(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第5403973号
(P5403973)
(45)発行日　平成26年1月29日(2014.1.29)　(24)登録日　平成25年11月8日(2013.11.8)

(51)Int.Cl.　　F I
*G06F 9/38 (2006.01)*　　G06F 9/38 350B
*G06F 9/30 (2006.01)*　　G06F 9/30 350A

請求項の数 11　(全 12 頁)

(21)出願番号　特願2008-230625 (P2008-230625)
(22)出願日　平成20年9月9日 (2008.9.9)
(65)公開番号　特開2009-70378 (P2009-70378A)
(43)公開日　平成21年4月2日 (2009.4.2)
審査請求日　平成23年8月17日 (2011.8.17)
(31)優先権主張番号　11/856170
(32)優先日　平成19年9月17日 (2007.9.17)
(33)優先権主張国　米国(US)

前置審査

(73)特許権者　390009531
インターナショナル・ビジネス・マシーンズ・コーポレーション
ＩＮＴＥＲＮＡＴＩＯＮＡＬ　ＢＵＳＩＮＥＳＳ　ＭＡＣＨＩＮＥＳ　ＣＯＲＰＯＲＡＴＩＯＮ
アメリカ合衆国１０５０４　ニューヨーク州　アーモンク　ニュー　オーチャード　ロード
(74)代理人　100108501
弁理士　上野　剛史
(74)代理人　100112690
弁理士　太佐　種一

最終頁に続く

(54)【発明の名称】アウト・オブ・オーダ・プロセッサにおける述語型実行のための方法および装置

(57)【特許請求の範囲】
【請求項１】
アウト・オブ・オーダ・プロセッサにおいて述語付き命令を処理するための方法であって、
述語付き命令を含む述語付きコード領域に関連した述語定義命令を検出するステップと、
前記述語定義命令の述語が解明されるまで、前記述語付きコード領域内に含まれた述語付き命令のリネーミングをストールするステップと、
前記述語が解明されたという表示を受け取るステップと、
選択されたパスに関連する述語付き命令をリネームするステップと、
選択されなかったパスに関連する述語付き命令を、リネームされた前記述語付き命令を実行する前に、廃棄するステップと、
を含む、方法。
【請求項２】
前記述語定義命令を検出するステップは、前記述語付きコード領域に関連した述語定義命令を前記アウト・オブ・オーダ・プロセッサのリネーム・ステージにおいて検出するステップを更に含む、請求項１に記載の方法。
【請求項３】
前記述語定義命令に関連したマスクをマスク・シフト・レジスタに格納するステップと、

前記述語付き命令のうちのどれをリネームおよび実行すべきかを決定するために前記述語および前記格納されたマスクに関して論理機能を遂行するステップと、
を更に含む、請求項２に記載の方法。
【請求項４】
前記述語は状態レジスタ値に基づいている、請求項１に記載の方法。
【請求項５】
前記述語定義命令をデコードするステップと、
前記述語付き命令をデコードするステップと、
を更に含む、請求項１に記載の方法。
【請求項６】
述語付き命令を含む述語付きコード領域に関連した述語定義命令をデコードするように構成されたデコーダと、
前記述語定義命令を検出するように、および、前記述語定義命令の述語が解明されるまで前記述語付き命令のリネーミングをストールするように構成されたパイプライン・ステージと、を含み、
前記パイプライン・ステージは、更に、
前記述語が解明されたという表示を受け取り、
選択されたパスに関連する述語付き命令をリネームし、
選択されなかったパスに関連する述語付き命令を、リネームされた前記述語付き命令を実行する前に、廃棄する、
アウト・オブ・オーダ・プロセッサ。
【請求項７】
前記パイプライン・ステージはリネーム・ステージに含まれる、請求項６に記載のアウト・オブ・オーダ・プロセッサ。
【請求項８】
前記リネーム・ステージは、更に、　前記述語定義命令に関連するマスクをマスク・シフト・レジスタに格納し、
前記述語付き命令のうちのどれをリネームおよび実行すべきかを決定するために前記述語および前記格納されたマスクに関して論理機能を遂行する
ように構成される、請求項７に記載のアウト・オブ・オーダ・プロセッサ。
【請求項９】
前記述語は状態レジスタ値に基づいている、請求項６に記載のアウト・オブ・オーダ・プロセッサ。
【請求項１０】
メモリ・サブシステムと、
前記メモリ・サブシステムに接続されたアウト・オブ・オーダ・プロセッサと、を含み、
前記アウト・オブ・オーダ・プロセッサは、
述語付き命令を含む述語付きコード領域に関連した述語定義命令をデコードするように構成されたデコーダと、
前記述語定義命令を検出するように、および、前記述語定義命令の述語が解明されるまで前記述語付き命令のリネーミングをストールするように構成されたリネーム・ステージとを含み、
前記リネーム・ステージは、更に、
前記述語が解明されたという表示を受け取り、
選択されたパスに関連する述語付き命令をリネームし、
選択されなかったパスに関連する述語付き命令を、リネームされた前記述語付き命令を実行する前に、廃棄する、
プロセッサ・システム。
【請求項１１】

