(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2014-182817
(P2014-182817A)
(43)公開日　平成26年9月29日(2014.9.29)

(51)Int.Cl.			F I			テーマコード（参考）
G06F	9/318	(2006.01)	G06F	9/30	320C	5B033
G06F	9/32	(2006.01)	G06F	9/32	320F	
			G06F	9/32	330C	

審査請求 有　請求項の数 33　OL　外国語出願　（全 41 頁）

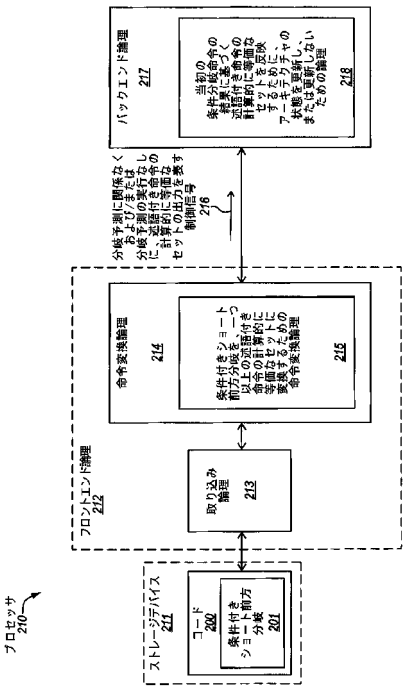
(21)出願番号　特願2014-47202(P2014-47202)
(22)出願日　平成26年3月11日(2014.3.11)
(31)優先権主張番号　13/838,450
(32)優先日　平成25年3月15日(2013.3.15)
(33)優先権主張国　米国(US)

(71)出願人　591003943
インテル・コーポレーション
アメリカ合衆国　９５０５４　カリフォルニア州・サンタクララ・ミッション　カレッジ　ブーレバード・２２００
(74)代理人　110000877
龍華国際特許業務法人
(72)発明者　グロチョウスキ、エドワード　ティー.
アメリカ合衆国　９５０５４　カリフォルニア州・サンタクララ・ミッション　カレッジ　ブーレバード・２２００　インテル・コーポレーション内

最終頁に続く

(54)【発明の名称】条件付きショート前方分岐の計算的に等価な述語付き命令への変換

(57)【要約】
【課題】条件分岐を処理するプロセッサを提供する。
【解決手段】本プロセッサは、条件付きショート前方分岐を取り込むための命令取り込み論理を含む。
当該条件付きショート前方分岐は、条件分岐命令と、当該条件分岐命令にプログラム順に連続して続く一つ以上の命令のセットと、を含み得る。この一つ以上の命令のセットは、条件分岐命令と当該条件分岐命令によって示される前方分岐ターゲット命令との間にある。また、本プロセッサは、命令取り込み論理と連結された命令変換論理を含む。命令変換論理は、条件付きショート前方分岐を、一つ以上の述語付き命令の計算的に等価なセットに変換するものである。また、他の諸プロセッサを、様々な方法およびシステムと併せ開示する。
【選択図】図２



【特許請求の範囲】
【請求項１】
条件分岐を処理するプロセッサであって、
条件付きショート前方分岐を取り込むための命令取り込み論理であって、前記条件付きショート前方分岐は、条件分岐命令と、一つ以上の命令のセットとを含み、前記一つ以上の命令のセットは、前記条件分岐命令と前記条件分岐命令によって示される前方分岐ターゲット命令との間にある、前記条件分岐命令にプログラム順に連続して続く命令取り込み論理と、
前記命令取り込み論理と連結された命令変換論理であって、前記条件付きショート前方分岐を、一つ以上の述語付き命令の計算的に等価なセットに変換する、命令変換論理と
を含む、プロセッサ。
【請求項２】
前記命令変換論理は、前記条件分岐命令を除去することができる、請求項１に記載のプロセッサ。
【請求項３】
前記命令変換論理が、前記条件分岐命令と前記前方分岐ターゲット命令との間にある前記一つ以上の命令の前記セットの各々を、述語なし命令から述語付き命令に変換する、請求項１または２に記載のプロセッサ。
【請求項４】
前記命令変換論理が、前記条件分岐命令と前記前方分岐ターゲット命令との間にある複数の命令の各々を、述語なし命令から述語付き命令に変換する、請求項１～３のいずれか一項に記載のプロセッサ。
【請求項５】
前記命令変換論理が、前記条件分岐命令が条件成立であるか、または不成立であるかの予測に関係なく、前記一つ以上の述語付き命令の前記計算的に等価なセットを表す信号を出力する、請求項１～４のいずれか一項に記載のプロセッサ。
【請求項６】
前記命令変換論理が、前記プロセッサのパイプラインの解読段階にハードウェア論理を含む、請求項１～５のいずれか一項に記載のプロセッサ。
【請求項７】
前記条件分岐命令と前記前方分岐ターゲット命令との間にある前記一つ以上の命令の前記セットが単一の移動命令からなり、前記命令変換論理は、前記単一の移動命令を条件付き移動命令に変換する、請求項１～６のいずれか一項に記載のプロセッサ。
【請求項８】
前記条件付き移動命令は、前記条件付き移動命令の条件が偽であるとき、例外処理を開始しない、請求項７に記載のプロセッサ。
【請求項９】
前記命令変換論理は、前記命令変換論理が前記条件分岐命令に対する分岐予測を知る必要なしに、前記一つ以上の述語付き命令の前記計算的に等価なセットを表す信号を出力する、請求項１に記載のプロセッサ。
【請求項１０】
前記命令変換論理と連結されたバックエンド論理をさらに含み、前記バックエンド論理は、前記一つ以上の述語付き命令の前記計算的に等価なセットを実行し、前記条件分岐命令が条件成立であると判定されると、前記一つ以上の述語付き命令の前記計算的に等価なセットの前記実行を反映するために、アーキテクチャの状態の更新をしないと決める、請求項１に記載のプロセッサ。
【請求項１１】
前記命令変換論理と連結されたバックエンド論理をさらに含み、前記バックエンド論理は、前記一つ以上の述語付き命令の前記計算的に等価なセットを実行し、前記条件分岐命令が条件不成立であると判定されると、前記一つ以上の述語付き命令の前記計算的に等価

なセットの前記実行を反映するために、アーキテクチャの状態の更新をすると決める、請求項１に記載のプロセッサ。
【請求項１２】
前記命令取り込み論理が、前記条件分岐命令の予測と関係なく、前記条件分岐命令と前記前方分岐ターゲット命令との間にある前記一つ以上の命令を常に取り込む、請求項１に記載のプロセッサ。
【請求項１３】
条件分岐を処理する方法であって、
条件付きショート前方分岐を取り込む段階であって、前記条件付きショート前方分岐は、条件分岐命令と、一つ以上の命令のセットとを含み、前記一つ以上の命令のセットは、前記条件分岐命令と前記条件分岐命令によって示される前方分岐ターゲット命令との間にある、前記条件分岐命令にプログラム順に連続して続く段階と、
前記条件付きショート前方分岐を、一つ以上の述語付き命令の計算的に等価なセットに変換する段階と
を含む、方法。
【請求項１４】
前記条件分岐命令が条件成立であるか、または不成立かどうかの予測に関係なく、前記一つ以上の述語付き命令の前記計算的に等価なセットを表す信号を、プロセッサのバックエンド論理に提供する段階をさらに含む、請求項１３に記載の方法。
【請求項１５】
前記変換する段階が、前記条件分岐命令と前記前方分岐ターゲット命令との間の前記一つ以上の命令の前記セットの各々を、述語なし命令から述語付き命令に変換する段階を含む、請求項１３または１４に記載の方法。
【請求項１６】
前記変換する段階が、前記条件分岐命令と前記前方分岐ターゲット命令との間の複数の命令の各々を、述語なし命令から述語付き命令に変換する段階を含む、請求項１３または１４に記載の方法。
【請求項１７】
前記変換する段階が前記条件分岐命令を除去する段階を含み、前記変換する段階は、プロセッサのパイプラインの解読段階で変換する段階を含む、請求項１３または１４に記載の方法。
【請求項１８】
分岐予測論理のオペレーションに関係なく、前記一つ以上の述語付き命令の前記計算的に等価なセットを表す信号を、プロセッサのバックエンド論理に提供する段階をさらに含む、請求項１３～１７のいずれか一項に記載の方法。
【請求項１９】
前記一つ以上の述語付き命令の前記計算的に等価なセットを実行する段階と、
実際上、前記条件分岐命令が条件成立であると判定する段階と、
前記条件分岐命令が条件成立であると判定するのに応じて前記一つ以上の述語付き命令の前記計算的に等価なセットの前記実行を反映するために、アーキテクチャの状態を更新しないと決める段階と、
をさらに含む、請求項１３～１８のいずれか一項に記載の方法。
【請求項２０】
条件分岐を処理する方法であって、
条件分岐命令を検出する段階と、
前記条件分岐命令の後にショート前方分岐が続いていると判定する段階と、
前記ショート前方分岐内の全ての命令が、対応する計算的に等価な述語付き命令に変換可能なことを判定する段階と
を含む、方法。
【請求項２１】

前記ショート前方分岐内の前記命令の前記全てを、前記対応する計算的に等価な述語付き命令に変換する段階と、前記条件分岐命令を除去する段階と、をさらに含む、請求項２０に記載の方法。

【請求項２２】

前記計算的に等価な述語付き命令を実行する段階と、

前記条件分岐命令が条件成立であると最終的に判定する段階と、

前記条件分岐命令が条件成立であると判定した後、前記計算的に等価な述語付き命令の前記実行を反映するために、アーキテクチャの状態を更新しないと決める段階と、

をさらに含む、請求項２０に記載の方法。

【請求項２３】

条件分岐を処理するためのシステムであって、

相互接続と、

前記相互接続に連結されたプロセッサであって、

条件付きショート前方分岐を取り込むための命令取り込み論理であって、前記条件付きショート前方分岐は、条件分岐命令と、一つ以上の命令のセットとを含み、前記一つ以上の命令のセットは、前記条件分岐命令と前記条件分岐命令によって示される前方分岐ターゲット命令との間にある、前記条件分岐命令にプログラム順に連続して続く命令取り込み論理と、

前記命令取り込み論理と連結された命令変換論理であって、前記条件付きショート前方分岐を、一つ以上の述語付き命令の計算的に等価なセットに変換する、前記命令変換論理と

を含む前記プロセッサと、

前記相互接続と連結されたダイナミックランダムアクセスメモリ（ＤＲＡＭ）と

を備える、システム。

【請求項２４】

前記命令変換論理が前記条件分岐命令を除去し、さらに、前記命令変換論理は、前記条件分岐命令と前記前方分岐ターゲット命令との間にある前記一つ以上の命令の前記セットの各々を、述語なし命令から述語付き命令に変換する、請求項２３に記載のシステム。

【発明の詳細な説明】

【技術分野】

【０００１】

本明細書で説明する諸実施形態は一般にプロセッサに関する。具体的には、本明細書で説明する諸実施形態は、一般にプロセッサ内での条件分岐の取り扱いに関する。

【背景技術】

【０００２】

ある種のプロセッサは、パイプライン化された実行を用いて実行段階をオーバーラップさせる。これは、複数の命令が同時に相異なる実行の段階にあることを可能にし、パフォーマンスの改善に役立つ。達成される並行処理の量は、パイプラインの深さが増すにつれ増加する傾向がある。時間がたつにつれ、特定のプロセッサは、パフォーマンスの改善を図って益々深いパイプラインを組み込むようになった。これらの深いパイプラインは、命令ストリームが分かっている場合は、パイプラインを満杯に保つことができ、後続命令の実行がパイプライン中の先行命令の結果を待つ必要がなく、より効率的となることになる。

【０００３】

一つの難題は、プロセッサによって実行されるプログラムまたはコードは、通常、条件分岐を包含していることである。かかる条件分岐の例には、「条件が一致、もしくは不一致の場合にジャンプする」型の命令、または当該技術分野で既知の他の条件付き制御のフロー変更命令が含まれる。条件分岐は、実行のフローを、条件によって二つの取り得る方向の一つに分岐させることができる。これら二つの方向は、多くの場合「条件成立パス」および「条件不成立パス」と呼ばれる。「条件不成立パス」は、通常、実行されているコ

ード中の次に続く命令に進み、一方、「条件成立パス」は、通常、途中にある一つ以上の命令を越えて不連続なターゲット命令にジャンプ、移動、または分岐する。分岐条件が成立するか、または不成立かどうかは、一般に、当該命令に関連付けられた条件の評価によって決まる（例えば、当該条件に一致するかしないか）。

【０００４】

パフォーマンスの改善に役立つように、現在のほとんどのプロセッサは、条件分岐の実際の方向が決まる前に、条件分岐の方向を予測するのを助力するための分岐予測器を備えている。一般に、条件分岐の実際の方向は、パイプラインの後の段階で条件が実際に評価されるまでは完全に未知である。しかしながら、分岐予測器は、分岐予測メカニズムまたは論理を用いて、条件分岐の方向を（例えば、過去の履歴に基づいて）予測することができる。これは、プロセッサのパフォーマンスを改善するための助力となり得る。分岐予測器がなければ、プロセッサは、さらなる命令をパイプライン中に取り込むことが可能になるために、条件分岐命令に関連付けられた条件の評価を待たねばならないこともあろう。しかしながら、分岐予測器は、最も可能性の高い条件分岐の方向を予測することによって、かかる時間浪費を回避する助力をすることができる。次いで、予測された分岐方向を用いてさらなる命令を取り込み、それらを推測で実行することが可能になる。

【発明の概要】

【発明が解決しようとする課題】

【０００５】

最終的には、予測された分岐方向が正しかったか、間違っていたかが判明することになる。予測された分岐方向が正しいことが判明した場合、推測で実行された命令の結果および／または状態を用いることができる。この場合、推測がなければ、条件分岐の実際の方向の評価を待つ間に休止状態だった、または少なくとも不十分な活用状態だった可能性のある、パイプライン段階のより高い利用率のおかげで、一般的にはプロセッサのパフォーマンスおよびスピードが増大することになる。しかしながら、逆に予測された分岐方向が間違っていた（例えば、分岐予測器がミス予測をした）ことが判明した場合は、推測で実行された命令による、当該条件分岐命令以降の一切の結果および／または状態は、通常、放棄しなければならないことになる。多くの場合、パイプラインはフラッシュされる（パイプライン中で現在走行中の命令が放棄される）ことになり、実行は、ミス予測が生じた条件分岐に巻き戻されて、今や正しいことが分かった別の分岐方向に再開始されることになろう。この結果は、パフォーマンス上の不利益およびエネルギ上の不利益の両方をもたらしがちなので、一般に望ましくない。

【０００６】

本発明は、本発明の実施形態を例示するために用いられる、以下の説明および添付の図面を参照することによって最良に理解することができよう。

【図面の簡単な説明】

【０００７】

【図１】条件付きショート前方分岐を含む、プログラムまたはコードの部分のブロック流れ図である。

【０００８】

【図２】条件分岐を処理するプロセッサの実施形態のブロック図である。

【０００９】

【図３】条件分岐を処理する方法の実施形態のブロック流れ図である。

【００１０】

【図４】条件付きショート前方分岐の検出および変換論理を示すブロック図である。

【００１１】

【図５】条件付きショート前方分岐を検出し変換するかどうかを判定する方法の実施形態のブロック流れ図である。

【００１２】

【図６】条件付きショート前方分岐の一つ以上の述語付き命令の計算的に等価なセットへ

の変換の例示的な実施形態を示す。
【００１３】
【図７】条件付きショート前方分岐を表しこれと計算的に等価な、一つ以上の述語付き命令のセットを反映するために、アーキテクチャの状態を更新し、または更新しないバックエンド論理の実施形態のブロック図である。
【００１４】
【図８Ａ】本発明の実施形態による、例示的なインオーダーパイプラインと、例示的なレジスターリネームおよびアウトオブオーダーの発行／実行のパイプラインと、の両方を示すブロック図である。
【００１５】
【図８Ｂ】本発明の実施形態による、例示的実施形態のインオーダーアーキテクチャコアと、プロセッサに含めるための例示的なレジスターリネームおよびアウトオブオーダーの発行／実行のアーキテクチャコアと、の両方を示すブロック図である。
【００１６】
【図９Ａ】本発明の実施形態による、単一のプロセッサコアと、オンダイ相互接続ネットワークへの当該コアの接続およびそのレベル２（Ｌ２）キャッシュのローカルサブセットとのブロック図である。
【００１７】
【図９Ｂ】本発明の実施形態による、図９Ａ中のプロセッサコアの部分の拡大図である。
【００１８】
【図１０】本発明の実施形態による、複数のコアを有することが可能であり、統合メモリコントローラを有することが可能で、且つ統合グラフィックスを有することが可能なプロセッサのブロック図である。
【００１９】
【図１１】本発明の実施形態による、システムのブロック図を示す。
【００２０】
【図１２】本発明の実施形態による、第一のさらに具体的な例示システムのブロック図を示す。
【００２１】
【図１３】本発明の実施形態による、第二のさらに具体的な例示システムのブロック図を示す。
【００２２】
【図１４】本発明の実施形態による、ＳｏＣのブロック図を示す。
【００２３】
【図１５】本発明の実施形態による、ソース命令セット中のバイナリ命令をターゲット命令セット中のバイナリ命令に変換するためのソフトウェア命令コンバータの使用を対照しているブロック図である。
【発明を実施するための形態】
【００２４】
本明細書において、条件付きショート前方分岐を、一つ以上の述語付き命令および／または条件付きで実行される命令の計算的に等価なセットに変換するためのプロセッサ、方法、およびシステムを開示する。以下の説明では、数多くの具体的詳細が記載される（例えば、条件分岐命令の具体的な型、具体的な条件付きショート前方分岐、論理の実装、プロセッサ構成、マイクロアーキテクチャの詳細、オペレーションのシーケンス、論理の分割／統合の詳細、システムコンポーネントの種類および相互関係など）。但し、本発明の実施形態はこのような具体的詳細がなくても実践が可能であることを理解されたい。別の例では、本説明の理解を曖昧にしないため、周知の回路、構造体、および技法は詳細に示していない。
【００２５】
図１は、プロセッサによって実行されるプログラムまたはコード１００の一部のブロッ

