(19)日本国特許庁(JP)

(12)特許公報(B2)

(11)特許番号
特許第7245833号
(P7245833)

(45)発行日　令和5年3月24日(2023.3.24)

(24)登録日　令和5年3月15日(2023.3.15)

(51)国際特許分類

		FI		
G06F	*9/38 (2018.01)*	G06F	9/38	370C
G06F	*15/80 (2006.01)*	G06F	15/80	

請求項の数　17　（全16頁）

(21)出願番号　特願2020-529105(P2020-529105)
(86)(22)出願日　平成30年8月2日(2018.8.2)
(65)公表番号　特表2020-530175(P2020-530175A)
(43)公表日　令和2年10月15日(2020.10.15)
(86)国際出願番号　PCT/US2018/045008
(87)国際公開番号　WO2019/028253
(87)国際公開日　平成31年2月7日(2019.2.7)
審査請求日　令和3年7月29日(2021.7.29)
(31)優先権主張番号　62/540,849
(32)優先日　平成29年8月3日(2017.8.3)
(33)優先権主張国・地域又は機関
米国(US)
(31)優先権主張番号　62/558,090
(32)優先日　平成29年9月13日(2017.9.13)
最終頁に続く

(73)特許権者　520039124
ネクスト シリコン リミテッド
Ｎｅｘｔ Ｓｉｌｉｃｏｎ Ｌｔｄ
イスラエル国 ギヴァタイム デレク イツハク ラビン ストリート ３３
３３ Ｄｅｒｅｃｈ Ｙｉｔｓｈａｋ Ｒａｂｉｎ Ｓｔｒｅｅｔ, Ｇｉｖａｔａｙｉｍ, Ｉｓｒａｅｌ
(74)代理人　110002952
弁理士法人鷲田国際特許事務所
(72)発明者　ラズ エラド
イスラエル国 ラマト ガン ビンヤミン ストリート １１
審査官　三坂 敏夫
最終頁に続く

(54)【発明の名称】　構成可能なハードウェアの実行時の最適化

(57)【特許請求の範囲】

【請求項１】

少なくとも１つの計算装置によって、プログラムコードの複数の部分を含むソフトウェアプログラムを実行するためのシステムであって、

少なくとも１つの計算装置に接続された計算グリッドであって、データ入力メッシュネットワーク、複数の論理要素、および複数のマルチプレクサ、を有し、前記複数の論理要素および前記複数のマルチプレクサは、複数の計算グループを形成し、前記論理要素の少なくとも１つおよび複数のマルチプレクサの少なくとも１つは、複数の計算グループのそれぞれを形成する、前記計算グリッドと、

プログラムコードの前記複数の部分のうち、プログラムコードの２つ以上の部分を含むプログラムコードパターンであって、各パターンは、前記ソフトウェアプログラムを実行するときにプログラムコードの前記２つ以上の部分のそれぞれの出現における関連性を示している少なくとも１つの前記プログラムコードパターンにアクセスし、

前記少なくとも１つのプログラムコードパターンのそれぞれに対し、

プログラムコードの前記２つ以上の部分の少なくとも一部を操作し、

プログラムコードの前記２つ以上の部分のうち、操作された少なくとも一部を実行するように、前記複数の計算グループの少なくともいくつかを操作し、

少なくとも１つの計算装置の代わりに、前記計算グリッドによって少なくとも１つのプログラムコードパターンを計算する、

ことを複数回実行することにより、前記プログラムコードの複数の部分の少なくともいく

つかを実行するように前記計算グリッドを操作するために適合された少なくとも１つの処理回路と、
を備える、システム。
【請求項２】
前記少なくとも１つの計算装置によって前記ソフトウェアプログラムを実行することは、前記計算グリッドがプログラムコードの前記複数の部分から選択されたプログラムコードの部分の第２のセットを実行する間に前記プログラムコードの複数の部分から選択されたプログラムコードの部分の第１のセットをそれによって実行することを含む、
請求項１に記載のシステム。
【請求項３】
前記少なくとも１つの処理回路は、
少なくとも１つの他の処理回路から前記少なくとも１つのプログラムコードパターンを受信すること、および、
プログラムコードの前記複数の部分のうちの１つをそれぞれが実行するための複数の呼出しを受信し、
前記複数の呼び出しにおいて、前記少なくとも１つのプログラムコードパターンを識別する、
ことにより、前記少なくとも１つのプログラムコードパターンを計算すること、
の少なくとも１つのためにさらに適合されている、
請求項１に記載のシステム。
【請求項４】
前記少なくとも１つのプログラムコードパターンは、
プログラムコードの前記２つ以上の部分のそれぞれのうちのコードの第１の部分およびコードの２つ以上の部分のそれぞれのうちのコードの第２の部分であって、前記複数の呼出しにおいて、プログラムコードの前記第１の部分およびプログラムコードの前記第２の部分が次々に出現する第１の頻度が所定のしきい値頻度を越えているような、前記第１の部分および前記第２の部分を有している、第１のプログラムコードパターンと、
プログラムコードの前記２つ以上の部分のそれぞれのうちのコードの第３の部分およびコードの前記２つ以上の部分のそれぞれのうちのコードの第４の部分であって、前記複数の呼出しにおいて、コードの前記第３の部分がコードの前記第４の部分を呼び出す第２の頻度が所定のしきい値頻度を越えているような、前記第３の部分および前記第４の部分を有している、第２のプログラムコードパターンと、
プログラムコードの前記２つ以上の部分のそれぞれのうちのコードの第５の部分をそれぞれ有している、第３のコードパターンおよび第４のコードパターンと、
の少なくとも１つを備える、
請求項３に記載のシステム。
【請求項５】
前記複数の計算グループの前記少なくともいくつかを操作することは、前記第１のプログラムコードパターンのために、プログラムコードの前記第１の部分を実行するように、前記複数の計算グループのうちの少なくとも１つの第１の計算グループを操作することを含んでおり、
前記少なくとも１つの第１の計算グループは、プログラムコードの前記第２の部分を実行するように構成された前記少なくとも１つの第２の計算グループにトポロジー的に近接している前記少なくとも１つの計算グリッド内に割り当てられている、
請求項４に記載のシステム。
【請求項６】
前記複数の計算グループの前記少なくともいくつかを操作することは、前記第２のプログラムコードパターンのために、プログラムコードの前記第３の部分を実行するように、前記複数の計算グループのうちの少なくとも１つの第３の計算グループを操作することを含んでおり、