前記リネーム・ステージは、更に、
前記述語定義命令に関連したマスクをマスク・シフト・レジスタに格納し、
前記述語付き命令のうちのどれを実行すべきかを決定するために前記述語および前記格納されたマスクに関して論理機能を遂行し、
前記論理機能の結果に基づいて前記述語付き命令の少なくとも１つをリネームする
ように構成される、請求項１０に記載のプロセッサ・システム。

【発明の詳細な説明】

【技術分野】

【０００１】

本発明は、一般的には、述語型の実行（predicated execution）に関し、特に、アウト・オブ・オーダ（out-of-order）・プロセッサにおける述語型の実行のための技術に関するものである。

【背景技術】

【０００２】

現在、ブランチ命令は、コンピュータ・プログラム（プログラム）においてどれだけの数のオペレーションを同時に実行し得るかに関する１つの尺度である命令レベル並列度（ＩＬＰ）の活用にとって大きな障害と考えられる。一般に、コンパイラおよびハードウェアは、十分なＩＬＰを達成するために、頻繁且つ正確なブランチ予測を行うことを要求される。一般に、ブランチ誤予測は、命令ストリームに取り込まれる不用なサイクルのためにパフォーマンスの低下を生じさせる。スーパースカラ・プロセッサおよび超長命令語（ＶＬＩＷ）プロセッサにおけるブランチ誤予測は、各不用なサイクルが複数の命令によりスループットを減少させることがあるスカラ・プロセッサにおけるブランチ誤予測よりもはるかに大きくパフォーマンスを低下させる。

【０００３】

ブール・ソース・オペランド（命令の述語（predicate）として知られている）の値に基づいた命令の条件付き実行と呼ばれる述語型の実行は、命令ストリームからブランチを除去するための技術を提供する。典型的な実施方法では、述語型の実行を使用するコンパイラは、条件付きのブランチを述語定義（predicate defining）命令に変換するために、および代替パスに沿った命令ストリームを述語付き命令に変換するためにif変換アルゴリズムを使用する。典型的な場合では、述語付き命令は、それらの述語値に関係なくフェッチされる。真の述語を有する命令は正常に実行されるが、偽の述語値を有する命令はヌル化にされ、従ってそのヌル化された命令はプロセッサ状態を修正することを妨げられる。一般に、述語型の実行は、コンパイラが複数の実行パスに沿って、低い命令フェッチ効率ではあるが、ハードウェアにＩＬＰを施すことを可能にする。

【０００４】

アウト・オブ・オーダ実行は、データ依存性を損なわない任意の順で命令が実行されることを可能にする。アウト・オブ・オーダ実行は、パイプライン処理およびスーパースカラ技術と関連して使用されてもよく、或いは、使用されなくてもよい。アウト・オブ・オーダ実行は、多くの高性能プロセッサにおいてプロセッサ・サイクルを利用するために使用され、それ以外には利用されないであろう。アウト・オブ・オーダ実行の主要な注目点は、オペレーションを遂行するために必要なデータが得られないときに生じるストールをプロセッサが回避することを可能にすることである。アウト・オブ・オーダ・プロセッサでは、命令はデータ順に、即ち、データ・オペランドがプロセッサのレジスタにおいて利用可能になる順に処理される。アウト・オブ・オーダ・プロセッサは、データが関連の命令にとって利用可能でないとき、実行する準備ができている他の命令でもってスロットを満たす。その後、アウト・オブ・オーダ・プロセッサは、命令がアウト・オブ・オーダで実行されたことをアーキテクチャ状態に対してコミットする前に、命令がプログラム順に、即ち、オリジナル・プログラムにおける命令の順に処理されたように見えるよう、結果を再順序付ける。

【０００５】

命令を成功裡に再順序付けするために、現在のアウト・オブ・オーダ・プロセッサは、リネーミングと呼ばれる技術を使用する。リネーミングは、命令によって生成された各結果レジスタに一意的な物理的位置を割当てることを含む。実施方法は、一般に、予約ステーション、レジスタ更新ユニット（ＲＵＵ）、物理的レジスタ・ファイル等を使用する。アウト・オブ・オーダ・プロセッサは、生成する命令の結果オペランドの可用性を追跡し、結果オペランドを得るべき一意的な物理的ロケーションを（従属的な消費命令(consumer instruction)に）指定する。叙述化（predication）は、各々がアウト・オブ・オーダ・プロセッサにおける一意的な物理的ロケーションにリネームされる複数の命令を同じアーキテクチャ上のレジスタに導入することが可能である。その結果、アウト・オブ・オーダ・プロセッサは、ソース・オペランドの正確なロケーションを（待機する消費命令にとっては）一意的に識別することができないであろう。従って、そのような状況を扱う明確なサポートがない場合、すべての作動時(in-flight)の生成命令(producerinstruction)が完了するまで、アウト・オブ・オーダ・プロセッサはストールしており、その結果、非効率的なパフォーマンスを生じる。述語予測を使用するプロセッサは多くの米国特許に開示されている（例えば、米国特許第7,085,919号、同第6,513,109号、および同第6,442,679号参照）。