ク流れ図であって、条件付きショート前方分岐１０１を含む。条件分岐命令１０３が、この条件付きショート前方分岐の開始部に置かれている。この条件分岐命令は「条件成立パス」および「条件不成立パス」を有する。条件成立パスは、条件分岐命令によって示された前方分岐ターゲット命令１０６に進む。条件分岐命令は、例えば、前方分岐ターゲット命令を示すための（例えば、条件分岐命令から分岐ターゲット命令までのオフセットを指定する）引数またはソースオペランドを有することが可能である。

【００２６】

様々な実施形態において、この条件分岐命令は、条件付きジャンプ命令、条件付き制御フロー飛び越し命令、または当該技術分野で既知の他の型の条件分岐命令を表し得る。Ｉｎｔｅｌアーキテクチャは、条件付きジャンプ命令のいくつかの適切な例を含む。適切な「条件が一致すればジャンプ」（ｊｃｃ）命令のいくつかの代表的な例としては、（ａ）上回れば（ｃａｒｒｙ ｆｌａｇ＝０ ａｎｄ ｚｅｒｏ ｆｌａｇ＝０）ショートジャンプする命令（ＪＡ）、（ｂ）キャリーならば（ｃａｒｒｙ ｆｌａｇ＝１）ショートジャンプする命令（ＪＣ）、（ｃ）０ならば（ｚｅｒｏ ｆｌａｇ＝１）ニアジャンプする命令（ＪＺ）、（ｄ）ゼロでなければ（ｚｅｒｏ ｆｌａｇ＝０）ショートジャンプする命令（ＪＮＺ）、（ｅ）以下ならば（ｃａｒｒｙ ｆｌａｇ＝１ ｏｒ ｚｅｒｏ ｆｌａｇ＝１）ニアジャンプする命令 （ＪＢＥ）、および（ｆ）大きくないならば（ｚｅｒｏ ｆｌａｇ＝１ ｏｒ ｓｉｇｎ ｆｌａｇ≠ＯＦ）ニアジャンプする命令（ＪＮＧ）が挙げられるが、これらに限定されない。また、Ｉｎｔｅｌ（登録商標）Ｉｔａｎｉｕｍ（登録商標）アーキテクチャも適切な分岐命令を含む。例えば、ｂｒ（分岐）命令は、分岐命令が限定述語を使って分岐をするかどうかを判定する、ｃｏｎｄ（条件）型を有する。述語が１の場合、分岐が成立し、１でない場合は不成立である。また、いくつかのＲＩＳＣアーキテクチャも、条件分岐命令のいくつかのの適切な例を含む。かかる条件分岐命令のいくつかの代表的な例としては、（ａ）等しければ分岐の命令（ＢＥＱ：ｂｒａｎｃｈ ｉｆ ｅｑｕａｌ）、（ｂ）不等であれば分岐の命令（ＢＮＥ：ｂｒａｎｃｈ ｉｆ ｎｏｔ ｅｑｕａｌ）、（ｃ）キャリークリアならば分岐の命令（ＢＣＣ：ｂｒａｎｃｈ ｉｆ ｃａｒｒｙ ｃｌｅａｒ）、（ｄ）キャリーセットならば分岐の命令（ＢＣＳ：ｂｒａｎｃｈ ｉｆ ｃａｒｒｙ ｓｅｔ）、（ｅ）符号付きが条件より大ならば分岐の命令（ＢＧＴ：ｂｒａｎｃｈ ｉｆ ｓｉｇｎｅｄ ｇｒｅａｔｅｒ ｔｈａｎ）、および（ｆ）オーバーフローがなければ分岐の命令（ＢＶＣ：ｂｒａｎｃｈ ｉｆ ｎｏ ｏｖｅｒｆｌｏｗ）が挙げられるが、これらに限定されない。

【００２７】

「条件不成立パス」は、条件分岐命令と当該条件分岐命令により示された前方分岐ターゲット命令との間にある、条件分岐命令にプログラム順に連続して続く一つ以上の命令のセット１０２に進む。図示の実施形態において、これらには、プログラムの順に条件分岐命令に直につながる少なくとも一つの命令１０４が含まれる。随意的に、いくつかの実施形態において、条件付きショート前方分岐１０１は、条件分岐命令と前方分岐ターゲット命令との間で、プログラムの順に一つ以上の他の任意の順次的な命令１０５を含めることもできる。いくつかの実施形態において、条件付きショート前方分岐には、条件分岐命令と前方分岐ターゲット命令との間に１～約５個または１～約３個の命令を含めることが可能であるが、但し、本発明の範囲はこれに限定されない。他の実施形態では、所与のアーキテクチャにおいて（例えばＩｔａｎｉｕｍアーキテクチャにおけるように）十分に多様な条件付き／述語付き命令が利用可能であることを前提として、もっと大きな条件前方分岐を使用することも可能である。

【００２８】

図２は、条件分岐を処理するプロセッサ２１０のある実施形態のブロック図である。いくつかの実施形態において、このプロセッサは汎用プロセッサ（例えば、デスクトップ、ラップトップ、および類似のコンピュータに使われるような種類の汎用マイクロプロセッサ）とすることができる。あるいは、このプロセッサを特定用途向けプロセッサとすることも可能である。適切な特定用途向けプロセッサの例としては、ネットワークプロセッサ

、通信用プロセッサ、暗号用プロセッサ、グラフィックスプロセッサ、コプロセッサ、内蔵プロセッサ、デジタル信号プロセッサ（ＤＳＰ）、およびコントローラ（例えば、マイクロコントローラ）が挙げられるが、これらに限定されない。このプロセッサは、様々な複合命令セットコンピューティング（ＣＩＳＣ）プロセッサ、様々な縮小命令セットコンピューティング（ＲＩＳＣ）プロセッサ、様々な超長命令語（ＶＬＩＷ）プロセッサ、これらの様々なハイブリッド、または他の種類全体のプロセッサの任意のものとすることができる。

【００２９】

このプロセッサは、フロントエンド論理２１２を有する。このフロントエンド論理は、命令取り込み論理２１３を含む。この命令取り込み論理は、ストレージデバイス２１１から、条件付きショート前方分岐２０１の命令を含め、コード２００の命令を取り込む。様々な実施形態において、このストレージデバイスには、キャッシュ（例えば、命令キャッシュ、ならびに命令およびデータキャッシュなど）またはプロセッサがシステム中に展開される際に連結が可能なメモリを含めることができる。いくつかの実施形態において、条件付きショート前方分岐には、条件分岐命令、および条件分岐命令と当該条件分岐命令によって示された前方分岐ターゲット命令との間にある、条件分岐命令にプログラム順に連続して続く一つ以上の命令のセットを含めることができる。

【００３０】

いくつかの実施形態において、この条件付きショート前方分岐は、図１の条件付きショート前方分岐と類似または同一であり得る。あるいは、この条件付きショート前方分岐を図１のものとは異なるものとすることもできる。取り込まれる命令は、実行のためプロセッサに供される、マシンコード命令、アセンブリ言語命令、マクロ命令、または他の比較的にハイレベルな命令、または制御信号を表し得る。これらの命令は、プロセッサの命令セットアーキテクチャ（ＩＳＡ）の一部とすることが可能である。ＩＳＡは、プログラミングに関する、プロセッサのアーキテクチャの一部であって、通常、プロセッサの、ネイティブな命令、アーキテクチャレジスター、データ型、アドレスモードなどを含む。

【００３１】

再度、図２を参照すると、命令変換論理２１４は命令取り込み論理２１３と連結されている。いくつかの実施形態において、命令変換論理には、命令解読論理および／またはプロセッサのパイプラインの解読段階にある論理を含めることができる。本明細書では、この命令解読論理を、解読論理、解読ユニット、またはデコーダと称することがある。命令変換論理は、ハードウェア（例えば、集積回路、トランジスタなど）、ファームウェア（例えば、不揮発性メモリに格納された命令）、ソフトウェア、またはこれらの組み合わせを含め、様々な異なるメカニズムを用いて実装することができる。いくつかの実施形態において、命令変換論理は、何らかのファームウェアおよび／またはソフトウェアに組み合わされ得る少なくとも何らかのオンダイのハードウェア論理を含む。命令変換論理を実装するための適切なメカニズムの例としては、マイクロコード読み取り専用メモリ（ＲＯＭ）、ルックアップテーブル、ハードウェア実装、プログラム可能論理アレイ（ＰＬＡ）、および当該技術分野で既知の命令解読および変換メカニズムが挙げられるが、これらに限定されない。

【００３２】

命令変換論理は、命令取り込みユニットから、取り込まれたマシン命令、マクロ命令、または他の比較的にハイレベルな命令、または制御信号を受信することが可能である。命令変換論理は、受信した命令または制御信号を、対応する比較的に低レベルのマイクロ命令、マイクロ演算、マイクロｏｐｓ、マイクロコードエントリポイント、もしくは他のより低レベルの命令または制御信号に変換することができる。適切な変換の例としては、解読、エミュレート、モーフ、翻訳、他の方法による変換、またはこれらの組み合わせが挙げられるが、これらに限定されない。比較的に低レベルの命令または制御信号は、さらに低レベル（例えば、回路レベルまたはハードウェアレベル）の演算を介して、比較的にハイレベルの命令または制御信号の演算を実施することができる。一つの態様において、比

較的に低レベルの命令または制御信号は、ネイティブのプロセッサハードウェア（例えば、実行ユニット、回路など）上で実行可能または実装可能であり得る。

【００３３】

再度、図２を参照すると、いくつかの実施形態において、条件付きショート前方分岐２０１の命令は、命令変換論理２１４にも提供することができる。命令変換論理は、条件付きショート前方分岐の命令を、計算的におよび／または機能的に等価な、一つ以上の述語付き命令のセットに変換する命令変換論理２１５を含む。いくつかの実施形態において、これら述語付き命令は、これらが当初の命令と（例えば、算術的、論理的に）同一または等価な演算を遂行できる点で、およびこれらが当初の命令と同一または等価なオペランド（例えば、アーキテクチャレジスター、メモリ場所など）上で作動する点で、当初の命令と計算的におよび／または機能的に等価であり得る。いくつかの実施形態において、条件付きショート前方分岐中の条件分岐命令の後の各命令は、述語なし命令から、対応する機能的に等価な述語付き命令に変換することが可能である。

【００３４】

いくつかの実施形態において、命令変換論理２１５は、変換の一部として条件分岐命令を除去することができる。いくつかの実施形態では、条件分岐マイクロ命令の条件態様は、少なくとも概念的には、条件付きショート前方分岐内（例えば、条件分岐命令と当該条件分岐命令のターゲットとの間）の一つ以上のマイクロ命令の各々に組み合わせることが可能である。いくつかの実施形態において、条件分岐命令（すなわち、分岐態様）が除去され、条件態様を、条件付きショート前方分岐内の他の命令の各々の解読されたバージョン（例えば、マイクロ命令）中に組み入れる、変換を実施することができる。

【００３５】

図示のように、いくつかの実施形態では、分岐予測器による、条件分岐命令が「条件成立」であるか、または「条件不成立」であるかの予測に関係なく、および／または分岐予測器が分岐予測をする必要さえなしに、および／または命令変換論理が分岐予測を知る必要なしに、述語付き命令の機能的におよび／または計算的に等価なセットに対応するおよび／またはこれを表す命令（例えば、マイクロ命令）または他の制御信号２１６を、命令変換論理からバックエンド論理２１７に出力することができる。すなわち、通常は、分岐が条件成立であると予測された場合には、パイプライン中に挿入されない、条件付きショート前方分岐内の命令を表す制御信号は、今や、分岐予測に関係なく、および／またはかかる分岐予測を行う必要さえなく、パイプライン中に挿入することが可能である。これは、分岐予測ミスの場合に、通常必要となるパイプラインのフラッシュ、および／または実行の巻き戻しを回避する必要性の助力となり得る。いくつかの実施形態において、変換を実施するのは、ソフトウェアコンパイラでなく、むしろプロセッサのハードウェアまたは他の論理である。

【００３６】

再度、図２を参照すると、バックエンド論理２１７は、当初の条件付きショート前方分岐の条件分岐命令に関連付けられた条件の最終的に判断された結果に基づいて、一つ以上の述語付き命令の機能的におよび／または計算的に等価なセットを反映するため、アーキテクチャの状態を更新し、または更新しない論理２１８を含む。いくつかの実施形態において、条件分岐命令の条件の結果が、条件分岐が条件成立と最終的に判断される場合、論理２１８は、述語付き命令の機能的におよび／または計算的に等価なセットを反映するために、アーキテクチャの状態を更新しないでおくことができる。逆に、条件分岐命令の条件の結果が、条件分岐が条件不成立であると最終的に判断される場合、論理２１８は、述語付き命令の機能的におよび／または計算的に等価なセットを反映するために、アーキテクチャの状態を更新することができる。いくつかの実施形態において、当該論理は、選択論理を用い、述語付き命令の結果を選択するか、もしくは、述語付き命令を無視、および／またはあたかもそれらが実行されなかったかのように作動して、従来方式による結果を選択する。

【００３７】

有利には、条件付きショート前方分岐の述語付き命令へのかかる変換、およびアーキテクチャの状態を更新し、もしくは更新しないためのかかる述語付き命令の使用は、ミス予測された条件分岐のスピードおよびエネルギ上の不利益を回避することによって、プロセッサのスピードおよびエネルギ保全の改善に役立たせることができる。ショートな前方へのループ内の、条件分岐命令に後続する命令群は、パイプライン中に入力されるが、最終的に条件分岐が条件成立であるか、または不成立であるかが判定されるまでは、条件付き実行制御を使って処理される。分岐予測ミスが生じた場合のプロセッサの実行の巻き戻し、および推測で実行された関連する状態の放棄の必要はない。かかる利点は、特に、益々深まるパイプラインを有する（例えば、パイプライン中に百単位の走行中の命令を有する）特定の大型アウトオブオーダープロセッサで大きいことになろう。

【００３８】

本説明が曖昧になるのを避けるため、比較的単純なプロセッサ２１０を提示し説明している。他の実施形態において、プロセッサには、随意的に、例えば、命令先取りバッファ、命令待ち行列、命令および／またはデータキャッシュ、命令および／またはデータ翻訳索引バッファ、マイクロ命令待ち行列、リネーム／割り当てユニット、マイクロ命令シーケンサ、実行ユニット、除去／完遂ユニット、レジスターリネームユニット、バスインターフェースユニット、第二および／またはよりハイレベルの命令および／またはデータキャッシュ、プロセスに含まれる他のコンポーネント、およびこれらの様々な組み合わせなど、他の周知のコンポーネントを含めることができる。文字通り数えきれないほどの、プロセッサ内でのコンポーネントの異なる組み合わせおよび構成があり、諸実施形態は、いかなる特定の組み合わせまたは構成にも限定されない。

【００３９】

図３は、条件分岐を処理する方法３２０の実施形態のブロック流れ図である。いくつかの実施形態において、図３のオペレーションおよび／または方法は、図２のプロセッサによって、および／またはその内部で実施することができる。本明細書で図２のプロセッサについて説明したコンポーネント、特徴、および特定の随意的詳細は、図３のオペレーションおよび／または方法に対しても随意に適用され、諸実施形態において、それらは、図２のプロセッサによって、および／またはその内部で実施することが可能である。あるいは、図３のオペレーションおよび／または方法は、類似のまたは全く異なるプロセッサまたは装置によって、および／またはその内部で実施することもできる。さらに、図２のプロセッサも、図３のものと同一の、類似の、またはそれらと全く異なるオペレーションおよび／または方法を実施することが可能である。

【００４０】

本方法は、ブロック３２１で、条件付きショート前方分岐を取り込む段階を含む。いくつかの実施形態において、条件付きショート前方分岐には、条件分岐命令と、条件分岐命令と当該条件分岐命令によって示された前方分岐ターゲット命令との間にある、条件分岐命令にプログラム順に連続して続く一つ以上の命令のセットを含めることができる。いくつかの実施形態において、この条件付きショート前方分岐は、図１の条件付きショート前方分岐と類似または同一であり得る。あるいは、この条件付きショート前方分岐を図１に示されたものとは異なるものとすることもできる。

【００４１】

また、本方法は、ブロック３２２で、条件付きショート前方分岐を、一つ以上の述語付き命令の計算的に等価なセットに変換する段階を含む。いくつかの実施形態において、これには、条件分岐命令と前方分岐ターゲット命令との間の一つ以上の各命令を、述語なし命令から対応する述語付き命令に変換する段階を含めることができる。いくつかの実施形態では、これには条件分岐命令を除去する段階を含めることもできる。

【００４２】

いくつかの実施形態において、本方法は、一つ以上の述語付き命令の計算的に等価なセットに対応するおよび／またはこれを表す信号を、プロセッサのバックエンド論理に提供する段階をさらに含めることが可能である。いくつかの実施形態において、これは、条件

分岐命令が条件成立か、または不成立かどうかの予測に関係なく行うことができる。いくつかの実施形態において、これは、分岐予測論理のオペレーションに関係なく、および／または分岐予測器が条件分岐予測に対する条件予測を行うのかどうかに関係なく、および／または、分岐予測がされても、命令変換論理がそれを知っているかどうかに関係なく、行うことが可能である。

【００４３】

図４は、条件付きショート前方分岐の検出および変換論理４１５を示すブロック図である。図５は、条件付きショート前方分岐を変換するかどうかを検出し判定する方法５３５の実施形態のブロック流れ図である。説明を効率化するために、図４の論理を参照しながら、図５のオペレーションおよび方法を説明することにする。但し、図５のオペレーションおよび方法は、図４のものと類似のまたはそれらと異なる実施形態によっても実施が可能であることを理解されたい。さらに、図４の論理が、図５のオペレーションと同一の、類似の、またはそれらと異なるオペレーションおよび方法を実施することも可能である。

