

(19)日本国特許庁(JP)

(12)特許公報(B2)

(11)特許番号
特許第7512032号
(P7512032)

(45)発行日 令和6年7月8日(2024.7.8)

(24)登録日 令和6年6月28日(2024.6.28)

(51)国際特許分類

F I

G 0 6 T 15/00 (2011.01)

G 0 6 T 15/00 5 0 1

請求項の数 18 外国語出願 (全34頁)

(21)出願番号	特願2019-218914(P2019-218914)	(73)特許権者	515293676
(22)出願日	令和1年12月3日(2019.12.3)		イマジネーション テクノロジーズ リミ
(65)公開番号	特開2020-109625(P2020-109625		テッド
	A)		Imagination Techno
(43)公開日	令和2年7月16日(2020.7.16)		logies Limited
審査請求日	令和4年11月7日(2022.11.7)		英国 ダブリューディー4 8エルゼッド
(31)優先権主張番号	1819810.1		ハートフォードシャー キングズ ラング
(32)優先日	平成30年12月4日(2018.12.4)		リー ホーム パーク エステート イマジ
(33)優先権主張国・地域又は機関	英国(GB)		ネーション ハウス
			Imagination House,
			Home Park Estate, K
			ings Langley, Hertf
			ordshire WD4 8LZ Un
			ited Kingdom
		(74)代理人	100091982

最終頁に続く

(54)【発明の名称】 タイル領域保護

(57)【特許請求の範囲】

【請求項1】

安全重要要素を含むシーンのタイルベースのレンダリングを実施するためのグラフィック処理システムであって、

幾何学形状処理フェーズにおいて、安全重要要素を含む保護されたタイルを識別するように構成された幾何学形状エンジンと、

断片処理フェーズにおいて、それぞれ、第一および第二の断片処理された出力を生成するように、一度目と二度目に、前記保護されたタイルのそれぞれを処理し、安全重要要素を含むとして識別されないレンダリングされるフレームの前記タイルに対して一度、断片処理を実行するように構成される断片処理エンジンと、

前記保護されたタイルのそれぞれに対して、前記第一および第二の断片処理された出力を比較し、前記第一および第二の断片処理された出力が一致しない場合に、フォールト信号を上げるように構成されたチェックユニットと、を含む、グラフィック処理システム。

【請求項2】

前記グラフィック処理システムが、それぞれが、前記幾何学形状処理フェーズおよび／または前記断片処理フェーズを実施するよう動作可能な一つまたは複数の処理ユニットを含む、請求項1に記載のグラフィック処理システム。

【請求項3】

前記グラフィック処理システムが、複数の処理ユニットを含み、前記断片処理フェーズが、前記複数の処理ユニットの第一の処理ユニットで一度目を実施され、前記複数の処理

ユニットの第二の処理ユニットで二度目に実施される、請求項 1 または 2 に記載のグラフィック処理システム。

【請求項 4】

前記グラフィック処理システムが、各保護されたタイルを前記第一および第二の処理ユニットに同時提出するように配置されている、請求項 3 に記載のグラフィック処理システム。

【請求項 5】

前記グラフィック処理システムが、三つ以上の処理ユニットを含み、前記第二の処理ユニットが前記第一の処理ユニット以外の前記複数の処理ユニットの任意の処理ユニットに拘束されるように構成されている、請求項 3 または 4 に記載のグラフィック処理システム。

10

【請求項 6】

前記チェックユニットが、

前記第一の処理ユニットからの前記出力の特徴である一つまたは複数の第一のシグニチャを形成し、

前記第二の処理ユニットからの前記出力の特徴である一つまたは複数の第二のシグニチャを形成し、

前記個々の第一および第二のシグニチャを比較することによって、前記第一および第二の断片処理された出力の比較を実施するように構成される、請求項 3 ～ 5 のいずれかに記載のグラフィック処理システム。

【請求項 7】

20

前記第一および第二の処理ユニットが、データストアで一つまたは複数のバッファに、一つまたは複数の書き込みを実行することによって、個々の第一および第二の断片処理された出力を書き込むようにそれぞれ構成され、前記チェックユニットが、前記第一の処理ユニットによる前記一つまたは複数のバッファへの一つまたは複数の書き込みのグループに各第一のシグニチャを形成し、前記第二の処理ユニットによる前記一つまたは複数のバッファへの一つまたは複数の書き込みのグループに各第二のシグニチャを形成するように構成される、請求項 6 に記載のグラフィック処理システム。

【請求項 8】

前記グラフィック処理システムが、前記第一の処理ユニットによる前記一つまたは複数のバッファへの書き込みを可能にし、前記第二の処理ユニットによる前記一つまたは複数のバッファへの書き込みを防止するように構成される、請求項 6 または 7 に記載のグラフィック処理システム。

30

【請求項 9】

前記幾何学形状エンジンが、それぞれ、第一および第二の幾何学形状処理された出力を生成するように、レンダリングされるフレームに対して、一度目と二度目に、前記幾何学形状処理フェーズを実行するように構成され、前記チェックユニットが、前記第一および第二の幾何学形状処理された出力を比較し、前記第一および第二の幾何学形状処理された出力が一致しない場合に、フォールト信号を上げるよう構成されている、請求項 1 ～ 8 のいずれかに記載のグラフィック処理システム。

【請求項 10】

40

前記幾何学形状エンジンが、どの要素がレンダリングされるフレームのタイルのそれぞれに含まれるかを識別するように、タイリングを実行するように構成され、かつ保護されたタイルとして、少なくとも部分的に安全重要要素を含む前記タイルをマークするように構成された、請求項 1 ～ 9 のいずれかに記載のグラフィック処理システム。

【請求項 11】

前記幾何学形状エンジンが、

それぞれ、第一および第二のタイル出力を生成するように、フレームの前記タイルの全てに関して、1 度目および 2 度目にタイリングを実行し、

前記チェックユニットで、前記第一および第二のタイル出力を比較して、前記第一および第二のタイル出力が一致しない場合に、フォールト信号を上げるように構成される、請

50

求項 1 ～ 1 0 のいずれかに記載のグラフィック処理システム。

【請求項 1 2】

前記シーンの要素を定義するドロークールを受信するためのインターフェースをさらに含み、前記インターフェースが、前記個々の要素を安全重要要素として前記グラフィック処理システムで取り扱うことができるように、ドロークールが安全重要であるとして識別されるのを許可するように構成される、請求項 1 ～ 1 1 のいずれかに記載のグラフィック処理システム。

【請求項 1 3】

各保護されたタイルが所定の安全レベルに従って処理するためである、請求項 1 ～ 1 2 のいずれかに記載のグラフィック処理システム。

【請求項 1 4】

安全重要要素を含むシーンのタイルベースのレンダリングを実施する方法であって、幾何学形状処理フェーズにおいて、安全重要要素を含む保護されたタイルを識別することと、

断片処理フェーズにおいて、それぞれ第一および第二の断片処理された出力を生成するように、1 度目および 2 度目に前記保護されたタイルのそれぞれを処理し、安全重要要素を含むとして識別されないレンダリングされるフレームの前記タイルに対して一度、断片処理を実行することと、

前記保護されたタイルのそれぞれに対して、前記第一および第二の断片処理された出力を比較し、前記第一および第二の断片処理された出力が一致しない場合に、フォールト信号を上げることと、を含む、方法。

【請求項 1 5】

安全重要要素を含むシーンのタイルベースのレンダリングを実施するためのグラフィック処理ユニットであって、

安全重要要素を含む保護されたタイルに関連するデータおよび安全重要要素を含まない保護されていないタイルに関連するデータを受信し、

断片処理フェーズにおいて、

それぞれ第一および第二の断片処理された出力を生成するように、1 度目および 2 度目に、各保護されたタイルを処理し、

個々の単一の断片処理された出力を生成するように、一度、各保護されていないタイルを処理するように構成され、

前記グラフィック処理ユニットが、保護されたタイルのそれぞれに対して、前記第一および第二の断片処理された出力を比較し、前記第一および第二の断片処理された出力が一致しない場合に、フォールト信号を上げるように構成されたチェックユニットを含む、グラフィック処理ユニット。

【請求項 1 6】

前記グラフィック処理ユニットが、各第一の断片処理された出力および各単一断片処理された出力をデータストアに書き込み、各第二の断片処理された出力が前記データストアに書き込まれるのを防止するように構成されている、請求項 1 5 に記載のグラフィック処理ユニット。

【請求項 1 7】

安全重要要素を含むシーンのタイルベースのレンダリングを実施する方法であって、

安全重要要素を含む保護されたタイルに関連するデータおよび安全重要要素を含まない保護されていないタイルに関連するデータを受信することと、

断片処理フェーズにおいて、

それぞれ第一および第二の断片処理された出力を生成するように、一度目および二度目に、各保護されたタイルを処理することと、

個々の単一の断片処理された出力を生成するように、一度、各保護されていないタイルを処理することと、

前記第一および第二の断片処理された出力を比較し、前記第一および第二の断片処理さ

10

20

30

40

50

れた出力が一致しない場合に、フォールト信号を上げることと、を含む、方法。

【請求項 18】

請求項 14 または 17 に記載の方法を実施するためのコンピュータプログラムコード。

【発明の詳細な説明】

【技術分野】

【0001】

本開示は、グラフィック処理システムおよびグラフィック処理システムでタスクを処理する方法に関する。

【背景技術】

【0002】

安全重要システムでは、システムの構成要素の少なくとも一部は、システム全体が、システムに必要とされる安全レベルを満たすために十分な安全目標を満たす必要がある。例えば、ほとんどの管轄区域では、車両におけるシートベルトのリトラクタは、こうした装置を含んだ車両が安全試験をパスするために特定の安全基準を満たす必要がある。同様に、車両のタイヤは、タイヤを装備した車両が特定の管轄地域に適切な安全試験をパスするために、特定の基準を満たす必要がある。安全重要システムは通常、故障が人々または環境の安全に対するリスクの大幅な増加を起こすシステムである。

【0003】

データ処理装置は、専用のハードウェアとして、または安全重要のソフトウェアを実行するためのプロセッサとして、安全重要システムの一体型部品を形成することが多い。例えば、航空機のフライバイワイヤーシステム、ドライバ支援システム、鉄道信号システム、および医療機器の制御システムは、通常、データ処理装置上で実行される安全重要システムである。データ処理装置が安全重要システムの一体化された部分を形成する場合、データ処理装置自体は、システム全体が適切な安全レベルを満たすことができるように、安全目標を満たす必要がある。自動車業界では、安全レベルは通常、機能安全基準 ISO 26262 で定義されている自動車用安全度水準 (ASIL) である。

【0004】

また、安全重要システムのデータ処理装置は、ソフトウェアを実行するプロセッサを含む。ハードウェアおよびソフトウェア要素の両方は、特定の安全目標を満たす必要がある。ソフトウェア障害は通常、プログラミングエラーまたは不適切なエラー処理によるシステム障害である。ソフトウェアについては、安全目標は通常、厳密な開発実践、コード監査および試験プロトコルによって達成される。プロセッサなどのデータ処理装置のハードウェア要素の場合、安全目標は、一定期間内の障害の最大数（多くの場合、時間内障害、または FIT として表される）および単一ポイント障害（単一ポイント障害メカニズム、または SPFM）および潜在的な障害（潜在障害メカニズム、または LFM）を検出するメカニズムの有効性などの一連のメトリックとして表される場合がある。データ処理ハードウェアは、ある程度のエラーを処理するように設計されていることが重要である。なぜなら、システムエラーを安全重要システムから完全に除外できたとしても、ランダムエラーが例えば、一時的なイベント（例えば、電離放射線、電圧スパイク、または電磁パルスによる）により、ハードウェアに導入される可能性がある。バイナリシステムにおいて、一時的なイベントが、メモリ内のランダムなビットフリップとプロセッサのデータパスによって生じる可能性がある。

【0005】

例えば、一つの構成要素が故障した時、別のものが同一のタスクを実行できる、または、チェックデータの使用（例えば、パリティビットまたはエラー修正コード）によって、ハードウェアがマイナーなデータ故障を検出および／または修正できるように、冗長性を提供することによって、データ処理ハードウェアで安全目標を達成するためのさまざまなアプローチがある。データプロセッサは、一対の同一の処理コア 101 および 102 が命令 103 の流れを並列に処理するよう構成されている図 1 に示す、デュアルロックステップ配列 100 内に提供され得る。処理コア (101) のいずれか一つの出力は、ロックス

10

20

30

40

50

テッププロセッサの出力104として使用され得る。処理コア101および102の出力が一致しない場合、フォールトを安全重要システムに上げることができる。しかしながら、第二の処理コアは必要であるため、デュアルロックステッププロセッサは必ず従来のプロセッサと比較して二倍のチップエリアを消費し、約二倍の電力を消費する。

【0006】

電離放射線および電圧スパイクなどの外因性因子によって誘発されるエラーの検出確率を改善するために、遅延105をコアの一つへの入力に導入することができる（一般に、対応する遅延106がその他のコアの出力に提供される）。さらにプロセッサコアをロックステッププロセッサ100に追加することにより、エラーのない出力を提供することが可能となりうる。プロセッサの出力は、その他のコアと一致しないその処理コアの出力が無視できない状態で、二つ以上の処理コアによって提供され得る。しかし、これはさらに、プロセッサのエリアおよび電力消費量を増加させる。

【0007】

高度なドライバ支援システムおよび自動車両は、重要なグラフィックおよび／またはベクトル処理能力を持つこのような安全重要用途に適したデータ処理システムを組み込むことができるが、デュアルロックステッププロセッサを実施するためのエリアおよび電力消費量の増加（およびコスト）は許容されないし望ましくないことがあり得る。例えば、ドライバ支援システムは多くの場合、ハザード、レーン位置、およびその他の情報をドライバに示すコンピュータ生成グラフィックを提供する。典型的には、これは、車両製造業者が従来型の機器クラスタを、速度および車両フォールト情報などの安全重要情報の表示がコンピュータ生成されることも意味する、コンピュータ生成機器クラスタと置き換えさせることにつながる。このような処理要求は、グラフィック処理ユニット（GPU）によって満たされ得る。しかしながら、自動車のコンテキストでは、高度なドライバ支援システムは通常、ISO 26262のASILレベルBを満たすデータ処理システムを必要とする。