前記少なくとも１つの第３の計算グループは、プログラムコードの前記第４の部分を実行するように構成された前記少なくとも１つの第４の計算グループにトポロジー的に近接している前記少なくとも１つの計算グリッド内に割り当てられている、
請求項４に記載のシステム。
【請求項７】
前記複数の計算グループの前記少なくともいくつかを操作することは、プログラムコードの前記第５の部分を実行するように、前記複数の計算グループのうちの少なくとも２つの計算グループを操作することを含む、
請求項４のシステム。
【請求項８】
前記複数の計算グループの前記少なくともいくつかを操作することは、前記少なくとも１つのプログラムコードパターンの少なくとも１つに対して、
プログラムコードの前記２つ以上の部分のそれぞれのうちの１つの少なくとも一部を実行するようにそれぞれ構成された前記複数の計算グループの少なくとも２つを識別すること、
少なくとも２つの他の計算グループが互いにトポロジー的に近接した状態で前記計算グリッド内に割り当てられるように、前記複数の計算グループの少なくとも２つの他の計算グループを選択すること、および、
前記少なくとも２つの計算グループが実行されたプログラムコードの前記２つ以上の部分のそれぞれのうちの１つの少なくとも一部を実行するように、前記少なくとも２つの他の計算グループのそれぞれを操作すること、
を含む、請求項１に記載のシステム。
【請求項９】
前記複数の計算グループの前記少なくともいくつかを操作することは、前記少なくとも１つのプログラムコードパターンの少なくとも１つのために、
プログラムコードの前記２つ以上の部分のそれぞれのプログラムコードの第１の部分を実行するように操作された前記複数の計算グループのうちの少なくとも１つの第１の計算グループを識別すること、および、
プログラムコードの前記２つ以上の部分のそれぞれのプログラムコードの第２の部分を実行するように、前記複数の計算グループのうちの少なくとも１つの第２の計算グループであって、前記少なくとも１つの計算グループにトポロジー的に近接している前記計算グループ内に割り当てられている前記第２の計算グループを操作すること、
を含む、請求項１に記載のシステム。
【請求項１０】
前記複数の計算グループの前記少なくともいくつかを操作することは、
前記複数の計算グループのいずれにもアクセスされておらず、かつ前記計算装置にもアクセスされていない前記複数の計算グループのうちの少なくとも１つの計算グループを解放すること、
をさらに含む、請求項１に記載のシステム。
【請求項１１】
前記少なくとも１つの処理回路は、前記ソフトウェアプログラムが前記少なくとも１つの計算装置および前記計算グリッドによって実行されている間、前記計算グリッドを操作するために適合される、
請求項１に記載のシステム。
【請求項１２】
前記少なくとも１つの計算装置は、
マルチコア中央処理装置（ＣＰＵ）、フィールドプログラマブルゲートアレイ（ＦＰＧＡ）、グラフィック処理装置（ＧＰＵ）、粗粒度再構成可能アーキテクチャ（ＣＧＲＡ）、ニューラルネットワークアクセラレータ、インテリジェンス処理ユニット（ＩＰＵ）、特定用途向け集積回路（ＡＳＩＣ）、量子コンピュータ、

のいずれか１つである、
請求項１に記載のシステム。
【請求項１３】
少なくとも１つの計算装置によって、プログラムコードの複数の部分を含むソフトウェアプログラムを実行するための方法であって、
プログラムコードの前記複数の部分のうちの少なくともいくつかを実行するように、データ入力メッシュネットワーク、複数の論理要素、および複数のマルチプレクサを有する計算グリッドの少なくとも一部分を操作するステップを備え、
前記複数の論理要素および前記複数のマルチプレクサは、複数の計算グループを形成しており、
前記論理要素の少なくとも１つおよび複数のマルチプレクサの少なくとも１つは、複数の計算グループのそれぞれを形成しており、
前記計算グリッドの前記少なくとも一部分を操作するステップは、
プログラムコードの前記複数の部分のうち、２つ以上を有するプログラムコードパターンであって、各パターンは、前記ソフトウェアプログラムを実行するときにプログラムコードの前記２つ以上の部分のそれぞれの出現における関連性を示している少なくとも１つの前記プログラムコードパターンにアクセスするステップと、
前記少なくとも１つのプログラムコードパターンのそれぞれに対し、
プログラムコードの前記２つ以上の部分の少なくとも一部を操作し、
プログラムコードの前記２つ以上の部分のうち、操作された少なくとも一部を実行するように、複数の計算グループの少なくともいくつかを操作するステップと、
前記少なくとも１つの計算装置の代わりに、前記計算グリッドによって少なくとも１つのプログラムコードパターンを計算するステップと、
を複数回実行することを含む、
方法。
【請求項１４】
プログラムコードの複数の部分を有するソフトウェアプログラムを少なくとも１つの計算装置によって実行するプロセスを処理ユニットに実行させる命令を格納している非一過性のコンピュータ可読媒体であって、前記プロセスは、
プログラムコードの前記複数の部分のうちの少なくともいくつかを実行するように、データ入力メッシュネットワーク、複数の論理要素、および複数のマルチプレクサを有する計算グリッドの少なくとも一部分を操作するステップを備え、
前記複数の論理要素および前記複数のマルチプレクサは、複数の計算グループを形成しており、
前記論理要素の少なくとも１つおよび複数のマルチプレクサの少なくとも１つは、複数の計算グループのそれぞれを形成しており、
前記計算グリッドの前記少なくとも一部分を操作するステップは、
プログラムコードの前記複数の部分のうち、２つ以上を有するプログラムコードパターンであって、各パターンは、前記ソフトウェアプログラムを実行するときにプログラムコードの前記２つ以上の部分のそれぞれの出現における関連性を示している少なくとも１つの前記プログラムコードパターンにアクセスするステップと、
前記少なくとも１つのプログラムコードパターンのそれぞれに対し、
プログラムコードの前記２つ以上の部分の少なくとも一部を操作し、
プログラムコードの前記２つ以上の部分のうち、操作された少なくとも一部を実行するように、複数の計算グループの少なくともいくつかを操作するステップと、
前記少なくとも１つの計算装置の代わりに、前記計算グリッドによって少なくとも１つのプログラムコードパターンを計算するステップと、
を複数回実行することを含む、
非一過性のコンピュータ可読媒体。
【請求項１５】

