情報をキャッシュするのに用いられるキャッシュ階層２３０をさらに含む。ストレージリソース２１５は、１つ以上のアプリケーション（例えば、１つ以上のゲーム等）に関連する停止されたワークロードの状態情報を記憶するサスペンドメモリ２３５をさらに含む。いくつかの実施形態では、ストレージリソース２１５は、他のレジスタ、バッファ、メモリ又はキャッシュも含む。グラフィックスパイプライン２００の共有リソースは、グラフィックスパイプライン２００のステージとストレージリソース２１５との間の通信をサポートするために用いられるメモリファブリック内の帯域幅も含む。

【００２２】

図３は、いくつかの実施形態による、種々のパイプライン上で実行される依存するワークロードの停止／再開動作をサポートする処理システム３００のブロック図である。処理システム３００は、図１に示す処理システム１００及び図２に示すグラフィックスパイプライン２００のいくつかの実施形態の一部を表す。

【００２３】

処理システム３００は、アプリケーション３０１，３０２，３０３を実行するように構成されている。これらのアプリケーションは、本明細書ではまとめて「アプリケーション３０１～３０３」と呼ばれる。アプリケーション３０１～３０３のいくつかの実施形態は、同時又は並列に実行される計算ワークロード及びグラフィックスワークロードを生成する。例えば、いくつかの実施形態では、アプリケーション３０１～３０３のうちの１つ以上は、物理エンジン等の計算ワークロードを利用してシーン内のオブジェクトの位置を決定し、グラフィックスワークロードを利用して、ユーザに表示されるオブジェクトをレンダリングするゲームアプリケーションである。したがって、コンピュータワークロードとグラフィックスワークロードとは、相互に依存している。例えば、いくつかの実施形態では、グラフィックスワークロードは、グラフィックスワークロードによってレンダリングされるオブジェクトの位置及び方向を示す入力を計算ワークロードから受信する。

【００２４】

処理システム３００は、処理システム３００のハードウェアリソース及びソフトウェアリソースを管理し、共通のサービスをアプリケーション３０１～３０３に提供するオペレーティングシステム（ＯＳ）３０５も実施する。アプリケーション３０１～３０３によって生成されたワークロードは、スケジューリングされ、ＯＳ３０５又はスケジューリングハードウェア／ソフトウェアを介してＧＰＵ３１０に提供される。例えば、アプリケーション３０１は、計算ワークロードに関連する命令の第１ストリームと、グラフィックスワークロードに関連する命令の第２ストリームと、を生成する。アプリケーション３０１によって生成された命令は、ＧＰＵ３１０の複数のパイプラインにおいて同時又は並列に実行するために、ＧＰＵ３１０に提供される。分かりやすくするために、複数のパイプラインは、図３に示されていない。ＧＰＵ３１０のパイプラインの１つのサブセットは、計算ワークロード及びグラフィックスワークロードを実行するために用いられ、ＧＰＵ３１０のパイプラインの別のサブセットは、ＯＳ３０５を実行するために用いられる。

【００２５】

デバイスドライバは、処理システム３００のハードウェアリソースへのアクセスをアプリケーション３０１～３０３及びＯＳ３０５に提供するために用いられる。図示した実施形態では、アプリケーション３０１～３０３は、対応するユーザモードドライバ（ＵＭＤ

）３１１，３１２，３１３に関連付けられている。これらのユーザモードドライバは、本明細書ではまとめて「ＵＭＤ３１１～３１３」と呼ばれる。ＵＭＤ３１１～３１３は、対応するアプリケーション３０１～３０３のコンパイル又は実行に応じて、対応するアプリケーション３０１～３０３にアタッチ（attach）し、通常、グラフィックス又は計算アプリケーションプログラミングインタフェース（ＡＰＩ）を実装する。ＯＳ３０５は、ＯＳ３０５のカーネルと密接に作用するカーネルモードドライバ（ＫＭＤ）３１５とインタフェースし、場合によっては、スケジューリング、電力管理等を含むがこれらに限定されない低レベルハードウェア機能へのアクセスを提供する。

【００２６】

処理システム３００内のメモリの一部３２０は、停止したアプリケーションの状態情報を記憶するために用いられる。一部３２０は、図１に示すメモリ１１５及び図２に示すストレージリソース２１５のいくつかの実施形態において実装される。図示した実施形態では、一部３２０は、サスペンドメモリ３２１，３２２，３２３を含む。これらのサスペンドメモリは、本明細書ではまとめて「サスペンドメモリ３２１～３２３」と呼ばれる。サスペンドメモリ３２１～３２３の各々は、アプリケーション３０１～３０３のうち対応するアプリケーションに関連付けられており、対応するアプリケーション３０１～３０３の状態情報を記憶するために用いられる。サスペンドメモリ３２１～３２３は、アプリケーション３０１～３０３が処理システム３００において実行を開始したことに応じて、アプリケーション３０１～３０３を停止したことに応じて、又は、処理システム３００における他の条件若しくはイベントに応じて、対応するアプリケーション３０１～３０３に割り当てられる。