【００４４】

図４を参照すると、条件付きショート前方分岐検出論理４２５は、条件付きショート前方分岐４０１を検出する。図５を参照すると、本方法は、ブロック５３６で、条件付きショート前方分岐を検出する段階を含む。いくつかの実施形態において、これには、条件分岐命令を検出する段階と、分岐が前方向きでショート（例えば、約１～約１０個の命令、または約１～約５個の命令）であるかどうかを判定するために、当該条件分岐命令の前方分岐ターゲットを調べる段階と、を含めることができる。

【００４５】

再度、図４を参照すると、条件付きショート前方分岐を変換するかどうかを判定する論理４２６が、条件付きショート前方分岐検出論理４２５と連結されている。いくつかの実施形態において、この判定は、条件付きショート前方分岐内の、条件分岐命令と当該条件分岐命令の前方分岐ターゲットとの間にある一つ以上の命令の全てに対し、対応する計算的に等価な命令が利用可能であるかどうかに基づいて行うことができる。例えば、これらの命令の一つ以上に対し計算的に等価な述語付き命令が存在しない場合は、変換しないと判定することができ、もしくは、これらの命令の全てに対し計算的に等価な述語付き命令が存在する場合は、変換すると判定することができる。論理４２６が、条件付きショート前方分岐を変換しないと判定した場合、従来式の条件付きショート前方分岐処理論理４２７に、その分岐を提供することが可能である。逆に、論理４２６が条件付きショート前方分岐を変換すると判定した場合は、当該分岐を命令変換論理４２８に提供することができる。

【００４６】

図５を参照すると、本方法は、ブロック５３７で、ショート前方分岐内の全ての命令が、対応する機能的におよび／または計算的に等価な述語付き命令に変換可能であるかどうかを判定する段階を含む。ショート前方分岐内の全命令が、対応する機能的におよび／または計算的に等価な述語付き命令に変換可能ではない場合は（すなわち、ブロック５３７の判定の「いいえ」）、ブロック５３８で、条件分岐命令の従来式の処理を行うことができる。また、ショート前方分岐内の全命令が、対応する機能的におよび／または計算的に等価な述語付き命令に変換可能な場合は（すなわち、ブロック５３７の判定の「はい」）、本方法はブロック５３９に進むことができる。

【００４７】

再度、図４を参照すると、命令変換論理４２８は、論理４２６と連結されている。命令変換論理は、条件付きショート前方分岐の命令を変換する。命令変換論理は、述語なし命令を計算的に等価な述語付き命令にマッピングおよび変換する論理４２９を含む。マッピングおよび変換論理は、各述語なし命令を機能的に等価な述語付き命令にマッピングし変換する。例えば、ある所与のソースオペランドのペアに可算演算を行い、和をある宛先に格納することを指定する加算命令は、同じ所与のソースオペランドのペアに可算演算を行い、和を同じ宛先に格納することを指定する（だが条件付きで）述語付き加算命令にマッ

ピングし変換することができる。また、別の実施形態において、一対一、の命令と述語付き命令との対応よりも、むしろ、一対複数、または複数対一、の命令と述語付き命令との対応を用いることが可能である。一つ以上の計算的および／または機能的に等価な述語付き命令４３０は、例えば、さらなる解読論理または他の命令変換論理に、および／またはバックエンド論理に向けて出力される。なお、条件分岐命令４０３は、この変換の過程で除去４３１される。図５を参照すると、本方法は、ブロック５３９で、条件付きショート前方分岐を、一つ以上の述語付き命令の機能的におよび／または計算的に等価なセットに変換する段階を含む。

【００４８】

図６は、条件付きショート前方分岐６０１の、一つ以上の述語付き命令の計算的に等価なセット６３０への変換の例示的な実施形態を示す。条件付きショート前方分岐６０１は、ターゲットオフセットを示し、条件が一致すればジャンプする（ｊｃｃ）命令６０３を含む。このｊｃｃ命令は、「条件成立パス」および「条件不成立パス」を有する。条件成立パスは、ｊｃｃ命令のターゲットオフセットに位置する前方分岐ターゲット命令６０６に進む。条件不成立パスは、ｍｏｖ命令６０４に進み、これは汎用レジスターｒｃｘを汎用レジスターｒｂｘに移動する命令である。これは単なる一例である。ｍｏｖ命令の後に、随意に、一つ以上のロード、格納、算術、または論理命令、あるいはこれらの組み合わせ６０５を置くことができる。

【００４９】

ｊｃｃ命令は、一つ以上の述語付き命令の計算的に等価なセット６３０から除去される６３１。また、条件成立および不成立パスも効果的に除去される。分岐予測の結果の如何に関係なく、あるいは分岐予測が行われたかどうかの如何にさえ関係なく、このショート前方分岐内には命令の飛び越しはない。一つ以上の述語付き命令の計算的に等価なセットは、ｍｏｖ命令６４０に対応するｃｍｏｖｚ命令を含み、これは、条件付きでまたは条件付き実行制御を使って汎用レジスターｒｃｘを汎用レジスターｒｂｘに移動するための命令である。ｃｍｏｖｚ命令は、Ｉｎｔｅｌアーキテクチャ中にある既存の条件命令であり、条件の状態に関係なく例外処理を開始する。別の選択肢として、いくつかの実施形態では、ｃｍｏｖｚ命令を、条件付き移動命令の条件が偽の場合は例外処理を開始しない類似の命令、と置き換えることが可能である。このように、条件付き命令が実行されたことが想定されていない場合、当該命令は完全に無視することができ、一切の潜在的例外または同類のものを開始しないことを含め、何の影響ももたらさない。また、一つ以上の、計算的に等価な述語付きのロード、格納、算術、または論理命令、あるいはこれらの組み合わせ６４１が含まれている。これらの命令は計算的に命令６０５と等価である。述語付き命令のセットが用いられると決まったならば、これらは、対応する当初の述語なしの命令のセットと正確に同じ結果を出す必要がある。いくつかの実施形態において、前方分岐ターゲット６０６は変換する必要がないことに留意する。

【００５０】

図７は、条件付きショート前方分岐を表しこれと計算的に等価な、一つ以上の述語付き命令のセット７１６を反映するために、アーキテクチャの状態を更新し、または更新しないバックエンド論理７１７の実施形態のブロック図である。いくつかの実施形態において、このバックエンド論理を図２のプロセッサとともに用いることができる。これに換えて、このバックエンド論理を、図２のものと類似なまたは異なるプロセッサとともに用いることもできる。さらに、図２のプロセッサを、同一の、類似の、もしくは異なるバックエンド論理とともに用いることも可能である。

【００５１】

インオーダー論理の例示的な実施形態と、レジスターリネームおよびアウトオブオーダースケジューリング／実行／除去論理の例示的な実施形態との両方を示す。実線のボックスは、インオーダー論理を表し、随意的な追加の破線のボックスは、レジスターリネームおよびアウトオブオーダースケジューリング／実行／除去論理を表す。インオーダー態様がアウトオブオーダー態様のサブセットであることを所与として、アウトオブオーダーの

説明をするが、但し、本発明はこれに限定されるものではない。

【００５２】

一つ以上の述語付き命令７１６が、バックエンド論理７１７に提供される。本明細書の他の箇所で説明しているように、これら一つ以上の述語付き命令は、条件付きショート前方分岐を表し、これと計算的に等価である。リネーム／割り当て論理７５０が、述語付き命令のセットを受信する。このリネーム／割り当て論理は、除去および／または完遂論理７５５、および一つ以上のスケジューラ論理７５１に連結されている。スケジューラ論理は、リザベーションステーション、集中型命令ウィンドウなどを含め、任意の数の相異なるスケジューラを表す。当該スケジューラ論理は、物理レジスターファイルユニット７５２に連結されている。この物理レジスターファイルユニットの各々は、一つ以上の物理レジスターファイルを表し、その相異なるファイルは、例えば、スカラ整数、スカラ浮動小数点、パック整数、パック浮動小数点、ベクトル整数、ベクトル浮動小数点、ステータス（例えば、実行対象の次の命令アドレスである命令ポインタ）など、一つ以上の相異なるデータ型を格納する。一つの実施形態において、物理レジスターファイルユニットは、ベクトルレジスターユニット、書き込みマスクレジスターユニット、およびスカラレジスターユニットを含む。これらのレジスターユニットには、アーキテクチャベクトルレジスター、ベクトルマスクレジスター、および汎用レジスターを設けることができる。いくつかの実施形態において、物理レジスターファイルユニットには、フラグレジスター７５３および予測レジスター７５４を含めることが可能で、これらの一方または双方を、述語付き命令による条件付き実行制御のために使用することができる。

【００５３】

物理レジスターファイルユニットは、除去／完遂論理７５５によってオーバーラップされ、レジスターリネームおよびアウトオブオーダー実行の実装を可能にする様々なやり方（例えば、リオーダバッファおよび除去レジスターファイルを使う、将来ファイル、履歴バッファ、および除去レジスターファイルを使う、レジスターマップおよびレジスターのプールを使うなど）を示す。除去／完遂論理および物理レジスターファイルユニットは、実行論理７５６に連結される。この実行論理は、一つ以上の実行ユニットのセット７５７、および一つ以上のメモリアクセスユニットのセット７５８を含む。この実行論理は、本明細書が開示する述語付き命令を実行することができる。当該実行ユニットは、様々な演算（例えば、シフト、可算、減算、乗算）を、様々な型のデータ（例えば、スカラ浮動小数点、パック整数、パック不動小数点、ベクトル整数、ベクトル浮動小数点）に対して実施することができる。いくつかの実施形態には、特定の機能または機能のセットに専用のいくつかの実行ユニットを含めることができ、一方、他の実施形態には、全機能を実行する一つだけの実行ユニットまたは全部が全機能を実行する複数の実行ユニットを含めることが可能である。

【００５４】

スケジューラ論理、物理レジスターファイルユニット、および実行論理は、複数もあり得るとして示し／説明している。というのは、特定の実施形態は、特定の種類のデータ／オペレーションに対して、別個のパイプラインを生成するからである（例えば、各々がそれ自体のスケジューラユニットを持つ、スカラ整数パイプライン、スカラ浮動小数点／パック整数／パック浮動小数点／ベクトル整数／ベクトル浮動小数点パイプライン、および／またはメモリアクセスパイプライン、物理レジスターファイルユニット、および／または実行クラスタなどである。さらに、メモリアクセスパイプラインが別個の場合、特定の実施形態は、このパイプラインの実行クラスタだけがメモリアクセスユニットを有するように実装される）。また、別個のパイプラインが使われている場合、これらパイプラインの一つ以上をアウトオブオーダーの発行／実行とし、残りをインオーダーとすることができることを理解されたい。

【００５５】

例示的なコアアーキテクチャ、プロセッサ、およびコンピュータアーキテクチャ

プロセッサコアは、相異なるやり方で、相異なる目的のため、相異なるプロセッサに実

装することができる。例えば、かかるコアの実装には、１）汎用コンピューティング向けの汎用インオーダーコアと、２）汎用コンピューティング向けの高性能の汎用アウトオブオーダーのコアと、３）主としてグラフィックスおよび／または科学技術（スループット）コンピューティング向けの特定用途向けコアと、を含めることが可能である。相異なるプロセッサの実装には、１）汎用コンピューティング向けの一つ以上の汎用インオーダーコア、および／または汎用コンピューティング向けの一つ以上の汎用アウトオブオーダーコアを含むＣＰＵと、２）主としてグラフィックスおよび／または科学技術（スループット）コンピューティング向けの一つ以上の特定用途向けコアと、を含むコプロセッサを含めることが可能である。かかる相異なるプロセッサは、相異なったコンピュータシステムアーキテクチャをもたらし、それらは、１）ＣＰＵから分離されたチップ上のコプロセッサ、２）ＣＰＵとして同一のパッケージ中の別個のダイ上のコプロセッサ、３）ＣＰＵとして同一のダイ上のコプロセッサ（この場合、かかるコプロセッサは、一体型グラフィックスおよびまたは科学技術（スループット）論理、または特定用途向けコアなど、特定用途向け論理といわれることもある）、ならびに４）同じダイの上に、前述のＣＰＵ（アプリケーションコアまたはアプリケーションプロセッサといわれることもある）、前述のコプロセッサ、および追加の機能を含むことが可能なチップ上のシステムを含み得る。次に、例示的なコアアーキテクチャを説明し、その後に例示的なプロセッサおよびコンピュータアーキテクチャを説明する。

【００５６】

例示的なコアアーキテクチャ

インオーダーおよびアウトオブオーダーのコアのブロック図　図８Ａは、本発明の実施形態による、例示的なインオーダーパイプライン、ならびに例示的なレジスターリネームおよびアウトオブオーダー発行／実行パイプラインの両方を示したブロック図である。図８Ｂは、本発明の実施形態による、例示的実施形態のインオーダーアーキテクチャコアと、プロセッサに含めるための例示的なアウトオブオーダーのレジスターリネーム発行／実行のアーキテクチャコアと、の両方を示すブロック図である。図８Ａ～８Ｂ中の実線のボックスは、インオーダーパイプラインおよびインオーダーコアを示し、一方、随意的追加の破線ボックスは、レジスターリネームおよびアウトオブオーダー発行／実行のパイプラインおよびコアを示す。インオーダー態様がアウトオブオーダー態様のサブセットであることを所与として、アウトオブオーダーの態様を説明する。

【００５７】

図８Ａにおいて、プロセッサパイプライン８００は、取り込み段階８０２、長さ解読段階８０４、解読段階８０６、割り当て段階８０８、リネーム段階８１０、スケジューリング（タスク指名または発行としても知られる）段階８１２、レジスター読み取り／メモリ読み取り段階８１４、実行段階８１６、書き戻し／メモリ書き込み段階８１８、例外処理段階８２２、および完遂段階８２４を含む。

【００５８】

図８Ｂは、実行エンジンユニット８５０に連結されたフロントエンドユニット８３０を含むプロセッサ８９０を示し、これらの両ユニットはメモリユニット８７０に連結されている。コア８９０は、縮小命令セットコンピューティング（ＲＩＳＣ）コア、複合命令セットコンピューティング（ＣＩＳＣ）コア、超長命令語（ＶＬＩＷ）コア、あるいはハイブリッドまたは別の型のコアとすることができる。さらなる別の選択肢として、コア８９０は、例えば、ネットワークまたは通信コア、圧縮エンジン、コプロセッサコア、汎用コンピューティンググラフィックス処理ユニット（ＧＰＧＰＵ）コア、グラフィックスコアなどの特定用途向けコアとすることが可能である。

【００５９】

フロントエンドユニット８３０は、分岐予測ユニット８３２を含み、このユニットは命令キャッシュユニット８３４に連結され、当該ユニットは命令の翻訳索引バッファ（ＴＬＢ）８３６に連結され、当該ユニットは命令取り込みユニット８３８に連結され、当該ユニットは解読ユニット８４０に連結されている。解読ユニット８４０（またはデコーダ）

は、命令を解読し、出力として、一つ以上のマイクロ演算、マイクロコードエントリポイント、マイクロ命令、他の命令、または他の制御信号を生成することができ、これらは、当初の命令から、解読され、または別途にこれらを反映し、またはこれらから導出されたものである。解読ユニット８４０は、様々の相異なるメカニズムを使って実装することができる。適切なメカニズムの例としては、ルックアップテーブル、ハードウェア実装、プログラム可能論理アレイ（ＰＬＡ）、マイクロコード読み取り専用メモリ（ＲＯＭ）などが挙げられるが、これらに限定されない。一つの実施形態において、コア８９０は、マイクロコードＲＯＭ、または特定のマイクロ命令に対するマイクロコードを格納する他の媒体を（例えば、解読ユニット８４０中に、または別途にフロントエンドユニット８３０内に）含む。解読ユニット８４０は、実行エンジンユニット８５０中のリネーム／アロケーターユニット８５２に連結されている。

【００６０】

実行エンジンユニット８５０は、除去ユニット８５４および一つ以上のスケジューラユニット８５６に連結されたリネーム／アロケーターユニット８５２を含む。スケジューラユニット８５６は、リザベーションステーション、集中型命令ウィンドウなどを含め、任意の数の相異なるスケジューラを表す。スケジューラ論理８５６は、物理レジスターファイルユニット８５８に連結されている。この物理レジスターファイルユニット８５８の各々は、一つ以上の物理レジスターファイルを表し、それらの各異なるファイルは、例えば、スカラ整数、スカラ浮動小数点、パック整数、パック浮動小数点、ベクトル整数、ベクトル浮動小数点、ステータス（例えば、実行対象の次の命令アドレスである命令ポインタ）など、一つ以上の相異なるデータ型を格納する。一つの実施形態において、物理レジスターファイルユニット８５８は、ベクトルレジスターユニット、書き込みマスクレジスターユニット、およびスカラレジスターユニットを含む。これらのレジスターユニットには、アーキテクチャベクトルレジスター、ベクトルマスクレジスター、および汎用レジスターを設けることができる。物理レジスターファイルユニット８５８は、除去論理８５４によってオーバーラップされ、レジスターリネームおよびアウトオブオーダー実行の実装を可能にする様々なやり方（例えば、リオーダバッファおよび除去レジスターファイルを使う、将来ファイル、履歴バッファ、および除去レジスターファイルを使う、レジスターマップおよびレジスターのプールを使うなど）を示す。除去ユニット８５４および物理レジスターファイルユニット８５８は、実行クラスタ８６０に連結されている。この実行クラスタ８６０は、一つ以上の実行ユニットのセット８６２、および一つ以上のメモリアクセスユニットのセット８６４を含む。実行ユニット８６２は、様々な演算（例えば、シフト、可算、減算、乗算）を、様々な型のデータ（例えば、スカラ浮動小数点、パック整数、パック不動小数点、ベクトル整数、ベクトル浮動小数点）に対して実施することができる。いくつかの実施形態には、特定の関数または関数のセットに専用のいくつかの実行ユニットを含めることができ、一方、他の実施形態には、全関数を実行する一つだけの実行ユニットまたは全部が全関数を実行する複数の実行ユニットを含めることが可能である。スケジューラユニット８５６、物理レジスターファイルユニット８５８、および実行クラスタ８６０は複数もあり得るとして示している。というのは、特定の実施形態は、特定の種類のデータ／オペレーションに対して、別個のパイプラインを生成するからである（例えば、各々がそれ自体のスケジューラユニットを持つことが可能な、スカラ整数パイプライン、スカラ浮動小数点／パック整数／パック浮動小数点／ベクトル整数／ベクトル浮動小数点パイプライン、および／またはメモリアクセスパイプライン、物理レジスターファイルユニット、および／または実行クラスタなどである。さらに、メモリアクセスパイプラインが別個の場合、特定の実施形態は、このパイプラインの実行クラスタだけがメモリアクセスユニット８６４を有するように実装される）。また、別個のパイプラインが使われている場合、これらパイプラインの一つ以上をアウトオブオーダーの発行／実行とし、残りをインオーダーとすることを理解されたい。