計算装置の動作を加速させる装置であって、
前記装置は、
少なくとも１つの関数を実行するための複数の呼出しを含むソフトウェアプログラムの少なくとも一部と、前記計算装置により実行される前記ソフトウェアプログラムの少なくとも別の一部を実行する計算グリッドを備え、
前記計算グリッドは、
データ入力メッシュネットワークと、
複数の論理素子と、
複数のマルチプレクサであって、前記複数の論理素子と前記複数のマルチプレクサが複数の計算グループを形成し、前記論理素子の少なくとも１つと前記複数のマルチプレクサの少なくとも１つが、前記計算グループのそれぞれを形成する、前記複数のマルチプレクサと、
を備え、
前記計算装置によって実行される少なくとも１つの関数を実行するための複数の呼出しの間で少なくとも１つのパターンが識別された場合、前記計算装置は、
前記少なくとも１つの関数を操作し、
前記少なくとも１つのパターンに基づいて、前記複数の計算グループの少なくとも１つを操作して、操作された前記少なくとも１つの関数を実行する、
装置。
【請求項１６】
前記少なくとも１つのパターンが少なくとも１つの関数を実行するための前記複数の呼出しにおける少なくとも１つの繰り返し関数の再出現を示している場合、前記計算装置は、前記複数の計算グループの少なくとも２つが前記少なくとも１つの繰り返し関数を実行するように前記複数の計算グループの少なくとも２つのそれぞれを操作する、
請求項１５に記載の装置。
【請求項１７】
前記少なくとも１つのパターンが、第１の関数および第２の関数を用いた演算の実行を示している場合、前記計算装置は、前記第１の関数を実行する前記複数の計算グループの少なくとも１つの第１計算グループを操作し、
前記少なくとも１つの第１計算グループは、前記計算グリッドにおいて、前記第２の関数を実行する前記複数の計算グループの少なくとも１つの第２計算グループにトポロジー的に近接するように割り当てられる、
請求項１５または１６に記載の装置。
【発明の詳細な説明】
【技術分野】
【０００１】
本開示は、全般的には、ハードウェアの実行時の最適化に関し、より詳細には、構成可能なハードウェアを実行時に最適化する手法に関する。
【背景技術】
【０００２】
技術が進歩するにつれて、より高い処理能力を有する、より強力な処理システムのニーズが急速に高まる。最近では、プロセッサは、高い計算スループットを提供し、かつ高い電力効率であることが期待されている。しかしながら既存の処理システムは、メモリへの明示的な格納を通じて伝えられる命令の逐次的なストリームを実行し、したがってモデルの非省電力性という課題がある。
【０００３】
現代の処理アーキテクチャにおいては、たとえプログラムの大部分がコードの小さな静的部分を繰り返す場合でも、各動的命令をフェッチしてデコードしなければならない。さらには、状態を明示的に記憶することが、命令間でデータを伝えるための唯一の経路であるため、機能ユニットとレジスタファイルの間で中間結果が繰り返し伝送される。現代の

計算アーキテクチャの制限としては、高い電力消費量、熱放散、ネットワークおよびＩ／Ｏのボトルネック、およびメモリパーティションが挙げられる。

【０００４】

例えば、フィールドプログラマブルゲートアレイ（ＦＰＧＡ）は、ソフトウェアによって構成されるハードウェア回路上で動作する。ＦＰＧＡでは、低レイテンシでの極めて高いスループット率が可能である。ＦＰＧＡの構成可能性は、マルチコアアーキテクチャにおけるコプロセッサとして使用することができ、またはシステムのＣＰＵの処理負荷をオフロードするためにクリティカルデータパス上に配置することができる。ＦＰＧＡの主たる欠点の１つは、柔軟なプログラミング性がないことである。さらに、ＦＰＧＡの計算能力は比較的低い。

【０００５】

プログラム可能な処理アーキテクチャの例は、マルチコアプロセッサである。マルチコアプロセッサのアーキテクチャは、２つ以上の独立した実際の処理ユニット（「コア」）を有する１つの計算要素を含み、処理ユニットはプログラム命令を読み取って実行するユニットである。これらの命令は、普通のＣＰＵ命令（例：加算、データの移動、分岐）である。このアーキテクチャでは、１つのプロセッサが、複数の命令を個別のコア上で並列に実行することができる。マルチコアプロセッサの主たる欠点は、高い電力消費量および低いスループットである。

【０００６】

処理アーキテクチャの別の例は、グラフィック処理装置（ＧＰＵ）である。ＧＰＵは、複数のタスクを同時に扱うように設計されている、より小さく、より効率的な何千ものコアからなる並列アーキテクチャに基づいている。ＧＰＵは、ディープラーニング、グラフィックスのレンダリング、機械学習のアプリケーションの計算タスクをアクセラレートさせるために利用することができる。ＧＰＵの主たる欠点は、高い電力消費量および高レイテンシである。さらにＧＰＵは、メモリ一貫性（memory coherency）がなく、したがって共有メモリには課題がある。

【０００７】

したがって、上述した欠陥を克服する処理アーキテクチャを提供することは有利であろう。

【発明の概要】

【課題を解決するための手段】

【０００８】

以下では、いくつかの例としての実施形態の要約を開示する。この要約は、そのような実施形態を読み手が基本的に理解できるように便宜的に提供されるものであり、本開示の範囲を完全に定義するものではない。この要約は、考えられるあらゆる実施形態の広範な概要ではなく、すべての実施形態の主要または重要な要素を識別することも、いずれかの態様またはすべての態様の範囲を示すことも意図していない。その唯一の目的は、１つまたは複数の実施形態のいくつかのコンセプトを、後から提示するさらに詳細な説明の導入部として簡略的な形で提示することである。本明細書では、本開示の１つの実施形態または複数の実施形態を指す目的で、便宜上、語「いくつかの実施形態」を使用することがある。