【特許文献１】米国特許第7,085,919号、
【特許文献２】米国特許第6,513,109号、
【特許文献３】米国特許第6,442,679号

【発明の開示】

【発明が解決しようとする課題】

【０００６】

本発明の目的は、アウト・オブ・オーダ・プロセッサにおいて予測実行するための方法および装置を提供することにある。

【課題を解決するための手段】

【０００７】

本発明の種々の実施態様によれば、アウト・オブ・オーダ・プロセッサにおいて述語付きコードを扱うための技術は、述語付きコード領域と関連した述語定義命令を検出することを含む。更に、述語付き定義命令の述語が解明されるまで、述語付きコード領域内の述語付き命令のリネーミングはストールされる。

【発明を実施するための最良の形態】

【０００８】

当業者には明らかなように、本発明は、方法、システム、装置、或いはコンピュータ・プログラムとして具体化することが可能である。従って、本発明は、全体的にハードウェアの実施例、全体的にソフトウェアの実施例（ファームウェア、常駐ソフトウェア、マイクロコード等を含む）、或いは、「回路」、「モジュール」、または「システム」として本明細書において一般的に参照されるソフトウェアおよびハードウェアを組み合わせた実施例の形態を取ることが可能である。本発明は、例えば、コンピュータ使用可能プログラム・コードを有するコンピュータ使用可能記憶媒体上でコンピュータ・プログラムの形、例えば、その媒体において具現化された１つまたは複数の設計ファイルの形を取ることが可能である。

【０００９】

任意の適当なコンピュータ使用可能記憶媒体またはコンピュータ可読記憶媒体が利用可能である。コンピュータ使用可能または可読記憶媒体は、例えば、電子的、磁気的、光学的、電磁気的、赤外線、或いは、半導体システム、装置、またはデバイスであってもよいが、それに限定されない。コンピュータ可読媒体の更に特定な例（非網羅的なリスト）は、下記のもの、即ち、ポータブル・コンピュータ・ディスケット、ハードディスク、ランダム・アクセス・メモリ（ＲＡＭ）、読取り専用メモリ（ＲＯＭ）、消去可能プログラマブルＲＯＭ（ＥＰＲＯＭまたはフラッシュ・メモリ）、ポータブル・コンパクト・ディスクＲＯＭ（ＣＤ－ＲＯＭ）、光学的記憶装置、または磁気記憶装置を含む。

【００１０】
オペランド・リネーミング前に述語値を予測するために、種々の技術を使用することが可能である。この場合、通常は、予測され且つ選定されたパス（predicted-takenpath）命令が処理されるが、予測され且つ選定されなかったパス（predicted non-taken path）命令はプロセッサ・バッファに保持される。述語値の実際の解明時に、述語予測が適正であるとわかった場合、余分なアクションは取られず、プロセッサ・バッファに保持された予測され且つ選定されなかったパスからの命令が廃棄される。予測が不適正であるとわかった場合、すべての予測され且つ選定された命令およびディスパッチされたその後のいずれの命令も廃棄される（即ち、予測され且つ選定された命令およびその後のいずれの命令もアーキテクチャ上の状態を更新することを許されない）。しかる後、ルーチン処理は、プロセッサ・バッファに保持された予測され且つ選定されなかった命令でもって再開する。少なくともいくつかの実施態様では、述語付きコード領域から命令を再フェッチする必要を回避するために、ロジックを使用することも可能である。

【００１１】
一般に、消費命令がそれのオペランドを得なければならないロケーションを、即ち、レジスタ・ファイルまたはバイパス・ロジックから正確に指定することが、プロセッサの適正なオペレーションには必要である。更に、アウト・オブ・オーダ実行は、一般に、そのオペランドを使い尽くすことを待っているすべての命令をウェイク・アップするために、レジスタ・オペランドの生成を明瞭に識別する技術を必要とする。残念ながら、叙述化（predication）は条件付きの書込み機能を導入し、それ自体で、アウト・オブ・オーダ・プロセッサにおけるウェイク・アップ・プロセスおよびバイパス・ロジックを混乱させる。本発明の種々の実施態様によれば、述語付きコード領域（選定されたパスおよび選定されなかったパスに関連した述語付き命令を含む）の処理は、対応する述語が解明されるまでストールされる。述語の解明時に、レジスタ・リネーミングが適正なパス（即ち、選定されたパス）に関連した述語付き命令に関して遂行され、不適正なパス（即ち、選定されなかったパス）に関連した述語付き命令は廃棄される。

【００１２】
本明細書において開示された技術は、ブランチ・パイプラインの実行ステージおよびブランチ・パイプラインのリネーム・ステージの間にデータ・パスを加えることによって具現化される。一般に、レジスタ・リネーム・ステージは、述語付きコード領域を認識し、解明された述語をブランチ・パイプラインの実行ステージから受け取り、選定されなかったパスに関連した述語付き命令を廃棄するように修正される。本実施例で使用されるように、述語定義命令は、述語の値をセットする命令を含み、その値に基づいてその後の命令（述語付きコード領域に含まれる）の実行が述語付きにされる。例えば、インテルIA-64アーキテクチャのような明確な叙述化のサポートを備えた命令セット・アーキテクチャ（ＩＳＡ）では、比較命令が述語定義命令である。