【００６１】

メモリアクセスユニットのセット８６４は、メモリユニット８７０に連結されており、

当該メモリユニットは、データＴＬＢユニット８７２を含み、これは、レベル２（Ｌ２）キャッシュユニット８７６に連結されたデータキャッシュユニット８７４に連結されている。一つの例示的実施形態において、メモリアクセスユニット８６４には、ロードユニット、アドレス格納ユニット、およびデータ格納ユニットを含めることができ、これらの各々は、メモリユニット８７０中のデータＴＬＢユニット８７２に連結されている。命令キャッシュユニット８３４は、メモリユニット８７０中のレベル２（Ｌ２）キャッシュユニット８７６にさらに連結されている。Ｌ２キャッシュユニット８７６は、一つ以上の他のレベルのキャッシュに、そして最終的には主メモリに連結される。

【００６２】

例として、例示的なレジスターリネームおよびアウトオブオーダー発行／実行コアアーキテクチャは、１）命令取り込み８３８が、取り込みおよび長さ解読段階８０２および８０４を遂行し、２）解読ユニット８４０が、解読段階８０６を遂行し、３）リネーム／アロケーターユニット８５２が、割り当て段階８０８およびリネーム段階８１０を遂行し、４）スケジューラユニット８５６が、スケジュール段階８１２を遂行し、５）物理レジスターファイルユニット８５８およびメモリユニット８７０が、レジスター読み取り／メモリ読み取り段階８１４を遂行し、実行クラスタ８６０が、実行段階８１６を遂行し、６）メモリユニット８７０および物理レジスターファイルユニット８５８が、書き戻し／メモリ書き込み段階８１８を遂行し、７）各種のユニットが、例外処理段階８２２に関与することができ、且つ８）除去ユニット８５４および物理レジスターファイルユニット８５８が、完遂段階８２４を遂行することによって、パイプライン８００を実施することができる。

【００６３】

コア８９０は、本明細書で説明した命令を含め、一つ以上の命令のセット（例えば、（最新のバージョンに加えられたいくつかの拡張を備えた）ｘ８６命令セット、カリフォルニア州、サニーベールのＭＩＰＳ　Ｔｅｃｈｎｏｌｏｇｉｅｓの ＭＩＰＳ命令セット、（カリフォルニア州、サニーベールのＡＲＭ　Ｈｏｌｄｉｎｇｓの（ＮＥＯＮなど随意的な追加拡張を備えた）ＡＲＭ命令セット）をサポートすることが可能である。一つの実施形態において、コア８９０は、パックデータ命令セット拡張機能（例えば、ＡＶＸ１、ＡＶＸ２）をサポートするための論理を含み、これにより、パックデータを使って実行される多くのマルチメディアアプリケーションが用いる演算が可能である。

【００６４】

このコアは、マルチスレッド処理（演算またはスレッドの２つ以上の並行セットの実行）をサポートすることが可能で、タイムスライス型マルチスレッド処理、同時マルチスレッド処理（単一の物理コアが、その物理コアが同時にマルチスレッド処理をしているスレッド群の各々に対し、論理コアを提供する）またはこれらの組み合わせ（例えば、Ｉｎｔｅｌのハイパースレッディング技術などにおける、時間スライス型取り込みおよび解読、およびその後の同時マルチスレッド処理）を含め、様々なやり方で処理を行うことができることを理解されたい。

【００６５】

レジスターリネーム処理をアウトオブオーダーの実行と関連させて説明しているが、レジスターリネーム処理はインオーダーアーキテクチャにおいて用いることも可能であることを理解されたい。また、プロセッサの例示的な実施形態には、別個の命令キャッシュ８３４、データキャッシュユニット８７４、および共用Ｌ２キャッシュユニット８７６が含まれているが、別の実施形態は、例えば、レベル１（Ｌ１）内部キャッシュ、または多重レベルの内部キャッシュなど、命令およびデータの両方に対し単一の内部キャッシュを有することも可能である。いくつかの実施形態において、本システムには、内部キャッシュと、コアおよび／またはプロセッサの外部にある外部キャッシュとの組合せを含めることができる。あるいは、キャッシュの全ては、コアおよび／またはプロセッサの外部にあってもよい。

【００６６】

具体的な例示のインオーダーコアアーキテクチャ

図９Ａ～９Ｂは、さらに具体的な例示インオーダーコアアーキテクチャのブロック図を示し、このコアは、チップ中のいくつかの論理ブロック（同じ型および／または異なる型の他のコアを含む）の一つであり得よう。当該論理ブロックは、高帯域幅相互接続ネットワーク（例えば、リングネットワーク）を介し、アプリケーションに応じて、いくつかの固定機能論理、メモリＩ／Ｏインターフェース、および他の必要なＩ／Ｏ論理と通信する。

【００６７】

図９Ａは、本発明の実施形態による、単一のプロセッサコアを、当該コアのオンダイ相互接続ネットワーク９０２への接続、およびそのレベル２（Ｌ２）キャッシュ９０４のローカルサブセットと共に示した、ブロック図である。一つの実施形態において、命令解読デコーダ９００は、パックデータ命令セット拡張機能を備えたｘ８６命令セットをサポートする。Ｌ１キャッシュ９０６は、キャッシュメモリのスカラユニットおよびベクトルユニット中への低遅延のアクセスを可能にする。（設計を単純化するための）一つの実施形態では、スカラユニット９０８およびベクトルユニット９１０は、別々のレジスターセット（それぞれ、スカラレジスター９１２およびベクトルレジスター９１４）を使用し、これらの間で伝送されるデータは、メモリに書き込まれ、その後レベル１（Ｌ１）キャッシュ９０６から読み戻されるが、本発明の別の実施形態は、異なるアプローチを用いることが可能である（例えば、単一のレジスターセットを使う、または、書き込みおよび読み戻しなしに、２つのレジスターファイルの間でデータの伝送を可能にする通信パスを含める）。

【００６８】

Ｌ２キャッシュのローカルサブセット９０４は、プロセッサコア当たり一つの別々のローカルサブセットに分割された、大域キャッシュＬ２の部分である。各プロセッサコアは、Ｌ２キャッシュ９０４の、自己専用のローカルサブセットへの直接アクセスパスを有する。プロセッサコアに読み取られたデータは、Ｌ２キャッシュのサブセット９０４中に格納され、これは、それぞれ自己専用のローカルＬ２キャッシュサブセットにアクセスしている他のプロセッサコアと並行して、迅速にアクセスすることが可能である。プロセッサコアによって書き込まれたデータは、そのコア専用のＬ２キャッシュサブセット９０４に格納され、必要に応じて他のサブセットからフラッシュされる。リングネットワークが、共用データに対する一貫性を確実にする。リングネットワークは双方向性で、プロセッサコア、Ｌ２キャッシュ、および他の論理ブロックなどのエージェントが、チップ内で相互に通信することを可能にする。各リングデータパスは、方向当たり１０１２ビット幅である。

【００６９】

図９Ｂは、本発明の実施形態による、図９Ａ中のプロセッサコアの部分の拡大図である。図９Ｂは、Ｌ１データキャッシュ９０４のＬ１データキャッシュ９０６Ａ部分、およびベクトルユニット９１０およびベクトルレジスター９１４に関するさらなる詳細を含む。具体的には、ベクトルユニット９１０は１６ビット幅のベクトル処理ユニット（ＶＰＵ：ｖｅｃｔｏｒ　ｐｒｏｃｅｓｓｉｎｇ　ｕｎｉｔ）であり（１６ビット幅のＡＬＵ９２８を参照）、これは、一つ以上の整数、単精度浮動、および倍精度浮動命令を実行する。ＶＰＵは、メモリ入力上で、スウィズルユニット９２０を使用するレジスター入力のスウィズル（順序調整）と、数値化ユニット９２２Ａ～Ｂを使用する数値変換と、レプリケーションユニット９２４を使用するレプリケーションとをサポートする。書き込みマスクレジスター９２６は、得られたベクトル書き込みの述語付けを可能にする。

【００７０】

一体型メモリコントローラおよびグラフィックスを備えるプロセッサ

図１０は、本発明の実施形態による、複数のコアを有することが可能であり、一体型メモリコントローラを有することが可能で、且つ一体型グラフィックスを有することが可能なプロセッサ１０００のブロック図である。図１０中の実線のボックスは、単一コア１０

０２Ａ、システムエージェント１０１０、一つ以上のバスコントローラユニットのセット１０１６を備えたプロセッサ１０００を表し、一方、随意的な追加の破線ボックスは、複数のコア１００２Ａ～Ｎと、システムエージェントユニット１０１０中の一つ以上の一体型メモリコントローラユニットのセット１０１４と、および特定用途向け論理１００８と、を備える別のプロセッサ１０００を表す。

【００７１】

しかして、プロセッサ１０００の異なった実装には、１）一体型グラフィックスおよび／または科学技術（スループット）論理（これには一つ以上のコアを含めることが可能）である特定用途向け論理１００８と、一つ以上の汎用コア（例えば、汎用インオーダーコア、汎用アウトオブオーダーコア、これら２つの組合せ）であるコア１００２Ａ～Ｎとを備えたＣＰＵ、２）主としてグラフィックスおよび／または科学技術（スループット）向けの、多数の特定用途向けコアであるコア１００２Ａ～Ｎを備えたコプロセッサ、および３）多数の汎用インオーダーコアであるコア１００２Ａ～Ｎを備えたコプロセッサを含めることができる。しかして、プロセッサ１０００は、汎用プロセッサ、コプロセッサ、あるいは、例えば、ネットワークまたは通信プロセッサ、圧縮エンジン、グラフィックスプロセッサ、ＧＰＧＰＵ（汎用グラフィックス処理ユニット）、ハイスループットメニーインテグレーテドコア（ＭＩＣ）コプロセッサ（３０以上のコアを含む）、内蔵プロセッサなどの特定用途向けプロセッサ、とすることが可能である。このプロセッサは、一つ以上のチップに実装することができる。プロセッサ１０００は、例えば、ＢｉＣＭＯＳ、ＣＭＯＳ、またはＮＭＯＳなどいくつかの処理技術の任意のものを使って、一つ以上の基板上に実装することが可能である。

【００７２】

メモリ階層には、コア内の一レベル以上のレベルのキャッシュ、一つ以上の共用キャッシュユニットのセット１００６、一体型メモリコントローラユニット１０１４のセットに連結された外部メモリ（図示せず）を含めることができる。共用キャッシュユニットのセット１００６には、レベル２（Ｌ２）、レベル３（Ｌ３）、レベル４（Ｌ４）、または他のレベルなど、一つ以上の中間レベルのキャッシュ、ラストレベルキャッシュ（ＬＬＣ）、および／またはこれらの組み合わせを含めることが可能である。一つの実施形態では、リングベースの相互接続ユニット１０１２が、一体型グラフィックス論理１００８、共用キャッシュユニットのセット１００６、およびシステムエージェントユニット１０１０／一体型メモリコントローラユニット１０１４を相互接続しているが、別の実施形態は、かかるユニットを相互接続するために任意の数の周知技術を用いることができる。一つの実施形態において、一つ以上のキャッシュユニット１００６と、コア１００２Ａ～Ｎとの間には一貫性が維持されている。

【００７３】

いくつかの実施形態において、コア１００２Ａ～Ｎの一つ以上はマルチスレッド処理機能を有する。システムエージェント１０１０は、コア１００２Ａ～Ｎを調整し作動するコンポーネントを含む。システムエージェント１０１０には、例えば、パワー制御ユニット（ＰＣＵ）およびディスプレイユニットを含めることができる。このＰＣＵは、コア１００２Ａ～Ｎのパワー状態および一体型グラフィックス論理１００８を調節するために必要な論理およびコンポーネントとすること、あるいはそれらを含むこと、が可能である。ディスプレイユニットは、外部に接続された一つ以上のディスプレイを駆動するためのものである。

【００７４】

コア１００２Ａ～Ｎは、アーキテクチャ命令セットに関して同種構成でも異種構成でも良い。すなわち、コア１００２Ａ～Ｎの２つ以上に、同一の命令セットを実行する能力を持たせ、他方、他のコアには、当該命令セットのサブセット、あるいは相異なる命令セットだけを実行する能力を持たせるなどのことができよう。

【００７５】

例示的なコンピュータアーキテクチャ　図１１～１４は、例示的なコンピュータアーキ

テクチャのブロック図である。ラップトップ、デスクトップ、ハンドヘルドＰＣ、携帯個人端末、エンジニアリングワークステーション、サーバ、ネットワークデバイス、ネットワークハブ、スイッチ、内蔵プロセッサ、デジタル信号プロセッサ（ＤＳＰ）、グラフィックスデバイス、ビデオゲームデバイス、セットトップボックス、マイクロコントローラ、携帯電話、携帯メディアプレイヤ、ハンドヘルドデバイス、および様々な他の電子デバイスに対する、当該技術分野で既知の他のシステム設計または構成も本アーキテクチャに好適である。一般に、本明細書で開示するプロセッサおよび／または他の実行論理の組み込みが可能な、多様なシステムおよび電子デバイスは、概ね好適である。

【００７６】

次いで、図１１を参照すると、本発明の実施形態によるシステム１１００のブロック図が示されている。システム１１００には、一つ以上のプロセッサ１１１０、１１１５を含めることができ、これらはコントローラハブ１１２０に連結されている。一つの実施形態において、コントローラハブ１１２０は、グラフィックスメモリコントローラハブ（ＧＭＣＨ）１１９０、および入力／出力ハブ（ＩＯＨ）１１５０（これは別のチップ上に置くことが可能）を含み、ＧＭＣＨ１１９０は、メモリおよびグラフィックスコントローラを含み、当該コントローラには、メモリ１１４０およびコプロセッサ１１４５が連結されている。ＩＯＨ１１５０は、入力／出力（Ｉ／Ｏ）デバイス１１６０をＧＭＣＨ１１９０に連結している。あるいは、メモリおよびグラフィックスコントローラの一つまたは両方は、（本明細書で前述したように）プロセッサ内に組み込まれており、メモリ１１４０およびコプロセッサ１１４５は、プロセッサ１１１０と、ＩＯＨ１１５０を備えた単一チップ中にあるコントローラハブ１１２０とに直接連結されている。

【００７７】

プロセッサ１１１５を随意に追加できることが、図１１中に切れ線によって示されている。各プロセッサ１１１０、１１１５には、本明細書で説明した処理コアの一つ以上を含めることができ、プロセッサ１０００の特定のバージョンとすることが可能である。

【００７８】

メモリ１１４０は、例えば、ダイナミックランダムアクセスメモリ（ＤＲＡＭ）、相変化メモリ（ＰＣＭ）、またはこれら２つの組み合わせとすることができる。少なくとも一つの実施形態に対し、コントローラハブ１１２０は、フロントサイドバス（ＦＳＢ）、ＱｕｉｃｋＰａｔｈインターコネクト（ＱＰＩ）のようなポイントツーポイントインターフェース、または類似の接続１１９５などのマルチドロップバスを介して、プロセッサ１１１０，１１１５と通信する。

【００７９】

一つの実施形態において、コプロセッサ１１４５は、例えば、ハイスループットＭＩＣプロセッサ、ネットワークまたは通信プロセッサ、圧縮エンジン、グラフィックスプロセッサ、ＧＰＧＰＵ、内蔵プロセッサなどの特定用途向けプロセッサである。一つの実施形態において、コントローラハブ１１２０には、一体型グラフィックスアクセラレータを含めることができる。

【００８０】

アーキテクチャ、マイクロアーキテクチャ、温度、電力消費の特性などを含め、長短所指標のスペクトルに関し、物理リソース１１１０、１１１５の間には多様な相違があり得る。

【００８１】

一つの実施形態において、プロセッサ１１１０は、一般型のデータ処理オペレーションを制御する命令を実行する。コプロセッサ命令を命令内に内蔵させることができる。プロセッサ１１１０は、これらのコプロセッサ命令が、所属のコプロセッサ１１４５によって実行されるべき型であることを認識する。これに応じて、プロセッサ１１１０は、これらのコプロセッサ命令（またはコプロセッサ命令を表現する制御信号）をコプロセッサ１１４５へのコプロセッサバスまたは他の相互接続上に発行する。コプロセッサ１１４５は、受信したコプロセッサ命令を引き受け実行する。

【００８２】
次いで図１２を参照すると、本発明の実施形態による、さらに具体的な第一例示システム１２００のブロック図が示されている。図１２に示されるように、マルチプロセッサシステム１２００は、ポイントツーポイント相互接続システムであり、第一プロセッサ１２７０と、ポイントツーポイント相互接続１２５０を介して連結された第二プロセッサ１２８０とを含む。プロセッサ１２７０および１２８０の各々は、プロセッサ１０００の特定のバージョンとすることができる。本発明の一つの実施形態において、プロセッサ１２７０および１２８０は、それぞれプロセッサ１１１０および１１１５であり、コプロセッサ１２３８はコプロセッサ１１４５である。別の実施形態では、プロセッサ１２７０および１２８０は、それぞれプロセッサ１１１０およびコプロセッサ１１４５である。