【0008】

自動車両は、安全重要な決定を行うために、リアルタイムで（例えば、レーダー、LIDAR、マップデータおよび車両情報から）非常に大量のデータを追加的に処理する必要がある。グラフィック処理ユニットはまた、こうした処理要求を満たすことができるが、自動車両における安全重要システムは通常、ISO 26262の最も厳格なASILレベルDを満たすために役立つ。

【発明の概要】

【0009】

この概要は、詳細な説明で以下にさらに説明されている概念を紹介するために提供されている。本要約は、主題の主要特徴または必須特徴を特定することを意図しておらず、請求項の主題の範囲を制限するために使用されることを意図していない。

【0010】

安全重要要素を含むシーンのタイルベースのレンダリングを実施するためのグラフィック処理システムが提供される。グラフィック処理システムは、幾何学形状処理フェーズにおいて、安全重要要素を含む保護されたタイルを識別するように構成された幾何学形状エンジンと、断片処理フェーズにおいて、それぞれ、第一および第二の断片処理された出力を生成するように、一度目と二度目に、保護されたタイルのそれぞれを処理するように構成される断片処理エンジンと、保護されたタイルのそれぞれに対して、第一および第二の断片処理された出力を比較し、第一および第二の断片処理された出力が一致しない場合に、フォールト信号を上げるように構成されたチェックユニットと、を含む。

【0011】

断片処理エンジンが、安全重要要素を含むとして識別されないレンダリングされるフレームのタイルに対して一度、断片処理を実行するよう構成され得る。

【0012】

グラフィック処理システムが、それぞれが、幾何学形状処理フェーズおよび／または断

10

20

30

40

50

片処理フェーズを実施するよう動作可能な一つまたは複数の処理ユニットをさらに含んでもよい。

【0013】

グラフィック処理システムが、複数の処理ユニットをさらに含んでもよく、断片処理フェーズが、複数の処理ユニットの第一の処理ユニットで一度目を実施され、複数の処理ユニットの第二の処理ユニットで二度目を実施され得る。

【0014】

グラフィック処理システムが、各保護されたタイルを第一および第二の処理ユニットに同時提出するように配置され得る。

【0015】

グラフィック処理システムが、三つ以上の処理ユニットをさらに含んでもよく、第二の処理ユニットが第一の処理ユニット以外の複数の処理ユニットの任意の処理ユニットに拘束されるように構成され得る。

【0016】

チェックユニットが、第一の処理ユニットからの出力の特徴である一つまたは複数の第一のシグニチャを形成し、第二の処理ユニットからの出力の特徴である一つまたは複数の第二のシグニチャを形成し、第一および第二のシグニチャを比較することによって、第一および第二の断片処理された出力の比較を実施するように構成され得る。

【0017】

第一および第二の処理ユニットが、データストアで一つまたは複数のバッファに、一つまたは複数の書き込みを実行することによって、個々の第一および第二の断片処理された出力を書き込むようにそれぞれ構成されてもよく、チェックユニットが、第一の処理ユニットによる一つまたは複数のバッファへの一つまたは複数の書き込みのグループに各第一のシグニチャを形成し、第二の処理ユニットによる一つまたは複数のバッファへの一つまたは複数の書き込みのグループに各第二のシグニチャを形成するように構成され得る。

【0018】

グラフィック処理システムが、第一の処理ユニットによる一つまたは複数のバッファへの書き込みを可能にし、第二の処理ユニットによる一つまたは複数のバッファへの書き込みを防止するように構成され得る。

【0019】

断片処理エンジンが、データベース上のメモリに書き込むように動作可能であってもよく、グラフィック処理システムが、第一の処理ユニットをミッション処理ユニットとして指定し、第二の処理ユニットを安全処理ユニットとして指定し、ミッション処理ユニットによるメモリへの書き込みを許可し、安全処理ユニットによるメモリへの書き込みを許可しないように構成され得る。

【0020】

幾何学形状エンジンが、それぞれ、第一および第二の幾何学形状処理された出力を生成するように、レンダリングされるフレームに対して、一度目と二度目に、幾何学形状処理フェーズを実行するように構成されてもよく、チェックユニットが、第一および第二の幾何学形状処理された出力を比較し、第一および第二の幾何学形状処理された出力が一致しない場合に、フォールト信号を上げるよう構成され得る。

【0021】

幾何学形状エンジンが、複数の幾何学形状ユニットを含んでもよく、幾何学形状処理フェーズが、複数の幾何学形状ユニットの第一の幾何学形状ユニットで一度目に少なくとも部分的に実施され、複数の幾何学形状ユニットの第二の幾何学形状ユニットで二度目に少なくとも部分的に実施され得る。

【0022】

チェックユニットが、第一の幾何学形状ユニットからの出力の特徴である一つまたは複数の第一のシグニチャを形成し、第二の幾何学形状ユニットからの出力の特徴である一つまたは複数の第二のシグニチャを形成し、個々の第一および第二のシグニチャを比較する

10

20

30

40

50

ことによって、第一および第二の幾何学形状処理された出力の比較を実施するように構成され得る。

【0023】

第一および第二の幾何学形状ユニットが、データストアで一つまたは複数のバッファに、一つまたは複数の書き込みを実行することによって、個々の第一および第二の幾何学形状処理された出力を書き込むようにそれぞれ構成されてもよく、チェックユニットが、第一の幾何学形状ユニットによるバッファへの一つまたは複数の書き込みのグループに各第一のシグニチャを形成し、第二の幾何学形状ユニットによるバッファへの一つまたは複数の書き込みのグループに各第二のシグニチャを形成するように構成され得る。

【0024】

幾何学形状エンジンが、どの要素がレンダリングされるフレームのタイルのそれぞれに含まれるかを識別するように、タイリングを実行するように構成され、かつ保護されたタイルとして、少なくとも部分的に安全重要要素を含むタイルをマークするように構成され得る。

【0025】

幾何学形状エンジンが、データバス上のメモリに書き込むように動作可能であってもよく、グラフィック処理システムが、第一の幾何学形状ユニットをミッション幾何学形状ユニットとして指定し、第二の幾何学形状ユニットを安全幾何学形状ユニットとして指定し、ミッション幾何学形状ユニットによるメモリへの書き込みを許可し、安全幾何学形状ユニットによるメモリへの書き込みを許可しないように構成され得る。

【0026】

幾何学形状エンジンが、シーンの要素を定義する幾何学形状データを受信するように配置されてもよく、幾何学形状処理フェーズが、幾何学形状データを、レンダリングされるフレームの視点から要素を表す変換された幾何学形状データに変換することを含んでもよい。

【0027】

幾何学形状エンジンが、それぞれ、第一および第二のタイル出力を生成するように、フレームのタイルの全てに関して、一度目および二度目にタイリングを実行し、チェックユニットで、第一および第二のタイル出力を比較して、第一および第二のタイル出力が一致しない場合に、フォールト信号を上げるように構成され得る。

【0028】

グラフィック処理システムは、シーンの要素を定義するドローコールを受信するためのインターフェースをさらに含み、インターフェースが、個々の要素を安全重要要素としてグラフィック処理システムで取り扱うことができるように、ドローコールが安全重要であるとして識別されるのを許可するように構成され得る。

【0029】

インターフェースで受信された各ドローコールが、個々の要素が安全重要要素であるかどうかを示す識別子を含んでもよい。

【0030】

データストアが、グラフィック処理システムのグラフィック処理ユニットに位置するか、または外部に位置する一つまたは複数のメモリ、および／またはグラフィック処理システムのグラフィック処理ユニットに位置するか、または外部に位置する一つまたは複数のキャッシュを含んでもよい。

【0031】

各保護されたタイルが所定の安全レベルに従って処理するためでもよい。

【0032】

安全重要要素を含むシーンのタイルベースのレンダリングを実施する方法を提供する。方法は、幾何学形状処理フェーズにおいて、安全重要要素を含む保護されたタイルを識別することと、断片処理フェーズにおいて、それぞれ第一および第二の断片処理された出力を生成するように、一度目および二度目に保護されたタイルのそれぞれを処理することと

10

20

30

40

50

、保護されたタイルのそれぞれに対して、第一および第二の断片処理された出力を比較し、第一および第二の断片処理された出力が一致しない場合に、フォールト信号を上げることを含む。

【0033】

方法は、断片処理フェーズにおいて、安全重要要素を含むものとして識別されていないタイルを一度処理することをさらに含んでもよい。

【0034】

安全重要要素を含むシーンのタイルベースのレンダリングを実施するためのグラフィック処理ユニットを提供する。グラフィック処理ユニットが、安全重要要素を含む保護されたタイルに関連するデータおよび安全重要要素を含まない保護されていないタイルに関連するデータを受信し、断片処理フェーズにおいて、それぞれ第一および第二の断片処理された出力を生成するように、一度目および二度目に、各保護されたタイルを処理し、個々の単一の断片処理された出力を生成するように、一度、各保護されていないタイルを処理するように構成され、グラフィック処理システムが、保護されたタイルのそれぞれに対して、第一および第二の断片処理された出力を比較し、第一および第二の断片処理された出力が一致しない場合に、フォールト信号を上げるように構成されたチェックユニットと、を含む。

【0035】

グラフィック処理ユニットが、各第一の断片処理された出力および各単一断片処理された出力をデータストアに書き込み、各第二の断片処理された出力がデータストアに書き込まれるのを防止するように構成され得る。

【0036】

グラフィック処理ユニットが、複数の処理ユニットを含み、各保護されたタイルに対して一度目の断片処理を実施するように構成された第一の処理ユニットと、各保護されたタイルに対して二度目の断片処理を実施するように構成された第二の処理ユニットとを含んでもよい。

【0037】

グラフィック処理ユニットが、複数の処理ユニットのいずれかで各保護されていないタイルを処理するように構成され得る。

【0038】

グラフィック処理ユニットが三つ以上の処理ユニットを含み、第一および第二の処理ユニットを含まない複数の処理ユニットのいずれかで各保護されていないタイルを処理するように構成され得る。

【0039】

安全重要要素を含むシーンのタイルベースのレンダリングを実施する方法を提供する。方法は、安全重要要素を含む保護されたタイルに関連するデータおよび安全重要要素を含まない保護されていないタイルに関連するデータを受信することと、断片処理フェーズにおいて、それぞれ第一および第二の断片処理された出力を生成するように、一度目および二度目に、各保護されたタイルを処理することと、個々の単一の断片処理された出力を生成するように、一度、各保護されていないタイルを処理することと、第一および第二の断片処理された出力を比較し、第一および第二の断片処理された出力が一致しない場合に、フォールト信号を上げることと、を含む。

【0040】

グラフィック処理システムおよび／またはグラフィック処理ユニットが集積回路上のハードウェア内に具体化され得る。集積回路製造システム、グラフィック処理システム、および／またはグラフィック処理ユニットにおいて、製造方法が提供され得る。集積回路製造システムで処理された時、グラフィック処理システムおよび／またはグラフィック処理ユニットを製造するために、システムを構成する、集積回路定義データセットが提供され得る。集積回路製造システムで処理された時に、集積回路製造システムにグラフィック処理システムおよび／またはグラフィック処理ユニットを製造させる、集積回路のコンピュ

10

20

30

40

50

ータ可読記述を格納した非一時的コンピュータ可読記憶媒体が提供され得る。

【0041】

グラフィック処理システムおよび／またはグラフィック処理ユニットを記述するコンピュータ可読集積回路記述を記憶した非一時的コンピュータ可読記憶媒体と、グラフィック処理システムおよび／またはグラフィック処理ユニットを具現化する集積回路の回路レイアウト記述を生成するために、集積回路記述を処理するように構成されたレイアウト処理システムと、回路レイアウト記述に従ってグラフィック処理システムおよび／またはグラフィック処理ユニットを製造するように構成された集積回路生成システムとを含む集積回路製造システムが提供され得る。

【0042】

本明細書に記載される方法を実施するためのコンピュータプログラムコードが提供され得る。コンピュータシステムで実行された時、コンピュータシステムに本明細書に記載される方法を実行させるコンピュータ可読命令を格納した非一時的コンピュータ可読記憶媒体が提供され得る。

【図面の簡単な説明】

【0043】

本発明は、添付図面を参照しながら例を説明する。図面では、

【図1】図1は、従来のデュアルロックステッププロセッサを示す。

【図2】図2は、車両用のコンピュータ生成機器クラスタを示す。

【図3】図3は、本明細書に記載の原理に従って構成されたグラフィック処理システムを示す。

【図4】図4は、本明細書に記載の原理に従って構成されたグラフィック処理システムの動作を図示した概略図である。

【図5】図5は、本明細書に記載の原理に従って構成されたグラフィック処理システムを含むデータ処理システムを示す。

【図6】図6は、本明細書に記載の原理に従って構成されたグラフィック処理システムの動作を示すフローチャートである。

【図7】図7は、集積回路製造システムの概略図である。

【発明を実施するための形態】

【0044】

以下の説明は、当業者が本発明を作成および使用することを可能にするために例として提示されている。本発明は、本明細書に記載される実施形態に限定されず、開示された実施形態に対するさまざまな修正は、当業者にとって明らかであろう。実施形態は、例としてのみ記述される。

【0045】

本開示は、シーンのタイルベースのレンダリングを実施するための一つまたは複数の処理ユニットを含むグラフィック処理システムに関連する。本明細書の原理に従って構成されたグラフィック処理システムは、任意のタイルベースのアーキテクチャを有してもよく、例えば、システムはタイルベースの伸長レンダリングを実施するよう動作可能であり得る。各処理ユニットは、任意の他の処理ユニットとは独立してタイルを処理することができる場合がある。処理ユニットは、例えば、任意の種類のグラフィックおよび／またはベクトルおよび／またはストリーム処理ユニットとし得る。各処理ユニットは、シーンのプリミティブの幾何学形状処理および／または断片処理を実施するためのレンダリングパイプラインを含みうる。各処理ユニットは、GPUの異なる物理コアであり得る。

【0046】

タイルは、グラフィック処理システムでレンダリングされるフレームの任意の部分であり得る。フレームは、画像またはビデオフレームの全てまたは一部であり得る。処理ユニットは、例えば、タイリング、幾何学形状処理、テクスチャマッピング、シェーディング、深さ処理、頂点処理、タイル加速、クリップ処理、間引処理、プリミティブアセンブリ、カラー処理、ステンシル処理、アンチエイリアシング、レイトレーシング、ピクセル化

