(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第3628379号
(P3628379)
(45)発行日　平成17年3月9日(2005.3.9)　(24)登録日　平成16年12月17日(2004.12.17)

(51)Int.Cl.[7]　F I
G06F　9/38　G06F　9/38　310B

請求項の数 8　(全 24 頁)

(21)出願番号　特願平7-134011
(22)出願日　平成7年5月31日(1995.5.31)
(65)公開番号　特開平7-334361
(43)公開日　平成7年12月22日(1995.12.22)
審査請求日　平成14年4月9日(2002.4.9)
(31)優先権主張番号　252030
(32)優先日　平成6年6月1日(1994.6.1)
(33)優先権主張国　米国(US)

(73)特許権者　591016172
アドバンスト・マイクロ・ディバイシズ・インコーポレイテッド
ＡＤＶＡＮＣＥＤ　ＭＩＣＲＯ　ＤＥＶＩＣＥＳ　ＩＮＣＯＲＰＯＲＡＴＥＤ
アメリカ合衆国、９４０８８－３４５３　カリフォルニア州、サニィベイル、ピィ・オゥ・ボックス・３４５３、ワン・エイ・エム・ディ・プレイス、メイル・ストップ・６８（番地なし）
(74)代理人　100064746
弁理士　深見　久郎
(74)代理人　100085132
弁理士　森田　俊雄

最終頁に続く

(54)【発明の名称】命令を処理するためのパイプラインを有するマイクロプロセッサ装置およびそれにおいて用いるためのプログラムカウンタ値を発生する装置

(57)【特許請求の範囲】
【請求項１】
命令を処理するためのパイプラインを有するマイクロプロセッサにおいて用いるためのプログラムカウンタ値を発生する装置であって、
上位の第１のプログラムカウンタ部分と、下位の第１のプログラムカウンタ部分と、第１のキャリー信号とを有する第１のプログラムカウンタ値を提供するための第１のプログラムカウンタ回路を備え、第１のカウンタ値は検索された命令のプログラムカウント値を表わし、さらに
下位の第１のプログラムカウンタ部分および第１のキャリー信号を受取り、下位の第１のプログラムカウンタ部分を用いて実行プログラムカウンタ値を発生するための実行プログラムカウンタ回路を備え、実行プログラムカウンタ回路は第１のプログラムカウンタ回路に結合され、実行プログラムカウンタ値は実行された命令のプログラムカウンタ値を表わし、
第１のプログラムカウンタ回路は、下位プログラムカウンタ値発生器回路と上位プログラムカウンタ値発生器回路とを含み、
下位プログラムカウンタ値発生器回路は、複数個のシーケンシャルな次のプログラムカウンタ値を提供するための複数個の加算器回路を含み、各加算器回路は現在のプログラムカウンタ値の下位の部分と、パイプライン内のそれぞれのシーケンシャルな命令の命令長値とを受取り、現在のプログラムカウンタ値の下位の部分を命令長値に加算して、次のプログラムカウンタ値の１つにおけるそれぞれの下位の部分および対応するキャリー信号を提

供する、命令を処理するためのパイプラインを有するマイクロプロセッサにおいて用いるためのプログラムカウンタ値を発生する装置。
【請求項２】
下位の第１のプログラムカウンタ部分およびキャリー信号をストアするための複数個の記憶位置をさらに備え、各記憶位置はパイプラインの各段に対応し、第１のプログラムカウンタ回路および実行プログラムカウンタ回路に結合される、請求項１に記載の装置。
【請求項３】
上位プログラムカウンタ値発生器回路はインクリメンタ回路を含み、インクリメンタ回路は現在のプログラムカウンタ値の上位ビットおよび増分信号を受取り、かつ次のプログラムカウンタ値の上位部分を提供する、請求項１に記載の装置。
【請求項４】
上位プログラムカウンタ値発生器回路はキャリーセレクタ回路を含み、キャリーセレクタ回路はキャリー信号を受取り、かつキャリー信号に基づく増分信号を提供する、請求項３に記載の装置。
【請求項５】
命令を処理するためのパイプラインを有するマイクロプロセッサ装置であって、
マイクロプロセッサコアを備え、マイクロプロセッサコアは命令を実行するための機能ユニットを含み、さらに
上位の第１のプログラムカウンタ部分、下位の第１のプログラムカウンタ部分、および第１のキャリー信号を有する第１のプログラムカウンタ値を提供するための第１のプログラムカウンタ回路を備え、第１のカウンタ値は検索された命令におけるプログラムカウント値を表わし、さらに
下位の第１のプログラムカウンタ部分および第１のキャリー信号を受取り、下位の第１のプログラムカウンタ部分を用いて実行プログラムカウンタ値を発生するための実行プログラムカウンタ回路を備え、実行プログラムカウンタ回路は第１のプログラムカウンタ回路に結合され、実行プログラムカウンタ値は実行された命令のプログラムカウンタ値を表わし、機能ユニットは命令を実行するとき実行プログラムカウンタ値を用い、
第１のプログラムカウンタ回路は、下位プログラムカウンタ値発生器回路と上位プログラムカウンタ値発生器回路とを含み、
下位プログラムカウンタ値発生器回路は、複数個のシーケンシャルな次のプログラムカウンタ値を提供するための複数個の加算器回路を含み、各加算器回路は現在のプログラムカウンタ値の下位の部分と、パイプライン内のそれぞれのシーケンシャルな命令の命令長値とを受取り、現在のプログラムカウンタ値の下位の部分を命令長値に加算して、次のプログラムカウンタ値の１つにおけるそれぞれの下位の部分および対応するキャリー信号を提供する、命令を処理するためのパイプラインを有するマイクロプロセッサ装置。
【請求項６】
下位の第１のプログラムカウンタ部分およびキャリー信号をストアするための複数個の記憶位置をさらに備え、各記憶位置はパイプラインの各段に対応し、第１のプログラムカウンタ回路および実行プログラムカウンタ回路に結合される、請求項５に記載の装置。
【請求項７】
上位プログラムカウンタ値発生器回路はインクリメンタ回路を含み、インクリメンタ回路は現在のプログラムカウンタ値の上位ビットおよび増分信号を受取り、かつ次のプログラムカウンタ値の上位部分を提供する、請求項５に記載の装置。
【請求項８】
上位プログラムカウンタ値発生器回路はキャリーセレクタ回路を含み、キャリーセレクタ回路はキャリー信号を受取り、かつキャリー信号に基づき増分信号を提供する、請求項７に記載の装置。
【発明の詳細な説明】
【０００１】
【発明の背景】

