(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号

特許第6081492号
(P6081492)

(45)発行日　平成29年2月15日(2017.2.15)　(24)登録日　平成29年1月27日(2017.1.27)

(51)Int.Cl.			F I		
G06F	15/167	(2006.01)	G06F	15/167	610B
G06F	9/38	(2006.01)	G06F	9/38	370C
G06T	1/20	(2006.01)	G06T	1/20	B

請求項の数 23　(全 35 頁)

(21)出願番号	特願2014-554817 (P2014-554817)
(86)(22)出願日	平成25年1月24日 (2013.1.24)
(65)公表番号	特表2015-513715 (P2015-513715A)
(43)公表日	平成27年5月14日 (2015.5.14)
(86)国際出願番号	PCT/US2013/022900
(87)国際公開番号	WO2013/112692
(87)国際公開日	平成25年8月1日 (2013.8.1)
審査請求日	平成27年12月28日 (2015.12.28)
(31)優先権主張番号	61/591,733
(32)優先日	平成24年1月27日 (2012.1.27)
(33)優先権主張国	米国 (US)
(31)優先権主張番号	13/747,947
(32)優先日	平成25年1月23日 (2013.1.23)
(33)優先権主張国	米国 (US)

早期審査対象出願

(73)特許権者　595020643
クゥアルコム・インコーポレイテッド
ＱＵＡＬＣＯＭＭ　ＩＮＣＯＲＰＯＲＡＴＥＤ
アメリカ合衆国、カリフォルニア州　９２１２１－１７１４、サン・ディエゴ、モアハウス・ドライブ　５７７５
(74)代理人　100108855
弁理士　蔵田　昌俊
(74)代理人　100109830
弁理士　福原　淑弘
(74)代理人　100103034
弁理士　野河　信久
(74)代理人　100075672
弁理士　峰　隆司

最終頁に続く

(54)【発明の名称】グラフィックス並列処理ユニットに関するバッファ管理

(57)【特許請求の範囲】

【請求項１】

データ処理動作をパイプライン方式で実行するための方法であって、

グラフィックス処理ユニット（ＧＰＵ）のシェーダプロセッサの第１のプログラマブル計算ユニットにおいて第１のスレッドを実行することであって、前記シェーダプロセッサは前記第１のプログラマブル計算ユニットを含む複数のプログラマブル計算ユニットを含むことと、

前記ＧＰＵの前記シェーダプロセッサの前記複数のプログラマブル計算ユニットの第２のプログラマブル計算ユニットにおいて第２のスレッドを実行することと、

前記ＧＰＵを含む集積回路（ＩＣ）内の管理ユニットを用いて、前記第１のスレッドの前記実行によって生成されたデータを前記複数のプログラマブル計算ユニットによって共有される前記ＩＣの外部の集積グローバルメモリ内のバッファ内に格納することの要求を前記第１のプログラマブル計算ユニットから受信することであって、前記第１のスレッドの前記実行によって生成された前記データは、前記第２のスレッドを実行する前記第２のプログラマブル計算ユニットによって消費されることになり、前記バッファは、先入れ先出し（ＦＩＦＯ）バッファを備えることと、

前記管理ユニットを用いて、前記第１のスレッドの前記実行によって生成された前記データが格納されるべき前記バッファ内の記憶場所を決定することと、

前記ＩＣを用いて、前記第１のスレッドの前記実行によって生成された前記データを前記バッファ内の前記決定された記憶場所に格納することと、を備える、方法。

【請求項２】
前記管理ユニットを用いて、前記バッファの状態情報を前記ＩＣ内に格納することをさらに備え、
前記バッファの前記状態情報は、前記バッファの開始アドレス、前記バッファの終了アドレス、生成されたデータが格納されるべき前記バッファ内のアドレス、及びデータが取り出されるべき前記バッファ内のアドレスのうちの１つ以上を含み、
前記バッファ内の前記記憶場所を決定することは、前記バッファの前記格納された状態情報に基づいて前記第１のスレッドの前記実行によって生成された前記データが格納されるべき前記バッファ内の前記記憶場所を決定することを備える請求項１に記載の方法。
【請求項３】
前記管理ユニットを用いて、前記第１のスレッドの前記実行によって生成された前記データの少なくとも一部を取り出すことの要求を前記第２のスレッドを実行する前記第２のプログラマブル計算ユニットから受信することと、
前記管理ユニットを用いて、前記第１のスレッドの前記実行によって生成される前記データが、前記第２のスレッドを実行する前記第２のプログラマブル計算ユニットによる消費のための取り出しのために入手可能であるかどうかを決定することと、をさらに備える請求項１に記載の方法。
【請求項４】
前記第２のプログラマブル計算ユニットから前記要求を受信することは、前記第１のスレッドの前記実行によって生成されたデータを格納することの前記要求を前記第１のプログラマブル計算ユニットから受信するのと同時に、受信する前に、又は受信した後に前記第２のプログラマブル計算ユニットから前記要求を受信することを備える請求項３に記載の方法。
【請求項５】
前記第２のスレッドによって要求された前記データが、前記第２のスレッドを実行する前記第２のプログラマブル計算ユニットによる消費のための取り出しのために入手可能でないときに、
前記管理ユニットを用いて、第３のスレッドを実行するように前記第２のプログラマブル計算ユニットに指示することと、
前記管理ユニットを用いて、前記第２のスレッドによって要求された前記データが、前記第２のスレッドを実行する前記第２のプログラマブル計算ユニットによる消費のための取り出しのために入手可能であるときに前記第２のプログラマブル計算ユニットに指示することと、
前記管理ユニットを用いて、前記第２のスレッドによって要求された前記データが前記第２のスレッドを実行する前記第２のプログラマブル計算ユニットによる消費のための取り出しのために入手可能であるときに前記第２のスレッドによって要求された前記データを消費するために前記第２のスレッドを実行するように前記第２のプログラマブル計算ユニットに指示することと、をさらに備える請求項３に記載の方法。
【請求項６】
前記管理ユニットを用いて、前記第２のスレッドによって要求された前記データに加えて前記集積グローバルメモリからデータを取り出すことと、
前記管理ユニットを用いて、前記第２のスレッドによって要求された前記データに加えての前記データを前記ＩＣ内のキャッシュに格納することと、をさらに備える請求項３に記載の方法。
【請求項７】
前記第１のスレッドを実行することは、カーネルの生成するスレッドを実行することを備え、前記第２のスレッドを実行することは、前記カーネルの消費するスレッドを実行することを備える請求項１に記載の方法。
【請求項８】
前記第１のスレッドを実行することは、生成するカーネルの前記第１のスレッドを実行

することを備え、前記第２のスレッドを実行することは、消費するカーネルのスレッドを実行することを備える請求項１に記載の方法。
【請求項９】
前記ＧＰＵは、前記管理ユニットを含み、前記ＦＩＦＯバッファは、リングバッファを備える、請求項１に記載の方法。
【請求項１０】
前記バッファ内の前記記憶場所を決定することは、前記バッファ内において前記データが格納されるべき前記記憶場所を前記第１のスレッドが示さずに、前記第１のスレッドの前記実行によって生成された前記データが格納されるべき前記バッファ内の前記記憶場所を決定することを備える請求項１に記載の方法。
【請求項１１】
装置であって、
複数のプログラマブル計算ユニットによって共有される集積グローバルメモリであって、前記集積グローバルメモリはバッファを含み、前記バッファは、先入れ先出し（ＦＩＦＯ）バッファを備える、前記集積グローバルメモリと、
集積回路（ＩＣ）であって、
グラフィックス処理ユニット（ＧＰＵ）であって、
前記複数のプログラマブル計算ユニットを備え、
前記複数のプログラマブル計算ユニットの第１のプログラマブル計算ユニットは、第１のスレッドを実行するように構成され、
前記複数のプログラマブル計算ユニットの第２のプログラマブル計算ユニットは、第２のスレッドを実行するように構成された前記ＧＰＵと、
前記第１のスレッドの前記実行によって生成されたデータを前記集積グローバルメモリ内の前記バッファ内に格納することの要求を前記第１のプログラマブル計算ユニットから受信し、及び
前記第１のスレッドの前記実行によって生成された前記データが格納されるべき前記バッファ内の記憶場所を決定するように構成された管理ユニットと、を備える、集積回路（ＩＣ）とを備え、前記第１のスレッドの前記実行によって生成された前記データは、前記第２のスレッドを実行する前記第２のプログラマブル計算ユニットによって消費され、
前記ＩＣは、前記第１のスレッドの前記実行によって生成された前記データを前記バッファ内の前記決定された記憶場所に格納するように構成される、装置。
【請求項１２】
前記管理ユニットは、前記バッファの状態情報を前記ＩＣ内に格納するように構成され、
前記バッファの前記状態情報は、前記バッファの開始アドレス、前記バッファの終了アドレス、生成されたデータが格納されるべき前記バッファ内のアドレス、及びデータが取り出されるべき前記バッファ内のアドレスのうちの１つ以上を含み、
前記管理ユニットは、前記バッファの前記格納された状態情報に基づいて前記第１のスレッドの前記実行によって生成された前記データが格納されるべき前記バッファ内の前記記憶場所を決定するように構成される請求項１１に記載の装置。
【請求項１３】
前記管理ユニットは、
前記第１のスレッドの前記実行によって生成された前記データの少なくとも一部を取り出すことの要求を前記第２のスレッドを実行する前記第２のプログラマブル計算ユニットから受信し、及び
前記第１のスレッドの前記実行によって生成される前記データが、前記第２のスレッドを実行する前記第２のプログラマブル計算ユニットによる消費のための取り出しのために入手可能であるかどうかを決定するように構成される請求項１１に記載の装置。
【請求項１４】
前記管理ユニットは、前記第１のスレッドの前記実行によって生成されたデータを格納

することの前記要求を前記第１のプログラマブル計算ユニットから受信するのと同時に、受信する前に、又は受信した後に前記第２のプログラマブル計算ユニットから前記要求を受信するように構成される請求項１３に記載の装置。

【請求項１５】

前記管理ユニットは、

前記第２のスレッドによって要求された前記データが、前記第２のスレッドを実行する前記第２のプログラマブル計算ユニットによる消費のための取り出しのために入手可能でないときに、第３のスレッドを実行するように前記第２のプログラマブル計算ユニットに指示し、

前記第２のスレッドによって要求された前記データが、前記第２のスレッドを実行する前記第２のプログラマブル計算ユニットによる消費のための取り出しのために入手可能であるときに前記第２のプログラマブル計算ユニットに指示し、及び

前記第２のスレッドによって要求された前記データが前記第２のスレッドを実行する前記第２のプログラマブル計算ユニットによる消費のための取り出しのために入手可能であるときに前記第２のスレッドによって要求された前記データを消費するために前記第２のスレッドを実行するように前記第２のプログラマブル計算ユニットに指示するように構成される請求項１３に記載の装置。

【請求項１６】

前記管理ユニットは、

前記第２のスレッドによって要求された前記データに加えて前記グローバルメモリからデータを取り出し、及び

前記第２のスレッドによって要求された前記データに加えて前記データを前記ＩＣ内のキャッシュに格納するように構成される請求項１３に記載の装置。

【請求項１７】

前記第１のスレッドは、カーネルの生成するスレッドを備え、前記第２のスレッドは、前記カーネルの消費するスレッドを備える請求項１１に記載の装置。

【請求項１８】

前記第１のスレッドは、生成するカーネルのスレッドを備え、前記第２のスレッドは、消費するカーネルのスレッドを備える請求項１１に記載の装置。

【請求項１９】

前記ＧＰＵは、前記管理ユニットを含み、前記ＦＩＦＯバッファはリングバッファを備える、請求項１１に記載の装置。

【請求項２０】

前記管理ユニットは、前記第１のスレッドが前記バッファ内で前記データが格納されるべき前記記憶場所を示さずに前記第１のスレッドの実行によって生成された前記データが格納されるべき前記バッファ内の前記記憶場所を決定するように構成される請求項１１に記載の装置。

【請求項２１】

前記装置は、映像デバイス、セットトップボックス、無線ハンドセット、パーソナルデジタルアシスタント、デスクトップコンピュータ、ラップトップコンピュータ、ゲームコンソール、ビデオ会議ユニット、及びタブレットコンピューティングデバイスのうちの１つを備える請求項１１に記載の装置。

【請求項２２】

装置であって、

バッファを含む複数のプログラマブル計算ユニットによって共有される集積グローバルメモリであって、前記バッファは、先入れ先出し（ＦＩＦＯ）バッファを備える集積グローバルメモリと、

集積回路（ＩＣ）であって、

グラフィックス処理ユニット（ＧＰＵ）であって、

第１のスレッドを実行するための手段と、

第２のスレッドを実行するための手段と、

前記第１のスレッドの前記実行によって生成されたデータを前記集積グローバルメモリ内の前記バッファ内に格納することの要求を前記第１のスレッドを実行するための前記手段から受信するための手段であって、前記第１のスレッドの前記実行によって生成された前記データは、前記第２のスレッドを実行するための前記手段によって消費される手段と、

前記第１のスレッドを実行するための前記手段によって生成された前記データが格納されるべき前記バッファ内の記憶場所を決定するための手段と、

前記第１のスレッドの前記実行によって生成された前記データを前記バッファ内の前記決定された記憶場所に格納するための手段と、を備えるグラフィックス処理ユニット（ＧＰＵ）、を備える集積回路（ＩＣ）と、を備える、装置。

【請求項２３】

コンピュータによって読み取り可能な非一時的な記憶媒体であって、

実行されたときに、

グラフィックス処理ユニット（ＧＰＵ）のシェーダプロセッサの第１のプログラマブル計算ユニットにおいて第１のスレッドを実行し、ここにおいて、前記シェーダプロセッサは前記第１のプログラマブル計算ユニットを含む複数のプログラマブル計算ユニットを含む、前記第１のスレッドを実行し、

前記ＧＰＵの前記シェーダプロセッサの前記複数のプロブラマブル計算ユニットの第２のプログラマブル計算ユニットにおいて第２のスレッドを実行し、

前記ＧＰＵを含む集積回路（ＩＣ）内の管理ユニットを用いて、前記第１のスレッドの前記実行によって生成されたデータを前記複数のプログラマブル計算ユニットによって共有される前記ＩＣの外部の集積グローバルメモリ内のバッファ内に格納することの要求を前記第１のプログラマブル計算ユニットから受信し、

前記管理ユニットを用いて、前記第１のスレッドの前記実行によって生成された前記データが格納されるべき前記バッファ内の記憶場所を決定し、及び

前記ＩＣを用いて、前記第１のスレッドの前記実行によって生成された前記データを前記バッファ内の前記決定された記憶場所に格納することを１つ以上のプロセッサに行わせる命令が格納されており、前記第１のスレッドの前記実行によって生成された前記データは、前記第２のスレッドを実行する前記第２のプログラマブル計算ユニットによって消費されることになり、前記バッファは、先入れ先出し（ＦＩＦＯ）バッファを備える、コンピュータによって読み取り可能な非一時的な記憶媒体。

【発明の詳細な説明】

【技術分野】

【０００１】

本出願は、ここにおける引用によってその内容全体が組み入れられている米国仮特許出願第６１／５９１，７３３号（出願日：２０１２年１月２７日）の利益を主張するものである。

【０００２】

本開示は、メモリアクセス管理に関するものである。本開示は、より具体的には、グラフィックス処理デバイス（ＧＰＵ）におけるメモリアクセス管理に関するものである。

【背景技術】

【０００３】

グラフィックス処理ユニット（ＧＰＵ）は、グラフィックス処理に加えての目的のために使用されている。例えば、グラフィックスに関連しないアプリケーションは、ＧＰＵの大規模な並列性を利用することによって上昇した速度で実行することができる。この結果、追加のグラフィックスに関連しない処理機能を提供し、汎用ＧＰＵ（ＧＰＧＰＵ）と呼ばれるＧＰＵが得られている。例えば、ＧＰＧＰＵは、１つ以上のシェーダコア（ｓｈａｄｅｒ　ｃｏｒｅ）を含み、シェーダコアは、グラフィックスに関連しないアプリケーションと同様に、グラフィックス関連のアプリケーションも実行するように構成される。

