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

特許第5301554号
(P5301554)

(45)発行日　平成25年9月25日(2013.9.25)　(24)登録日　平成25年6月28日(2013.6.28)

(51)Int.Cl.　　F I
G06F 9/42 (2006.01)　　G06F 9/42 320A
G06F 9/38 (2006.01)　　G06F 9/42 330A
　　G06F 9/38 330F

請求項の数 25　(全 22 頁)

(21)出願番号　特願2010-533174 (P2010-533174)
(86)(22)出願日　平成20年10月31日 (2008.10.31)
(65)公表番号　特表2011-503718 (P2011-503718A)
(43)公表日　平成23年1月27日 (2011.1.27)
(86)国際出願番号　PCT/US2008/081947
(87)国際公開番号　WO2009/059100
(87)国際公開日　平成21年5月7日 (2009.5.7)
審査請求日　平成22年6月23日 (2010.6.23)
(31)優先権主張番号　11/934,264
(32)優先日　平成19年11月2日 (2007.11.2)
(33)優先権主張国　米国 (US)

(73)特許権者　595020643
クゥアルコム・インコーポレイテッド
ＱＵＡＬＣＯＭＭ　ＩＮＣＯＲＰＯＲＡＴＥＤ
アメリカ合衆国、カリフォルニア州　９２１２１－１７１４、サン・ディエゴ、モアハウス・ドライブ　５７７５
(74)代理人　100108855
弁理士　蔵田　昌俊
(74)代理人　100091351
弁理士　河野　哲
(74)代理人　100088683
弁理士　中村　誠
(74)代理人　100109830
弁理士　福原　淑弘

最終頁に続く

(54)【発明の名称】プロシージャリターンシーケンスを加速するための方法およびシステム

(57)【特許請求の範囲】
【請求項１】
パイプラインプロセッサの中のプロシージャから戻るときにリンクスタックからリターンアドレスを取り出すための方法であって、
ソフトウェアスタックから情報を取り出すための検索命令を識別することと、
前記検索命令に基づいて取り出された情報を分岐先アドレスとして用いて前記リターンアドレスへと分岐するためのブランチ命令を識別することと、
前記検索命令と前記ブランチ命令を識別することに応じて、前記リンクスタックから前記リターンアドレスを取り出すことと、
前記リターンアドレスを使用して第２の命令をフェッチすることと、
を備える方法。
【請求項２】
前記検索命令は、ＰＯＰ命令である、請求項１に記載の方法。
【請求項３】
前記検索命令は、ロード命令である、請求項１に記載の方法。
【請求項４】
前記ブランチ命令は、ＢＸ命令である、請求項１に記載の方法。
【請求項５】
前記ブランチ命令は、ＭＯＶ命令である、請求項１に記載の方法。
【請求項６】

前記の前記検索命令を識別することは、前記情報を格納するレジスタを識別することをさらに備える、請求項１に記載の方法。
【請求項７】
レジスタリストを保持することをさらに備え、前記レジスタリストは、前記情報を格納するレジスタを識別するためのものである、請求項１に記載の方法。
【請求項８】
前記レジスタリストを保持することは、前記レジスタリストで識別された第２のレジスタの内容が上書きされたことを決定すること、および、
前記レジスタリストから前記第２のレジスタを取り除くこと
を備える、請求項７に記載の方法。
【請求項９】
前記ブランチ命令を識別することは、検出ロジック回路によって実行される、請求項１に記載の方法。
【請求項１０】
前記検出ロジック回路は、プリデコードロジック回路に含まれる、請求項９に記載の方法。
【請求項１１】
前記検出ロジック回路は、デコードロジック回路に含まれる、請求項９に記載の方法。
【請求項１２】
前記ブランチ命令を識別することは、命令キャッシュの中の前記ブランチ命令にフラグ付けすることをさらに備える、請求項１に記載の方法。
【請求項１３】
命令キャッシュに結合されたラインバッファであって、前記ラインバッファから前記命令キャッシュへ命令がロードされるものである、ラインバッファと；
前記命令キャッシュに結合され、予測リターンアドレスを記憶するリンクスタックを有するフェッチロジック回路とであって、前記命令キャッシュから命令を取り出すためのフェッチロジック回路と；
前記ラインバッファと通信するプリデコードロジック回路とであって、前記プリデコードロジック回路は、プロシージャリターンシーケンスを識別するための検出ロジック回路をさらに備えるものであり、前記プロシージャリターンシーケンスは、ソフトウェアスタックから情報を取り出すための検索命令と、前記検索命令に基づいて取り出された情報を分岐先アドレスとして用いて分岐するためのブランチ命令とを備えるものである、プリデコードロジック回路と；
前記プロシージャリターンシーケンスの前記識別に応じて前記リンクスタックから前記予測リターンアドレスの１つの予測されたリターンアドレスを取り出す手段と；
を備えるパイプラインプロセッサ。
【請求項１４】
前記検出ロジック回路は、前記ブランチ命令が、前記ラインバッファから前記命令キャッシュへとロードされるときに、前記プロシージャリターンシーケンスの前記ブランチ命令にフラグ付けする、請求項１３に記載のパイプラインプロセッサ。
【請求項１５】
前記フェッチロジック回路は、前記ブランチ命令から前記プロシージャリターンシーケンスを識別する、請求項１４に記載のパイプラインプロセッサ。
【請求項１６】
前記フェッチロジック回路内のリターンセレクタロジック回路をさらに備え、前記リターンセレクタロジック回路は、前記ブランチ命令から前記プロシージャリターンシーケンスを識別する、請求項１５に記載のパイプラインプロセッサ。
【請求項１７】
前記検索命令は、ＰＯＰ命令である、請求項１３に記載のパイプラインプロセッサ。
【請求項１８】

前記検索命令は、ロード命令である、請求項１３に記載のパイプラインプロセッサ。
【請求項１９】
前記ブランチ命令は、ＢＸ命令である、請求項１３に記載のパイプラインプロセッサ。
【請求項２０】
予測されたリターンアドレスを記憶するリンクスタックを有し、命令キャッシュから命令をフェッチするように構成されたフェッチロジック回路と、
前記フェッチロジック回路に結合されたデコードロジック回路と、
を備え、前記フェッチされた命令は、前記デコードロジック回路によって復号可能であり、前記デコードロジック回路は、検出ロジック回路をさらに備え、前記検出ロジック回路は、ソフトウェアスタックから情報を取り出すための検索命令と、前記検索命令に基づいて取り出された情報を分岐先アドレスとして用いて分岐するためのブランチ命令とを備えるプロシージャリターンシーケンスを識別するように構成され、パイプラインプロセッサは、前記プロシージャリターンシーケンスの前記識別に応じて前記リンクスタックから前記予測されたリターンアドレスのうちの１つの予測されたリターンアドレスを取り出すように構成される、パイプラインプロセッサ。
【請求項２１】
前記フェッチロジック回路は、前記リンクスタックから取り出された前記予測されたリターンアドレスを使用して命令をフェッチする、請求項２０に記載のパイプラインプロセッサ。
【請求項２２】
前記検索命令は、ＰＯＰ命令である、請求項２０に記載のパイプラインプロセッサ。
【請求項２３】
前記検索命令は、ロード命令である、請求項２０に記載のパイプラインプロセッサ。
【請求項２４】
前記ブランチ命令は、前記検索命令に基づいて識別されるアドレスへと分岐するためのものである、請求項２０に記載のパイプラインプロセッサ。
【請求項２５】
前記ブランチ命令は、移動命令である、請求項２０に記載のパイプラインプロセッサ。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、一般にコンピュータシステムに関し、そしてより詳細にはプロセッサ内のポップブランチ命令シーケンスを識別することにより、リターンシーケンスを加速するための方法およびシステムに関する。
【背景技術】
【０００２】
プロセッサによって実行される大部分のプログラムは、サブルーチンまたはプロシージャを含んでいる。プロシージャは、プロシージャ呼び出しシーケンスによってアクセスされるコードのモジュールである。ひとたびプロシージャが、完了された後には、命令実行は、プロシージャリターンシーケンス(procedure return sequence)の実行によって呼び出し側(caller)に戻される。
【０００３】
いくつかのプロセッサアーキテクチャ内において、プロシージャのコールおよびリターンのシーケンスは、一連の命令へとコンパイルされることができる。例えば、プロシージャ呼び出しシーケンスは、ブランチおよびリンクの命令によって追随されるＰＵＳＨ命令から成ることができる。ＰＵＳＨ命令（単数または複数）は、プロシージャ内の命令によって使用されるパラメータをソフトウェアスタック(software stack)上に保存することができる。ＰＵＳＨ命令の後に、プロセッサは、ブランチおよびリンクの命令を実行することができる。ブランチおよびリンクの命令は、命令のフェッチおよび実行が、プロシージャの開始アドレスにおいて開始するようにさせ、そしてリターンアドレスまたはリンクア

ドレスとして知られている、ブランチおよびリンクの命令に続く次の逐次命令のアドレスをリンクレジスタ(link register)に保存する。リンクレジスタは、プロセッサによって使用される専用レジスタ、あるいは汎用レジスタ(general purpose registers)（ＧＰＲ）のうちの１つとすることができる。プロシージャ内において、リンクレジスタ内容は、一般的にソフトウェアスタック上へとプッシュされ、その結果、その値は、別のプロシージャが、元の呼び出し側に戻る前に呼び出される場合には、上書きされないようになる。