この発明はマイクロプロセッサに関し、より特定的にはマイクロプロセッサのプログラムカウンタ値を更新するためのメカニズムに関する。

【０００２】

マイクロプロセッサとは１つまたは非常に少数の半導体チップ上で実現されるプロセッサである。半導体チップ技術はマイクロプロセッサ内の回路密度およびスピードをますます増大させているが、しかしながらマイクロプロセッサと外部メモリとの間の配線は、パッケージング技術によって制限されている。オンチップ配線は極めて安価であるが、オフチップ配線は非常に高価である。マイクロプロセッサの性能を改善することを意図するいかなる技術も、増大した回路密度およびスピードを利用する一方でパッケージング技術およびプロセッサとその外部メモリとの間の物理的な分離の制限内に留まらなければならない。回路密度を高めることによって、より一層複雑な設計への道が開けるとはいえ、マイクロプロセッサの動作はユーザがそのマイクロプロセッサをどうやって使うか理解できるように単純かつ明快なもののままでなければならない。

【０００３】

既存のマイクロプロセッサの大多数はスカラ計算を目的としているが、スーパースカラマイクロプロセッサはマイクロプロセッサの進化における次の論理的なステップである。スーパースカラという語は、スカラ命令の同時実行によって性能を向上させたコンピュータの実現例を表わす。スカラ命令は、汎用マイクロプロセッサにおいて典型的に見出されるタイプの命令である。今日の半導体処理技術を用いれば、単一のプロセッサチップにかつては大規模科学計算用プロセッサにしか応用できなかった高性能な技術を組入れることができる。しかしながら、大規模プロセッサに応用される技術の多くは、スカラ計算には不適切であるか、マイクロプロセッサに応用するにはあまりに高価であるかのどちらかである。

【０００４】

マイクロプロセッサはアプリケーションプログラムを実行する。１つのアプリケーションプログラムには、命令のグループが含まれる。アプリケーションプログラムを実行するにあたって、プロセッサは何らかのシーケンスにおいて命令をフェッチし、実行する。１つの命令の実行にさえ、いくつかのステップが関わっている。このステップは、命令をフェッチするステップと、それをデコードするステップと、そのオペランドを組立てるステップと、命令によって特定される動作を行なうステップと、命令の結果を記憶装置に書込むステップとを含む。命令の実行は周期的クロック信号によって制御される。クロック信号の周期はプロセッサのサイクル時間である。

【０００５】

プロセッサがプログラムを完了させるのにかかる時間は、３つの要因によって決定される。すなわち、そのプログラムを実行するのに必要な命令の数と、１つの命令を実行するのに必要なプロセッササイクルの平均数と、プロセッサのサイクル時間とである。プロセッサの性能はプロセッサがプログラムを完了させるのにかける時間を低減することによって向上するが、これにはこれらの要因の１つ以上を低減することが要求される。

【０００６】

マイクロプロセッサの性能を向上させる１つの方法は、パイプライン化と呼ばれる技術を用いて、異なった命令におけるステップを重複させることによるものである。命令をパイプライン化するには、命令実行の様々なステップがパイプライン段と呼ばれる独立的なユニットによって行なわれる。パイプライン段はクロックドレジスタによって分離される。異なった命令のステップが、異なったパイプライン段において独立的に実行される。パイプライン化は、命令を重複させ、それによりプロセッサが同時に１つより多くの命令を扱うことができるようにすることによって、１つの命令の実行に必要な総合的な時間を、短縮することはないものの、命令の実行に必要なサイクルをかなりの数、低減する。これはプロセッサのサイクル時間を増大させることなく、またしばしば低減して行なわれる。パイプライン化は、典型的には命令１つあたりのサイクルの平均数を３のファクタ分も低減する。しかしながら、分岐命令を実行する場合、パイプラインは時折、分岐動作の結果が

知られ、正しい命令が実行のためにフェッチされるまで、止まってしまうかもしれない。この遅延は、分岐遅延ペナルティとして知られている。パイプライン段の数を増やすことは、典型的には命令１つあたりのサイクルの平均数に関連して分岐遅延ペナルティを増やすことにも繋がる。

【０００７】

典型的なマイクロプロセッサは、１つ１つのプロセッササイクルごとに１つの命令を実行する。スーパースカラプロセッサが低減する命令１つあたりのサイクルの平均数は、異なったパイプライン段での命令の同時実行を可能にするだけでなく、同一のパイプライン段において命令を同時に実行できるようにすることにより、パイプライン化されたスカラプロセッサで可能な数を超える。スーパースカラという語は、科学計算でよくあるようなベクトルまたはアレイ上での多数の同時動作と区別される、スカラ量上での多数の同時動作を強調するものである。

【０００８】

スーパースカラプロセッサはコンセプトとしては単純であるが、性能の向上を達成するには、プロセッサのパイプラインを広くする以上のことがなされている。パイプラインを広くすればサイクル１つあたりに１つより多くの命令を実行することが可能になるが、命令における所与のどのシーケンスもこの能力を利用できるという保証はない。命令は互いに独立しているのではなく、相互に関連しており、これらの相関のせいで、命令によっては同一のパイプライン段に入ることが妨げられる。さらに、命令をデコードし、実行するためのプロセッサのメカニズムは、同時に実行され得る命令を発見する能力において大きな違いとなり得る。

【０００９】

スーパースカラ技術は、命令のセットおよび他のアーキテクチャ的な特徴からは独立したプロセッサ機構に大きく関わっている。したがって、スーパースカラ技術の魅力の１つは、既存のアーキテクチャとコードの互換性があるプロセッサを開発できる可能性である。多くのスーパースカラ技術は、縮小命令セットコンピュータ（ＲＩＳＣ）または複合命令セットコンピュータ（ＣＩＳＣ）アーキテクチャのいずれにも等しく良好に適用される。しかしながら、ＲＩＳＣアーキテクチャの多くにおける規則性のため、スーパースカラ技術は最初はＲＩＳＣプロセッサの設計に応用されてきた。

【００１０】

命令ポインタ（ＩＰ）とも呼ばれる、プログラムカウンタ（ＰＣ）は、命令がメモリからフェッチされ実行される際の、命令のメモリアドレスを保存する。プログラムカウンタとして言及される、プログラムカウンタ値を維持し更新するためのプログラムカウンタメカニズムは、インクリメンタと、セレクタと、レジスタとを含む。各命令がフェッチされデコードされるにつれ、次の順次命令のアドレスが、インクリメンタを用いてプログラムカウンタの現在の値に現在の命令のバイト長を加え、この次の順次命令をレジスタ内に位置づけることによって、形成される。分岐が行なわれると、目的命令のアドレスが増分された値の代わりにセレクタによって選択され、この目的アドレスがレジスタ内に位置づけられる。