【発明の概要】
【０００４】
概して、本開示は、グローバルメモリ内に存在しておりグラフィックス処理ユニット（ＧＰＵ）のためのデータを格納するバッファを管理するための技法に関するものである。例えば、ＧＰＵを含む集積回路（ＩＣ）チップは、パイプライン管理ユニットを含む。パイプライン管理ユニットは、グローバルメモリ内の１つ以上のバッファの状態情報を維持するように構成することができる。ＧＰＵで実行中のアプリケーションがグローバルメモリ内のバッファにアクセスするときには、グローバルメモリ内のバッファの状態情報は、ＩＣチップ内部において入手可能である。このように、ＧＰＵは、グローバルメモリ内のバッファの状態情報を決定するためにオフチップメモリアクセスを行う必要がない。
【０００５】
一例では、本開示は、データ処理動作をパイプライン方式で実行するための方法について説明する。その方法は、グラフィックス処理ユニット（ＧＰＵ）のシェーダプロセッサの第１のプログラマブル計算ユニット（ｐｒｏｇｒａｍｍａｂｌｅ　ｃｏｍｐｕｔｅ　ｕｎｉｔ）において第１のスレッドを実行することと、ＧＰＵのシェーダプロセッサの第２のプログラマブル計算ユニットにおいて第２のスレッドを実行することと、を含む。その方法は、ＧＰＵを含む集積回路（ＩＣ）内の管理ユニットを用いて、第１のスレッドの実行によって生成されたデータをＩＣ外部のグローバルメモリ内のバッファに格納することの要求を第１のプログラマブル計算ユニットから受信することも含む。この例では、第１のスレッドの実行によって生成されたデータは、第２のスレッドを実行する第２のプログラマブル計算ユニットによって消費される。さらに、この例では、バッファは、先入れ先出し方式（ＦＩＦＯ）のバッファ及びリングバッファのうちの１つを備える。その方法は、管理ユニットを用いて、第１のスレッドの実行によって生成されたデータが格納されるべきバッファ内の記憶場所（ｌｏｃａｔｉｏｎ）を決定することと、ＩＣを用いて、第１のスレッドの実行によって生成されたデータをバッファ内の決定された記憶場所に格納することと、を含む。
【０００６】
一例では、本開示は、装置について説明する。その装置は、バッファを含むグローバルメモリを含む。この例では、バッファは、先入れ先出し方式（ＦＩＦＯ）のバッファ及びリングバッファのうちの１つを備える。その装置は、グラフィックス処理ユニット（ＧＰＵ）と管理ユニットとを含む集積回路（ＩＣ）も含む。ＧＰＵは、第１のスレッドを実行するように構成された第１のプログラマブル計算ユニットと、第２のスレッドを実行するように構成された第２のプログラマブル計算ユニットと、を含む。管理ユニットは、第１のスレッドの実行によって生成されたデータをグローバルメモリ内のバッファに格納することの要求を第１のプログラマブル計算ユニットから受信するように構成される。この例では、第１のスレッドの実行によって生成されたデータは、第２のスレッドを実行する第２のプログラマブル計算ユニットによって消費される。管理ユニットは、第１のスレッドの実行によって生成されたデータが格納されるべきバッファ内の記憶場所を決定するようにも構成される。この例では、ＩＣは、第１のスレッドの実行によって生成されたデータをバッファ内の決定された記憶場所に格納するように構成される。
【０００７】
一例では、本開示は、装置について説明する。その装置は、グローバルメモリと、集積回路（ＩＣ）と、を含む。グローバルメモリは、バッファを含む。この例では、バッファは、先入れ先出し方式（ＦＩＦＯ）のバッファ及びリングバッファのうちの１つを備える。ＩＣは、第１のスレッドを実行するための手段と、第２のスレッドを実行するための手段と、を備えるグラフィックス処理ユニット（ＧＰＵ）を含む。ＩＣは、第１のスレッドの実行によって生成されたデータをグローバルメモリ内のバッファ内に格納することの要求を第１のスレッドを実行するための手段から受信するための手段も含む。この例では、第１のスレッドの実行によって生成されたデータは、第２のスレッドを実行するための手段によって消費される。ＩＣは、第１のスレッドを実行するための手段によって生成され

たデータが格納されるべきバッファ内の記憶場所を決定するための手段と、第１のスレッドの実行によって生成されたデータをバッファ内の決定された記憶場所に格納するための手段と、も含む。

【０００８】

一例では、本開示は、実行されたときに、グラフィックス処理ユニット（ＧＰＵ）のシェーダプロセッサの第１のプログラマブル計算ユニットにおいて第１のスレッドを実行すること、及びＧＰＵのシェーダプロセッサの第２のプログラマブル計算ユニットにおいて第２のスレッドを実行することを１つ以上のプロセッサに行わせる命令が格納されているコンピュータによって読み取り可能な記憶媒体について説明する。命令は、ＧＰＵを含む集積回路（ＩＣ）内の管理ユニットを用いて、第１のスレッドの実行によって生成されたデータをＩＣの外部のグローバルメモリ内のバッファ内に格納することの要求を第１のプログラマブル計算ユニットから受信することも１つ以上のプロセッサに行わせる。この例では、第１のスレッドの実行によって生成されたデータは、第２のスレッドを実行する第２のプログラマブル計算ユニットによって消費される。さらに、この例では、バッファは、先入れ先出し方式（ＦＩＦＯ）のバッファ及びリングバッファのうちの１つを備える。命令は、管理ユニットを用いて、第１のスレッドの実行によって生成されたデータが格納されるべきバッファ内の記憶場所を決定すること、及び、ＩＣを用いて、第１のスレッドの実行によって生成されたデータをバッファ内の決定された記憶場所に格納することも１つ以上のプロセッサに行わせる。

【０００９】

１つ以上の例の詳細が添付された図面及び以下の説明において示される。その説明と図面から、及び請求項からその他の特徴、目的、及び利点が明確になるであろう。

【図面の簡単な説明】

【００１０】

【図１】本開示において説明される１つ以上の例によるデバイスの例を示したブロック図である。

【図２】グラフィックス処理ユニット（ＧＰＵ）及びグローバルメモリをさらに詳細に例示したブロック図である。

【図３】本開示において説明される１つ以上の例による技法例を示したフローチャートである。

【図４】本開示において説明される１つ以上の例による他の技法例を示したフローチャートである。

【図５】図１のデバイスをさらに詳細に例示したブロック図である。

【発明を実施するための形態】

【００１１】

グラフィックス処理ユニット（ＧＰＵ）は、１つ以上のアプリケーションを実行するように構成されるシェーダプロセッサを含むことができる。これらのアプリケーションの例は、シェーダプログラム、例えば、バーテックスシェーダ（ｖｅｒｔｅｘ　ｓｈａｄｅｒ）（頂点シェーダ）、ハルシェーダ（ｈｕｌｌ　ｓｈａｄｅｒ）、フラグメントシェーダ（ｆｒａｇｍｅｎｔ　ｓｈａｄｅｒ）、幾何シェーダ（ｇｅｏｍｅｔｒｙ　ｓｈａｄｅｒ）、及びグラフィックス処理に関連するその他の該アプリケーション、を含む。さらに、幾つかのアプリケーション開発者は、ＧＰＵの大規模な並列性を利用し、グラフィックスに関連しないアプリケーションをＧＰＵで実行するのが有益であるとみなすであろう。例えば、ＧＰＵによって提供される処理の並列性は、並列行列演算がグラフィックス処理に関連していないときでさえも、それらの行列演算を実行するのに適することができる。グラフィックスに関連しないアプリケーションのその他の例は、並列演算の素早い実行が有益であることができる流体力学又は線形代数に関連する技法を含む。

【００１２】

該グラフィックスに関連しないアプリケーションを実行することが可能なＧＰＵは、汎用ＧＰＵ（ＧＰＵ）であるとみなすことができる。例えば、ＧＰＵがグラフィックスに関

連しないアプリケーションを実行中であるときには、ＧＰＵは、ＧＰＧＰＵとして機能している。ほとんどすべてのＧＰＵは、ＧＰＧＰＵとして機能するように構成することができる。

【００１３】

例示を目的として、本開示は、ＧＰＧＰＵとして機能しているＧＰＵに関して技法を説明する。しかしながら、それらの技法は、ＧＰＵがＧＰＧＰＵとして機能している（すなわち、グラフィックスに関連しないアプリケーションを実行している）事例には限定されない。さらに、本開示において説明される技法は、あらゆるタイプの処理ユニット、例えば、中央処理装置（ＣＰＵ）、アクセラレータ、又はその他のカスタムデバイスによって実装することができる。それらの技法は、ＧＰＵに関して説明されが、それらの技法は、その他のタイプの処理ユニットにも拡張可能であることが理解されるべきである。

【００１４】

ＧＰＵ内のシェーダプロセッサは、複数のシェーダコア（これらのコアはグラフィックス関連及びグラフィックスに関連しない、の両方のアプリケーションに関する命令を実行できることを示すためにプログラマブル計算ユニットとも呼ばれる）を含むことができる。プログラマブル計算ユニットの各々は、そのプログラマブル計算ユニットによって実行される命令に関して予約されているローカルメモリ、及びそれらの命令の実行によって生成されたデータ、例えば、スレッドの実行中に生成される即座の結果、を含むことができる。プログラマブル計算ユニットのローカルメモリは、その他のプログラマブル計算ユニットによってはアクセス不能であることができる。幾つかの例では、ＧＰＵで実行されるべき異なるアプリケーションは、異なるプログラマブル計算ユニットによって実行することができる。

【００１５】

本開示において説明される技法では、グラフィックス関連のアプリケーションはシェーダと呼ばれ、グラフィックスに関連しないアプリケーションは、カーネルと呼ばれる。例えば、シェーダ（すなわち、グラフィックス関連のアプリケーション）の例は、バーテックスシェーダと、フラグメントシェーダと、幾何シェーダと、含み、ただしこれらに限定されない。カーネル（すなわち、グラフィックスに関連しないアプリケーション）の例は、行列演算、流体力学、画像処理動作、映像処理動作、等を行うためのアプリケーションを含む。

【００１６】

さらに、カーネルは、ＧＰＵによって実行されるアプリケーションのみに必ずしも限定する必要はなく、ＧＰＵの固定機能ユニット（ｆｉｘｅｄ－ｆｕｎｃｔｉｏｎ　ｕｎｉｔ）（すなわち、プログラミング不能なユニット）も含む。例示のみを目的として、本開示において説明される技法は、ＧＰＵで実行されるアプリケーションであるカーネルに関して説明される。例えば、それらの技法は、ＧＰＵがＧＰＧＰＵとして機能するためにＧＰＵのシェーダプロセッサで実行するグラフィックスに関連しないアプリケーションに関して説明される。

【００１７】

カーネルは、複数のワークグループ、タスク、又はスレッドを含むことができる（これらはすべて、本開示では同義語として用いられる）。例えば、スレッドは、カーネルのその他のスレッドから独立して実行することができるカーネルの命令の組であることができる。幾つかの例では、カーネルを実行するためには、プログラマブル計算ユニットのうちの１つ以上がカーネルの１つ以上のスレッドを各々実行することができる。例えば、第１のプログラマブル計算ユニットは、カーネルの第１のスレッドを実行することができ、第２のプログラマブル計算ユニットは、同じカーネルの第２のスレッドを実行することができる。幾つかの例では、１つのプログラマブル計算ユニットが１つのカーネルの１つ以上のスレッドを実行することができ、他のプログラマブル計算ユニットは、他のカーネルの１つ以上のスレッドを実行することができる。幾つかの例では、２つの組み合わせが可能である（すなわち、幾つかのプログラマブル計算ユニットが同じカーネルの異なるスレッ

ドを実行中であり、他方、幾つかのその他のプログラマブル計算ユニットが異なるカーネルのスレッドを実行中である）。

【００１８】

概して、ＧＰＵは、単一プログラム多データ（ＳＰＭＤ）プログラミングモデルを実装するように構成することができる。ＳＰＭＤプログラミングモデルでは、ＧＰＵは、（例えば、スレッドとして）複数のプログラマブル計算ユニットでカーネルを実行することができ、各プログラマブル計算ユニットは、それ自体のデータに関する機能を実行する。さらに、ＳＰＭＤプログラミングモデルでは、プログラマブル計算ユニットは、現在の命令がプログラマブル計算ユニットによって実行されていることを示す各々のプログラムカウンタを含む。

【００１９】

ＧＰＵは、大規模な並列性を処理のために提供する一方で、ＧＰＵは、パイプライン方式でカーネルを実行するのにはあまり適していない。パイプライン方式でカーネルを実行することは、１つのカーネルによって生成されたデータが他のカーネルによって消費されるような形でカーネルを実行することを意味する。他の例として、パイプライン方式でカーネルを実行することは、同じカーネルの他のスレッドによって消費されることになるデータを生成するカーネルのスレッドを実行することを意味する。本開示では、データを生成するスレッドは、生成するスレッド（ｐｒｏｄｕｃｅｒ　ｔｈｒｅａｄ）と呼ぶことができ、データを受信するスレッドは、消費するスレッド（ｃｏｍｓｕｍｅｒ　ｔｈｒｅａｄ）と呼ぶことができる。

【００２０】

幾つかの例では、生成するスレッド及び消費するスレッドは、同じカーネルのスレッドであることができる。幾つかの例では、生成するスレッド及び消費するスレッドは、異なるカーネルのスレッドであることができる。これらの例では、生成するスレッドを含むカーネルは生成するカーネルと呼ぶことができ、消費するスレッドを含むカーネルは消費するカーネルと呼ぶことができる。

【００２１】

例えば、パイプライン方式でカーネルを実行することは、第１のスレッド（例えば、カーネルの生成するスレッド）が第２のスレッド（例えば、同じカーネル又は異なるカーネルの消費するスレッド）によって消費されるデータを生成することと考えることができる。（第１のスレッドに関する消費者であった）この第２のスレッドは、第３のスレッドに関する生成するスレッドであることができる（例えば、第２のスレッドが、第３のスレッドによって消費されるデータを生成する）。第３のスレッドは、第１及び第２のスレッドを含むカーネルと異なるカーネルに関するスレッドであることができ又は第１及び第２のスレッドを含むカーネルのうちの１つに関するスレッドであることができる。この例では、第１、第2、及び第３のスレッドは、処理パイプラインを形成するとみなすことができる。

【００２２】

パイプライン方式でカーネルを実行することは、カーネル又はスレッドを逐次に（例えば、次々に）実行することを要求するものであるとは解釈されるべきでない。例えば、上例では、ＧＰＵが第１、第２、及び第３のスレッドのうちの２つ以上を並列で（例えば、同時に）実行することが可能である。しかしながら、ＧＰＵがスレッドを逐次で実行することも可能であり、この場合も依然として、パイプライン方式でカーネルを実行するとみなすことができる。

【００２３】

カーネルの生成するスレッドを実行するプログラマブル計算ユニットは、生成されたデータをグローバルメモリ（すなわち、ＧＰＵを含む集積回路（ＩＣ）の外部のオフチップ、システムメモリ）に出力する必要があり、ここで、グローバルメモリは、例えば、システムバスを介してアクセス可能であることができる。同じカーネル又は異なるカーネルの消費するスレッドを実行する他のプログラマブル計算ユニットは、グローバルメモリから

生成されたデータを受信する必要がある。さらに詳細に説明されるように、既存のＧＰＵに関しては、グローバルメモリの管理は、計算、時間、及び／又は電力の点で非効率的になり、その結果、パイプライン方式でカーネルを実行するときに性能不良になる可能性がある。

【００２４】

本開示は、計算、時間、及び電力の点で効率的なグローバルメモリの管理のための技法について説明する。より詳細に説明されるように、ＧＰＵを含む集積回路（ＩＣ）は、パイプライン管理ユニット（ＰＭＵ）を含むことができる。代替として、ＧＰＵ自体がＰＭＵを含むことができる。ＰＭＵは、消費されるべき生成データを格納するグローバルメモリの状態情報を管理するように構成することができる。例えば、プロセッサ又はＧＰＵ自体は、グローバルメモリに格納されるべきプログラマブル計算ユニットによって生成されたデータをグローバルメモリに格納するときの記憶場所をグローバルメモリ内において予約することができる。グローバルメモリ内のこれらの予約された記憶場所は、複数のバッファとみなすことができる。幾つかの例では、複数のバッファは、リングバッファ又は先入れ先出し（ＦＩＦＯ）バッファを形成することができる。リングバッファは、ＦＩＦＯバッファの一例であるとみなすことができる。