【００８３】
プロセッサ１２７０および１２８０は、それぞれ一体型メモリコントローラ（ＩＭＣ）ユニット１２７２および１２８２を含んで示されている。また、プロセッサ１２７０は、そのバスコントローラユニットの部分として、ポイントツーポイント（Ｐ－Ｐ）インターフェース１２７６および１２７８を含み、同様に、第二プロセッサ１２８０は、Ｐ－Ｐインターフェース１２８６および１２８８を含む。プロセッサ１２７０、１２８０は、ポイントツーポイント（Ｐ－Ｐ）インターフェース回路１２７８、１２８８を使い、Ｐ－Ｐインターフェース１２５０を介して情報を交換することができる。図１２に示されるように、ＩＭＣ１２７２および１２８２は、プロセッサを、それぞれのメモリ、すなわちメモリ１２３２およびメモリ１２３４に連結しており、これらメモリは、それぞれのプロセッサにローカルに配属された主メモリの部分である。

【００８４】
プロセッサ１２７０、１２８０の各々は、ポイントツーポイントインターフェース回路１２７６、１２９４、１２８６、１２９８を使い、個別のＰ‐Ｐインターフェース１２５２、１２５４を介してチップセット１２９０と情報を交換することができる。チップセット１２９０は、随意的に、高性能インターフェース１２３９を介してコプロセッサ１２３８と情報を交換することができる。一つの実施形態において、コプロセッサ１２３８は、例えば、ハイスループットＭＩＣプロセッサ、ネットワークまたは通信プロセッサ、圧縮エンジン、グラフィックスプロセッサ、ＧＰＧＰＵ、内蔵プロセッサなどの特定用途向けプロセッサである。

【００８５】
共用キャッシュ（図示せず）は、いずれかのプロセッサ内にまたは両方のプロセッサの外部に、Ｐ－Ｐ相互接続を介してプロセッサに接続して含めることができ、プロセッサが低電力モードにされた場合、どちらかのまたは両方のプロセッサのローカルキャッシュ情報をその共用キャッシュに格納するようにすることができる。

【００８６】
チップセット１２９０は、インターフェース１２９６を介して第一バス１２１６に連結することができる。一つの実施形態において、第一バス１２１６は、周辺構成要素相互接続（ＰＣＩ）バス、あるいはＰＣＩエキスプレスバスまたは別の第三世代Ｉ／Ｏ相互接続バスなどのバスとすることが可能であるが、但し、本発明の範囲はこれらに限定されない。

【００８７】
図１２に示されるように、様々なＩ／Ｏデバイス１２１４を、第一バス１２１６を第二バス１２２０に連結しているバスブリッジ１２１８とともに、第一バス１２１６に連結することができる。一つの実施形態において、コプロセッサ、ハイスループットＭＩＣプロセッサ、ＧＰＧＰＵ、アクセラレータ（例えば、グラフィックスアクセラレータまたはデジタル信号処理（ＤＳＰ）ユニットなど）、フィールドプログラム可能ゲートアレイ、または任意の他のプロセッサなど、一つ以上の追加のプロセッサ１２１５が第一バス１２１６に連結される。一つの実施形態において、第二バス１２２０は、ローピンカウント（ＬＰＣ）バスとすることができる。一つの実施形態において、例えば、キーボードおよび／

またはマウス１２２２、通信デバイス１２２７、ならびにディスクドライブまたは命令／コードおよびデータ１２３０を包含することが可能な他の大容量ストレージデバイス等のストレージユニット１２２８を含め、様々なデバイスを第二バス１２２０に連結することが可能である。さらに、オーディオＩ／Ｏ１２２４を第二バス１２２０に連結することもできる。さらに、他のアーキテクチャも可能である。例えば、図１２のポイントツーポイントアーキテクチャの代わりに、システムにマルチドロップバスまたは他の同様なアーキテクチャを実装することができる。

【００８８】

次いで図１３を参照すると、本発明の実施形態による、さらに具体的な第二例示システム１３００のブロック図が示されている。図１３の他の態様が曖昧になるのを避けるために、図１２および１３中の同じ参照番号を付された同じ要素、および図１２の特定の態様は、図１３から省かれている。

【００８９】

図１３は、プロセッサ１２７０、１２８０に、一体型メモリ、およびそれぞれにＩ／Ｏ制御論理（「ＣＬ」）１２７２および１２８２含めることができることを示す。しかして、ＣＬ１２７２、１２８２は、一体型メモリコントローラユニットを含み、Ｉ／Ｏ制御論理も含む。図１３は、メモリ１２３２、１２３４がＣＬ１２７２、１２８２に連結されていることだけでなく、Ｉ／Ｏデバイス１３１４が制御論理１２７２、１２８２に連結されていることも示している。レガシーＩ／Ｏデバイス１３１５もチップセット１２９０に連結されている。

【００９０】

次いで図１４を参照すると、本発明の実施形態によるＳｏＣ１４００のブロック図が示されている。図１０中のものと同様な要素には同じ参照番号が付されている。また、破線のボックスは、さらに高度なＳｏＣ上の随意的な機能である。図１４において、相互接続ユニット１４０２は、一つ以上のコア２０２Ａ～Ｎのセットおよび共用キャッシュユニット１００６を含むアプリケーションプロセッサ１４１０と、システムエージェント１０１０と、バスコントローラユニット１０１６と、一体型メモリコントローラユニット１０１４と、一体型グラフィックス論理、画像プロセッサ、オーディオプロセッサ、およびビデオプロセッサを含み得る一つ以上のコプロセッサ１４２０のセットと、スタティックランダムアクセスメモリ（ＳＲＡＭ）ユニット１４３０と、直接メモリアクセス（ＤＭＡ）ユニット１４３２と、一つ以上の外部ディスプレイに連結するためのディスプレイユニット１４４０と、に連結されている。一つの実施形態において、コプロセッサ１４２０は、例えば、ネットワークまたは通信プロセッサ、圧縮エンジン、ＧＰＧＰＵ、ハイスループットＭＩＣプロセッサ、内蔵プロセッサなど、特定用途向けプロセッサを含む。

【００９１】

本明細書で開示するメカニズムの実施形態は、ハードウェア、ソフトウェア、ファームウェア、または、かかる実装アプローチの組み合わせ中に実装することができる。本発明の実施形態は、少なくとも一つのプロセッサ、ストレージシステム（揮発性および不揮発性のメモリおよび／またはストレージ要素を含む）、少なくとも一つの入力デバイス、および少なくとも一つの出力デバイスを含むプログラム可能なシステム上で実行される、コンピュータプログラムまたはプログラムコードとして実装することが可能である。

【００９２】

図１２中に示されたコード１２３０などのプログラムコードを入力命令に用いて、本明細書で説明した機能を遂行し出力情報を生成することができる。この出力情報を、周知のやり方で、一つ以上の出力デバイスに適用することが可能である。本出願の目的の上で、処理システムは、例えばデジタル信号プロセッサ（ＤＳＰ）、マイクロコントローラ、特定用途向け集積回路（）、またはマイクロプロセッサなどのプロセッサを有する一切のシステムを含む。

【００９３】

プログラムコードは、処理システムと通信するため、ハイレベルの手続き型またはオブ

ジェクト指向のプログラミング言語で実装することができる。また、当該プログラムコードは、必要な場合、アセンブリまたはマシン言語で実装することも可能である。実際上、本明細書で説明したメカニズムは、いかなる特定のプログラミング言語に対する範囲にも限定されない。いずれの場合にも、言語はコンパイラ型言語またはインタープリタ型言語とすることができる。

【００９４】

少なくとも一つの実施形態の一つ以上の態様は、マシン可読媒体上に格納された、プロセッサ内の様々な論理を表す典型的な命令群によって実装することができ、これら命令群は、マシンによって読み取られたとき、そのマシンに本明細書で説明した技法を遂行するための論理を作製させる。「ＩＰコア」として知られるかかるリプレゼンテーションは、有形のマシン可読媒体に格納して、論理またはプロセッサを実際に作る作製マシン中にロードするため、様々な顧客または製造施設に供給することができる。

【００９５】

かかるマシン可読ストレージ媒体としては、ハードディスク、フロッピディスクを含む任意の他の種類のディスク、光ディスク、コンパクトディスク読み取り専用メモリ（ＣＤ－ＲＯＭ）、再書き込み可能コンパクトディスク（ＣＤ－ＲＷ）、および光磁気ディスク、読み取り専用メモリ（ＲＯＭ）などの半導体デバイス、ダイナミックランダムアクセスメモリ（ＤＲＡＭ）、スタティックランダムアクセスメモリ（ＳＲＡＭ）などのランダムアクセスメモリ（ＲＡＭ）、消去可能プログラム可能読み取り専用メモリ（ＥＰＲＯＭ）、フラッシュメモリ、電気的に消去可能プログラム可能読み取り専用メモリ（ＥＥＰＲＯＭ）、相変化メモリ（ＰＣＭ）、磁気または光カード、もしくは、電子的命令を格納するのに好適の任意の他の種類の媒体などのストレージ媒体を含む、マシンまたはデバイスによって製造または形成された、非一時的な有形の物品の装置が挙げられ得るが、これらに限定されない。

【００９６】

したがって、本発明の実施形態は、本明細書に記載の構造体、回路、装置、プロセッサおよび／またはシステムの特徴を定義する、ハードウェア記述言語（ＨＤＬ）などの命令または設計データを包含する、非一時的な有形のマシン可読媒体も含む。かかる実施形態はプログラム製品といわれることもある。

【００９７】

エミュレーション（バイナリ変換、コードモーフなどを含む）　いくつかの例において、命令を、ソース命令セットからターゲット命令セットに変換するために命令コンバータを用いることができる。例えば、この命令コンバータは、ある命令を、コアによって処理される一つ以上の他の命令に、（例えば、静的バイナリ変換、動的コンパイルを含む動的バイナリ変換を用いて）変換、モーフ、エミュレート、または別途に変換することが可能である。この命令コンバータは、ソフトウェア、ハードウェア、ファームウェア、またはこれらの組み合わせに実装することができる。当該命令コンバータは、プロセッサ上、プロセッサの外部、または一部をプロセッサ上に一部を外部に配置することができる。

【００９８】

図１５は、本発明の実施形態による、ソース命令セット中のバイナリ命令をターゲット命令セット中のバイナリ命令に変換するためのソフトウェア命令コンバータの使用を対照しているブロック図である。示された実施形態では、この命令コンバータは、ソフトウェア命令コンバータであるが、但し、当該命令コンバータは、ソフトウェア、ファームウェア、ハードウェア、またはこれらの様々な組み合わせに実装することも可能である。図１５は、ｘ８６コンパイラ１５０４を用いて、ハイレベル言語１５０２のプログラムをコンパイルし、少なくとも一つのｘ８６命令セットコアを備えたプロセッサ１５１６がネイティブに実行できるｘ８６バイナリコード１５０６の生成が可能であることを示している。少なくとも一つのｘ８６命令セットコアを備えたプロセッサ１５１６は、（１）Ｉｎｔｅｌｘ８６命令セットコアの命令セットの実質的な部分、または（２）少なくとも一つのｘ８６命令セットコアを備えたＩｎｔｅｌプロセッサと実質的に同じ結果を達成するために

、少なくとも一つのｘ８６命令セットコアを備えたＩｎｔｅｌプロセッサ上で実行することを目的とした、アプリケーションのオブジェクトコードバージョン、または他のソフトウェアを互換的に実行または別途に処理することによって、少なくとも一つのｘ８６命令セットコアを備えたＩｎｔｅｌプロセッサと、実質的に同じ機能を遂行できる任意のプロセッサを表す。ｘ８６コンパイラ１５０４は、追加の連鎖処理のあるなしにかかわらず、少なくとも一つのｘ８６命令セットコアを備えたプロセッサ１５１６上で実行可能なｘ８６バイナリコード１５０６（例えば、オブジェクトコード）を生成するコンパイラを表す。同様に、図１５は、別の命令セットコンパイラ１５０８を用いて、ハイレベル言語１５０２によるプログラムをコンパイルし、少なくとも一つのｘ８６命令セットコアを持たないプロセッサ１５１４（例えば、カリフォルニア州、サニーベールのＭＩＰＳ　ＴｅｃｈｎｏｌｏｇｉｅｓのＭＩＰＳ命令セットおよび／またはカリフォルニア州、サニーベールのＡＲＭ　ＨｏｌｄｉｎｇｓのＡＲＭ命令セットを実行するコアを備えたプロセッサ）がネイティブに実行できる別の命令セットバイナリコード１５１０の生成が可能であることを示している。命令コンバータ１５１２は、ｘ８６バイナリコード１５０６を、ｘ８６命令セットコアを持たないプロセッサ１５１４によってネイティブに実行が可能なコードに変換するために使われる。この変換されたコードは、変換能力のある命令コンバータを造るのが難しいので、相手方の命令セットバイナリコード１５１０と同じになる可能性は低い。但し、変換されたコードは相手方の命令セットからの命令で構成されており、全般的オペレーションは達成されることになろう。しかして、命令コンバータ１５１２は、エミュレーション、シミュレーション、または任意の他の処理を介して、ｘ８６命令セットプロセッサまたはコアを持たないプロセッサまたは他の電子デバイスが、ｘ８６バイナリコード１５０６を実行できるようにするソフトウェア、ファームウェア、ハードウェア、またはこれらの組み合わせを表す。

【００９９】

いくつかの実施形態において、バックエンド論理７１７は、一つ以上の述語付き命令７１６の実行を反映するために、アーキテクチャの状態を更新するかどうかを決めることが可能である。いくつかの実施形態において、リネーム／割り当て論理７５０、スケジューラ論理７５１、実行論理７５６、除去／完遂論理７５５、物理レジスターファイルユニット７５２、またはこれらの何らかの組み合わせは、述語付き命令に関連して、条件付き実行制御または条件付き実行を行うことができる。いくつかの実施形態において、バックエンド論理は、究極的にまたは最終的に条件分岐命令が条件成立であると判定されるとき、一つ以上の述語付き命令の計算的に等価なセットを実行したことを反映するために、アーキテクチャの状態を更新しないと決めることが可能である。あるいは、この論理は、究極的にまたは最終的に条件分岐命令が条件不成立であると判定するとき、計算的に等価な一つ以上の述語付き命令のセットを実行したことを反映するために、アーキテクチャの状態を更新すると決めることができる。なお、条件分岐命令は必ずしも評価する必要はないが、当該論理は条件分岐命令に関連する条件の結果だけは知る必要があることに注意されたい。例として、各条件付き命令は、一つ以上の述語ビット（例えば、述語レジスター７５４中の一つ以上のビット）、一つ以上のフラグビット（例えば、フラグレジスター７５３中の一つ以上のビット）または類似のビットを規定することができる。述語付き命令により規定されたビットまたはビット群の値一つの値（例えば、ビットが設定されたまたは真）を有する場合、その命令は実行され、その結果はアーキテクチャの状態に反映される。もしくは、ビットまたはビット群が別の異なる値（例えば、ビットが空または偽）を有する場合、当該命令は実行されないか、あるいはその結果または影響は、アーキテクチャの状態に反映できない。

【０１００】

図１、４、５、６、または７のいずれかに対して説明されたコンポーネント、機能、および詳細は、図２または３のいずれに対しても随意的に使用することが可能である。さらに、本装置のいずれかに対して、本明細書で説明したコンポーネント、特徴、および詳細は、本明細書で説明した方法のいずれにも随意的に使用することができ、諸実施形態にお

いてかかる装置によっておよび／またはそれとともに実行することができる。
【０１０１】
例示的な実施形態　以下の例はさらなる実施形態に関連する。これらの例の中の特質的事項を一つ以上の実施形態中の任意の箇所において用いることができる。
【０１０２】
例１は、条件分岐を処理するプロセッサである。本プロセッサは、条件付きショート前方分岐を取り込むための命令取り込み論理を含み、当該条件付きショート前方分岐は、条件分岐命令と、条件分岐命令と当該条件分岐命令によって示される前方分岐ターゲット命令との間にある、条件分岐命令にプログラム順に連続して続く一つ以上の命令のセットと、を含み得る。また、本プロセッサは、命令取り込み論理と連結された命令変換論理を含み、この命令変換論理は、条件付きショート前方分岐を、一つ以上の述語付き命令の計算的に等価なセットに変換するものである。
【０１０３】
例２は、例１の主題を含み、随意的に、命令変換論理が、条件分岐命令を除去するものである。
【０１０４】
例３は、例１の主題を含み、随意的に、命令変換論理が、条件分岐命令と前方分岐ターゲット命令との間にある一つ以上の命令のセットの各々を、述語なし命令から述語付き命令に変換するものである。
【０１０５】
例４は、例１の主題を含み、随意的に、命令変換論理が、条件分岐命令と前方分岐ターゲット命令との間にある複数の命令の各々を、述語なし命令から述語付き命令に変換するものである。
【０１０６】
例５は、例１の主題を含み、随意的に、命令変換論理が、条件分岐命令が条件成立か、または不成立であるかの予測に関係なく、一つ以上の述語付き命令の計算的に等価なセットを表す信号を出力するものである。
【０１０７】
例６は、例１の主題を含み、随意的に、命令変換論理が、プロセッサのパイプラインの解読段階にハードウェア論理を含む。
【０１０８】
例７は、例１の主題を含み、随意的に、条件分岐命令と前方分岐ターゲット命令との間にある一つ以上の命令のセットが単一の移動命令から成り、命令変換論理は、当該移動命令を条件付き移動命令に変換するものである。
【０１０９】
例８は、例７の主題を含み、随意的に、条件付き移動命令は、当該条件付き移動命令の条件が偽であるとき、条件付き移動命令は例外処理を開始しない。
【０１１０】
例９は、例１～７のいずれかの主題を含み、随意的に、命令変換論理は、当該命令変換論理が条件分岐命令に対する分岐予測を知る必要なしに、一つ以上の述語付き命令の計算的に等価なセットを表す信号を出力するものである。
【０１１１】
例１０は、例１～７のいずれかの主題を含み、随意的に、命令変換論理と連結されたバックエンド論理をさらに含み、当該バックエンド論理は、一つ以上の述語付き命令の計算的に等価なセットを実行し、条件分岐命令が条件成立と判断されるとき、一つ以上の述語付き命令の計算的に等価なセットが実行されたことを反映するために、アーキテクチャの状態の更新をしないと決めるものである。
【０１１２】
例１１は、例１～７のいずれかの主題を含み、随意的に、命令変換論理と連結されたバックエンド論理をさらに含み、当該バックエンド論理は、一つ以上の述語付き命令の計算