【０００４】

プロシージャがそのファンクションを完了した後に、プロセッサは、リンクアドレス（プロシージャ呼び出し命令に続く次の逐次命令アドレス）において命令実行を再開するためにプロシージャリターンシーケンスを実行する。リターンアドレスは、多くの場合にソフトウェアスタック上に保存されるので、プロシージャリターンシーケンスは、最初に、フェッチされるべき次のグループの命令を決定するそのアドレスを使用するために、リターンアドレスをソフトウェアスタックから取り出す必要がある。

【０００５】

プロシージャリターンシーケンスは、１つまたは複数の命令から成ることができる。いくつかのプロセッサアーキテクチャにおいては、プロシージャリターンシーケンスは、次のリターンアドレスをソフトウェアスタックから読み取り、そしてプログラムカウンタ(program counter)（ＰＣ）をアップデートすることができる、ＰＯＰ命令やロード命令などの単一命令とすることができる。あるいは、プロセッサは、プロシージャリターンシーケンスを完了するためにその値をプログラムカウンタへと移動する前に、ソフトウェアスタックからＧＰＲなどの中間レジスタへとリンクアドレスを読み取るためにＰＯＰ命令またはロード命令を使用することもできる。他の例示的な例においては、プロセッサは、プロシージャからのリターンが、リンクレジスタ(link register)（ＬＲ）に保存される値をＰＣへと移動する命令とすることができることを決定することができる。プロセッサが、プロシージャコールの後にこれらのプロシージャリターンシーケンスのうちのどれかに出合うときに、プロセッサは、ソフトウェアスタックから取り出されるリターンアドレス値を使用してプロシージャ呼び出し命令に続く次の逐次命令へと後方にジャンプする。

【０００６】

追加のロジックは、命令処理の効率を改善するためにプロセッサのハードウェアに追加されることができる。例えば、リンクスタックは、命令フェッチを高速化するためにプロセッサのフェッチロジックに追加されることができる。当業者は、リンクスタックが、ソフトウェアスタック上にやはり存在することもできるリターンアドレスを含むことができることを認識する。しかしながら、リンクスタックは、ソフトウェアスタックとは独立して動作する。リンクスタックに関連するハードウェアロジックは、プロシージャのコールとリターンとを識別する。プロシージャコール命令が、実行に先立って識別されるときに、関連するリターンアドレスは、リンクスタック上へとロードされる。逆に、プロシージャリターンが、識別されるときには、関連するリターンアドレスは、リンクスタックから取り出され、そして命令フェッチを再開するために使用される。実行すべき命令を待つことと、ソフトウェアスタックからのリターンアドレスを取り出すこととの代わりに、プロセッサは、リンクスタックに記憶されるアドレスを使用して推測的に(speculatively)命令をフェッチすることができる。

【０００７】

プロセッサが進化するにつれて、プロシージャリターンシーケンスは、変化し続けている。いくつかのプロセッサアーキテクチャにおいては、プロシージャリターンは、複数の命令から成ることができる。リンクスタックをサポートするハードウェアロジックが、これらの命令をプロシージャリターンシーケンスとして認識しない場合、リターンアドレスは、リンクスタックから取り出されなくてもよく、そして結果としてリンクスタックは、命令シーケンスと同期が合わなくなる可能性がある。リンクスタックが同期が合わなくなるときに、リンクスタックは、複数のアドレス予測ミスを引き起こし得る誤ったリターンアドレス情報を提供する可能性がある。

【発明の概要】
【０００８】
したがって、ある種の命令シーケンス、より詳細にはＰＯＰ（またはロード）およびブランチの命令シーケンスをプロシージャリターンシーケンスとして認識するプロセッサ回路を有する必要性が、産業界において存在している。本開示は、この必要性を認識し、そして命令パイプラインの中で早期にプロシージャリターンに対応する命令を識別する回路を有するプロセッサを開示している。プロシージャリターンを識別した後に、プロセッサは、リンクスタックからの次のリターンアドレスを使用することにより次のグループの命令をフェッチする。ＰＯＰおよびブランチの命令シーケンスをプログラムリターンとして認識することにより、プロセッサは、リンクスタックから取り出される正しいアドレスに基づいて命令をフェッチすることを継続することができる。
【０００９】
パイプラインプロセッサにおいてプロシージャから戻るときにリンクスタックからリターンアドレスを取り出すための方法が、開示される。本方法は、リンクスタックからリターンアドレスを取り出すように動作する検索命令(retrieve instruction)を識別する。本方法は、リターンアドレスへと分岐するように動作するブランチ命令を識別する。本方法は、識別される命令とブランチ命令との両方に応じてリンクスタックからリターンアドレスを取り出す。本方法は、リターンアドレスを使用して後続の命令をフェッチする。
【００１０】
パイプラインプロセッサが、開示される。パイプラインプロセッサは、ラインバッファ(line buffer)を有する。ラインバッファは、命令キャッシュに結合される。プロセッサは、命令キャッシュに結合されるフェッチロジック回路も有する。フェッチロジック回路は、予測リターンアドレスを記憶するリンクスタックを有し、そこで命令は、ラインバッファから命令キャッシュへとロードされる。フェッチロジック回路は、命令キャッシュから命令を取り出す。パイプラインプロセッサは、ラインバッファと通信するプリデコードロジック回路(pre-decode logic circuitry)も有し、そこでプリデコードロジック回路は、プロシージャリターンシーケンスを識別するための検出ロジック回路を有する。プロシージャリターンシーケンスは、ソフトウェアスタックからリターンアドレスを取り出すように動作する検索命令と、取り出されたリターンアドレスへと分岐するブランチ命令として識別される。パイプラインプロセッサは、プロシージャリターンシーケンスの識別に応じてリンクスタックから予測されたリターンアドレスを取り出す。
【００１１】
パイプラインプロセッサが、開示される。パイプラインプロセッサは、フェッチロジック回路を有する。フェッチロジック回路は、予測されたリターンアドレスを記憶するリンクスタックを有する。フェッチロジック回路は、命令キャッシュから命令をフェッチする。パイプラインプロセッサは、フェッチロジック回路に結合されるデコードロジック回路も有し、そこでフェッチされた命令は、デコードロジック回路によって復号される。デコードロジック回路は、さらに検出ロジック回路を有し、そこで検出ロジック回路は、プロシージャリターンシーケンスを識別する。プロシージャリターンシーケンスは、ソフトウェアスタックからアドレスを取り出す検索命令と、取り出されたアドレスへと分岐するように動作するブランチ命令とである。パイプラインプロセッサは、プロシージャリターンシーケンスの識別に応じてリンクスタックから予測されたリターンアドレスを取り出す。パイプラインプロセッサは、プロシージャリターンの識別に応じてリンクスタックから予測されたリターンアドレスを取り出す。
【００１２】
本発明のより完全な理解、ならびに本発明のさらなる特徴および利点は、以下の詳細な説明と、添付の図面とから明らかであろう。
【図面の簡単な説明】
【００１３】
【図１】図１は、本発明の一実施形態を使用したプロセッサのハイレベルのロジックハー

ドウェアブロック図を示している。
【図２】図２は、図１のプロセッサによって実行される１つの例示のグループの命令を示している。
【図３】図３は、本発明の一実施形態に従って検出ロジック回路を組み込んだ、図１のＣＰＵの上位パイプラインと下位パイプラインとのより詳細なブロック図を示している。
【図４】図４は、図３のフェッチロジック回路のより詳細な図を示している。
【図５】図５は、検出ロジック回路を利用した上位パイプラインと下位パイプラインとの代替実施形態を示している。
【図６】図６は、プログラムリターンを認識して、そして命令をフェッチするためにリンクスタックを使用して、図１のプロセッサによって実行される命令プロセスフローを示すフローチャートを示している。
【図７】図７は、図４の上位パイプラインを使用してプロセッサによって実行される代替命令プロセスフローを示すフローチャートを示している。
【詳細な説明】
【００１４】
添付の図面に関連して以下で述べられる詳細な説明は、本発明の様々な例示の実施形態の説明として意図され、そして本発明が実行されることができる唯一の実施形態を表すように意図されてはいない。詳細な説明は、本発明の完全な理解を提供する目的のための特定の詳細を含んでいる。しかしながら、本発明が、これらの特定の詳細なしに実行されることができることは、当業者にとって明らかであろう。いくつかの例においては、よく知られている構造およびコンポーネントは、本発明の概念をあいまいにすることを回避するためにブロック図形式で示される。頭字語と、他の説明的な専門用語とは、単に便宜上、そして明快にするために使用されることができ、そして本発明の範囲を限定するように意図されてはいない。
【００１５】
図１は、以下に説明されるように本発明の一実施形態を利用したスーパースカラプロセッサ(superscalar processor)１００のハイレベル図を示している。プロセッサ１００は、制御信号１０４を経由して命令キャッシュ１０６に結合される中央演算処理装置(central processing unit)（ＣＰＵ）１０２を有する。命令キャッシュ１０６はまた、ラインバッファ１０７に、そして汎用バス１１０によってメモリ１０８にも結合される。ＣＰＵ１０２は、ラインバッファ１０７を経由してメモリ１０８から命令キャッシュ１０６への命令のローディング(loading)を制御する。ＣＰＵ１０２は、下位パイプライン１６０および１６５に結合された上位パイプライン１５０を有する。下位パイプライン１６０および１６５内には、実行ステージ２２０および２２５がある。実行ステージ２２０内には、実行ユニット(execution units)（ＥＵ）１３０Ａがあり、そして実行ステージ２２５内には、ＥＵ１３０Ｂがある。
【００１６】
当業者が理解するように、命令キャッシュ１０６は、メモリ１０８と、プロセッサ１００との間の速度ギャップを埋めるように設計された専用メモリとすることができる。メモリ１０８からフェッチされる命令は、プロセッサのクロック速度で読み取られることができるより高速な命令キャッシュ１０６に配置される。命令が、命令キャッシュ１０６の中に存在しない場合、プロセッサ１００は、メモリ１０８から命令を取り出す。命令が、メモリ１０８から取り出されるときに、それは、最初にラインバッファ１０７にロードされ、そして最終的には命令キャッシュ１０６へと書き込まれる。
【００１７】
命令キャッシュ１０６が、命令でロードされた後に、ＣＰＵ１０２は、制御信号１０４を経由してそれらにアクセスする。命令は、命令キャッシュ１０６から上位パイプライン１５０へとロードされる。命令は、上位パイプライン１５０の中で処理され、次いでさらなる処理のために下位パイプライン１６０または１６５へと送られる。図３～５の考察に関連して説明されるように、プロセッサは、特定の命令シーケンスを検出するように設計

