(19)日本国特許庁(JP)

(12)特許公報(B2)

(11)特許番号
特許第7313381号
(P7313381)

(45)発行日　令和5年7月24日(2023.7.24)

(24)登録日　令和5年7月13日(2023.7.13)

(51)国際特許分類
Ｇ０６Ｆ　9/38 (2018.01)

ＦＩ
Ｇ０６Ｆ　9/38　３７０Ｃ

請求項の数　15　（全29頁）

(21)出願番号　特願2020-565488(P2020-565488)
(86)(22)出願日　令和1年5月9日(2019.5.9)
(65)公表番号　特表2021-525420(P2021-525420A)
(43)公表日　令和3年9月24日(2021.9.24)
(86)国際出願番号　PCT/US2019/031443
(87)国際公開番号　WO2019/226355
(87)国際公開日　令和1年11月28日(2019.11.28)
審査請求日　令和4年5月9日(2022.5.9)
(31)優先権主張番号　15/988,900
(32)優先日　平成30年5月24日(2018.5.24)
(33)優先権主張国・地域又は機関
米国(US)

(73)特許権者　591025439
ザイリンクス インコーポレイテッド
ＸＩＬＩＮＸ ＩＮＣＯＲＰＯＲＡＴＥＤ
アメリカ合衆国 カリフォルニア州 ９５１２４－３４００ サン ホセ ロジック ドライブ ２１００
(74)代理人　110002077
園田・小林弁理士法人
(72)発明者　ソー， ソーレン ティー.
アメリカ合衆国 カリフォルニア ９５１２４， サン ノゼ， ロジック ドライブ ２１００
(72)発明者　タルワラ， イドリス アイ.
アメリカ合衆国 カリフォルニア ９５１２４， サン ノゼ， ロジック ドライブ
最終頁に続く

(54)【発明の名称】　ハードウェアアクセラレーションのためのハードウェアリソースの埋込みスケジューリング

(57)【特許請求の範囲】
【請求項１】
集積回路であって、当該集積回路は、
ホストプロセッサによってオフロードされた操作を実行するように構成された複数の算出ユニット、及び
ハードウェアアクセラレーションのためのスケジューラ
を備え、前記スケジューラは、
複数のスロットを有し、前記複数の算出ユニットによる実行のためにホストプロセッサからオフロードされたコマンドを記憶するように構成されたコマンド待ち行列と、
前記コマンド待ち行列の前記スロットに対応するビットロケーションを有するステータスレジスタと、
前記コマンド待ち行列と前記ステータスレジスタとに結合されたコントローラと
を備え、前記コントローラが、前記コマンド待ち行列の前記スロットに記憶された前記コマンドを実行し、且つ前記コマンド待ち行列からのどのコマンドの実行が完了したかを示すために前記ステータスレジスタの前記ビットロケーションを更新するように、前記集積回路の前記複数の算出ユニットをスケジュールするように構成され、
各コマンドが、対応するコマンドを実行できる前記複数の算出ユニットのそれぞれを特定するビットマスクを有するヘッダを含み、
前記コントローラが、前記ヘッダのローカルコピーをキャッシュして、前記ヘッダの前記ローカルコピーから読み取られた選択されたコマンドの前記ビットマスクに基づいて、

選択されたコマンドを実行できる前記複数の算出ユニットのうちの１つまたは複数から選択された算出ユニットを決定し、前記選択された算出ユニットがアイドルであると決定し、実行のために、前記選択されたコマンドの引数を前記選択された算出ユニットに転送し、且つ前記選択された算出ユニットを開始するように構成された、集積回路。

【請求項２】

前記ステータスレジスタが、前記ホストプロセッサによって読み取られたことに応答して、そこに記憶されたコンテンツを消去するように構成された、請求項１に記載の集積回路。

【請求項３】

前記コマンド待ち行列に記憶された前記コマンドが、前記対応するコマンドを実行するために前記複数の算出ユニットによって使用される引数を含む、請求項１または２に記載の集積回路。

【請求項４】

前記コントローラが、前記コマンドの現在のステータスを示すために前記ヘッダの前記ローカルコピー内の値を更新するように構成され、且つ前記現在のステータスがフリーであることを示すことに応答して、前記コントローラが前記コマンドに対応する前記スロットを更新する、請求項１から３のいずれか一項に記載の集積回路。

【請求項５】

前記コントローラは、前記選択されたコンピュータユニットが前記選択されたコマンドの実行を完了したかどうかを決定するために、ポーリングモードと割込みモードとの間で切り替えるように構成された、請求項１から４のいずれか一項に記載の集積回路。

【請求項６】

前記コントローラが、プログラムコードを実行するように構成されたプロセッサである、請求項１から５のいずれか一項に記載の集積回路。

【請求項７】

前記プロセッサが、前記集積回路のプログラマブル回路を使用して実装されるソフトプロセッサである、請求項６に記載の集積回路。

【請求項８】

前記コントローラは、前記選択されたコマンドの実行が完了したと決定したことに応答して、前記選択されたコマンドを含むスロットに対応する、前記ステータスレジスタ中の前記ビットロケーションに書き込み、前記スロットがフリーであることを指示するように構成された、請求項１から７のいずれか一項に記載の集積回路。

【請求項９】

ハードウェアアクセラレーションのためにコマンドをスケジュールする方法であって、前記方法は、

集積回路内で実行されるコマンド待ち行列のスロット内に、ホストプロセッサから受信されたコマンドを記憶することであって、前記コマンドが、前記集積回路内に実装された複数の算出ユニットから選択された算出ユニットによる実行のために前記ホストプロセッサからオフロードされる、コマンドを記憶すること、

前記集積回路内に実装され且つ前記コマンド待ち行列に結合されたコントローラ内で、前記コマンドのヘッダのローカルコピーをキャッシュすることであって、前記ヘッダが、前記コマンドを実行できる前記複数の算出ユニットのうちの１つまたは複数のサブセットを示すビットマスクを含む、前記コマンドのヘッダのローカルコピーをキャッシュすること、

前記算出ユニットがアイドルであると決定することと、前記ヘッダの前記ローカルコピーから読み取られた前記ビットマスクとに基づいて、前記算出ユニットが、前記コマンドを実行できる前記複数の算出ユニットの１つまたは複数の前記サブセット内にあると決定することによって、前記算出ユニットを選択すること、

前記集積回路内に実装された前記コントローラを使用して、前記コマンド待ち行列の前記スロットに記憶された前記コマンドを実行するように前記算出ユニットをスケジュールす

ること、及び
前記コマンドの実行が完了したと決定したことに応答して、前記集積回路内に実装されたステータスレジスタ中のビットロケーションを書き込むことであって、前記ビットロケーションが、前記コマンドを記憶する前記コマンド待ち行列の前記スロットに対応する、ビットロケーションを書き込むこと
を含む、方法。
【請求項１０】
前記コントローラを使用して、前記コマンドの現在のステータスを示すために前記ヘッダの前記ローカルコピー内の値を更新すること、及び、前記現在のステータスがフリーであることを示すことに応答して、前記コントローラが前記コマンドに対応する前記スロットを更新すること
をさらに含む、請求項９に記載の方法。
【請求項１１】
前記ステータスレジスタが、前記ホストプロセッサによって読み取られたことに応答して、前記ステータスレジスタに記憶されたコンテンツを消去すること
をさらに含む、請求項９または１０に記載の方法。
【請求項１２】
前記コマンド待ち行列に記憶された絵前記コマンドが、前記コマンドを実行するために前記複数の算出ユニットによって使用される引数を含む、請求項９から１１のいずれか一項に記載の方法。
【請求項１３】
前記算出ユニットが前記コマンドの実行を完了したかどうかを決定するために、前記コントローラをポーリングモードと割込みモードとの間で切り替えること
をさらに含む、請求項９から１２のいずれか一項に記載の方法。
【請求項１４】
実行のために、前記コマンドの引数を前記算出ユニットに転送すること、及び
前記算出ユニットを開始すること
をさらに含み、
前記ビットロケーションを前記書き込むことは、前記スロットがフリーであることを指示する、
請求項９から１３のいずれか一項に記載の方法。
【請求項１５】
前記算出ユニットから割込みを受信することによって、または前記算出ユニットをポーリングすることによって、前記算出ユニットが前記コマンドの実行を完了したと決定すること
をさらに含む、請求項１４に記載の方法。
【発明の詳細な説明】
【技術分野】
【０００１】
本開示は、集積回路（ＩＣ）に関し、より詳細には、ハードウェアアクセラレーションを促進するためのハードウェアリソースの埋込みスケジューリング（ｅｍｂｅｄｄｅｄ　ｓｃｈｅｄｕｌｉｎｇ）に関する。
【背景技術】
【０００２】
異種コンピューティングプラットフォーム（ＨＣＰ：ｈｅｔｅｒｏｇｅｎｅｏｕｓ　ｃｏｍｐｕｔｉｎｇ　ｐｌａｔｆｏｒｍ）は、ホストプロセッサと１つまたは複数の他の異なる処理デバイスとを含むデータ処理システムのタイプを指す。ホストプロセッサは、一般的に、中央処理ユニット（ＣＰＵ）として実装される。ホストプロセッサは、インターフェース回路を通して他の処理デバイスに結合される。他の処理デバイスは、アーキテクチャ上、ホストプロセッサとは異なる。さらに、処理デバイスは、ホストプロセッサから

オフロードされた動作を実施することと、動作の結果をホストプロセッサにとって利用可能にすることとを行うことが可能である。

【０００３】

いくつかのＨＣＰ内では、処理デバイスは、プログラムコードを実行するように適応される。そのような処理デバイスは、一般的に、ホストとは異なる命令セットアーキテクチャを有する。これらの他のプロセッサの例は、限定はしないが、（１つまたは複数の）グラフィックス処理ユニット（ＧＰＵ）、（１つまたは複数の）デジタル信号プロセッサ（ＤＳＰ）などを含む。

【０００４】

他のＨＣＰでは、ホストプロセッサからオフロードされた動作を実施する処理デバイスは、プログラムコードをハードウェアアクセラレートする（ｈａｒｄｗａｒｅ　ａｃｃｅｌｅｒａｔｅ）ように適応されたデバイスを含む。これらの処理デバイスは、オフロードされた動作を実装する回路を含む。回路は、プロセッサ（たとえば、ＣＰＵ）によって実行されたとき、オフロードされた動作を実施することが可能であるプログラムコードと機能的に等価である。ハードウェアアクセラレーションが可能な処理デバイスの例は、フィールドプログラマブルゲートアレイ（ＦＰＧＡ）、部分的にプログラム可能な集積回路（ＩＣ）、特定用途向けＩＣ（ＡＳＩＣ）など、プログラマブルＩＣを含む。明らかに、ＨＣＰは、１つまたは複数がプログラムコードを実行するように適応され、１つまたは複数の他のものがプログラムコードをハードウェアアクセラレートするように適応された、処理デバイスの組合せを含み得る。

【０００５】

ホストプロセッサは、処理デバイスに動作をオフロードし、処理デバイスから結果を取り出すことの責任を負う。処理デバイスとの間で必要なデータを効率的に移動するホストプロセッサの能力は、ＨＣＰの全体的効率および性能に著しく影響を及ぼすことがある。

【発明の概要】

【０００６】

１つまたは複数の実施形態は、ハードウェアアクセラレーションのためのスケジューラを含む集積回路（ＩＣ）を対象とする。スケジューラは、複数のスロットを有し、ＩＣの算出ユニット（ｃｏｍｐｕｔｅ　ｕｎｉｔ）による実行のためにホストプロセッサからオフロードされたコマンドを記憶するように構成されたコマンド待ち行列を含み得る。スケジューラは、コマンド待ち行列のスロットに対応するビットロケーションを有するステータスレジスタを含み得る。スケジューラは、コマンド待ち行列とステータスレジスタとに結合されたコントローラをも含み得る。コントローラは、コマンド待ち行列のスロットに記憶されたコマンドを実行し、コマンド待ち行列からのどのコマンドを実行し終えたかを指示するためにステータスレジスタのビットロケーションを更新するように、ＩＣの算出ユニットをスケジュールするように構成され得る。

【０００７】

いくつかの実施形態では、ステータスレジスタは、読み取られたことに応答して、そこに記憶されたコンテンツを消去するように構成され得る。

【０００８】

いくつかの実施形態では、スケジューラは、通信リンクを介してホストプロセッサと通信し、コマンド待ち行列の利用可能なスロット内にコマンドを記憶するように構成されたインターフェースをさらに含み得る。