【０００９】

開示されている実施形態のさまざまな態様は、構成可能な処理アーキテクチャを実行時に最適化する方法を含む。本方法は、少なくとも１つの関数を実行するための複数の呼出しを受け取るステップと、受け取った複数の呼出しの間で少なくとも１つのパターンを識別するステップと、少なくとも１つの関数を計算するために、この少なくとも１つのパターンに基づいて、構成可能な処理アーキテクチャの少なくとも一部を操作するステップと、を含む。

【００１０】

開示されている実施形態のさまざまな態様は、構成可能な処理アーキテクチャをさらに

含む。本システムは、計算グリッドと、処理ユニットと、処理ユニットに接続されているメモリであって、メモリが命令を含み、命令が処理ユニットによって実行されたとき、命令が、少なくとも１つの関数を実行するための複数の呼出しを受け取り、受け取った複数の呼出しの間で少なくとも１つのパターンを識別し、少なくとも１つの関数を計算するために、この少なくとも１つのパターンに基づいて、構成可能な処理アーキテクチャ内の計算グリッドを操作する、ように、処理ユニットを構成する、メモリと、を備えている。

【００１１】

本明細書に開示されている主題は、本明細書の最後における請求項に具体的に示され、明確に特許請求されている。本発明の上記およびそれ以外の目的、特徴、および利点は、添付の図面を参照しながら行われる以下の詳細な説明から明らかであろう。

【図面の簡単な説明】

【００１２】

【図１Ａ】一実施形態に係る構成可能な処理アーキテクチャの概略図である。

【図１Ｂ】一実施形態に係る構成可能な処理アーキテクチャの計算グリッドの概略図である。

【図２Ａ】一実施形態に係る、それぞれのパターンの識別および関数の構成を示している概略的なシミュレーションである。

【図２Ｂ】一実施形態に係る、それぞれのパターンの識別および関数の構成を示している概略的なシミュレーションである。

【図２Ｃ】一実施形態に係る、それぞれのパターンの識別および関数の構成を示している概略的なシミュレーションである。

【図２Ｄ】一実施形態に係る、それぞれのパターンの識別および関数の構成を示している概略的なシミュレーションである。

【図３】一実施形態に係る構成可能なハードウェアの実行時の最適化の流れ図である。

【発明を実施するための形態】

【００１３】

重要な点として、本明細書に開示されている実施形態は、本明細書における革新的な教示内容の多数の有利な使用の例にすぎない。本出願の明細書中に行われている記述は、一般的に、特許請求されるさまざまな実施形態のいずれも必ずしも制限しない。さらに、いくつかの記述は、いくつかの独創的な特徴にあてはまるが、それ以外の特徴にはあてはまらないことがある。一般には、特に明記しない限り、一般性を失うことなく、１つの要素は複数の要素であってもよく、逆も同様である。図面においては、類似する数字は、いくつかの図を通じて類似する部分を指している。

【００１４】

図１Ａは、一実施形態に係る構成可能な処理アーキテクチャ１００の例としての概略図を示している。処理アーキテクチャ１００は、計算グリッド１１０と、メモリ（プログラムメモリ１０２など）に結合されている処理ユニット（回路）１０１とを含む。

【００１５】

処理ユニット１０１は、計算グリッド１１０でのプログラムコードの一部の実行を最適化するための処理を実行するように構成されている。プログラムコードの一部は、関数、基本ブロック、またはその両方（まとめて関数と称する）を含むことができる。基本ブロックとは、ブロックの中からのジャンプ、またはブロックの中へのジャンプが存在しないような、連続する命令を有するプログラムコードの断片である。

【００１６】

一実施形態においては、構成可能な処理アーキテクチャ１００は、計算装置の動作をアクセラレートさせるように構成されている。このような装置としては、例えば、マルチコア中央処理装置（ＣＰＵ）、フィールドプログラマブルゲートアレイ（ＦＰＧＡ）、グラフィック処理装置（ＧＰＵ）、特定用途向け集積回路（ＡＳＩＣ）、量子コンピュータ、光コンピューティング、ニューラルネットワークアクセラレータ、またはこれらの組合せ、が挙げられる。

【００１７】
開示されている実施形態によれば、アクセラレーションは、例えば、計算装置（図示していない）上ではなく、計算グリッド１１０上で関数を実行することによって達成される。計算グリッド１１０は、本明細書の中で後から図１Ｂを参照しながらさらに説明するように、論理素子のアレイおよびマルチプレクサ（ＭＵＸ）を含む。
【００１８】
計算グリッド１１０による関数の実行は、射影グラフ（projection graph）を使用して関数を計算グループに射影することによって行われる。計算グループとは、グリッド１１０内でＭＵＸを介して接続されている論理素子のセットである。グリッド１１０内で論理素子を正しく割り当てて選択することによって、関数の最適化された実行を達成することができる。
【００１９】
具体的には計算グリッド１１０は、グリッド１１０の動作を最適化するように、一部分が処理ユニット１０１によって構成される。この目的のため、処理ユニット１０１は、複数の最適化処理を実行する。
【００２０】
一実施形態においては、処理ユニット１０１は、計算装置内の計算要素によって動作する関数のための複数の呼出しを受け取るように構成されている。この呼出しは、複数の異なる関数、同じ関数、または両方の呼出し、メモリの異なる部分、メモリの同じ部分、または両方の呼出し、などとすることができる。関数は、算術演算、論理演算、またはその両方（ただしこれらに限定されない）を含む計算演算を実行することができる。
【００２１】
処理ユニット１０１は、受け取った複数の呼出しの間で少なくとも１つのパターンを識別するように構成されている。別の実施形態によれば、識別を外部処理によって行うことができ、その後にその識別を処理ユニット１０１に転送することができる。パターンは、一実施形態においては、２つ以上の関数が、ある所定のしきい値を超える関連性で再出現することである。しきい値は、特定の関数の再出現（re-occurrence）の特定の統計分析などに基づいて、経時的に動的に変更することができる。すなわち、例えば２つの関数が比較的頻繁に次々と実行されるとき、これら２つの関数の間の関連性のパターンが求められる。
【００２２】
処理ユニット１０１は、関数の実行を最適化するため、求められたパターンに基づいて計算グリッド１１０を構成する、または別の方法で操作する。関数の操作は、複数の処理（複製、拡張、縮小、インライン化、修正、またはこれらの組合せなどを含み、ただしこれらに限定されない）を通じて、達成することができる。
【００２３】
例としての一実施形態においては、再配置処理を実行するとき、処理ユニット１０１は、頻繁に互いを呼び出す２つの計算グループがグリッド１１０内でトポロジー的に互いに近くに配置されるように、計算グリッド１１０を構成する。
【００２４】
別の実施形態においては、処理ユニット１０１は、それまでの使用頻度が高い計算グループのセットが多くのインスタンスに複製されるように、計算グリッド１１０を構成する。さらなる実施形態においては、計算グループの呼出し元は、複製された新しい計算グループの間で負荷分散する。
【００２５】
さらに別の実施形態においては、インライン化処理を実行するときには、処理ユニット１０１は、頻繁に互いを呼び出す２つの論理上の計算グループが、これら２つの論理上の計算グループを結合する１つの論理上の計算グループとして構成および再構成されるように、計算グリッド１１０を構成する。
【００２６】