されたロジック回路を有することができる。これらの特定の命令シーケンスは、プロシージャリターンに対応することができる。プロシージャリターン命令シーケンスが識別された後に、プロセッサ１００は、本発明の複数の実施形態に従ってこれらの命令に基づいてファンクションを実行することができる。

【００１８】

上位パイプライン１５０の中の命令上で実行されるいくつかの例示の処理ファンクションは、命令をフェッチすることと、命令を位置合わせすること(aligning)と、命令を復号することと、命令を下位パイプライン１６０または１６５に対して発行することなどとを含むことができる。下位パイプライン１６０および１６５内において、命令は、実行ユニット１３０Ａおよび１３０Ｂによって実行されることができ、それらの結果が、記録される。

【００１９】

ＰＯＰおよびブランチの命令シーケンスを使用したプロシージャリターンを有する実例のグループの命令２００が、図２に示される。命令２６０と、命令のオペレーション２７０と、命令を実行するモジュール２８０とが、示されている。明確にする目的のために、プロシージャそれら自体による使用のためにソフトウェアスタック上でパラメータをプッシュすることになるどのような命令も、このグループの命令２００から省略されている。プロシージャが実行する実際のファンクションを構成することになるどのような命令もまた、省略されている。図２に示される命令は、プロシージャを呼び出し、リターンアドレスをリンクレジスタ（この例においてはＧＰＲ　R_{14}）に保存し、リターンアドレスをソフトウェアスタック上に記憶し、ソフトウェアスタックからリターンアドレスを取り出し、そしてリターンアドレスに位置する命令を処理することを継続する命令である。グループの命令２００は、それらが命令実行のトレース中にそうであるようなプログラム順序で図２に示されている。当業者は、トレースされた命令が、プロセッサがフェッチしている可能性がある実際のコードのサブセットであり、それらが実行されるべきであるように示されることを理解する。グループの命令２００は、３つのネストされたプロシージャから成る。

【００２０】

グループの命令２００内には、３つのプロシージャコールと、それらの関連するリターンとがある。最初のプロシージャコールは、命令Ａであり、この命令は、プロシージャＰＲＯＣ１を呼び出す。命令Ｂは、プロシージャＰＲＯＣ１内の準備命令(preparatory instruction)であり、現在のリターンアドレスをソフトウェアスタック上へと保存する。命令Ｃは、第２のプロシージャコール命令であり、プロシージャＰＲＯＣ２を呼び出す。命令Ｄは、プロシージャＰＲＯＣ２内の別の準備命令であり、ＰＲＯＣ２に関連するリターンアドレスをソフトウェアスタック上へと保存する。最後のプロシージャコール命令は、命令Ｅであり、この命令は、プロシージャＰＲＯＣ３を呼び出す。

【００２１】

プロシージャコール命令に対応してプロシージャリターン命令がある。最初のプロシージャリターン命令は、命令Ｆである。以前のプロセッサアーキテクチャにおいては、命令Ｆは、プロシージャリターン命令として認識される。次の２つの命令、組み合わされた命令ＧおよびＨは、別のプロシージャリターンを表す。一般に、以前のプロセッサアーキテクチャにおいては、ＰＯＰ命令とブランチ命令との命令の組合せは、ハードウェアリンクスタックによる使用のためのプロシージャリターンとして適切に識別されない可能性がある。これらの以前のプロセッサにおける結果として、リンクスタック上の次のリターンアドレスは、命令ＧおよびＨが識別されるときに取り出されない可能性がある。一実施形態を使用したプロセッサは、この可能なリンクスタック破損を軽減することができる。一実施形態においては、命令Ｈが、プロシージャリターン命令として識別された後に、プロセッサ１００は、リンクスタックから次のアドレスを取り出し、そして命令をフェッチすることを継続するために取り出されたアドレスを使用することができる。この例においては、リンクスタック上の次のアドレスは、プロシージャＰＲＯＣ１を戻って指し示し、そし

てより詳細には、それは命令Ｃに続く次の逐次命令（命令Ｉ）を指し示す。命令Ｈは、暗黙ブランチ命令(implicit branch instruction)と称されることもできる。

【００２２】

次の２つの命令、命令ＩおよびＪはまた、プロシージャリターンシーケンスとしても解釈される。命令Ｊが、プロセッサ１００によってプロシージャリターン命令として識別されるときに、リンクスタック上の次のアドレスは、取り出され、そして命令フェッチを継続するために使用される。命令Ｊは、明示ブランチ命令(explicit branch instruction)である。この例においては、リンクスタックポイントを離れた次のアドレスは、プログラム実行を主プログラムへと逆に戻す。以前のプロセッサアーキテクチャにおいては、命令ＩとＪとの組合せは、ハードウェアリンクスタックによる使用のためのプロシージャリターンシーケンスとして適切に識別されていない可能性がある。図３～７の考察においてもっと詳細に説明されるように、本発明の様々な実施形態は、ＰＯＰとブランチとの命令の組合せをプロシージャリターンシーケンスとして識別する。

【００２３】

図３は、本発明の一実施形態を利用したＣＰＵ１０２のより詳細なブロック図を示している。ＣＰＵ１０２内において、上位パイプライン１５０は、制御信号１０４によって命令キャッシュ１０６に結合された、フェッチロジック回路２０２を含むフェッチステージ２０３を有する。またＣＰＵ１０２の中には、検出ロジック回路２５０を有するプリデコードロジック回路２０１がある。プリデコードロジック回路２０１は、命令キャッシュ１０６に結合されたラインバッファ１０７に結合される。フェッチステージ２０３は、順に発行ステージ２０７に結合されたデコードステージ２０５に結合される。デコードステージ２０５に結合されて、命令についての特有の情報を復号するデコードロジック回路（説明図を簡単にするために示されず）がある。発行ステージ２０７内には、下位パイプライン１６０および１６５に対して発行される命令に先立って命令を保持するいくつかの命令待ち行列（図示の容易のために図示せず）があってもよい。

【００２４】

当業者が理解しうるように、パイプラインステージは、命令を保持するように設計されたレジスタ、または１グループのレジスタを有することができる。命令が特定のステージに入ると、プロセッサ１００は、その命令をそのステージにリンクされたレジスタ、または１グループのレジスタにロードする。命令が、各ステージ内のレジスタまたは１グループのレジスタに保持されるときに、ロジック回路は、命令に応じてある種のオペレーションを実行することができる。ロジック回路が、意図されたオペレーションを実行した後に、次いで命令は、次の逐次ステージへと渡される。さらに、命令が、上位パイプライン１５０の中にある間、それらは、様々なロジック回路によって「処理され」る。命令を処理することは、命令をフェッチすることと、命令を復号することと、命令を位置合わせすることと、命令を発行することなどとを含むことができる。

【００２５】

命令は、上位パイプライン１５０に入り、そしてフェッチステージ２０３から発行ステージ２０７を通して移動する。命令は、フェッチステージ２０３中においてフェッチロジック回路２０２によってフェッチされる。命令がフェッチされた後に、それらは、デコードステージ２０５中においてデコードロジック回路によって復号される。デコードステージ２０５の後に、命令は、発行ステージ２０７の中で処理される。命令が、発行ステージ２０７を離れた後に、命令は、下位パイプライン１６０または下位パイプライン１６５のいずれかの中で実行される。上記に論じられるように、下位パイプライン１６０内には、実行ステージ２２０とＥＵ１３０Ａとがある。下位パイプライン１６５内には、実行ステージ２２５とＥＵ１３０Ｂとがある。下位パイプライン１６０および１６５は、それぞれレジスタファイル２３０または２３５にアクセスする。

【００２６】

プリデコードロジック回路２０１は、命令が命令キャッシュ１０６に保存されるのに先立って命令についての情報を部分的に復号し、そして識別するためにプロセッサ１００に

よって使用されることができる。プリデコードされた情報は、命令が命令キャッシュ１０６に記憶されるときに、命令と一緒に保存されることができる。プリデコードロジック回路２０１内において、検出ロジック回路２５０は、命令の間の相互依存性を識別することができる。例えば、検出ロジック回路２５０は、いつＰＯＰ命令とブランチ命令とが同じレジスタを利用するかを識別するように設計されることができる。図４の考察において説明されるように、検出ロジック回路２５０が、ＰＯＰ命令とブランチ命令とから成る命令シーケンスをプロシージャコールからのリターンとして識別した後に、フェッチロジック回路２０２は、ブランチ命令が命令キャッシュ１０６からフェッチされるときに、この情報を解釈する。