【００１１】

プログラムカウンタ値は２つの目的を果たす。プログラムカウンタ値は、フェッチされ実行されるべき次の命令のメモリアドレスを提供する。プログラムカウンタ値はまた、命令ストリームの実行を止めた問題に遭遇した命令のアドレスを識別する。このアドレスはデバッグをする目的で、または訂正動作が行なわれた後でなされるかもしれない命令ストリームの実行の続行のために用いられてもよい。

【００１２】

マイクロプロセッサにおいてパイプライン化の実現例を用いる場合、プログラムカウンタ値は、この値が命令フェッチアドレスを提供するパイプラインの始めに維持される。この値はフェッチＰＣ値として言及される。このフェッチＰＣ値はパイプラインに入る命令を指し示す。命令がパイプライン段に沿って伝播するにつれ、後続する命令がフェッチされ

、パイプライン内に位置づけられる。したがって、フェッチＰＣ値は第１段以外のパイプラインの段にある命令には対応しない。命令ストリームの実行を止める問題のほとんどは、パイプラインの初めよりも終わりの近く、または終わりにおいて検出される傾向にあるので、命令のためのプログラムカウンタ値は命令が実行されている間維持されていなければならない。この値は実行ＰＣ値と呼ばれる。

【００１３】

実行ＰＣ値を維持するには２つの方法が知られている。第１の方法は、ある命令のＰＣ値が、その命令とともにパイプラインを下っていくためのものである。この方法では、各パイプライン段は実行ＰＣ値をストアするための付加的な記憶装置を必要とする。必要とされる付加的な記憶装置の量は、パイプライン段の数に比例する。第２の方法は、パイプラインの終わりにおいてＤＣ回路を二重にするものである。この方法では、命令の長さの情報だけがパイプライン内の命令に伴う。非分岐命令が完了する際、命令の長さの値は実行ＰＣ値に加算され、次の命令のための実行ＰＣ値が提供される。分岐命令の完了にあたっては、増分された値ではなく、その分岐のための目的アドレスが実行ＰＣ値として提供される。

【００１４】

【発明の概要】

別個に下位プログラムカウンタ値ビットを提供するフェッチプログラムカウンタ回路と、下位プログラムカウンタ値ビットを用いて実行プログラムカウンタ値を発生する実行プログラムカウンタ回路とを設けることによって、実行プログラムカウンタ値を迅速かつ効率的に発生することが可能になるということが発見されている。また、複数個の下位プログラムカウンタビット値を受取り、単一の下位プログラムカウンタビット値を選択する、実行プログラムカウンタ回路を設けることにより、多重パイプラインプロセッサにおいて実行プログラムカウンタ値を容易に発生することが可能になるということも、発見されている。

【００１５】

【詳細な説明】

この発明を実施するために企図されるベストモードの詳細な説明を以下に述べる。この説明は本発明にとって例示的なものとして意図されており、限定的なものととられるべきではない。

【００１６】

図１を参照して、本発明はＸ８６命令セットを実行するスーパースカラＸ８６マイクロプロセッサ１００との関連で最もよく理解することができる。マイクロプロセッサ１００は４８６ＸＬバスまたは他の従来のマイクロプロセッサバスを介して物理的にアドレス指定された外部メモリ１０１に結合される。マイクロプロセッサ１００は命令キャッシュ１０４を含み、命令キャッシュ１０４はバイトキュー１０６に結合され、バイトキュー１０６は命令デコーダ１０８に結合される。命令デコーダ１０８はＲＩＳＣコア１１０に結合される。ＲＩＳＣコア１１０は、レジスタファイル１１２およびリオーダバッファ１１４を、算術論理ユニット１３１（ＡＬＵ０）および算術論理ユニット・シフトユニット１３２（ＡＬＵ１　＆　ＳＨＦ）、特殊レジスタブロック１３３（ＳＲＢ）、ロード／ストアユニット１３４（ＬＳＳＥＣ）、分岐セクション１３５（ＢＲＮＳＥＣ）、および浮動小数点ユニット１３６（ＦＰＵ）などの様々な機能ユニットとともに含む。

【００１７】

ＲＩＳＣコア１１０は、ＡおよびＢオペランドバス１１６、タイプおよびディスパッチ（ＴＡＤ）バス１１８、および結果バス１４０を含み、これらは機能ユニットならびに変位および命令ストア（ＩＮＬＳ）バス１１９に結合され、この変位および命令ロードストア（ＩＮＬＳ）バス１１９は、命令デコーダ１０８とロード／ストアユニット１３４との間に結合される。ＡおよびＢオペランドバス１１６は、レジスタファイル１１２およびリオーダバッファ１１４にも結合される。ＴＡＤバス１１８は命令デコーダ１０８にも結合される。結果バス１４０はリオーダバッファ１１４にも結合される。加えて、分岐セクショ

ン１３５はリオーダバッファ１１４、命令デコーダ１０８、および命令キャッシュ１０４にＸ目的バス１０３を経由して結合される。ＡおよびＢオペランドバス１１６は４本の並列４１ビット幅Ａオペランドバスおよび４本の並列４１ビット幅Ｂオペランドバス、ならびに４本の並列１２ビット幅Ａタグバス、４本の並列１２ビット幅Ｂタグバス、１本の１２ビット幅Ａタグ有効バス、１本の１２ビット幅Ｂタグ有効バス、４本の４ビット幅行先タグバス、および４本の８ビット幅オペレーションコードバスを含む。タイプおよびディスパッチバス１１８は、４本の３ビット幅タイプコードバスおよび１本の４ビット幅ディスパッチバスを含む。変位およびＩＮＬＳバス１１９は、２本の３２ビット幅変位バスおよび２本の８ビット幅ＩＮＬＳバスを含む。

【００１８】

命令キャッシュ１０４に加えて、マイクロプロセッサ１００はデータキャッシュ１５０（ＤＣＡＣＨＥ）および物理タグ回路１６２をも含む。データキャッシュ１５０はＲＩＳＣコアのロード／ストア機能ユニット１３４に結合され、かつプロセッサ内アドレスおよびデータ（ＩＡＤ）バス１０２に結合される。命令キャッシュ１０４はまた、ＩＡＤバス１０２とも結合される。物理タグ回路１６２は、命令キャッシュ１０４とデータキャッシュ１５０との双方と、ＩＡＤバスを介して対話する。命令キャッシュ１０４およびデータキャッシュ１５０は双方とも線形にアドレス可能なキャッシュである。命令キャッシュ１０４およびデータキャッシュ１５０は物理的には分離しているが、しかしながらこれらのキャッシュは双方とも同じアーキテクチャを用いて編成される。