【００１３】
本発明の１つの実施態様によれば、アウト・オブ・オーダ・プロセッサにおいて述語付きコードを扱うための技術は、述語付きコード領域に関連した述語定義命令を検出することを含む。述語付きコード領域内の述語付き命令のリネーミングは、述語定義命令の述語が解明されるまでストールされる。

【００１４】
本発明のもう１つの実施態様によれば、アウト・オブ・オーダ・プロセッサはデコーダおよびパイプライン・ステージを含む。デコーダは、述語付き命令を含む述語付きコード領域に関連した述語定義命令をデコードするように構成される。パイプライン・ステージは、述語定義命令を検出するように、および、述語定義命令の述語が解明されるまで述語付き命令のリネーミングをストールするように構成される。パイプライン・ステージは、述語の解明時に、選定されなかったパスに関連する述語付き命令を廃棄するように、または、選定されなかったパスに関連する述語付き命令を、アーキテクチャ状態を変更しないノー・オペレーション（ＮＯＰ）命令に変換するようにも構成され得る。

【００１５】
本発明の1つの実施例によれば、プロセッサ・システムはメモリ・サブシステムおよびアウト・オブ・オーダ・プロセッサを含み、アウト・オブ・オーダ・プロセッサはメモリ・サブシステムに接続されたデコーダおよびパイプライン・ステージを含む。デコーダは、述語付き命令を含む述語付きコード領域に関連した述語定義命令をデコードするように構成される。パイプライン・ステージは、述語定義命令を検出するように、および述語定義命令の述語が解明されるまで述語付き命令のリネーミングをストールするように、構成される。パイプライン・ステージは、述語の解明時に、選定されなかったパスに関連する述語付き命令を廃棄するように、または、選定されなかったパスに関連する述語付き命令を、アーキテクチャ状態を変更しないノー・オペレーション（ＮＯＰ）命令に変換するようにも構成され得る。

【００１６】
本明細書に開示された技術は、間接的な叙述化サポート（ARM Thumbアーキテクチャのような）を備えたアーキテクチャと同様に明示的な叙述化サポートを備えたＩＳＡに等しく適用可能である。１つの例として、既存のＩＳＡは、Thumbアーキテクチャのif-then命令と同様の述語定義命令（例えばemask命令）によって拡大し得る（例えば、米国特許第7,178,011号参照）。叙述化を含まない例示的な擬似コードは下記のように記述される。

```
 isequal cr7, 0, r10
branch cr7, ELSE
 sub r1=r1, r2
 load r4=[r1]
 jump MERGE
ELSE: add r1=r1, r2
 load r3=[r1]
 add r5=r3, r4
MERGE: store r５, [r１]
```

【００１７】
上記の擬似コードは、次のような述語付きコードと置換することが可能である。

```
isequal cr7, 0, r10
emaskcr7, 11001b, 5
add r1=r1, r2
load r3=[r1]
sub r1=r1, r2
load r4=[r1]
add r5=r3, r4
MERGE: store r5, [r1]
```

【００１８】
上記述語付きコードでは、emask命令が叙述化セマンティックスを提供する責任を負う。特に、emask命令は、１番目、２番目、および５番目の述語付き命令が第１パスに属し、３番目および４番目の述語付き命令が第２パスに属することによって、５つの後続する命令（述語付きコード領域内に含まれる）の実行を制御しなければならない実行マスクを指定する。戻された述語値（状態レジスタ７（ｃｒ７））に従って、第１パスの述語付き命令または第２パスの述語付き命令のいずれかが実行され、逆に、第２パスの述語付き命令または第１パスの述語付き命令のいずれかが廃棄されるか、ノー・オペレーション（ＮＯＰ）命令に変換される。

【００１９】
図１を参照すると、２つのチップ・レベル・マルチプロセッサ（ＣＭＰ）１０２を含む例示のプロセッサ・システム１００が示され、各マルチプロセッサ１０２は２つのプロセッサ１０４を含む。少なくとも１つの実施例では、プロセッサ１０４の各々は第１レベル（Ｌ１）キャッシュ・メモリ（個別に示されていない）を含み、そのＬ１キャッシュは共

有第２レベル（Ｌ２）キャッシュ・メモリ（キャッシュ）１０６に接続される。Ｌ２キャッシュ１０６は、第３レベル（Ｌ３）キャッシュ１１４およびファブリック・コントローラー１０８に接続される。ファブリック・コントローラー１０８は、メモリ・サブシステム１１２に接続されたメモリ・コントローラ１１０に接続される。メモリ・サブシステム１１２は、アプリケーションに適した量の揮発性および不揮発性メモリを含む。ファブリック・コントローラー１０８は、種々のＣＭＰ１０２の間およびプロセッサ１０４とメモリ・サブシステム１１２との間のコミュニケーションを容易にし、この場合、インターフェースとして機能する。