【００２７】

プリデコードされた情報を命令に関連づけることは、命令が命令キャッシュ１０６にロードされるときに、命令に関連する情報フィールド内の特定のロケーションの中の１ビットを設定することによって遂行されることができる。プリデコードされた情報を命令キャッシュ１０６に保存することはまた、命令にフラグ付けすること(flagging)と称されることもできる。例えば、命令がプロシージャリターン命令であることを決定した後に、命令がプロシージャリターン命令であることを識別する命令ヘッダの中の１つのロケーションの中で、１ビットが、設定されることができる。あるいは、プロセッサ１００は、プリデコードされた情報を識別された１つまたは複数の命令についての命令ヘッダへと符号化することもできる。このようにして、プロセッサ１００は、選択された、またはあらかじめ決定された判断基準に基づいて異なる命令についての異なる情報を符号化するために多ビットを使用することができる。プリデコードされた情報は、命令が命令キャッシュ１０６からフェッチされるときに、取り出されることができる。次いでプロセッサ１００は、識別された情報に基づいてある種のファンクションを実行することができる。

【００２８】

図４は、本発明の一実施形態によるフェッチロジック回路２０２を示している。フェッチロジック回路２０２は、アドレス選択ｍｕｘ（マルチプレクサ）３０２を制御するアドレスセレクタロジック回路３２０を含んでいる。アドレスセレクタロジック回路３２０は、リターンセレクタロジック回路３５０を含んでいる。アドレス選択ｍｕｘ３０２の入力に結合されて、リンクスタック３０４に由来するリンクスタック出力３１６がある。リンクスタックロジック回路３１０は、アドレスセレクタロジック回路３２０と通信し、そしてリンクスタック３０４の入力と出力との両方を制御する。リンクスタック３０４は、プロシージャコールが識別されるときに、アドレスバスからリターンアドレスを受け取る。

【００２９】

リンクスタック３０４内に、予測リターンアドレスは、保存されることができる。リンクスタック３０４は、プロシージャリターンに関連するリターンアドレスに対応する命令アドレスを記憶するメモリの後入れ先出し(last in first out)（ＬＩＦＯ）部分とすることができる。リンクスタック３０４は、ソフトウェアスタックとは独立に動作する。命令が、命令パイプラインの中で早期にプロシージャリターン命令として識別されるときに、プロセッサ１００は、下位パイプライン１６０または１６５の中で実行すべきプロシージャリターンを待つ代わりに、リンクスタック上に記憶されるリターンアドレスを使用して命令を先んじてフェッチすることができる。

【００３０】

図４に示されるように、アドレス選択ｍｕｘ３０２は、次の逐次プログラムアドレスを受け取ることができる。次の逐次プログラムアドレスは、８つのアドレスロケーションだけ増分された現在のプログラムカウンタ（ＰＣ＋８）とすることができる。本実施形態においては、命令は、各命令が４バイトの長さである場合の一度に２命令を命令キャッシュ１０６からフェッチされる。他のプロセッサ実施形態において、次の逐次プログラムアドレスは、異なる量だけ増分されたプログラムカウンタとすることができる。上述されるように、アドレス選択ｍｕｘ３０２は、リンクスタック３０４から予測アドレス情報を受け取ることもできる。プロセッサ１００が、プロシージャリターンが起こっていることを決

定するときに、リンクスタック３０４の中の次のアドレスは、取り出され、そして次のグループの命令をフェッチすべき開始ロケーションとして使用される。

【００３１】

アドレス選択ｍｕｘ３０２は、他のソースからアドレス情報を受け取ることができる。例えば、ブランチターゲットアドレスキャッシュ(branch target address cache)（ＢＴＡＣ）は、命令をフェッチするために使用されるアドレスを提供することができる。あるいは、割込みアドレス(interrupt address)が、命令をフェッチするために使用されることもできる。図示を容易にするために、アドレスのこれらの他のソースは、示されていない。

【００３２】

アドレスセレクタロジック回路３２０は、その入力のうちのどれが、アドレス選択ｍｕｘ３０２を通して渡され、そして次のグループの命令をフェッチするために使用されることになるかを決定する。アドレスセレクタロジック回路３２０が、フェッチされるべき次のグループのアドレスが次の逐次アドレス（ＰＣ＋８）であることを決定する場合、ＰＣ＋８の入力が、選択される。あるいは、アドレスセレクタロジック回路３２０内のリターンセレクタロジック回路３５０が、リンクスタック３０４が次のフェッチアドレスを含むことを決定する場合には、リンクスタック出力３１６が、選択される。

【００３３】

リンクスタック３０４を利用するために、プロセッサ１００は、いつプロシージャコールと対応するリターンとが、上位パイプライン１５０内の命令処理シーケンス中に識別されるかを決定する必要がある。リンクスタック３０４が、予測的に命令をフェッチするために使用されるので、プロセッサ１００は、後続の命令をフェッチする前に実行すべき命令を待つことはない。その代わりに、プロセッサ１００が、上位パイプライン１５０の中でプロシージャコール命令として識別した後に、プロセッサ１００は、プロシージャコールに関連するリターンアドレスをアドレスバスを経由してリンクスタック３０４へとロードする。次いで、プロセッサ１００は、プロシージャの命令をフェッチする。

【００３４】

プロシージャの終わりに、プロセッサ１００は、プロシージャリターンシーケンスに出合う。プロシージャリターンシーケンスの結果として、プロセッサは、対応するリターンアドレスを取り出し、そして命令フェッチを再開するそのリターンアドレスへと分岐するように、リンクスタック３０４を「ポップする(pop)」ことになる。プロセッサ１００は、プロシージャリターン命令を識別し、そしてリンクスタックから次のリターンアドレスを取り出す。プロシージャリターン命令は、ソフトウェアスタックを読み取り、そしてＰＣを書き込むＰＯＰ命令またはロード命令とすることができる。リターンセレクタロジック回路３５０が、特定のＰＯＰ命令がプロシージャリターンであることを識別する場合、そのときにはリターンセレクタロジック回路３５０は、アドレスセレクタロジック回路３２０に、リンクスタック出力３１６が、アドレス選択ｍｕｘ３０２を通して方向づけられるようにするようにさせる。次いで、リンクスタック３０４から取られるリターンアドレスは、次の組の命令をフェッチするために使用される。

【００３５】

上記に説明されたように、プロシージャリターンシーケンスは、１つまたは複数の命令から成ることができる。例えば、いくつかのＡＲＭインプリメンテーションにおいては、リンクレジスタ（R_{14}）に記憶される値に対するブランチ命令は、プロシージャリターンとして解釈されることができる。代わりに、リンクレジスタ（R_{14}）の値をプログラムカウンタ（R_{15}）へと移動する移動命令は、プロシージャリターンとして解釈されることもできる。プロセッサ１００が、正確にプロシージャリターンを識別することが、重要である。プロセッサ１００が、正確にプロシージャリターンを識別しない場合には、リンクスタック３０４は、プロシージャリターン命令に関して同期が合わなくなることになる。リンクスタック３０４が、同期が合わなくなる場合、プロセッサ１００は、ブランチ補正シーケンスへと進む必要がある可能性があり、そして実行性能は、影響を受ける可能

性がある。

【００３６】

プロセッサ命令セットが、進化しているので、代替命令シーケンスは、プロシージャリターンシーケンスとして識別されることができる。例示の一実施形態においては、特定のレジスタに記憶される値に対するブランチ命令によって追随される特定のレジスタに対するリターンアドレスをポップするＰＯＰ命令またはロード命令（ＰＣをアップデートしない）は、プロシージャリターンシーケンスとして解釈されることができる。ブランチ命令は、ＰＯＰ命令に続く次の逐次命令であってもよく、あるいはそうでなくてもよい。

【００３７】

ＰＯＰおよびブランチの命令から成るプロシージャリターンシーケンスの識別を容易にするために、両方の命令に関連した情報が、集められる。プロシージャリターンのＰＯＰ命令は、１つまたは複数のレジスタに関与する可能性がある。ＰＯＰ命令が識別されるときに、ＰＯＰ命令のレジスタリストは、保存され、そして任意の後続の命令のレジスタターゲットと比較されることができる。レジスタリストの保存することと、比較することとは、ＰＯＰ命令が、識別されていることを維持することと称されることもできる。非ブランチ命令が、そのレジスタに対するブランチが出合われる前に、ＰＯＰ命令に関連するレジスタリストの中で識別されるレジスタを利用する場合、そのレジスタは、保存済みのレジスタリスト(saved register list)から無視される(discounted)。保存済みのレジスタリストの中のレジスタを使用しないブランチ命令が、保存済みのレジスタリストの中のレジスタを使用するブランチ命令の前に出合われる場合、以前のＰＯＰについてのＰＯＰ－ブランチリターンシーケンスについての探索(search)は、終了される。レジスタリストの中のレジスタを使用するブランチ命令が、出合われるときに、次いでプロセッサ１００は、プロシージャリターンが、処理されていることを決定することができる。結果として、次いで、リンクスタック３０４の最上部におけるアドレスが、取り出され、そして次のグループの命令をフェッチするために使用されることができる。