【００１９】

マイクロプロセッサ１００はまた、メモリ管理ユニット（ＭＭＵ）１６４とバスインタフェースユニット１６０（ＢＩＵ）とを含む。ＴＬＢ１６４はＩＡＤバスと物理タグ回路１６２とに結合される。バスインタフェースユニット１６０は物理タグ回路１６２、データキャッシュ１５０、ならびにＩＡＤバス１０２および４８６ＸＬバスなどの外部マイクロプロセッサバスに結合される。

【００２０】

マイクロプロセッサ１００は命令のシーケンスを含むコンピュータプログラムを実行する。コンピュータプログラムは典型的には、ハードディスク、フロッピーディスク、またはコンピュータシステム内に位置づけられる他の不揮発性記憶媒体にストアされる。プログラムが実行されると、そのプログラムは記憶媒体からメインメモリ１０１にロードされる。一旦プログラムにおける命令および関連のデータがメインメモリ１０１内に入ると、個々の命令が実行のために準備され、最後にマイクロプロセッサ１００によって実行される。

【００２１】

メインメモリ１０１内にストアされた後、命令はバスインタフェースユニット１６０を経由して命令キャッシュ１０４に送られ、そこで命令は一時的に保持される。Ｘ８６命令は命令キャッシュ１０４によって、命令キャッシュ１０４が発生するフェッチプログラムカウンタ値を用いて命令デコーダ１０８に与えられる。

【００２２】

命令デコーダ１０８は命令を調べ、行なうべき適切な動作を判断する。たとえば、デコーダ１０８はある特定の命令が、ＰＵＳＨ、ＰＯＰ、ＬＯＡＤ、ＳＴＯＲＥ、ＡＮＤ、ＯＲ、ＥＸＯＲ、ＡＤＤ、ＳＵＢ、ＮＯＰ、ＪＵＭＰ、条件付きＪＵＭＰ（ＢＲＡＮＣＨ）、または他の命令のいずれであるかを判断するだろう。デコーダ１０８がどの特定の命令を存在しているものと判断するかに従い、対応する１つまたは複数のＲＩＳＣ動作（ＲＯＰ）がＲＩＳＣコア１１０の適切な機能ユニットにディスパッチされ、各ＲＯＰまたはＲＯＰのセットに対応するデコードＰＣ値が発生される。デコードＰＣ値はフェッチＰＣ値と非同期に発生される。

【００２３】

命令は、典型的には次に述べるフォーマットで複数のフィールドを含む。すなわちこのフォーマットは、ＯＰ　ＣＯＤＥ、ＯＰＥＲＡＮＤ　Ａ、ＯＰＥＲＡＮＤ　Ｂ、およびＤＥ

ＳＴＩＮＡＴＩＯＮである。たとえば、命令ＡＤＤ　Ａ，Ｂ，Ｃは、レジスタＡの内容をレジスタＢの内容に加算し、その結果をレジスタＣの中に位置づけろ、という意味である。ＬＯＡＤおよびＳＴＯＲＥ動作は、わずかに異なったフォーマットを用いる。たとえば、命令ＬＯＡＤ　Ａ，Ｂ，Ｃは、アドレスから検索されたデータを結果バス上に位置づけろ、ということを意味し、ここにおいてＡ、Ｂ、およびＣはＡオペランドバス、Ｂオペランドバス、および変位バス上に位置づけられたアドレスの構成要素を表わし、これらのアドレスの構成要素は組合せられて論理アドレスを提供し、この論理アドレスはセグメントベースと組合せられて線形アドレスを提供し、この線形アドレスからデータは検索される。またたとえば、命令ＳＴＯＲＥ　Ａ，Ｂ，Ｃは、あるアドレスによって指し示された位置にデータをストアしろという意味であり、ここでＡはＡオペランドバス上に位置づけられたストアデータであり、ＢおよびＣはＢオペランドバスおよび変位バス上に位置づけられたアドレスの構成要素を表わしており、これらのアドレスの構成要素は組合せられて論理アドレスを形成し、この論理アドレスはセグメントベースと結合されて線形アドレスを提供し、この線形アドレスにデータはストアされる。

【００２４】

ＯＰ　ＣＯＤＥは、命令デコーダ１０８からオペレーションコードバスを介してＲＩＳＣコア１１０の機能ユニットへ与えられる。特定の命令のためのＯＰＣＯＤＥが適切な機能ユニットに与えられなければならないだけでなく、命令のための指定されたＯＰＥＲＡＮＤも検索され、機能ユニットに送られなければならない。特定のオペランドの値がまだ計算されていなければ、機能ユニットが命令を実行できるようになる前に、まずその値が計算され機能ユニットに与えられなければならない。たとえば、現在の命令が先行する命令に依存する場合、現在の命令が実行できるようになる前に、先行する命令の結果が決定されなければならない。この状況は、依存性と呼ばれる。

【００２５】

特定の命令が機能ユニットによって実行されるために必要なオペランドは、レジスタファイル１１２、リオーダバッファ１１４によってオペランドバスに与えられるか、結果バス１４０を介して機能ユニットから送られるかのいずれかである。オペランドバスはオペランドを適切な機能ユニットに伝える。一旦機能ユニットがＯＰ　ＣＯＤＥ、ＯＰＥＲＡＮＤ　Ａ、およびＯＰＥＲＡＮＤ　Ｂを受取ると、機能ユニットはその命令を実行し、その結果を結果バス１４０に位置づける。結果バス１４０はすべての機能ユニットの出力およびリオーダバッファ１１４に結合されている。

【００２６】

リオーダバッファ１１４は先入れ先出し（ＦＩＦＯ）装置として管理される。ある命令が命令デコーダ１０８によってデコードされると、対応するエントリがリオーダバッファ１１４内に割当てられる。命令によって計算された結果値は次に、その命令の実行が完了すると割当てられたエントリに書込まれる。結果値は続いてレジスタファイル１１２に書込まれ、命令に関連づけられた例外がなく、命令に影響を与える未決定の推論的分岐がなければ、その命令はリタイアされる。命令がリタイアされると、それに関連の実行プログラムカウンタ値もレジスタファイル１１２内のプログラムカウンタレジスタにストアされる。関連のエントリがリオーダバッファ１１４の先頭に到達しても命令が完了していなかった場合、リオーダバッファ１１４の進行はその命令が完了するまで止められる。しかしながらさらなるエントリを割当て続けることができる。