【００２５】

ＰＭＵは、オフチップグローバルメモリ内のバッファの状態情報を示す情報を、ＩＣ又はＧＰＵの内部に（例えば、オンチップの内部キャッシュメモリ内に）格納することができる。一例として、ＰＭＵは、グローバルメモリ内のバッファの開始アドレス及び終了アドレスを示す情報を格納することができる。他の例として、ＰＭＵは、生成されたデータが格納されるべき複数のバッファ内にバッファのアドレスを、及び消費されるべきデータが読み取られる複数のバッファ内にバッファのアドレスを格納することができる。さらに他の例として、データを必要とする消費するカーネルのスレッドを実行中のプログラマブル計算ユニットがデータを必要としない消費するカーネルのその他のスレッドの実行を続けることができるように、生成するカーネルがデータの生成を完了したかどうかを示す情報を格納することができる。

【００２６】

本開示において説明される技法では、ＰＭＵは、生成するスレッドによって生成されたデータをバッファ内に格納することの要求を受信することができ、及び、消費するスレッドによる消費のために生成するスレッドによって生成されたデータをバッファから取り出すことの要求を受信することができる。ＰＭＵは、バッファの格納された状態情報に基づいて、生成するスレッドの実行によって生成されたデータが格納されるべきバッファ内の記憶場所を決定することができ、及び、バッファの格納された状態情報に基づいて消費するバッファによって消費されるべきデータが取り出されるべき記憶場所を決定することができる。

【００２７】

ＧＰＵを含むＩＣの内部又はＧＰＵ自体の内部に格納された情報を用いてグローバルメモリの状態情報を管理することによって、本開示において説明される技法は、ＧＰＵがグローバルメモリにアクセスする必要がある回数を最小限にすることができる。例えば、ＰＭＵは、ＧＰＵを含むＩＣの外部の該情報にアクセスすることによってデータが格納される又は取り出されるアドレスを決定する必要がない。ＧＰＵがグローバルメモリにアクセスする必要がある回数を最小限にすることは、電力消費量を低減させ、システムバス帯域幅負荷を低減させ、及びレーテンシーを短縮することができる。

【００２８】

さらに、以下においてより詳細に説明されるように、既存のＧＰＵでは、カーネルがグローバルメモリを管理する命令を含む必要がある。ＧＰＵは、該グローバルメモリ管理命令を実行するクロックサイクルを浪費することがあり、計算上非効率的である可能性がある。ＰＭＵがグローバルメモリの状態情報を管理することで、カーネルは、グローバルメモリ管理命令を含める必要がなく、その結果、カーネル命令の複雑さが低下し、さらに、

実行する必要があるカーネル命令が少なくなる。このように、本開示において説明される技法は、計算効率を促進させることができる。

【００２９】

図１は、本開示において説明される１つ以上の例によるデバイスの例を示したブロック図である。デバイス１０の例は、映像デバイス、例えば、メディアプレーヤー、セットトップボックス、無線ハンドセット、例えば、携帯電話、パーソナルデジタルアシスタント（ＰＤＡ）、デスクトップコンピュータ、ラップトップコンピュータ、ゲームコンソール、ビデオ会議ユニット、タブレットコンピューティングデバイス、等を含み、ただしこれらに限定されない。デバイス１０は、図１において例示されるコンポーネントに加えてのそれらを含むことができる。

【００３０】

例示されるように、デバイス１０は、集積回路（ＩＣ）１２と、グローバルメモリ２０と、を含む。グローバルメモリ２０は、デバイス１０のためのメモリであるとみなすことができる。グローバルメモリ２０は、１つ以上のコンピュータによって読み取り可能な記憶媒体を備えることができる。グローバルメモリ２０の例は、ランダムアクセスメモリ（ＲＡＭ）、電気的消去可能プログラマブル読み取り専用メモリ（ＥＥＰＲＯＭ）、フラッシュメモリ、又は、希望されるプログラムコードを命令及び／又はデータ構造の形態で搬送又は格納するために使用することができ及びコンピュータ又はプロセッサによってアクセス可能であることができるその他のあらゆる媒体を含み、ただしこれらに限定されない。

【００３１】

幾つかの態様では、グローバルメモリ２０は、本開示においてプロセッサ１４及びＧＰＵ１６に帰する機能を果たすことをプロセッサ１４及び／又はＧＰＵ１６に行わせる命令を含むことができる。従って、グローバルメモリ２０は、実行されたときに、様々な機能を実行することを１つ以上のプロセッサ（例えば、プロセッサ１４及びＧＰＵ１６）に行わせる命令を格納しているコンピュータによって読み取り可能な記憶媒体であることができる。

【００３２】

グローバルメモリ２０は、幾つかの例では、非一時的な記憶媒体であるとみなすことができる。用語“非一時的な”は、記憶媒体が搬送波又は伝搬された信号において具現化されないことを示すことができる。しかしながら、用語“非一時的な”は、グローバルメモリ２０が取り外し不能であるか又はその内容が静的であることを意味するとは解釈されるべきでない。一例として、グローバルメモリ２０は、デバイス１０から取り外して他のデバイスに移動させることができる。他の例として、グローバルメモリ２０に実質的に類似するグローバルメモリを、デバイス１０に挿入することができる。幾つかの例では、非一時的記憶媒体は、（例えば、ＲＡＭにおいて）経時で変化する可能性があるデータを格納することができる。

【００３３】

ＩＣ１２は、プロセッサ１４と、グラフィックス処理ユニット（ＧＰＵ）１６と、パイプライン管理ユニット（ＰＭＵ）１８と、を含む。ＩＣ１２は、プロセッサ１４、ＧＰＵ１６、及びＰＭＵＩ８を収納する又は形成するあらゆるタイプの集積回路であることができる。例えば、Ｃ１２は、チップパッケージ内の処理チップであるとみなすことができる。ＰＭＵ１８は、ＩＣ１２の一部を形成するハードウェアユニットであることができ又はＧＰＵ１６内のハードウェアであることができる。ＰＭＵ１８は、ＩＣ１２内又はＧＰＵ内のハードウェアで実行されるソフトウェアであることが可能である。例示及び説明の目的上、技法は、ハードウェアユニットであるＰＭＵ１８に関して説明される。

【００３４】

プロセッサ１４、ＧＰＵ１６、及びＰＭＵＩ８は、単一のＩＣ１２の一部として例示されているが、本開示の態様はそのようには限定されない。幾つかの例では、プロセッサ１４及びＧＰＵ１６は、異なる集積回路（すなわち、異なるチップパッケージ）内に収納す

ることができる。これらの例では、ＰＭＵ１８は、ＧＰＵ１６と同じ集積回路内に収納することができる。幾つかの例では、ＰＭＵ１８は、ＧＰＵ１６の一部として形成することができる。一例として、プロセッサ１４及びＧＰＵ１６は、同じ集積回路（すなわち、同じチップパッケージ）内に収納することができ、ＰＭＵ１８は、ＧＰＵ１６内に形成することができる。他の例として、プロセッサ１４及びＧＰＵ１６は、異なる集積回路（すなわち、異なるチップパッケージ）内に収納することができ、ＰＭＵ１８は、ＧＰＵ１６内に形成することができる。

【００３５】

プロセッサ１４、ＧＰＵ１６、及びＰＭＵ１８の例は、デジタル信号プロセッサ（ＤＳＰ）、汎用マイクロプロセッサ、特定用途向け集積回路（ＡＳＩＣ）、フィールドプログラマブルロジックアレイ（ＦＰＧＡ）、又はその他の同等の集積回路又はディスクリートロジック回路を含み、ただしこれらに限定されない。幾つかの例では、ＧＰＵ１６及びＰＭＵ１８は、以下においてより詳細に説明されるように、グラフィックス処理に適する大規模な並列処理能力をＧＰＵ１６に提供し及び管理するグローバルメモリ２０をＰＭＵ１８に提供する集積回路及び／又はディスクリートロジック回路を含む専用ハードウェアであることができる。幾つかの例では、ＧＰＵ１６は、汎用処理も含むことができ、汎用処理タスク（例えば、グラフィックスに関連しないタスク）を実装するときには汎用ＧＰＵ（ＧＰＧＰＵ）と呼ぶことができる。

【００３６】

プロセッサ１４は、ホストと時々呼ばれ、デバイス１０の中央処理装置（ＣＰＵ）であることができる。プロセッサ１４は、様々なタイプのアプリケーションを実行することができる。アプリケーションの例は、ウェブブラウザ、電子リーダー、電子メールアプリケーション、スプレッドシート、ビデオゲーム、映像再生、音声再生、ワードプロセシング、又は、表示のためにビュー可能なオブジェクトを生成するその他のアプリケーション、又はその他のタイプのアプリケーションを含む。グローバルメモリ２０は、１つ以上のアプリケーションの実行のための命令を格納することができる。

【００３７】

幾つかの例では、プロセッサ１４は、処理タスク、例えば、大規模な並列動作を要求するタスク、をＧＰＵ１６に委ねることができる。一例として、グラフィックス処理は、大規模な並列動作を要求し、プロセッサ１４は、該グラフィックス処理タスクをＧＰＵ１６に委ねることができる。幾つかの例では、プロセッサ１４は、グラフィックス処理に関連しないタスクをＧＰＵ１６に委ねることができる。例えば、行列演算は、並列動作を要求し、ＧＰＵ１６のほうがプロセッサ１４と比較して該動作を実装するのに適している。

【００３８】

タスクを実装するために、ＧＰＵ１６は、１つ以上のアプリケーションを実行するように構成することができる。例えば、グラフィックス関連の処理に関しては、ＧＰＵ１６は、バーテックスシェーダ、フラグメントシェーダ、及び幾何学シェーダ、等のアプリケーションを実行することができる。グラフィックスに関連しない処理に関しては、ＧＰＵ１６は、該処理用に設計されたアプリケーション（例えば、行列演算を実装するためのアプリケーション又は流体力学に関するアプリケーション）を実行することができる。いずれの例に関しても（例えば、グラフィックス関連の処理又はグラフィックスに関連しない処理）、プロセッサ１４は、１つ以上のアプリケーションを実行するようにＧＰＵ１６に命令することができる。

【００３９】

プロセッサ１４は、特定のアプリケーション処理インタフェース（ＡＰＩ）によりＧＰＵ１６と通信することができる。例えば、プロセッサ１４は、命令、例えば、ＡＰＩを利用して１つ以上のアプリケーションを実行するようにＧＰＵ１６に命令する命令、をＧＰＵ１６に送信することができる。該ＡＰＩの例は、Ｍｉｃｒｏｓｏｆｔ（登録商標）によるＤｉｒｅｃｔＸ（登録商標）、Ｋｈｒｏｎｏｓグループによる０ｐｅｎＧＬ（登録商標）、及びＫｈｒｏｎｏｓグループによる０ｐｅｎＣＬ（登録商標）を含む。しかしながら

、本開示の態様は、ＤｉｒｅｃｔＸ、ＯｐｅｎＧＬ又はＯｐｅｎＣＬ　ＡＰＩには限定されず、開発済みの、現在開発中の、又は将来開発予定のその他のタイプのＡＰＩにまで拡張することができる。さらに、本開示において説明される技法は、ＡＰＩにより機能することは要求されず、プロセッサ１４及びＧＰＵ１６は、あらゆる通信技法を利用することができる。

【００４０】

一例として、グラフィックス関連のアプリケーションに関しては、プロセッサ１４は、ＯｐｅｎＧＬ　ＡＰＩを用いてＧＰＵと通信することができる。グラフィックスに関連しないアプリケーションに関しては、プロセッサ１４は、ＯｐｅｎＣＬ　ＡＰＩを用いてＧＰＵと通信することができる。繰り返すと、本開示において説明される技法は、プロセッサ１４がＯｐｅｎＧＬ及び／又はＯｐｅｎＣＬ　ＡＰＩを用いてＧＰＵ１６と通信することは必ずしも要求しない。

【００４１】

ＧＰＵ１６が実行するグラフィックス関連のアプリケーションは、シェーダと呼ぶことができ、ＧＰＵ１６が実行するグラフィックスに関連しないアプリケーションは、カーネルと呼ぶことができる。例えば、グローバルメモリ２０は、シェーダ及びカーネルの命令を格納することができ、プロセッサ１４で実行するコンパイラは、シェーダ及びカーネルの命令をＧＰＵ１６での実行のためのオブジェクトコードに変換することができる。他の例として、グローバルメモリ２０は、ＧＰＵ１６が取り出して実行するシェーダ及びカーネルのオブジェクトコードを格納することができる。

【００４２】

シェーダの例は、グラフィックス関連の処理のためのバーテックスシェーダと、フラグメントシェーダと、幾何シェーダと、を含む。カーネルの例は、（例えば、線形代数又は流体力学に関する）グラフィックス処理に関連しないアプリケーションを含む。追加例として、カーネルは、画像処理及び映像処理に関するアプリケーションを含む。

【００４３】

ＧＰＵ１６は、シェーダプロセッサを含むことができ、シェーダプロセッサは、シェーダ及びカーネルを実行することができる。例えば、ＧＰＵ１６のシェーダプロセッサは、１つ以上のシェーダコア（プログラマブル計算ユニットと呼ばれる）を含むことができ、１つ以上のプログラマブル計算ユニットの各々がカーネルを実行することができる。

【００４４】

カーネルは、ＧＰＵ１６で実行するアプリケーションとして説明されるが、カーネルはそのように限定的であるとはみなされるべきでない。カーネルのその他の例は、ＧＰＵ１６の固定機能ユニットを含む。例えば、ＧＰＵ１６は、プログラマブル計算ユニットと、固定機能ユニットと、を含む。プログラマブル計算ユニットは、アプリケーションを実行することによって機能上の柔軟性を提供することができる。固定機能ユニットは、機能上の柔軟性を提供しないハードウェアユニットであることができ、特定の目的のために設計することができる。概して、用語カーネルは、グラフィックスに関連しない目的のためにデータを受信し、そのデータを処理し、及びそのデータを出力とするアプリケーション又はハードウェアユニットを意味する。しかしながら、例示を目的として、本開示において説明される技法は、カーネルが固定機能ユニットである例にまで拡張可能であるという理解の下でカーネルがアプリケーションである例を用いて説明される。

【００４５】

本開示において説明される技法では、１つのプログラマブル計算ユニットがカーネルの全命令を実行するのではなく、複数のプログラマブル計算ユニットがカーネルの一部分を実行するのが可能である。カーネルの一部分は、ワークグループ、タスク、又はスレッドと呼ぶことができる（すべて同意語である）。例えば、カーネルのワークグループ、タスク、又はスレッドは、そのカーネルのその他のワークグループ、タスク、又はスレッドから独立して実行することができる命令の組である。

【００４６】

幾つかの例では、１つ以上のプログラマブル計算ユニットの第１の組がカーネルのスレッドを実行することができ、１つ以上のプログラマブル計算ユニットの第２の組がカーネルのスレッドを実行することができる。幾つかの場合は、プログラマブル計算ユニットの第１の組及びプログラマブル計算ユニットの第２の組が実行するスレッドは、同じカーネルのスレッドであることができる。幾つかの場合は、プログラマブル計算ユニットの第１の組及びプログラマブル計算ユニットの第２の組が実行するスレッドは、異なるカーネルのスレッドであることができる。これらの例のいずれでも、スレッドのうちの１つは、生成されたデータをスレッドの他の１つに出力する必要がある。換言すると、ＧＰＵ１６は、パイプライン方式でカーネルを実行することができる。