的に等価なセットを実行し、条件分岐命令が条件不成立と判定されるのに応じて一つ以上の述語付き命令の計算的に等価なセットが実行されたことを反映するために、アーキテクチャの状態の更新をすると決めるものである。

【０１１３】

例１２は、例１～７のいずれかの主題を含み、随意的に、命令取り込み論理が、条件分岐命令の予測と関係なく、条件分岐命令と前方分岐ターゲット命令との間にある一つ以上の命令を常に取り込むものである。

【０１１４】

例１３は、条件分岐を処理する方法である。本方法は、条件付きショート前方分岐を取り込む段階であって、当該条件付きショート前方分岐は、条件分岐命令と、条件分岐命令とと当該条件分岐命令によって示される前方分岐ターゲット命令との間にある、条件分岐命令にプログラム順に連続して続く一つ以上の命令のセットと、を含む。また、本方法は、当該条件付きショート前方分岐を、一つ以上の述語付き命令の計算的に等価なセットに変換する段階も含む。

【０１１５】

例１４は、例１３の主題を含み、随意的に、条件分岐命令が条件成立か、または不成立かどうかの予測に関係なく、一つ以上の述語付き命令の計算的に等価なセットを表す信号を、プロセッサのバックエンド論理に提供する段階をさらに含む。

【０１１６】

例１５は、例１３の主題を含み、随意的に、変換する段階が、条件分岐命令と前方分岐ターゲット命令との間の一つ以上の命令のセットの各命令を、述語なし命令から述語付き命令に変換する段階を含む。

【０１１７】

例１６は、例１３の主題を含み、随意的に、変換する段階が、条件分岐命令と前方分岐ターゲット命令との間の複数の命令の各々を、述語なし命令から述語付き命令に変換する段階を含む。

【０１１８】

例１７は、例１３の主題を含み、随意的に、変換する段階が、条件分岐命令を除去し、当該変換する段階が、プロセッサのパイプラインの解読段階で変換する段階を含む。

【０１１９】

例１８は、例１３～１７のいずれかの主題を含み、随意的に、分岐予測論理のオペレーションに関係なく、一つ以上の述語付き命令の計算的に等価なセットを表す信号をプロセッサのバックエンド論理に提供する段階をさらに含む。

【０１２０】

例１９は、例１３～１７のいずれかの主題を含む。本主題は、一つ以上の述語付き命令の計算的に等価なセットを実行する段階を含む。また、本主題は、実際上、条件分岐命令が条件成立であると判定する段階と、条件分岐命令が条件成立であると判定するのに応じて一つ以上の述語付き命令の計算的に等価なセットを実行したことを反映するために、アーキテクチャの状態を更新しないと決める段階と、を含む。

【０１２１】

例２０は、例１３～１７のいずれかの主題を含む。本主題は、一つ以上の述語付き命令の計算的に等価なセットを実行する段階を含む。また、本主題は、実際上、条件分岐命令が条件不成立であると判定する段階を含む。本主題は、条件分岐命令が条件不成立であると判定するのに応じて一つ以上の述語付き命令の計算的に等価なセットを実行したことを反映するために、アーキテクチャの状態を更新すると決める段階も含む。

【０１２２】

例２１は、例１３～１７のいずれかの主題を含み、随意的に、取り込む段階が、条件分岐命令が条件成立であると予測される場合にあっても、条件分岐命令と前方分岐ターゲット命令との間にある一つ以上の命令を取り込む段階を含む。

【０１２３】

例２２は、条件分岐を処理する方法である。本方法は、条件分岐命令を検出する段階を含む。また、本方法は、当該条件分岐命令の後にショート前方分岐が続いていることを判定する段階を含む。本方法は、当該ショート前方分岐内の全ての命令が、対応する計算的に等価な述語付き命令に変換可能なことを判定する段階も含む。

【０１２４】

例２３は、例２２のいずれかの主題を含み、随意的に、ショート前方分岐内の全ての命令を、対応する計算的に等価な述語付き命令に変換する段階と、条件分岐命令を除去する段階と、をさらに含む。

【０１２５】

例２４は、例２２～２３のいずれかの主題を含む。本主題は計算的に等価な述語付き命令を実行する段階を含む。本主題は条件分岐命令が条件成立であると最終的に判定する段階を含む。本主題は、条件分岐命令が条件成立であると判定した後、計算的に等価な述語付き命令を実行したことを反映するために、アーキテクチャの状態を更新しないと決める段階を含む。

【０１２６】

例２５は、条件分岐を処理するためのシステムである。本システムは相互接続を含む。また、本システムは、相互接続に連結されたプロセッサを含み、当該プロセッサは条件付きショート前方分岐を取り込むための命令取り込み論理を含み、当該条件付きショート前方分岐は、条件分岐命令と、条件分岐命令と当該条件分岐命令によって示される前方分岐ターゲット命令との間にある、条件分岐命令にプログラム順に連続して続く一つ以上の命令のセットと、を含み得る。本システムは、命令取り込み論理と連結された命令変換論理も含み、この命令変換論理は、条件付きショート前方分岐を、一つ以上の述語付き命令の計算的に等価なセットに変換することができる。また、本システムは、相互接続と連結されたダイナミックランダムアクセスメモリ（ＤＲＡＭ）も含む。

【０１２７】

例２６は、例２５の主題を含み、随意的に、命令変換論理が、条件分岐命令を除去するものであり、さらに、命令変換論理は、条件分岐命令と前方分岐ターゲット命令との間にある一つ以上の命令のセットの各命令を、述語なし命令から述語付き命令に変換するものである。

【０１２８】

例２７は、例２５～２６のいずれかの主題を含み、命令変換論理と連結されたバックエンド論理を含み、当該バックエンド論理は、一つ以上の述語付き命令の計算的に等価なセットを実行し、条件分岐命令が条件成立であると判断されるとき、一つ以上の述語付き命令の計算的に等価なセットが実行されたことを反映するために、アーキテクチャの状態の更新をしないと決めるものである。

【０１２９】

例２８は、例１３～１７のいずれかの方法を実施するための装置を含む。

【０１３０】

例２９は、例１３～１７のいずれかの方法を実施するための手段を含む装置を含む。

【０１３１】

例３０は、例２２～２３のいずれかの方法を実施するための装置を含む。

【０１３２】

例３１は、例２２～２３のいずれかの方法を実施するための手段を含む装置を含む。

【０１３３】

例３２は、本明細書に記載の方法を実質上実施するプロセッサを含む。

【０１３４】

例３３は、本明細書に記載の方法を実質上実施するための手段を含むプロセッサを含む。

【０１３５】

本明細書および特許請求の範囲において、用語「連結された（ｃｏｕｐｌｅｄ）」およ

び「接続された（ｃｏｎｎｅｃｔｅｄ）」ならびにこれらの派生語が使われていることがある。上記の用語は、相互に同義語として意図されていないことを理解されたい。むしろ、特定の実施形態では、「接続された」は、２つ以上の要素が、相互に直接に物理的または電気的に接触していることを示すために用いられ得る。「連結された」は、２つ以上の要素が物理的または電気的に直接接触していることを意味し得る。しかしながら、「連結された」は、２つ以上の要素は相互に直接接触はしていないが、それでも相互に協働し相互作用をしていることをも意味し得る。例えば、命令変換論理は、介在する命令待ち行列または他の命令ストレージによって命令取り込み論理と結合され得る。図中の矢印は、接続および連結を示すために使われている。

【０１３６】

本明細書および特許請求の範囲において、用語「論理」が使われていることがある。本明細書で使用するように、論理は、ハードウェア、ファームウェア、ソフトウェアなどのモジュール、またはこれらの組み合せを含み得る。論理の例には、集積回路、特定用途向け集積回路、アナログ回路、デジタル回路、プログラム済み論理デバイス、命令を含むメモリデバイスなどが含まれる。いくつかの実施形態において、ハードウェア論理は、トランジスタ、および／またはゲート、ならびに他のあり得る回路コンポーネントを含み得る。

【０１３７】

用語「および／または」が使われていることがある。本明細書で使用するように、用語「および／または」は、一方または他方、または両方を意味する（例えば、Ａおよび／またはＢは、ＡまたはＢ、またはＡとＢを意味する）。

【０１３８】

上記の説明において、説明目的で、本発明の実施形態の徹底した理解を提供するために、数多くの具体的な詳細を述べてきた。しかしながら、当業者には、一つ以上の他の実施形態が、これらの具体的な詳細の一部がなくても実践可能なことは明白であろう。これら具体的な実施形態は、本発明を、限定するためでなく例示的な諸実施形態を通して説明するために提示されている。本発明の範囲は、これらの具体例によってではなく、特許請求の範囲によってのみ定まる。他の例では、周知の回路、構造体、デバイス、およびオペレーションは、説明の理解が曖昧になるのを避けるため、ブロック図の形、あるいは詳細なしに示されている。

【０１３９】

適切な場合、参照番号または参照番号の末端部分は、特記されているかまたは明らかに異なるものを除き、同様または同一の特徴を随意に有し得る、対応するまたは類似の要素を示すために、図を通して繰り返して使われている。一部の例において、複数のコンポーネントが記載されている場合、それらが単一のコンポーネントに組み込まれていることがある。他の例では、単一のコンポーネントが記載されている場合に、それが複数のコンポーネントに分けられていることもある。

【０１４０】

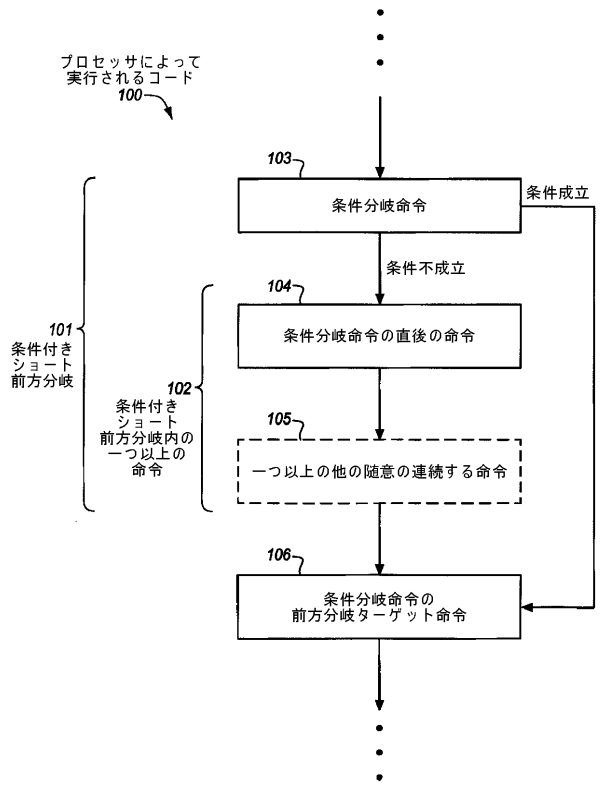
様々なオペレーションおよび方法を説明してきた。これら方法の一部は、フロー図の中では比較的基本的な形で記載されているが、随意に、これらの方法にオペレーションを加える、および／またはこれらからオペレーションを削除することが可能である。さらに、フロー図は、例示的な実施形態によるオペレーションの特定の順序を示しているが、その特定の順序は例示的なものである。別の実施形態は、随意に、異なった順序で、特定のオペレーションを組み合わせて、特定のオペレーションをオーバーラップする、などしてそれらのオペレーションを実施することができる。

【０１４１】

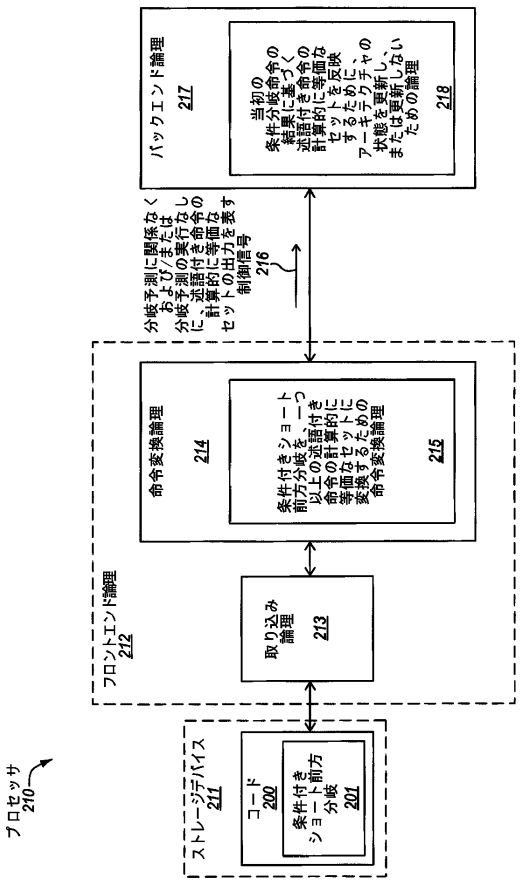
また、当然のことながら、本明細書全体を通して、例えば、「一つの実施形態」、「ある実施形態」、または「一つ以上の実施形態」への言及は、ある特定の特徴が本発明の実践の中に含まれ得ることを意味することを理解されたい。同様に、これも当然ながら、本開示を効率化し本発明の様々な態様の理解を助力するために、説明の中で、様々な特徴が

、単一の実施形態、図面、またはそれらの記述中に一緒にグループ化されていることがある。しかしながら、本開示の方法は、本発明が特許請求の範囲において明示的に列挙されたよりも多くの特徴を必要とする意図を反映していると解釈されない。むしろ、添付の特許請求の範囲に反映されているように、本発明の態様は、開示された単一の実施形態の全特徴よりも少なくしてもよい。したがって、本明細書に添付された請求項は、各請求項が本発明の別個の実施形態として各々独立しているものとして、ここに明示で本明細書に組み入れられる。

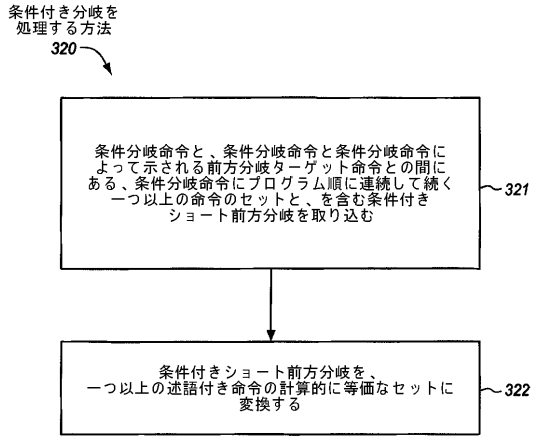
【図１】



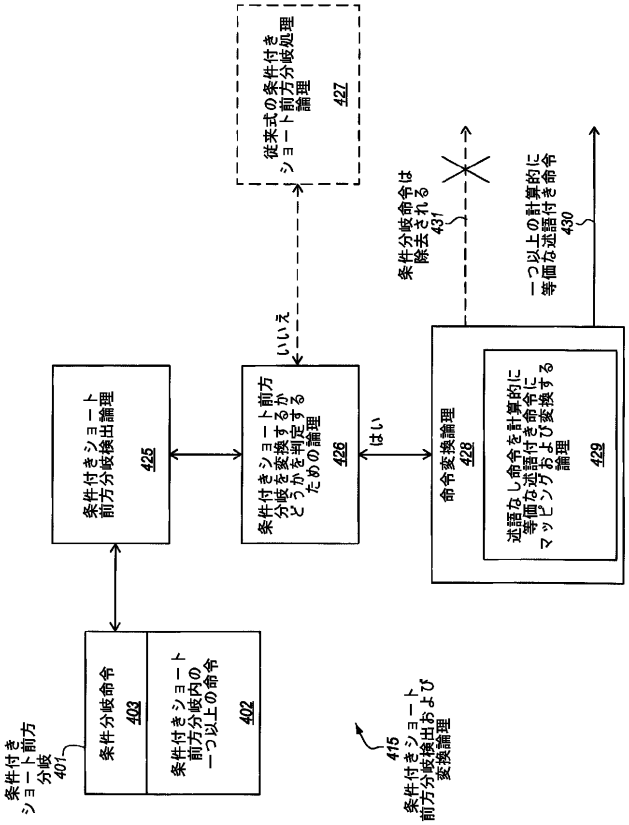
【図２】



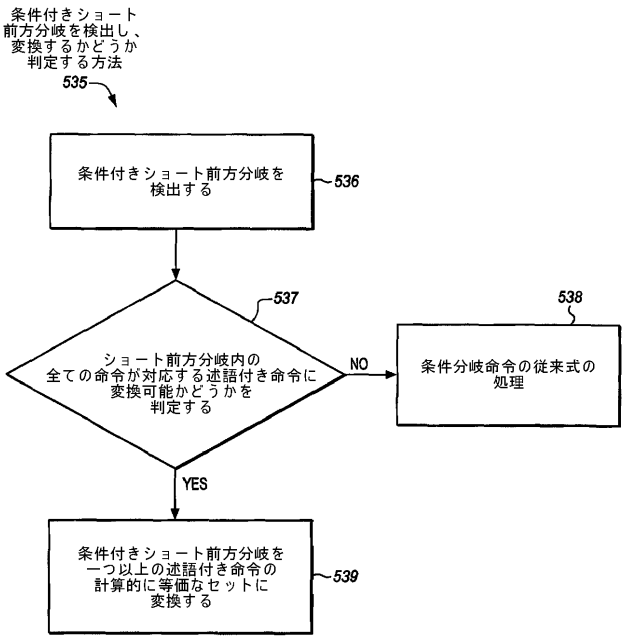
【図３】


条件付き分岐を
処理する方法
320
条件分岐命令と、条件分岐命令と条件分岐命令に
よって示される前方分岐ターゲット命令との間に
ある、条件分岐命令にプログラム順に連続して続く
一つ以上の命令のセットと、を含む条件付き
ショート前方分岐を取り込む
321
条件付きショート前方分岐を、
一つ以上の述語付き命令の計算的に等価なセットに
変換する
322