【０００９】

いくつかの実施形態では、コマンド待ち行列に記憶されたコマンドは、それぞれのコマンドを実行するために算出ユニットによって使用される引数を含み得る。

【００１０】

いくつかの実施形態では、各コマンドは、算出ユニットのうちのどれがコマンドを実行することができるかを指定し得る。

【００１１】

いくつかの実施形態では、コントローラは、コマンドを、算出ユニットに、算出ユニットのうちのどれが各それぞれのコマンドを実行することができ、アイドルであるかに基づいて割り当てるように構成され得る。

【００１２】

いくつかの実施形態では、コントローラは、プログラムコードを実行するように構成されたプロセッサであり得る。

【００１３】

いくつかの実施形態では、プロセッサは、集積回路のプログラマブル回路を使用して実装されるソフトプロセッサであり得る。

【００１４】

いくつかの実施形態では、コントローラは、選択されたコマンドがその上で稼働することができる選択された算出ユニットがアイドルであると決定し、実行のために、選択されたコマンドの引数を選択された算出ユニットに転送し、選択された算出ユニットを開始するように構成され得る。

【００１５】

いくつかの実施形態では、コントローラは、選択されたコマンドを実行し終えたと決定したことに応答して、選択されたコマンドを含むスロットに対応する、ステータスレジスタ中のビットロケーションに書き込み、スロットがフリーであることを指示するように構成され得る。

【００１６】

１つまたは複数の実施形態は、ＩＣを使用してハードウェアアクセラレーションのためにコマンドをスケジュールする方法を対象とする。本方法は、ＩＣ内のコマンド待ち行列のスロット内に、ホストプロセッサから受信されたコマンドを記憶することであって、コマンドが、ＩＣの算出ユニットによる実行のためにホストプロセッサからオフロードされる、コマンドを記憶することを含み得る。本方法は、コントローラを使用して、コマンド待ち行列のスロットに記憶されたコマンドを実行するように算出ユニットをスケジュールすることを含み得る。本方法は、コマンドを実行し終えたと決定したことに応答して、ＩＣ内のステータスレジスタ中のビットロケーションを書き込むことであって、ビットロケーションが、コマンドを記憶するコマンド待ち行列のスロットに対応する、ビットロケーションを書き込むことを含み得る。

【００１７】

いくつかの実施形態では、本方法は、コントローラ内に、コマンド待ち行列のスロットに記憶されたコマンドのヘッダのローカルにキャッシュされたコピーを維持することと、ヘッダのローカルにキャッシュされたコピー中の値を更新することによってスロットのステータスを更新することとをさらに含み得る。

【００１８】

いくつかの実施形態では、本方法は、ステータスレジスタが読み取られたことに応答して、ステータスレジスタに記憶されたコンテンツを消去することをさらに含み得る。

【００１９】

いくつかの実施形態では、コマンド待ち行列に記憶されたコマンドは、コマンドを実行するために算出ユニットによって使用され得る引数を含む。

【００２０】

いくつかの実施形態では、本方法は、各コマンドから、算出ユニットのうちのどれがコマンドを実行することができるかを決定することをさらに含み得る。

【００２１】

いくつかの実施形態では、本方法は、コマンドを、算出ユニットに、算出ユニットのうちのどれが各それぞれのコマンドを実行することができ、アイドルであるかに基づいて割り当てることをさらに含み得る。

【００２２】

いくつかの実施形態では、本方法は、選択されたコマンドがその上で稼働することがで

きる選択された算出ユニットがアイドルであると決定することと、実行のために、選択されたコマンドの引数を選択された算出ユニットに転送することと、選択された算出ユニットを開始することとをさらに含み得る。

【００２３】

いくつかの実施形態では、ビットロケーションを書き込むことは、スロットがフリーであることを指示し得る。

【００２４】

いくつかの実施形態では、本方法は、選択された算出ユニットから割込みを受信することによって、選択された算出ユニットが選択されたコマンドを実行し終えたと決定することをさらに含み得る。

【００２５】

いくつかの実施形態では、本方法は、選択された算出ユニットをポーリングすることによって、選択された算出ユニットが選択されたコマンドを実行し終えたと決定することをさらに含み得る。

【００２６】

本発明の概要セクションは、いくつかの概念を導入するために提供されるにすぎず、請求される主題の重要な、または本質的な特徴を識別するために提供されるものではない。本発明の構成の他の特徴は、添付の図面および以下の発明を実施するための形態から明らかになろう。

【００２７】

本発明の構成は、添付の図面において例として示される。しかしながら、図面は、本発明の構成を、図示される特定の実装形態のみに限定するものと解釈されるべきではない。様々な態様および利点が、以下の発明を実施するための形態を検討し、図面を参照すると明らかになろう。

【図面の簡単な説明】

【００２８】

【図１】本開示内で説明される１つまたは複数の実施形態とともに使用するための異種コンピューティングプラットフォームの一例を示す図である。

【図２】ホストプロセッサからコマンドをオフロードする例示的な方法を示す図である。

【図３】スケジューラによって実施される、ホストプロセッサからのコマンドを処理する方法を示す図である。

【図４】スケジューラによって実施される、ホストプロセッサからのコマンドを処理する別の例示的な方法を示す図である。

【図５】算出ユニットのステータスを決定する例示的な方法を示す図である。

【図６】集積回路（ＩＣ）のための例示的なアーキテクチャを示す図である。

【発明を実施するための形態】

【００２９】

本開示は、新規の特徴を定義する特許請求の範囲で締めくくるが、本開示内で説明される様々な特徴は、図面とともにその説明を考慮することにより、より良く理解されると考えられる。本明細書で説明される（１つまたは複数の）プロセス、（１つまたは複数の）機械、（１つまたは複数の）製造物およびその任意の変形形態は、例示のために提供される。本開示内で説明される特定の構造的および機能的詳細は、限定するものとして解釈されるべきではなく、単に、特許請求の範囲のための基礎として、およびほぼすべての適切に詳細な構造において説明される特徴を様々に採用するように当業者に教示するための代表的基礎として解釈されるべきである。さらに、本開示内で使用される用語および句は、限定するものではなく、むしろ、説明される特徴の理解可能な説明を提供するものである。

【００３０】

本開示は、集積回路（ＩＣ）に関し、より詳細には、ハードウェアアクセラレーションを促進するためのハードウェアリソースの埋込みスケジューリングに関する。本開示内で説明される本発明の構成によれば、ハードウェアスケジューラが、ハードウェアアクセラ

レーションのために使用されるＩＣ内に埋め込まれる。ハードウェアスケジューラは、異種コンピューティングプラットフォーム（ＨＣＰ）のホストプロセッサからの動作の、ＩＣ内のハードウェアリソースへのオフローディングを支援する。本開示内では、ハードウェアリソースは、「算出ユニット」と呼ばれる。ハードウェアスケジューラは、算出ユニットの動作を管理し、追跡することが可能である。

【００３１】

従来のＨＣＰでは、スケジューリングは、ホストプロセッサによって実行されるソフトウェアプロセスとして実装される。ホストプロセッサは、動作を実施するために算出ユニットがいつ利用可能になるかを決定するために、ＩＣの個々の算出ユニットを継続的にポーリングするというタスクを与えられる。算出ユニットのポーリングは、かなりの量のホストプロセッサ時間を消費する。さらに、スケジューラはホストプロセッサにおいて実行するので、ホストプロセッサは、ＩＣにコマンドを送る前に、算出ユニットが利用可能になるまで待たなければならない。ホストプロセッサは、コマンドを実行するために算出ユニットが利用可能になったと決定したことに応答してのみ、コマンドを送る。したがって、算出ユニットが利用可能になる時間と、算出ユニットがコマンドに対する実行を始めることが可能になる時間との間に、遅延が生じる。この遅延は、算出ユニットによって必要とされるデータを含むコマンドを、ホストプロセッサから算出ユニットを含むＩＣに転送するのに必要な時間を含む。

【００３２】

本明細書で説明される本発明の構成によれば、ハードウェアスケジューラは、算出ユニット（たとえば、ホストプロセッサからオフロードされた実際の動作を実施するハードウェアリソース）を含む同じＩＣ中で実装される。ハードウェアスケジューラは、ＩＣ内の算出ユニットの利用可能性を追跡することが可能である。ハードウェアスケジューラは、ホストプロセッサから送られたコマンド（たとえば、動作）を受信することも可能である。ハードウェアスケジューラは、算出ユニットが（１つまたは複数の）コマンドを実行するために利用可能になるような時間まで、コマンド待ち行列にコマンドを記憶することが可能である。したがって、ホストプロセッサは、利用可能な算出ユニットについて継続的にポーリングする必要なしに、および／またはＩＣにコマンドを送る前に算出ユニットが利用可能になるのを待つ必要なしに、コマンドを送り、他のタスクを続けることが可能である。

【００３３】

コマンドがハードウェアスケジューラ内で待ち行列に入れられ、算出ユニットが利用可能になると、算出ユニットへのコマンドの転送は、ホストプロセッサ関与を必要としない。ハードウェアスケジューラは、算出ユニットにコマンドを転送する。ハードウェアスケジューラと算出ユニットとは同じＩＣ中にあるので、算出ユニットにコマンドを転送するための時間は比較的小さい。したがって、算出ユニットが利用可能になる時間と、算出ユニットがコマンドの実行を始める時間との間の遅延が低減される。

【００３４】

ハードウェアスケジューラは、さらに、コマンドがいつ実行し終えたかを決定することが可能である。コマンドが実行し終えたと決定したことに応答して、ハードウェアスケジューラは、ホストプロセッサに通知することが可能である。たとえば、ハードウェアスケジューラは、コマンドが実行を終えたと決定したことに応答して、ハードウェアスケジューラ自体内に含まれるステータスレジスタに書き込むことが可能である。ステータスレジスタは、コマンドのステータスと、実行を終えた特定のコマンドとを確認するために、ホストプロセッサによって読み取られ得る。ステータスレジスタの使用は、ホストプロセッサが、ＩＣの算出ユニットの各々を個々に管理することとは対照的に「コマンド」レベルで動作することを可能にする。

【００３５】

図を参照しながら、本発明の構成のさらなる態様が以下でより詳細に説明される。例示を単純および明快にするために、図に示されている要素は、必ずしも一定の縮尺で描かれ

ているとは限らない。たとえば、要素のうちのいくつかの寸法は、明快のために、他の要素に対して誇張され得る。さらに、適切と見なされる場合、対応する、類似する、または同様の特徴を指示するために、参照番号が図の間で繰り返される。

【００３６】

図１は、異種コンピューティングプラットフォーム（ＨＣＰ）１００の一例を示す。図１の例では、ＨＣＰ１００は、ホストシステム１０２と、ハードウェアアクセラレーションデバイスとして使用されるＩＣ１５０とを含む。

【００３７】

ホストシステム１０２は、コンピュータまたはサーバなど、データ処理システムとして実装され得る。ホストシステム１０２は、インターフェース回路１１５を通してメモリ１１０および１つまたは複数の入出力（Ｉ／Ｏ）デバイスに結合されたホストプロセッサ１０５を含む。ホストシステム１０２は、メモリ１１０内にコンピュータ可読命令（プログラムコード）を記憶することが可能である。メモリ１１０は、コンピュータ可読記憶媒体の一例である。ホストプロセッサ１０５は、インターフェース回路１１５を介してメモリ１１０からアクセスされるプログラムコードを実行することが可能である。

【００３８】

メモリ１１０は、たとえば、ローカルメモリおよびバルク記憶デバイス（ｂｕｌｋ ｓｔｏｒａｇｅ ｄｅｖｉｃｅ）など、１つまたは複数の物理メモリデバイスを含み得る。ローカルメモリは、概してプログラムコードの実際の実行中に使用される（１つまたは複数の）非永続的メモリデバイスを指す。ローカルメモリの例は、ランダムアクセスメモリ（ＲＡＭ）、および／または、プログラムコードの実行中のプロセッサによる使用のために好適である様々なタイプのＲＡＭ（たとえば、ダイナミックＲＡＭまたは「ＤＲＡＭ」あるいはスタティックＲＡＭまたは「ＳＲＡＭ」）のいずれかを含む。バルク記憶デバイスは、永続的データ記憶デバイスを指す。バルク記憶デバイスの例は、限定はしないが、ハードディスクドライブ（ＨＤＤ）、ソリッドステートドライブ（ＳＳＤ）、フラッシュメモリ、読取り専用メモリ（ＲＯＭ）、消去可能プログラマブル読取り専用メモリ（ＥＰＲＯＭ）、電気的消去可能プログラマブル読取り専用メモリ（ＥＥＰＲＯＭ）、または他の好適なメモリを含む。ホストシステム１０２は、プログラムコードが実行中にバルク記憶デバイスから取り出されなければならない回数を低減するために少なくともあるプログラムコードの一時的記憶を行う１つまたは複数のキャッシュメモリ（図示せず）をも含み得る。