10

20

30

40

50

、および切りばめ処理など、グラフィック処理の任意の態様の一部または全部を実施するように構成され得る。異なる処理ユニットは、レンダリングの異なる態様を実施するように構成され得る。一部の実施例では、処理ユニットの全ては同一であり得る。

【0047】

一般に、断片処理エンジンは、断片処理を集合的または個別に実施するよう配置されたグラフィック処理システムの部分を含む。断片処理エンジンは、グラフィック処理システムの特定のアーキテクチャに適切な任意の方法で配置された一つまたは複数の機能ユニットを含んでもよい。一般に、幾何学形状エンジンは、タイリングを含む幾何学形状処理を集合的または個別に実施するよう配置されたグラフィック処理システムの部分を含む。幾何学形状エンジンは、グラフィック処理システムの特定のアーキテクチャに適切な任意の方法で配置された一つまたは複数の機能ユニットを含んでもよい。グラフィック処理システムの部分は、その断片処理エンジンとその幾何学形状エンジンとの間で共有され得る。本明細書で説明し、図で説明した幾何学形状ユニット、タイリングエンジン、および処理ユニットの構成は、例として提供され、また当然のことながら、幾何学形状ユニット、タイリングエンジンおよび処理ユニットの異なる構成が他のアーキテクチャで採用され得る。

10

【0048】

本明細書に記載の原理に従って構成されたグラフィック処理システムは、任意の種類のシーンをレンダリングするように配置され得る。一部のアプリケーションでは、レンダリング処理のフォールトが識別されて適切なアクションが得られるように、シーンのレンダリングを保護する必要がある。例えば、自動車のコンテキストでは、グラフィック処理システムを使用して、ダッシュボード表示画面で表示するために機器クラスタをレンダリングし得る。機器クラスタは、車両の速度や車両のフォールトの詳細など、ドライバに重要情報を提供する。このような重要情報がドライバに確実に提示されることが重要であり、車両規則は通常、ISO 26262 標準のASIL Bなどの所定の安全レベルに従って重要情報がレンダリングされることを必要とする。機器クラスタはさらに、時間およびどのラジオステーションが選択されているかなどの重要情報を提供し得る。

20

【0049】

図2は、本明細書に記載の実施例で言及されるグラフィック処理システムにおけるレンダリングのための機器クラスタ200を図示する。機器クラスタは、ダイヤルの端の周りに速度値208と、角度方向が車両の現在の速度を示す針207とを含む従来のダイヤルの形態の速度計202を含む。機器クラスタは、油温ゲージ203、情報アイコン204（例えば、選択されたラジオステーションを示す）、非重要警告アイコン205（例えば、空調システムでのフォールトを示す）、および重要警告アイコン206（例えば、深刻なエンジン問題を示す）をさらに含む。

30

【0050】

表示要素の速度計202および重要警告アイコン206のみが、車両およびその使用者の安全に重要である。ISO 26262 標準のASIL Bなどの義務付けられた安全レベルを満足するように、これらの表示要素をレンダリングすることが必要である場合がある。油温ゲージ203、情報アイコン204、および非重要警告アイコン205は、その安全レベルにレンダリングする必要はない。一つまたは複数の表示要素202～208および／または表示要素への背景は、三次元レンダリングを必要とし得る。例えば、表示要素は、車両の位置および／または一つまたは複数の車両カメラからのビデオフィードを示すマップをオーバーラップすることができる。

40

【0051】

レンダリングされた機器クラスタを表すフレームをレンダリングするために使用されるレンダリングした空間は、それぞれが複数のピクセルを含む複数のタイル201に分割される。ハイライトされたタイル209のみが、重要表示要素の少なくとも一部がハイライトされたタイルのそれぞれとオーバーラップするという点で、重要表示要素を含む。ハイライトされたタイル209は、以下でより詳細に検討される保護されたタイルの例である。

50

【0052】

本明細書に記載の原理に従って構成されたグラフィック処理システムを、図3にグラフィック処理ユニット（GPU）300の形態で示す。GPU300は、図2に示す機器クラスタ200をレンダリングするのに適している。GPUの動作は、図6に示す例示的なフローチャートを参照して説明される。GPU300は、PU0～PU（n）として図で標識された複数の処理ユニット301を含む。図3には二つの処理ユニット303および304のみが示されているが、一般的にはより多くなる。一般に、GPU300は、タイルベースのレンダリングを実施するための適切な任意のアーキテクチャを持ちうる。図3に示す例では、GPUは、例えば、プロセスタイプおよびその他のデータなどのさまざまな機能要素を持つシステム部分302を含み、これはCPU305およびメモリ306などの外部装置と通信し、処理ユニット301で実行される処理を保存する。

10

【0053】

図3では、システム部分302は、処理ユニット301でタイルの三次元レンダリングを制御するための3Dデータマネージャ307と、タイルの二次元レンダリングを制御するための2Dデータマネージャ308とを含む。システム部分302はさらに、例えば、GPUの低レベル管理を実行してGPUに向けられた制御メッセージのためのインターフェースを提供し得るファームウェア309を含む。このようなユニットの動作は、GPUの特定のアーキテクチャによって異なる。データバス318は、GPUの要素およびその処理ユニットの通信を介して提供され得る。

【0054】

キャッシュ311は、処理ユニット301で処理するためのタイルの受け取り、および／または処理ユニットから処理されたデータを受信するためのGPUで提供され得る。一部の実施例では、キャッシュはメモリ306の内容をキャッシュし得る。メモリ306は、断片処理を実施する際に、その中に幾何学形状処理の出力（変換された幾何学形状およびタイルリストなど）が、その後の使用のために書かれうる、パラメータバッファ312を含みうる。

20

【0055】

複数の処理ユニット301の各々は、幾何学形状ユニット319および断片プロセッサ320を含んでもよい。一部の実施例では、幾何学形状ユニット319および断片プロセッサ320は、共通のハードウェア要素（例えば、算術論理単位（ALU）を共有してもよく、これは必要に応じて幾何学的または断片処理のいずれかの処理に割り当てられ得る。幾何学形状ユニットは、GPUに提出した601幾何学形状データを処理するよう構成されている。幾何学形状データは、レンダリングされるシーンの要素を表す。幾何学形状データは、例えば、レンダリングされるプリミティブ、シーンで幾何学的形状を説明する頂点データ、およびレンダリングされる物体のうちの一つまたは複数を含みうる。シーンの要素は、一つまたは複数のプリミティブ、一つまたは複数の幾何学的形状に関連する頂点データ（例えば、一つまたは複数の幾何学的形状の頂点の座標を識別する頂点データ）、および物体のうちの一つまたは複数とし得る。例えば、幾何学形状データは、図2に示す機器クラスタの表示要素を表す一組のプリミティブを含みうる。各プリミティブは、機器クラスタを生成するように配置されたソフトウェアプロセスからの適切なドローコールによって作成され得る。プリミティブは、物体またはシーンの他の部分が構成され得る基本的幾何学的形状であり得る。プリミティブは、例えば、三角形、線、または点であり得る。

30

【0056】

一部の実施例では、幾つかの態様の幾何学形状処理は、幾何学形状ユニット319以外のGPUの部分で実施され得る。例えば、幾何学形状処理は、断片プロセッサ320において部分的に実施されてもよく、例えば、頂点シェーディングはシェーダーコア321で実施され得る。一部の実施例では、タイリングは、以下に記述されるように、タイリングエンジン310で実施され得る。一部の実施例では、断片処理の一部の態様は、断片プロセッサ320以外のGPUの部分で実施され得る。

40

50

【0057】

幾何学形状データ（例えば、ソフトウェアプロセスからのドローコールに応答して生成される）は、メモリ306内に保持され、処理ユニットで処理するためのメモリからキャッシュ311に読み取られ得る。各処理ユニットは、その処理ユニットで処理されるデータを保持するための、および／または処理ユニットで生成された処理されたデータを受信するデータキャッシュ324を含んでもよい。処理ユニットで処理するための幾何学形状データは、その処理ユニットのデータキャッシュ324に読み取られてもよい。

【0058】

幾何学形状ユニット319は、レンダリングされるフレームの視点からのシーンを表す処理された幾何学形状データを形成するように、シーンの要素を説明する幾何学形状データを変換するよう構成されている。処理された幾何学形状データは、シーンのプリミティブを定義する変換された頂点データを含みうる。幾何学形状ユニットは、例えば、頂点処理（例えば、頂点シェーディング）、クリップ処理、プロジェクション、間引処理、およびタイリングなどを含む、幾何学形状データ上の任意の適切な処理を実施し得る。

【0059】

一般に、処理された幾何学形状データは、グラフィック処理システムのいかなる部分でも生成され得る。例えば、断片処理が実施される同一の処理ユニットの幾何学形状ユニット、断片処理が実施される処理ユニットに対し異なる処理ユニットの幾何学形状ユニット、またはその他に位置する幾何学形状ユニット（例えば、GPUのシステム部分312またはソフトウェア内で）である。

【0060】

幾何学形状ユニットは、グラフィック処理システムでシーンをレンダリングするプロセスの幾何学形状フェーズの少なくとも一部を実施する。幾何学形状フェーズはまた、タイリング（すなわち、シーンのどの要素がレンダリングされるフレームの各タイル内に落ちるかを特定する）を含む。タイリングは、各タイルに対して、そのタイル内に落ちる要素（例えば、プリミティブ）のリストを生成することを含みうる。

【0061】

図3に示す例では、タイリングはタイリングエンジン310で実施される。各タイリングエンジンは、どの要素が、レンダリングされるフレームの各タイルに含まれるかを識別するために、幾何学形状ユニット319から受信した処理された幾何学形状データ上で動作するように構成され得る。一部の実施例では、処理された幾何学形状データのタイリングは、幾何学形状ユニット319またはグラフィック処理システムの任意の他の部分において実施され得る。タイリングは、適切な任意のアルゴリズム（例えば、完全なタイリング、境界ボックスアプローチまたは階層タイリングアプローチを使用して）に従って実施され得る。多くのこうしたアルゴリズムは既知であり、本明細書でさらに考察されない。タイルは、タイルのエリアのいかなる部分とオーバーラップするために（例えば、要素がタイルのピクセルの全てまたは一部とオーバーラップする時に）、タイリングアルゴリズムによってその要素の任意の部分が計算されるシーンの要素を含むと考えられ得る。

【0062】

全体としてのシーンのための処理された幾何学形状データ（例えば、変換された頂点データ）を保存し得る。さらに、レンダリングされるフレームの各タイルについて、そのタイルとオーバーラップする要素（例えば、プリミティブ）のリストも保存し得る。このようにして、要素のリストは、シーンのための処理された幾何学形状データの変換された要素を意味し得る。（例えば、複数のタイルをオーバーラップするシーンの要素により）このような要素のための処理された幾何学形状データの保存をコピーするのを回避する。他の例では、各タイルの断片処理を実施するために必要な処理された幾何学形状データの一部または全てが、各タイルに対して別個に保存され得る。「タイル」は、レンダリングされるフレームの所与のタイルに関する幾何学形状フェーズの出力が断片処理のために処理ユニットに送信され得るという意味で、断片処理のための処理ユニットに提出され得る。

【0063】

幾何学形状フェーズ（例えば、幾何学形状ユニット 3 1 9 および／またはタイリングエンジン 3 1 0）からの処理されたデータは、断片処理フェーズでのその後の使用のための任意の適切な位置に保存され得る。例えば、幾何学形状処理（変換された頂点データおよびタイルリストなど）の出力は、メモリ 3 0 6（例えば、パラメータバッファ 3 1 2）に保存されてもよく、キャッシュ 3 1 1 を通してアクセスされ得る。一部の実施例では、幾何学形状フェーズからの処理されたデータは、データキャッシュ 3 2 4 などの処理ユニット 3 0 1 に保持され得る。

【0064】

幾何学形状処理は、レンダリングされる完全なフレームに関して実施され、タイルごとに実施されない。これは、幾何学的形状が処理されるまで、例えば、シーンの要素が、レンダリングされるフレームのタイルに関連して位置する場合、要素の見かけのサイズ、およびそれらの要素が見えるかどうか、知られていないことが理由である。

【0065】

一部の実施例では、幾何学形状ユニット 3 1 9 は、断片プロセッサ 3 2 0 に対して異なる処理ユニットで提供され得る。幾何学形状ユニットの数は、GPU の断片プロセッサの数とは異なってもよい。一部の実施例では、グラフィック処理システムは、下記の方法で幾何学形状処理を繰り返すように構成された単一の幾何学形状ユニットを含みうる。

【0066】

各断片プロセッサ 3 2 0 は、幾何学形状ユニット 3 1 9 によって生成された処理された幾何学形状データ上で、タイルベースの様式で断片処理を実施するように構成される。例えば、グラフィック処理システムの各断片プロセッサは、一度に一つのタイルの処理された幾何学形状データを受信するように配置され得る。GPU は、複数の断片プロセッサ（例えば、処理ユニット 3 0 3 および 3 0 4 における断片プロセッサ）を含むので、異なる断片プロセッサは、異なるタイルに対して断片処理を同時に実施するように配置され得る。所与のタイルの幾何学形状フェーズからの処理されたデータは、任意の適切な形態で提供され得る。例えば、所与のタイルの幾何学形状フェーズからの処理されたデータは、そのタイルの要素のリスト、およびそれらの各要素を説明する変換された幾何学形状データ、またはそのタイルの要素のリストで識別される要素のそれぞれに対する変換された幾何学形状データを含みうる。

【0067】

タイルは、任意の適切な方法で断片処理のための処理ユニットに割り当てられ得る。例えば、複数の処理ユニットの断片プロセッサは、一度に一つのタイルについて、各処理ユニットがキャッシュ 3 1 1 でパラメータバッファ 3 1 2 からタイルのために処理された幾何学形状データをそのデータキャッシュ 3 2 4 に読み込むように配置することによって、フレームのタイルを介して集合的に作業するように構成され得る。