【００２７】

対応するアプリケーション３０１～３０３の実行が再開されたことに応じて、状態情報がサスペンドメモリ３２１～３２３から読み出される。例えば、アプリケーション３０１に関する計算ワークロード及びグラフィックスワークロードを実行する複数のパイプラインを構成するために用いられる状態情報は、ＧＰＵ３１０上のアプリケーション３０１の実行が停止したことに応じて、サスペンドメモリ３２１に記憶される。アプリケーション３０２は、アプリケーション３０１の停止によって利用可能になったＧＰＵ３１０のパイプライン上で実行を開始する。アプリケーション３０２の完了又は停止に応じて、サスペンドメモリ３２１に記憶された状態情報がＧＰＵ３１０に提供される。このＧＰＵは、この状態情報を用いて、アプリケーション３０１に関する計算ワークロード又はグラフィックスワークロードを以前に実行していた同じ計算ユニット又は固定機能ハードウェア上のグラフィックスパイプラインを構成する。次に、ＧＰＵ３１０は、アプリケーション３０１が停止された状態と同じ状態からアプリケーション３０１の実行を再開する。

【００２８】

サスペンドメモリ３２１～３２３をアプリケーション３０１～３０３に別々に割り当てることによって、アプリケーション３０１～３０３を任意の順序で停止又は再開することができる。例えば、アプリケーション３０１が停止した場合、対応するサスペンドメモリ３２２，３２３に記憶された状態情報に基づいて、アプリケーション３０２，３０３（又は、一部３２０内の対応するサスペンドメモリに記憶された状態情報を有する他のアプリケーション）のうち何れかを再開することができる。したがって、サスペンドメモリ３２１～３２３は、メモリスタックのように動作しない。さらに、アプリケーション３０１～３０３間のユーザモードタスクのスイッチングは、対応するアプリケーション３０１～３０３の介入なしに実行時に行われる。その代わりに、ＯＳ３０５は、アプリケーション３０１～３０３の停止又は再開を行う。さらに、ＯＳ３０５によって停止可能なアプリケーション３０１～３０３の数は、サスペンドメモリ３２１～３２３を割り当てるのに利用可能なメモリの一部３２０のサイズによってのみ制限される。

【００２９】

処理システム３００のいくつかの実施形態は、パーシステントメモリ３２５を含む。このパーシステントメモリは、例えば、サスペンドメモリ３２１～３２３がＤＲＡＭ等の揮

発性メモリ素子を用いて実装されている場合等のように、割り当てられたサスペンドメモリ３２１～３２３が電源切断時に情報を保持しない場合に、処理システム３００の他の部分に情報を保持することができる。例えば、パーシステントメモリ３２５は、メモリの一部３２０から電力が除去された場合に情報を保持し、サスペンドメモリ３２１～３２３に供給される電圧は、一部３２０を実装するのに使用されるメモリ素子内にデータを保持するのに不十分である。パーシステントメモリ３２５の実施例としては、ハードディスクなどの磁気記憶装置、ソリッドステートドライブ（ＳＳＤ）等のソリッドステート記憶装置、不揮発性ランダムアクセスメモリ（ＮＶＲＡＭ）、処理システム３００内の他の素子とは異なる電源メッシュに取り付けられたメモリ素子、又は、電力若しくは電圧の喪失若しくは低下によってサスペンドメモリ３２１～３２３が情報を保持することができない場合にパーシステントメモリ３２５に情報を保持させる他の技術若しくは構造を用いて実装されたメモリが挙げられる。

【００３０】

処理システム３００は、サスペンドメモリ３２１～３２３に供給される電力又は電圧の喪失又は低下に応じて、１つ以上のサスペンドメモリ３２１～３２３内の情報をパーシステントメモリ３２５に移動させることによって、電源切断／電源投入サイクルの後に、アプリケーション３０１～３０３のうち何れかの実行を再開することができる。例えば、処理システム３００が電源切断動作を開始したことに応じて、１つ以上のアプリケーション３０１～３０３の状態情報を、１つ以上のサスペンドメモリ３２１～３２３からパーシステントメモリ３２５に移動させる。パーシステントメモリ３２５は、処理システム３００が電源切断状態にある間、状態情報を保持する。次に、電力又は電圧がサスペンドメモリ３２１～３２３に復帰したことに応じて（例えば、処理システム３００が電源投入状態に戻ったことに応じて）、状態情報が、パーシステントメモリ３２５から１つ以上のサスペンドメモリ３２１～３２３に書き戻される。次いで、アプリケーション３０１～３０３は、サスペンドメモリ３２１～３２３に記憶された情報を用いて、以前に記憶された状態から動作を開始することができ、例えば、アプリケーション３０１～３０３は、処理システム３００の電源を切断する前の同じ状態から動作を再開することができる。