【００３９】

インターフェース回路１１５の例は、限定はしないが、システムバスと入出力（Ｉ／Ｏ）バスとを含む。インターフェース回路１１５は、様々なバスアーキテクチャのいずれかを使用して実装され得る。バスアーキテクチャの例は、限定はしないが、拡張業界標準アーキテクチャ（ＥＩＳＡ）バス、アクセラレーテッドグラフィックスポート（ＡＧＰ）、ビデオエレクトロニクス規格協会（ＶＥＳＡ）ローカルバス、ユニバーサルシリアルバス（ＵＳＢ）、および周辺構成要素相互接続エクスプレス（ＰＣＩｅ）バスを含み得る。

【００４０】

Ｉ／Ｏデバイス１２０は、直接、または介在するＩ／Ｏコントローラを通してのいずれかで、ホストシステム１０２、たとえば、インターフェース回路１１５に結合され得る。Ｉ／Ｏデバイス１２０の例は、限定はしないが、キーボード、ディスプレイデバイス、ポインティングデバイス、１つまたは複数の通信ポート、およびネットワークアダプタを含む。ネットワークアダプタは、ホストシステム１０２が、介在するプライベートまたは公衆ネットワークを通して他のシステム、コンピュータシステム、リモートプリンタ、および／またはリモート記憶デバイスに結合されるようになることを可能にする回路を指す。モデム、ケーブルモデム、イーサネットカード、およびワイヤレストランシーバが、ホストシステム１０２とともに使用され得る異なるタイプのネットワークアダプタの例である。

【００４１】

１つまたは複数の実施形態では、メモリ１１０は、ホストプロセッサ１０５によって実

行され得るオペレーティングシステム（図示せず）および１つまたは複数のアプリケーション（図示せず）を記憶する。ランタイムライブラリも、ホストプロセッサ１０５によって実行され得る。１つまたは複数の実施形態では、ランタイムライブラリは、ホストプロセッサ１０５によって実行される（１つまたは複数の）アプリケーションとリンクされるか、またはさもなければ、それと一体化され得る。ランタイムライブラリは、ＩＣ１５０と通信するために使用される機能を含む。

【００４２】

ホストシステム１０２は、実装されるデバイスおよび／またはシステムの特定のタイプに応じて、図示された構成要素よりも少数の構成要素、または図１に示されていない追加の構成要素を含み得る。さらに、含まれる特定のオペレーティングシステム、（１つまたは複数の）アプリケーション、および／またはＩ／Ｏデバイスは、システムタイプに基づいて変動し得る。さらに、例示的な構成要素のうちの１つまたは複数は、別の構成要素に組み込まれるか、またはさもなければ、別の構成要素の一部分を形成し得る。たとえば、プロセッサが、少なくともあるメモリを含み得る。ホストシステム１０２は、図１のアーキテクチャまたはそれと同様のアーキテクチャを使用して各々実装される単一のコンピュータあるいは複数のネットワーク化されたまたは相互接続されたコンピュータを実装するために使用され得る。

【００４３】

ＩＣ１５０は、通信リンク１２５を介してホストシステム１０２に通信可能にリンクされる。たとえば、ＩＣ１５０は、ホストシステム１０２内に含まれ得る回路板に結合され得る。１つまたは複数の実施形態では、通信リンク１２５はＰＣＩｅリンクである。ただし、通信リンク１２５は様々な異なる接続および／または通信プロトコルのうちのいずれかを使用して実装され得ることと、提供された例は限定として意図されていないこととを諒解されたい。

【００４４】

ＩＣ１５０は、通信リンク１２５を介してホストシステム１０２と通信することが可能であるインターフェース１５５を含む。特定の実施形態では、インターフェース１５５は、ダイレクトメモリアクセス（ＤＭＡ）回路を含むＰＣＩｅインターフェースである。インターフェース１５５のＤＭＡ部分は、メモリコントローラ１８５を介してコマンド待ち行列１６５、ステータスレジスタ１７０、および／またはメモリ１３０など、１つまたは複数のメモリを読み取り、および／またはそれに書き込むことが可能であり、それにより、ホストシステム１０２が、そのようなメモリを読み取り、および／またはそれに書き込むことを可能にする。

【００４５】

さらに、スケジューラ１６０が、コマンド待ち行列１６５と、ステータスレジスタ１７０と、コントローラ１７５とを含み得る。別々に示されているが、１つまたは複数の他の実施形態では、インターフェース１５５はスケジューラ１６０の一部と見なされ得る。

【００４６】

コマンド待ち行列１６５は、メモリとして実装され得る。たとえば、コマンド待ち行列１６５は、ランダムアクセスメモリ（ＲＡＭ）として実装され得る。１つまたは複数の実施形態では、コマンド待ち行列１６５は、ＩＣ１５０内に１つまたは複数のブロックＲＡＭ（ＢＲＡＭ）として実装される。コマンド待ち行列１６５は、（たとえば、ランタイムライブラリを実行する際に）ホストプロセッサ１０５によって書き込まれ、コントローラ１７５によって読み取られ得る。コマンド待ち行列１６５は、「スロット」と呼ばれる固定サイズの複数の領域に区分され得る。各スロットは、ホストプロセッサ１０５からのコマンド（たとえば、１つのコマンド）を記憶することが可能である。

【００４７】

特定の実施形態では、各スロットは同じサイズである。一態様では、スロットのサイズは、スケジューラ１６０によってハンドリングされることになる最も大きいコマンドのサイズが知られているので、スケジューラ１６０を含む回路設計のコンパイル時間において

構成され得る。スロットは、スケジューラ１６０によってハンドリングされる最も大きいコマンドを記憶するようにサイズ決定され得る。１つまたは複数の他の実施形態では、スロットのサイズは、ホストプロセッサ１０５によって実行されるランタイムライブラリによって構成され得る。いずれの場合も、ホストプロセッサ１０５は、コマンド待ち行列１６５の利用可能なスロットに新しいコマンドを書き込むことが可能である。

【００４８】

ステータスレジスタ１７０は、メモリとして実装され得る。ステータスレジスタ１７０は、コマンド待ち行列１６５中に含まれているスロットの数よりも大きいかまたはそれに等しい数のビットを含むようにサイズ決定され得る。したがって、ステータスレジスタ１７０の各ビット位置は、コマンド待ち行列１６５のスロットに対応し得る。ステータスレジスタ中のビット位置は、対応するスロットに記憶されたコマンドが実行を終えたかどうかを指示する。たとえば、ステータスレジスタ１７０中のビット位置「Ｘ」に書き込まれた「１」は、コマンド待ち行列１６５のスロット「Ｘ」に記憶されたコマンドが実行を完了したことを指示する。ホストプロセッサ１０５は、ランタイムライブラリの実行によって、コマンド待ち行列１６５からの、いずれかのコマンドが実行し終えたかどうか、および／またはどのコマンドが実行し終えたかを決定するために、コマンド待ち行列１６５中のビットを読み取ることが可能である。

【００４９】

コントローラ１７５は、コマンド待ち行列１６５内のスロット（およびそこに記憶されたコマンド）のステータスを追跡することと、算出ユニット１８０のステータスを追跡することと、コマンド待ち行列１６５からのコマンドが実行を終えたことを指示するためにステータスレジスタ１７０に書き込むこととを行うことが可能である。１つまたは複数の実施形態では、コントローラ１７５は、ＩＣ１５０中に埋め込まれ、プログラムコードを実行することが可能であるプロセッサとして実装される。特定の実施形態では、プロセッサは、ハードワイヤードである。他の実施形態では、プロセッサは、ＩＣ１５０のプログラマブル回路を使用して実装される「ソフトプロセッサ」である。コントローラ１７５は、コントローラ１７５に本明細書で説明される様々な動作を実施させるファームウェアを実行し得る。

【００５０】

算出ユニット１８０は、ハードウェアアクセラレーションのために使用される。算出ユニット１８０は、ホストシステム１０２から、より詳細にはホストプロセッサ１０５からオフロードされた動作を実施するために使用され得る。図１の例では、算出ユニット１８０は、コントローラ１７５に結合される。コントローラ１７５は、コマンド待ち行列１６５からコマンドを読み取ることと、そのコマンドを算出ユニット１８０のうちの利用可能なものに提供することとを行うことが可能である。１つまたは複数の実施形態では、コントローラ１７５は、各それぞれの算出ユニット１８０がビジーであるのかアイドルであるのかを決定するために、算出ユニット１８０をポーリングすることが可能である。１つまたは複数の他の実施形態では、算出ユニット１８０は、算出ユニットがコマンドを実行し終えたことを指示するために、コントローラ１７５への割込みを生成することが可能である。

【００５１】

１つまたは複数の実施形態では、コントローラ１７５は、オンチップ相互接続（図示せず）を通して算出ユニット１８０に結合される。インターフェース１５５も、オンチップ相互接続（図示せず）を介してコマンド待ち行列１６５に、およびステータスレジスタ１７０に結合され得る。オンチップ相互接続の一例は、アドバンストマイクロコントローラバスアーキテクチャ（ＡＭＢＡ：Ａｄｖａｎｃｅｄ Ｍｉｃｒｏｃｏｎｔｒｏｌｌｅｒ Ｂｕｓ Ａｒｃｈｉｔｅｃｔｕｒｅ）拡張可能インターフェース（ＡＸＩ：ＡＭＢＡ ｅＸｔｅｎｓｉｂｌｅ Ｉｎｔｅｒｆａｃｅ）バスである。ＡＭＢＡ ＡＸＩバスは、回路ブロックおよび／またはシステムの間にオンチップ接続を確立する際に使用するための埋込みマイクロコントローラバスインターフェースである。ＡＸＩは、インターフェースの例示的

な例として提供され、本開示内で説明される実施形態の限定として意図されていない。コントローラ１７５と算出ユニット１８０とを結合するために使用されるインターフェースの他の例は、限定はしないが、他のタイプのバス、ネットワークオンチップ（ＮｏＣ）、クロスバー、または他のタイプのスイッチを含み得る。

【００５２】

メモリ１３０は、メモリコントローラ１８５を介して算出ユニット１８０および／またはコントローラ１７５によってアクセスされ（たとえば、読み取られ、および／または書き込まれ）得るオフチップメモリとして含まれ得る。１つまたは複数の実施形態では、ホストプロセッサ１０５も、メモリコントローラ１８５を介してメモリ１３０にアクセス（たとえば、メモリ１３０を読み取り、および／またはメモリ１３０を書き込み）得る。メモリ１３０は、ＩＣ１５０が結合される回路板に結合され得る。したがって、特定の実施形態では、算出ユニット１８０は、実行されたコマンドの結果をメモリ１３０に記憶し得る。ホストプロセッサ１０５は、次いで、メモリ１３０から結果を取り出し得る。

【００５３】

説明と例示の容易さとのために、「オープンコンピューティング言語（Ｏｐｅｎ　Ｃｏｍｐｕｔｉｎｇ　Ｌａｎｇｕａｇｅ）」または「ＯｐｅｎＣＬ（商標）」という専門用語が、本出願全体にわたって使用される。ＨＣＰをサポートする様々な異なるフレームワークおよび／または言語のうちのいずれかが使用され得ることを諒解されたい。したがって、本発明の構成は、ＯｐｅｎＣＬに限定されるものではない。むしろ、本開示内で説明される実施形態は、様々な好適なおよび異なるＨＣＰフレームワークのうちのいずれかとともに使用され得る。本開示内で説明される実施形態とともに使用され得る他のＨＣＰおよび／またはハードウェアアクセラレーションフレームワークの例は、限定はしないが、オープンマルチプロセシング（Ｏｐｅｎ　Ｍｕｌｔｉ－Ｐｒｏｃｅｓｓｉｎｇ）（ＯｐｅｎＭＰ（登録商標））およびＣＵＤＡ（登録商標）を含む。

【００５４】

図２は、図１に関して説明されたＨＣＰ１００のホストプロセッサ１０５によって実施されるコマンドをオフロードする例示的な方法２００を示す。方法２００は、ＨＣＰ１００が動作している状態において始まり得る。ホストプロセッサ１０５は、少なくとも部分的に、ＩＣ１５０およびその中に含まれているスケジューラ１６０と通信するための機能を提供するランタイムライブラリの実行を通して、図２を参照しながら説明される動作を実施することが可能である。１つまたは複数の実施形態では、ホストプロセッサ１０５は、ランタイムライブラリを実行する際に、専用コマンドスレッドを実行する。ホストプロセッサ１０５は、コマンドスレッドを実行する際に、図２で説明される動作を実施することが可能である。

【００５５】

ブロック２０５において、ホストプロセッサは、カーネルイベントが受信されたかどうかを決定することが可能である。カーネルイベントは、ホストプロセッサによって実行されるアプリケーションから受信され得、ハードウェアアクセラレータへの機能のオフローディングを要求する。カーネルイベントが受信された場合、方法２００はブロック２１０に進む。カーネルイベントが受信されなかった場合、方法２００はブロック２０５をループし続け、カーネルイベントの発生を待つ。