【００４７】

上述されるように、パイプライン方式でカーネルを実行することは、１つのスレッドによって生成されたデータが他のスレッドによって消費され、この他のスレッドによって生成されたデータがさらに他のスレッドによって消費され、以下同様であるような形でカーネルを実行することを意味することができる。これらの例では、スレッドは、異なるカーネル又は同じカーネルのスレッドであることができ、又は、幾つかのスレッドは異なるカーネルに関するものであり、その他のスレッドは同じカーネルに関するものであることができる。これらの例では、カーネルは、データが生成及び消費されるパイプラインを形成するとみることができる。例えば、同じカーネル又は異なるカーネルの第１、第２、及び第３のスレッドが１つのパイプを形成することができ、その中で、第１のスレッドがデータを生成し、処理のために第2のスレッドによって消費されるためにそのデータを送信する。第2のスレッドは、受信されたデータを処理してデータを生成し、生成されたデータを処理のために第３のスレッドに送信し、以下同様である。

【００４８】

この例では、第１のスレッドは、生成するスレッドと呼ぶことができ、第２のスレッドは、第１のスレッドにとっての消費するスレッド、第３のスレッドにとっての生成するスレッドと呼ぶことができ、第３のスレッドは、消費するスレッドと呼ぶことができる。第１、第２、及び第３のスレッドが異なるカーネル（例えば、それぞれ、第１、第２、及び第３のカーネル）に関するものである例では、第１のカーネルは生成するカーネルと呼ぶことができ、第２のカーネルは、第１のカーネルにとっての消費するカーネル、第３のカーネルにとっての生成するカーネルと呼ぶことができ、第３のカーネルは、消費するカーネルと呼ぶことができる。

【００４９】

既存のＧＰＵでは、パイプライン方式でカーネルを実行することは、計算上及び電力上非効率的である可能性がある。例えば、プログラマブル計算ユニットの各々は、プログラマブル計算ユニットによって実行される命令を格納するための、処理されるデータを格納するための、及び生成されるデータを格納するためのローカルメモリを含むことができ、生成することができる中間結果を含む。しかしながら、プログラマブル計算ユニットのローカルメモリは、その他のプログラマブル計算ユニットによってアクセスすることができない。

【００５０】

従って、幾つかの例では、パイプライン方式でカーネルを実行するために、ＧＰＵ１６は、プログラマブル計算ユニットのローカルメモリに格納された生成データを取り出し、その生成データをグローバルメモリ２０に格納することができる。グローバルメモリ２０にデータを格納することは、データをオフチップで格納すると呼ぶことができ、その理由は、グローバルメモリ２０は、ＧＰＵ１６を収納する集積回路の外部に（すなわち、ＩＣ１２の外部に）存在するためである。ＧＰＵ１６は、グローバルメモリ２０に格納されたデータを取り出し、取り出されたデータを他のプログラマブル計算ユニットのローカルメモリ内にローディングすることができる。

【００５１】

説明するための例として、第１のプログラマブル計算ユニットが生成するカーネルのス

レッドを実行中であると仮定する。この例では、第１のプログラマブル計算ユニットは、生成するカーネルのスレッドの実行によって生成されたデータを第１のプログラマブル計算ユニットのローカルメモリに格納することができる。ＧＰＵ１６は、第１のプログラマブル計算ユニットのローカルメモリから生成されたデータを取り出し、生成されたデータをグローバルメモリ２０内に格納することができる。

【００５２】

この例において、第２のプログラマブル計算ユニットが消費するカーネルのスレッドを実行中であると仮定する。この例では、ＧＰＵ１６は、生成するカーネルによって生成されたデータをグローバルメモリ２０から取り出し、第２のプログラマブル計算ユニットのローカルメモリ内にデータをローディングすることができる。これで、消費するカーネルは、第２のプログラマブル計算ユニットのローカルメモリ内に格納されたデータを消費することができる。

【００５３】

上例において、ＧＰＵ１６は、第２のプログラマブル計算ユニットが第１のプログラマブル計算ユニットのローカルメモリにアクセスしないため、生成するカーネルによって生成されたデータをグローバルメモリ２０内に格納する必要がある。このように、グローバルメモリ２０は、後続して消費されることになる生成されたデータの中間的なストレージとして機能する。

【００５４】

概して、生成されたデータがグローバルメモリ２０に格納される方法及び／又はグローバルメモリ２０からデータが取り出される方法を管理することは、処理上及び計算上非効率的になる可能性がある。一例として、非効率的ではあるが、データがグローバルメモリ２０に格納される方法をカーネルが管理することが可能である。例えば、カーネルは、データを格納するための又は格納されたデータを取り出すためのグローバルメモリ２０内のアドレス（例えば、ポインタ）を決定することをプログラマブル計算ユニットの算術論理ユニット（ＡＬＵ）に行わせる命令を含むことが可能である。

【００５５】

他の例として、グローバルメモリ２０は、原子カウンタを格納することができる。原子カウンタの値は、消費されるデータを入手可能であるかどうかを示すことができる。例えば、生成するカーネルは、グローバルメモリ２０に格納された原子カウンタの現在値を読み取る命令を含むことができる。生成するカーネルは、生成するカーネルが格納したデータの量に基づいて原子カウンタの値を変更する命令と、原子カウンタの変更された値をグローバルメモリ２０内に再び書き込む命令と、も含むことができる。

【００５６】

消費するカーネルは、グローバルメモリ２０に格納された原子カウンタの値を定期的に検査する命令を含むことができる。原子カウンタの値が十分に大きいときには、消費するカーネルは、消費されるべきデータが入手可能であると決定することができる。例えば、原子カウンタの値がＸであり、さらに、生成するカーネルがＮの量のデータを生成したと仮定する。この例では、消費するカーネルは、原子カウンタの値を定期的に検査することを消費するカーネルのスレッドを実行中のプログラマブル計算ユニットに行わせる命令を含むことができる。原子カウンタの値がＸ＋Ｎであるとプログラマブル計算ユニットが決定したときには、プログラマブル計算ユニットは、格納されたデータを消費のためにグローバルメモリ２０から取り出すようにＧＰＵ１６に要求することができる。

【００５７】

この方法により、ソフトウェア（すなわち、カーネルの命令）を用いてカーネルをパイプライン方式で実行することが可能である。しかしながら、カーネル内の命令を用いてパイプライン方式でカーネルを実行することが非効率的である様々な理由が存在する。例えば、データを格納すべきアドレス又はグローバルメモリ２０内のどの場所にデータが格納されているかを決定する命令をカーネル内に含むことは、不必要に電力を消費すること、及び、グローバルメモリ２０内のアドレスを決定するための命令を処理してクロックサイ

クルを浪費することをプログラマブル計算ユニットのＡＬＵに要求することになる。
【００５８】
さらに、原子カウンタの値を定期的に検査することは、オフチップで（すなわち、グローバルメモリ２０内の）情報にアクセスするようにＧＰＵ１６に要求することになる。原子カウンタの値をグローバルメモリ２０から読み取ること及び原子カウンタの変更された値をグローバルメモリ２０に書き込むことは、望ましくない量の電力を消費する可能性がある。さらに、例示されるように、ＩＣ１２は、メモリバス２４を介してグローバルメモリ２０に結合される。メモリバス２４が処理できるデータの量には帯域幅上の限界がある。従って、ＧＰＵ１６が原子カウンタの値を読み取る及び書き込むことができるときには遅延が存在することがある。
【００５９】
さらに、データが消費するカーネルによって消費されるために入手可能であるときは不明であるため、消費するカーネルを実行するプログラマブル計算ユニットは、データが消費のために入手可能であるかどうかを決定するために原子カウンタの値を検査することをＧＰＵＩ１６に定期的に行わせることになる。原子カウンタの値を定期的に検査することは、消費するカーネルのスレッドを“スピニング”（ｓｐｉｎｎｉｎｇ）した状態にとどまらせることになる可能性がある。例えば、原子カウンタの読み取り値が、データは消費のためにまだ完全には入手可能でないことを示す場合は、プログラマブル計算ユニットは、プログラマブル計算ユニットが原子カウンタの値を再度検査するまで、消費するカーネルのスレッドの実行を休止することができる。データが依然として入手可能でない場合は、プログラマブル計算ユニットは再度待機し、データが入手可能であるかどうかを再度検査することをＧＰＵ１６に行わせる。この例では、消費するカーネルのスレッドは、消費されるべきデータがグローバルメモリ２０において入手可能でない時間の間ビジー－待機状態にとどまることになる。換言すると、スピニング中には、プログラマブル計算ユニットは、どのような機能も実行していることができず、データの消費を遅延させるおそれがある。
【００６０】
（例えば、原子カウンタの値を読み取ることによって）データが入手可能であるかどうかをプログラマブル計算ユニットが決定する頻度が高い場合は、ＧＰＵ１６は、グローバルメモリ２０に格納された原子カウンタの値をあまりにも頻繁に読み取りすぎることによって電力を浪費する可能性がある。データが入手可能であるかどうかをプログラマブル計算ユニットが決定する頻度が低い場合は、データが入手可能な時点とＧＰＵ１７６がデータを取り出す時点の間に時間の浪費が生じ、それもデータの消費を遅延させる。
【００６１】
さらに、グローバルメモリ２０が原子カウンタを格納する上記の技法のうちの一部において、１つのカーネルが原子カウンタの値を読みとり中、変更中、及び書き込み中であるときには、その他のカーネルは、原子カウンタの値を読み取ること、変更すること、又は書き込むことを許容されない。該事例では、２つの生成するスレッドがグローバルメモリ２０での格納のために同時にデータを出力する必要があるときに、それらのスレッドのうちの１つはデータを出力することができるが、他方のスレッドは、原子カウンタにアクセスできないためデータを出力することができない。該事例では、格納へのアクセスが拒否されたスレッドは、原子カウンタへのアクセスが可能になるまでスピンするおそれがあり、原子カウンタへのアクセスが可能になった時点で、格納へのアクセスを拒否されたスレッドが、グローバルメモリ２０にデータを出力することができる。２つの消費するスレッドが同時にデータにアクセスしようと試みたときにも同じことが起きるおそれがある。
【００６２】
本開示において説明される技法は、上述される技法と比較して、ＧＰＵ１６がより効率的にパイプライン方式でカーネルを実行するのを可能にすることができる。より詳細に説明されるように、パイプライン管理ユニット（ＰＭＵ）１８は、様々なスレッドによって生成されるデータ及び様々なスレッドによって消費されることになるデータの状態情報を

格納するように構成することができる。この方法により、ＧＰＵ１６は、オフチップで連続的に情報にアクセスし、データがどこに格納されているか及びデータがいつ消費のために入手可能であるかを示す必要がない。むしろ、ＰＭＵ１８は、該情報を内部に（すなわち、ＩＣ１２内に）格納することができる。

【００６３】

例示されるように、グローバルメモリ２０は、バッファ２２Ａ乃至２２Ｎ（総称してバッファ２２と呼ばれる）を含むことができる。バッファ２２は、グローバルメモリ２０内の格納のための記憶場所であることができる。バッファ２２の例は、先入れ先出し（ＦＩＦＯ）バッファ又はリングバッファを含む。

【００６４】

プロセッサ１４は、グローバルメモリ２０内に常駐するバッファ数を定義し及びグローバルメモリ２０内の記憶場所を予約するように構成することができる。例えば、プロセッサ１４は、バッファ２２の開始及び終了記憶場所（すなわち、開始及び終了アドレス）を定義することができる。プロセッサ１４は、ＧＰＵ１６のシェーダプロセッサ内に常駐するプログラマブル計算ユニット数に基づいてグローバルメモリ２０内に常駐するバッファ数を定義することができる。一例として、プロセッサ１４は、各プログラマブル計算ユニットに関して１つ以上の入力バッファ２２（すなわち、プログラマブル計算ユニットで実行中のカーネルによって消費されるデータを格納する１つ以上のバッファ）及び各プログラマブル計算ユニットに関してゼロ以上の出力バッファ２２（すなわち、ＧＰＵ１６のプログラマブル計算ユニットで実行中のカーネルによって生成されたデータを格納するゼロ以上のバッファ）が存在するような形でグローバルメモリ２０内に常駐するバッファ数を定義することができる。

【００６５】

さらに、プロセッサ１４は、バッファのサイズを定義するように構成することができる。例えば、プロセッサ１４は、バッファ２２の各々の中における格納用記憶場所の数（例えば、バッファ２２の長さ）を定義するように構成することができる。プロセッサ１４は、格納用記憶場所の各々に格納することができるデータの量（例えば、バッファ２２の幅）を定義することもできる。幾つかの例では、プロセッサ１４は、バッファ２２にデータを予めポピュレート（ｐｒｅ－ｐｏｐｕｌａｔｅ）することができる。

【００６６】

幾つかの例では、プロセッサ１４は、最低数のバッファ２２を定義するように構成することができる。一例として、プロセッサ１４は、最低数１２８のバッファ２２を定義するように構成することができる。バッファ２２の最低数が１２８であることは、例示を目的とするものであり、限定するものであるとは解釈されるべきでない。バッファ２２の最低数は、１２８よりも多いこと又は少ないことができる。幾つかの例では、バッファ２２の最低数に関する要求は必要がない、

プロセッサ１４は、バッファ２２の状態を決定するために様々な命令を実行するようにも構成することができる。例えば、プロセッサ１４は、バッファ２２に格納されたデータをＩＣ１２又はＧＰＵ１６のバッファ内にコピーする命令及びＩＣ１２又はＧＰＵ１６のバッファ内に格納されたデータをバッファ２２内にコピーする命令を実行することができる。プロセッサ１４は、バッファ２２に格納されたデータの量を定義する命令、及び（例えば、バッファが壊れていないことを確認するために）バッファ２２の長さ及び幅を確認する命令も実行することができる。プロセッサ１４がバッファ２２の状態を決定するのを可能にする命令の該実行は、すべての例で要求されるわけではないが、カーネルの開発者がＧＰＵ１６ではなくプロセッサ１４で命令を実行することによってバッファ２２の状態を決定するのに役立つことができる。

【００６７】

幾つかの例では、プロセッサ１４は、バッファ２２に関する拡大係数（ａｍｐｌｉｆｉｃａｔｉｏｎ　ｆａｃｔｏｒ）を定義するように構成することができる。拡大係数は、バッファ２２のうちの１つに格納するためにカーネルのスレッドによって生成することがで

きる要素の最大数を示すことができる。拡大係数は、データを格納すべきバッファ２２のうちの１つがすべての生成されたデータを格納することができない状況にとって必要になるであろう。この結果、バッファ２２内での不十分な格納スペースに起因してカーネルの実行を停止させることがあり、立ち往生状態になる可能性がある（例えば、カーネルが実行状態にまったく戻らない）。

【００６８】

該立ち往生が生じる機会を最小限にするために、プロセッサ１４は、グローバルメモリ２０の大きな部分を予約することができる（例えば、ほとんどのあらゆるタイプのデータを格納する上で十分な大きさの長くて幅広いバッファ２２を定義する）。これは、幾つかの場合は適切に機能することができるが、グローバルメモリ２０の大きな部分を予約するのが可能でないその他に関しては適切でない。幾つかの事例では、開発者は、カーネルがあまりにも多すぎるデータを生成せず、それによって立ち往生の機会を最小にするような形でカーネルを開発することができる。

【００６９】

プロセッサ１４がバッファ２２を定義するとして説明されているが、本開示において説明される技法は、そのようには限定されない。幾つかの例では、プロセッサ１４以外の処理ユニットがバッファ２２を定義するように構成することができる。幾つかの例では、ＧＰＵ１６がバッファ２２を定義するのが可能である。しかしながら、説明を簡単にするために、これらの技法は、プロセッサ１４がバッファ２２を定義するとして説明される。

【００７０】

プロセッサ１４は、バッファ２２の情報をパイプライン処理ユニット（ＰＭＵ）１８に送信することができる。例えば、ＰＭＵ１８は、バッファ２２の数、バッファ２２の開始及び終了アドレス、バッファ２２の長さと幅を示す情報、及び、プロセッサ１４がバッファ２２に関して決定したその他の情報を受信することができる。ＰＭＵ１８は、バッファ２２の該状態情報をＩＣ１２内のレジスタ内に格納することができる。プロセッサ１４からのバッファ２２の情報が存在することで、ＰＭＵ１８は、プログラマブル計算ユニットで実行中のカーネルのスレッドがデータを生成及び消費しているときにバッファ２２の状態情報を管理するように構成することができる。