【００２０】

図２を参照すると、本発明に従って構成されるプロセッサ１０４の１つの例示的な命令パイプライン２００が示される。プロセッサ１０４は、例えば、同時的マルチスレッディング（ＳＭＴ）モードまたはシングル・スレッド（ＳＴ）モードで作動し得る。プロセッサ１０４がＳＭＴモードで作動しているとき、プロセッサ１０４は、複数のスレッドのためのプログラム・カウンタを格納するために、複数の個別の命令フェッチ・アドレス・レジスタを使用し得る。この場合、命令フェッチ（ＩＦ）ステージ２０２における命令フェッチは複数のスレッドの間で交互になる。ＳＴモードでは、一般に、プロセッサ１０４は１つのプログラム・カウンタしか使用せず、各プロセッサ・サイクル当たり単一のスレッドのための命令をフェッチする。１つの実施例では、プロセッサ１０４は、（命令キャッシュ（ＩＣ）ステージ２０４において）各プロセッサ・サイクル当たり８個までの命令を命令キャッシュからフェッチするように構成されてもよい。ＳＭＴモードでは、複数の（例えば２個の）スレッドが命令キャッシュおよび命令変換機構を共用し得る。

【００２１】

１つの所与のプロセッサ・サイクルでは、一般に、すべてのフェッチされた命令が同じスレッドに関連付けられる。１つの実施例では、プロセッサ１０４は、ブランチ予測（ＢＰ）ステージ２０６においてブランチに関するフェッチされた命令をスキャンする。ブランチがＢＰステージ２０６において検出される場合、プロセッサ１０４は、例えば、２つのスレッドにより共用される３つのブランチ履歴テーブル（ＢＨＴ）を使用してブランチ方向を予測し得る。１つの実施例では、ＢＨＴのうちの２つが、ブランチ方向を予測するために２モードの且つパス相関のブランチ予測機構を使用する。適正な方向を予測するためにどの予測機構がより適当であるかを予測するように第３のＢＨＴが構成され得る。フェッチされた命令が複数のブランチを含む場合、ＢＰステージ２０６はすべてのブランチを同時に予測するように構成されてもよい。ブランチ方向の予測に加えて、ＢＰステージ２０６は現命令グループ・サイクルにおいて選択されたブランチのターゲットを予測するようにも構成されてもよい。プロセッサ１０４は、サブルーチン・リターンのターゲットを予測するためのリターン・スタックを、例えば、各スレッドに対して１つのリターン・スタックを使用し得る。

【００２２】

プロセッサ１０４は、他のブランチのターゲットを予測するために共用ターゲット・キャッシュを使用し得る。選択されたブランチが存在する場合、プロセッサ１０４は、ブランチのターゲット・アドレスをプログラム・カウンタにロードするように構成される。選択されたブランチが存在しない場合、プロセッサ１０４は、次の順次命令のアドレスをプログラム・カウンタにロードするように構成される。プロセッサ１０４は、第１デコード（Ｄ０）ステージ２０８において、予測されたパスにおけるフェッチされた命令を複数のスレッドに対する個別の命令フェッチ・キューに入れるように構成される。スレッド優先順位に基づいて、プロセッサ１０４は、命令フェッチ・キューの１つから命令を選び、デコード・ステージＤ１（２１０）、デコード・ステージＤ２（２１２）、およびデコード・ステージＤ３（２１４）において１つのグループを形成ように構成される。述語付き命令を識別するためのロジックが、例えば、デコード・ステージＤ１（２１０）およびデコード・ステージＤ２（２１２）において使用されてもよい。１つの実施例では、１つのグループ内のすべての命令が同じスレッドから生じ、並行してデコードされる。プロセッサ

１０４は、命令のグループがディスパッチされるような資源が使用可能になるまで、命令を（転送（Xfer）ステージ２１６における）キュー内に入れる。

【００２３】

１つの実施例では、各ディスパッチされた命令グループは、グローバル完了テーブル（ＧＣＴ）におけるエントリに対応し、そのグループ内の各命令は、適切な発行キュー内のエントリに対応する。各ロードおよびストア命令は、アウト・オブ・オーダ実行ハザードを検知するために、ロード再順序付けキューおよびストア再順序付けキュー内のエントリをそれぞれ占有する。ディスパッチに必要なすべての資源が命令グループにおいて使用可能になるとき、その命令グループはグループ・ディスパッチ（ＧＤ）ステージ２１８においてディスパッチされる。命令は、ＩＦステージ２０２とＧＤステージ２１８との間のパイプライン・ステージをプログラム順に流れる。ディスパッチの後、各命令は、命令における論理的なレジスタ番号を物理的なレジスタにマップするリネーミング・ステージ（またはマッピング（ＭＰ）ステージ）２２８を流れる。