【００５６】

たとえば、ホストプロセッサは、カーネル呼出しに関連する各ワークグループについてスケジューラのコマンド待ち行列にコマンドを書き込むことが可能である。別個のカーネル呼出し（たとえば、カーネルイベント）が、並行して稼働することが可能である。ホストプロセッサは、実行されるべきより多くのワークグループがあるかどうか、またはすべてのワークグループがその特定のカーネルイベントについてスケジュールされたかどうかを決定するために、各カーネルイベントを検査することが可能である。

【００５７】

ブロック２１０において、ホストプロセッサは、カーネルイベントについてスケジュー

ルされるべきそれ以上のワークグループがあるかどうかを決定することが可能である。ホストプロセッサは、たとえば、カーネルイベントによって表される、実施されるべき機能を、１つまたは複数のワークグループに区分することが可能である。処理されるべき１つまたは複数のワークグループが残っている場合、方法２００はブロック２１５に進む。処理されるべきさらなるワークグループが残っていない場合、方法２００はブロック２４０に進む。

【００５８】

概して、ホストプロセッサは、機能が、実行されているホストプロセッサアプリケーションに従って実行される準備ができたときに直ちに、そのような機能をスケジュールすることが可能である。説明されるように、ホストプロセッサは、算出ユニットが準備ができているかどうかを検査するためにポーリングを実施する必要がない。同様に、ホストプロセッサは、コンピューティングユニットを開始することを担当しない。

【００５９】

ブロック２１５において、ホストプロセッサは、コマンド待ち行列中のスロットを取得するために、スケジューラのステータスレジスタを読み取ることが可能である。述べられたように、ステータスレジスタ中の各ビットがコマンド待ち行列中のスロットに対応する。したがって、読み取られたときに「１」の値を有するステータスレジスタのビット位置は、ビット位置に対応するコマンド待ち行列のスロットに記憶されたコマンドが実行を完了したことを指示する。したがって、そのスロットは、フリーであるか、または、ホストプロセッサから新しいコマンドを受信するために利用可能である。

【００６０】

ブロック２２０において、ホストプロセッサは、コマンド待ち行列中のスロットが利用可能であるかどうかを決定する。たとえば、ホストプロセッサがステータスレジスタから１つまたは複数の「１」値を読み取った場合、ステータスレジスタから読み取られた「１」値の数に等しい数の、コマンド待ち行列内のスロットが利用可能である。さらに、ホストプロセッサは、読み取られた「１」値のビット位置に基づいて、コマンド待ち行列のどのスロットが利用可能であるかを知る。コマンド待ち行列中のスロットが利用可能である場合、方法２００はブロック２２５に進む。ホストプロセッサが、ステータスレジスタを読み取った後に、コマンド待ち行列中のどのスロットも利用可能でない（たとえば、ステータスレジスタがすべて「０」値を含んでいる）と決定した場合、方法２００は、処理を続けるためにブロック２０５にループバックし得る。ホストプロセッサは、ループし、コマンド待ち行列内のスロットを取得することを試み続けて、受信されたイベントを処理し得る。特定の実施形態では、コマンド待ち行列は、算出ユニットよりも多くのスロットを含み得る。この点について、算出ユニットの利用可能性は、コマンド待ち行列が、さらなるコマンドを記憶するための余地（たとえば、フリースロット）を有するかどうかに関して決定的でない。

【００６１】

１つまたは複数の実施形態では、ステータスレジスタは、読取り時に消去する（ｃｌｅａｒ－ｏｎ－ｒｅａｄ）ように実装される。したがって、スケジューラは、ホストプロセッサがステータスレジスタを読み取ることなしに複数回ステータスレジスタに書き込むことが可能である。ステータスレジスタが、たとえば、ホストプロセッサによって、読み取られたことに応答して、ステータスレジスタは、そこに記憶されたコンテンツを自動的に消去するように構成される。

【００６２】

ブロック２２５において、ホストプロセッサは、コマンド待ち行列内のフリースロットについてのアドレスを算出することが可能である。たとえば、ステータスレジスタを読み取った後に、ホストプロセッサは、コマンド待ち行列中のどのスロットが利用可能であるかを知る。各スロットが固定サイズを有するので、ホストプロセッサは、コマンドを記憶するためのコマンド待ち行列中の利用可能なスロットについてのアドレスを算出することが可能である。１つまたは複数の実施形態では、ホストプロセッサは、コマンドを記憶す

るためのコマンド待ち行列中の第１の利用可能なスロットを選定することが可能である。その場合、ホストプロセッサは、コマンド待ち行列中の第１の利用可能なスロットについてのアドレスを算出する。

【００６３】

ブロック２３０において、ホストプロセッサは、次のワークグループのためのコマンドを作成することが可能である。コマンドは、ヘッダと、１つまたは複数の引数とを含み得る。コマンドのヘッダは、コマンドのタイプを指示するオペコードと、ＩＣの算出ユニットのうちのどれがコマンドを実行するために使用され得るかを指定するビットマスクとを含み得る。コマンドの引数は、レジスタマップと呼ばれるコマンドの一部分中に含まれ得る。コマンドの引数は、コマンドを実行する際に（１つまたは複数の）算出ユニットによって処理されるデータである。異なるタイプのコマンドは、様々なサイズの、異なる数の引数を含み得る。

【００６４】

１つまたは複数の実施形態では、ホストプロセッサは、各コマンド内に１つのワークグループを含む。１つまたは複数の他の実施形態では、ホストプロセッサは、コマンド中に２つ以上のワークグループを含めることが可能である。

【００６５】

例示的な非限定的な例として、コマンドの１つのタイプは、ＳＴＡＲＴ＿ＫＥＲＮＥＬコマンドである。ＳＴＡＲＴ＿ＫＥＲＮＥＬコマンドは、コマンドのタイプを指示する一意のオペコードを有する。各カーネルイベントに応答して、ホストプロセッサは、カーネルイベントの各ワークグループについてＳＴＡＲＴ＿ＫＥＲＮＥＬコマンドを生成することが可能である。述べられたように、他の実施形態では、ＳＴＡＲＴ＿ＫＥＲＮＥＬコマンドは、２つ以上のワークグループを含み得る。ＳＴＡＲＴ＿ＫＥＲＮＥＬコマンドは、オペコードと、どの算出ユニットがコマンドを実行するために使用され得るかを指示する算出ユニットビットマスクと、引数を含むレジスタマップ部分とを含む。引数は、所与のコマンドについて算出ユニットを開始するのに必要なデータを含む。

【００６６】

ＳＴＡＲＴ＿ＫＥＲＮＥＬコマンド以外の他のコマンド（たとえば、オペコード）が使用され得ることを諒解されたい。そのような他のコマンドは、スケジューラによってサポートされる様々な異なる目的のためのものである。たとえば、ホストプロセッサは、算出ユニット実行進捗のデバッギングを実装し、ＩＣ内のハードウェアリソースに関する情報を取り出し、および／または電力監視を実施する、スケジューラが実行するためのコマンドを生成し得る。

【００６７】

ブロック２３５において、ホストプロセッサは、ブロック２３０において生成されたコマンドをスケジューラのコマンド待ち行列に書き込むことが可能である。たとえば、ホストプロセッサは、ブロック２２５において算出されたアドレスを有するコマンド待ち行列中のスロットに、通信リンクを介してコマンドを書き込むことが可能である。したがって、コマンドは、コマンド待ち行列上に効果的にプッシュされる。コマンド待ち行列が利用可能なスロットを有する限り、ホストプロセッサは、スロット内にコマンドを記憶し続け得る。

【００６８】

ブロック２４０において、処理すべきさらなるワークグループがない場合、ホストプロセッサは、カーネルイベントに関連するワークグループが実行を終えたかどうかを検査することが可能である。スケジューラは、実行を終えたコマンドを記憶するコマンド待ち行列のスロットに対応するステータスレジスタ中のビットロケーションに「１」の値を書き込むことによって、コマンドが実行を終えたことをホストプロセッサに通知することが可能である。

【００６９】

１つまたは複数の実施形態では、ホストプロセッサは、ステータスレジスタをポーリン

グすること（たとえば、ステータスレジスタを周期的に読み取ること）が可能である。１つまたは複数の他の実施形態では、スケジューラは、ステータスレジスタが書き込まれたことと、ホストプロセッサがステータスレジスタを読み取るべきであることとを指示するホストプロセッサへの割込みを生成することが可能である。

【００７０】

いずれの場合も、ブロック２４０において、ホストプロセッサは、カーネルイベントのためのワークグループのためのコマンドが実行し終えたかどうかを決定するために、ステータスレジスタを読み取ることが可能である。ホストプロセッサが、ワークグループが実行を完了しなかったと決定した場合、方法２００は、処理を続けるためにブロック２０５にループバックし得る。ホストプロセッサが、ワークグループが実行を完了したと決定した場合、方法２００はブロック２４５に進み得る。ブロック２４５において、ホストプロセッサは、カーネルイベントを完了しているものとしてマークすることが可能である。

【００７１】

１つまたは複数の実施形態では、ホストプロセッサがステータスレジスタを読み取るたびに、ホストプロセッサは、フリーであるスロットのステータスと完了したコマンドとを記憶することが可能である。この点について、図２は、例示のために提供される。説明される動作の特定の順序は、ホストプロセッサがスロットのステータスを記憶することが可能であるので、変動し得る。コマンド待ち行列のスロットは、スロットを最後に占有したコマンドが処理または実行されるまで、さらなるコマンドを記憶するために再使用され得ない。

【００７２】

図３は、図１に関して説明されたスケジューラ１６０によって実施される、ホストプロセッサからのコマンドを処理する方法３００を示す。方法３００は、ホストプロセッサからのコマンドを処理するためにスケジューラによって実施され得る動作の簡略化されたバージョンである。たとえば、図３は、図４を参照しながらより詳細に説明されるように使用され得るコマンド待ち行列中のスロットの異なるステータスの各々を示さない。

【００７３】

方法３００は、ホストプロセッサがＩＣ１５０にコマンドを送った状態において始まり得る。ブロック３０５において、コマンドが、ホストプロセッサから受信され、コマンド待ち行列のスロットに記憶される。たとえば、ＩＣ内のインターフェースは、通信リンクを介してホストプロセッサからコマンドを受信することが可能である。インターフェースは、ホストプロセッサによって指定されたコマンド待ち行列のアドレスにおいてコマンドを記憶する。説明されるように、ホストプロセッサは、コマンド待ち行列のスロットのうちのどれがフリーであるかを決定することと、フリースロットのアドレスへのコマンドの書込みを始動することとを行うことが可能である。

【００７４】

ブロック３１０において、スケジューラのコントローラは、新しいコマンドを検出することが可能である。１つまたは複数の実施形態では、コントローラは、コマンド待ち行列中のスロットの各々を検査することと、書き込まれたときにホストプロセッサからの新しいコマンドを検出することとを行うことが可能である。特定の実施形態では、コマンド待ち行列のスロットは、４つの異なる状態のうちのいずれかにおいて存在し得る。たとえば、スロットは、フリーであるか、新しいか、待ち行列中であるか、または稼働中であり得る。コントローラがスロット中の新しいコマンドを検出したとき、コントローラは、コマンド待ち行列からコマンドのヘッダを読み取ることと、コントローラ内でヘッダをローカルにキャッシュすることとを行うことが可能である。コントローラがスロットのステータス（たとえば、および／またはスロットに記憶されたコマンド）を変更したとき、コントローラは、処理時間を低減するために、コマンド待ち行列に記憶されたヘッダとは対照的に、ローカルにキャッシュされたヘッダを更新することが可能である。

【００７５】

ブロック３１５において、コントローラは、ＩＣ内の算出ユニットの利用可能性を追跡

することが可能である。たとえば、コントローラは、算出ユニットがビジーであるのかアイドルであるのかを決定することが可能である。ブロック３２０において、コントローラは、コマンド待ち行列に記憶されたコマンドを実行するために、利用可能である、たとえばアイドルである算出ユニットをスケジュールすることが可能である。１つまたは複数の実施形態では、スケジュールされた算出ユニットは、アイドルであるものと、また、コマンドのヘッダ内に含まれるビットマスクに従ってコマンドを実行することを可能にされるものとである。ブロック３２５において、コントローラは、算出ユニットにコマンドを提供し、算出ユニットを開始することが可能である。ブロック３２５において、たとえば、コントローラは、算出ユニットにコマンドの引数を提供し、算出ユニットを開始する。