【００７１】

例えば、カーネルのスレッドを実行するプログラマブル計算ユニットがデータを生成して生成されたデータを出力後は、ＰＭＵ１８は、データを受信し、データが格納されるべきアドレスを決定することができる。例えば、ＰＭＵ１８は、バッファ２２のうちのいずれにデータを格納すべきかを決定することができる。バッファ２２がリングバッファ又はＦＩＦＯバッファである例では、ＰＭＵ１８は、バッファ２２の初めと終わりを識別するポインタに関する情報を格納することができる。リングバッファの場合は、ＰＭＵ１８は、有効なデータの初め及び有効なデータの終わりを識別するポインタに関する情報も格納することができる。

【００７２】

従って、生成されたデータが格納されるべきアドレス又は消費のためにデータが取り出されるアドレスを決定することをプログラマブル計算ユニットに行わせる命令を含むカーネルではなく、ＰＭＵ１８を、生成されたデータが格納されるべきアドレス又は消費のためにデータが取り出されるアドレスを決定するように構成することができる。この方法により、ＧＰＵ１６は、クロックサイクルを浪費せず、さらに、プログラマブル計算ユニットのＡＬＵは、データが格納されるべき又はデータが取り出されるべきアドレスを決定するのに処理電力を浪費することがない。

【００７３】

さらに、ＰＭＵ１８は、消費されるべきデータの消費準備が整っている時を決定するように構成することができる。例えば、グローバルメモリ２０が原子カウンタを格納するのではなく、ＰＭＵ１８がＩＣ１２内にローカルで（例えば、ＩＣ１２内のローカルキャッシュメモリ内のレジスタ内に）原子カウンタを格納することができる。一例として、生成

するスレッドを実行するプログラマブル計算ユニットがデータを出力時に、ＰＭＵ１８は、内部に格納された原子カウンタの値を読み取り、生成されたデータの量に基づいて原子カウンタの値を変更し、原子カウンタの変更された値をＩＣ１２内に書き込むことができる。この例では、消費スレッドを実行するプログラマブル計算ユニットが原子カウンタの値を読み取るときに、ＧＰＵ１６は、オフチップのグローバルメモリ２０にアクセスすることによって原子カウンタの値を決定する必要がない。代わりに、ＰＭＵ１８は、原子カウンタの値を提供することができる。

【００７４】

幾つかの例では、原子カウンタの値をローカルで格納するＰＭＵ１８は、スピニングを低減させることができる。例えば、消費するスレッドを実行するプログラマブル計算ユニットは、消費するスレッドによって消費されるべきデータの要求を出力することができる。この例では、ＰＭＵ１８は、（例えば、ローカルで格納された原子カウンタの値に基づいて）消費されるべきデータが入手可能であるかどうかを決定することができる。

【００７５】

データが消費のためにまだ入手可能でないとＰＭＵ１８が決定した場合は、ＰＭＵ１８は、プログラマブル計算ユニットがまだ入手可能でないデータに依存しない（例えば、同じカーネルの又は可能な場合は異なるカーネルの）異なるスレッドに切り換わるべきであることをプログラマブル計算ユニットに示すことができる。換言すると、ＰＭＵ１８は、プログラマブル計算ユニットがその他のスレッドの実行を続けることができるようにするためまだ入手可能でないデータを必要とする消費スレッドはスリープ状態にすべきであることを示すことができる。データが入手可能であることが、原子カウンタのローカルで格納された値に基づいてＰＭＵ１８によって決定された時点で、ＰＭＵ１８は、プログラマブル計算ユニットが現在入手可能なデータを用いて消費スレッドを実行することができるようにスリープ中のスレッドに戻る（すなわち、スレッドを起こす）ようにプログラマブル計算ユニットに命令することができる。この方法により、データが消費のためにまだ入手可能でないときに、消費スレッドを実行するプログラマブル計算ユニットが、ビジー待機状態にとどまるのではなく、カーネルのその他のスレッドを実行することができる。

【００７６】

他の例として、異なるプログラマブル計算ユニットで実行中の同じカーネルの２つの生成するスレッドが、バッファ２２のうちの同じそれに同時にデータを書き込むことを試みたときには、ＰＭＵ１８は、生成するスレッドのうちの１方にアクセスを許容し、他方の生成するスレッドに対してはアクセスを拒否することができる。この例では、ＰＭＵ１８は、アクセスを拒否されたスレッドを実行しているプログラマブル計算ユニットに対してカーネルのその他のスレッドを実行するように命令することができる。バッファ２２への書き込みアクセスが可能になったことがＰＭＵ１８によって決定された時点で、ＰＭＵ１８は、アクセスが拒否されたスレッドを実行していたプログラマブル計算ユニットに対して、バッファ２２への書き込みアクセスが現在は可能であることを示すことができる。この方法により、アクセスを拒否されたスレッドを実行していたプログラマブル計算ユニットが追加のスレッドを実行することができる。

【００７７】

同様に、２つの消費するスレッドが、バッファ２２のうちの同じそれから同時にデータを読み取ることを試みたときには、ＰＭＵ１８は、消費するスレッドのうちの１方にアクセスを許容し、他方の消費するスレッドに対してはアクセスを拒否することができる。２つのスレッドが同時に書き込んでいる例と同様に、２つのスレッドが同時に読み取っているこの例では、ＰＭＵ１８は、アクセスを拒否されたスレッドを実行しているプログラマブル計算ユニットに対してその他のスレッドを実行するように命令することができる。バッファ２２への読み取りアクセスが可能になったことがＰＭＵ１８によって決定された時点で、ＰＭＵ１８は、アクセスが拒否されたスレッドを実行していたプログラマブル計算ユニットに対して、バッファ２２への読み取りアクセスが現在は可能であることを示すことができる。この方法により、アクセスを拒否されたスレッドを実行していたプログラマ

ブル計算ユニットが追加のスレッドを実行することができる。

【００７８】

この方法により、グローバルメモリ２０内のバッファ２２を定義するプロセッサ１４、及びグローバルメモリ２０内のバッファ２２の状態を管理するＰＭＵ１８は、ＧＰＵ１６によるパイプライン方式でのカーネルの効率的な実行を可能にすることができる。一例として、ＰＭＵ１８は、パイプライン方式でカーネルを実行するために必要なオフチップアクセスの数を最小限にすることができる。他の例として、ＰＭＵ１８は、データが格納されるべきアドレス又はデータが取り出されるべきアドレスを決定することができるため、ＧＰＵ１６は、該アドレスを決定するためにカーネル内で命令を実行することによって該アドレスを決定するのに電力及びクロックサイクルを浪費しないことができる。換言すると、ＰＭＵ１８は、データを格納する又は取り出す場所を決定する命令をスレッドが含むことなしにデータが格納されるべきアドレス又は取り出されるべきアドレスを決定することができる。さらに、ＰＭＵ１８は、プログラマブル計算ユニットがスピニングせずにカーネルのスレッドを実行するのを可能にすることができる。例えば、生成するカーネルからのデータをまだ入手可能でないときには、ＰＭＵ１８は、消費するカーネルのその他のスレッド（例えば、生成するカーネルからのデータを要求しないスレッド）が実行するのを可能にすることができる。

【００７９】

図２は、グラフィックス処理ユニット（ＧＰＵ）及びグローバルメモリをさらに詳細に例示したブロック図である。例えば、図２は、図１のＧＰＵ１６及びグローバルメモリ２０をさらに詳細に例示したものである。例示されるように、ＧＰＵ１６は、シェーダプロセッサ２６と、固定機能ユニット３０と、パイプライン管理ユニット（ＰＭＵ）１８と、キャッシュ３４と、スケジューラ４０と、レジスタ４４と、を含む。幾つかの例では、レジスタ４４は、キャッシュ３４の一部であることができる。図２において示される例では、ＰＭＵ１８は、ＧＰＵ１６内で形成されるとして例示される。しかしながら、上述されるように、ＰＭＵ１８は、ＧＰＵ１６の外部で、及びＧＰＵ１６と同じ集積回路内で形成することができる。

【００８０】

シェーダプロセッサ２６は、プログラマブル計算ユニット２８Ａ乃至２８Ｎ（総称してプログラマブル計算ユニット２８と呼ばれる）を含むことができ、それらは、シェーダコアとみなすことができる。固定機能ユニット３０は、固定機能計算ユニット３２Ａ乃至３２Ｎ（総称して固定機能計算ユニット３２と呼ばれる）を含む。シェーダプロセッサ２６及び固定機能ユニット３０は、プログラマブル計算ユニット２８及び固定機能ユニット３２のうちの１つ以上を含むことができる（例えば、例示されるそれらよりも多い又は少ない）。

【００８１】

プログラマブル計算ユニット２８は、上述されるように機能することができる。例えば、プログラマブル計算ユニット２８は、グラフィックスに関連するアプリケーション及びグラフィックスに関連しないアプリケーションの両方（例えば、シェーダ及びカーネル）を実行することができる。例えば、プログラマブル計算ユニット２８は、デバイス言語（例えば、ＯｐｅｎＣＬ　Ｃ言語）で書かれたカーネルを実行することができる。上述されるように、プログラマブル計算ユニット２８のうちの１つは、即座の結果を格納するための及びプログラマブル計算ユニット２８で実行中のカーネルのスレッド間で共有するためのローカルメモリを含むことができる。プログラマブル計算ユニット２８の各々のローカルメモリは、その他のプログラマブル計算ユニット２８によってアクセスすることができない。幾つかの例では、プログラマブル計算ユニット２８のうちの１つが、プログラマブル計算ユニット２８のうちの他の１つがカーネルのスレッドを実行すべき時間をスケジューリングすることが可能である。

【００８２】

幾つかの例では、プログラマブル計算ユニット２８のうちの１つは、プログラマブル計

算ユニット２８のうちの１つ以上のその他のそれらにデータを送信することができる。例えば、パイプライン方式でカーネルを実行するために、生成するスレッドを実行する、プログラマブル計算ユニット２８のうちの第１のそれは、プログラマブル計算ユニット２８の第２のそれにデータ（例えば、グラフィックスに関連しないデータ）を出力することができる。上述されるように、プログラマブル計算ユニット２８のうちの送信するそれ（例えば、生成するスレッドを実行するプログラマブル計算ユニット）は、バッファ、例えば、グローバルメモリ２０のバッファ２２のうちの１つ、にデータを格納することができ、プログラマブル計算ユニット２８のうちの受信するそれ（例えば、消費するスレッドを実行するプログラマブル計算ユニット）は、グローバルメモリ２０のバッファ２２のうちの１つからデータを取り出すことができる。

【００８３】

図２において示されるように、幾つかの例では、ＧＰＵ１６は、内部キャッシュ３４を含むことができる。しかしながら、キャッシュ３４は、ＧＰＵ１６の内部に限定されるのではなく、ＩＣ１２の内部に存在することができる。幾つかの例では、生成されたデータをオフチップで（例えば、グローバルメモリ２０内に）格納するのではなく、ＧＰＵ１６がＧＰＵ１６又はＩＣ１２の内部にデータを格納することが可能である。例えば、プログラマブル計算ユニット２８のうちの送信するそれは、図２の例ではＧＰＵ１６内に存在するキャッシュ３４内のバッファ３６Ａ乃至３６Ｎ（総称してバッファ３６と呼ばれる）のうちの１つ以上の中にデータを格納することができるが、ＩＣ１２内及びＧＰＵ１６の外部に存在することが可能である。プログラマブル計算ユニット２８の受信するそれは、キャッシュ３４内のバッファ３６からデータを取り出すことができる。キャッシュ３４内のバッファは、グローバルメモリ２０内のバッファ２２のキャッシュバックされた（ｃａｃｈｅ－ｂａｃｋｅｄ）バッファであることができる。換言すると、グローバルメモリ２０のバッファ２２は、消費するスレッドによって消費されるべき生成するスレッドによって生成された完全なデータを格納することができ、バッファ３６は、グローバルメモリ２０からデータにアクセスすることと比較して素早いアクセスを行うために生成されたデータの一部を格納するキャッシュとして機能することができる。

【００８４】

キャッシュ３４内のバッファは、バッファ２２に類似することができる。例えば、バッファ３６は、ＦＩＦＯバッファ又はリングバッファであることができる。キャッシュ３４は、オフチップメモリ（例えば、グローバルメモリ２０のバッファ２２）にアクセスすることに関連するメモリレーテンシー及び電力消費を回避するためにバッファ３６を含めるのが望ましいであろう。しかしながら、格納のために利用可能なスペースは限られているためバッファ３６のみを利用するのは実際的でない。この方法により、バッファ３６内にデータの一部を格納してバッファ２２内へのスピルオーバー（ｓｐｉｌｌ　ｏｖｅｒ）を考慮するのが可能である。

【００８５】

バッファ３６及びバッファ２２は、ＧＰＵ１６がパイプライン方式でカーネルを実行するのを可能にすることができる。例えば、バッファ３６及びバッファ２２は、プログラマブル計算ユニット２８間での通信を考慮したデータ構造であるとみなすことができる。バッファ３６及びバッファ２２は、プログラマブル計算ユニットで実行中のカーネルが出力することができる最低データ量よりも多くのデータ（例えば、２つ以上のデータユニット）を格納するように構成することができる。この方法により、プログラマブル計算ユニット２８のうちの１つで実行するカーネルのスレッドは、バッファ３６及びバッファ２２に格納されており及びプログラマブル計算ユニット２８のうちの他の１つで実行する他のカーネルのスレッドに消費のために渡すことができる可変の量のデータを生成することができる。

【００８６】

固定機能計算ユニット３２は、固定された機能を提供することができ及び（限定しない一例として）ハードウェアユニットとして形成することができる。固定機能計算ユニット

３２は、デバイス言語を用いて書かれた特定の組み込み式カーネルを実行するとみなすことができる。例えば、プログラマブル計算ユニット２８は、機能上の柔軟性を提供する一方で、固定機能計算ユニット３２は、各々の機能上の柔軟性を制限することができる。例えば、固定機能計算ユニット３２は、特定のグラフィックス機能を提供するラステライゼーションユニット（ｒａｓｔｅｒｉｚａｔｉｏｎ　ｕｎｉｔ）、プリミティブ（ｐｒｉｍｉｔｉｖｅ）アセンブリユニット、ビューポート変換ユニット、及びその他の該ユニットを含むことができる。

【００８７】

幾つかの例では、固定機能計算ユニット３２は、各々の特定の機能を実行するためにハードワイヤすることができる。さらに、固定機能計算ユニット３２は、固定機能計算ユニット３２の他の１つがいつ実行すべきかをスケジューリングすることも可能である。さらに、幾つかの事例では、ＧＰＵ１６が固定機能計算ユニット３２のうちの特定のそれを含まない場合は、利用不能な固定機能計算ユニットの機能を実行するカーネルを開発することが可能である。換言すると、カーネルは、利用不能な固定機能計算ユニットの固定機能の挙動をエミュレートすることができる。例えば、固定機能テッセレータ（ｔｅｓｓｅｌｌａｔｏｒ）を利用可能でない場合は、開発者は、テッセレータの固定機能挙動をエミュレートするテッセーションカーネルを開発し、プログラマブル計算ユニット２８のうちの１つ以上においてカーネルを実行することができる。

【００８８】

幾つかの例では、ＧＰＵ１６は、スケジューラ４０を含むことができる。スケジューラ４０は、様々なプログラマブル計算ユニット２８及び固定機能ユニット３２にスレッド及び動作を割り当てることができる。例えば、スケジューラ４０は、その他のプログラマブル計算ユニット２８が利用不足である一方でいずれも利用過剰にならないようにするためにプログラマブル計算ユニット２８によって実施されるタスクを負荷均衡化することができる。スケジューラ４０は、ハードウェアまたはハードウェアで実行するソフトウェアとして実装することができる。