【0068】

断片プロセッサは、タイルの処理された幾何学形状データ上の任意の適切な処理を実施するように構成され得る。例えば、断片プロセッサは、テクスチャ処理、シェーダー処理、ラスタライゼーション、隠面消去、およびアルファ処理のうち一つまたは複数を実施し得る。断片プロセッサは、断片処理の異なる態様を実施するための一つまたは複数のユニット（例えば、パイプライン内に配置される）を含みうる。例えば、図 3 に示すように、断片プロセッサは、シェーダー処理および隠面消去を行うためのシェーダーコア 3 2 1、ラスタライゼーションを実施するためのラスタライザ 3 2 2、およびテクスチャ処理を実施するためのテクスチャユニット 3 2 3 を含んでもよい。複数の所与のタイプのユニットが、断片プロセッサパイプラインで提供されてもよく、例えば、図 3 に示す例では、二つのシェーダーコアが提供されている。シェーダー処理は一般的に、断片処理の実質的な割合を表し、複数のシェーダーコア上に容易に分布することができる。

【0069】

各断片プロセッサが提供される処理ユニットは、断片プロセッサによって実行される処理に関連付けられたデータを保持するためのデータキャッシュ 3 2 4 を含みうる。断片処

10

20

30

40

50

理中に実施される計算の出力は、一つまたは複数のバッファに書き込まれうる。例えば、（例えばピクセルの）カラー値を保存するためのカラーバッファ、（例えば、ピクセル）を保存するための深さバッファ、およびどのタイルの部分（例えば、ピクセル）がレンダリングされるかについての指示を保存するためのステンシルバッファの一つまたは複数である。こうしたバッファは、システムメモリ306、GPUキャッシュ311、および処理ユニットデータキャッシュ324のうちの一つまたは複数を含む、GPUアーキテクチャに適した任意の方法で維持され得る。こうしたバッファの使用は当技術分野で周知であり、ここで詳細には考察されない。

【0070】

図3に図示したタイルベースの遅延レンダリングアーキテクチャは、単に例である。一般的に、処理ユニットおよびGPUは、任意のタイルベースのレンダリングアーキテクチャを持ちうる。

【0071】

本開示の原理に従って構成されたグラフィック処理システムは、図2の機器クラスタなどの安全性重要表示要素を含むフレームをレンダリングするように動作可能である。これは、図3に示すGPUを参照しながら、例により説明する。

【0072】

幾何学形状ユニット319での処理のために受け取られた幾何学形状データは、安全性が重要である要素（例えば、レンダリングされるシーンの物体、プリミティブまたはその他の部分）を含みうる。安全重要要素は、任意の適切な方法で、幾何学形状データで識別され得る。例えば、シーンの物体を構成する一つまたは複数のプリミティブを幾何学的に記述する頂点データは、それらのプリミティブが安全性で需要であり、それに応じてレンダリングされるかどうかを指示する識別子を含む。

【0073】

データ処理システムにおけるGPUの典型的な配置を図5に図示しており、GPU 300がハードウェア502においてメモリ306およびCPU305とともに示されている。GPU、CPUおよびメモリは、データバス503上で通信するように配置され得る。データ処理システムは、一つまたは複数のプロセス505と、GPUのためのドライバ504とを含むソフトウェア環境501を含む。ドライバは、プロセス505からドローコールを受信するように配置されたアプリケーションプログラミングインターフェース（API）506を提供し、GPUによってシーンを表示画面507にレンダリングする。例えば、APIは、OpenGL APIであってもよく、またプロセスは、GPUが機器クラスタを車両のダッシュボードでの表示画面に機器クラスタをレンダリングさせるように、OpenGLドローコールを発行するように配置された、図2に示される、計測クラスタのソフトウェアコントローラであり得る。

【0074】

レンダリングされるシーンを定義する幾何学形状データのソースは、シーンのどの要素が安全性で重要であるかを特定できる。例えば、プロセスが、ドローコールまたは一つまたは複数のドローコールの群が安全重要要素に関連することを識別することを可能にするためにメカニズムを提供することができる。OpenGLフレームワークでは、例えば、APIは、glEnable()を呼び、およびglDisable()が、ドローコールが安全重要要素に関連するかどうかを示すドローコールの状態ビットを設定するために使用し得る。適切なメカニズムは、ドライバに提供され得る（例えば、プロセスによって呼び出され得るドライバの機能として）、またはより一般的にはソフトウェア、ファームウェアおよびハードウェアの任意の組み合わせで提供され得る。こうしたメカニズムは、アプリケーション開発者が、シーンのどの部分が安全性で重要性であるか、および、現在記述されている方法で、レンダリングされるフレームのどのタイルが保護されるかを制御することを可能にすることができる。

【0075】

一般的に、安全重要要素は、任意の適切な方法で識別され得る。例えば、それらの要素

10

20

30

40

50

(例えば、一つまたは複数のフラグとして、またはデータを含む制御メッセージとして)を定義する幾何学形状データを有する一つまたは複数の識別子を提供することにより、および／または幾何学形状データのどの要素が安全上に重要でありおよび／または幾何学形状データのどの要素が安全上に重要でないかを識別するデータ構造(例えば、ルックアップテーブル)によって、識別され得る。

【0076】

本明細書に記載の原理に従って構成されたグラフィック処理システムは、レンダリングされるフレームのどのタイルが安全重要要素を含むかを、幾何学形状フェーズで特定するように構成されている。こうしたタイルは、保護されたタイルと呼ばれうる。保護されたタイルは、所定の安全レベルを満たすためにグラフィック処理システムで処理され得る。保護されたタイルの識別は、タイリングの間に実施され得る。例えば、所与のタイルとオーバーラップする要素を識別する上で、それらの要素のいずれかが安全重要要素である場合、そのタイルは保護されたタイルであると識別され得る。保護されたタイルは、グラフィック処理システム内で任意の適切な方法でマークされ得る。例えば、保護されたタイルは、幾何学形状フェーズ(例えば、そのタイルに関連する要素のリストまたはリストに関連するヘッダー構造において)によって出力されるデータに設定される識別子(例えばフラグ)、タイルの幾何学形状データと関連付けられた制御ストリーム内のパラメータ、およびフレームの各タイルが保護されるかどうかを識別するデータ構造(例えば、ルックアップテーブル)の一つまたは複数によって識別され得る。

【0077】

どのタイルが安全重要要素を含むかを、幾何学形状フェーズの中で識別し、かつ断片処理のための処理された幾何学形状データを有する情報を提供することによって、本明細書に記載の原理にしたがって構成された、グラフィック処理システムは、安全重要要素を含む保護されたタイルおよび安全重要要素を含まない、故に、保護されたレンダリングを必要としない保護されていないタイルの両方を処理するように動作可能である。システムの各処理ユニットは、任意の他の処理ユニットと独立して任意のタスクを処理するように配置されてもよく、異なる処理ユニットが異なるタスクを処理することを可能にする。一部の実施例では、保護されたタイルおよび保護されていないタイルの両方の断片処理は、システムの異なる処理ユニットで同時に処理され得る。一部の実施例では、保護されたタイルおよび保護されていないタイルの両方の断片処理は、同一の処理ユニットで実施され得る。

【0078】

本明細書に記載される原理に従って構成されたグラフィック処理システムによって実施される保護されたレンダリングは、例によりGPU300によって実施されるレンダリングを概略的に図示する図4を参照しながら、および例によりGPUの動作を図示した図6に示すフローチャートにより、詳細に説明する。図4(a)は、幾何学形状フェーズを示し、図4(b)は、断片処理フェーズを示し、図4(c)は、幾何学形状処理または断片処理中にフォールトが発生しない場合のレンダリング処理の出力として形成されたレンダリングされたタイルを示す。

【0079】

冗長化断片処理

グラフィック処理システムは、各保護されたタイルに少なくとも二回の断片処理を行い、それらのパスの出力を比較してそれらが一致することをチェックするように構成される。二つの断片処理パスの出力が一致しない場合、フォールト信号が上昇してフォールトが発生したことを示す。こうした冗長化処理を実施することにより、本明細書に記載の原理に従って構成されたグラフィック処理システムは、保護されたタイルの処理に関して必要な安全レベルを満たすように配置され得る。保護されたタイルの各処理パスを実行するように構成された処理ユニットは、特定の実装に応じて、同一の処理ユニットであってもよく、または同一の処理ユニットでなくてもよい。所与の保護されたタイルの各処理パスは、一つまたは複数の他のパスと同時に処理されてもよく、または処理されなくてもよい。

【0080】

適切な任意のアプローチを使用して、タイルを断片処理のための処理ユニットに分配し、保護されたタイルに対して少なくとも二回の断片処理が実施されるように配置され得る。一般に、GPUは、任意の種類のアーキテクチャを持つことができ、またタイルを処理ユニットに分配するための任意のスキームを利用し得る。断片処理は、保護されたタイルとして識別されないタイル（すなわち、保護されていないタイル）にたいし、単一の処理ユニットで一度実施される。

【0081】

一部の実施例では、GPUは、その処理ユニットのうち一つまたは複数をミッション処理ユニットとして指定するように、およびその処理ユニットのうちの一つまたは複数を安全処理ユニットとして指定するように構成され得る。各々のミッション処理ユニットは、保護されたタイルがミッション処理ユニットおよびその個々の安全処理ユニットに提出されるように、個々の安全処理ユニットを有し得る。グラフィック処理ユニットは、例えば、GPUの初期化の際（例えば、GPU用ソフトウェアドライバがこのような指定を実施する）、および／または動的に実行時に一つまたは複数の新しいフレームおよび／またはタイルをレンダリングするように配置されることに応答して、ミッション処理ユニットおよび安全処理ユニットを指定するように構成され得る。ミッション処理ユニット数は、GPUで指定された安全処理ユニットの数とは異なりうる。図3および4に示す実施例では、特定のタイルの断片処理を実施する目的で、処理ユニット303は、ミッション処理ユニットに指定され、および処理ユニット304は安全処理ユニットに指定される。

【0082】

幾何学形状処理フェーズが完了すると、タイルは断片処理のために処理ユニットに送信され得る。保護されたタイル（例えば、一つまたは複数の安全重要要素を含む保護されたタイルであることを示す識別子セットを有するタイル）は、処理のためのミッションおよび安全処理ユニット（例えば、処理ユニット303および304の両方）に提出され得る。図6の612、614を参照。レギュラータイル（例えば、安全重要要素を含まないために識別子セットを持たないタイル）を処理ユニットのいずれかに提出することができる（例えば、処理ユニット303および304のいずれか）。

【0083】

タイルは、GPUの構成要素により断片処理のための処理ユニットに割り当てられ得る。例えば、3D負荷について、3Dデータマネージャ307は、タイルを処理ユニットに割り当てるように構成されてもよく、保護されたタイルに遭遇すると、3Dデータマネージャは、そのタイルを断片処理のための二つの処理ユニット（例えば、ミッション処理ユニットまたは安全処理ユニットの両方）に割り当ててもよく、保護されたと識別されないタイルに遭遇すると、3Dデータマネージャは、そのタイルを断片処理のための任意の一つの処理ユニット（例えば、ミッション処理ユニットまたは安全処理ユニットのいずれか）に割り当ててもよい。一部の実施例では、タイルを構成する処理された幾何学形状データ（例えば、変換された頂点データ）は、パラメータバッファ312から読み取られてもよい。

【0084】

一部の実施例では、GPUは、処理ユニットがそれらのタイルを処理する能力に基づいて、断片処理のためにタイルを処理ユニットに分配するように構成され得る。例えば、各処理ユニットは、処理ユニットが処理のために新しいタイルを受け取ることができる時に（例えば、その現在のタイルの処理の完了時）、パラメータバッファ312から処理のためにタイルを取り出し得る。これは、幾何学形状データが、断片処理のため、幾何学形状フェーズから強調表示された断片プロセッサ320へ提供されることを表す、図4の矢印401によって図示されている。一部の実施例では、タイルは、複数のユニットの任意の処理ユニットで初めて処理され、複数のユニットの任意の処理ユニットで二度目に（またはさらに）処理され得る。これは、処理負荷を効率的に広げ、処理ユニットが不必要にアイドル状態になる状況を回避するように、タイル（初めてまたはさらなるパス上で）を処理ユニットに動的に割り当てることができることを意味する。

【0085】

一部の実施例では、グラフィック処理システムは、保護されたタイルおよび保護されていないタイルが同一の処理ユニットで処理できないように構成され得る。これは、保護されていないタイルを処理するときが発生するフォールトが保護されたタイルで実行された処理を破損または停止するのを回避するのに役立つ。例えば、グラフィック処理システムは、保護されたタイルの断片処理を実施するために指定された処理ユニットが、保護されていないタイルの断片処理を行うことを禁止し得るように構成され得る。一部の実施例において、グラフィック処理システムにおける処理ユニットの合計数のサブセットは、保護されたタイルの処理用に指定されてもよく、またグラフィック処理ユニットの異なるサブセットは、保護されていないタイルの処理用に指定され得る。

10

【0086】

保護されたタイルは、例えば、図4に図示した様式で断片処理のための異なる処理ユニットに同時に送信され得る。こうした配置は、メモリーテクスチャからの読み取りに消費される読み取り帯域幅および断片処理中に必要とされ得る他のデータの減少を助けることができる。こうした配置はまた、所与の処理ユニット固有の永久フォールトに対する保護を提供することができる一方で、異なる処理ユニットが同一の断片処理計算を実行する時間における自然な変動による一時的フォールトに対する保護を維持することができる。これは、こうしたデータが両方の断片処理パスに対してGPUのキャッシュ（例えば、キャッシュ311）で利用可能である可能性が高いため、メモリ（例えば、メモリ306）からそのデータを再フェッチする必要性を回避することができるためである。一般的に、異なる処理ユニットで実行される断片処理は、同期化される必要はなく、後続の処理パスが行われた時に、必要なデータがキャッシュでまだ使用可能である時間に、タイルの異なる断片処理パスが密接に起こることによって十分である。必要なデータが利用できる可能性は、キャッシュのサイズの適切な選択によって制御され得る。

20

【0087】

保護されたタイルでの断片処理を実行するように構成された各処理ユニットからの出力は、それらの出力が一致するかどうかをチェックするために比較される。所与の保護されたタイルを処理するように配置された処理ユニットからチェックユニットに提供される出力データは、例えば、一つまたは複数のカラーバッファ、ステンシルバッファおよび深さバッファなどの、一つまたは複数のバッファへの書き込みを含みうる。一部の実施例では、チェックユニットに提供される出力データは、タイルの中間または最終的にレンダリングされた出力であり得る。保護されたタイルに関して実施された断片処理の完全な出力をチェックすることで、断片処理中に発生するフォールトを捕捉する可能性が最大化され、GPUが高い安全レベルを達成するのに役立つ。他の例では、チェックユニットは、フレーム内の全ての保護されたタイルに対する断片処理の出力に関連するデータを蓄積し、タイルごとではなく、フレームごとに比較チェックを実行するように構成され得る。