【００３１】

図４は、いくつかの実施形態による、パイプラインのサブセット上で実行される依存するワークロードの停止／再開動作をサポートする処理システム４００のブロック図である。処理システム４００は、図１に示す処理システム１００、図２に示すグラフィックスパイプライン２００、及び、図３に示す処理システム３００のいくつかの実施形態の一部を表す。

【００３２】

処理システム４００は、グラフィックスパイプライン４１０，４１５と、計算パイプライン４２０，４２１，４２２，４２３，４２４，４２５，４２６，４２７を実装するＧＰＵ４０５と、を含む。これらの計算パイプラインは、本明細書ではまとめて「計算パイプライン４２０～４２７」と呼ばれる。グラフィックスパイプライン４１０，４１５又は計算パイプライン４２０～４２７のいくつかの実施形態は、図２に示すグラフィックスパイプライン２００を用いて実装される。グラフィックスパイプライン４１０，４１５及び計算パイプライン４２０～４２７は、グラフィックスワークロード及び計算ワークロードの各々を同時又は並列に実行するように構成されている。実装に応じて、グラフィックスパイプライン４１０，４１５及び計算パイプライン４２０～４２７は、ＧＰＵ４０５に実装されたプログラム可能なハードウェア及び固定機能実行ハードウェアを共有してもよいし、共有しなくてもよい。本明細書で説明するように、いくつかの実施形態では、グラフィックスパイプライン４１０，４１５及び計算パイプライン４２０～４２７上で実行されるワークロードは、互いに依存する。

【００３３】

図３に示すＯＳ３０５等のＯＳ４３０、及び、図３に示すアプリケーション３０１～３０３等の１つ以上のアプリケーション４３５は、処理システム４００上で実行される。Ｏ

Ｓ４３０及びアプリケーション４３５は、ＧＰＵ４０５内のパイプラインの異なるサブセットを用いて同時又は連続的に実行することができる。図示した実施形態では、ＯＳ４３０は、グラフィックスパイプライン４１５及び計算パイプライン４２０，４２１，４２３を含むパイプラインのサブセット上で実行される。アプリケーション４３５は、グラフィックスパイプライン４１０及び計算パイプライン４２２，４２４～４２７を含む、パイプラインの異なる、相互に排他的なサブセット上で実行される。ＯＳ４３０及びアプリケーション４３５を実行しているサブセットの状態は、対応する状態情報のセットによって定義される。

【００３４】

処理システム４００は、図２に示すサスペンドメモリ２３５及び図３に示すサスペンドメモリ３２１～３２３等の１つ以上のサスペンドメモリ４４０を含む。サスペンドメモリ４４０は、アプリケーション４３５に割り当てられており、アプリケーション４３５が停止したときの状態情報を記憶するために用いられる。例えば、パイプライン４１０，４２２、４２４～４２７の状態情報は、アプリケーション４３５によって生成されたグラフィックスワークロード及び計算ワークロードの実行が停止したことに応じて、サスペンドメモリ４４０に記憶される。パイプライン４１０，４２２，４２４～４２７は、パイプライン４１５，４２０，４２１上のＯＳ４３０の実行が中断することなく継続している間、選択的に停止される。パイプライン４１０，４２２，４２４～４２７上でのアプリケーション４３５の実行が一旦停止されると、他のアプリケーションによって生成された他のグラフィックスワークロード又は計算ワークロードが、実行のためにパイプライン４１０，４２２，４２４～４２７にディスパッチされる。いくつかの実施形態では、パイプライン４２０，４２１，４２３の状態情報をＯＳサスペンドメモリ（分かりやすくするために図示していない）に書き込み、パイプライン４２０，４２１，４２３上のＯＳワークロードの実行を選択的に停止することによって、ＯＳ４３０の実行が停止される。ＯＳ４３０及びアプリケーション４３５の両方によって生成されたワークロードの実行を停止し、対応する状態情報をＯＳサスペンドメモリ及びサスペンドメモリ４４０の各々に記憶してもよい。

【００３５】

パイプライン４１０，４２２，４２４～４２７の一部又は全ての動作状態は、他のグラフィックスワークロード又は計算ワークロードの完了又は停止に応じて、これらの以前の状態に選択的に戻される。例えば、アプリケーション４３５の状態情報を用いて、パイプライン４１０，４２２，４２４～４２７を実装するプロセッサコアを、アプリケーション４３５を停止する前の状態に構成する。したがって、アプリケーション４３５は、アプリケーション４３５が停止したときに実行が中断された状態から正確に実行を再開することができる。ＯＳ４３０が停止した場合、ＯＳ４３０に関連する以前に停止した任意のアプリケーションを再開する前に、ＯＳサスペンドメモリから状態情報を読み出し、この情報を用いてパイプライン４２０，４２１，４２３を構成することによって、ＯＳ４３０が再開される。