【００２７】

各機能ユニットは、その命令のためのオペランドがまだ機能ユニットにとって利用可能なものではないためにまだ完了していない命令からのＯＰ　ＣＯＤＥをストアするためのそれぞれの予約ステーション回路（ＲＳ）１２０～１２６を含む。各予約ステーション回路は命令のＯＰ　ＣＯＤＥをタグとともにストアする。このタグは後に予約ステーション回路に到達するであろう欠けているオペランドのための場所を予約しておくものである。この技術は、未決定の命令が予約ステーションにおいてそのオペランドとともに組立てられている間に、マイクロプロセッサ１００が他の命令を実行し続けることができるようにす

ることによって、性能を強化するものである。
【００２８】
マイクロプロセッサ１００は、デコーダ１０８をＲＩＳＣコア１１０の機能ユニットから分離することによって、順序が乱れた発行に対処する。より特定的には、リオーダバッファ１１４および機能ユニットの予約ステーションは、分布命令ウインドウを効果的に確立する。したがって、デコーダ１０８は命令がすぐには実行できない場合でさえ、命令のデコードを続けることができる。命令ウインドウは命令のプールとして働く。この命令のプールから、機能ユニットは前に進み続けつつ命令を引出し、実行する。命令ウインドウはこのようにして、マイクロプロセッサ１００にルックアヘッド能力をもたらす。依存性がクリアされ、オペランドが利用可能なものとなるにつれ、ウインドウ内のより多くの命令が機能ユニットによって実行され、デコーダはさらに多くのデコードされた命令でウインドウを充填し続ける。
【００２９】
マイクロプロセッサ１００はＲＩＳＣコアの分岐セクション１３５を用いてその性能を強化する。分岐が起こると、次の命令はその分岐の結果に依存するので、プログラムにおける命令ストリーム内の分岐はマイクロプロセッサの命令をフェッチする能力の妨げとなる。分岐セクション１３５は、分岐が行なわれるべきかどうかを判断する。加えて、命令キャッシュ１０４は先行する分岐の結果の実行記録をとり続けるための分岐目的バッファを含む。この記録に基づき、ある特定のフェッチされた分岐の間に、フェッチされた分岐命令がどの分岐をとるであろうかを決定するための判断がなされる。例外または分岐セクション１３５の決定に基づく分岐の誤予測があれば、その場合誤って予測された分岐命令に続いて割当てられたリオーダバッファ１１４の内容は廃棄される。
【００３０】
図２は、命令キャッシュ１０４のブロック図である。「可変バイト長命令に特に適した、プリデコードされた命令キャッシュおよびそのための方法（Ｐｒｅ－Ｄｅｃｏｄｅｄ Ｉｎｓｔｒｕｃｔｉｏｎ Ｃａｃｈｅ ａｎｄ Ｍｅｔｈｏｄ Ｔｈｅｒｅｆｏｒ Ｐａｒｔｉｃｕｌａｒｙ Ｓｕｉｔａｂｌｅ ｆｏｒ Ｖａｒｉａｂｌｅ Ｂｙｔｅ－Ｌｅｎｇｔｈ Ｉｎｓｔｒｕｃｔｉｏｎｓ）」と題された連続番号＿＿＿＿＿で出願日が＿＿＿＿＿の米国出願が、引用により援用されており、命令キャッシュ１０４の構造および動作をより詳細に述べている。
【００３１】
命令キャッシュ１０４は線形にアドレス指定された１６キロバイトのフォーウェイセットアソシアティブキャッシュである。各セットは２５６個のエントリを含み、各エントリは１６バイト命令ブロックと、線形アドレスタグと、次に予測実行される分岐情報とを含む。命令キャッシュ１０４はキャッシュ制御装置１７０と、アドレス回路１７２と、プリデコード回路１７４と、キャッシュアレイ１７６とを含む。キャッシュ制御装置１７０は、命令キャッシュ１０４の様々な動作を統制するための制御信号を与える。アドレス回路１７２は、分岐セクション１３５から受取られる論理目的プログラムカウンタ（ＴＡＲＧＥＴ ＰＣ）に基づく線形フェッチプログラムカウンタ（ＦＥＴＣＨ ＰＣ）、または線形タグアレイ１８２からの線形タグ、およびサクセサアレイからの対応するサクセサ情報を発生する。アドレス回路１７２はまた、アドレスの発生と、外部メモリからのプリフェッチ命令に関連のＸ８６保護検査とをもたらす。アドレス回路１７２は論理アドレスと線形アドレスとの間の変換を行なうための変換回路として機能する。プリデコード回路１７４はＩＡＤバス１０２を介してプリフェッチされたＸ８６命令バイトを受取り、各Ｘ８６命令バイトのためのプリデコードビットを割当て、プリデコードされたＸ８６命令バイトをキャッシュアレイ１７６の中に書込む。キャッシュアレイ１７６はプリデコード回路１７４から受取られた命令をストアし、線形ＦＥＴＣＨ ＰＣ信号によってアドレス指定されると、これらの命令をバイトキュー１０６に与える。
【００３２】
キャッシュアレイ１７６は３つの主要なアレイに編成される。それらはすなわち、命令キ

ャッシュストアアレイ１８０と、線形タグアレイ１８２と、サクセサアレイ１８４とである。命令キャッシュストアアレイ１８０は、１６バイト命令をストアする。線形タグアレイ１８２は、命令に対応する線形アドレスタグをストアする。サクセサアレイ１８４は、命令に対応する、分岐予測をサポートするために用いられる予測された分岐情報をストアする。これらのアレイの各々は、アドレス回路１７２によって提供される、線形ＦＥＴＣＨ　ＰＣアドレスによってアドレス指定されている。

【００３３】

図３を参照して、命令デコーダ１０８は２段のパイプライン化されたデコーダであって、これはバイトキュー１０６からプリデコードされたＸ８６命令バイトを受取り、それらをそれぞれのＲＯＰのシーケンスに変換し、複数個のディスパッチ位置からＲＯＰを迅速にディスパッチする。多数命令発行のための機会を最大限にするため、最も単純な命令についてはこの変換はハードワイヤード高速変換経路内で扱われる。好ましい実施例では、ハードワイヤード高速変換経路は３つのＲＯＰまたはそれ未満のものにマッピングするＸ８６命令に適用される。３つより多くのＲＯＰを必要とする命令およびあまり使われない命令は、マイクロコードＲＯＭ内に含まれるマイクロコードシーケンスによって扱われる。Ｘ８６命令がマイクロコードＲＯＭにマッピングしたときには常に、プリデコードされたＸ８６命令情報が複数のディスパッチ位置において二重にされ、各ディスパッチ位置が独立的かつ他のディスパッチ位置と並行に動けるようにする。