【００２４】

本発明の種々の実施態様によれば、述語付き命令のためのレジスタ・リネーミングは、述語定義命令に関連した述語が解明されるまで（即ち、述語定義命令が実行され、述語が解明されるまで）、ＭＰステージ２２８においてストールされる。１つの実施例では、ＭＰステージ２２８は、述語定義命令に関連したマスクを格納するマスク・シフト・レジスタ（個別的に示されていない）を使用する。述語が解明され、ＭＰステージ２２８に戻されるとき、どの述語付き命令を実行しなければならないか決めるために述語およびマスクに関して論理的機能（例えば、排他的ＮＯＲ）が遂行される。レジスタ・リネーミングは、実行されるべき述語付き命令（即ち、選択されたパスに沿った述語付き命令）に関して遂行され、実行されるべきではない述語付き命令（即ち、選択されなかったパスに沿った述語付き命令）は廃棄される。

【００２５】

１つの実施例では、プロセッサ１０４は、１２０個の物理的汎用レジスタ（ＧＰＲ）および１２０の物理的浮動小数点レジスタ（ＦＰＲ）を実装している。複数のスレッドが実行されるとき、プロセッサ１０４は、それら複数のスレッドがレジスタ・ファイルを動的に共用するように構成される。ＳＴモードにおけるアウト・オブ・オーダ・プロセッサ処理を容易にするために、プロセッサ１０４は、すべての物理的レジスタを単一スレッドにとって利用可能にして、より高い命令レベル並列度を促進するように構成される。レジスタ・リネーミングの後、命令は、複数のスレッドによって共用される発行キューに入る。プロセッサ１０４は複数の発行キューを使用し得る。浮動小数点発行キューは、１つまたは複数の浮動小数点パイプライン（実行ユニット）２２６を入力するために使用され、ブランチ発行キューはブランチ・パイプライン（実行ユニット）２２０を入力するために使用され、状態レジスタ論理的キューは状態レジスタ論理演算パイプライン（実行ユニット）（図示されていない）を入力するために使用され、結合発行キューは１つまたは複数の固定小数点（整数）パイプライン（実行ユニット）に２２４および１つまたは複数のロード・ストア・パイプライン（実行ユニット）２２２を入力するために使用され得る。１つの実施例では、プロセッサ１０４は８個の実行ユニット（図２にはそれらのすべてが示されているわけではない）を含み、その各々は各サイクルで命令を実行することができる。

【００２６】

パイプライン２２０～２２６を通して命令を追跡するためのロジックを単純化するために、プロセッサ１０４は、命令をグループとして追跡するように構成されてもよい。この場合、ディスパッチされた命令の各グループは、ディスパッチ時にＧＣＴにおけるエントリを占有する。この場合、そのグループがコミットされるとき、エントリはＧＣＴから割振り解除される。ＧＣＴにおけるエントリはプログラム順であり且つ所与のスレッドからのものであるが、連続したエントリが異なるスレッドに属することも可能である。１つの命令に対する入力オペランドがすべて利用可能であるとき、それは発行の資格を有するようになる。発行ロジックは、発行キューにおける適格の命令の中から１つを選択し、発行

（ＩＳＳ）ステージにおける実行のためにそれを発行する。命令発行に関しては、通常、２つのスレッドからの命令の間に区別は存在しない。

【００２７】

命令は、それが発行されると、レジスタ・ファイル（ＲＦ）ステージにおいて１つまたは複数の関連入力物理的レジスタを読取り、実行（ＥＸ）ステージにおいて実行し、書き戻し（ＷＢ）ステージにおいて関連出力物理的レジスタに結果を書き戻す。１つの実施例では、各浮動小数点パイプライン２２６が６サイクル実行パイプ（Ｆ１乃至Ｆ６ステージ）を有する。各ロード・ストア・パイプライン２２２において、加算器が、計算アドレス（ＥＡ）ステージにおいて読取るまたは書込むべきアドレスを計算し、データ・キャッシュがデータ・キャッシュ（ＤＣ）ステージにおいてアクセスされる。ロード命令に関しては、一旦データが戻されると、フォーマッタが、フォーマット（Ｆｍｔ）ステージにおいてキャッシュ・ラインから適正なバイトを選び、しかる後、ＷＢステージにおいてデータをレジスタに書込む。グループにおけるすべての命令が（例外を生じることなく）実行されてしまい且つそのグループが所与のスレッドのうちの最も古いグループであるとき、そのグループは完了（ＣＰ）ステージ２３２においてアーキテクチャ的状態にコミットされる。１つの実施例では、２つの命令グループがサイクル毎に、即ち、２つのスレッドの各々から１つをコミットし得る。１つの実施例では、プロセッサ１０４は、６４ＫバイトのＬ１命令キャッシュおよび３２Ｋバイトのデータ・キャッシュ（個別に示されてなく、それぞれ、２ウェイおよび４ウェイ関連性有する）を含む。或る実施例では、第１レベルのデータ変換テーブル（個別に示されていない）が完全に連想的なものであり、１２８個のエントリを含んでいる。