【００７６】

ブロック３３０において、コントローラが、算出ユニットがコマンドの実行を終えたと決定したことに応答して、コントローラは、実行し終えたコマンドが読み取られたコマンド待ち行列のスロットに対応するステータスレジスタ中のビットロケーションに「１」の値を書き込むことが可能である。たとえば、実行し終えたコマンドがコマンド待ち行列のスロット３に記憶された場合、コントローラは、コマンド待ち行列のスロット３に対応するステータスレジスタ中のビット位置に「１」の値を書き込む。ブロック３３５において、コントローラは、スロットのステータスをフリーに更新することが可能である。述べられたように、コントローラは、コマンドのヘッダのローカルにキャッシュされたコピーを更新することによって、スロットのステータスを更新し得る。

【００７７】

図４は、図１に関して説明されたスケジューラ１６０によって実施される、ホストプロセッサからのコマンドを処理する別の例示的な方法４００を示す。方法４００は、ホストプロセッサからのコマンドを処理するためにスケジューラによって実施され得る動作のより詳細なバージョンを示す。方法４００は、コントローラがコマンド待ち行列のスロットをループすることが可能である一例を示す。方法４００は、コントローラが、どのように、コマンド待ち行列のスロットを追跡し、それに応じてスロットのステータスを更新するかをさらに示す。

【００７８】

ブロック４０５において、コントローラは、処理すべきさらなるスロットがあるかどうかを決定することが可能である。処理されるべき１つまたは複数のスロットが残っていると決定したことに応答して、方法４００はブロック４１０に進む。他の場合、方法４００は終了し得る。方法４００は、周期的にまたは特定のイベントに応答して、コマンド待ち行列のスロットを処理するために、コントローラによって新たに開始され得ることを諒解されたい。

【００７９】

ブロック４１０において、コントローラは、スロット、たとえば、現在選択されているスロットについてのヘッダのステータスを検査することが可能である。コントローラは、たとえば、スロットについてのキャッシュされたヘッダに記憶された値を読み取ることが可能である。コントローラによって検査された値は、スロットのステータスを、フリーであるか、新しいか、待ち行列中であるか、または稼働中であるものとして指示する。

【００８０】

スロットがフリーであることを指示する値に応答して、方法４００はブロック４１５に進む。ブロック４１５において、コントローラは、スロットについてのヘッダをコマンド待ち行列から直接読み取ることが可能である。説明されるように、ヘッダを読み取る際に、コントローラは、スロットについてのヘッダの新しいローカルにキャッシュされたコピーを記憶することが可能である。ブロック４２０において、コントローラは、新しいコマンドがスロット中で受信されたかどうかを決定する。一例では、コントローラは、新しいコマンドがスロットに記憶されたかどうかを決定するために、新たにキャッシュされたヘッダを前のキャッシュされたヘッダと比較することが可能である。

【００８１】

新しいコマンドがコマンド待ち行列のスロットに記憶されたと決定したことに応答して、方法４００はブロック４２５に進む。スロットが新しいコマンドを記憶していないと決定したことに応答して、方法４００は、さらなるスロットを処理し続けるためにブロック４０５にループバックする。ブロック４２５において、コントローラは、スロットが新しいことを指示するように、ヘッダのローカルにキャッシュされたコピー中の値を設定することが可能である。

【００８２】

方法４００は、コントローラが、（ブロック４１０における）スロットのステータスが新しいと決定した場合、ブロック４２５からまたはブロック４１０からブロック４３０に進み得る。ブロック４３０において、コントローラは、ＩＣの算出ユニットのうちのどれがコマンドを実行するために使用され得るかを決定する。たとえば、コントローラ内でキャッシュされたコマンドのヘッダは、どの算出ユニットがコマンドを実行するために使用され得るかを指示する１つまたは複数のビットを含み得る。コントローラは、算出ユニットのうちのどれ（たとえば、特定の算出ユニットおよび算出ユニットの数）がコマンドを実行することが可能である（たとえば、コマンドを実行することを可能にされる）かを決定するために、これらのビットを読み取ることが可能である。

【００８３】

１つまたは複数の実施形態では、コマンドによって使用され得る算出ユニットは、コマンドのヘッダ内でビットマスクとして符号化され得る。ビットマスクは、コントローラによって維持される算出ユニットインデックスに対応する位置においてビットを含み得る（たとえば、ここで、各算出ユニットが、対応する算出ユニットインデックスによって識別され得る）。１つまたは複数の実施形態では、算出ユニットアドレスは、非連続であり、任意のアドレス範囲だけ分離され得る。したがって、コントローラは、算出ユニットインデックスを算出ユニットについてのアドレスにマッピングするルックアップテーブルで構成され得る。コントローラは、算出ユニットインデックスを、算出ユニットインデックスによって識別された算出ユニットについてのアドレスにマッピングするためにルックアップテーブルを使用することによって、ビットマスクによって指定された算出ユニットのアドレスを決定することが可能である。

【００８４】

１つまたは複数の他の実施形態では、算出ユニットアドレスは、連続であり得、固定アドレス範囲だけ分離され得る。その場合、コントローラは、コマンドのヘッダから読み取られたビットマスクに基づいて、コマンドを実行するために使用され得る算出ユニットの各々のアドレスを決定することが可能である。特定の実施形態では、固定アドレス範囲は４ｋであり得る。その場合、コントローラは、インデックスを１２（４ｋ）ビットだけ左側にシフトすることによって、算出ユニットアドレスを決定し得る。

【００８５】

ブロック４３５において、コントローラは、コマンドの（１つまたは複数の）引数のアドレスを決定する。特定の実施形態では、各算出ユニットは、引数が書き込まれ得るレジスタマップ（たとえば、レジスタマップインターフェース）を有する。算出ユニットのレジスタマップは、コマンドのペイロードに記憶されたコマンドのレジスタマップ部分との１対１の対応を有し得る。ブロック４３５において、コントローラは、コマンドのペイロード中のレジスタマップのロケーションにコマンド待ち行列にオフセットすることによって、コマンドのレジスタマップのアドレスを決定することが可能である。

【００８６】

ブロック４４０において、コントローラは、コマンドの引数のサイズを読み取ることが可能である。ブロック４４５において、コントローラは、スロットが待ち行列中であることを指示するようにヘッダ値を設定することが可能である。説明されるように、コントローラは、ヘッダのローカルにキャッシュされたコピー内のヘッダ値を更新することと、実際のコマンド待ち行列のスロット内のヘッダを、そのままにすることとを行うことが可能である。

【００８７】
方法４００は、コントローラが、スロットのステータスが待ち行列中であると（ブロック４１０において）決定した場合、ブロック４４５からまたはブロック４１０からブロック４５０に進み得る。スロットの待ち行列中ステータスは、スロットが、算出ユニットによって実施される準備ができているコマンドを含むことを意味する。したがって、ブロック４５０において、コントローラは、ビットマスクに従ってコマンドを実行するために使用され得る（１つまたは複数の）算出ユニットのステータスを決定することが可能である。ビットマスクによって指定された（１つまたは複数の）算出ユニットがビジーであると決定したことに応答して、方法４００は、現在のスロット内のコマンドが算出ユニットにオフロードされないことがあるので、コマンド待ち行列の別のスロットに対して処理するために、ブロック４０５にループバックする。（１つまたは複数の）算出ユニットのうちの１つまたは複数がアイドルであると決定したことに応答して、方法４００はブロック４５５に進む。

【００８８】
ブロック４５５において、コントローラは、ブロック４５０において決定された算出ユニットに、現在処理されているスロットのコマンドの引数を書き込む。１つまたは複数の実施形態では、コントローラは、ブロック４５０において決定された算出ユニットのうちの第１の利用可能な算出ユニットを選択する。たとえば、コントローラは、コマンド待ち行列中のコマンドのレジスタマップ（たとえば、ペイロード）からの（１つまたは複数の）引数を、選択された算出ユニットのレジスタマップに転送するために、メモリコピーを実施することが可能である。コントローラは、たとえば、算出ユニットのアドレス（たとえば、ブロック４３０に関して説明されたように決定された算出ユニットのレジスタマップについてのベースアドレス）に書き込むことによって、コマンドのレジスタマップのコンテンツを算出ユニットにコピーし得る。

【００８９】
ブロック４６０において、算出ユニットにコマンドの引数を書き込んだ後に、コントローラは、算出ユニットを開始することが可能である。ブロック４６５において、コントローラは、稼働中のステータスを指示するように、コマンドのヘッダのローカルにキャッシュされたコピー内の値を設定することが可能である。稼働中ステータスは、コマンドが１つまたは複数の算出ユニットによって現在実行されていることを指示する。

【００９０】
方法４００は、コントローラが、スロットのステータスが稼働中であると（ブロック４１０において）決定した場合、ブロック４６５からまたはブロック４１０からブロック４７０に進み得る。図４は、コントローラが、各算出ユニットがビジー状態にあるのかアイドル状態にあるのかを決定するために算出ユニットをポーリングするように構成された例示的な実装形態を示す。したがって、ブロック４７０において、コントローラは、算出ユニットのステータスを決定する。コントローラは、たとえば、算出ユニットがビジーであるのかアイドルであるのかを決定するために算出ユニットをポーリングする。応答して、算出ユニットは、現在のステータスをビジーまたはアイドルとして指示する信号を提供し得る。

【００９１】
算出ユニットがビジーであると決定したことに応答して、方法４００は、コマンド待ち行列の次のスロットのステータスを検査するために、ブロック４０５にループバックする。算出ユニットがアイドルであると決定したことに応答して、コントローラは、ステータスレジスタを更新することによってホストプロセッサに通知することが可能である。コントローラは、たとえば、算出ユニットによって実行されたコマンドを含むスロットに対応するステータスレジスタのビット位置に「１」の値を書き込み得る。ブロック４８０において、コントローラは、フリーのステータスを指示するようにヘッダ値を設定する。説明されるように、コントローラは、実際のコマンド待ち行列内のヘッダとは対照的に、コマンドのヘッダのローカルにキャッシュされたコピー内のヘッダ値を更新することが可能で

ある。

【００９２】

図５は、算出ユニットのステータスを決定する例示的な方法５００を示す。１つまたは複数の実施形態では、算出ユニットは、ステータスの変化を指示するために、割込みを生成するように構成され得る。そのような場合、コントローラは、ステータスを決定するために算出ユニットをポーリングする必要がない。図５は、算出ユニットがそのような割込みを生成するように構成された実施形態においてコントローラによって実施され得る動作を示す。図５の方法５００は、たとえば、算出ユニットが割込みを生成するように構成され、コントローラがステータスのために算出ユニットをポーリングしない実施形態において、図４の例におけるブロック４７０、４７５および４８０の代わりに使用され得る。

【００９３】

ブロック５０５において、コントローラは、算出ユニットからの割込みが受信されたかどうかを決定する。算出ユニットによって生成された割込みは、たとえば、算出ユニットが動作を完了し、アイドル状態に入ったことを指示し得る。割込みが算出ユニットから受信されなかったと決定したことに応答して、方法５００は、算出ユニットからの受信された割込みについて検査し続けるために、ブロック５０５にループバックし得る。算出ユニットからの割込みが受信されたと決定したことに応答して、方法５００はブロック５１０に進み得る。

【００９４】

ブロック５１０において、コントローラは、どの算出ユニットが割込みをトリガしたかを決定する。１つまたは複数の実施形態では、コントローラは、動作を完了した各算出ユニットについてのビットセットをもつビットマスクを受信する割込みハンドラルーチンを実行することが可能である。したがって、割込みハンドラルーチンは、１つのコールにおいて、ビットマスクにおいて指定された算出ユニットの各々を処理することが可能である。方法５００は２つ以上の算出ユニットからの割込みを処理し得るが、例示のために、算出ユニットからの単一の割込みのハンドリングが説明される。

【００９５】

ブロック５１５において、コントローラは、ブロック５０５において割込みが検出された特定のライン上で割込みを無効にすることが可能である。ブロック５２０において、コントローラは、割込みを消去するために、割込みを生成した算出ユニット中のステータスレジスタを消去することが可能である。ブロック５２５において、コントローラは、算出ユニットによって実行されるコマンドを記憶するコマンド待ち行列のスロットが今フリーであることを指示するために、算出ユニットによって実行されるコマンドのヘッダのローカルにキャッシュされたコピー内のヘッダ値を更新することが可能である。

【００９６】

ブロック５３０において、コントローラは、コマンド待ち行列の対応するスロットに記憶されたコマンドが実行を終えたことを指示するために、ステータスレジスタ内の適切なビットを更新することが可能である。説明されるように、ステータスレジスタに書き込むことは、スケジューラが、コマンドが実行を完了したことをホストシステムに通知することを可能にする。ブロック５３５において、コントローラは、算出ユニットに対応するライン上で割込みを有効にすることが可能である。