30

【0088】

一部の実施例では、断片プロセッサ320は、断片処理計算の出力が記述される一つまたは複数のバッファを維持するように配置される。例えば、断片プロセッサは、カラーバッファ、ステンシルバッファ、および深さバッファのうちの一つまたは複数を持続し得る。こうしたバッファは、断片処理が実施される処理ユニットのデータキャッシュ324に保持され得る。一部の実施例では、断片処理フェーズでの処理ユニットの出力は、その個々のバッファへの書き込みである。

40

【0089】

図3では、処理ユニットの出力を適切に指示するためにフィルタ313が提供されている。フィルタは、図3に示す例では、GPUのさまざまなユニットが通信するデータバス318に結合されている。図に示すフィルタは概略的であり、GPUの周りのデータの流れは、GPUのアーキテクチャに応じて多くの異なる方法で実施され得ることが理解されよう。一般に、GPUの周りのデータの流れは、任意の適切な方法で、およびGPUの任意の一つまたは複数のユニットによって実施され得る。

50

【0090】

保護されたタイルに関して処理ユニットから出力データを受信すると、フィルタはその出力データをチェックユニット314に向けるように構成される。チェックユニットは、同一の保護されたタイルの異なる断片処理パスからの出力データを比較するように構成されている。出力データは、同一または異なる処理ユニットからのものとし得る。一部の実施例では、所与の保護されたタイルに関連して形成された出力データは、上述の方法で指定されたミッションおよび安全処理ユニットから受信される。一部の実施例では、タイル上の断片処理を実施するように配置された処理ユニットからの出力データは、そのバッファ（例えば、カラーおよび／またはステンシルおよび／または深さバッファ）の一つまたは複数への書き込みのセットを含む。

10

【0091】

断片処理フェーズの出力は、フレームバッファへの書き込みに適切なレンダリングされたタイルであってもよく、一つまたは複数のさらなる処理ステップ（例えば、シームレスな出力フレームを生成するために、タイルをステッチおよび／またはブレンドする）の潜在的な対象となる。図2の機器クラスタ200をレンダリングするために適用された時、保護されたレンダリングされたタイル403の図4（c）には、GPU300に関して断片処理フェーズによって出力される例が示されている。レンダリングされたタイルは、メモリ306（例えば、フレームバッファに）に書き込まれてもよい。

【0092】

一部の実施例では、チェックユニット314は、同一の保護されたタイルの異なる断片処理パスから受信した出力データを直接比較するように構成されており、例えば、チェックユニットは、異なるパスから出力データ（例えば、バッファへの書き込み）をストリームし、データ（ビット、バイト、ワード、または任意の他の適切なレベル）を直接比較し得る。出力ストリームの比較は、チェックユニット314のフォールト検出ユニット316で実施され得る。チェックユニットが、チェックユニットで受信された出力データストリームの間の時間オフセットを補正するために（例えば、一つの処理ユニットで断片処理が、別の処理ユニットでの断片処理の前にあることにより、または、単一の処理ユニットが一度目および二度目に、保護されたタイルを逐次的に処理するように構成されていることにより）、一つまたは複数のバッファが、チェックユニットにまたはGPUの他の場所に提供され得る。

20

30

【0093】

一部の実施例では、チェックユニット314は、各断片処理パスに対して、個々の処理ユニットから受信した出力データの特徴である一つまたは複数のシグニチャを形成するように構成される。チェックユニットは、任意の適切な方法でこうしたシグニチャを形成するように構成され得る。一部の実施例では、チェックユニットは、その処理ユニットでの断片処理が完了したら、処理ユニットからの完全な出力に対するシグニチャを形成するように構成される。

【0094】

一部の実施例では、チェックユニットは、保護されたタイルの断片処理を実施するように配置された処理ユニットからの出力データに対するシグニチャを形成するように構成される。例えば、チェックユニットは、バッファに断片プロセッサによる一つまたは複数の書き込みのセットに対するシグニチャを形成するように構成されてもよく、図6の613および615を参照のこと。例えば、断片プロセッサがそのカラー、深さ、またはステンシルバッファのうちの一つに書き込むたびに、その書き込みに関してシグニチャを生成するチェックユニットに対し、書き込みが、チェックユニット314に（フィルタ313によって）パスされ得る。チェックユニットがバッチの書き込みにシグニチャを生成するように構成された状態で、一つまたは複数の書き込みが、チェックユニットへの提出のために一緒にバッチ処理され得る。断片プロセッサにより、バッファへの書き込みに形成されたシグニチャは、データストア315に蓄積され得る。シグニチャは、シグニチャが関連する保護されたタイル（例えば、識別子またはタイルの座標）、およびシグニチャが関連

40

50

するバッファを示す情報を有するデータストアで関連付けられてもよい。（例えば、カラー、ステンシル、および深さバッファのうち、書き込みが、シグニチャが形成されることについて、指示されるバッファ）。断片処理中に書き込まれたバッファの内容は、個々の処理ユニットによって実施される断片処理の提示であるため、断片処理中に作製されたバッファへの書き込みに形成されるシグニチャの全体セットは、その処理ユニットからの完全な処理された出力の提示である。

【0095】

チェックユニットは、各パスで実施された断片処理が一致していることを確認するために、タイルに対する各断片処理パスに関連して形成されたシグニチャを比較する616ように構成されている。処理パス用に生成されたシグニチャは、後続の使用のためにデータストア315に保存され得る（例えば、同一の保護されたタイルに対して別のパスのシグニチャと比較される）。フォールト検出ユニット316は、例えば、データストア315に保持されたシグニチャを比較することによって、シグニチャの比較を実行するように構成され得る。チェックユニットは、例えば、チェックサム、CRC、ハッシュ、または出力データ上の指紋計算を実行し得る。シグニチャは、異なる断片処理パスに提供される出力データの比較を促進するために、処理ユニットからの出力データよりもコンパクトな形態で処理ユニットによって保護されたタイルで実行される処理の表示を提供する。例えば、シグニチャは、断片処理の間にバッファに書き込まれるデータのコンパクトな表現を提供し得る。チェックユニット314のフォールト検出ユニット316は、所与の保護されたタイルに関して、チェックユニットで形成されたシグニチャを比較するように構成される。シグニチャの比較は、任意の適切なレベルの粒度（例えば、ビット、バイト、ワードレベル）で実行され得る。

【0096】

フォールト検出ユニット316は、断片処理が一对のパスの少なくとも一つに対して完了する前に、所与の一对の断片処理パスに関連して形成されたシグニチャデータの比較を開始するように構成され得る。これにより、フォールト信号をできるだけ早く上げることができ、例えば、個々の断片処理パスが放棄され（および潜在的に再開される）、GPUでの浪費された処理を最小化することができる。各シグニチャがバッファへの一つまたは複数の書き込みに関して形成される実施例では、フォールト検出ユニット316は、同一の保護されたタイル（および潜在的に同一のバッファ）の異なる処理パスからデータストア315に格納された対応するシグニチャを比較するように配置され得る。こうした処理パスは、GPUの一对の処理ユニット（例えば、ミッションおよび安全処理ユニット）で同時に実施され得る。シグニチャは、フォールト検出ユニットで対応するバッファへの書き込み操作のシグニチャの比較を容易にするために、保護されたタイルの各処理パスに対して、データストア315で順次保存され得る。

【0097】

一部の実施例では、チェックユニットは、フレームの保護された全てのタイルの処理が完了したら、同一の保護されたタイルの異なる断片処理パスに関し処理ユニットからの出力を比較するように配置され得る。例えば、チェックユニットは、本明細書に記載されるアプローチのいずれかに従い、断片処理されたデータに対するシグニチャを生成し、かつデータストア315でそれらのシグニチャを蓄積するように構成され得る。全てのシグニチャがフレームの保護されたタイルの全てに対して利用可能になったら、チェックユニット314は、そのフォールト検出ユニット316で対応するシグニチャの対の間で比較を実行し、もしシグニチャのいずれかが一致しない場合、全体としてフレームに関してフォールトを上げるように構成され得る。

【0098】

チェックユニットは、複数のチェックユニットインスタンスを含んでもよく、例えば、各チェックユニットインスタンスは、GPUの処理ユニットの異なるサブセットから出力データを受信するように構成され得る。チェックユニットは、処理ユニットからの出力に含まれるメモリアドレスおよび／または制御データ上で追加的に動作することが好ましく

、これによって本明細書に記述した検証操作は、より幅広いフォールトを識別するのに役立つ。

【0099】

フォールト検出ユニット316は、同一の保護されたタイルの異なるパスで実行される断片処理が一致しないと判定するとき618、フォールト信号317を上げる619ように構成されている。説明したように、こうした比較は、例えば、断片処理された出力を直接比較すること、および／または断片処理された出力の特徴的なシグニチャを比較することによってもよい。フォールトにより、GPUでの安全違反につながる可能性がある。例えば、図2に示す機器クラスタの保護されたタイル209のいずれかをレンダリングするときのフォールトは、速度計などの機器クラスタの安全重要要素の破損を引き起こす可能性がある。

10

【0100】

フォールト信号317は、任意の適切な方法で提供され得る。フォールト信号は、GPU300の出力として提供され得る。フォールト信号は、例えば、制御データ、メモリ306に書き出されたデータ、およびGPUまたはGPUが接続されているシステムのレジスタまたはメモリに書き込まれたデータのうちの一つまたは複数とし得る。フォールト信号は、同一の保護されたタイルの異なるパスからの処理された出力が同一ではなく、処理された出力を生成した処理ユニットの一つまたは複数からの処理された出力が無効であることを示す。同一の保護されたタイルを処理するように構成された処理ユニットの出力は、電離放射線または電圧スパイクなどの一時的なイベントによって、またはハードウェア、ソフトウェアおよびファームウェアの組み合わせにおけるバグによる永久エラーによって異なりうる。

20

【0101】

フォールト検出ユニット316は、同一の保護されたタイルの異なるパス時に生成された出力データを比較するためである。二つ以上の処理ユニットは、同一の保護されたタイルを処理するように配置され得る。異なるパスで実行される処理は、同時に行われてもよく、または同時でなくてもよい。二つの処理パスが異なる二つの処理ユニットで実行される場合、これら処理ユニットによって生成される出力データ比較は、処理ユニットの対によって実行される処理が一致しているかどうかを示す。フォールト信号は、処理ユニットの一つでフォールトが発生したことを示しているが、どのユニットがフォールトを経験したかを示していない。

30

【0102】

三つ以上の処理ユニットのグループが、同一の保護されたタイルを処理するように配置されている場合、それらの処理ユニットからの出力データを比較することは、処理ユニットによって実行される処理が一致しているかどうかを示す。処理ユニットのグループの一つの処理された出力が他の二つの処理ユニットの出力と一致しない場合、フォールト信号は、グループの処理ユニットの一つでフォールトが発生したことを示し、グループの処理ユニットのどちらで、フォールトが発生したかをさらに示すことができる。これは、フォールトがその出力が二つ以上の他の処理ユニットからの出力と一致しない処理ユニットで発生したと仮定できるためである。

40

【0103】

同一のタイルの異なる断片処理パスからの出力が一致する場合、グラフィック処理システムで生成されたレンダリングされたタイルは、有効とみなされ、適用に適した方法で使用され得る（例えば、図2の例で、レンダリングされたタイルデータは計測コンソール200の一部とし得る）。同一のタイルの異なる断片処理パスからの出力が一致しない場合、フォールト信号が上げられない場合があるか、一部の実施例では、出力が一致することを示す信号が上げられ得る。

【0104】

フォールト信号は、例えば、制御メッセージ、割込み、制御データのフラグ、レジスタで設定された一つまたは複数のビット、およびデータパケットの一つまたは複数を含む、

50

フォールトが発生したことの任意の種類が表示であり得る。

【0105】

グラフィック処理システムは、各保護されたタイルに対して処理ユニットで生成された処理された出力の一つまたは複数を外部メモリ306（例えば、フレームバッファ）に書き出すように構成され得る。保護されたタイルに関連して生成された出力は、消費されたメモリ帯域幅を最小化するために、そのタイルに対して実行される処理パスの一つのみから記述され得る。一部の実施例では、フィルタ313は、ミッション処理ユニット（例えば、カラー、ステンシルおよび深さバッファに書き込む）で発生した処理された出力をメモリ617に書き出し、安全処理ユニットで生成された処理された出力をメモリに書き出さないように構成される。これは、図4に図示されており、矢印402は、レンダリングされた出力タイル403を形成するように、メモリ306へのデータを書き込むミッション処理ユニット303を表す。

10

【0106】

一部の実施例では、フィルタ313は、所与の保護されたタイルに関して、処理ユニットから第一の利用可能な処理された出力（例えば、カラー、ステンシルまたは深さバッファへの書き込み）を、メモリに書き出すように構成される。このようにして、GPUが位置するデータ処理システムに対して、レンダリングされた出力がより早く利用できるようになる。一般に、所与の保護されたタイルを処理するように配置された処理ユニットからの処理されたタイル出力データの全てまたは一つまたは複数が、メモリ306に書き出され得る。

20

【0107】

フォールト検出ユニットが、外部メモリ（または外部メモリへの書き込みのキャッシュ）に書き出されたレンダリングされたタイルが無効であると判断した場合、グラフィック処理システムは、レンダリングされたタイルが無効として廃棄されるように、および／またはマークされるように構成され得る。

【0108】

特定の保護されたタイルに対して断片処理が完了したら、レンダリングされたタイル出力（例えば、メモリ306で）は、適用に適切のように使用することができる620。GPU300が機器クラスタ200をレンダリングするように構成された実施例では、レンダリングされたタイルは、機器クラスタが提供される物理的表示器に書き込まれる。