【００３８】

前述のように、プリデコードロジック回路２０１（図３）は、同じレジスタを利用するＰＯＰおよびブランチの命令シーケンスを識別している可能性があり、そして結果として、ブランチ命令は、プロシージャリターン命令として識別される。プロセッサ１００は、ブランチ命令が、命令キャッシュ１０６に記憶されたときに、この情報を命令ヘッダに保存している可能性がある。フェッチロジック回路２０２が、ブランチ命令を用いて保存されたプリデコードされた情報を取り出すときに、プロセッサ１００は、ブランチ命令がプロシージャリターンであることを識別するためにリターンセレクタロジック回路３５０を使用する。リターンセレクタロジック回路３５０が、ブランチ命令がプロシージャリターンであることを決定した後に、リターンセレクタロジック回路３５０は、アドレス選択ロジック回路３２０が、アドレス選択ｍｕｘ３０２を通してリンクスタック出力３１６を方向づけるようにする。リターンセレクタロジック回路３５０はまた、リンクスタックの中の次の値が、戻されるようにするリンクスタックロジック回路３１０と通信する。結果として、リンクスタックアドレスは、次の組の命令をフェッチするために使用される。

【００３９】

図５は、ＰＯＰ／ブランチ命令シーケンスから成るプロシージャリターンを検出することができるデコードロジック回路を有する代替実施形態に従って、上位パイプライン１５１を有するＣＰＵ１０２を示している。より詳細には、ＣＰＵ１０２は、検出ロジック回路４５０を有するデコードロジック回路４０６を含んでいる。命令が、デコードロジック回路４０６によって復号されるので、命令に関連した情報は、識別される。検出ロジック回路４５０は、いつプロシージャリターンが識別されるかを決定するために復号された命令を監視することができる。以上で論じられるように、プロシージャリターンシーケンスは、１つまたは複数の命令から成る。検出ロジック回路４５０は、ＰＯＰ命令と後続のブランチ命令とが復号されるときに、プロシージャリターンシーケンスが起こることを決定することができる。

【００４０】
検出ロジック回路４５０が、プロシージャリターンが識別されていることを決定するときに、検出ロジック回路４５０は、この情報をリターンセレクタロジック回路３５０に対して伝え、このリターンセレクタロジック回路は、次にこの情報をリンクスタックロジック回路３１０に対して伝える（図４）。次いでリターンセレクタロジック回路３５０は、アドレスセレクタロジック回路３２０が、アドレス選択ｍｕｘ３０２を通してリンクスタック出力３１６を方向づけるようにする。次いで、リンクスタック３０４から取られるリターンアドレスは、次の組の命令をフェッチするために使用される。

【００４１】
実施形態に関連する発明の概念は、図２の中のグループの命令２００を戻って参照することにより、さらに説明されることができる。命令Ａは、プロシージャＰＲＯＣ１のコールである。命令Ａが、ＰＲＯＣ１へと分岐するときに、プロセッサ１００は、次の逐次アドレスをリンクレジスタ（R_{14}）に記憶する。次の逐次アドレスは、主プログラムに戻ることに関連するリターンアドレスである。命令Ａが、プロシージャコールとして識別されるときに、リンクスタックロジック回路３１０は、命令Ａに関連するリターンアドレスが、リンクスタック３０４へとロードされるようにする。図２に示されるように、命令Ａは、主プログラムの一部分である。命令Ａは、ＰＲＯＣ１へと分岐し、そして次の処理された命令は、命令Ｂである。

【００４２】
命令Ｂは、ＰＲＯＣ１内の最初の命令であり、そしてプロシージャＰＲＯＣ２のコールのための準備命令である。命令Ｂは、R_{14}の値をソフトウェアスタック上へとプッシュすることにより、現在のリターンアドレスを保存する。次に、命令Ｃが、処理される。命令Ｃは、プロシージャＰＲＯＣ２のコールである。命令Ｃが、プロシージャコールとして識別されるときに、リンクスタックロジック回路３１０は、命令Ｃに関連するリターンアドレスをリンクスタック３０４上へと保存する。命令Ｃは、プロシージャＰＲＯＣ２へと分岐し、そして処理される次の命令は、命令Ｄである。

【００４３】
命令Ｄは、プロシージャＰＲＯＣ２内の最初の命令であり、そしてR_{14}の値をソフトウェアスタック上へとプッシュすることにより現在のリターンアドレスを保存する。命令Ｄは、別の準備命令であり、次のプロシージャコール命令（命令Ｅ）についての準備を行う。命令Ｅが、プロシージャコールとして識別されるときに、リンクスタックロジック回路３１０は、命令Ｅに関連するリターンアドレスが、リンクスタック３０４上へとロードされるようにする。命令Ｅは、プロシージャＰＲＯＣ２内の２番目の命令であり、そしてプロシージャＰＲＯＣ３を呼び出す。命令Ｅは、命令Ｆ、プロシージャＰＲＯＣ３内の最初の命令、に関連するアドレスへと分岐する。命令Ｆは、プロシージャＰＲＯＣ３内の唯一の命令であり、そしてリターンである。特に、命令Ｆは、現在、リンクレジスタ（R_{14}）の中の値へと分岐する。一般に、既存のプロセッサアーキテクチャにおいては、命令Ｆは、命令リターンとして認識される。命令Ｆが処理されるときに、検出ロジック回路４５０は、命令Ｆがプロシージャリターンであることを決定し、そしてリンクスタック３０４上の次のリターンアドレスが、取り出されるようにする。プロセッサは、プロシージャＰＲＯＣ２へと戻すためにリターンアドレスを使用する。

【００４４】
プロシージャＰＲＯＣ２内において、処理されるべき次の命令は、ソフトウェアスタックから現在の値を「ポップ」して出し、そしてそれをレジスタR_{12}に保存する命令Ｇである。説明図を簡単にするために、命令Ｇは、単一のレジスタを「ポップ」する。しかしながら、代替実施形態においては、ＰＯＰ命令は、複数のレジスタについての複数の値を戻すことができる。この代替実施形態においては、プロセッサ１００は、レジスタリストの中のこれらのレジスタのうちの１つをブランチターゲットアドレスとして使用して、レジスタリストを後続のブランチ命令と比較するために、「ポップされた」レジスタのリストを保持することができる。一実施形態においては、検出ロジック回路４５０は、「ポッ

プされた」レジスタのリストを記憶することができる。

【００４５】

命令Ｈは、今やＲ$_{12}$の中にある取り出されたアドレスへと分岐する。たとえ命令Ｈが、明示ブランチ命令（ＢＸ）でないとしても、それは同等なブランチ命令である。当業者が理解するように、ＭＯＶ、ＰＣ、Ｒ$_{N}$はまた、暗黙ブランチ命令として解釈されることもできる。図６および７の命令フローチャート６００および７００の中で説明されるように、検出ロジック回路２５０、４５０は、「ポップされた」レジスタ（命令ＨのＲ$_{12}$）に対するブランチ命令と一緒にＰＯＰ命令（命令Ｇ）が、プロシージャリターンシーケンスを構成することを決定する。結果として、プロセッサ１００は、次のフェッチアドレスを提供するためにリンクスタック３０４を使用し、そして命令フェッチは、プロシージャＰＲＯＣ１へと戻る。

【００４６】

命令Ｈを処理した後に、命令フェッチは、プロシージャＰＲＯＣ１へと戻り、そして命令Ｉを識別する。命令Ｉは、ソフトウェアスタックからの次の値をＲ$_{2}$へとポップする。依然としてプロシージャＰＲＯＣ１内において、命令Ｊは、Ｒ$_{2}$に記憶されるアドレスへと分岐する。命令Ｈと同様に、命令Ｊは、以前に「ポップされた」レジスタに記憶されるアドレスへと分岐する。結果として、検出ロジック回路２５０、４５０は、命令Ｊがプロシージャリターン命令であることを決定し、そしてリンクスタック３０４からの次の値が、次のグループの命令をフェッチするために使用される。この例においては、命令Ｊが処理された後に、命令Ｋが、フェッチされる。命令Ｋは、図３に示されるように、主プログラム内の任意の命令とすることができる。

【００４７】

一実施形態においては、プロセッサ１００は、命令Ｆと、命令ＧおよびＨと、ＩおよびＪとのシーケンスが、プロシージャリターンとして解釈されるべきであることを識別するために検出ロジック回路２５０を使用する。結果として、１組の命令２００が、検出ロジック回路２５０によってラインバッファ１０７の中で出合われるときに、命令Ｆ、Ｈ、およびＪは、命令キャッシュ１０６に保存されるプリデコードされた情報を用いてプロシージャリターン命令であるものとしてプリデコードされる。したがって、命令Ｆ、Ｈ、およびＪが、フェッチロジック回路２０２によって命令キャッシュ１０６からフェッチされるときに、リターン選択ロジック回路３５０は、リターンアドレスが、次のグループの命令をフェッチするために使用されるリンク３０４から取り出されるようにする。

【００４８】

代替実施形態においては、検出ロジック回路４５０は、命令Ｆと、命令ＧおよびＨと、ＩおよびＪとのシーケンスが、プロシージャリターンとして解釈されるべきであることを識別するように設計されることもできる。この場合には、グループの命令２００が、デコードステージ２０５において復号されるときに、検出ロジック回路４５０は、命令Ｆ、Ｈ、およびＪが、プロシージャリターン命令であることを識別し、そしてこれをリターンセレクタロジック回路３５０に伝える。次いでリターンセレクタロジック回路３５０は、リンクスタック３０４内の次のリターンアドレスが、次のフェッチアドレスを決定するために使用されるようにする。

【００４９】

図６は、図３のＣＰＵ１０２内の検出ロジック回路２５０を有するプロセッサ１００によって実行されるステップを示す命令フロー６００を示している。図示を容易にするために、フローチャート６００は、ＣＰＵ１０２内のラインバッファ１０７が、単一の命令幅にすぎず、そしてそれらの命令は、キャッシュラインアドレスの開始からのシーケンスの中で戻されることを仮定している。当業者は、いくつかのプロセッサが、逐次順序を外れた複数の命令を処理することができるラインバッファを有することができることを理解する。ここにおいて説明されるような発明の概念は、いずれのタイプのプロセッサにも適用されることができる。