【００８９】

図２において、グローバルメモリ２０は、バッファ４２Ａ乃至４２Ｎ（総称してバッファ４２と呼ばれる）を含むことができ、キャッシュ３４は、バッファ３８Ａ乃至３８Ｎ（総称してバッファ３８と呼ばれる）を含むことができる。バッファ３８は、必ずしもすべての例で存在する必要はなく、バッファ４２に格納されたコマンドに関するキャッシュバックされた格納を提供するための任意選択のオンチップキャッシュとして形成することができる。バッファ４２及びバッファ３８は、コマンド待ち行列とみなすことができる。すべてのプログラマブル計算ユニット２８に関してコマンド待ち行列（例えば、バッファ４２及びバッファ３８のうちの１つ）及び各タイプの固定機能計算ユニット３２に関して１つの待ち行列が存在することができる。バッファ４２及びバッファ３８は、ゼロ以上のエントリを格納することができる。

【００９０】

バッファ４２及び任意選択のオンチップバッファ３８は、プログラマブル計算ユニット２８及び固定機能計算ユニット３２に関する仕事量のスケジューリングを援助することができる。例えば、バッファ４２は、様々なタスクを実行することをプログラマブル計算ユニット２８及び固定機能計算ユニット３２に命令するコマンドを格納することができる。例えば、バッファ４２内の各エントリは、カーネルのスレッドを実行することを利用可能な１つ以上のプログラマブル計算ユニット２８に行わせるための情報を格納すること、及びカーネル引数値に関する情報および依存性情報を格納することができる。幾つかの例では、カーネルのスレッド間での依存性は、１つ以上のプログラマブル計算ユニット２８がカーネルを実行する前に満たす必要がある。

【００９１】

バッファ２２は、プロセッサ１４（図１）及びＧＰＵ１６の両方によってアクセス可能である。一例として、プロセッサ１４は、上述される様々なＡＰＩにより呼を用いてバッ

ファ２２にアクセスすることができる。ＧＰＵ１６は、プログラマブル計算ユニット２８で実行されたカーネルに基づいてバッファ２２にアクセスすることができる。例えば、カーネルは、生成されたデータをグローバルメモリ２０内に格納するための機能付きで開発することができる。

【００９２】

例示されるように、ＧＰＵ１６は、パイプライン管理ユニット（ＰＭＵ）１８も含むことができる。上述されるように、ＰＭＵ１８は、グローバルメモリ２０内でのバッファ２２の状態を管理することができる。さらに、ＰＭＵ１８は、キャッシュ３４内でのバッファ３６の状態を管理することができる。

【００９３】

例えば、ＰＭＵ１８は、バッファ２２及びバッファ３６の長さ及び幅を格納することによってバッファ２２及びバッファ３６の状態を管理することができ、生成されたデータを格納するために利用可能なバッファ２２及びバッファ３６の数を含む。一例として、ＰＭＵ１８は、プログラマブル計算ユニット２８で実行するカーネルの前にバッファ２２を割り当てることができ、及び、カーネルの実行終了時点でバッファ２２の割り当てを解除することができる。

【００９４】

他の例として、ＰＭＵ１８は、ヘッダポインタ、現在のオフセット、最大深さ、等に関する情報をオンチップレジスタ４４に格納することができる。幾つかの例では、ＰＭＵ１８は、テクスチャパラメータがグラフィックス処理において格納される方法と類似の方法でバッファ２２及びバッファ３６の状態情報を格納することができる。

【００９５】

バッファ２２は、いずれのバッファ２２にデータを格納し又はいずれのバッファ２２からデータを取り出すかを決定し、バッファ内でデータを格納すべき又はデータを取り出すべき格納記憶場所を決定し（例えば、アドレスを決定し）、及びプログラマブル計算ユニット２８の異なるそれらがデータ崩壊を生じさせるバッファからの情報へのアクセスを試みないようにするという意味で管理を要求することができる。ＰＭＵ１８に、該管理のタスクを負わせることができる。例えば、ＰＭＵ１８を含むＧＰＵ１６又はＰＭＵ１８を含むＧＰＵ１６を含むＩＣの場合、バッファ２２の管理は、ＩＣの外部ではなくＧＰＵ１６を含むＩＣ内にローカル化することができる。この結果、低減された電力消費量、及びプログラマブル計算ユニット２８で実行するカーネルの効率的な実行を達成させることができる。

【００９６】

一例して、ＰＭＵ１８は、原子カウンタをレジスタ４４内に格納することができる。レジスタ４４は、キャッシュ３４の一部、又はＧＰＵ１６又はＩＣ１２内のその他のメモリの一部であることができる。原子カウンタは、プログラマブル計算ユニット２８のうちの１つに関するアクセスが可能であるかどうか（例えば、読み取るためのデータが入手可能であるかどうか又は２つ以上のカーネルが同じバッファ２２から同時に書き込むこと又は読み取ることを試みているかどうか）を示すことができる。原子カウンタに基づき、ＰＭＵ１８は、２つのスレッドが同時にデータを書き込むのを試みた場合に発生する可能性があるバッファ２２のデータ崩壊を回避するためにプログラマブル計算ユニット２８のうちの１つにアクセスを適切に許容し、プログラマブル計算ユニット２８のうちのその他のそれらにアクセスを拒否することができる。幾つかの例では、ＰＭＵ１８がプログラマブル計算ユニット２８のうちの１つにアクセスを拒否したときには、ＰＭＵ１８は、アクセス（例えば、スレッド）を要求するタスクがスリープ状態になるのを許容し、及び、プログラマブル計算ユニット２８のうちの拒否されたそれがその他のタスク（例えば、スレッド）の実行を継続するのを許容することができる。プログラマブル計算ユニット２８のうちの拒否されたそれへのアクセスが可能になったときに、ＰＭＵ１８はそのタスクをウェークアップさせ、さらなる実行のためにそのタスクにデータを提供することができる。この方法により、プログラマブル計算ユニット２８は完全にアイドル状態にならず、プログラ

マブル計算ユニット２８のその他のタスクを実行することができる。

【００９７】

幾つかの例では、グローバルメモリ２０の１つのバッファ２２からデータを取り出す必要があるときには、ＰＭＵ１８は、必要なデータ以上のデータを取り出すことができる。例えば、ＰＭＵ１８は、要求されたデータの開始位置及び終了位置を決定することができる。しかしながら、ＰＭＵ１８は、要求されたデータの決定された終了位置以降にバッファ２２内に格納される追加データを取り出すことができる。ＰＭＵ１８は、バッファ３６において格納スペースが利用可能であるとＰＭＵ１８が決定したときに該追加データを取り出すことができる。上述されるように、ＰＭＵ１８は、グローバルメモリ２０内のバッファ２２、及びキャッシュ３４内のバッファ３６の両方を管理することができる。ＰＭＵ１８は、取り出されたデータをキャッシュ３４に格納することができる。この方法により、追加データが必要なときに既にＧＰＵ１６内に該データが存在している。追加データ（例えば、要求されたデータに加えてのデータ）をバッファ３６に格納することは、ＧＰＵ１６が（例えば、グローバルメモリ２０から）オフチップでデータにアクセスしなければならない回数をさらに減らすことができる。

【００９８】

データにアクセスするために、プログラマブル計算ユニット２８は、バッファにアクセスするためのポインタを利用することができる（例えば、カーネルは、ポインタを用いてデータにアクセスするように開発することができる）。幾つかの例では、ＰＭＵ１８は、プログラマブル計算ユニット２８がデータに適切にアクセスすることができるようにポインタ情報を維持することができる。例えば、プログラマブル計算ユニット２８は、バッファ２２に関する情報を要求する専用命令をＰＭＵ１８に出力することができる。該命令は、バッファ内の要素数、バッファ内に格納されているデータ量（例えば、バッファの幅）、情報の格納場所に関する情報、及びその他の該情報を含むことができる。この方法により、プログラマブル計算ユニット２８がバッファ２２に適切にアクセスするようにすることは、ＧＰＵ１６を収納するＩＣの内部で行うことができ、それは、ＧＰＵ１６を収納するＩＣ外部でのアクセスを減らすことができる。

【００９９】

一例として、データが壊れるか失われないようにするために、生成するカーネルは、バッファ２２の範囲（開始点及び終了点）を問い合わせる命令を含むように開発することができる。この例では、生成するカーネルを実行中であるプログラマブル計算ユニット２８のうちの１つは、バッファ２２の範囲の問い合わせをＰＭＵ１８に出力することができる。ＰＭＵ１８は、（プロセッサ１４がバッファ２２を定義したときにプロセッサ１４から情報を受信することによって）バッファ２２の範囲の情報をレジスタ４４に格納しておくことができる。ＰＭＵ１８は、バッファ２２の範囲の結果を生成するカーネルに戻すことができる。

【０１００】

他の例として、パイプライン方式でカーネルを実行するために、幾つかの例では、パイプライン内でのデータの順序を維持する必要がある。例えば、第１のカーネルが第２のカーネルによって消費されるべきデータを生成することになると仮定する。しかしながら、この場合は、第３のカーネルも、第１及び第２のカーネルが実行中に実行中であることが可能である。この場合は、第１のカーネルによって生成されたデータ及び第３のカーネルによって生成されたデータの順序を変更することが可能であり、その結果第２のカーネルが不正確なデータを消費することになる可能性がある。

【０１０１】

適切な順序設定を保証するために、幾つかの例では、バッファ２２がアクセスのために利用可能であるかどうかを示す原子カウンタに加えて、ＰＭＵ１８が追加の原子カウンタをレジスタ４４に格納することができる。これらの追加の原子カウンタは、デバイス－原子カウンタと呼ぶことができる。例えば、バッファ２２のうちの各々の１つと関連付けられたデバイス－原子カウンタが存在することができる。さらに、ＰＭＵ１８又はスケジュ

ーラ４０は、各カーネルの各スレッドによって生成されたデータを格納すべきバッファ２２内の相対的位置を定義するトークンをそのスレッドに割り当てるように構成することができる。スレッドに関するこのトークンは、デバイス－原子カウンタの現在値であることができる。

【０１０２】

例えば、ＰＭＵ１８は、０のトークン値を有するデータを最初に消費する第１の消費スレッドを割り当て、１のトークン値を有するデータを２番目に消費する第２の消費スレッドを割り当て、以下同様である。これらの消費スレッドの各々は、デバイス－原子カウンタの値をＰＭＵ１８に要求することができる。デバイス－原子カウンタの現在値が消費スレッドのトークン値に等しい場合は、消費するスレッドはデータを消費することができる。そうでない場合は、消費するスレッドはデータを消費することができない。

【０１０３】

トークン値がデバイス－原子カウンタの値と等しい消費スレッドがデータを消費した後は、ＰＭＵ１８は、デバイス－原子カウンタの値を更新することができる。幾つかの例では、消費するスレッドが消費するデータ量は固定させることができ、ＰＭＵ１８は、固定された量のデータがバッファ２２から取り出された後にデバイス－原子カウンタの値を更新することができる。しかしながら、幾つかの例では、消費するスレッドが消費するデータの量は、固定することができない。これらの例では、消費するスレッドがデータの受信を終了した後に、消費するスレッドは、次の消費するスレッドがデータを消費することができるようにＰＭＵ１８がデバイス－原子カウンタの値を増加すべきであることをＰＭＵ１８に示すことができる。この方法により、デバイス－原子カウンタは、ＰＭＵ１８がその値をレジスタ４４に格納すること及び更新することができ、データが消費されるべき順序が守られ、順番を狂わせてデータを受信すべきでない消費するスレッドが順番外でデータを受信しないように保証することができる。

【０１０４】

他の例として、ＰＭＵ１８は、立ち往生の機会を最小限にするためにレジスタ４４に情報を格納することができる。例えば、上述されるように、プロセッサ１４は、バッファ２２のうちの１つ内に格納するためにカーネルのスレッドによって生成することができる要素の最大数を示すバッファ２２に関する拡大係数を定義するように構成することができる。カーネルが拡大係数によって定義されるよりも多いデータを生成する場合は、カーネルは、立ち往生する（実行を停止する）可能性がある。プロセッサ１４は、拡大係数の値をＰＭＵ１８に提供することができ、ＰＭＵ１８は、拡大係数の値をレジスタ４４内に格納することができる。

【０１０５】

幾つかの例では、立ち往生する機会を最小限にするために、開発者は、拡大係数の値を要求する命令をカーネル内に含めることができる。カーネルを実行中のプログラマブル計算ユニット２８のうちの１つは、拡大係数の値の要求をＰＭＵ１８に出力することができる。代わりに、ＰＭＵ１８は、カーネルを実行中のプログラマブル計算ユニット２８のうちの１つに拡大係数の値を示すことができる。カーネルのスレッドによって生成されるデータの量が拡大係数よりも大きくなるとプログラマブル計算ユニット２８が決定した場合は、プログラマブル計算ユニット２８は、生成されたデータの量が拡大係数と等しくなった時点でカーネルの実行を停止することができ、及び、既に生成されていたデータが消費された時点でカーネルの残りのスレッドの実行をスケジューリングすることができる。

【０１０６】

立ち往生を最小限にするための上記の技法に加えて又はその代わりに、ＰＭＵ１８は、生成されたデータが拡大係数と等しくなるまでＰＭＵ１８がバッファ２２にデータを格納することができる事前スケジューリングを実装することができる。ＰＭＵ１８は、残りのデータはバッファ３６に格納することができる。換言すると、ＰＭＵ１８は、バッファ２２にデータを格納する要求が“安全な”範囲内にあるようにすること、バッファ２２にデータを格納する要求が代わりにバッファ３６に格納されるようにすることができる。

【０１０７】
図３は、本開示において説明される１つ以上の例による技法例を示したフローチャートである。図３において例示されるように、プログラマブル計算ユニット２８のうちの１つは、ＧＰＵ１６のシェーダプロセッサ２６においてカーネルの１つ以上のスレッドを実行することができる（４６）。ＩＣ１２内又はＧＰＵ１６内にあるＰＭＵ１８は、カーネルの１つ以上のスレッドに関して、ＩＣ１２の外部に存在するグローバルメモリ２０にデータを格納する又はＩＣ１２の外部に存在するグローバルメモリ２０からデータを取り出す要求をプログラマブル計算ユニット２８のうちの１つから受信することができる（４８）。
【０１０８】
ＰＭＵ１８は、データの格納又は取り出しを要求したプログラマブル計算ユニット２８のうちの１つに関してアクセスが許容可能であるかどうかを決定することができる（５０）。アクセスが不可能（５０のいいえ）である場合は、プログラマブル計算ユニット２８のうちの１つは、カーネルの追加スレッドを実行することができる（５２）。この例では、ＰＭＵ１８は、アクセスが可能になったときにプログラマブル計算ユニットのうちの１つに示すことができる。
【０１０９】
アクセスが可能である（５０のはい）場合は、プログラマブル計算ユニット２８のうちの１つは、グローバルメモリ２０内のバッファ（例えば、バッファ２２のうちの１つ）内においてデータが格納されるか又は取り出される記憶場所を決定することができる（５２）。例えば、ＰＭＵ１８は、データが格納されるか又は取り出されるグローバルメモリ２０内の記憶場所（すなわち、アドレス）を決定することができる（５４）。決定された記憶場所に基づき、ＧＰＵ１６は、グローバルメモリ２０内のバッファ２２のうちの１つ内の決定された記憶場所にデータを格納する又は決定された記憶場所からデータを取り出すことができる（５６）。
【０１１０】
幾つかの例では、バッファ２２のうちの１つ内の記憶場所を決定するために、ＰＭＵ１８は、グローバルメモリ２０内においてデータが格納されるべき又は取り出されるべき記憶場所をカーネルの１つ以上のスレッドが示さずに記憶場所を決定することができる。この方法により、カーネルは、データを格納すべき又はデータが取り出されるべきグローバルメモリ２０内の記憶場所を決定するための命令を含む必要がない。
【０１１１】
幾つかの例では、ＰＭＵ１８は、要求されたデータに加えてデータを取り出すことができる。これらの例では、ＰＭＵ１８は、追加データをキャッシュ３４に格納することができる。幾つかの例では、ＰＭＵ１８は、バッファ２２の状態情報をプロセッサ１４から受信することができる。これらの例では、ＰＭＵ１８は、受信された状態情報に基づいてデータが格納又は取り出されるべきバッファ２２のうちのそれら内の記憶場所を決定することができる。
【０１１２】
図４は、本開示において説明される１つ以上の例による他の技法例を示したフローチャートである。例示されるように、ＧＰＵ１６のシェーダプロセッサ２６の第１のプログラマブル計算ユニット（例えば、プログラマブル計算ユニット２８のうちの１つ）は、第１のスレッドを実行することができる（５８）。ＧＰＵ１６のシェーダプロセッサ２６の第２のプログラマブル計算ユニット（例えば、プログラマブル計算ユニット２８のうちの他の１つ）は、第２の異なるスレッドを実行することができる（６０）。
【０１１３】
ＧＰＵ１６を含むＩＣ１２内に存在するＰＭＵ１８は、第１のスレッドの実行によって生成されたデータを、ＩＣ１２の外部にあるグローバルメモリ２０内のバッファ（例えば、バッファ２２のうちの１つ）内に格納する要求を第１のプログラマブル計算ユニットから受信することができる（６２）。この例では、第１のスレッド（例えば、生成するスレ