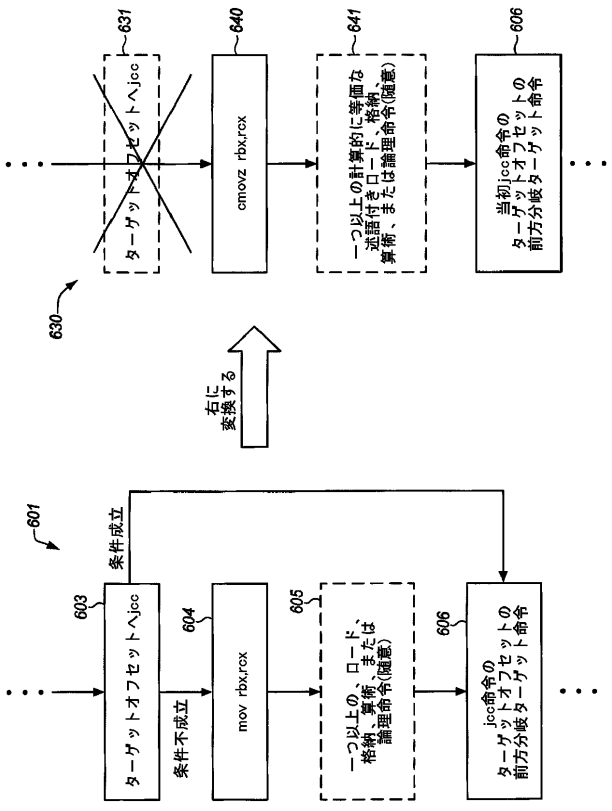
【図４】


従来式の条件付き
ショート前方分岐処理
論理
427
いいえ
条件分岐命令は
除去される
431
一つ以上の計算的に
等価な述語付き命令
430
はい
条件付きショート
前方分岐検出
論理
425
条件付きショート前方
分岐を変換するか
どうかを判定する
ための論理
426
命令変換論理
428
述語なし命令を計算的に
等価な述語付き命令に
マッピングするおよび変換する
論理
429
条件付き
ショート前方
分岐
401
条件分岐命令
403
条件付きショート
前方分岐内の
一つ以上の命令
402
415
条件付きショート
前方分岐検出および
変換論理

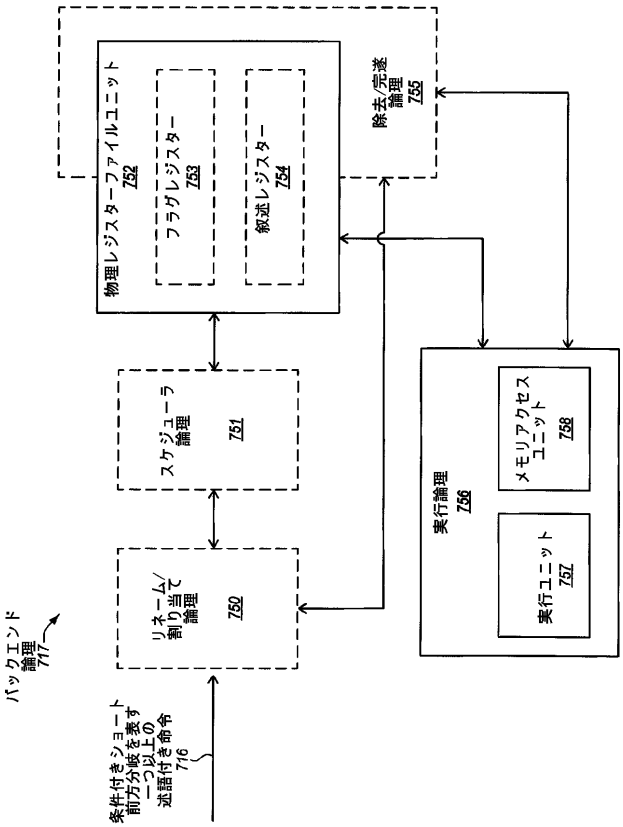
【図５】


条件付きショート
前方分岐を検出し、
変換するかどうか
判定する方法
535
条件付きショート前方分岐を
検出する
536
537
ショート前方分岐内の
全ての命令が対応する述語付き命令に
変換可能かどうかを
判定する
NO
538
条件分岐命令の従来式の
処理
YES
条件付きショート前方分岐を
一つ以上の述語付き命令の
計算的に等価なセットに
変換する
539

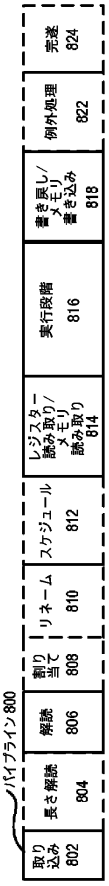
【図６】


631
ターゲットオフセットへjcc
640
cmovz rbx,rcx
641
一つ以上の計算的に等価な
述語付きロード、格納、
算術、または論理命令(随意)
606
当初jcc命令の
ターゲットオフセットの
前方分岐ターゲット命令
630
右に
変換する
601
条件成立
603
ターゲットオフセットへjcc
条件不成立
604
mov rbx,rcx
605
一つ以上の、ロード、
格納、算術、または
論理命令(随意)
606
jcc命令の
ターゲットオフセットの
前方分岐ターゲット命令

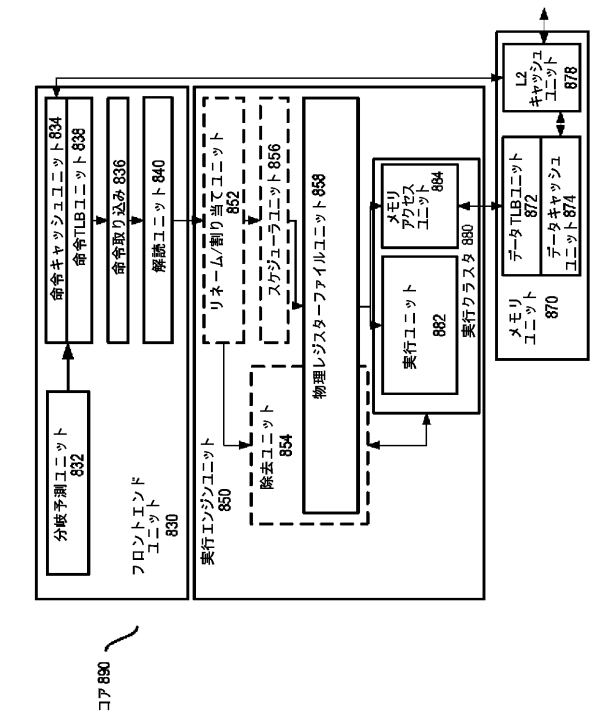
【図７】



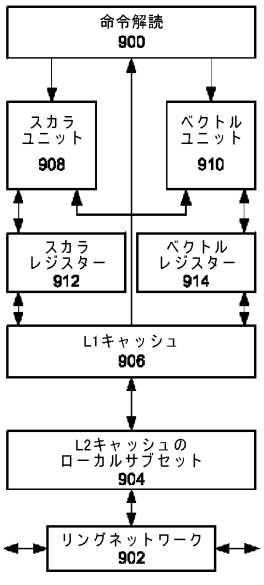
【図８Ａ】



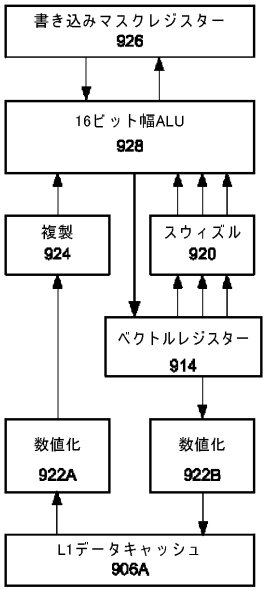
【図８Ｂ】



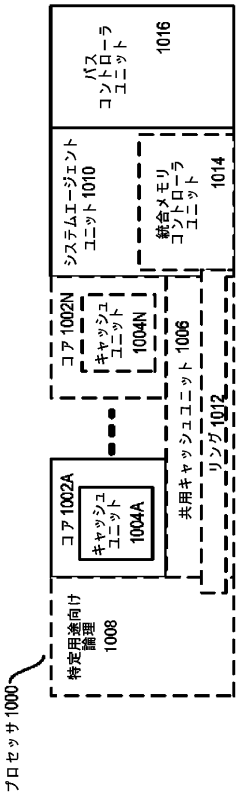
【図９Ａ】



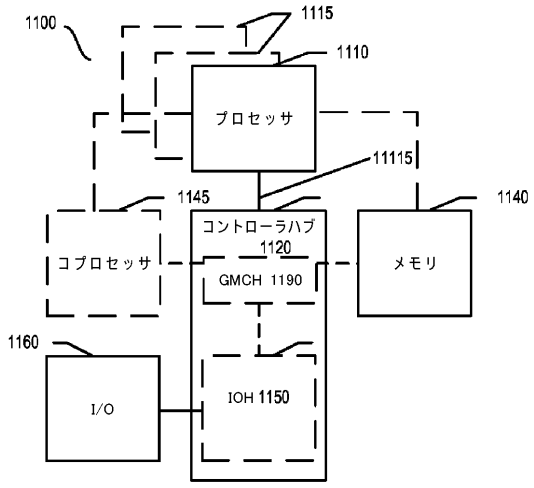
【図９Ｂ】



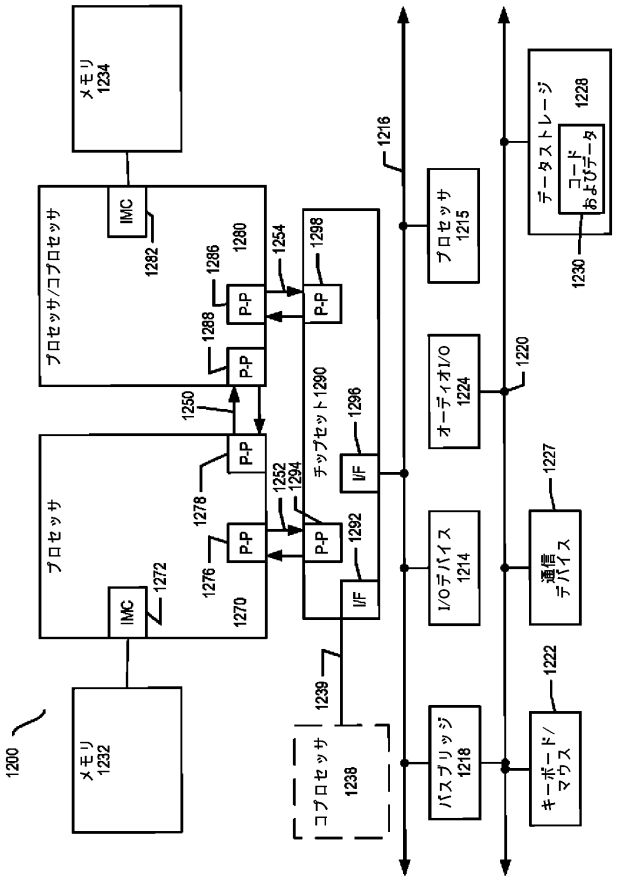
【図１０】



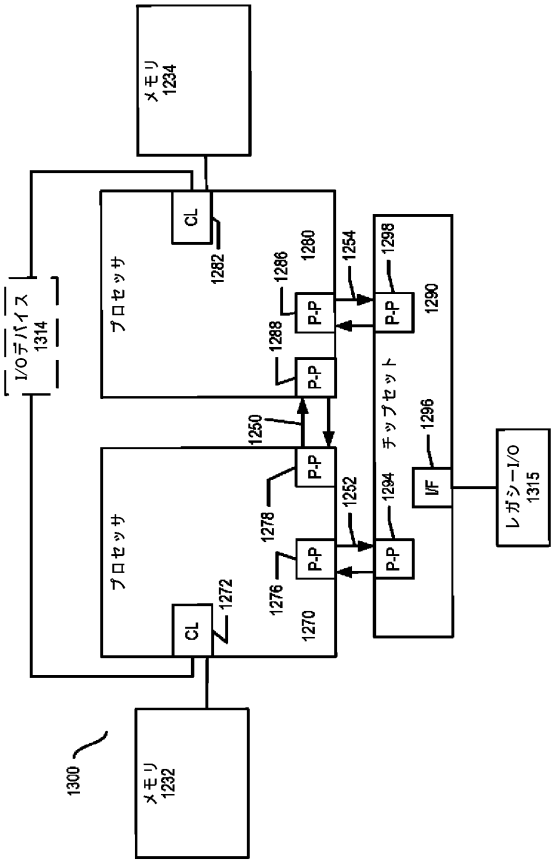
【図１１】



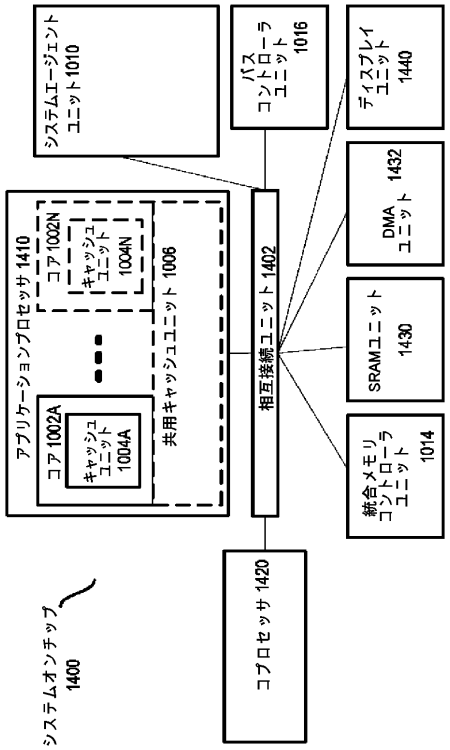
【図１２】



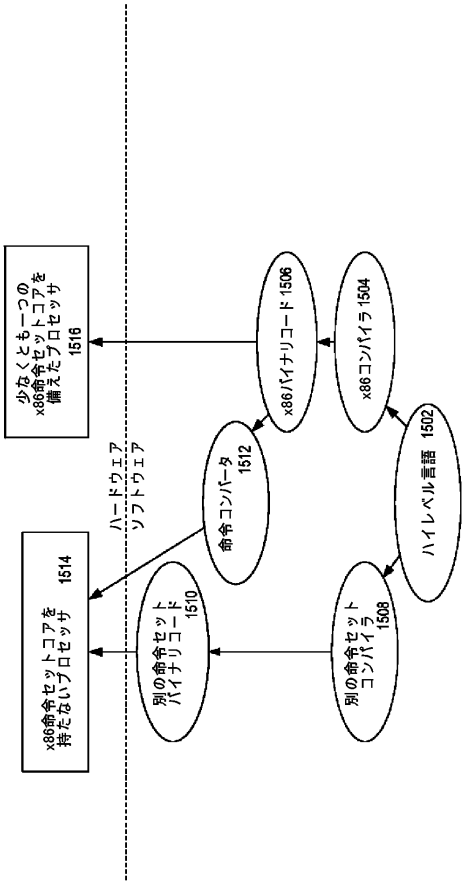
【図１３】


メモリ
1234
CL
1282
プロセッサ
1280
1286
1254
1298
P-P
1288
P-P
1290
チップセット
1250
P-P
1296
I/F
1294
1252
1278
P-P
P-P
1276
1270
CL
1272
プロセッサ
I/Oデバイス
1314
レガシーI/O
1315
メモリ
1232
1300

【図１４】


システムエージェント
ユニット1010
バス
コントローラ
ユニット
1016
ディスプレイ
ユニット
1440
アプリケーションプロセッサ 1410
コア 1002N
キャッシュ
ユニット
1004N
共用キャッシュユニット 1006
コア 1002A
キャッシュ
ユニット
1004A
相互接続ユニット 1402
DMA
ユニット
1432
SRAMユニット
1430
統合メモリ
コントローラ
ユニット
1014
コプロセッサ 1420
システムオンチップ
1400

【図１５】


少なくともひとつの
x86命令セットコアを
備えたプロセッサ
1516
x86バイナリコード 1506
x86コンパイラ 1504
命令コンバータ
1512
ハイレベル言語 1502
ハードウェア
ソフトウェア
x86命令セットコアを
持たないプロセッサ 1514
別の命令セット
バイナリコード
1510
別の命令セット
コンパイラ
1508