【００５０】

命令フロー６００は、開始ブロック６０２から開始される。ブロック６０２から、命令フローは、ブロック６０４へと進み、ここでラインバッファ１０７の中の最初の命令は、検出ロジック回路２５０によって処理される。次いで、命令フロー６００は、決定ブロック６０６へと進む。決定ブロック６０６において、検出ロジック回路２５０は、命令が知られているプロシージャリターンであるかどうかを決定する。前述のように、知られているプロシージャリターンは、ＰＯＰ／ブランチシーケンスを除外して先に識別されたプロシージャリターンのうちのどれにすることもできる。決定ブロック６０６において、検出ロジック回路２５０が、命令が以上で知られているプロシージャリターンであることを決定する場合、命令フロー６００は、ブロック６２６へと進み、ここで命令は、プロシージャリターンとして識別され、あるいはフラグ付けされる。決定ブロック６０６において、検出ロジック回路２５０が、命令が以上で知られているプロシージャリターンでないことを決定する場合には、命令フローは、決定ブロック６１０へと進む。

【００５１】

決定ブロック６１０において、検出ロジック回路２５０は、命令が、ポップされたレジスタリストの中にプログラムカウンタ（ＰＣ）を有さないＰＯＰ命令であるかどうかを決定する。命令が、レジスタリストの中にＰＣのないＰＯＰ命令でない場合、命令フロー６００は、決定ブロック６２８へと進む。そうでなくて命令がレジスタリストの中にＰＣを含まないＰＯＰ命令である場合には、命令フロー６００は、ブロック６１２へと進む。ブロック６１２において、検出ロジック回路２５０は、任意の後続の命令を分析する際に使用のためのＰＯＰ命令のレジスタリストをラインバッファ１０７に保存する。

【００５２】

ブロック６１２から、命令フローは、ブロック６１４へと進む。ブロック６１４において、検出ロジック回路２５０は、ラインバッファ１０７から次の命令を取り出す。プロセスフローは、ブロック６１４から決定ブロック６１６へと続く。決定ブロック６１６において、検出ロジック回路２５０は、ラインバッファ１０７の中の次の命令が、レジスタリストに保存されるレジスタのうちのどれかに対するブランチ命令であるかどうかを決定する。命令が、レジスタリストの中のレジスタに対するブランチである場合、命令フローは、ブロック６２６へと進み、ここで命令は、プロシージャリターン命令としてフラグ付けされる。決定ブロック６１６において、検出ロジック回路２５０が、命令が保存済みのレジスタリストの中のブランチ命令でないことを決定する場合、命令フロー６００は、決定ブロック６１７へと続く。

【００５３】

決定ブロック６１７において、検出ロジック回路２５０は、命令が、ブランチ命令であるかどうかを決定する。命令が、ブランチ命令である場合、命令フローは、決定ブロック６２８へと進む。決定ブロック６１７において、検出ロジック回路２５０が、命令がブランチ命令でないことを決定する場合、命令フローは、決定ブロック６１８へと進む。決定ブロック６１８において、検出ロジック回路２５０は、命令が、保存済みのレジスタリストの中のレジスタのうちのどれかを上書きするかどうかを決定する。命令が、保存済みのレジスタリストの中のレジスタのうちのどれかを上書きする場合、命令フロー６００は、ブロック６２０へと続き、ここで上書きされたレジスタは、保存済みのレジスタリストから取り除かれる。ブロック６２０から、命令フロー６００は、決定ブロック６２２へと続く。

【００５４】

決定ブロック６１８において、検出ロジック回路２５０は、命令が、保存済みのレジスタリストの中の任意のレジスタを上書きしなかったことを決定する場合、命令フロー６００は、決定ブロック６２２へと進む。決定ブロック６２２において、検出ロジック回路２５０は、ラインバッファ１０７について残っている任意の命令があるかどうかを決定する。ラインバッファについて残っている命令がない場合、命令フロー６００は、ブロック６２４で終了する。ラインバッファ１０７の中に残っている命令がある場合、命令フロー６００は、ブロック６１４へと戻って進み、ここでラインバッファ１０７の中の次の命令が

処理される。

【００５５】

ブロック６２６において、検出ロジック回路は、リターン命令として命令にタグを付ける。前述のように、リターン命令にタグを付けることは、フェッチロジック回路２０２が、命令が命令キャッシュ１０６からフェッチされるときにリターン命令を識別することを可能にする。ブロック６２６から、命令フロー６００は、決定ブロック６２８へと進む。決定ブロック６２８において、検出ロジック回路２５０は、ラインバッファ１０７の中に処理されるように残っている任意の命令があるかどうかを決定する。ラインバッファ１０７の中に処理されるように残っている命令がない場合、命令フロー６００は、ブロック６２４において終了する。処理されるように残っている追加の命令がある場合には、命令フロー６００は、ブロック６０４へと進み、ここで次の命令が、検出ロジック回路２５０によって処理される。

【００５６】

図７は、図４の上位パイプライン１５１に結合されたデコードロジック回路４０６の中に検出ロジック回路４５０を有するＣＰＵ１０２によって実行されるステップを示す命令フロー７００を示している。図示を容易にするために、命令フロー７００の中で概説される命令の処理は、デコードロジック回路４０６が、プロセッササイクル当たりに単一の命令を処理することを仮定している。当業者は、いくつかのプロセッサが、プロセッササイクル当たりに複数の命令を処理することができるデコードロジック回路を有することができることを理解する。ここにおいて説明される発明の概念は、いずれのタイプのプロセッサにも適用されることができる。

【００５７】

命令フロー７００は、開始ブロック７０２から開始される。ブロック７０２から、命令フローは、ブロック７０４へと進み、ここで命令は、デコードロジック回路４０６によってデコードステージ２０５の中で処理される。ブロック７０４から、命令フローは、決定ブロック７０６へと続く。決定ブロック７０６において、検出ロジック回路４５０は、命令がプロシージャリターンであるかどうかを決定する。この例においては、検出ロジック回路４５０は、命令がＰＯＰ／ブランチシーケンス以外の前もって知られているプロシージャリターンのうちのどれかである場合に、命令がプロシージャリターンであることを決定する。検出ロジック回路４５０が、命令がプロシージャリターンであることを決定する場合、命令フロー７００は、ブロック７０８へと続く。検出ロジック回路４５０が、命令がプロシージャリターンでないことを決定する場合には、命令フローは、決定ブロック７１０へと続く。

【００５８】

決定ブロック７１０において、検出ロジック回路４５０は、命令が、レジスタリストの中にプログラムカウンタ（ＰＣ）を有さないＰＯＰ命令であるかどうかを決定する。命令が、そのレジスタリストの中にＰＣのないＰＯＰ命令でない場合、プロセスフローは、ブロック７０４へと後方に戻る。決定ブロック７１０において、検出ロジック回路４５０が、復号された命令が、そのレジスタリストの中にＰＣを含まないＰＯＰ命令であることを決定する場合、命令フロー７００は、ブロック７１２へと続く。プロセッサ１００は、ソフトウェアスタックから複数のレジスタをポップすることができる可能性があるので、ブロック７１２において、検出ロジック回路４５０は、ポップされたレジスタリストを保存する。ブロック７１２から、命令フロー７００は、ブロック７１４へと進む。

【００５９】

ブロック７１４において、プロセッサ１００は、次の命令をデコードステージ２０５へとロードし、そしてデコードロジック回路４０６は、その命令を処理する。命令が、ブロック７１４においてロードされた後に、命令フロー７００は、決定ブロック７１６へと進む。決定ブロック７１６において、検出ロジック回路４５０は、命令が、保存済みのレジスタリストの中のレジスタに対するブランチであるかどうかを決定する。検出ロジック回路４５０が、命令が保存済みのレジスタリストの中のレジスタに対するブランチであるこ

とを決定する場合、プロセスフローは、ブロック７０８へと続く。検出ロジック回路４５０が、命令が保存済みのレジスタリストの中のレジスタに対するブランチ命令でなかったことを決定する場合には、命令フロー７００は、決定ブロック７１８へと進む。

【００６０】

決定ブロック７１８において、検出ロジック回路４５０は、命令が、ブランチ命令であるかどうかを決定する。命令が、ブランチ命令である場合、命令フローは、ブロック７０４へと後方に戻り、ここで次の命令は、デコードステージ２０５へとロードされる。命令が、決定ブロック７１８においてブランチ命令でない場合には、命令フロー７００は、決定ブロック７２０へと進む。決定ブロック７２０において、検出ロジック回路４５０は、命令が、保存済みのレジスタリストの中のレジスタを上書きするかどうかを決定する。

【００６１】

命令が、保存済みのレジスタリストの中のレジスタを上書きしない場合、命令フロー７００は、ブロック７１４へと戻り、ここで次の命令は、デコードステージ２０５へとロードされ、そしてデコードロジック回路４０６によって処理される。命令が、決定ブロック７２０において保存済みのレジスタリストの中のレジスタを上書きする場合、命令フロー７００は、ブロック７２２へと続き、ここで上書きされたレジスタは、保存済みのレジスタリストから取り除かれる。ブロック７２２から、命令フロー７００は、ブロック７１４へと戻り、ここで次の命令は、デコードステージ２０５へとロードされ、そしてデコードロジック回路４０６によって処理される。