【００２８】

図３を参照すると、本発明の種々の実施態様に従って、アウト・オブ・オーダ・プロセッサにおいて述語付きコードを処理する例示的プロセス３００が示される。ブロック３０２において、プロセス３００が開始され、その時点で、制御が判断ブロック３０４に移る。ブロック３０４では、述語定義命令（既知の操作コード（ＯＰコード）に相当する）が、例えば、プロセッサ・パイプライン２００のＭＰステージ２２８において検出されない場合、制御はブロック３１６に移る。ブロック３０４において、述語定義命令が検出される場合、制御はブロック３０６に移る。ブロック３０６では、述語定義命令に関連したマスクが、その後の使用のためにマスク・シフト・レジスタ（例えば、ＭＰステージ２２８に含まれる）に格納される。マスクは、述語付き命令の各々が２つの代替パス（即ち、選択されたパスまたは選択されなかったパス）のどちらに属するかを表わす。しかる後、判断ブロック３０８において、その述語定義命令に関連した述語値が使用可能であるかどうかが決定される。

【００２９】

図２に示されるように、述語が、ブランチ・パイプライン２２０のＥＸステージの出力からＭＰステージ２２８に返送される。ブロック３０８において述語定義命令の述語が利用可能でない場合、制御はブロック３１０に移り、そこでは、述語付き命令のリネーミングがストールされる。ブロック３１０に続いて、制御はブロック３０８に移る。ブロック３０８において、述語定義命令の述語が使用可能である場合、制御はブロック３１２に移る。上述のように、述語定義命令を実行した結果 （即ち、状態レジスタの値）が、ブランチ・パイプライン２２０のＥＸステージの出力からＭＰ（リネーム）ステージ２２８に返送される。ブロック３１２において、後続の述語付き命令のうちのどれがリネームされ且つ実行されるべきか、およびその述語付き命令のうちのどれが廃棄されまたはＮＯＰ命令に変換されるべきかを決定するために、マスクおよび述語に関して論理的機能（例えば、排他的ＮＯＲ機能）が遂行される。次に、ブロック３１４において、（その論理的機能に従って）リネームされると思われる述語付き命令がリネームされ、（その論理的機能に従って）リネームされると思われない述語付き命令が廃棄されるかまたはＮＯＰ命令に変換される。しかる後、リネームされた命令が実行されてもよい。ブロック３１４に続いて、制御はプロセス３１６に移り、そこでは次の述語定義命令が検出されるまでプロセスは

停止する。

【００３０】

本明細書において開示された技術は、プロセッサ・コアおよび関連の命令セット・アーキテクチャに対する最小の修正でもって具現化することが可能である。従って、アウト・オブ・オーダ・プロセッサにおける述語付きコードの効率的な処理を容易にする技術が開示された。

【００３１】

添付図面におけるフローチャートおよびブロック図は、本発明の種々の実施例に従って、システム、方法、およびコンピュータ・プログラムの可能な実施態様のアーキテクチャ、機能、およびオペレーションを示す。これに関して、フローチャートまたはブロック図における各ブロックはモジュール、セグメント、またはコード部分を表わし得る。そのコード部分は、指定された論理機能を具現化するための１つまたは複数の実行可能命令を含む。或る代替方法では、ブロックにおいて示された機能は、図面に示された順序を外れて生じ得るということに留意すべきである。例えば、連続して示された２つのブロックは、実は、実質的に同時に実行され得るし、或いは、関連する機能次第で、逆順に実行されてもよいことがある。更に、ブロック図および／またはフローチャートの各ブロック、並びにブロック図および／またはフローチャートにおけるブロックの組合せが、指定された機能または動作を遂行する特殊目的のハードウェア・ベースのシステム、或いは特殊目的のハードウェアおよびコンピュータ命令の組合せによって具現化され得ることも留意すべきであろう。

【００３２】

本明細書において使用される用語は、特定の実施例のみを説明することを目的とするものであり、発明を限定することを意図するものではない。

【００３３】

「特許請求の範囲」の各請求項におけ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、動作、および均等物は、他の請求項における他の要素と組み合わせてその機能を遂行するための任意の構造、材料、または動作を含むように意図される。本発明の説明は、図解および記述を目的として示されたが、網羅的であることまたは開示された形式に限定されることを意図したものではない。本発明の範囲および趣旨から逸脱することなく、多くの修正および変更が当業者には明らかであろう。実施例は、本発明の原理および実用的な応用例を最適に説明するために、および当業者が、特定の意図された用途に適するように様々な修正を含む様々な実施例に関しても本発明を理解することを可能にするために、選択および記述された。

【００３４】

従って、本発明を詳細に且つ好適な実施例を参照して説明したが、「特許請求の範囲」において定義された発明の範囲を逸脱することなく、修正および変更が可能であることは明らかであろう。