【００３６】

図５は、いくつかの実施形態による、ＧＰＵのパイプラインのサブセット上で実行される第１アプリケーションを停止して、当該パイプラインのサブセットを第２アプリケーションに利用させる方法５００のフロー図である。方法５００は、図１に示す処理システム１００、図２に示すグラフィックスパイプライン２００、図３に示す処理システム３００、及び、図４に示す処理システム４００のいくつかの実施形態で実施される。図示した実施形態では、第２アプリケーションは、事前に停止されており、パイプラインのサブセットの状態を定義する状態情報は、対応するサスペンドメモリに記憶されている。しかしながら、方法５００のいくつかの実施形態は、処理システムにおいて以前に実行されていなかったアプリケーションの実行を開始するためにも用いられる。

【００３７】

ブロック５０５において、第１アプリケーションは、ＧＰＵのパイプラインのサブセット上で実行される。例えば、第１アプリケーションは、ＧＰＵの計算パイプラインのサブ

セット及びグラフィックスパイプラインのサブセット上で計算ワークロード及びグラフィックスワークロードを実行する。第１アプリケーションは、計算ワークロード及びグラフィックスワークロード等の第１ワークロード及び第２ワークロードを生成する。第２ワークロードは、第１ワークロードに依存しており、例えば、第１ワークロードは、実行中に第２ワークロードに入力を提供する。

【００３８】

判別ブロック５１０において、処理システムは、停止条件が検出されたかどうかを判別する。停止条件が検出されない場合、ＧＰＵは、パイプラインのサブセット上で第１アプリケーションを実行し続ける。停止条件が検出された場合、方法は、ブロック５１５に進む。

【００３９】

ブロック５１５において、第１アプリケーションの状態情報が、第１サスペンドメモリに記憶される。第１サスペンドメモリは、第１アプリケーションの実行が開始した場合に、停止条件に応じて、又は、処理システム内の他の条件若しくはイベントに応じて、第１アプリケーションに割り当てられる。状態情報は、第１アプリケーションによって生成されたワークロードを実行しているパイプラインのサブセットの状態を表す。パイプラインの他の相互に排他的なパイプラインのサブセットは、他のワークロードを実行し続け、他のサブセットの状態情報は、第１サスペンドメモリに記憶されない。方法５００のこの時点で、第１アプリケーションを実行していたパイプラインのサブセットは、他のアプリケーションに割り当てるのに利用可能である。

【００４０】

ブロック５２０において、第２アプリケーションが事前に停止していた場合、第２アプリケーションの状態情報が第２サスペンドメモリから読み出される。これ以外の場合、第２アプリケーションが新たなアプリケーションとして実行される。この状態情報を用いて、パイプラインのサブセットを、第２アプリケーションの停止前の状態に再構成する。したがって、第２アプリケーションは、第２アプリケーションの停止前のパイプライン状態から正確に実行を再開することができる。

【００４１】

ブロック５２５において、処理システムは、第２サスペンドメモリから読み出された状態情報によって表される状態から開始するパイプラインのサブセットを用いて、第２アプリケーションを実行する。

【００４２】

図６は、いくつかの実施形態による、１つ以上のサスペンドメモリからパーシステントメモリに状態情報を移動させる方法６００のフロー図である。方法は、図３に示す処理システム３００のいくつかの実施形態で実施される。

【００４３】

ブロック６０５において、第１アプリケーション及び第２アプリケーションの状態情報が、対応するサスペンドメモリに記憶される。例えば、いくつかの実施形態では、第１アプリケーションの状態情報は、ＧＰＵ内のパイプラインのサブセット上の第１アプリケーションの実行が停止した場合に、第１サスペンドメモリに記憶される。次に、第２アプリケーションが、このパイプラインのサブセット上で実行され、後に停止される。この時点で、第２アプリケーションの状態情報が第２サスペンドメモリに記憶される。次いで、第３アプリケーションが、パイプラインのサブセット上で実行を開始する。

【００４４】

ブロック６１０において、パイプラインのサブセット上の第３アプリケーションの実行が完了する。

【００４５】

判別ブロック６１５において、処理システムは、電源切断条件が存在するかどうかを判別する。電源切断条件が存在しない場合、処理システムは、電力状態を監視し続ける。電源切断条件が検出された場合、方法６００は、ブロック６２０に進む。