さらに別の実施形態においては、拡張処理を実行するときには、処理ユニット１０１は、分岐およびループを含む論理上の計算グループが、インライン化処理および再配置処理を使用してループが展開された状態で再構成されるように、計算グリッド１１０を構成する。

【００２７】

さらに別の実施形態においては、縮小処理を実行するときには、処理ユニット１０１は、最適化された実行をもはや提供しない複製または拡張された計算グループが、再構成により解放されるように、計算グリッド１１０を構成する。縮小された計算グループの呼出し元は、別の計算グループに参照される。参照がなくなった時点で、計算グループを再構成により解放することができる。すなわち、その計算グループに関連付けられるリソースを解放することができる。

【００２８】

なお、処理ユニット１０１は、電力消費量、メモリアクセス、レイテンシ、ゲート使用回数（gate usage count）、およびスループット、のうちの少なくとも１つに関連する必要性に従って、より良好な性能が達成されるように計算グリッド１１０を構成することに留意されたい。

【００２９】

処理ユニット１０１は、１つまたは複数のハードウェア論理要素および論理回路として実施することができる。使用することのできるハードウェア論理要素のタイプの実例としては、例えば、以下に限定されないが、汎用マイクロプロセッサ、マイクロコントローラ、マルチコアＣＰＵ、ＦＰＧＡ、ＧＰＵ、ＡＳＩＣ、量子プロセッサ、光コンピューティングプロセッサ、ニューラルネットワークアクセラレータ、粗粒度再構成可能アーキテクチャ（ＣＧＲＡ：coarse-grained configurable architecture）、インテリジェンス処理ユニット（intelligence processing unit）など、または、情報の計算もしくは別の操作を実行することのできる任意の他のハードウェア論理要素、が挙げられる。いくつかの実施形態においては、アクセラレートさせる対象の計算装置は、処理ユニット１０１の役割を果たす。

【００３０】

いくつかの実施形態においては、メモリバウンドである関数を含む、メモリ的に厳しいコード断片の分析が実行される。このような関数は、計算装置のメモリの近くに再配置される計算グループにマッピングされる。このような分析は、Ｉ／Ｏ関連の動作に適用することもできる。例えば、ネットワークポート、記憶装置、ＰＣＩ－ｅ、ビデオセンサーなどに関する動作である。

【００３１】

次に図１Ｂを参照し、例としての一実装形態においては、計算グリッド１１０は、複数の論理素子１２０（まとめて１つ以上のＬＥ　１２０と称する、または個別にＬＥ　１２０と称する）を含む。一実施形態においては、ＬＥ　１２０は、論理演算子（ＡＮＤ、ＯＲ、ＮＯＴ、ＸＯＲなど）またはこれらの組合せとすることができる。さらに別の実施形態においては、ＬＥ　１２０を、ルックアップテーブル（ＬＵＴ）演算を実行するように構成することができる。さらに別の実施形態においては、ＬＥ　１２０を、高レベルの算術関数（固定小数点数または浮動小数点数の加算、減算、乗算、除算、指数など）を実行するように構成することができる。さらに別の実施形態においては、ＬＥ　１２０は、シフト演算（左シフト、右シフトなど）を実行することができる。

【００３２】

なお、計算グリッド１１０内の各ＬＥ　１２０は、上述した演算関数のいずれか１つ以上に対応するように構成することができることに留意されたい。いくつかの構成においては、第１のグループの１つ以上のＬＥ　１２０が第１の演算を実行することができ（例：論理演算子）、第２のグループの１つ以上のＬＥ　１２０が別の演算（例：シフト演算）を実行することができ、以下同様である。特定の構成においては、すべてのＬＥ　１２０が、同じ演算関数を実行することができる。