【００６２】

ここにおいて開示される実施形態に関連して説明される様々な例示の論理ブロック、モジュール、回路、要素、および／またはコンポーネントは、ここにおいて説明される機能を実行するように設計された汎用プロセッサ、デジタル信号プロセッサ(digital signal processor)（ＤＳＰ）、特定用途向け集積回路(application specific integrated circuit)（ＡＳＩＣ）、フィールドプログラマブルゲートアレイ(field programmable gate array)（ＦＰＧＡ）または他のプログラマブルロジックコンポーネント、ディスクリートゲート(discrete gate)またはトランジスタロジック、ディスクリートハードウェアコンポーネント(discrete hardware components)、あるいはそれらの任意の組合せを用いてインプリメントされ、または実行されることができる。汎用プロセッサは、マイクロプロセッサとすることができるが、代替案においてはプロセッサは、従来の任意のプロセッサ、コントローラ、マイクロコントローラ、または状態機械とすることもできる。プロセッサは、コンピューティングコンポーネントの組合せ、例えば、ＤＳＰとマイクロプロセッサとの組合せ、複数のマイクロプロセッサ、ＤＳＰコアと組み合わされた１つまたは複数のマイクロプロセッサ、あるいは他のそのような任意のコンフィギュレーション、としてインプリメントされることもできる。

【００６３】

特定の実施形態が、ここにおいて示され、そして説明されているが、当業者は、同じ目的を達成するように予測される任意の構成が、示される特定の実施形態の代わりにされることができることと、本発明が、他の環境において他のアプリケーションを有することとを理解する。本願は、本発明の任意の適応または変形をカバーするように意図される。添付の特許請求の範囲は、ここにおいて説明される特定の実施形態だけに本発明の範囲を限定するようには決して意図されない。

下記に出願時の請求項１－２５に対応する記載を付記１－２５として表記する。

付記１

パイプラインプロセッサの中のプロシージャから戻るときにリンクスタックからリターンアドレスを取り出すための方法であって、

ソフトウェアスタックからリターンアドレスを取り出すように動作可能な検索命令を識別することと、

前記リターンアドレスへと分岐するように動作可能なブランチ命令を識別することと、

識別される前記命令と前記ブランチ命令との両方に応じて、前記リンクスタックから前

記リターンアドレスを取り出すことと、
前記リターンアドレスを使用して後続の命令をフェッチすることと、
を備える方法。
付記２
前記検索命令は、ＰＯＰ命令である、付記１に記載の方法。
付記３
前記検索命令は、ロード命令である、付記１に記載の方法。
付記４
前記ブランチ命令は、ＢＸ命令である、付記１に記載の方法。
付記５
前記ブランチ命令は、ＭＯＶ命令である、付記１に記載の方法。
付記６
前記の前記検索命令を識別することは、前記リターンアドレスを含むレジスタを識別することをさらに備える、付記１に記載の方法。
付記７
前記検索命令を識別することは、レジスタリストを保持することをさらに備え、前記レジスタリストは、複数のレジスタを有し、前記複数のレジスタの中の少なくとも１つのレジスタは、前記リターンアドレスを含む、付記１に記載の方法。
付記８
前記レジスタリストを保持することは、前記複数のレジスタのうちのどれかが、後続の命令によって上書きされる場合に、前記レジスタリストからレジスタを取り除くことを備える、付記７に記載の方法。
付記９
前記ブランチ命令を識別することは、検出ロジック回路によって実行される、付記１に記載の方法。
付記１０
前記検出ロジック回路は、プリデコードロジック回路と共に含まれる、付記９に記載の方法。
付記１１
前記検出ロジック回路は、デコードロジック回路と共に含まれる、付記９に記載の方法。
付記１２
前記ブランチ命令を識別することは、命令キャッシュの中の前記ブランチ命令にフラグ付けすることをさらに備える、付記１に記載の方法。
付記１３
命令キャッシュに結合されたラインバッファと；
前記命令キャッシュに結合され、予測リターンアドレスを記憶するリンクスタックを有するフェッチロジック回路と、なお命令は、前記ラインバッファから前記命令キャッシュへとロードされ、前記フェッチロジック回路は、前記命令キャッシュから命令を取り出す；
前記ラインバッファと通信するプリデコードロジック回路と、なお前記プリデコードロジック回路は、プロシージャリターンシーケンスを識別するための検出ロジック回路をさらに備え、前記プロシージャリターンシーケンスは、ソフトウェアスタックからリターンアドレスを取り出すように動作可能な検索命令と前記取り出されたリターンアドレスに分岐するブランチ命令とを備え、前記パイプラインプロセッサは、前記プロシージャリターンシーケンスの前記識別に応じて前記リンクスタックから前記予測されたリターンアドレスを取り出す；
を備えるパイプラインプロセッサ。
付記１４
前記検出ロジック回路は、前記ブランチ命令が、前記ラインバッファから前記命令キャ

ッシュへとロードされるときに、前記プロシージャリターンシーケンスの前記ブランチ命令にフラグ付けする、付記１３に記載のパイプラインプロセッサ。
付記１５
前記フェッチロジック回路は、前記フラグ付けされた情報から前記プロシージャリターンシーケンスを識別する、付記１４に記載のパイプラインプロセッサ。
付記１６
前記フェッチロジック回路内のリターンセレクタロジック回路は、前記フラグ付けされた情報から前記リターンシーケンスを識別する、付記１５に記載のパイプラインプロセッサ。
付記１７
前記検索命令は、ＰＯＰ命令である、付記１３に記載のパイプラインプロセッサ。
付記１８
前記検索命令は、ロード命令である、付記１３に記載のパイプラインプロセッサ。
付記１９
前記ブランチ命令は、ＢＸ命令である、付記１３に記載のパイプラインプロセッサ。
付記２０
予測されたリターンアドレスを記憶するリンクスタックを有し、命令キャッシュから命令をフェッチするフェッチロジック回路と、
前記フェッチロジック回路に結合されたデコードロジック回路と、
を備え、前記フェッチされた命令は、前記デコードロジック回路によって復号され、前記デコードロジック回路は、検出ロジック回路をさらに備え、前記検出ロジック回路は、ソフトウェアスタックからアドレスを取り出すように動作可能な検索命令と、前記取り出されたアドレスに分岐するように動作可能なブランチ命令とを備えるプロシージャリターンシーケンスを識別し、パイプラインプロセッサは、前記プロシージャリターンシーケンスの前記識別に応じて前記リンクスタックから前記予測されたリターンアドレスを取り出す、パイプラインプロセッサ。
付記２１
前記フェッチロジック回路は、前記取り出されたアドレスを使用して命令をフェッチする、付記２０に記載のパイプラインプロセッサ。
付記２２
前記検索命令は、ＰＯＰ命令である、付記２０に記載のパイプラインプロセッサ。
付記２３
前記検索命令は、ロード命令である、付記２０に記載のパイプラインプロセッサ。
付記２４
前記ブランチ命令は、前記検索命令によって識別されるアドレスへと分岐する、付記２０に記載のパイプラインプロセッサ。
付記２５
前記ブランチ命令は、ＭＯＶ命令である、付記２０に記載のパイプラインプロセッサ。