【００４６】
ブロック６２０において、第１アプリケーション及び第２アプリケーションの状態情報がサスペンドメモリから移動され、パーシステントメモリに記憶される。パーシステントメモリは、処理システムが電源切断状態にある間、情報を保持することができる。次に、処理システムは、電源が切断され、電源切断状態に移行する。これにより、サスペンドメモリへの電力又は電圧の供給が中断される。サスペンドメモリに記憶された情報は、電力供給又は電圧供給の中断に応じて喪失又は破壊される（corrupted）。
【００４７】
判別ブロック６２５において、処理システムは、電源投入条件が存在するかどうかを判別する。電源投入条件が検出されない場合、処理システムは、電力状態を監視し続ける。電源投入条件が存在するとプロセッサが判別した場合、方法６００は、ブロック６３０に進む。
【００４８】
ブロック６３０において、第１アプリケーションの状態情報が、パーシステントメモリから移動され、より速いが揮発性である、対応する第１サスペンドメモリであって、第１アプリケーションに割り当てられた第１サスペンドメモリに記憶される。
【００４９】
ブロック６３５において、第２アプリケーションのための状態情報が、パーシステントメモリから移動され、より速いが揮発性である、対応する第２サスペンドメモリであって、第２アプリケーションに割り当てられた第２サスペンドメモリに記憶される。この時点で、記憶された状態情報を用いてパイプラインを再構成することによって、第１アプリケーション又は第２アプリケーションが、パイプラインのサブセット上での実行のために再開される。
【００５０】
第１態様では、方法は、処理ユニットのパイプラインの第１サブセットにおいて第１ワークロードを実行することと、処理ユニットのパイプラインの第２サブセットにおいて、第１ワークロードに依存する第２ワークロードを実行することと、第１ワークロード及び第２ワークロードを停止することと、第１ワークロード及び第２ワークロードを停止したことに応じて、第１ワークロード及び第２ワークロードの状態情報を第１メモリに記憶することと、を含む。
【００５１】
第１態様の一実施形態では、方法は、第１ワークロードを実行することが、処理ユニットの計算パイプラインにおいて計算ワークロードを実行することを含み、第２ワークロードを実行することが、処理ユニットのグラフィックスパイプラインにおいてグラフィックスワークロードを実行することを含む。別の実施形態では、方法は、第１ワークロード及び第２ワークロードを実行することと同時に、処理ユニットのパイプラインの第３サブセットにおいて第３ワークロードを実行することを含み、パイプラインの第１サブセット、第２サブセット及び第３サブセットは、相互に排他的である。特定の実施形態では、方法は、第１ワークロード及び第２ワークロードを停止することが、パイプラインの第３サブセットにおいて第３ワークロードの実行を継続している間、第１ワークロード及び第２ワークロードを停止することを含む。
【００５２】
第１態様のさらなる実施形態では、方法は、第１ワークロード及び第２ワークロードを停止したことに応じて、パイプラインの第１サブセット及び第２サブセットにおいて少なくとも１つの第４ワークロードを実行することを含む。特定の実施形態では、方法は、少なくとも１つの第４ワークロードの完了又は停止に応じて、第１メモリに記憶された状態情報に基づいてパイプラインの第１サブセット及び第２サブセットを構成することと、第１パイプライン及び第２パイプラインを構成した後に、第１ワークロード及び第２ワークロードの実行を第１パイプライン及び第２パイプラインにおいてそれぞれ再開することと、を含む。より特定の実施形態では、方法は、第１ワークロード及び第２ワークロードの

実行を再開することが、停止する前に第１ワークロード及び第２ワークロードを実行した計算ユニットのセットと同じ計算ユニットのセットにおいて第１ワークロード及び第２ワークロードの実行を再開することを含む。

【００５３】

第１態様のさらに別の実施形態では、方法は、処理ユニットの電源を切断したことに応じて、状態情報を第１メモリから第２メモリに移動させることであって、第２メモリは、処理ユニットの電源が切断されている間、状態情報を保持する、ことを含む。特定の実施形態では、方法は、処理ユニットの電源を投入したことに応じて、状態情報を第２メモリから第１メモリに移動させること含む。

【００５４】

第２態様では、装置は、複数のパイプラインを実装するように構成された複数の計算ユニットを含む処理ユニットを備え、処理ユニットは、複数のパイプラインの第１サブセットにおいて第１ワークロードを実行することと、複数のパイプラインの第２サブセットにおいて、第１ワークロードに依存する第２ワークロードを実行することと、第１ワークロード及び第２ワークロードを停止することと、第１ワークロード及び第２ワークロードを停止したことに応じて、第１ワークロード及び第２ワークロードの状態情報を第１メモリに記憶することと、を行うように構成されている。

【００５５】

第２態様の一実施形態では、複数のパイプラインの第１サブセットは、計算ワークロードを実行するように構成された計算パイプラインを含み、複数のパイプラインの第２サブセットは、グラフィックスワークロードを実行するように構成されたグラフィックスパイプラインを含む。別の実施形態では、処理ユニットは、第１ワークロード及び第２ワークロードを実行することと同時に、複数のパイプラインの第３サブセットにおいて第３ワークロードを実行するように構成されており、複数のパイプラインの第１サブセット、第２サブセット及び第３サブセットは、相互に排他的である。特定の実施形態では、処理ユニットは、パイプラインの第３サブセットにおいて第３ワークロードの実行を継続している間、第１ワークロード及び第２ワークロードを停止するように構成されている。

【００５６】

第２態様の別の実施形態では、処理ユニットは、第１ワークロード及び第２ワークロードを停止したことに応じて、パイプラインの第１サブセット及び第２サブセットにおいて少なくとも１つの第４ワークロードを実行するように構成されている。特定の実施形態では、パイプラインの第１サブセット及び第２サブセットが、少なくとも１つの第４ワークロードの完了又は停止に応じて、第１メモリに記憶された状態情報に基づいて構成され、処理ユニットは、パイプラインの第１サブセット及び第２サブセットを構成した後に、パイプラインの第１サブセット及び第２サブセットにおいて第１ワークロード及び第２ワークロードの実行をそれぞれ再開する。より特定の実施形態では、処理ユニットは、停止する前に第１ワークロード及び第２ワークロードを実行した複数の計算ユニットのサブセットと同じ複数の計算ユニットのサブセットにおいて第１ワークロード及び第２ワークロードの実行を再開するように構成されている。