【００３３】
一実施形態においては、計算グリッド１１０は、マルチプレクサ、デマルチプレクサ、スイッチなど複数のデータルーティング中継部（data routing junction）１３０（まとめて１つ以上のＭＵＸ　１３０と称する、または個別にＭＵＸ　１３０と称する）をさらに含む。ＭＵＸ　１３０は、データを１つ以上のＬＥ　１２０にルーティングする、および１つ以上のＬＥ　１２０からのデータをルーティングする、ように構成されている。
【００３４】
計算グリッド１１０は、複数の計算グループから構成されている。各計算グループは、１つのＭＵＸ　１３０を介して接続されているＮ×Ｍ個のＬＥ　１２０を含む。「Ｎ」および「Ｍ」の値は、異なるかまたは等しい整数である。一実施形態においては、Ｍは、Ｎ－１に等しい（Ｍ＝Ｎ－１）。ＭＵＸ　１３０は、計算グリッド１１０内の２つの隣接する行からの１つ以上のＬＥ　１２０を接続している。
【００３５】
一実施形態においては、各ＭＵＸ　１３０は、リテラル値を維持するための複数のレジスタ（図示していない）を含む。ＭＵＸ　１３０は、実行される演算の条件、接続テーブル、およびコード有するようにさらに構成される。１つ以上のＭＵＸ　１３０は、バス（図示していない）を介して処理ユニット１０１によって構成される。
【００３６】
別の実施形態においては、計算グリッド１１０は、データ入力メッシュネットワーク（data input mesh network）の役割を果たす、フィーダ（feeder）（Ｆｒ）１４０のメッシュネットワークを含む。フィーダ１４０は、１つ以上のＬＥ　１２０によって処理するために入力（例えば関数のパラメータ、ただしこれに限定されない）を送り込むように構成されている。一実施形態においては、フィーダ１４０は、内部チップ相互接続（ＩＣＩ：internal chip interconnection）として実施される。
【００３７】
なお、計算グリッド１１０の操作は、アクセラレートさせる対象の計算装置の動作と同時にリアルタイムで実行されることを理解されたい。一実施形態によれば、操作は、１つまたは複数の関数を別の実行装置に移動させるステップをさらに含むことができ、すなわち、ある関数がアクセラレータ内に必要ではなく、したがってその関数がＣＰＵにおいて実行されるように再配置されることを決定することができる。
【００３８】
なお、１つ以上のＬＥ　１２０、１つ以上のＭＵＸ　１３０、および１つ以上のフィーダ１４０は、ハードウェア、ソフトウェア、ファームウェア、またはこれらの任意の組合せにおいて実施してよいことをさらに理解されたい。例示的な一実施形態においては、計算グリッド１１０（およびそのさまざまな素子）は、半導体デバイスとして実施される。別の実施形態においては、計算グリッド１１０は、一部が半導体デバイスとして実施され、一部がソフトウェア、ファームウェア、またはこれらの組合せとして実施される。一例として、より高い確率で、または多くの回数出現するものと判定される論理経路を、半導体デバイスにおいて実施することができ、その一方で、より低い確率で出現する、またはめったに出現しないと判定される論理経路を、ソフトウェアにおいて実施することができる。なお、このような実施形態は、同種の実施に頼る別の実施形態よりも、例えば実行時間や電力消費量における全体的な予測見返りコストが低くなりうることに留意されたい。論理経路は、処理ユニット１０１によって、または計算グリッド１１０内の専用ハードウェア素子によって検出することができる。
【００３９】
図２Ａおよび図２Ｂは、それぞれ、一実施形態に係る、特定の関数の再配置の例としてのシミュレーション２００Ａおよび２００Ｂである。再配置処理は、頻繁に互いを呼び出す２つの論理上の計算グループ（基本ブロックまたは関数など）が、計算グリッド内で互いに物理的に近くに位置するように、計算グリッドを構成または操作するステップを含む。
【００４０】

図２Ａおよび図２Ｂに示した例としてのシミュレーションには、複数のＬＥ　２２０－１～２２０－Ｎを有する計算グリッド２１０を示してあり、Ｎは１より大きいかまたは１に等しい整数である（グリッド２１０には、単に図面を簡略化する目的でＭＵＸは含まれていない）。計算グループが、射影グラフに基づいてそれぞれのＬＥ　２２０にマッピングまたは射影されている。ＬＥ　２２０－１，２２０－２，２２０－３を含む計算グループが、ＬＥ　２２０－４，２２０－５，２２０－６を含む計算グループに関連付けられることを識別した時点で、図２Ｂに示したように、ＬＥ　２２０－１，２２０－２，２２０－３を含む計算グループが、ＬＥ　２２０－４，２２０－５，２２０－６を含む計算グループの近傍に再配置される。

【００４１】

図２Ｃおよび図２Ｄは、それぞれ、特定の関数の複製の、例としてのシミュレーション２００Ｃおよび２００Ｄである。複製処理は、計算の速度を高め、かつボトルネックを広げる目的で、頻繁に呼び出される計算グループにマッピングされている１つ以上のＬＥ２２０がグリッド２１０上で何度か複製されるように、計算グリッド２１０を構成するステップを含む。

【００４２】

例としての一実施形態においては、複製処理は、新たに複製されたインスタンスを対象とする負荷分散処理をさらに含むことができ、これは例えば、呼出し元からの元のブロックの参照を、呼出し元がすべての複製を均一に呼び出すように再構成することによる。別の例としては、呼出し元からの参照が、複製を負荷分散する、新たに構成可能なＭＵＸに再構成される。

【００４３】

頻繁に呼び出される関数の計算グループを複製することによって計算グリッド２１０を最適化することができると判定されたとき、そのような各グループが複製され、元のグループから遠く離して再配置される。図２Ｃおよび図２Ｄに示した例では、ＬＥ　２２０－４，２２０－５，２２０－６を含む計算グループが複製される。複製されたグループは、このような関数の呼出しを強化する目的で、計算グリッド２１０上で元のグループのＬＥから遠く離して配置される。このことは図２Ｄに示してあり、図２Ｄは、図２Ｃに示した配置との比較における、複製されたグループを含む。複製されたグループを、他のグループより低い頻度で呼び出される関数のグループに置き換えることができる。これに代えて、複製された論理計算要素が性能に必要ないものと識別されたとき、操作は縮小処理、すなわち複製解放処理を含むことができ、したがってグリッドから削除される。

【００４４】

図３は、一実施形態に係る、構成可能な処理アーキテクチャにおいて動作する計算グリッドを実行時に最適化する方法を示す、例としての流れ図３００である。Ｓ３１０においては、少なくとも１つの関数を実行するための複数の要求を受け取る。これらの要求は、演算を実行するためにインタフェースを介して受け取る呼出しとすることができる。一例として、要求された演算は、例えばプログラムメモリ（例：図１Ｂのプログラムメモリ１０２）からのメモリのロードとすることができる。