【図１】

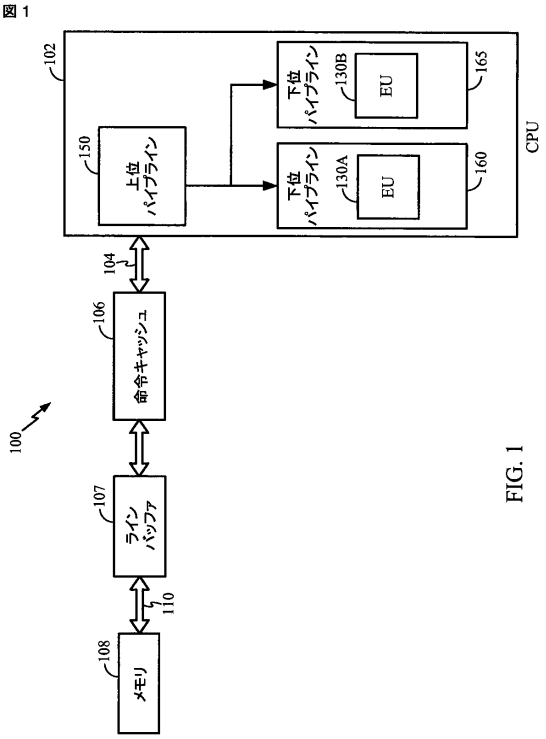

FIG. 1

【図２】

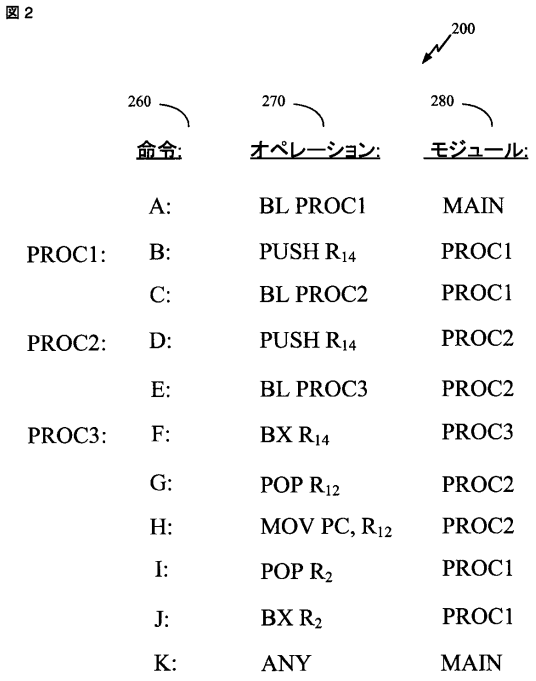

FIG. 2

【図３】

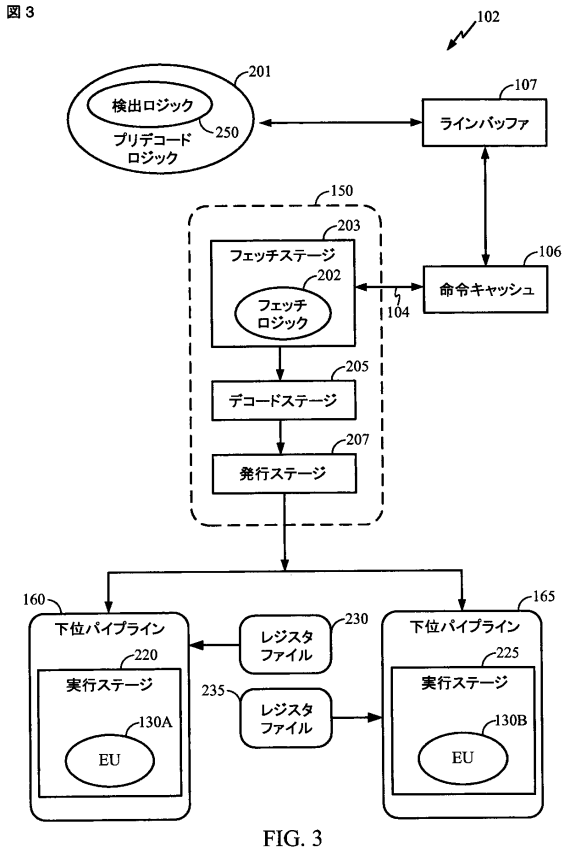

FIG. 3

【図４】

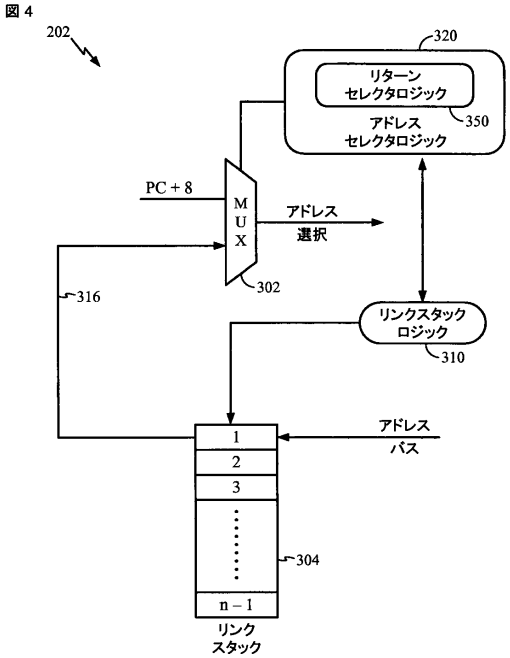

FIG. 4

【図５】

図５

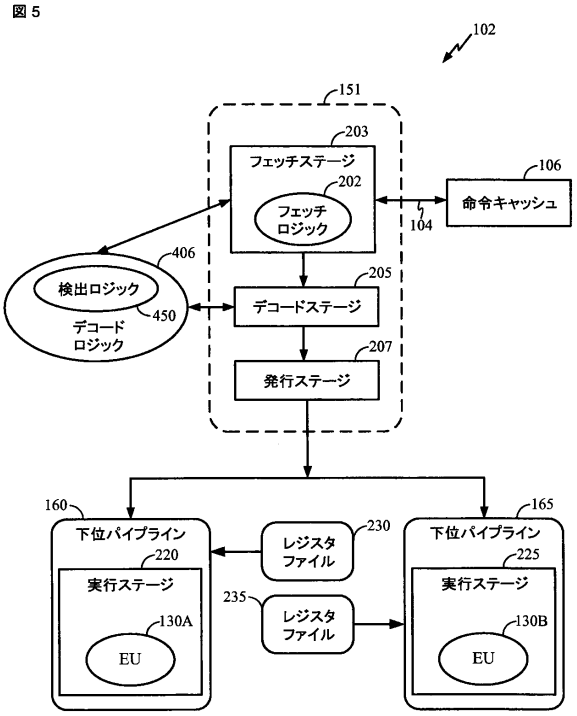

FIG. 5

【図６】

図６

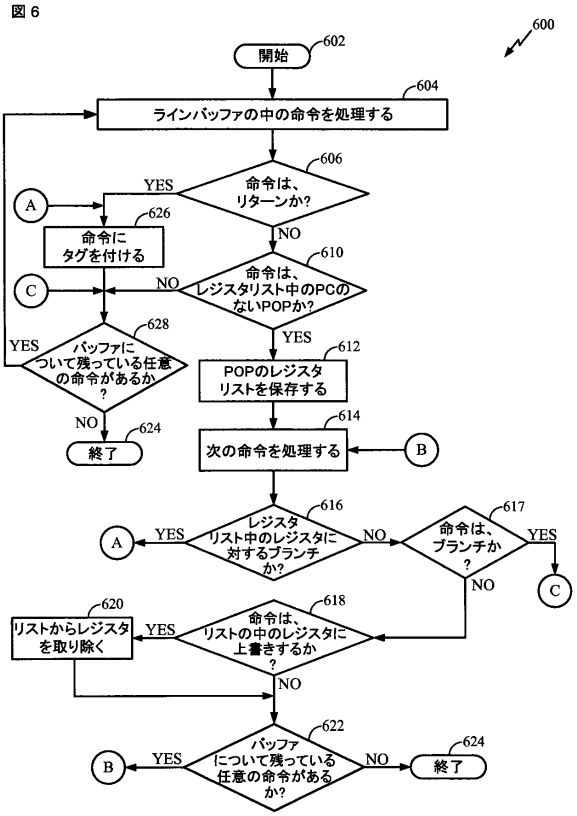

FIG. 6

【図７】

図７

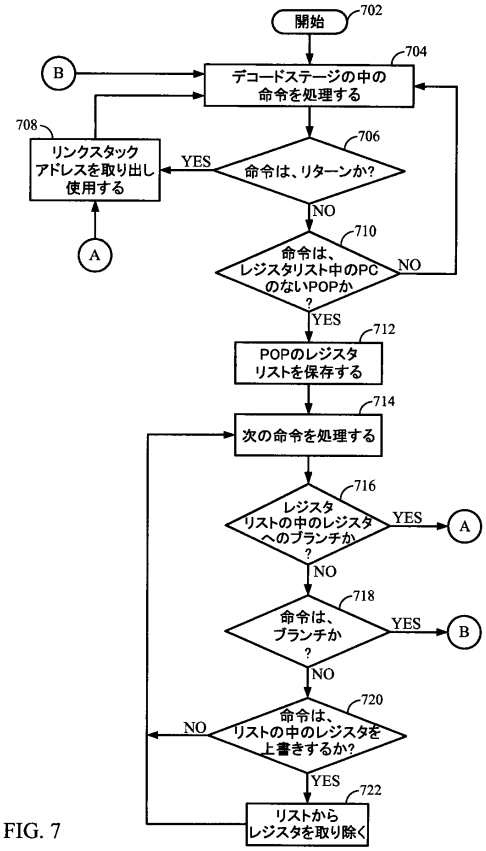

FIG. 7

フロントページの続き

(74)代理人 100075672
弁理士 峰 隆司
(74)代理人 100095441
弁理士 白根 俊郎
(74)代理人 100084618
弁理士 村松 貞男
(74)代理人 100103034
弁理士 野河 信久
(74)代理人 100119976
弁理士 幸長 保次郎
(74)代理人 100153051
弁理士 河野 直樹
(74)代理人 100140176
弁理士 砂川 克
(74)代理人 100101812
弁理士 勝村 紘
(74)代理人 100124394
弁理士 佐藤 立志
(74)代理人 100112807
弁理士 岡田 貴志
(74)代理人 100111073
弁理士 堀内 美保子
(74)代理人 100134290
弁理士 竹内 将訓
(74)代理人 100127144
弁理士 市原 卓三
(74)代理人 100141933
弁理士 山下 元
(72)発明者 モロウ、マイケル・ウィリアム
アメリカ合衆国、カリフォルニア州 ９２１２１、サン・ディエゴ、モアハウス・ドライブ ５７７５
(72)発明者 ディーフェンダーファー、ジェームズ・ノリス
アメリカ合衆国、カリフォルニア州 ９２１２１、サン・ディエゴ、モアハウス・ドライブ ５７７５

審査官 清木 泰

(56)参考文献 特開２００１－１００９９３（ＪＰ，Ａ）
特開平０７－２８１８９２（ＪＰ，Ａ）
特開平０７－２３９７８２（ＪＰ，Ａ）
特開２００３－２５６１９７（ＪＰ，Ａ）
米国特許第０５８１２８１３（ＵＳ，Ａ）
米国特許第０６３７４３５０（ＵＳ，Ｂ１）
本永朝雄，6809アセンブリプログラミング，日本，株式会社サイエンス社，１９８６年１１月２５日，Pages:62-68
Stephen B. Fubber，VLSI RISC Architecture and Organization，米国，Marcel Dekker Inc.，１９８９年，Pages:231-233

(58)調査した分野(Int.Cl., ＤＢ名)
Ｇ０６Ｆ９／３０－９／４２