【００５７】

第２態様のさらなる実施形態では、処理ユニットは、処理ユニットの電源を切断したことに応じて、状態情報を第１メモリから第２メモリに移動させるように構成されており、第２メモリは、処理ユニットの電源が切断されている間、状態情報を保持する。特定の実施形態では、処理ユニットは、処理ユニットの電源を投入したことに応じて、状態情報を第２メモリから第１メモリに移動させるように構成されている。

【００５８】

第３態様では、方法は、処理ユニットの計算ユニットの第１サブセットに実装された複数のパイプラインの第１サブセットで実行されている第１ワークロードを停止することであって、ワークロード間に依存関係が存在する、ことと、第１ワークロードの状態情報を記憶することと、記憶された状態情報に基づいて、計算ユニットの第１サブセットに実装

された複数のパイプラインの第１サブセットにおいて第１ワークロードの実行を再開することと、を含む。

【００５９】

第３態様の一実施形態では、第１ワークロードを停止することは、計算ユニットの第２サブセットに実装された複数のパイプラインの第２サブセットにおいて少なくとも１つの第２ワークロードが実行されている間、第１ワークロードを停止することを含む。

【００６０】

いくつかの実施形態では、上記の装置及び技術は、図１～図６を参照して上述した処理システム等の１つ以上の集積回路（ＩＣ）デバイス（集積回路パッケージ又はマイクロチップとも呼ばれる）を含むシステムに実装される。これらのＩＣデバイスの設計及び製造には、電子設計自動化（ＥＤＡ）及びコンピュータ支援設計（ＣＡＤ）ソフトウェアツールが使用される。これらの設計ツールは、通常、１つ以上のソフトウェアプログラムとして表される。１つ以上のソフトウェアプログラムは、回路を製造するための製造システムを設計又は適合するための処理の少なくとも一部を実行するように１つ以上のＩＣデバイスの回路を表すコードで動作するようにコンピュータシステムを操作する、コンピュータシステムによって実行可能なコードを含む。このコードは、命令、データ、又は、命令及びデータの組み合わせを含むことができる。設計ツール又は製造ツールを表すソフトウェア命令は、通常、コンピューティングシステムがアクセス可能なコンピュータ可読記憶媒体に記憶される。同様に、ＩＣデバイスの設計又は製造の１つ以上のフェーズを表すコードは、同じコンピュータ可読記憶媒体又は異なるコンピュータ可読記憶媒体に記憶されてもよいし、同じコンピュータ可読記憶媒体又は異なるコンピュータ可読記憶媒体からアクセスされてもよい。

【００６１】

コンピュータ可読記憶媒体は、命令及び／又はデータをコンピュータシステムに提供するために、使用中にコンピュータシステムによってアクセス可能な任意の非一時的な記憶媒体又は非一時的な記憶媒体の組み合わせを含む。かかる記憶媒体には、限定されないが、光媒体（例えば、コンパクトディスク（ＣＤ）、デジタル多用途ディスク（ＤＶＤ）、ブルーレイ（登録商標）ディスク）、磁気媒体（例えば、フロッピー（登録商標）ディスク、磁気テープ、磁気ハードドライブ）、揮発性メモリ（例えば、ランダムアクセスメモリ（ＲＡＭ）、キャッシュ）、不揮発性メモリ（例えば、読み出し専用メモリ（ＲＯＭ）、フラッシュメモリ）、又は、微小電気機械システム（ＭＥＭＳ）ベースの記憶媒体が含まれ得る。コンピュータ可読記憶媒体は、コンピュータシステムに内蔵されてもよいし（例えば、システムＲＡＭ又はＲＯＭ）、コンピュータシステムに固定的に取り付けられてもよいし（例えば、磁気ハードドライブ）、コンピュータシステムに着脱可能に取り付けられてもよいし（例えば、光学ディスク又はユニバーサルシリアルバス（ＵＳＢ）ベースのフラッシュメモリ）、有線又は無線のネットワークを介してコンピュータシステムに接続されてもよい（例えば、ネットワークアクセス可能なストレージ（ＮＡＳ））。