【００４５】

Ｓ３２０においては、受け取ったシステムコールを分析して、２つ以上のシステムコールの間で少なくとも１つのパターンが識別されるかをチェックする。この識別は、呼出しを次々と受け取ったことの識別、または呼出しを次々と受け取らなかったことの識別とすることができる。一例として、パターンは、２つの関数が次々に実行されることを示す。このようなパターンが識別される場合、実行はＳ３３０に続き、そうでない場合、実行はＳ３１０に戻る。

【００４６】

Ｓ３３０においては、識別されたパターンを分析して、計算グリッドの修正または再構成が必要であるかを判定する。例えば、分析は、例えばある関数を実行するためにすでに割り当てられた計算グループがグリッド内に存在することをパターンが示しているか判定

するステップを含む。別の例としては、分析は、関数の再出現をパターンが示しているか判定するステップを含む。別の例としては、分析は、１つの関数が別の関数を呼び出すことをパターンが示しているか判定する。

【００４７】

Ｓ３４０においては、分析に基づいて、計算グリッドの操作が必要であるかをチェックする。必要である場合、実行はＳ３５０に続く。そうでない場合、実行はＳ３６０に続く。

【００４８】

Ｓ３５０においては、受け取った呼出しによって呼び出される関数を最適化するために、計算グリッドを修正する、または別の方法で再構成する。計算グリッドの修正または再構成の一部として実行することのできる処理のさまざまな例は、上に説明してある。

【００４９】

Ｓ３６０においては、さらなる要求を受け取っているかをチェックし、受け取っている場合、実行はＳ３１０に続き、そうでない場合、実行は終了する。

【００５０】

本明細書に開示されている実施形態は、ハードウェア、ファームウェア、ソフトウェア、またはこれらの任意の組合せとして実施することができる。任意の適切なアーキテクチャを備えたマシンに、アプリケーションプログラムをアップロードして実行することができる。マシンは、１つまたは複数の中央処理装置（「ＣＰＵ」）、メモリ、および入力／出力インタフェースなどのハードウェアを有するコンピュータプラットフォーム上に実施されることが好ましい。

【００５１】

コンピュータプラットフォームは、オペレーティングシステムおよびマイクロ命令コードをさらに含むことができる。本明細書に記載されているさまざまな処理および関数は、ＣＰＵ（そのようなコンピュータまたはプロセッサが明示的に示されているか否かを問わない）によって実行することのできるマイクロ命令コードの一部またはアプリケーションプログラムの一部のいずれか、またはこれらの任意の組合せとすることができる。

【００５２】

これに加えて、コンピュータプラットフォームには、追加のネットワークファブリック記憶装置（network fabric storage unit）や印刷装置など、さまざまな他の周辺装置を接続することができる。さらに、非一過性のコンピュータ可読媒体は、一過性の伝搬信号を除く任意のコンピュータ可読媒体である。

【００５３】

本明細書に記載されているすべての例、および条件を表す語（conditional language）は、この技術分野を発展させるために本発明者によって提供される本開示およびコンセプトの原理を読み手が理解できるように支援するための教示目的を意図しており、そのような具体的に記載されている例および条件に限定されないものと解釈されたい。

【００５４】

なお、本明細書において「第１の」、「第２の」などの指定を使用して要素が言及されている場合、一般的には、それによって、それらの要素の数量または順序が制限されることはない。そうではなく、本明細書では、これらの指定は、一般に２つ以上の要素、または要素の２つ以上のインスタンスを区別する簡便な方法として使用されている。したがって、第１の要素および第２の要素という表現は、２つの要素のみを採用してよいことを意味するのではなく、何らかの形で第１の要素が第２の要素に先行しなければならないことを意味する。また、特に明記されていない限り、要素のセットは、１つまたは複数の要素を含む。

【００５５】

本明細書において使用されているとき、後ろに項目のリストが続く句「の少なくとも１つ」は、リストされている項目のいずれかを個別に利用することができる、またはリストされている項目の２つ以上の任意の組合せを利用できることを意味する。例えば、システムが、「Ａ、Ｂ、Ｃの少なくとも１つ」を含むものと説明されている場合、そのシステム

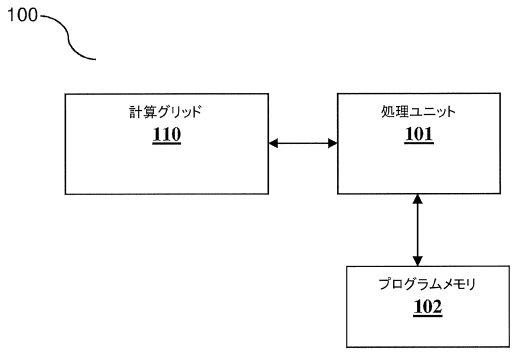
は、Ａのみを含む、Ｂのみを含む、Ｃのみを含む、ＡとＢを組合せで含む、ＢとＣを組合せで含む、ＡとＣを組合せで含む、またはＡとＢとＣを組合せで含む、ことができる。

【００５６】

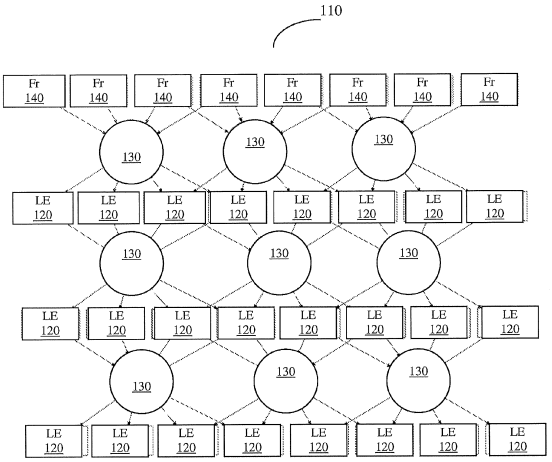
本出願は、米国仮特許出願第６２／５４０，８４９号（出願日：２０１７年８月３日）および米国仮特許出願第６２／５５８，０９０号（出願日：２０１７年９月１３日）の利益を主張し、これらの文書の内容は参照により本明細書に組み込まれている。