【手続補正書】
【提出日】平成26年5月21日(2014.5.21)
【手続補正１】
【補正対象書類名】特許請求の範囲
【補正対象項目名】全文
【補正方法】変更
【補正の内容】
【特許請求の範囲】
【請求項１】
条件分岐を処理するプロセッサであって、
条件付きショート前方分岐を取り込むための命令取り込み論理であって、前記条件付きショート前方分岐は、条件分岐命令と、一つ以上の命令のセットとを含み、前記一つ以上の命令のセットは、前記条件分岐命令と前記条件分岐命令によって示される前方分岐ターゲット命令との間にある、前記条件分岐命令にプログラム順に連続して続く命令取り込み論理と、
前記命令取り込み論理と連結された命令変換論理であって、前記条件付きショート前方分岐を、一つ以上の述語付き命令の計算的に等価なセットに変換する、命令変換論理と
を含む、プロセッサ。
【請求項２】
前記命令変換論理は、前記条件分岐命令を除去することができる、請求項１に記載のプロセッサ。
【請求項３】
前記命令変換論理が、前記条件分岐命令と前記前方分岐ターゲット命令との間にある前記一つ以上の命令の前記セットの各々を、述語なし命令から述語付き命令に変換する、請求項１または２に記載のプロセッサ。
【請求項４】
前記命令変換論理が、前記条件分岐命令と前記前方分岐ターゲット命令との間にある複数の命令の各々を、述語なし命令から述語付き命令に変換する、請求項１～３のいずれか一項に記載のプロセッサ。
【請求項５】
前記命令変換論理が、前記条件分岐命令が条件成立であるか、または不成立であるかの予測に関係なく、前記一つ以上の述語付き命令の前記計算的に等価なセットを表す信号を出力する、請求項１～４のいずれか一項に記載のプロセッサ。
【請求項６】
前記命令変換論理が、前記プロセッサのパイプラインの解読段階にハードウェア論理を含む、請求項１～５のいずれか一項に記載のプロセッサ。
【請求項７】
前記条件分岐命令と前記前方分岐ターゲット命令との間にある前記一つ以上の命令の前記セットが単一の移動命令からなり、前記命令変換論理は、前記単一の移動命令を条件付き移動命令に変換する、請求項１～６のいずれか一項に記載のプロセッサ。
【請求項８】
前記条件付き移動命令は、前記条件付き移動命令の条件が偽であるとき、例外処理を開始しない、請求項７に記載のプロセッサ。
【請求項９】
前記命令変換論理は、前記命令変換論理が前記条件分岐命令に対する分岐予測を知る必要なしに、前記一つ以上の述語付き命令の前記計算的に等価なセットを表す信号を出力する、請求項１～７のいずれか一項に記載のプロセッサ。
【請求項１０】
前記命令変換論理と連結されたバックエンド論理をさらに含み、前記バックエンド論理は、前記一つ以上の述語付き命令の前記計算的に等価なセットを実行し、前記条件分岐命

令が条件成立であると判定されると、前記一つ以上の述語付き命令の前記計算的に等価なセットが前記実行を反映するために、アーキテクチャの状態の更新をしないと決める、請求項１～７のいずれか一項に記載のプロセッサ。

【請求項１１】

前記命令変換論理と連結されたバックエンド論理をさらに含み、前記バックエンド論理は、前記一つ以上の述語付き命令の前記計算的に等価なセットを実行し、前記条件分岐命令が条件不成立であると判定されると、前記一つ以上の述語付き命令の前記計算的に等価なセットの前記実行を反映するために、アーキテクチャの状態の更新をすると決める、請求項１～７のいずれか一項に記載のプロセッサ。

【請求項１２】

前記命令取り込み論理が、前記条件分岐命令の予測と関係なく、前記条件分岐命令と前記前方分岐ターゲット命令との間にある前記一つ以上の命令を常に取り込む、請求項１～７のいずれか一項に記載のプロセッサ。

【請求項１３】

条件分岐を処理する方法であって、

条件付きショート前方分岐を取り込む段階であって、前記条件付きショート前方分岐は、条件分岐命令と、一つ以上の命令のセットとを含み、前記一つ以上の命令のセットは、前記条件分岐命令と前記条件分岐命令によって示される前方分岐ターゲット命令との間にある、前記条件分岐命令にプログラム順に連続して続く段階と、

前記条件付きショート前方分岐を、一つ以上の述語付き命令の計算的に等価なセットに変換する段階と

を含む、方法。

【請求項１４】

前記条件分岐命令が条件成立であるか、または不成立かどうかの予測に関係なく、前記一つ以上の述語付き命令の前記計算的に等価なセットを表す信号を、プロセッサのバックエンド論理に提供する段階をさらに含む、請求項１３に記載の方法。

【請求項１５】

前記変換する段階が、前記条件分岐命令と前記前方分岐ターゲット命令との間の前記一つ以上の命令の前記セットの各々を、述語なし命令から述語付き命令に変換する段階を含む、請求項１３に記載の方法。

【請求項１６】

前記変換する段階が、前記条件分岐命令と前記前方分岐ターゲット命令との間の複数の命令の各々を、述語なし命令から述語付き命令に変換する段階を含む、請求項１３に記載の方法。

【請求項１７】

前記変換する段階が前記条件分岐命令を除去する段階を含み、前記変換する段階は、プロセッサのパイプラインの解読段階で変換する段階を含む、請求項１３に記載の方法。

【請求項１８】

分岐予測論理のオペレーションに関係なく、前記一つ以上の述語付き命令の前記計算的に等価なセットを表す信号を、プロセッサのバックエンド論理に提供する段階をさらに含む、請求項１３～１７のいずれか一項に記載の方法。

【請求項１９】

前記一つ以上の述語付き命令の前記計算的に等価なセットを実行する段階と、

実際上、前記条件分岐命令が条件成立であると判定する段階と、

前記条件分岐命令が条件成立であると判定するのに応じて前記一つ以上の述語付き命令の前記計算的に等価なセットの前記実行を反映するために、アーキテクチャの状態を更新しないと決める段階と、

をさらに含む、請求項１３～１７のいずれか一項に記載の方法。

【請求項２０】

前記一つ以上の述語付き命令の前記計算的に等価なセットを実行する段階と、

実際上、前記条件分岐命令が条件不成立であると判定する段階と、
前記条件分岐命令が条件不成立であると判定するのに応じて、前記一つ以上の述語付き命令の前記計算的に等価なセットを前記実行したことを反映するために、アーキテクチャの状態を更新すると決める段階と、
をさらに含む、請求項１３～１７のいずれか一項に記載の方法。
【請求項２１】
前記取り込む段階は、前記条件分岐命令が条件成立であると予測される場合にあっても、前記条件分岐命令と前記前方分岐ターゲット命令との間にある前記一つ以上の命令を取り込む段階を含む、請求項１３～１７のいずれかに一項に記載の方法。
【請求項２２】
条件分岐を処理する方法であって、
条件分岐命令を検出する段階と、
前記条件分岐命令の後にショート前方分岐が続いていると判定する段階と、
前記ショート前方分岐内の全ての命令が、対応する計算的に等価な述語付き命令に変換可能なことを判定する段階と
を含む、方法。
【請求項２３】
前記ショート前方分岐内の前記命令の前記全てを、前記対応する計算的に等価な述語付き命令に変換する段階と、前記条件分岐命令を除去する段階と、をさらに含む、請求項２２に記載の方法。
【請求項２４】
前記計算的に等価な述語付き命令を実行する段階と、
前記条件分岐命令が条件成立であると最終的に判定する段階と、
前記条件分岐命令が条件成立であると判定した後、前記計算的に等価な述語付き命令の前記実行を反映するために、アーキテクチャの状態を更新しないと決める段階と
をさらに含む、請求項２２または２３に記載の方法。
【請求項２５】
条件分岐を処理するためのシステムであって、
相互接続と、
前記相互接続に連結されたプロセッサであって、前記プロセッサは、
条件付きショート前方分岐を取り込むための命令取り込み論理であって、前記条件付きショート前方分岐は、条件分岐命令と、一つ以上の命令のセットとを含み、前記一つ以上の命令のセットは、前記条件分岐命令と前記条件分岐命令によって示される前方分岐ターゲット命令との間にある、前記条件分岐命令にプログラム順に連続して続く命令取り込み論理と、
前記命令取り込み論理と連結された命令変換論理であって、前記命令変換論理は、前記条件付きショート前方分岐を、一つ以上の述語付き命令の計算的に等価なセットに変換する、前記論理と
を含む前記プロセッサと、
前記相互接続と連結されたダイナミックランダムアクセスメモリ（ＤＲＡＭ）と
を備える、システム。
【請求項２６】
前記命令変換論理が前記条件分岐命令を除去し、さらに、前記命令変換論理は、前記条件分岐命令と前記前方分岐ターゲット命令との間にある前記一つ以上の命令の前記セットの各々を、述語なし命令から述語付き命令に変換する、請求項２５に記載のシステム。
【請求項２７】
命令変換論理と連結されたバックエンド論理であって、前記バックエンド論理は、前記一つ以上の述語付き命令の前記計算的に等価なセットを実行し、前記条件分岐命令が条件成立であると判定されるとき、前記一つ以上の述語付き命令の前記計算的に等価なセットの前記実行を反映するために、アーキテクチャの状態の更新をしないと決める、前記バッ

クエンド論理をさらに含む、請求項２５または２６に記載のシステム。
【請求項２８】
請求項１３～１７のいずれか一項に記載の方法を実施するための装置。
【請求項２９】
請求項１３～１７のいずれか一項に記載の方法を実施するための手段を含む装置。
【請求項３０】
請求項２２または２３に記載の方法を実施するための装置。
【請求項３１】
請求項２２または２３に記載の方法を実施するための手段を含む装置。
【請求項３２】
本明細書に記載の方法を実質上実施するプロセッサ。
【請求項３３】
本明細書に記載の方法を実質上実施するための手段を含むプロセッサ。
【手続補正２】
【補正対象書類名】明細書
【補正対象項目名】０１４１
【補正方法】変更
【補正の内容】
【０１４１】
また、当然のことながら、本明細書全体を通して、例えば、「一つの実施形態」、「ある実施形態」、または「一つ以上の実施形態」への言及は、ある特定の特徴が本発明の実践の中に含まれ得ることを意味することを理解されたい。同様に、これも当然ながら、本開示を効率化し本発明の様々な態様の理解を助力するために、説明の中で、様々な特徴が、単一の実施形態、図面、またはそれらの記述中に一緒にグループ化されていることがある。しかしながら、本開示の方法は、本発明が特許請求の範囲において明示的に列挙されたよりも多くの特徴を必要とする意図を反映していると解釈されない。むしろ、添付の特許請求の範囲に反映されているように、本発明の態様は、開示された単一の実施形態の全特徴よりも少なくしてもよい。したがって、本明細書に添付された請求項は、各請求項が本発明の別個の実施形態として各々独立しているものとして、ここに明示で本明細書に組み入れられる。
（項目１）
条件分岐を処理するプロセッサであって、
条件付きショート前方分岐を取り込むための命令取り込み論理であって、上記条件付きショート前方分岐は、条件分岐命令と、一つ以上の命令のセットとを含み、上記一つ以上の命令のセットは、上記条件分岐命令と上記条件分岐命令によって示される前方分岐ターゲット命令との間にある、上記条件分岐命令にプログラム順に連続して続く命令取り込み論理と、
上記命令取り込み論理と連結された命令変換論理であって、上記条件付きショート前方分岐を、一つ以上の述語付き命令の計算的に等価なセットに変換する、命令変換論理と
を含む、プロセッサ。
（項目２）
上記命令変換論理は、上記条件分岐命令を除去することができる、項目１に記載のプロセッサ。
（項目３）
上記命令変換論理が、上記条件分岐命令と上記前方分岐ターゲット命令との間にある上記一つ以上の命令の上記セットの各々を、述語なし命令から述語付き命令に変換する、項目１または２に記載のプロセッサ。
（項目４）
上記命令変換論理が、上記条件分岐命令と上記前方分岐ターゲット命令との間にある複数の命令の各々を、述語なし命令から述語付き命令に変換する、項目１～３のいずれか一

項に記載のプロセッサ。
（項目５）
上記命令変換論理が、上記条件分岐命令が条件成立であるか、または不成立であるかの予測に関係なく、上記一つ以上の述語付き命令の上記計算的に等価なセットを表す信号を出力する、項目１～４のいずれか一項に記載のプロセッサ。
（項目６）
上記命令変換論理が、上記プロセッサのパイプラインの解読段階にハードウェア論理を含む、項目１～５のいずれか一項に記載のプロセッサ。
（項目７）
上記条件分岐命令と上記前方分岐ターゲット命令との間にある上記一つ以上の命令の上記セットが単一の移動命令からなり、上記命令変換論理は、上記単一の移動命令を条件付き移動命令に変換する、項目１～６のいずれか一項に記載のプロセッサ。
（項目８）
上記条件付き移動命令は、上記条件付き移動命令の条件が偽であるとき、例外処理を開始しない、項目７に記載のプロセッサ。
（項目９）
上記命令変換論理は、上記命令変換論理が上記条件分岐命令に対する分岐予測を知る必要なしに、上記一つ以上の述語付き命令の上記計算的に等価なセットを表す信号を出力する、項目１に記載のプロセッサ。
（項目１０）
上記命令変換論理と連結されたバックエンド論理をさらに含み、上記バックエンド論理は、上記一つ以上の述語付き命令の上記計算的に等価なセットを実行し、上記条件分岐命令が条件成立であると判定されると、上記一つ以上の述語付き命令の上記計算的に等価なセットの上記実行を反映するために、アーキテクチャの状態の更新をしないと決める、項目１に記載のプロセッサ。
（項目１１）
上記命令変換論理と連結されたバックエンド論理をさらに含み、上記バックエンド論理は、上記一つ以上の述語付き命令の上記計算的に等価なセットを実行し、上記条件分岐命令が条件不成立であると判定されると、上記一つ以上の述語付き命令の上記計算的に等価なセットの上記実行を反映するために、アーキテクチャの状態の更新をすると決める、項目１に記載のプロセッサ。
（項目１２）
上記命令取り込み論理が、上記条件分岐命令の予測と関係なく、上記条件分岐命令と上記前方分岐ターゲット命令との間にある上記一つ以上の命令を常に取り込む、項目１に記載のプロセッサ。
（項目１３）
条件分岐を処理する方法であって、
条件付きショート前方分岐を取り込む段階であって、上記条件付きショート前方分岐は、条件分岐命令と、一つ以上の命令のセットとを含み、上記一つ以上の命令のセットは、上記条件分岐命令と上記条件分岐命令によって示される前方分岐ターゲット命令との間にある、上記条件分岐命令にプログラム順に連続して続く段階と、
上記条件付きショート前方分岐を、一つ以上の述語付き命令の計算的に等価なセットに変換する段階と
を含む、方法。
（項目１４）
上記条件分岐命令が条件成立であるか、または不成立かどうかの予測に関係なく、上記一つ以上の述語付き命令の上記計算的に等価なセットを表す信号を、プロセッサのバックエンド論理に提供する段階をさらに含む、項目１３に記載の方法。
（項目１５）
上記変換する段階が、上記条件分岐命令と上記前方分岐ターゲット命令との間の上記一

つ以上の命令の上記セットの各々を、述語なし命令から述語付き命令に変換する段階を含む、項目１３または１４に記載の方法。
（項目１６）
上記変換する段階が、上記条件分岐命令と上記前方分岐ターゲット命令との間の複数の命令の各々を、述語なし命令から述語付き命令に変換する段階を含む、項目１３または１４に記載の方法。
（項目１７）
上記変換する段階が上記条件分岐命令を除去する段階を含み、上記変換する段階は、プロセッサのパイプラインの解読段階で変換する段階を含む、項目１３または１４に記載の方法。
（項目１８）
分岐予測論理のオペレーションに関係なく、上記一つ以上の述語付き命令の上記計算的に等価なセットを表す信号を、プロセッサのバックエンド論理に提供する段階をさらに含む、項目１３～１７のいずれか一項に記載の方法。
（項目１９）
上記一つ以上の述語付き命令の上記計算的に等価なセットを実行する段階と、
実際上、上記条件分岐命令が条件成立であると判定する段階と、
上記条件分岐命令が条件成立であると判定するのに応じて上記一つ以上の述語付き命令の上記計算的に等価なセットの上記実行を反映するために、アーキテクチャの状態を更新しないと決める段階と、
をさらに含む、項目１３～１８のいずれか一項に記載の方法。
（項目２０）
条件分岐を処理する方法であって、
条件分岐命令を検出する段階と、
上記条件分岐命令の後にショート前方分岐が続いていると判定する段階と、
上記ショート前方分岐内の全ての命令が、対応する計算的に等価な述語付き命令に変換可能なことを判定する段階と
を含む、方法。
（項目２１）
上記ショート前方分岐内の上記命令の上記全てを、上記対応する計算的に等価な述語付き命令に変換する段階と、上記条件分岐命令を除去する段階と、をさらに含む、項目２０に記載の方法。
（項目２２）
上記計算的に等価な述語付き命令を実行する段階と、
上記条件分岐命令が条件成立であると最終的に判定する段階と、
上記条件分岐命令が条件成立であると判定した後、上記計算的に等価な述語付き命令の上記実行を反映するために、アーキテクチャの状態を更新しないと決める段階と、
をさらに含む、項目２０に記載の方法。
（項目２３）
条件分岐を処理するためのシステムであって、
相互接続と、
上記相互接続に連結されたプロセッサであって、
条件付きショート前方分岐を取り込むための命令取り込み論理であって、上記条件付きショート前方分岐は、条件分岐命令と、一つ以上の命令のセットとを含み、上記一つ以上の命令のセットは、上記条件分岐命令と上記条件分岐命令によって示される前方分岐ターゲット命令との間にある、上記条件分岐命令にプログラム順に連続して続く命令取り込み論理と、
上記命令取り込み論理と連結された命令変換論理であって、上記条件付きショート前方分岐を、一つ以上の述語付き命令の計算的に等価なセットに変換する、上記命令変換論理と

を含む上記プロセッサと、
上記相互接続と連結されたダイナミックランダムアクセスメモリ（ＤＲＡＭ）と
を備える、システム。
（項目２４）
上記命令変換論理が上記条件分岐命令を除去し、さらに、上記命令変換論理は、上記条件分岐命令と上記前方分岐ターゲット命令との間にある上記一つ以上の命令の上記セットの各々を、述語なし命令から述語付き命令に変換する、項目２３に記載のシステム。

フロントページの続き

(72)発明者　ディクソン、マーティン　ジー.
アメリカ合衆国　９５０５４　カリフォルニア州・サンタクララ・ミッション　カレッジ　ブーレバード・２２００　インテル・コーポレーション内
(72)発明者　サンティアゴ、ヤズミン　エー.
アメリカ合衆国　９５０５４　カリフォルニア州・サンタクララ・ミッション　カレッジ　ブーレバード・２２００　インテル・コーポレーション内
(72)発明者　ナイク、ミシャリ
アメリカ合衆国　９５０５４　カリフォルニア州・サンタクララ・ミッション　カレッジ　ブーレバード・２２００　インテル・コーポレーション内

Ｆターム(参考) 5B033 BB03　CA09　CA13

【外国語明細書】
2014182817000001.pdf