【００９７】

本開示は、算出ユニットが、実行が完了したことをスケジューラに通知するために割込みを生成することが可能である実施形態（割込みモード）と、スケジューラが、実行が終わったかどうかを決定するために算出ユニットをポーリングすることが可能である実施形態（ポーリングモード）とを説明する。いくつかの場合には、算出ユニットポーリングは、スケジューラによって実装され得る。他の場合には、算出ユニットは、説明されたように割込みを生成するように構成され得る。割込みモードを使用するとき、算出ユニットをポーリングするためにホストプロセッサによって実行されるスレッドは、コマンド待ち行列が新しいコマンドを記憶するための余地を有するときのみ稼働する純粋なスケジューリ

ングスレッドとして稼働するように構成され得る。
【００９８】
さらに他の場合には、ポーリングと、算出ユニットの生成した割込みとの両方が、組み合わせて使用され得る。たとえば、割込みハンドリングは、複数のカーネルが同時に稼働しているとき、ポーリングに勝る改善された性能を提供し得る。ポーリングモードでは、算出ユニットステータスレジスタは、稼働しているコマンドの各々について読み取られなければならない。割込みモードでは、算出ユニットステータスレジスタは、算出ユニットが、完了した実行を告知するためにスケジューラに割込みをかけない限り、読み取られない。したがって、限られた数の算出ユニットが稼働している場合、ポーリングモードは割込みモードよりも速くなり得る。稼働している算出ユニットの数がしきい値数を超えたとき、実施形態は、割込みモードに遷移し得る。実施形態は、算出ユニットのしきい値数と比較される実行している算出ユニットの数に基づいて、ポーリングモードと割込みモードとの間で遷移し得る。
【００９９】
本明細書で説明される本発明の構成によれば、スケジューラは、コマンドにおいて指定された算出ユニットのいずれか上にワークグループをスケジュールすることが可能である。スケジューラは、できる限り多くのワークグループをアイドル算出ユニット上でスケジュールし、さらなるワークグループをスケジュールする前に算出ユニットが再びアイドルになるのを待ち得る。
【０１００】
特定の実施形態では、コマンド待ち行列がいっぱいである場合、ホストプロセッサは、コマンド待ち行列がさらなるコマンドのための余地を有するという通知を待ち得る。通知は、ホストプロセッサがスケジューラをポーリングすることによって、またはスケジューラ（たとえば、コントローラ）が、コマンド待ち行列内の利用可能な空間を指示するプロセッサへの割込みを生成することによって、実装され得る。
【０１０１】
図６は、ＩＣのための例示的なアーキテクチャ６００を示す。一態様では、アーキテクチャ６００は、プログラマブルＩＣ内に実装され得る。たとえば、アーキテクチャ６００は、ＦＰＧＡを実装するために使用され得る。アーキテクチャ６００はまた、ＩＣのシステムオンチップ（ＳＯＣ）タイプを表し得る。ＳＯＣは、プログラムコードを実行するプロセッサと、１つまたは複数の他の回路とを含むＩＣである。他の回路は、ハードワイヤード回路、プログラマブル回路、および／またはそれらの組合せとして実装され得る。回路は、互いと、および／またはプロセッサと協働して動作し得る。
【０１０２】
図示のように、アーキテクチャ６００は、いくつかの異なるタイプのプログラマブル回路、たとえば、論理、ブロックを含む。たとえば、アーキテクチャ６００は、マルチギガビットトランシーバ（ＭＧＴ：ｍｕｌｔｉ－ｇｉｇａｂｉｔ　ｔｒａｎｓｃｅｉｖｅｒ）６０１、構成可能論理ブロック（ＣＬＢ）６０２、ランダムアクセスメモリブロック（ＢＲＡＭ）６０３、入出力ブロック（ＩＯＢ）６０４、構成およびクロッキング論理（ＣＯＮＦＩＧ／ＣＬＯＣＫＳ）６０５、デジタル信号処理ブロック（ＤＳＰ）６０６、特殊なＩ／Ｏブロック６０７（たとえば、構成ポートおよびクロックポート）、ならびにデジタルクロックマネージャ、アナログデジタル変換器、システム監視論理などの他のプログラマブル論理６０８を含む、多数の異なるプログラマブルタイルを含み得る。
【０１０３】
いくつかのＩＣでは、各プログラマブルタイルは、プログラマブル相互接続要素（ＩＮＴ）６１１を含み、ＩＮＴ６１１は、各隣接するタイル中の対応するＩＮＴ６１１との間の規格化された接続を有する。したがって、ＩＮＴ６１１は、まとめると、示されているＩＣのためのプログラマブル相互接続構造を実装する。各ＩＮＴ６１１は、図６の上部に含まれる例によって示されているように、同じタイル内のプログラマブル論理要素との間の接続をも含む。

【０１０４】
たとえば、ＣＬＢ６０２は、ユーザ論理を実装するようにプログラムされ得る構成可能論理要素（ＣＬＥ）６１２と、単一のＩＮＴ６１１とを含み得る。ＢＲＡＭ６０３は、１つまたは複数のＩＮＴ６１１に加えてＢＲＡＭ論理要素（ＢＲＬ）６１３を含み得る。一般的に、タイル中に含まれるＩＮＴ６１１の数は、タイルの高さに依存する。描かれているように、ＢＲＡＭタイルは、５つのＣＬＢと同じ高さを有するが、他の数（たとえば、４つ）も使用され得る。ＤＳＰタイル６０６は、適切な数のＩＮＴ６１１に加えてＤＳＰ論理要素（ＤＳＰＬ）６１４を含み得る。ＩＯＢ６０４は、たとえば、ＩＮＴ６１１の１つのインスタンスに加えてＩ／Ｏ論理要素（ＩＯＬ）６１５の２つのインスタンスを含み得る。ＩＯＬ６１５に接続された実際のＩ／Ｏパッドは、ＩＯＬ６１５のエリアに制限されないことがある。

【０１０５】
図６に描かれている例では、ダイの中心の近くの、たとえば、領域６０５、６０７、および６０８から形成された、列状エリアが、構成、クロック、および他の制御論理のために使用され得る。この列から延びる水平エリア６０９が、プログラマブルＩＣの幅にわたってクロックおよび構成信号を分散させるために使用され得る。

【０１０６】
図６に示されているアーキテクチャを利用するいくつかのＩＣは、ＩＣの大部分を作り上げる規則的な列状構造を損なう追加の論理ブロックを含む。追加の論理ブロックは、プログラマブルブロックおよび／または専用回路であり得る。たとえば、ＰＲＯＣ６１０として示されているプロセッサブロックが、ＣＬＢおよびＢＲＡＭのいくつかの列にまたがる。

【０１０７】
一態様では、ＰＲＯＣ６１０は、ＩＣのプログラマブル回路を実装するダイの一部として作製される専用回路として、たとえば、ハードワイヤードプロセッサとして実装され得る。ＰＲＯＣ６１０は、個々のプロセッサ、たとえば、プログラムコードを実行することが可能な単一のコアから、１つまたは複数のコア、モジュール、コプロセッサ、インターフェースなどを有するプロセッサシステム全体まで、複雑さに幅がある様々な異なるプロセッサタイプおよび／またはシステムのいずれかを表し得る。

【０１０８】
別の態様では、ＰＲＯＣ６１０は、アーキテクチャ６００から省略され、説明されるプログラマブルブロックの他の種類のうちの１つまたは複数と置き換えられ得る。さらに、そのようなブロックは、ＰＲＯＣ６１０の場合のようにプログラムコードを実行することができるプロセッサを形成するためにプログラマブル回路の様々なブロックが使用され得るという点で、「ソフトプロセッサ」を形成するために利用され得る。

【０１０９】
「プログラマブル回路」という句は、ＩＣ内のプログラマブル回路要素、たとえば、本明細書で説明される様々なプログラマブルまたは構成可能回路ブロックまたはタイル、ならびに、ＩＣにロードされた構成データに従って様々な回路ブロック、タイル、および／または要素を選択的に結合する相互接続回路を指す。たとえば、ＣＬＢ６０２およびＢＲＡＭ６０３など、ＰＲＯＣ６１０の外部にある、図６に示されている回路ブロックは、ＩＣのプログラマブル回路と見なされる。

【０１１０】
概して、プログラマブル回路の機能性は、構成データがＩＣにロードされるまで確立されない。ＦＰＧＡなど、ＩＣのプログラマブル回路をプログラムするために、構成ビットのセットが使用され得る。（１つまたは複数の）構成ビットは、一般的に、「構成ビットストリーム」と呼ばれる。概して、プログラマブル回路は、構成ビットストリームをＩＣに最初にロードしなければ、動作可能でないか、または機能可能でない。構成ビットストリームは、プログラマブル回路内に特定の回路設計を効果的に実装する。回路設計は、たとえば、プログラマブル回路ブロックの機能的態様と、様々なプログラマブル回路ブロッ

クの間の物理的接続性とを指定する。
【０１１１】
「ハードワイヤード」または「ハード化（ｈａｒｄｅｎ）」される、すなわち、プログラマブルでない回路が、ＩＣの一部として製造される。プログラマブル回路とは異なり、ハードワイヤード回路または回路ブロックは、構成ビットストリームのローディングを通してＩＣの製造後に実装されない。ハードワイヤード回路は、概して、たとえば、構成ビットストリームを、ＩＣ、たとえば、ＰＲＯＣ６１０に最初にロードすることなしに機能可能である、専用回路ブロックおよび相互接続を有すると見なされる。
【０１１２】
いくつかの事例では、ハードワイヤード回路は、ＩＣ内の１つまたは複数のメモリ要素に記憶されたレジスタ設定または値に従って設定または選択され得る１つまたは複数の動作モードを有し得る。動作モードは、たとえば、ＩＣへの構成ビットストリームのローディングを通して設定され得る。この能力にもかかわらず、ハードワイヤード回路が、ＩＣの一部として製造されたとき、動作可能であり、特定の機能を有するので、ハードワイヤード回路はプログラマブル回路と見なされない。
【０１１３】
ＳＯＣの場合、構成ビットストリームは、プログラマブル回路内に実装されるべきである回路と、ＰＲＯＣ６１０またはソフトプロセッサによって実行されるべきであるプログラムコードとを指定し得る。いくつかの場合には、アーキテクチャ６００は、適切な構成メモリおよび／またはプロセッサメモリに構成ビットストリームをロードする専用構成プロセッサを含む。専用構成プロセッサは、ユーザ指定のプログラムコードを実行しない。他の場合には、アーキテクチャ６００は、構成ビットストリームを受信し、構成ビットストリームを適切な構成メモリにロードし、および／または実行のためのプログラムコードを抽出するために、ＰＲＯＣ６１０を利用し得る。
【０１１４】
スケジューラ１６０は、図６に関して説明されるようなアーキテクチャを有するＩＣを使用して実装され得る。たとえば、ＢＲＡＭは、コマンド待ち行列を実装するために使用され得る。ＰＲＯＣ６１０は、コントローラ１７５を実装するために使用され得る。ＩＣのプログラマブル回路内のレジスタは、ステータスレジスタ１７０を実装するために使用され得る。プログラマブル回路は、インターフェース１５５と算出ユニット１８０とを実装するために使用され得る。特定の実施形態では、メモリコントローラ１８５は、ハードワイヤードである。他の実施形態では、メモリコントローラ１８５はまた、プログラマブル回路を使用して実装される。
【０１１５】
図６は、プログラマブル回路、たとえば、プログラマブルファブリックを含むＩＣを実装するために使用され得る例示的なアーキテクチャを示すことを意図される。たとえば、１つの列中の論理ブロックの数、列の相対幅、列の数および順序、列中に含まれる論理ブロックのタイプ、論理ブロックの相対サイズ、および図６の上部に含まれる相互接続／論理実装形態は、例示にすぎない。実際のＩＣでは、たとえば、ＣＬＢの２つ以上の隣接する列は、一般的に、ユーザ回路設計の効率的な実装を容易にするために、ＣＬＢが現れるところならどこでも含まれる。しかしながら、隣接するＣＬＢ列の数は、ＩＣの全体的サイズとともに変動し得る。さらに、ＩＣ内のＰＲＯＣ６１０などのブロックのサイズおよび／または配置は、例示のためのものにすぎず、限定として意図されていない。
【０１１６】
説明のために、特定の名称が、本明細書で開示される様々な発明概念の完全な理解を提供するために記載される。しかしながら、本明細書で使用される専門用語は、本発明の構成の特定の態様を説明するためのものにすぎず、限定するものではない。
【０１１７】
本明細書で定義される単数形「ａ」、「ａｎ」および「ｔｈｅ」は、文脈が別段に明確に指示するのでなければ、複数形をも含むものとする。