【図面の簡単な説明】

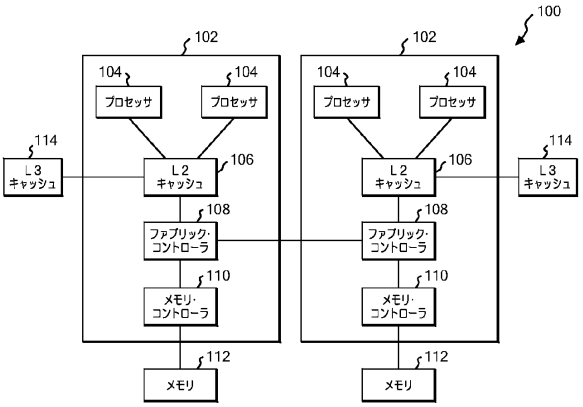
【００３５】

【図１】本発明の種々の実施態様に従って構成され得る例示的なプロセッサ・システムのブロック図である。

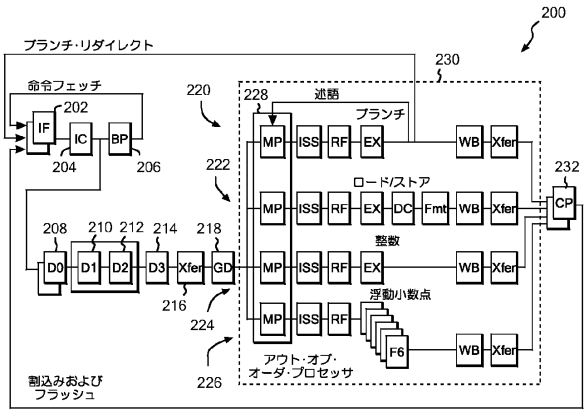
【図２】本発明の種々の実施例に従って、図１のプロセッサ・システムのプロセッサにおいて使用され得る例示的なプロセッサ命令パイプラインのブロック図である。

【図３】図２における例示的なプロセッサ命令パイプラインにおいて使用される例示的なプロセスのフローチャートである。

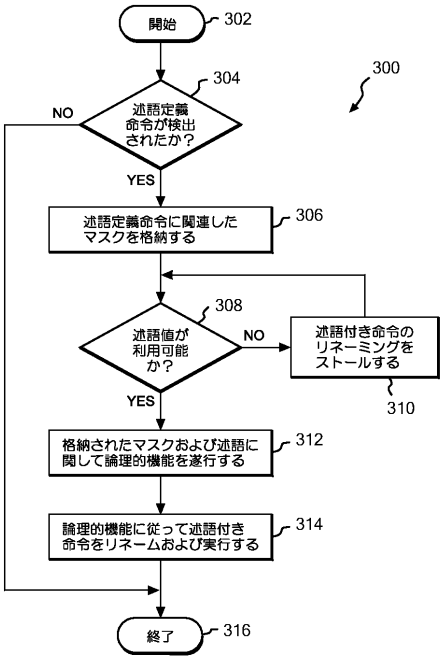
【図１】


100
102
102
104
104
104
104
プロセッサ
プロセッサ
プロセッサ
プロセッサ
114
L3
キャッシュ
L2
キャッシュ
106
L2
キャッシュ
106
114
L3
キャッシュ
108
ファブリック・
コントローラ
108
ファブリック・
コントローラ
110
メモリ・
コントローラ
110
メモリ・
コントローラ
112
メモリ
112
メモリ

【図２】


200
ブランチ・リダイレクト
230
命令フェッチ
220
228
述語
ブランチ
IF
202
IC
BP
204
206
222
MP
ISS
RF
EX
WB
Xfer
ロード/ストア
DC
Fmt
232
CP
208
210
212
214
218
D0
D1
D2
D3
Xfer
GD
整数
216
224
浮動小数点
F6
割込みおよび
フラッシュ
226
アウト・オブ・
オーダ・プロセッサ

【図３】


300
開始
302
304
述語定義
命令が検出
されたか？
NO
YES
306
述語定義命令に関連した
マスクを格納する
308
述語値が
利用可能
か？
NO
述語付き命令の
リネーミングを
ストールする
310
YES
312
格納されたマスクおよび述語に
関して論理的機能を遂行する
314
論理的機能に従って述語付き
命令をリネームおよび実行する
終了
316

フロントページの続き

(72)発明者 ラム・ランガン
アメリカ合衆国７８７５１、テキサス州オースチン、アベニュー ビー ４３１０ ナンバー２０１
(72)発明者 ウィリアム・エヴァン・スパイト
アメリカ合衆国７８７３３、テキサス州オースチン、パディナ・ドライブ ２４０７
(72)発明者 マーク・ダブリュー・スティーブンソン
アメリカ合衆国７８７５９、テキサス州オースチン、シエラ・オークス １０９２３
(72)発明者 リキシン・ジャン
アメリカ合衆国７８７２６、テキサス州オースチン、ダーク・スター １００００８

審査官 篠塚 隆

(56)参考文献 特表２００６－５２６８１３（ＪＰ，Ａ）
特開平２－８７２２９（ＪＰ，Ａ）
特開平１０－２８３１８５（ＪＰ，Ａ）
Perry H. Wang et al., Register Renaming and Scheduling for Dynamic Execution of Predicated Code, The Seventh International Symposium on High-Performance Computer Architecture, 2001. HPCA. , IEEE, ２００１年, p.15-25
中森 章, コンピュータアーキテクチャ その直感的アプローチ（４） 並列処理の基本, スーパースカラ, Ｏｈ！Ｘ, ソフトバンクパブリッシング株式会社, ２００１年 3月 1日, ２００１春号, ｐ．２５６－２７５

(58)調査した分野(Int.Cl., ＤＢ名)
Ｇ０６Ｆ９／３０
９／３８