【００３４】

第１のデコード段では、命令デコーダ１０８はＸ８６のプリデコードされた命令のためのＲＯＰが高速経路において発生されるべきかマイクロコードＲＯＭ経路において発生されるべきかを決定する。ＲＯＰシーケンスのための情報は、ＲＯＰを４つまで用いて１サイクルあたり４つまでのＸ８６命令のための高速経路論理によって発生されるか、または１つのプリデコードされたＸ８６命令のためのマイクロコードＲＯＭから読出されるかのいずれかである。高速経路およびＲＯＰを発生することが要求されているマイクロコードＲＯＭ経路からの情報は、ＲＯＰがそこに行くことが指定されている機能ユニットのタイプ、機能ユニットで実行されるべき特定の単純なＲＩＳＣのような命令、ＲＯＰのためのソースおよび行先ポインタ、ＲＯＰのサイズ情報、ロードまたはストアＲＯＰの場合のアドレス指定モード、ならびにもしあればＲＯＰのための命令からの即値フィールドを含む。好ましくは、マイクロコードＲＯＭアクセスは高速経路デコード機能とは混合されない。これによりマイクロコードＲＯＰをシフトしなければならなくなることが回避される。第２のデコード段では、命令デコーダ１０８は高速経路またはマイクロコードＲＯＭのいずれかからのＲＯＰ情報を選択して増加させ、完全なＲＯＰをもたらし、これらは機能ユニット１３１～１３６において実行される。

【００３５】

命令デコーダ１０８はまた、完全にディスパッチされプリデコードされたＸ８６命令がバイトキュー１０６の外へシフトされ、次のディスパッチされていないまたは部分的にディスパッチされているプリデコードされたＸ８６命令が「キューの先頭」にシフトされるように、バイトキュー１０６におけるシフトを制御する。命令デコーダ１０８はまた、バイトキュー１０６における問題を検出し、マイクロコードエントリポイントを強制することによって、命令キャッシュ１０４を適切なように間接的に再び導く。命令デコーダ１０８はまた、命令キャッシュ１０４内のキャッシュ再充填論理によって、ならびに分岐セクション１３５における例外およびマイクロ分岐誤予測論理によって開始されるマイクロコードＲＯＭエントリポイントを受入れる。分岐セクション１３５によって発生されるマイクロコードエントリポイントは、ＸＴＡＲＧＥＴバス１０３を介して命令デコーダ１０８に伝えられる。命令デコーダ１０８はまた、各ＲＯＰに対応するデコードプログラムカウンタ値を発生する。

【００３６】

命令デコーダ１０８は、ＲＯＰマルチプレクサ２００と、４つのディスパッチ位置回路２１０、２１１、２１２、２１３と、４つのＲＯＰセレクタ回路２２０、２２１、２２２、

２２３と、ＲＯＰ共有回路２２６とを、命令デコード制御回路２３０、ＲＯＭ制御２４０、およびデコードプログラムカウンタ回路２４２とともに含む。ＲＯＰマルチプレクサ２００（ＲＯＰＭＵＸ）は、バイトキュー１０６のエントリを４つのディスパッチ位置２１０～２１３に導く。各ディスパッチ位置は、それぞれの高速コンバータ２５０、２５１、２５２、２５３（それぞれＦＡＳＴＣＯＮＶ０、ＦＡＳＴＣＯＮＶ１、ＦＡＳＴＣＯＮＶ２、およびＦＡＳＴＣＯＮＶ３）と、それぞれの共通段２６０、２６１、２６２、２６３（それぞれＩＣＯＭＭＯＮ０、ＩＣＯＭＭＯＮ１、ＩＣＯＭＭＯＮ２、およびＩＣＯＭＭＯＮ３）と、それぞれのマイクロコードＲＯＭ２７０、２７１、２７２、２７３（それぞれＭＲＯＭ０、ＭＲＯＭ１、ＭＲＯＭ２、およびＭＲＯＭ３）とを含む。マイクロコードＲＯＭ２７０～２７３は、マイクロコードＲＯＭ命令デコード制御装置２４０（ＩＤＥＣＣＮＴＬ）によって制御される。ＲＯＰマルチプレクサ２００、ディスパッチ位置回路２１０～２１３、マイクロコードＲＯＭ命令デコード制御装置２３０、およびデコードプログラムカウンタ回路２４２は、一般に命令デコーダ１０８の第１段を形成する。命令デコーダ１０８の第２段は一般に、ＲＯＰセレクタ回路２２０～２２３（それぞれＲＯＰＳＥＬＥＣＴ０、ＲＯＰＳＥＬＥＣＴ１、ＲＯＰＳＥＬＥＣＴ２、およびＲＯＰＳＥＬＥＣＴ３）、およびＲＯＰ共有回路２２６（ＲＯＰＳＨＡＲＥＤ）において実現される。

【００３７】

命令デコーダ１０８は命令デコード制御装置２３０によって制御される。命令デコード制御装置２３０は、ＴＡＤバス１１８上に命令タイプ情報をもたらし、現在のディスパッチウインドウ内のＲＯＰのうちいくつがＲＩＳＣコア１１０によって受入れられ得るかを予測し、この予測を鑑みてどのようにバイトキュー１０６をシフトさせるか命令キャッシュ１０４に知らせ、バイトキュー１０６の先頭におけるプリデコードされたＸ８６命令のためにまだこれからディスパッチされるべきＲＯＰの数をＲＯＰマルチプレクサ２００に知らせ、マイクロコードおよび制御ＲＯＭにアクセスするなど、一般的な制御機能を提供するために、組合せ論理またはプログラマブルアレイ論理において実現される論理を含む。これらの機能を提供するために、命令デコード制御装置２３０はＲＩＳＣコア１１０の機能ユニット１３１～１３６およびプロセッサ１００の他のユニットから様々な情報を受取る。

【００３８】

各高速コンバータ２５０、２５１、２５２、および２５３は、「単純な」Ｘ８６命令（すなわち３つ以下のＲＯＰにマッピングするもの）に対して多くのタイプの高速変換を行なう。各ディスパッチ位置における高速コンバータはＸ８６命令を、そのＸ８６命令を実行するのに必要なＲＯＰのシーケンスのそれぞれ１つ（１つのシーケンスは１つまたはそれ以上のＲＯＰである）に変換し、ある一定のプレフィックスおよびＳＩＢバイトのためのＲＯＰの動作を変形する。高速コンバータ２５０～２５３はプログラマブルアレイ論理として実現されるか、第２のデコード段の位相１において前にラッチされた組合せ論理しとて実現されるかのいずれかである。