【００６２】

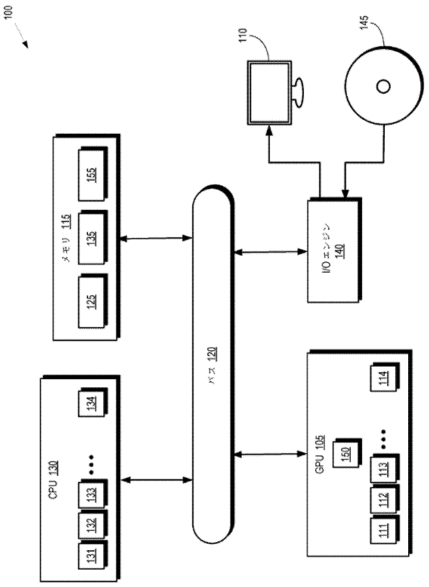
いくつかの実施形態では、上記の技術のいくつかの態様は、ソフトウェアを実行する処理システムの１つ以上のプロセッサによって実装されてもよい。ソフトウェアは、非一時的なコンピュータ可読記憶媒体に記憶され、又は、非一時的なコンピュータ可読記憶媒体上で有形に具現化された実行可能命令の１つ以上のセットを含む。ソフトウェアは、１つ以上のプロセッサによって実行されると、上記の技術の１つ以上の態様を実行するように１つ以上のプロセッサを操作する命令及び特定のデータを含むことができる。非一時的なコンピュータ可読記憶媒体は、例えば、磁気若しくは光ディスク記憶デバイス、例えばフラッシュメモリ等のソリッドステート記憶デバイス、キャッシュ、ランダムアクセスメモリ（ＲＡＭ）、又は、他の不揮発性メモリデバイス等を含むことができる。非一時的なコンピュータ可読記憶媒体に記憶された実行可能命令は、ソースコード、アセンブリ言語コード、オブジェクトコード、又は、１つ以上のプロセッサによって解釈若しくは実行可能な他の命令フォーマットであってもよい。

【００６３】
上述したものに加えて、概要説明において説明した全てのアクティビティ又は要素が必要とされているわけではなく、特定のアクティビティ又はデバイスの一部が必要とされない場合があり、１つ以上のさらなるアクティビティが実行される場合があり、１つ以上のさらなる要素が含まれる場合があることに留意されたい。さらに、アクティビティが列挙された順序は、必ずしもそれらが実行される順序ではない。また、概念は、特定の実施形態を参照して説明された。しかしながら、当業者であれば、特許請求の範囲に記載されているような本発明の範囲から逸脱することなく、様々な変更及び変形を行うことができるのを理解するであろう。したがって、明細書及び図面は、限定的な意味ではなく例示的な意味で考慮されるべきであり、これらの変更形態の全ては、本発明の範囲内に含まれることが意図される。

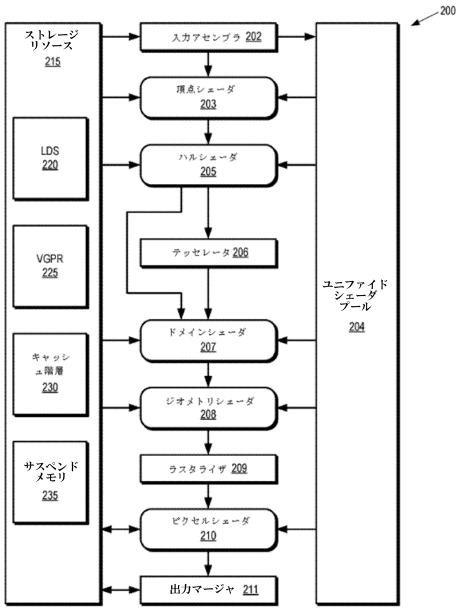
【００６４】
利益、他の利点及び問題に対する解決手段を、特定の実施形態に関して上述した。しかし、利益、利点、問題に対する解決手段、及び、何かしらの利益、利点若しくは解決手段が発生又は顕在化する可能性のある特徴は、何れか若しくは全ての請求項に重要な、必須の、又は、不可欠な特徴と解釈されない。さらに、開示された発明は、本明細書の教示の利益を有する当業者には明らかな方法であって、異なっているが同様の方法で修正され実施され得ることから、上述した特定の実施形態は例示にすぎない。添付の特許請求の範囲に記載されている以外に本明細書に示されている構成又は設計の詳細については限定がない。したがって、上述した特定の実施形態は、変更又は修正されてもよく、かかる変更形態の全ては、開示された発明の範囲内にあると考えられることが明らかである。したがって、ここで要求される保護は、添付の特許請求の範囲に記載されている。

【図面】

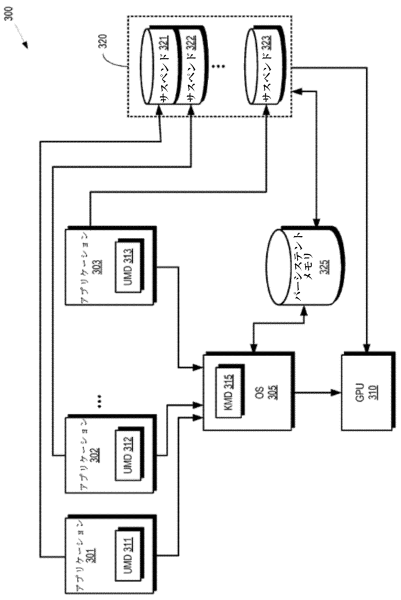
【図１】


100
110
145
メモリ 115
155
135
125
I/O エンジン 140
バス 120
CPU 130
124
133
132
131
GPU 105
150
114
113
112
111