ッド）の実行によって生成されたデータは、第2のスレッド（例えば、消費するスレッド）を実行する第2のプログラマブル計算ユニットによって消費されることになる。さらに、バッファは、先入れ先出し（ＦＩＦＯ）バッファ及びリングバッファのうちの１つであることができ、リングバッファは、ＦＩＦＯバッファの一例である。

【０１１４】

ＰＭＵ１８は、第１のスレッドの実行によって生成されたデータが格納されるべきバッファ内の記憶場所を決定することができる（６４）。ＩＣ１２は、第１のスレッドの実行によって生成されたデータをバッファ内の決定された記憶場所に格納することができる（６６）。第１のスレッドの実行によって生成されたデータをバッファ内の決定された記憶場所に格納するＩＣ１２は、データを格納するＩＣ、データを格納するＧＰＵ１６、及び／又はデータを格納するＰＭＵ１８を含むことが理解されるべきである。換言すると、データを格納するＩＣ１２は、ＩＣ１２又はデータを格納するＩＣ１２内のいずれかのコンポーネントを意味する。

【０１１５】

幾つかの例では、ＰＭＵ１８は、バッファ２２の状態情報をＩＣ１２内（例えば、レジスタ４４内）に格納することができる。ＰＭＵ１８は、バッファ２２の該状態情報をプロセッサ１４から受信することができる。バッファ２２の状態情報は、バッファ２２の開始アドレス、バッファ２２の終了アドレス、生成されたデータが格納されるべきバッファ２２内のアドレス、及びデータが取り出されるべきバッファ内のアドレスのうちの１つ以上を含むことができる。これらの例では、ＰＭＵ１８は、バッファ２２の格納された状態情報に基づいて第１のスレッドの実行によって生成されたデータが格納されるためのバッファ内の記憶場所を決定することができる。さらに、幾つかの例では、ＰＭＵ１８は、第１のスレッドがバッファ内でデータが格納されるべき記憶場所を示さずに第１のスレッドの実行によって生成されたデータを格納するためのバッファ内の記憶場所を決定することができる。

【０１１６】

ＰＭＵ１８は、第１のスレッドの実行によって生成されたデータの少なくとも一部を取り出すことの要求を第２のスレッドを実行する第２のプログラマブル計算ユニットから受信することもできる。ＰＭＵ１８は、第１のスレッドの実行によって生成されるデータが、第２のスレッドを実行する第２のプログラマブル計算ユニットによる消費のために取り出すために入手可能であるかどうかを決定することができる。幾つかの例では、ＰＭＵ１８は、第１のスレッドの実行によって生成されたデータを格納することの要求を第１のプログラマブル計算ユニットから受信するのと同時に、受信する前に、又は受信した後に第２のプログラマブル計算ユニットからの要求を受信することができる。

【０１１７】

第２のスレッドによって要求されたデータが、第２のスレッドを実行する第２のプログラマブル計算ユニットによる消費のために取り出し可能でないときには、ＰＭＵ１８は、第３のスレッドを実行するように第2のプログラマブル計算ユニットに指示することができる。ＰＭＵ１８は、第2のスレッドによって要求されたデータが、第2のスレッドを実行する第２のプログラマブル計算ユニットによる消費のための取り出しのために入手可能であるときを第２のプログラマブル計算ユニットに示すこともできる。ＰＭＵ１８は、第２のスレッドによって要求されたデータが第２のスレッドを実行する第２のプログラマブル計算ユニットによる消費のための取り出しのために入手可能であるときに、第２のスレッドによって要求されたデータを消費するために第２のスレッドを実行するように第２のプログラマブル計算ユニットに指示することができる。

【０１１８】

幾つの事例では、第１のスレッドは、カーネルの生成するスレッドであることができ、第２のスレッドは、同じカーネルの消費するスレッドであることができる。幾つかの事例では、第１のスレッドは、生成するカーネルのスレッドであることができ、第２のスレッドは、消費するカーネルのスレッドであることができる。

【０１１９】
図５は、図１のデバイスをより詳細に例示したブロック図である。例えば、図５は、デバイス１０をさらに示す。デバイス１０の例は、無線デバイス、携帯電話、パーソナルデジタルアシスタント（ＰＤＡ）、ビデオディスプレイを含むビデオゲームコンソール、モバイルビデオ会議ユニット、ラップトップコンピュータ、デスクトップコンピュータ。テレビセットトップボックス、タブレットコンピューティングデバイス、電子ブックリーダー、等を含み、ただしこれらに限定されない。デバイス１０は、プロセッサ１４と、ＧＰＵ１６、グローバルメモリ２０と、ディスプレイ６８と、ユーザインタフェース７０と、トランシーバモジュール７２と、を含むことができる。示される例では、ＰＭＵ１８は、ＧＰＵ１６内に形成される。幾つかの例では、ＰＭＵ１８は、ＧＰＵ１６を収納する同じＩＣ（すなちわ、ＩＣ１２）内で形成することができる。同じく例示されるように、ＣＰＵ１６は、ＩＣ１２内に常駐する。しかしながら、プロセッサ１４も、ＩＣ１２内に常駐することができる。
【０１２０】
デバイス１０は、明確化を目的として図４には示されていない追加のモジュール又はユニットを含むことができる。例えば、デバイス１０は、デバイス１０がモバイル無線電話である例において電話通信を有効にするためのスピーカーとマイクとを含むことができ、これらのいずれも図４には示されていない。さらに、デバイス１０内に示される様々なモジュール及びユニットは、デバイス１０のすべての例において必要であるわけではない。例えば、ユーザインタフェース７０及びディスプレイ６８は、デバイス１０がデスクトップコンピュータである例ではデバイス１０の外部に存在することができる。他の例として、ユーザインタフェース７０は、モバイルデバイスのタッチ感応式又はプレゼンス感応式（ｐｒｅｓｅｎｃｅ－ｓｅｎｓｉｔｉｖｅ）ディスプレイである例ではディスプレイ６８の一部であることができる。
【０１２１】
図４のプロセッサ１４、ＧＰＵ１６、ＰＭＵ１８、及びグローバルメモリ２０は、図１のプロセッサ１４、ＧＰＵ１６、ＰＭＵ１８、及びグローバルメモリ２０と同様であることができる。ユーザインタフェース７０の例は、トラックボールと、マウスと、キーボードと、その他のタイプの入力デバイスと、を含み、ただしこれらに限定されない。ユーザインタフェース７０は、タッチ画面であることもでき、及びディスプレイ６８の一部として組み入れることができる。トランシーバモジュール７２は、デバイス１０と他のデバイス又はネットワークとの間の無線又は有線通信を可能にする回路を含むことができる。トランシーバモジュール７２は、変調器と、復調器と、増幅器と、有線又は無線通信のためのその他の該回路と、を含むことができる。ディスプレイ６８は、液晶ディスプレイ（ＬＣＤ）、陰極線管（ＣＲＴ）ディスプレイ、プラズマディスプレイ、タッチ感応式ディスプレイ、プレゼンス感応式ディスプレイ、又は他のタイプの表示装置を備えることができる。
【０１２２】
１つ以上の例では、説明される機能は、ハードウェア、ソフトウェア、ファームウェア、又はそれらのあらゆる組み合わせにおいて実装することができる。ソフトウェアにおいて実装される場合は、それらの機能は、コンピュータによって読み取り可能な媒体において１つ以上の命令又はコードとして格納又は送信すること及びハードウェアに基づく処理ユニットによって実行することができる。コンピュータによって読み取り可能な媒体は、コンピュータによって読み取り可能な記憶媒体を含むことができ、それは、有形な媒体、例えば、データ記憶媒体、又は、例えば、通信プロトコルにより、１つの場所から他へのコンピュータプログラムの転送を容易にするあらゆる媒体を含む通信媒体、に対応する。このように、コンピュータによって読み取り可能な媒体は、概して、（１）非一時的である有形なコンピュータによって読み取り可能な記憶媒体又は（２）通信媒体、例えば、信号又は搬送波、に対応することができる。データ記憶媒体は、本開示において説明される技法の実装のために命令、コード及び／又はデータ構造を取り出すために１つ以上のコン

ピュータ又は１つ以上のプロセッサによってアクセスすることができるあらゆる利用可能な媒体であることができる。コンピュータプログラム製品は、コンピュータによって読み取り可能な媒体を含むことができる。

【０１２３】

一例により、及び制限することなしに、該コンピュータによって読み取り可能な記憶媒体は、希望されるプログラムコードを命令又はデータ構造の形態で格納するために使用することができ及びコンピュータによってアクセス可能であるＲＡＭ、ＲＯＭ、ＥＥＰＲＯＭ、ＣＤ－ＲＯＭ又はその他の光学ディスク記憶装置、磁気ディスク記憶装置、又はその他の磁気記憶デバイス、フラッシュメモリ、又はその他のいずれかの媒体を備えることができる。さらに、どのような接続も、コンピュータによって読み取り可能な媒体であると適切に呼ばれる。例えば、命令が、同軸ケーブル、光ファイバケーブル、より対線、デジタル加入者ライン（ＤＳＬ）、又は無線技術、例えば、赤外線、無線、及びマイクロ波、を用いてウェブサイト、サーバ、又はその他の遠隔ソースから送信される場合は、該同軸ケーブル、光ファイバケーブル、より対線、ＤＳＬ、又は無線技術、例えば赤外線、無線、及びマイクロ波、は、媒体の定義の中に含まれる。しかしながら、コンピュータによって読み取り可能な記憶媒体およびデータ記憶媒体は、接続、搬送波、信号、又はその他の一時的な媒体は含まず、代わりに、非一時的な、有形の記憶媒体を対象とすることが理解されるべきである。ここにおいて用いられるときのディスク（ｄｉｓｋ及びｄｉｓｃ）は、コンパクトディスク（ＣＤ）（ｄｉｓｃ）と、レーザディスク（ｄｉｓｃ）と、光ディスク（ｄｉｓｃ）と、デジタルバーサタイルディスク（ＤＶＤ）（ｄｉｓｃ）と、フロッピー（登録商標）ディスク（ｄｉｓｋ）と、Ｂｌｕ－ｒａｙ（登録商標）ディスク（ｄｉｓｃ）と、を含み、ここで、ｄｉｓｋは、通常は磁気的にデータを複製し、ｄｉｓｃは、レーザを用いて光学的にデータを複製する。上記の組み合わせも、コンピュータによって読み取り可能な媒体の適用範囲内に含められるべきである。

【０１２４】

命令は、１つ以上のプロセッサ、例えば、１つ以上のデジタル信号プロセッサ（ＤＳＰ）、汎用マイクロプロセッサ、特定用途向け集積回路（ＡＳＩＣ）、フィールドプログラマブルロジックアレイ（ＦＰＧＡ）又はその他の同等の集積回路又はディスクリート論理回路によって実行することができる。従って、ここにおいて用いられる場合の用語“プロセッサ”は、上記の構造又はここにおいて説明される技法の実装に適するあらゆるその他の構造のうちのいずれかを意味することができる。さらに、幾つかの態様では、ここにおいて説明される機能は、符号化および復号のために構成された専用のハードウェア及び／又はソフトウェアモジュール内において提供されること、又は組み合わされたコーデック内に組み入れることができる。さらに、技法は、１つ以上の回路又は論理素子内に完全に実装することが可能である。

【０１２５】

本開示の技法は、無線ハンドセット、集積回路（ＩＣ）又は一組のＩＣ（例えば、チップセット）を含む非常に様々なデバイス又は装置内に実装することができる。本開示では、開示される技法を実施するように構成されたデバイスの機能上の態様を強調するために様々なコンポーネント、モジュール、又はユニットが説明されるが、異なるハードウェアユニットによる実現は必ずしも要求しない。むしろ、上述されるように、様々なユニットは、適切なソフトウェア及び／又はファームウェアと関係させて、コーデックハードウェアユニット内において結合させること又は上述されるように１つ以上のプロセッサを含む相互運用的なハードウェアユニットの集合によって提供することができる。

【０１２６】

様々な例が説明されている。これらの及びその他の例は、以下の請求項の範囲内である。

以下に、本願出願の当初の特許請求の範囲に記載された発明を付記する。

［Ｃ１］　データ処理動作をパイプライン方式で実行するための方法であって、

グラフィックス処理ユニット（ＧＰＵ）のシェーダプロセッサの第１のプログラマブル

計算ユニットにおいて第１のスレッドを実行することと、
前記ＧＰＵの前記シェーダプロセッサの第２のプログラマブル計算ユニットにおいて第２のスレッドを実行することと、
前記ＧＰＵを含む集積回路（ＩＣ）内の管理ユニットを用いて、前記第１のスレッドの前記実行によって生成されたデータを前記ＩＣの外部のグローバルメモリ内のバッファ内に格納することの要求を前記第１のプログラマブル計算ユニットから受信することであって、前記第１のスレッドの前記実行によって生成された前記データは、前記第２のスレッドを実行する前記第２のプログラマブル計算ユニットによって消費されることになり、前記バッファは、先入れ先出し（ＦＩＦＯ）バッファ及びリングバッファのうちの１つを備えることと、
前記管理ユニットを用いて、前記第１のスレッドの前記実行によって生成された前記データが格納されるべき前記バッファ内の記憶場所を決定することと、
前記ＩＣを用いて、前記第１のスレッドの前記実行によって生成された前記データを前記バッファ内の前記決定された記憶場所に格納することと、を備える、方法。
［Ｃ２］　前記管理ユニットを用いて、前記バッファの状態情報を前記ＩＣ内に格納することをさらに備え、
前記バッファの前記状態情報は、前記バッファの開始アドレス、前記バッファの終了アドレス、生成されたデータが格納されるべき前記バッファ内のアドレス、及びデータが取り出されるべき前記バッファ内のアドレスのうちの１つ以上を含み、
前記バッファ内の前記記憶場所を決定することは、前記バッファの前記格納された状態情報に基づいて前記第１のスレッドの前記実行によって生成された前記データが格納されるべき前記バッファ内の前記記憶場所を決定することを備えるＣ１に記載の方法。
［Ｃ３］　前記管理ユニットを用いて、前記第１のスレッドの前記実行によって生成された前記データの少なくとも一部を取り出すことの要求を前記第２のスレッドを実行する前記第２のプログラマブル計算ユニットから受信することと、
前記管理ユニットを用いて、前記第１のスレッドの前記実行によって生成される前記データが、前記第２のスレッドを実行する前記第２のプログラマブル計算ユニットによる消費のための取り出しために入手可能であるかどうかを決定することと、をさらに備えるＣ１に記載の方法。
［Ｃ４］　前記第２のプログラマブル計算ユニットから前記要求を受信することは、前記第１のスレッドの前記実行によって生成されたデータを格納することの前記要求を前記第１のプログラマブル計算ユニットから受信するのと同時に、受信する前に、又は受信した後に前記第２のプログラマブル計算ユニットから前記要求を受信することを備えるＣ３に記載の方法。
［Ｃ５］　前記第２のスレッドによって要求された前記データが、前記第２のスレッドを実行する前記第２のプログラマブル計算ユニットによる消費のための取り出しのために入手可能でないときに、
前記管理ユニットを用いて、第３のスレッドを実行するように前記第２のプログラマブル計算ユニットに指示することと、
前記管理ユニットを用いて、前記第２のスレッドによって要求された前記データが、前記第２のスレッドを実行する前記第２のプログラマブル計算ユニットによる消費のための取り出しのために入手可能であるときに前記第２のプログラマブル計算ユニットに示すことと、
前記管理ユニットを用いて、前記第２のスレッドによって要求された前記データが前記第２のスレッドを実行する前記第２のプログラマブル計算ユニットによる消費のための取り出しのために入手可能であるときに前記第２のスレッドによって要求された前記データを消費するために前記第２のスレッドを実行するように前記第２のプログラマブル計算ユニットに指示することと、をさらに備えるＣ３に記載の方法。
［Ｃ６］　前記管理ユニットを用いて、前記第２のスレッドによって要求された前記データに加えて前記グローバルメモリからデータを受信することと、