【００３９】

めったに使われないＸ８６命令および実行するのに３つより多くのＲＯＰからなるＲＯＰシーケンスを必要とするＸ８６命令は、マイクロコードＲＯＭにマッピングされる。一般に３つより多くのＲＯＰにマッピングするＸ８６命令は、ＣＡＬＬゲート命令、ＳＴＲＩＮＧ移動命令、および卓越した浮動小数点ルーチンなどの複合対話型命令である。この場合、命令デコーダ１０８内のＲＯＭ制御装置２４０は、エントリポイントとして知られるマイクロコードＲＯＭアドレスを形成し、このエントリポイントに基づき、マイクロコードＲＯＭ２７０～２７３にストアされたマイクロコードＲＯＭ命令からのＲＯＰシーケンスを、１サイクルあたり４つのＲＯＰにおいて読出す。各マイクロコードＲＯＭ２７０、２７１、２７２、および２７３は、高速コンバータにおいては変換可能でないＸ８６命令を扱うための１０２４×５９ＲＯＭアレイである。

【００４０】

ディスパッチ位置２１０、２１１、２１２、および２１３もまた、それぞれの共通段２６

０、２６１、２６２、および２６３を含む。各共通段はパイプライン段として機能する。各共通段２６０、２６１、２６２、および２６３はそれぞれの高速コンバータおよびマイクロコードＲＯＭと関連づけられている。命令が高速経路命令であるかマイクロコードＲＯＭ命令であるかに関わりなくある特定のやり方で効率的に扱うことができ、かつマイクロコードＲＯＭにおける発生を必要とはしないＸ８６命令変換動作は部分的に、変換を必要としない共通データとともに、高速コンバータまたはマイクロコードＲＯＭにおける様々な段の命令部分の処理を伴うステップでそれぞれの共通段を介して、実行されパイプライン化される。各共通段はまた、高速経路命令およびＭＲＯＭ命令の双方によって用いられる内部命令デコーダアドレス計算を行なうために使用されるレジスタポインタを追跡する。

【００４１】

各セレクタ回路２２０、２２１、２２２、および２２３は、高速コンバータおよびパイプライン段の出力またはマイクロコードＲＯＭおよびパイプライン段の出力のいずれかを選択し、ディスパッチ情報を発生する。各セレクタ回路はまた、即値フィールド定数値を選択し、機能ユニット１３１～１３６への即値アドレスまたは定数を駆動する。ＲＯＰ共有ユニット２２６はディスパッチ位置２１０～２１３の各々におけるＲＯＰ選択回路からの情報に応答し、すべてのディスパッチ位置によって共有されるリソースのためのディスパッチ情報を発生する。

デコードプログラムカウンタ回路２４２はバイトキュー１０６から送られるＸ８６命令の各々における論理プログラムカウンタ値を追跡する。バイトキュー１０６はプリフェッチストリーム内に非順次フェッチを検出すると、その目的のバイトおよび新しい目的アドレスをデコードプログラムカウンタ回路２４２に示す。デコードプログラムカウンタ回路２４２は次にデコードプログラムカウンタ値を発生し、これは分岐セクション１３５ならびにディスパッチされた各ＲＯＰのためのデコードプログラムカウンタ値における下位ビットおよびキャリービットに与えられる。ＬＰＣビットおよびキャリービットは、命令デコーダ１０８によってリオーダバッファ１１４に与えられる。

【００４２】

デコードＰＣ回路２４２レジスタ内のデコードＰＣレジスタの中に維持されるＲＯＰデコードプログラムカウンタ値が、Ｘ８６アーキテクチャにおける論理アドレスとして維持されるので、セグメント間分岐を行なうときには、命令キャッシュ１０４にアクセスするのに用いられる線形アドレスからデコードＰＣレジスタを更新するために必要な論理アドレスへの変換が必要である。これは命令キャッシュ１０４において、まず線形アドレスからセグメントベースポインタを減算して取除き、論理アドレスを得ることによってなし遂げられる。この３２ビットの論理アドレスは次に命令デコーダ１０８のデコードプログラムカウンタ回路２４２に駆動され、これは次にＲＯＰデコードプログラムカウンタ値を発生する。セグメント間分岐を行なわない場合、デコードＰＣ値はデコードプログラムカウンタ回路２４２で更新される。

【００４３】

分岐に続く順次命令については、デコードプログラムカウンタ回路２４２は開始位置と終了位置との間でバイトキュー内のＸ８６バイトの数をカウントし、これを開始論理デコードプログラムカウンタ値に加算して、次のデコードプログラムカウンタ値を発生する。

【００４４】

命令デコーダ制御装置２３０は、命令デコーダ１０８の様々なユニットの動作を制御するための適切なステートマシンのいずれかである。

【００４５】

図４を参照して、デコードプログラムカウンタ発生器回路２４２は、キュー３００と、スキャン論理３０２と、上位部分プログラムカウンタラッチ３０４と、下位部分プログラムカウンタラッチ３０６と、上位部分プログラムカウンタ発生器回路３０８と、下位部分プログラムカウンタ発生器回路３１０と、プログラムカウンタ値セレクタ回路３１２とを含む。