【図面】

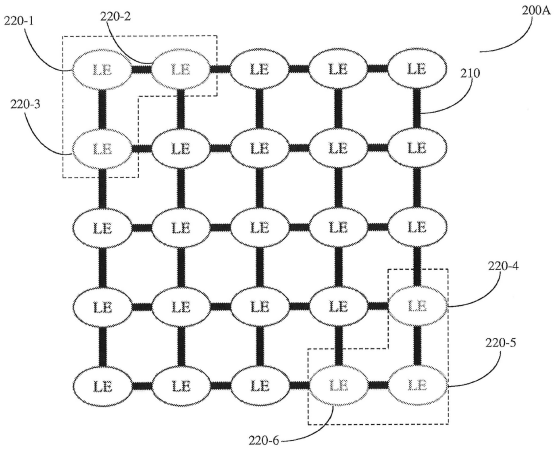
【図１Ａ】



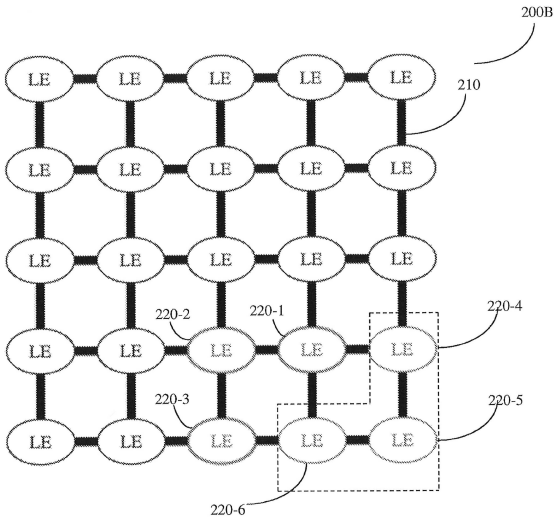
【図１Ｂ】



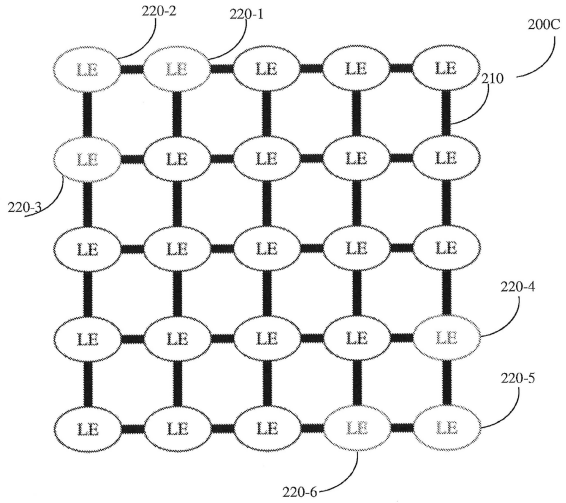
【図２Ａ】



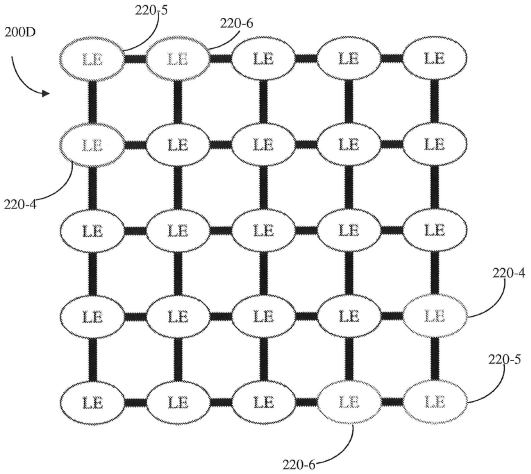
【図２Ｂ】



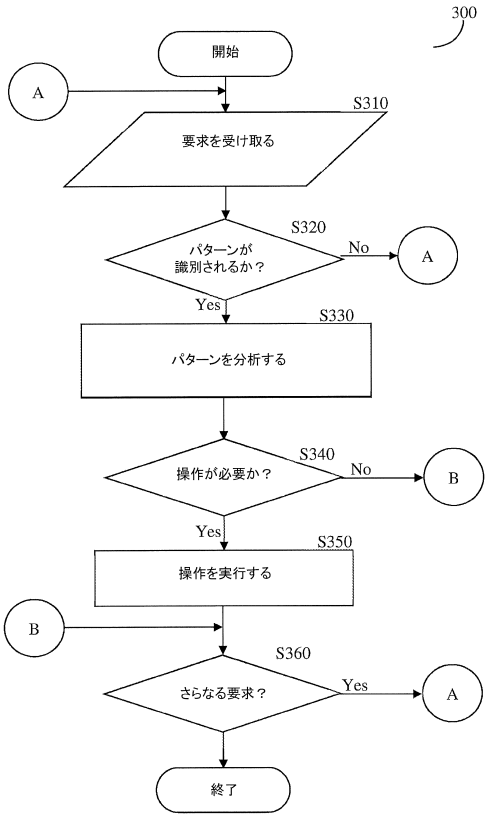
【図２Ｃ】


220-2
220-1
200C
210
220-3
220-4
220-5
220-6
LE

【図２Ｄ】


200D
220-5
220-6
220-4
220-4
220-5
220-6
LE

【図３】


300
開始
A
S310
要求を受け取る
S320
パターンが
識別されるか？
No
A
Yes
S330
パターンを分析する
S340
操作が必要か？
No
B
Yes
S350
操作を実行する
B
S360
さらなる要求？
Yes
A
終了

フロントページの続き

(33)優先権主張国・地域又は機関
米国(US)

(56)参考文献
特開２００９－１６３３２８（ＪＰ，Ａ）
特表２００２－５２６８２６（ＪＰ，Ａ）
国際公開第２０１７／０２９７４３（WO，Ａ１）
特開２００９－２３８２２１（ＪＰ，Ａ）
特表２００５－５０５０３０（ＪＰ，Ａ）
国際公開第２００８／０２６７３１（WO，Ａ１）
佐野 雅彦 他，ＦＰＧＡ化のためのｋ－ａｒｙ ｎ－ｃｕｂｅ型相互結合網用のルータの設計，情報処理学会研究報告，社団法人情報処理学会，1996年05月16日，第96巻 第39号，第31頁－第36頁

(58)調査した分野 (Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　９／３８
Ｇ０６Ｆ　１５／８０