30

【0109】

保護されたレンダリングを実施するための一つのアプローチは、シーン全体を二回レンダリングし、例えば、図1に図示したデュアルロックステップ構成に配置された一対のGPUで実施され得るように、レンダリングされた出力が一致することをチェックすることである。しかしながら、GPUの多くのアプリケーションでは、シーンの全ての要素が安全性で重要であるとは限らない。例えば、図2では、レンダリングされた機器クラスタを表すタイルの約40%のみが、安全重要要素を含む。本開示は、保護されたレンダリングのための効率的なアプローチを提供し、その中で断片処理は、安全重要要素を含む幾何形状フェーズで識別された保護されたタイルのみに対して重複している。断片処理は典型的には、GPUのレンダリング性能を支配するため、本アプローチは、処理がシーンの安全重要要素に関係しない場合でも、両方の処理コアが必ず同一のレンダリング動作を実行しなければならない、デュアルロックステッププロセッサなどの従来のシステム上で実質的なパフォーマンス利益を提供できる。ロックステッププロセッサは、その二つの（または複数の）コアが一緒にロックされ、個別に利用されることができず、処理された全てのフレームに対して柔軟性のない100%のチップエリアコストをもたらす問題を抱えている。

40

【0110】

冗長化幾何学形状処理

幾何学的形状計算におけるエラーは、シーンの非安全重要要素に関連するものでも、フレームの保護されたタイルに書き込まれる要素がレンダリングされるのを生じさせ得る。

50

例えば、図2を参照すると、情報アイコン204に関連する幾何学的形状計算のエラーは、保護されたタイル209の一つまたは複数にわたってアイコンが誤ってレンダリングされ、機器クラスタの安全重要要素を不明瞭することになりうる。従って、任意の安全重要要素を含むフレームをレンダリングする幾何学形状フェーズは、グラフィック処理システムで少なくとも二回実施され、各パスの出力は、幾何学的形状計算が一致することをチェックするように比較される。幾何学形状処理パスの出力が一致しない場合、フォールト信号が上がり、フォールトが発生したことを示す。こうした冗長化幾何学的形状計算は、上述のいずれかのアプローチに従って、冗長的に断片処理を行うために実施され得る。冗長化処理を実施することによって、本明細書に記載の原理に従って構成されたグラフィック処理システムは、保護されたタイルの処理に関して必要な安全レベルを満たすように配置され得る。

10

【0111】

各幾何学形状処理パスを実行するように構成された幾何学形状ユニットは、特定の実装に応じて、同一の幾何学形状ユニットであってもなくてもよい。例えば、幾何学形状ユニット319は、幾何学形状データの処理を繰り返し得る。これは、許容される。なぜなら、典型的には断片処理は、要求される処理時間に関して、幾何学形状処理を支配し、一部のアーキテクチャでは、これは、GPUでデータ転送を減少させることができる。例えば、処理ユニット301において、幾何学形状ユニット319で処理される幾何学形状データは、第二のパスで、再使用のため、データキャッシュ324で維持され、故に、二度目に、キャッシュ311からの幾何学形状データの読み取りを回避することができる。しかし、同一のロジックで幾何学形状処理を繰り返すことが、単に第二のパスで繰り返される永久エラーを示さない。

20

【0112】

しかしながら、上述のように、フレームに対する異なる幾何学形状処理パスは異なる幾何学形状ユニットで行うことができる。こうした配置は、特定の幾何学形状ユニットに固有の永久フォールトに対する保護を提供できる一方、異なる幾何学形状ユニットが同一の幾何学的形状計算を実行する時間における自然な変動による一時的フォールトに対する保護を維持することができる。所与のフレームに対する各幾何学形状処理パスは、そのフレームの一つまたは複数の他のパスと同時に処理され得るし、処理されなくてもよい。

【0113】

任意の適切なアプローチは、幾何学形状データを処理のための幾何学形状ユニットに分配するために使用され得る。一般に、GPUは、任意の種類のアーキテクチャを有してもよく、幾何学形状データを幾何学形状ユニットに分配するための任意のスキームを利用し得る。シーンが安全重要要素を含むかどうかについてのチェックを実施することができる。こうしたチェックは、グラフィック処理システムの任意の場所で実施され得る。例えば、ドライバは、フレームに関するドローコールのいずれかが、要素を、安全上需要であるとして識別するかどうかを判断し得る。そのため、識別子は、グラフィック処理システムが、幾何学形状処理を繰り返す保護モードでフレームを処理することを示す、フレーム全体に設定され得る。他の例では、フレームが安全重要要素を含むことを識別することはアプリケーション開発者の責任であってもよく、そのため幾何学形状処理が繰り返される。例えば、グラフィック処理システムを、幾何学形状処理が繰り返され、幾何学形状処理パスの出力が比較される保護モードに入らせるために、グラフィックAPI、および／またはGPUドライバに対する一つまたは複数のコールによって処理を構成することができる。

30

40

【0114】

フレームに安全重要要素が含まれない場合、幾何学形状処理は、グラフィック処理システムの一つまたは複数の幾何学形状ユニットにおいて、一度だけ、従来の方法で実施され得る。

【0115】

一部の実施例では、グラフィック処理システムは、一つまたは複数のミッション幾何学形状ユニットおよび一つまたは複数の安全幾何学形状ユニットを指定するように構成され

50

、幾何学形状データは、ミッションおよび安全幾何学形状ユニットに提出され得る。ミッション幾何学形状ユニットは、ミッション処理ユニットで指定されてもよく、また安全幾何学形状ユニットは、断片処理を実施するための上記の例に記載された安全処理ユニットで指定され得る。一部の実施例では、グラフィック処理システムの一つまたは複数の幾何学形状ユニットは、複数の処理ユニットでは提供されない。幾何学形状ユニットが処理ユニットに提供される実施例では、例えば、図3に示す例では、ミッション幾何学形状ユニットは、指定されたミッション処理ユニットで幾何学形状ユニットであってもよく、および安全幾何学形状ユニットは、指定された安全処理ユニットで幾何学形状ユニットであり得る。こうしたミッション602および安全603の幾何学形状ユニットにおける幾何学形状データの処理を図6に示す。

【0116】

各々のミッション幾何学形状ユニットは、安全重要要素を含むフレームの幾何学形状データがミッション幾何学形状ユニットおよび個々の安全幾何学形状ユニットに提出されるように、個々の安全幾何学形状ユニットを持ちうる。グラフィック処理ユニットは、例えば、GPUの初期化の際（例えば、GPU用ソフトウェアドライバがこのような指定を実施する）、および／または動的に実行時に一つまたは複数の新しいフレームをレンダリングするように配置されることに応答して、ミッション幾何学形状ユニットおよび安全幾何学形状ユニットを指定するように構成され得る。ミッション幾何学形状ユニットの数は、GPUで指定された安全幾何学形状ユニットの数とは異なってもよい。

【0117】

幾何学形状データは、GPUの構成要素による幾何学形状処理のための幾何学形状ユニットに割り当てられ得る。例えば、3D負荷については、3Dデータマネージャ307は、処理ユニットに幾何学形状データを割り当てるように構成され得る。GPUが複数の幾何学形状ユニットを含む実施例では、3Dデータマネージャは、幾何学形状データを処理のための二つの幾何学形状ユニット（例えば、ミッションおよび安全処理ユニットの両方）に割り当てうる。幾何学形状データは、繰り返しの幾何学形状処理が有効でない場合（例えば、レンダリングされるレフレームが任意の安全重要要素を含まない、および／またはGPUが非保護モードにある）、単一の幾何学形状ユニットに割り当てられ得る。

【0118】

GPUに複数の幾何学形状ユニットを含む一部の実施例では、GPUは、そのデータを処理するための幾何学形状ユニットの能力に基づいて、幾何学形状データを幾何学形状ユニットに配分するように構成され得る。例えば、各幾何学形状ユニットは、幾何学形状ユニットが処理のための新しい幾何学形状データを受信することができる時（例えば、その現在の幾何学形状データの処理の完了時）、メモリ306のバッファからフレームの幾何学形状データを取得し得る。一部の実施例では、幾何学形状データは、複数のユニットの任意の幾何学形状ユニットで初めて処理され、複数のユニットの任意の幾何学形状ユニットで二度目に（またはさらに）処理され得る。このように、処理負荷を効率的に広げ、アイドルの幾何学形状ユニットを持たないように、幾何学形状データ（初めてのまたはさらなるパス上で）を幾何学形状ユニットに動的に割り当てることができる。

【0119】

各幾何学形状処理パスからの出力は比較され604、これらの出力が一致するかどうかをチェックする。この比較は、断片処理パスからの出力を確認するために本明細書に記述される方法のいずれかのチェックユニット314で実施され得る。フィルタ313は、検証のためにチェックユニットに幾何学形状ユニットの出力を同様に向けてもよい。所与のフレームに関して幾何学形状ユニットから出力データを受信すると、フィルタはその出力データをチェックユニット314に向けるように構成される。

【0120】

チェックユニットは、同一のフレームの異なる幾何学形状処理パスからの出力データを比較するように構成されている。出力データは、同一または異なる幾何学形状ユニットからのものとし得る。一部の実施例では、出力データは、指定されたミッションおよび安全

10

20

30

40

50

処理ユニットから受信される。

【0121】

一部の実施例では、チェックユニット314は、同一のフレームの異なる幾何学形状処理パスに対して受信された出力データを直接比較するように構成されており、例えば、チェックユニットは、異なるパスから出力データをストリームし、データ（ビット、バイト、ワード、または任意の他の適切なレベル）を直接比較し得る。これは606で図6に示されている。出力ストリームの比較は、チェックユニット314のフォールト検出ユニット316で実施され得る。チェックユニットが、チェックユニットで受信された出力データストリームの間の時間オフセットを補正するために（例えば、一つの幾何学形状ユニットで幾何学形状処理が、別の幾何学形状ユニットでの幾何学形状処理の前にあることにより、または、単一の幾何学形状ユニットが1度目および2度目に、フレームを逐次的に処理するように構成されていることにより）、一つまたは複数のバッファが、チェックユニットにまたはGPUの他の場所に提供され得る。幾何学形状ユニットからの出力ストリームのこうした直接的な比較を実行することによって、処理された幾何学形状データ（一般的には変換された頂点データおよびタイルリスト）は、断片プロセッサからの出力よりも一般的に実質的に小さく、また出力データの直接比較はGPUで迅速かつ効率的に実施され得る。

10

【0122】

一部の実施例では、チェックユニット314は、各幾何学形状処理パスに対して、個々の幾何学形状ユニットから受信した出力データの特徴である一つまたは複数のシグニチャを形成するように構成される。チェックユニットは、各パスで実施された幾何学形状処理が一致していることを確認するために、各幾何学形状処理パスに関連して形成されたシグニチャを比較するように構成されている。一部の実施例では、幾何学形状ユニットによるパラメータバッファ312への各書き込みに関して、シグニチャが形成される。フォールト検出ユニット316は、シグニチャの比較を実施するように構成され得る。幾何学形状パス用に生成されたシグニチャは、後続の使用のためにデータストア315に保存され得る（例えば、同一のフレームに対して別のパスのシグニチャと比較される）。チェックユニットは、例えば、チェックサム、CRC、ハッシュ、または出力データ上の指紋計算を実行し得る。幾何学形状パスに生成された一つまたは複数のシグニチャは、異なる幾何学形状処理パスに提供される出力データの比較を容易にするために、幾何学形状ユニット自体からの出力データよりもコンパクトな形態で、幾何学形状ユニットにより、フレームの幾何学形状データに実行される処理の表示を提供する。チェックユニット314のフォールト検出ユニット316は、所与のフレームに関してチェックユニットで形成されたシグニチャを比較するように構成される。シグニチャの比較は、任意の適切なレベルの粒度（例えば、ビット、バイト、ワードレベル）で実行され得る。

20

30

【0123】

チェックユニットは、複数のチェックユニットインスタンスを含んでもよく、例えば、各チェックユニットインスタンスは、GPUの処理ユニットの異なるサブセットから出力データを受信するように構成されてもよく、および／または別個のチェックユニットが、幾何学形状および断片の処理されたデータをチェックするために提供され得る。チェックユニットは、フレームの幾何学形状データを処理する幾何学形状ユニットによって生成された少なくとも処理されたデータで動作する。チェックユニットは、幾何学形状ユニットからの出力に含まれるメモリアドレスおよび／または制御データ上で追加的に動作してもよく、これによって本明細書に記述した検証操作は、より幅広いフォールトを識別するのに役立つ。

40

【0124】

フォールト検出ユニット316は、同一のフレームの異なるパスで実行される幾何学形状処理が一致しないと判定するとき、フォールト信号317を上げる607ように構成されている。説明したように、こうした比較は、例えば、幾何学形状処理された出力を直接比較すること、および／または幾何学形状処理された出力の特徴的なシグニチャを比較す

50

ることによってもよい。フォールトは、GPUでの安全違反につながる可能性がある。これは、断片処理が本明細書に記載される冗長化アプローチに従って保護されているかどうかを問わないものである。例えば、図2に示す機器クラスタに関し、実施された幾何学的形状計算のフォールトは、速度計などの機器クラスタの安全重要要素の破損を引き起こす可能性がある。

【0125】

幾何学形状処理のエラーに関連して上げられたフォールト信号317は、任意の適切な方法で提供され得る。フォールト信号は、同一のフレームの異なる幾何学形状パスからの処理された出力が同じではなく、処理された出力を生成した幾何学形状ユニットの一つまたは複数からの処理された出力が無効であることを示す。こうしたフォールト信号は、GPU300の出力として提供されてもよく、および／またはGPUで内部的に使用され得る。例えば、幾何学形状処理のエラーに関連して上げられたフォールト信号は、例えば、幾何学形状データを一对のミッションおよび安全幾何学形状ユニットに再提出すること、幾何学形状データを幾何学形状ユニットに二度以上再提出すること、および幾何学形状データをGPUの任意の二つの利用可能な幾何学形状ユニットに再提出すること、のうちの一つまたは複数により、保護された幾何学形状処理を再開され得る。フォールト信号317は、制御データ、メモリ306に記述されたデータ、およびGPUまたはGPUが接続されているシステムのレジスタまたはメモリに書き込まれたデータのうちの一つまたは複数とし得る。同じ幾何学形状データを処理するように構成された幾何学形状ユニットの出力は、電離放射線または電圧スパイクなどの一時的なイベントによって、またはハードウェア、ソフトウェアおよびファームウェアの組み合わせにおけるバグによる永久エラーによって異なりうる。