【０１１８】
本明細書で定義される「少なくとも１つ」、「１つまたは複数」、および「および／または」という用語は、別段に明記されていない限り、運用において連言的と選言的の両方である、オープンエンド表現である。たとえば、「Ａ、Ｂ、およびＣのうちの少なくとも１つ」、「Ａ、Ｂ、またはＣのうちの少なくとも１つ」、「Ａ、Ｂ、およびＣのうちの１つまたは複数」、「Ａ、Ｂ、またはＣのうちの１つまたは複数」、および「Ａ、Ｂ、および／またはＣ」という表現の各々は、Ａのみ、Ｂのみ、Ｃのみ、ＡとＢを一緒に、ＡとＣを一緒に、ＢとＣを一緒に、またはＡとＢとＣを一緒に、を意味する。
【０１１９】
本明細書で定義される「自動的に」という用語は、ユーザ介入なしに、を意味する。本明細書で定義される「ユーザ」という用語は、人間を意味する。
【０１２０】
本明細書で定義される「コンピュータ可読記憶媒体」という用語は、命令実行システム、装置、またはデバイスが使用するための、あるいはそれとともに使用するためのプログラムコードを含んでいるかまたは記憶する記憶媒体を意味する。本明細書で定義される「コンピュータ可読記憶媒体」は、それ自体は、一時的な伝搬信号でない。コンピュータ可読記憶媒体は、限定はしないが、電子記憶デバイス、磁気記憶デバイス、光記憶デバイス、電磁記憶デバイス、半導体記憶デバイス、または上記の任意の好適な組合せであり得る。本明細書で説明される、様々な形態のメモリが、コンピュータ可読記憶媒体の例である。コンピュータ可読記憶媒体のより具体的な例の非網羅的なリストは、ポータブルコンピュータディスケット、ハードディスク、ＲＡＭ、読取り専用メモリ（ＲＯＭ）、消去可能プログラマブル読取り専用メモリ（ＥＰＲＯＭまたはフラッシュメモリ）、電子的消去可能プログラマブル読取り専用メモリ（ＥＥＰＲＯＭ）、スタティックランダムアクセスメモリ（ＳＲＡＭ）、ポータブルコンパクトディスク読取り専用メモリ（ＣＤ－ＲＯＭ）、デジタル多用途ディスク（ＤＶＤ）、メモリスティック、フロッピーディスクなどを含み得る。
【０１２１】
本明細書で定義される「する場合（ｉｆ）」という用語は、文脈に応じて、「するとき（ｗｈｅｎ）」または「すると（ｕｐｏｎ）」または「に応答して（ｉｎ　ｒｅｓｐｏｎｓｅ　ｔｏ）」または「に反応して（ｒｅｓｐｏｎｓｉｖｅ　ｔｏ）」を意味する。したがって、「それが決定された場合」または「［述べられた条件またはイベント］が検出された場合」という句は、文脈に応じて、「決定すると」または「決定したことに応答して」あるいは「［述べられた条件またはイベント］を検出すると」または「［述べられた条件またはイベント］を検出したことに応答して」または「［述べられた条件またはイベント］を検出したことに反応して」を意味すると解釈され得る。
【０１２２】
本明細書で定義される「に反応して」という用語および上記で説明されたような同様の言い回し、たとえば、「する場合」、「するとき」、または「すると」は、アクションまたはイベントに容易に応答または反応することを意味する。応答または反応は、自動的に実施される。したがって、第２のアクションが第１のアクション「に反応して」実施される場合、第１のアクションの発生と第２のアクションの発生との間に因果関係がある。「に反応して」という用語は、因果関係を指示する。
【０１２３】
本明細書で定義される「一実施形態（ｏｎｅ　ｅｍｂｏｄｉｍｅｎｔ）」、「一実施形態（ａｎ　ｅｍｂｏｄｉｍｅｎｔ）」、「１つまたは複数の実施形態」、「特定の実施形態」という用語、または同様の言い回しは、実施形態に関して説明される特定の特徴、構造、または特性が、本開示内で説明される少なくとも１つの実施形態に含まれることを意味する。したがって、本開示全体にわたる、「一実施形態では（ｉｎ　ｏｎｅ　ｅｍｂｏｄｉｍｅｎｔ）」、「一実施形態では（ｉｎ　ａｎ　ｅｍｂｏｄｉｍｅｎｔ）」、「１つまたは複数の実施形態では」、「特定の実施形態では」という句、および同様の言い回しの出

現は、必ずしもそうとは限らないが、すべて、同じ実施形態を指し得る。「実施形態」および「構成」という用語は、本開示内では互換的に使用される。

【０１２４】

本明細書で定義される「プロセッサ」という用語は、少なくとも１つのハードウェア回路を意味する。ハードウェア回路は、プログラムコード中に含まれている命令を行うように構成され得る。ハードウェア回路は集積回路であり得る。プロセッサの例は、限定はしないが、中央処理ユニット（ＣＰＵ）、アレイプロセッサ、ベクトルプロセッサ、デジタル信号プロセッサ（ＤＳＰ）、ＦＰＧＡ、プログラマブル論理アレイ（ＰＬＡ）、ＡＳＩＣ、プログラマブル論理回路、およびコントローラを含む。

【０１２５】

本明細書で定義される「出力」という用語は、物理メモリ要素、たとえば、デバイスに記憶すること、ディスプレイまたは他の周辺出力デバイスに書き込むこと、別のシステムに送ることまたは送信すること、エクスポートすることなどを意味する。

【０１２６】

本明細書で定義される「リアルタイム」という用語は、ユーザまたはシステムが、特定のプロセスまたは決定が行われるのに十分に即時であると感じる、あるいは、プロセッサが、何らかの外部プロセスについていくことを可能にする、処理応答性のレベルを意味する。

【０１２７】

本明細書で定義される「実質的に」という用語は、具陳された特性、パラメータ、または値が正確に達成される必要がないこと、ただし、たとえば、当業者に知られている許容差、測定誤差、測定精度限界、および他のファクタを含む、偏差または変動が、特性が提供することを意図された効果を妨げない量で生じ得ることを意味する。

【０１２８】

第１の、第２のなどの用語は、様々な要素を説明するために本明細書で使用され得る。これらの用語は、別段に述べられていない限り、または文脈が別段に明確に指示しない限り、ある要素を別の要素と区別するために使用されるにすぎないので、これらの要素はこれらの用語によって限定されるべきでない。

【０１２９】

コンピュータプログラム製品は、プロセッサに本明細書で説明される本発明の構成の態様を行わせるためのコンピュータ可読プログラム命令をその上に有する（１つまたは複数の）コンピュータ可読記憶媒体を含み得る。本開示内では、「プログラムコード」という用語は、「コンピュータ可読プログラム命令」という用語と互換的に使用される。本明細書で説明されるコンピュータ可読プログラム命令は、コンピュータ可読記憶媒体からそれぞれのコンピューティング／処理デバイスに、あるいはネットワーク、たとえば、インターネット、ＬＡＮ、ＷＡＮおよび／またはワイヤレスネットワークを介して外部コンピュータまたは外部記憶デバイスにダウンロードされ得る。ネットワークは、銅伝送ケーブル、光伝送ファイバー、ワイヤレス送信、ルータ、ファイアウォール、スイッチ、ゲートウェイコンピュータ、および／またはエッジサーバを含むエッジデバイスを含み得る。各コンピューティング／処理デバイス中のネットワークアダプタカードまたはネットワークインターフェースは、ネットワークからコンピュータ可読プログラム命令を受信し、そのコンピュータ可読プログラム命令を、それぞれのコンピューティング／処理デバイス内のコンピュータ可読記憶媒体に記憶するためにフォワーディングする。

【０１３０】

本明細書で説明される本発明の構成のための動作を行うためのコンピュータ可読プログラム命令は、アセンブラ命令、命令セットアーキテクチャ（ＩＳＡ）命令、機械命令、機械依存命令、マイクロコード、ファームウェア命令、あるいは、オブジェクト指向プログラミング言語および／または手続き型プログラミング言語を含む１つまたは複数のプログラミング言語の任意の組合せで書き込まれたソースコードまたはオブジェクトコードのいずれかであり得る。コンピュータ可読プログラム命令は、状態設定データを含み得る。コ

ンピュータ可読プログラム命令は、完全にユーザのコンピュータ上で、部分的にユーザのコンピュータ上で、スタンドアロンソフトウェアパッケージとして、部分的にユーザのコンピュータ上でおよび部分的にリモートコンピュータ上で、あるいは完全にリモートコンピュータまたはサーバ上で実行し得る。後者のシナリオでは、リモートコンピュータは、ＬＡＮまたはＷＡＮを含む任意のタイプのネットワークを通してユーザのコンピュータに接続され得るか、あるいは接続は、（たとえば、インターネットサービスプロバイダを使用してインターネットを通して）外部コンピュータに対して行われ得る。いくつかの場合には、たとえば、プログラマブル論理回路、ＦＰＧＡ、またはＰＬＡを含む電子回路が、本明細書で説明される本発明の構成の態様を実施するために、電子回路を個人化するためにコンピュータ可読プログラム命令の状態情報を利用することによって、コンピュータ可読プログラム命令を実行し得る。

【０１３１】

本発明の構成のいくつかの態様が、方法、装置（システム）、およびコンピュータプログラム製品のフローチャート例示図および／またはブロック図を参照しながら本明細書で説明された。フローチャート例示図および／またはブロック図の各ブロック、ならびにフローチャートの例示図および／またはブロック図中のブロックの組合せが、コンピュータ可読プログラム命令、たとえば、プログラムコードによって実装され得ることを理解されよう。

【０１３２】

これらのコンピュータ可読プログラム命令は、汎用コンピュータ、専用コンピュータ、または機械を作り出すための他のプログラマブルデータ処理装置のプロセッサに提供され得、その結果、コンピュータまたは他のプログラマブルデータ処理装置のプロセッサを介して実行する命令は、フローチャートおよび／またはブロック図の１つまたは複数のブロックにおいて指定された機能／行為を実装するための手段を作成する。これらのコンピュータ可読プログラム命令はまた、コンピュータ、プログラマブルデータ処理装置、および／または他のデバイスに特定の様式で機能するように指示することができるコンピュータ可読記憶媒体に記憶され得、その結果、命令が記憶されたコンピュータ可読記憶媒体は、フローチャートおよび／またはブロック図の１つまたは複数のブロックにおいて指定された動作の態様を実装する命令を含む製造品を備える。

【０１３３】

コンピュータ可読プログラム命令はまた、コンピュータ実装プロセスを作り出すために、一連の動作をコンピュータ、他のプログラマブルデータ処理装置または他のデバイス上で実施させるように、コンピュータ、他のプログラマブル装置、または他のデバイスにロードされ得、その結果、コンピュータ、他のプログラマブル装置、または他のデバイス上で実行する命令は、フローチャートおよび／またはブロック図の１つまたは複数のブロックにおいて指定された機能／行為を実装する。

【０１３４】

図中のフローチャートおよびブロック図は、本発明の構成の様々な態様によるシステム、方法、およびコンピュータプログラム製品の可能な実装形態のアーキテクチャ、機能性、および動作を示す。この点について、フローチャートまたはブロック図中の各ブロックは、指定された動作を実装するための１つまたは複数の実行可能な命令を備える、命令のモジュール、セグメント、または部分を表し得る。

【０１３５】

いくつかの代替実装形態では、ブロック中で言及される動作は、図中で言及される順序から外れて行われ得る。たとえば、関与する機能性に応じて、連続して示されている２つのブロックが、実質的に同時に実行され得るか、またはブロックが、時々、逆の順序で実行され得る。他の例では、ブロックは、概して小さい数字から順に実施され得、さらに他の例では、１つまたは複数のブロックは、変動順で実施され得、結果は、記憶され、後続の、または直後にこない他のブロックにおいて利用される。また、ブロック図および／またはフローチャート例示図の各ブロック、ならびにブロック図および／またはフローチャ

ート例示図中のブロックの組合せが、指定された機能または行為を実施するかあるいは専用ハードウェアとコンピュータ命令との組合せを行う専用ハードウェアベースシステムによって実装され得ることに留意されたい。

【０１３６】

以下の特許請求の範囲において見られ得るすべての手段またはステップおよび機能要素の対応する構造、材料、行為、および等価物は、特に主張されるように、他の請求される要素と組み合わせて機能を実施するための任意の構造、材料、または行為を含むことを意図される。

【０１３７】

１つまたは複数の実施形態は、ハードウェアアクセラレーションのためのスケジューラを含む集積回路（ＩＣ）を対象とする。スケジューラは、複数のスロットを有し、ＩＣの算出ユニットによる実行のためにホストプロセッサからオフロードされたコマンドを記憶するように構成されたコマンド待ち行列を含み得る。スケジューラは、コマンド待ち行列のスロットに対応するビットロケーションを有するステータスレジスタを含み得る。スケジューラは、コマンド待ち行列とステータスレジスタとに結合されたコントローラをも含み得る。コントローラは、コマンド待ち行列のスロットに記憶されたコマンドを実行し、コマンド待ち行列からのどのコマンドを実行し終えたかを指示するためにステータスレジスタのビットロケーションを更新するように、ＩＣの算出ユニットをスケジュールするように構成され得る。