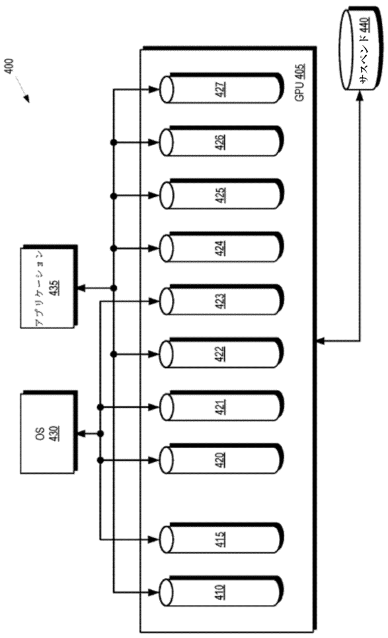
【図２】


200
ストレージリソース 215
LDS 220
VGPR 225
キャッシュ階層 230
サスペンドメモリ 235
入力アセンブラ 202
頂点シェーダ 203
ハルシェーダ 205
テッセレータ 206
ドメインシェーダ 207
ジオメトリシェーダ 208
ラスタライザ 209
ピクセルシェーダ 210
出力マージャ 211
ユニファイドシェーダプール 204

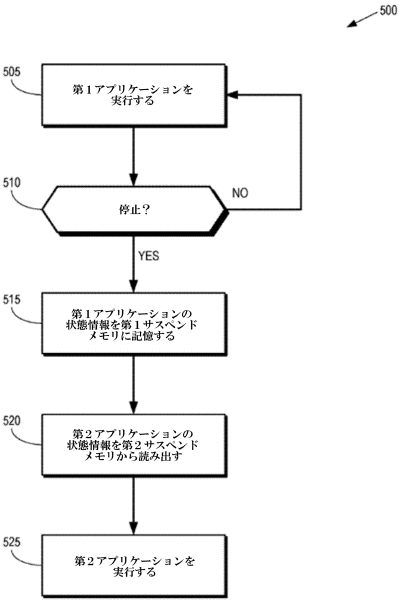
【図３】


300
320
サスペンド 321
サスペンド 322
サスペンド 323
アプリケーション 303
UMD 313
アプリケーション 302
UMD 312
アプリケーション 301
UMD 311
パーシステントメモリ 325
KMD 315
OS 305
GPU 310

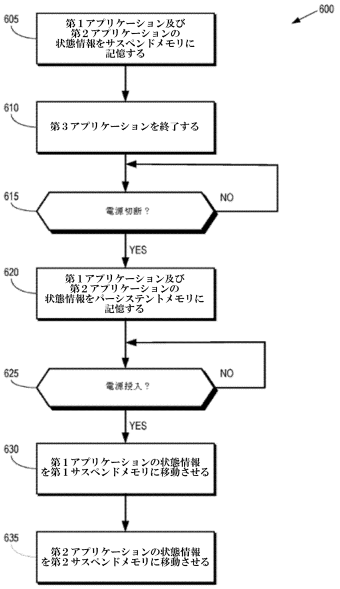
【図４】


400
サスペンド 440
GPU 405
427
426
425
424
423
422
421
420
415
410
アプリケーション 435
OS 430

【図５】


500
505
第１アプリケーションを
実行する
510
停止？
NO
YES
515
第１アプリケーションの
状態情報を第１サスペンド
メモリに記憶する
520
第２アプリケーションの
状態情報を第２サスペンド
メモリから読み出す
525
第２アプリケーションを
実行する

【図６】


600
605
第１アプリケーション及び
第２アプリケーションの
状態情報をサスペンドメモリに
記憶する
610
第３アプリケーションを終了する
615
電源切断？
NO
YES
620
第１アプリケーション及び
第２アプリケーションの
状態情報をパーシステントメモリに
記憶する
625
電源投入？
NO
YES
630
第１アプリケーションの状態情報
を第１サスペンドメモリに移動させる
635
第２アプリケーションの状態情報
を第２サスペンドメモリに移動させる

フロントページの続き

アメリカ合衆国 ９５０５４ カリフォルニア州、サンタ クララ、オーガスティン ドライブ ２４８５
(72)発明者 マイケル マントル
アメリカ合衆国 ９５０５４ カリフォルニア州、サンタ クララ、オーガスティン ドライブ ２４８５

審査官 岡本 俊威
(56)参考文献 米国特許出願公開第２０１７／００９１８９５（ＵＳ，Ａ１）
米国特許出願公開第２０１５／０１７８８７９（ＵＳ，Ａ１）
米国特許出願公開第２０１４／０１９８１１６（ＵＳ，Ａ１）
(58)調査した分野 (Int.Cl.，ＤＢ名)
Ｇ０６Ｔ １／２０
Ｇ０６Ｆ ９／３８