【0126】

フォールト検出ユニット316は、同一のフレームの異なる幾何学形状パスで生成された出力データを比較するよう動作可能である。説明した通り、二つ以上の幾何学形状ユニットは、同一のフレームを処理するように配置され得る。異なるパスで実行される処理は、同時に行われてもよく、または同時でなくてもよい。二つの幾何学形状処理パスが異なる二つの幾何学形状ユニットで実行される場合、これら幾何学形状ユニットによって生成される出力データ比較は、幾何学形状ユニットの対によって実行される処理が一致しているかどうかを示す。フォールト信号は、幾何学形状ユニットの一つでフォールトが発生したことを示しているが、どのユニットがフォールトを経験したかを示していない。

【0127】

三つ以上の幾何学形状ユニットのグループが、同一のフレームを処理するように配置されている場合、それらの幾何学形状ユニットからの出力データを比較することは、幾何学形状ユニットによって実行される処理が一致しているかどうかを示す。幾何学形状ユニットのグループの一つの処理された出力が他の二つの幾何学形状ユニットの出力と一致しない場合、フォールト信号は、グループの幾何学形状ユニットの一つでフォールトが発生したことを示し、グループの幾何学形状ユニットのどちらで、フォールトが発生したかをさらに示すことができる。これは、フォールトがその出力が二つ以上の他の幾何学形状ユニットからの出力と一致しない幾何学形状ユニットで発生したと仮定できるためである。

【0128】

異なる幾何学形状処理パスの出力が同一のである場合、処理された幾何学形状は有効とみなされ、これらの幾何学形状パスの出力の一つまたは複数が本明細書に記載の原理に従って断片処理に利用できるようになりうる。同一のフレームの異なる幾何学形状処理パスからの出力が一致しない場合、フォールト信号が上げられない場合があるか、一部の実施例では、出力が一致することを示す信号が上げられ得る。

【0129】

幾何学形状フェーズに関連して上げられたフォールト信号は、断片処理フェーズに関連して上昇するフォールト信号と同一であっても異なってもよい。幾何学形状フェーズに関連して上げられたフォールト信号は、例えば、制御メッセージ、割込み、制御データ

10

20

30

40

50

のフラグ、レジスタで設定された一つまたは複数のビット、およびデータパケットの一つまたは複数を含む、フォールトが発生したことの任意の種類の表示であり得る。

【0130】

グラフィック処理システムは、幾何学形状処理フェーズから検証された出力をGPU（例えば、パラメータバッファ312）のデータストアおよび／またはメモリ306に書き出すように構成され得る。フレームに関連して生成された幾何学形状処理された出力の一つのみを書き出すことにより、データストアにおけるデータの重複を回避することができる。一部の実施例では、フィルタ313は、ミッション幾何学形状ユニットで生成された幾何学形状処理された出力をデータストアおよび／またはメモリ306に書き込ませ605、安全幾何学形状ユニットで生成された処理された出力をデータストアおよび／またはメモリ306に書き込ませないように構成される。

10

【0131】

一部の実施例では、フィルタ313は、フレームに関して、幾何学形状ユニットからの最初に利用可能な出力をデータストアおよび／またはメモリ306に書き込ませるように構成される。このように、断片処理のために幾何学形状処理された出力がより早く利用可能である。一部の実施例では、フレームのタイルの断片処理は、幾何学形状処理された出力の検証を待つことなく、幾何学形状処理された出力がフレームに対して利用できるやいなや開始し得る。その出力がその後に無効な状態であることが見出された場合、断片処理は放棄されて、無効な幾何学形状処理された出力が廃棄される（例えば、処理を実行する断片プロセッサ320はフラッシュされ得る）。

20

【0132】

一部の実施例では、フレームの最後の幾何学形状処理パスの出力は、以前のパスの出力が破棄された状態で、データストアおよび／またはメモリ306に書き込まれる。このようにして、最後の処理パスの出力が知られると、全ての出力データがフォールト検出ユニットで比較できるようになる。これにより、GPU周辺で不要なデータ転送を回避し、キャッシュに書き込むことができる。

【0133】

タイリングは、保護されたタイルおよび保護されていないタイルの両方を含む、フレームの全てのタイルに対して二度実施され得る608。これは、タイリングにおけるエラーが、例えば、保護されていないタイル内に含まれている安全重要要素につながる可能性があるためである。処理された幾何学形状データのタイリングが幾何学形状ユニットで実施される実施例では、タイリングの重複は、上述の原理に従い、一つまたは複数の幾何学形状ユニットでの幾何学形状処理の重複を通して達成され得る。処理された幾何学形状データのタイリングが幾何学形状ユニットの外側で実施される実施例では、そのタイリングを実行するユニットは、フレームのタイリングを少なくとも二度繰り返す。タイリングは、保護されたタイルおよび保護されていないタイルの両方を含む、フレームの全てのタイルに対して二度実施され得る。こうした繰り返しは、幾何学形状処理の繰り返しについて本明細書に記載されるアプローチのいずれかに従って実施され得る。異なるタイリングパスの出力が検証609のためチェックユニット314に対して提供され、（例えば、フィルタ313によるデータの適切な方向を通して）、フォールト信号317は、異なるタイリングパスの出力が一致しない場合、上げられる610。一部の実施例では、図3に示す例のように、二つ以上のタイリングエンジン310が提供されている。本明細書に記載のアプローチのいずれかにより、タイリングエンジンの一つは、ミッションタイリングエンジンを指定することができ、タイリングエンジンの一つは、安全タイリングエンジンを指定することができる。タイリングエンジンによって消費されるチップエリアは通常、GPUの合計チップエリアの小さい比率であり、保護されていないタイルを含むフレームの全てのタイルの上にタイリングを複製する追加的なタイリングエンジンを提供することにおいて、実質的なエリアコストは存在しない。

30

40

【0134】

レンダリングされるフレームの各タイルに存在する要素の一覧は、チェックユニットで

50

検証される二つのタイリングの出力上で、データストア（例えば、GPUキャッシュ311のパラメータバッファ312）に書き込まれてもよい611。

【0135】

本明細書に記載の原理に従って構成されたグラフィック処理システムは、それらのタイルに対して実施される処理に関して所定の安全レベルを満たすように、保護されたタイルをレンダリングするように構成され得る。グラフィック処理システムは、所定の安全レベルを満たしていない保護されていない経路上の保護されていないタイルを同時に処理し得る。例えば、グラフィック処理システムは、保護されたタイルのレンダリングに関して、ISO26262のASIL B基準またはASIL D基準を満たすように認証され得る。

【0136】

本明細書に記載した実施例に従って構成されたグラフィック処理システムは、フレーム全体のレンダリングを単に繰り返すことによって保護されたレンダリングを実施する、従来のシステム（例えば、デュアルロックステップ構成のGPUなど）に対し、一つまたは複数の利点を持ちうる。

【0137】

第一に、安全重要要素を含むタイルのみの断片処理を繰り返すことにより、本システムは、レンダリングされるシーンの安全重要要素を保護するために、フレーム全体にわたって断片処理を繰り返す必要があるシステムに対し、実質的に改善された性能を提供できる。

【0138】

第二に、本システムは、レンダリングのために提出されたシーンに応じて、システムがタイルの一部の、または全ての保護された断片処理を実施し得るかまたは全く実施しないため、既存のシステムよりも柔軟性を提供し得る。これにより、アプリケーション開発者は、レンダリングされるシーンの安全重要部分のみを保護し、安全上重要でないシーンの部分を保護しないことによって、効率的な節約を行うことができる。

【0139】

第三に、保護された断片処理はタイルごとに実施されるが、フレーム全体の幾何学形状フェーズは保護され、保護されていないタイルに存在する要素に関連する幾何学的形状計算において生じるエラーが、保護されたタイルのレンダリングにエラーを発生させない。実質的により多くのチップエリアは一般的に、幾何学形状処理よりも断片処理のために提供されるため、本明細書に記述した実施例の一部に従い、追加的幾何学形状ユニットを提供することは（例えば、幾何学形状パスにおける同時処理が異なる幾何学形状ユニットでフレームに対して実施される）、合計GPUエリアを実質的に増大させない。また、断片処理によって消費される処理時間は一般的に、幾何学形状処理によって消費される処理時間よりも実質的に大きいので、本明細書に記述した実施例の一部に従って連続的に幾何学形状処理を繰り返すことは（例えば、幾何学形状パスが同じ幾何学形状ユニットでのフレームに対し反復される）、フレームを処理するために取る時間を実質的に増加させない。

【0140】

第四に、断片処理中のフォールトはフレームの完全なレンダリングの後ではなく、個別のタイルの完了時に検出できるため、システムのフォールト反応時間は実質的に減少し得る。一部の実施例では、タイルの断片処理を完了する前にフォールトをチェックすることによって（例えば、断片処理中に実行された一つまたは複数のバッファへの書き込みを介して生成されるシグニチャをチェックすることによって）、システムのフォールト反応時間をさらに減少させることができる。

【0141】

第五に、各保護されたタイルについて、断片処理パスの一つのみの出力が（例えば、ミッション処理ユニットから）、メモリに書き出され得るので、フレームの保護されたレンダリングを実施するときに消費されたメモリ帯域幅は、低減され得る。さらに、保護されたタイルを異なる処理ユニットで同時に処理することによって、GPUでの共通キャッシュの使用は、両方の処理ユニットが所定の断片処理動作のために取り出された同一のデータを使用できるため、メモリからのデータ（例えば、テクスチャデータ）のコストのかか

10

20

30

40

50

るダブルフェッチングを防止できる。ダブルフェッチが起こった場合は、保護されたタイルに対してのみであり、フレーム全体ではない。

【0142】

第六に、本システムは、保護されたレンダリングを少ない断片処理動作で実施することを可能にすることができるため、本明細書に記載の原理に従って構成されたGPUの電力消費量は、従来の保護されたGPUのものよりも低くすることができる。

【0143】

第七に、アプリケーション開発者は、アプリケーションに大幅な変更を加える必要なく（例えば、レンダリングのためGPUにシーンの形状を送信するソフトウェアまたはファームウェア処理など）、シーンの安全重要部分の保護されたレンダリングを可能にし得る。一部の実施例では、処理は、GPUへのドローコールに関連して、フラグまたはその他の識別子を（例えば、GPUのドライバと協働するAPIの手段によって）、ドローコールに関連する幾何学的形状が安全上重要であり、保護された経路上にレンダリングされることを示すために、設定するだけである。これにより、例えば、ドローコールを二度提出することによって、および／またはシーンの異なる部分が別個にレンダリングされ、その後、表示装置に書き込む前にフレームバッファで組み合わせることにより、アプリケーション自体において、保護されたレンダリングを扱う必要性を回避する。

【0144】

図2～5および7のグラフィック処理システムおよびデータ処理システムは、いくつかの機能ブロックを含むとして示されている。これは概略的のみであり、かかるエンティティの異なるロジック要素間の厳密な分割を定義することを意図していない。各機能ブロックは、任意の適切な方法で提供され得る。グラフィック処理システムによって形成される本明細書に記載の中間値は、いずれの点でもグラフィック処理システムによって物理的に生成される必要はなく、またその入力と出力の間のグラフィック処理システムによって実行される処理を都合よく説明する論理値を単に示し得ることが理解されるべきである。

【0145】

本明細書に記載のグラフィック処理ユニットは、集積回路上のハードウェア内に具体化され得る。本明細書に記載されるグラフィック処理システムは、本明細書に記載される方法のいずれかを実施するように構成され得る。

【0146】

本明細書で使用されるコンピュータプログラムコードおよびコンピュータ可読命令は、本明細書で使用される場合、機械言語、解釈言語、またはスクリプティング言語で発現されるコードを含む、任意の種類の実行可能コードを指す。実行可能コードには、バイナリコード、マシンコード、バイトコード、集積回路（ハードウェア説明言語またはネットリストなど）を定義するコード、およびプログラミング言語コード（C、Java、OpenCLなど）で表されるコードが含まれる。実行可能コードは、例えば、仮想マシンまたはその他のソフトウェア環境で適切に実行され、処理され、解釈され、コンパイルされ、実施されるとき、実行可能コードが支持されたコンピュータシステムのプロセッサにコードによって指定されたタスクを実行させる、ソフトウェア、ファームウェア、スクリプト、モジュールまたはライブラリの任意の種類であり得る。コンピュータ可読記憶媒体の例には、ランダムアクセスメモリ（RAM）、リードオンリーメモリ（ROM）、光学ディスク、フラッシュメモリ、ハードディスクメモリ、および磁気および光を利用するその他のメモリデバイス、および機械によってアクセス可能な命令または他のデータを保存できる他の技術が含まれる。

【0147】

プロセッサ、コンピュータ、またはコンピュータシステムは、命令を実行できるように、処理能力を有する任意の種類の装置、機械または専用回路、またはその収集もしくは部分であり得る。プロセッサは、CPU、GPU、ベクトルプロセッサ、テンソルプロセッサ、システム・オン・チップ、ステートマシン、メディア・プロセッサ、特定用途向け集積回路（ASIC）、プログラマブル・ロジックアレイ、フィールドプログラマブルゲー

10

20

30

40

50

トアレイ（FPGA）など、任意の種類の汎用または専用プロセッサとし得る。コンピュータまたはコンピュータシステムは、一つまたは複数のプロセッサを含んでもよい。

【0148】

また、所望の機能を実行するために集積回路の設計に使用される、または、プログラム可能なチップの設計に使用される、HDL（ハードウェア説明言語）ソフトウェアなどの本明細書に記載のハードウェアの構成を定義するソフトウェアを包含することも意図されている。すなわち、集積回路製造システムで処理される場合、本明細書に記載の方法のいずれかを実行するように構成されたグラフィック処理ユニットを製造するため、または本明細書に記載の任意の装置を含むグラフィック処理ユニットを製造する、システムを構成する、集積回路定義データセットの形態でコンピュータ可読プログラムコードがエンコードされたコンピュータ可読記憶媒体が提供され得る。集積回路定義データセットは、例えば、集積回路記述であり得る。

10

【0149】

集積回路製造システムにおける製造方法、本明細書に記載のグラフィック処理ユニットが提供され得る。集積回路製造システムで処理された時、グラフィック処理ユニットを製造する方法を実施する、集積回路定義データセットが提供され得る。