前記管理ユニットを用いて、前記第２のスレッドによって要求された前記データに加えての前記データを前記ＩＣ内のキャッシュに格納することと、をさらに備えるＣ３に記載の方法。
［Ｃ７］　前記第１のスレッドを実行することは、カーネルの生成するスレッドを実行することを備え、前記第２のスレッドを実行することは、前記カーネルの消費するスレッドを実行することを備えるＣ１に記載の方法。
［Ｃ８］　前記第１のスレッドを実行することは、生成するカーネルの前記第１のスレッドを実行することを備え、前記第２のスレッドを実行することは、消費するカーネルのスレッドを実行することを備えるＣ１に記載の方法。
［Ｃ９］　前記ＧＰＵは、前記管理ユニットを含むＣ１に記載の方法。
［Ｃ１０］　前記バッファ内の前記記憶場所を決定することは、前記第１のスレッドの前記実行によって生成された前記データが前記バッファ内において格納されるべき前記記憶場所を前記第１のスレッドが示さずに前記データが格納されるべき前記バッファ内の前記記憶場所を決定することを備えるＣ１に記載の方法。
［Ｃ１１］　装置であって、
バッファを含むグローバルメモリであって、前記バッファは、先入れ先出し（ＦＩＦＯ）バッファ及びリングバッファのうちの１つを備えるグローバルメモリと、
集積回路（ＩＣ）であって、
グラフィックス処理ユニット（ＧＰＵ）であって、
第１のスレッドを実行するように構成された第１のプログラマブル計算ユニットと、
第２のスレッドを実行するように構成された第２のプログラマブル計算ユニットと、を備えるＧＰＵと、
前記第１のスレッドの前記実行によって生成されたデータを前記グローバルメモリ内の前記バッファ内に格納することの要求を前記第１のプログラマブル計算ユニットから受信し、及び
前記第１のスレッドの前記実行によって生成された前記データが格納されるべき前記バッファ内の記憶場所を決定するように構成された管理ユニットと、を備える、集積回路（ＩＣ）とを備え、前記第１のスレッドの前記実行によって生成された前記データは、前記第２のスレッドを実行する前記第２のプログラマブル計算ユニットによって消費され、
前記ＩＣは、前記第１のスレッドの前記実行によって生成された前記データを前記バッファ内の前記決定された記憶場所に格納するように構成される、装置。
［Ｃ１２］　前記管理ユニットは、前記バッファの状態情報を前記ＩＣ内に格納するように構成され、
前記バッファの前記状態情報は、前記バッファの開始アドレス、前記バッファの終了アドレス、生成されたデータが格納されるべき前記バッファ内のアドレス、及びデータが取り出されるべき前記バッファ内のアドレスのうちの１つ以上を含み、
前記管理ユニットは、前記バッファの前記格納された状態情報に基づいて前記第１のスレッドの前記実行によって生成された前記データが格納されるべき前記バッファ内の前記記憶場所を決定するように構成されるＣ１１に記載の装置。
［Ｃ１３］　前記管理ユニットは、
前記第１のスレッドの前記実行によって生成された前記データの少なくとも一部を取り出すことの要求を前記第２のスレッドを実行する前記第２のプログラマブル計算ユーザから受信し、及び
前記第１のスレッドの前記実行によって生成される前記データが、前記第２のスレッドを実行する前記第２のプログラマブル計算ユニットによる消費のための取り出しために入手可能であるかどうかを決定するように構成されるＣ１１に記載の装置。
［Ｃ１４］　前記管理ユニットは、前記第１のスレッドの前記実行によって生成されたデータを格納することの前記要求を前記第１のプログラマブル計算ユニットから受信するのと同時に、受信する前に、又は受信した後に前記第２のプログラマブル計算ユニットから

前記要求を受信するように構成されるＣ１３に記載の装置。
［Ｃ１５］　前記管理ユニットは、
前記第２のスレッドによって要求された前記データが、前記第２のスレッドを実行する前記第２のプログラマブル計算ユニットによる消費のための取り出しのために入手可能でないときに、第３のスレッドを実行するように前記第２のプログラマブル計算ユニットに指示し、
前記第２のスレッドによって要求された前記データが、前記第２のスレッドを実行する前記第２のプログラマブル計算ユニットによる消費のための取り出しのために入手可能であるときに前記第２のプログラマブル計算ユニットに示し、及び
前記第２のスレッドによって要求された前記データが前記第２のスレッドを実行する前記第２のプログラマブル計算ユニットによる消費のための取り出しのために入手可能であるときに前記第２のスレッドによって要求された前記データを消費するために前記第２のスレッドを実行するように前記第２のプログラマブル計算ユニットに指示するように構成されるＣ１３に記載の装置。
［Ｃ１６］　前記管理ユニットは、
前記第２のスレッドによって要求された前記データに加えて前記グローバルメモリからデータを取り出し、及び
前記第２のスレッドによって要求された前記データに加えての前記データを前記ＩＣ内のキャッシュに格納するように構成されるＣ１３に記載の装置。
［Ｃ１７］　前記第１のスレッドは、カーネルの生成するスレッドを備え、前記第２のスレッドは、前記カーネルの消費するスレッドを備えるＣ１１に記載の装置。
［Ｃ１８］　前記第１のスレッドは、生成するカーネルのスレッドを備え、前記第２のスレッドは、消費するカーネルのスレッドを備えるＣ１１に記載の装置。
［Ｃ１９］　前記ＧＰＵは、前記管理ユニットを含むＣ１１に記載の装置。
［Ｃ２０］　前記管理ユニットは、前記第１のスレッドの前記実行によって生成された前記データが前記バッファ内において格納されるべき前記記憶場所を前記第１のスレッドが示さずに前記データが格納されるべき前記バッファ内の前記記憶場所を決定することを備えるＣ１１に記載の装置。
［Ｃ２１］　前記装置は、映像デバイス、セットトップボックス、無線ハンドセット、パーソナルデジタルアシスタント、デスクトップコンピュータ、ラップトップコンピュータ、ゲームコンソール、ビデオ会議ユニット、及びタブレットコンピューティングデバイスのうちの１つを備えるＣ１１に記載の装置。
［Ｃ２２］　装置であって、
バッファを含むグローバルメモリであって、前記バッファは、先入れ先出し（ＦＩＦＯ）バッファ及びリングバッファのうちの１つを備えるグローバルメモリと、
集積回路（ＩＣ）であって、
グラフィックス処理ユニット（ＧＰＵ）であって、
第１のスレッドを実行するための手段と、
第２のスレッドを実行するための手段と、
前記第１のスレッドの前記実行によって生成されたデータを前記グローバルメモリ内の前記バッファ内に格納することの要求を前記第１のスレッドを実行するための手段から受信するための手段であって、前記第１のスレッドの前記実行によって生成された前記データは、前記第２のスレッドを実行するための前記手段によって消費される手段と、
前記第１のスレッドを実行するための前記手段によって生成された前記データが格納されるべき前記バッファ内の記憶場所を決定するための手段と、
前記第１のスレッドの前記実行によって生成された前記データを前記バッファ内の前記決定された記憶場所に格納するための手段と、を備えるグラフィックス処理ユニット（ＧＰＵ）、を備える集積回路（ＩＣ）と、を備える、装置。
［Ｃ２３］　コンピュータによって読み取り可能な記憶媒体であって、
実行されたときに、

グラフィックス処理ユニット（ＧＰＵ）のシェーダプロセッサの第１のプログラマブル計算ユニットにおいて第１のスレッドを実行し、

前記ＧＰＵの前記シェーダプロセッサの第２のプログラマブル計算ユニットにおいて第２のスレッドを実行し、

前記ＧＰＵを含む集積回路（ＩＣ）内の管理ユニットを用いて、前記第１のスレッドの前記実行によって生成されたデータを前記ＩＣの外部のグローバルメモリ内のバッファ内に格納することの要求を前記第１のプログラマブル計算ユニットから受信し、

前記管理ユニットを用いて、前記第１のスレッドの前記実行によって生成された前記データが格納されるべき前記バッファ内の記憶場所を決定し、及び

前記ＩＣを用いて、前記第１のスレッドの前記実行によって生成された前記データを前記バッファ内の前記決定された記憶場所に格納することを１つ以上のプロセッサに行わせる命令が格納されており、前記第１のスレッドの前記実行によって生成された前記データは、前記第２のスレッドを実行する前記第２のプログラマブル計算ユニットによって消費されることになり、前記バッファは、先入れ先出し（ＦＩＦＯ）バッファ及びリングバッファのうちの１つを備える、コンピュータによって読み取り可能な記憶媒体。

【図１】

図1

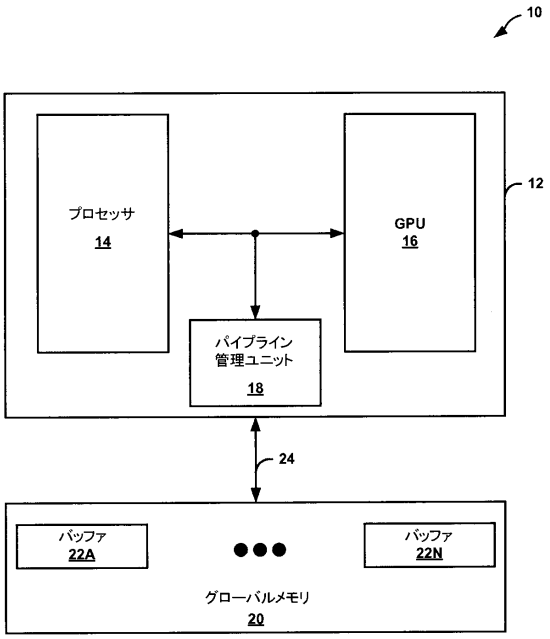

FIG. 1

【図２】

図2

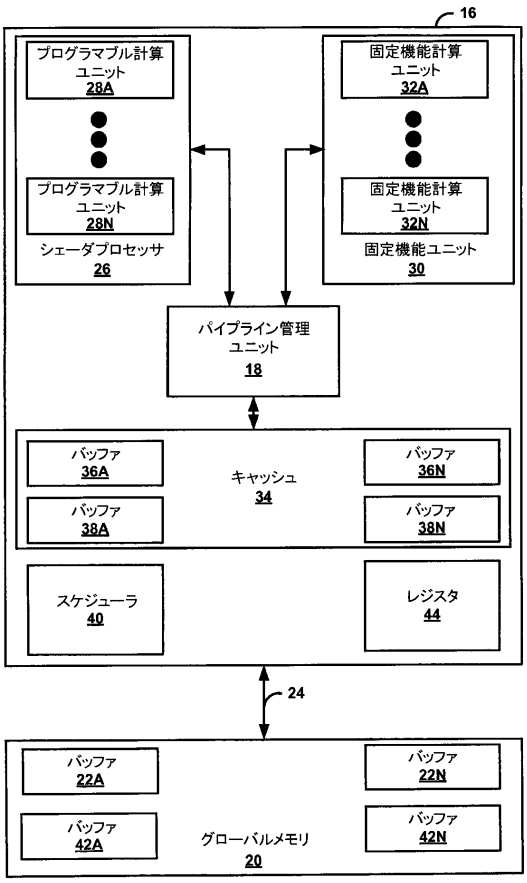

FIG. 2

【図３】

図３

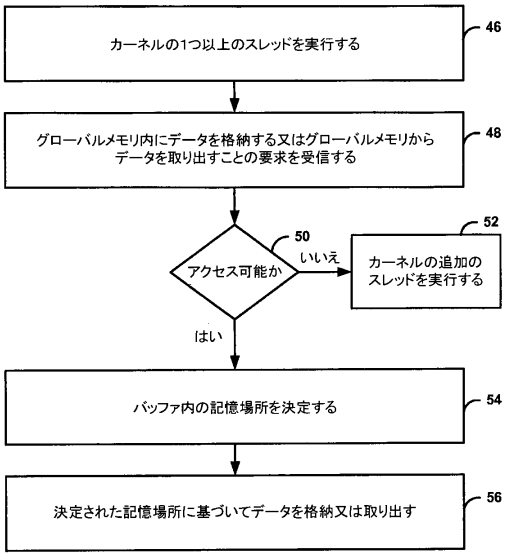

FIG. 3

【図４】

図４

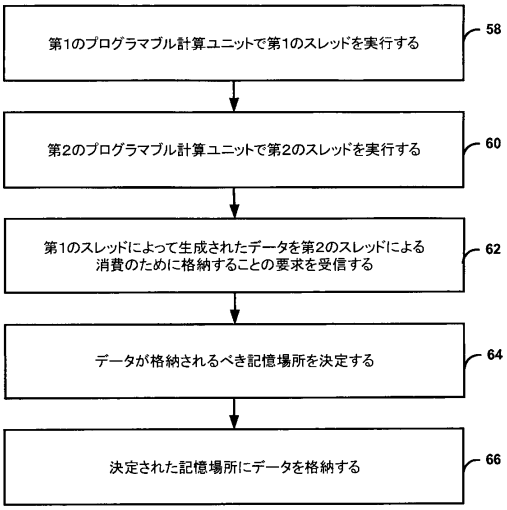

FIG. 4

【図５】

図５

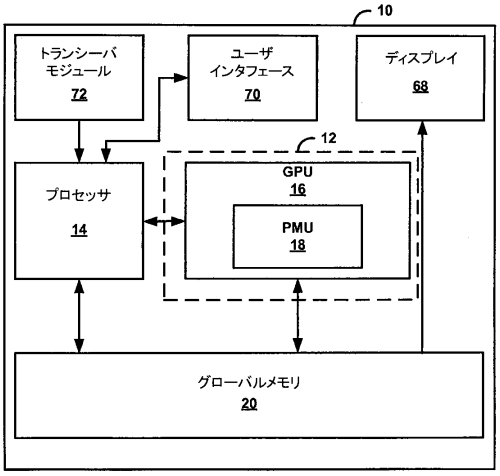

FIG. 5

フロントページの続き

(74)代理人 100153051
弁理士 河野 直樹
(74)代理人 100140176
弁理士 砂川 克
(74)代理人 100158805
弁理士 井関 守三
(74)代理人 100179062
弁理士 井上 正
(74)代理人 100124394
弁理士 佐藤 立志
(74)代理人 100112807
弁理士 岡田 貴志
(74)代理人 100111073
弁理士 堀内 美保子
(72)発明者 ボウルド、アレクセイ・ブイ.
アメリカ合衆国、カリフォルニア州 ９２１２１、サン・ディエゴ、モアハウス・ドライブ ５７７５
(72)発明者 ゴエル、ビネート
アメリカ合衆国、カリフォルニア州 ９２１２１、サン・ディエゴ、モアハウス・ドライブ ５７７５

審査官 清木 泰

(56)参考文献 特開２００７－１２２５３７（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ１５／１６－１５／１７７
Ｇ０６Ｆ ９／３０－ ９／４２
Ｇ０６Ｆ１５／８０
Ｇ０６Ｔ １／００－ １／４０
Ｇ０６Ｔ ３／００－ ５／５０
Ｇ０６Ｔ ９／００－ ９／４０
Ｇ０６Ｔ１１／００－１１／４０
Ｇ０６Ｔ１５／００－１７／００
Ｇ０６Ｔ１７／１０－１７／３０
Ｇ０６Ｆ１２／０８－１２／１２