【０１３８】

一態様では、ステータスレジスタは、読み取られたことに応答して、そこに記憶されたコンテンツを消去するように構成され得る。

【０１３９】

別の態様では、スケジューラは、通信リンクを介してホストプロセッサと通信し、コマンド待ち行列の利用可能なスロット内にコマンドを記憶するように構成されたインターフェースを含み得る。

【０１４０】

別の態様では、コマンド待ち行列に記憶されたコマンドは、それぞれのコマンドを実行するために算出ユニットによって使用される引数を含み得る。

【０１４１】

別の態様では、各コマンドは、算出ユニットのうちのどれがコマンドを実行することを可能にされるかを指定し得る。

【０１４２】

別の態様では、コントローラは、コマンドを、算出ユニットに、算出ユニットのうちのどれがアイドルであり、各それぞれのコマンドを実行することを可能にされるかに基づいて割り当てるように構成され得る。

【０１４３】

別の態様では、コントローラは、プログラムコードを実行するように構成されたプロセッサであり得る。プロセッサは、ＩＣのプログラマブル回路を使用して実装されるソフトプロセッサであり得る。

【０１４４】

別の態様では、コントローラは、選択されたコマンドがその上で稼働することが可能である選択された算出ユニットがアイドルであると決定し、実行のために、選択されたコマンドの引数を選択された算出ユニットに転送し、選択された算出ユニットを開始するように構成され得る。

【０１４５】

別の態様では、コントローラは、選択されたコマンドを実行し終えたと決定したことに応答して、選択されたコマンドを含むスロットに対応する、ステータスレジスタ中のビットロケーションに書き込み、スロットがフリーであることを指示するように構成され得る。

【０１４６】

１つまたは複数の実施形態は、ＩＣを使用してハードウェアアクセラレーションのためにコマンドをスケジュールする方法を対象とする。本方法は、ＩＣ内のコマンド待ち行列のスロット内に、ホストプロセッサから受信されたコマンドを記憶することであって、コマンドが、ＩＣの算出ユニットによる実行のためにホストプロセッサからオフロードされる、コマンドを記憶することを含み得る。本方法は、コントローラを使用して、コマンド待ち行列のスロットに記憶されたコマンドを実行するように算出ユニットをスケジュールすることを含み得る。本方法は、コマンドを実行し終えたと決定したことに応答して、ＩＣ内のステータスレジスタ中のビットロケーションを書き込むことであって、ビットロケーションが、コマンドを記憶するコマンド待ち行列のスロットに対応する、ビットロケーションを書き込むことを含み得る。

【０１４７】

一態様では、本方法は、コントローラ内に、コマンド待ち行列のスロットに記憶されたコマンドのヘッダのローカルにキャッシュされたコピーを維持することと、ヘッダのローカルにキャッシュされたコピー中の値を更新することによってスロットのステータスを更新することとを含み得る。

【０１４８】

別の態様では、本方法は、ステータスレジスタが読み取られたことに応答して、ステータスレジスタに記憶されたコンテンツを消去することを含み得る。

【０１４９】

別の態様では、コマンド待ち行列に記憶されたコマンドは、コマンドを実行するために算出ユニットによって使用される引数を含み得る。

【０１５０】

別の態様では、本方法は、各コマンドから、算出ユニットのうちのどれがコマンドを実行することを可能にされるかを決定することを含み得る。

【０１５１】

別の態様では、本方法は、コマンドを、算出ユニットに、算出ユニットのうちのどれがアイドルであり、各それぞれのコマンドを実行することを可能にされるかに基づいて割り当てることを含み得る。

【０１５２】

別の態様では、本方法は、選択されたコマンドがその上で稼働することが可能である選択された算出ユニットがアイドルであると決定することと、実行のために、選択されたコマンドの引数を選択された算出ユニットに転送することと、選択された算出ユニットを開始することとを含み得る。

【０１５３】

別の態様では、ビットロケーションの書込みは、スロットがフリーであることを指示し得る。

【０１５４】

別の態様では、本方法は、選択された算出ユニットから割込みを受信することによって、選択された算出ユニットが選択されたコマンドを実行し終えたと決定することを含み得る。

【０１５５】

別の態様では、本方法は、選択された算出ユニットをポーリングすることによって、選択された算出ユニットが選択されたコマンドを実行し終えたと決定することを含み得る。

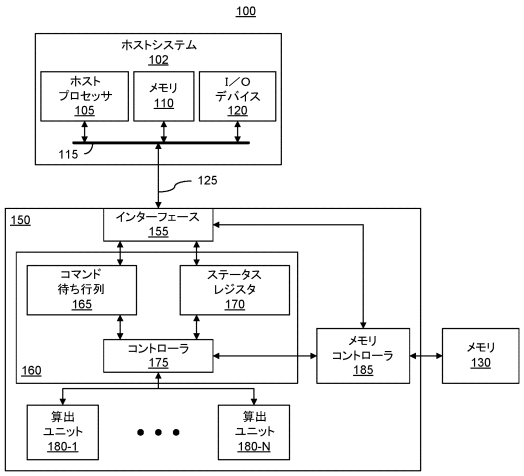
【０１５６】

本明細書で提供される本発明の構成の説明は、例示のためであり、網羅的なものでも、開示される形式および例に限定されるものでもない。本明細書で使用される専門用語は、本発明の構成の原理、実際的適用例、または市場で見られる技術に対する技術的改善を説明するために、および／あるいは、他の当業者が本明細書で開示される本発明の構成を理解することを可能にするために選定された。説明される本発明の構成の範囲および趣旨から逸脱することなく、修正および変形が当業者に明らかになり得る。したがって、そのよ

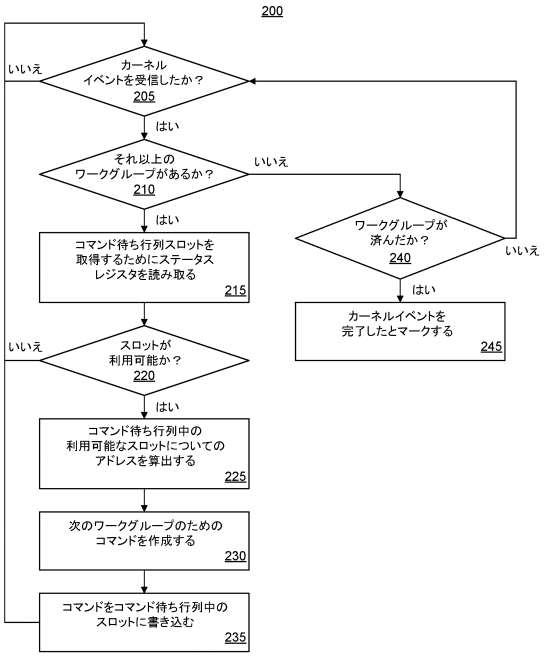
うな特徴および実装形態の範囲を指示するものとして、上記の開示に対してではなく、以下の特許請求の範囲に対して参照が行われるべきである。

【図面】

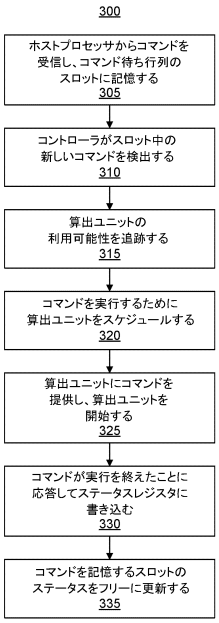
【図１】



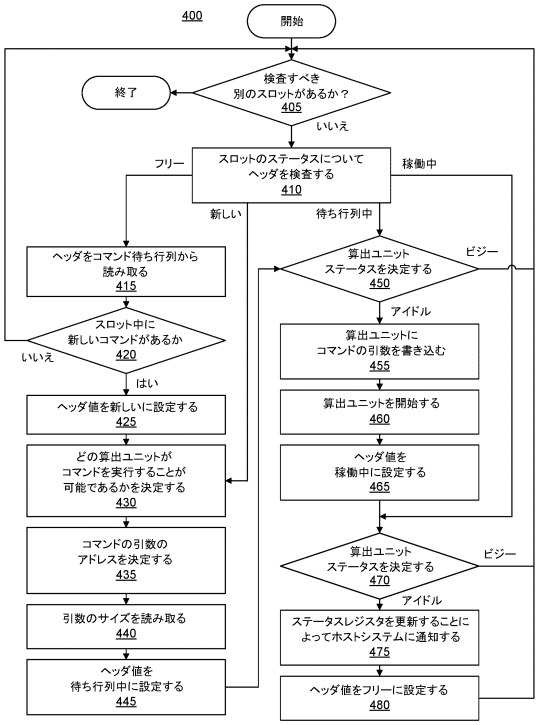
【図２】



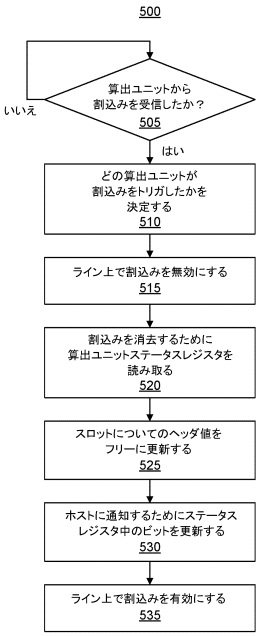
【図３】



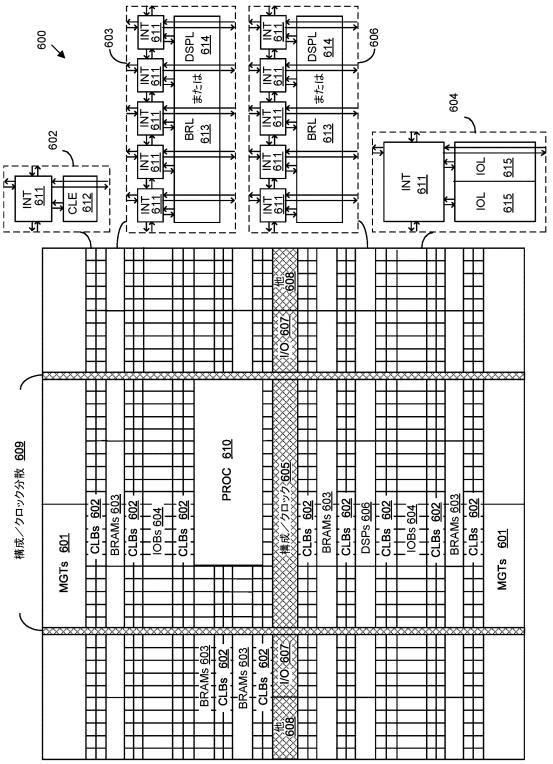
【図４】



【図５】


500
算出ユニットから
割込みを受信したか？
505
いいえ
はい
どの算出ユニットが
割込みをトリガしたかを
決定する
510
ライン上で割込みを無効にする
515
割込みを消去するために
算出ユニットステータスレジスタを
読み取る
520
スロットについてのヘッダ値を
フリーに更新する
525
ホストに通知するためにステータス
レジスタ中のビットを更新する
530
ライン上で割込みを有効にする
535

【図６】


600
603
606
602
604
INT 611
CLE 612
DSPL 614
BRL 613
または
IOL 615
PROC 610
MGTs 601
CLBs 602
BRAMs 603
IOBs 604
DSPs 606
I/O 607
他 608
構成／クロック分散 609

フロントページの続き

２１００
(72)発明者　パレク， ウマン
アメリカ合衆国 カリフォルニア ９５１２４， サン ノゼ， ロジック ドライブ ２１００
(72)発明者　サンタン， ソナル
アメリカ合衆国 カリフォルニア ９５１２４， サン ノゼ， ロジック ドライブ ２１００
(72)発明者　ニーマ， ヘム シー．
アメリカ合衆国 カリフォルニア ９５１２４， サン ノゼ， ロジック ドライブ ２１００

審査官　坂東 博司
(56)参考文献　米国特許出願公開第２０１５／２３４６７９（ＵＳ，Ａ１）
米国特許出願公開第２０１３／０１１７５３３（ＵＳ，Ａ１）
米国特許出願公開第２０１６／００９８３６５（ＵＳ，Ａ１）
米国特許出願公開第２０１６／０３３５２１５（ＵＳ，Ａ１）
米国特許出願公開第２０１３／００３１５５３（ＵＳ，Ａ１）
米国特許出願公開第２０１４／０３４４８１５（ＵＳ，Ａ１）
(58)調査した分野　(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　９／３８