【0150】

集積回路定義データセットは、例えば、ネットリスト、プログラム可能なチップを構成するためのコードとして、あらゆるレベルの集積回路を定義するハードウェア記述言語として、レジスタ転送レベル（RTL）コードとして、VerilogやVHDLなどの高レベルの回路表現として、およびOASIS（RTM）やGDSIIなどの低レベルの回路表現として、コンピュータコードの形式であり得る。集積回路（例えば、RTL など）を論理的に定義する高レベルのレベルは、表示によって定義されるように集積回路の製造定義を生成するために、回路要素の定義およびこれらの要素の組み合わせ規則を含むソフトウェア環境のコンタクトの中で、集積回路の製造の定義を生成するよう構成されたコンピュータシステムで処理され得る。一般的に、機械を定義するために、コンピュータシステムで実行されるソフトウェアを含んだケースでは、一つまたは複数の中間ユーザステップ（例えば、コマンド、変数などを設ける）が、集積回路の製造定義を生成するために構成されるコンピュータシステムが集積回路の製造定義を生成するように、集積回路を定義するコードを実行するために必要とされ得る。

20

30

【0151】

グラフィック処理ユニットを製造するためにシステムを構成するために、集積回路製造システムで集積回路定義データセットを処理する一例を図7に関して説明する。

【0152】

図7は、本明細書の実施例のいずれかに記載のグラフィック処理ユニットを製造するように構成された集積回路（IC）製造システム1002の実施例を示す。特に、IC製造システム1002は、レイアウト処理システム1004および集積回路生成システム1006を含む。IC製造システム1002は、IC定義データセット（例えば、本書のいずれかの例で説明したグラフィック処理ユニット）を受信し、IC定義データセットを処理し、IC定義データセットに従ってICを生成する（例えば、本明細書の実施例のいずれかに記載されるグラフィック処理ユニットを具体化する）ように構成されている。IC定義データセットの処理は、本明細書の実施例のいずれかに記載のグラフィック処理ユニットを実施する集積回路を製造するために、IC製造システム1002を構成する。

40

【0153】

レイアウト処理システム1004は、IC定義データセットを受信および処理して、回路レイアウトを決定するように構成されている。IC定義データセットからの回路レイアウトを決定する方法は、当該技術分野で公知であり、例えば、論理的構成要素（例えば、NAND、NOR、AND、OR、MUXおよびFLIP-FLOP構成要素）の観点から生成される回路のゲートレベル表現を決定するために、RTLコードを合成することを含む。論理構成要素の位置情報を決定することにより、回路レイアウトを回路のゲー

50

トレベル表現から決定することができる。これは、回路レイアウトを最適化するために、自動的にまたはユーザーの関与によって行われうる。レイアウト処理システム1004が回路レイアウトを決定したとき、回路レイアウト定義をIC生成システム1006に出力し得る。回路レイアウト定義は、例えば、回路レイアウト記述であり得る。

【0154】

IC発生システム1006は、当技術分野で知られているように、回路レイアウト定義に従ってICを生成する。例えば、IC世代システム1006は、半導体デバイス製造プロセスを実施してICを生成することができ、これは、電子回路が半導体材料のウェーハ上で徐々に作成される光リソグラフィおよび化学処理工程の複数ステップ配列を含み得る。回路レイアウト定義は、回路定義によるICを生成するためのリソグラフィプロセスで使用できるマスクの形態であり得る。あるいは、IC生成システム1006に提供される回路レイアウト定義は、IC生成システム1006が使用できるコンピュータ可読コードの形態であってもよく、ICの生成に使用するために適切なマスクを形成し得る。

【0155】

IC製造システム1002によって実施される異なるプロセスを、例えば一つの当事者によって全て実施し得る。別の方法として、IC製造システム1002は、一部のプロセスが異なる場所で実施され得るように、分配システムであってもよく、異なる当事者によって実施され得る。例えば、(i)発生回路のゲートレベル表現を形成するための、IC定義データセットを表すRTLコードを合成する段階、(ii)ゲートレベル表現に基づいて回路レイアウトを生成する段階、(iii)回路レイアウトにしたがい、マスクを形成する段階、(iv)マスクを使用して集積回路を組み立てる段階のいくつかは、異なる場所および／または異なる当事者によって実行され得る。

【0156】

他の例では、集積回路製造システムにおける集積回路定義データセットの処理は、回路レイアウトを決定するために、IC定義データセットを処理しないで、グラフィック処理ユニットを製造するシステムを構成し得る。例えば、集積回路定義データセットは、FPGAなどの再構成可能プロセッサの構成を定義してもよく、そのデータセットの処理は、定義された構成を有する再構成可能なプロセッサ（例えば、構成データをFPGAにロードすることによって）を生成するためにIC製造システムを構成し得る。

【0157】

いくつかの実施形態では、集積回路製造定義データセットは、集積回路製造システムで処理された時に、集積回路製造システムに本明細書に記載の装置を生成させてもよい。例えば、集積回路製造定義データセットによって図7について上述した方法での集積回路製造システムの構成は、本明細書に記載される装置を製造することを引き起こし得る。

【0158】

一部の実施例では、集積回路定義データセットは、データセットで定義されたハードウェア上で実行される、またはデータセットで定義されたハードウェアと組み合わせて実行され得るソフトウェアを含むことができる。図7に示す実施例では、IC生成システムは、集積回路の製造において、集積回路定義データセットで定義されたプログラムコードに従って、集積回路上にファームウェアをロードするために、または集積回路と併用するための集積回路のプログラムコードを提供するために、集積回路定義データセットによってさらに構成され得る。

【0159】

デバイス、装置、モジュール、および／またはシステム（および本明細書に実施される方法）における本出願に記載される概念の実施は、既知の実施と比較した場合の性能改善を生じさせうる。性能改善には、計算性能の増加、待ち時間の減少、スループットの向上、および／または電力消費量の低減のうちの一つまたは複数が含まれうる。こうしたデバイス、装置、モジュール、およびシステム（例えば、集積回路）の製造中、性能改善が物理的実施に対してトレードオフとなりえ、それによって製造方法が改善される。例えば、性能改善は、レイアウトエリアに対してトレードされる場合があり、それによって既知の

実施の性能と一致するが、より少ないシリコンを使用する。これは、例えば、機能ブロックをシリアル化して再使用する、または、デバイス、装置、モジュール、および／またはシステムの要素間で機能ブロックを共有することによって、実行され得る。逆に、デバイス、装置、モジュール、およびシステム（低減されたシリコンエリアなど）の物理的実装の改善を生じさせる本出願に記載された概念は、改善された性能のためにトレードされ得る。これは、例えば、所定のエリア予算内のモジュールの複数のインスタンスを製造することによって行われうる。

【 0 1 6 0 】

出願人は、特徴または特徴の組み合わせが、本明細書で開示される問題を解決するかどうかに関係なく、当業者の一般的な一般知識に照らして、そのような特徴または組み合わせが全体として本明細書に基づいて実行されることができる程度まで、本明細書に記載されている各個別の特徴および二つ以上のそのような特徴の任意の組み合わせを単独で開示する。前述の説明の観点から、本発明の範囲内でさまざまな修正を行うことができることは当業者には明らかであろう。

10

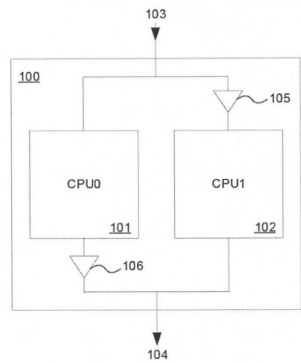
20

30

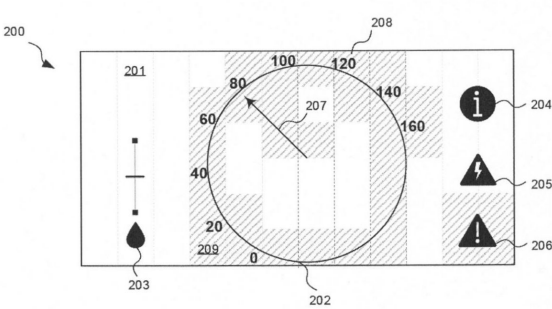
40

50

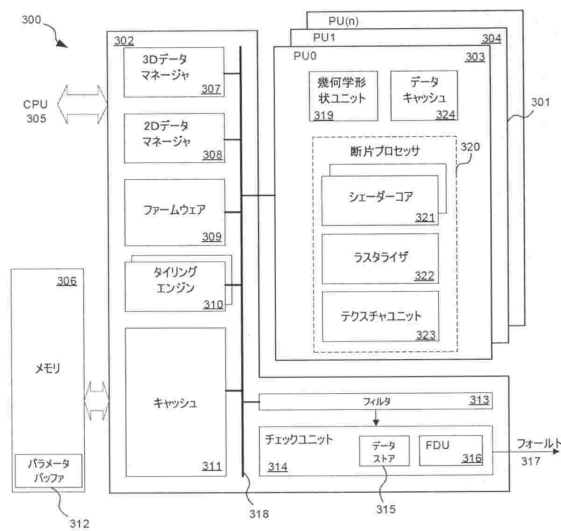
【図面】
【図 1】



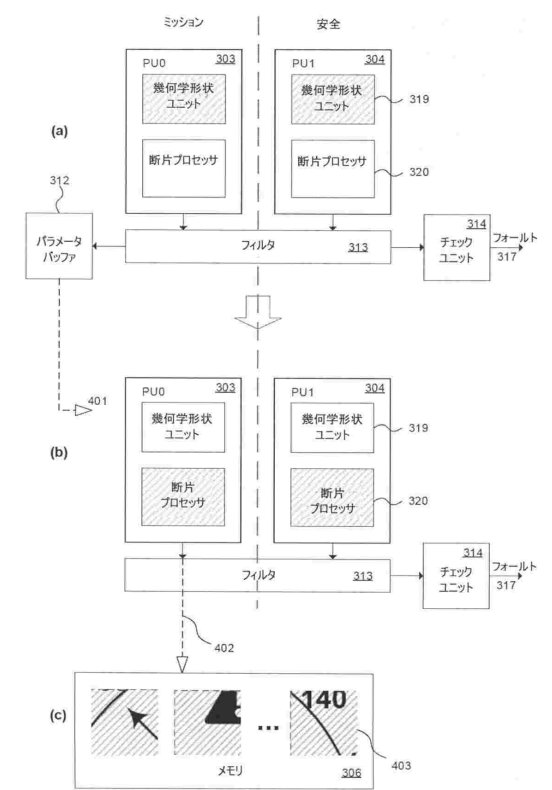
【図 2】



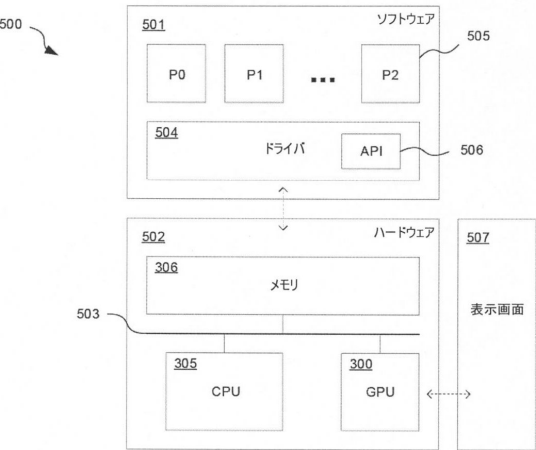
【図 3】



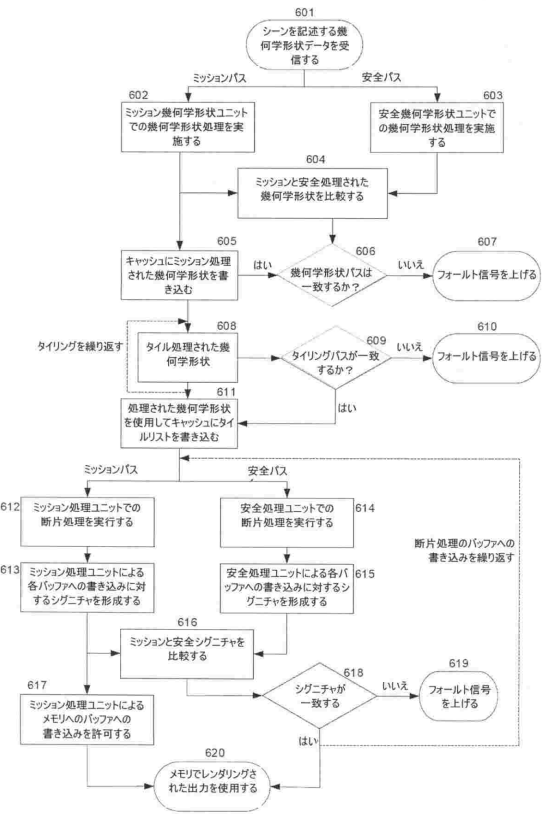
【図 4】



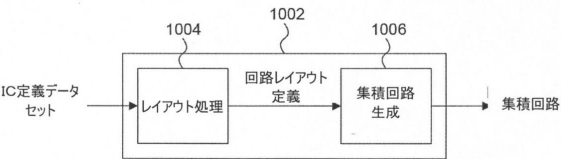
【図 5】



【図 6】



【図 7】



10

20

30

40

50

フロントページの続き

弁理士 永井 浩之
(74)代理人 100091487
弁理士 中村 行孝
(74)代理人 100105153
弁理士 朝倉 悟
(74)代理人 100107582
弁理士 関根 毅
(74)代理人 100118843
弁理士 赤岡 明
(74)代理人 100202429
弁理士 石原 信人
(72)発明者 イアン、キング
イギリス国ハートフォードシャー、キングズ、ラングリー、ホーム、パーク、エステート、イマジ
ネーション、ハウス、ケアオブ、イマジネーション、テクノロジーズ、リミテッド
(72)発明者 ジェイミー、ブルーム
イギリス国ハートフォードシャー、キングズ、ラングリー、ホーム、パーク、エステート、イマジ
ネーション、ハウス、ケアオブ、イマジネーション、テクノロジーズ、リミテッド
審査官 益戸 宏
(56)参考文献 米国特許出願公開第2016/0379332 (US, A1)
米国特許出願公開第2018/0165814 (US, A1)
米国特許出願公開第2014/0181594 (US, A1)
(58)調査した分野 (Int.Cl., DB名)
G06T 15/00