【００４６】
キュー３００は３２ビット非順次プログラムカウンタ値を有効ビットと同様に保持し、非順次プログラムカウンタ値のビット４：３１を上位部分プログラムカウンタラッチ３０４に与える。キュー３００は非順次プログラムカウンタ値のビット０：３をスキャン論理３０２に与える。
【００４７】
スキャン論理３０２は命令デコーダ１０８によって処理されている４つのＲＯＰに対応する４つの４ビットオフセット値を下位プログラムカウンタ発生器回路３１０に与える。４つのオフセット値の各々は、キュー１０６内にストアされるＸ８６命令とキュー３００内にストアされる現在のデコードプログラムカウンタ値との間のオフセットを表わす。キュー１０６内にストアされる各Ｘ８６命令は、１つより多くのＲＯＰにマッピングしてもよい。同じサイクル内で命令デコーダ１０８によって処理されている４つのＲＯＰが１つのＸ８６命令に対応していてもよいため、４つのオフセット値はすべて同じ値を有していてもよい。これらのオフセット値は、キュー１０６の先頭における命令のための開始ビットとキュー１０６内の次の命令のための開始ビットとを選択するスキャン論理３０２によって発生される。オフセットは、同じサイクル内の命令デコーダ１０８によって処理されている４つのＲＯＰに対して累積される。すなわち、オフセットは第１のＲＯＰとオフセットがそのために発生されているＲＯＰとの間のバイトの数に等しく、これは２つのＲＯＰ間にいくつのＲＯＰがあるかには関わりがない。
【００４８】
下位プログラムカウンタ発生器回路３１０は、加算器３２０～３２３とマルチプレクサ３２６とを含む。各加算器３２０、３２１、３２２、３２３は、４ビットオフセット値の１つを受取る。加算器３２０～３２３は、これらの４ビットオフセット値をラッチ３０６からの現在のデコードプログラムカウンタ値におけるビット０：３に加算し、次に来るかもしれないプログラムカウンタ値における下位ビットすなわちビット０：３を得る。次に来るかもしれないプログラムカウンタ値の各々は、ディスパッチされるべきＲＯＰに対応する。次に来るかもしれないプログラムカウンタ値のビット０：３は、マルチプレクサ３２６に与えられる。マルチプレクサ３２６はこれらの次に来るかもしれないプログラムカウンタ値をマルチプレクスして、対応するＲＯＰが命令デコーダ１０８によってディスパッチされる際にどの命令がキュー３００から取除かれるかに基づき、次のプログラムカウンタ値をデコードプログラムカウンタラッチ３０６に与える。ラッチ３０６はマルチプレクスされた値を現在のデコードプログラムカウンタ値としてストアし、次のデコードプログラムカウンタ値を得るにあたって用いる。
【００４９】
下位プログラムカウンタ発生器回路３１０は、各加算器からのキャリービットを上位プログラムカウンタ発生器回路３０８のキャリーセレクタ３３４に与える。下位プログラムカウンタ発生器回路３１０はまた、これらのキャリービットをカウンタ値選択回路３１２にも与える。
【００５０】
上位プログラムカウンタラッチ３０４は、次のプログラムカウンタ値のビット４：３１を上位プログラムカウンタ発生器回路３０８に与える。上位プログラムカウンタ発生器回路は、アドレスインクリメンタ３３０と、マルチプレクサ３３２と、キャリーセレクタ３３４とを含む。インクリメンタ３３０は現在のデコードＰＣ値の上位２８ビットを受取り、増分されたデコードＰＣ値、すなわち上位２８ビットが１だけ増分された値を提供する。この増分されたデコードＰＣ値は、やはり現在のデコードＰＣ値を受取るマルチプレクサ３３４に与えられる。マルチプレクサ３３２はこれら２つの値のうち１つを、キャリーセレクタ回路３３４の制御下で次のデコードＰＣ値として提供する。キャリーセレクタ回路３３４は下位部分プログラムカウンタ発生器回路３１０からのキャリービットを用いて、キャリービットのうちどれがディスパッチされるべき次のＲＯＰに対応するかを判断し、かつこのキャリービットの値を用いてマルチプレクサ３３２を制御する。ディスパッチさ

れるべき次のＲＯＰのキャリービットが、キャリーを示す活性であれば、次のデコードＰＣ値の上位２８ビットには増分されたデコードＰＣ値が用いられる。そうでなければ、次のデコードＰＣ値の上位２８ビットには現在のデコードＰＣ値の上位２８ビットが用いられる。

【００５１】

カウンタ値選択回路３１２は、マルチプレクサ３３６とディスパッチ割当マルチプレクサ３３８を含む。マルチプレクサ３３６は、キュー３００からどの命令が、命令デコーダ１０８によって対応するＲＯＰがディスパッチされる際に取除かれるかに基づき、デコードプログラムカウンタ値（ＤＰＣ）のビット０：３として、次に来るかもしれない４つの下位プログラムカウンタ値のうち１つを選択する。マルチプレクサ３３６はこのデコードプログラムカウンタ値をリオーダバッファ１１４に与える。ディスパッチ割当マルチプレクサ３３８は次の下位プログラムカウンタ値（ＬＰＣ）およびキャリー値（ＣＡＲＲＹ）を受取り、これらの値をディスパッチされる各ＲＯＰのためにリオーダバッファ１１４に与える。

【００５２】

上位プログラムカウンタ発生器回路３０８はデコードプログラムカウンタ値のビット４：３１をプログラムカウンタ値選択回路３１２に与える。プログラムカウンタ値選択回路３１２はこれらのビットをマルチプレクサ３３６からの選択された下位ビット０：３と組合せ、完全な、すなわちビット０～３１の、デコードプログラムカウンタ（ＤＰＣ）値を提供する。

【００５３】

図５を参照して、リオーダバッファ１１４は循環先入れ先出し（ＦＩＦＯ）回路であって、推論的に実行されるＲＯＰの相対的な順序を追跡する。リオーダバッファの記憶位置は、それぞれリタイアの結果をレジスタファイル１１２に送り、命令デコーダ１０８からの不整合なＲＯＰを受取るために先頭および末尾キューポインタを用いて、動的に割当てられる。命令がデコードされると、その結果値はリオーダバッファ１１４内の位置または行先に割当てられ、その行先レジスタ番号はこの位置と関連づけられる。依存性のない後続の命令については、関連づけられたＡおよびＢオペランドバスがレジスタファイル１１２から駆動される。しかしながら、後続する命令が依存性を有しており、名前を変更された行先レジスタを参照してその中にストアされていると考えられる値を得る場合、リオーダバッファ１１４内でエントリがアクセスされる。結果をその中で得ることができれば、それはオペランドバスにもたらされる。結果が得られない場合、このリオーダバッファエントリを識別するタグが、ＡおよびＢオペランドタグバス１１６の１本のオペランドタグバスに与えられる。結果またはタグはオペランドバスまたはオペランドタグバスを介して機能ユニットにもたらされる。機能ユニットにおける実行の完了により結果が得られた場合、その結果およびそれらのそれぞれの結果タグは、バス５本分の幅の結果バスと５本の結果タグバス１４０とを介してリオーダバッファ１１４および機能ユニットの予約ステーションにもたらされる。

【００５４】

リオーダバッファ１１４は実行プログラムカウンタレジスタを更新するためにＲＯＰと関連しているプログラムカウンタ値を保持する圧縮形式を用いる。この圧縮形式に必要なのは、下位４バイトとビット増分すなわちキャリーの表示のみである。順次的Ｘ８６バイトについては、付加されるバイトの数は１５個以下であり、非順次的フェッチについては、分岐が成功裏に完了すると、新しい分岐目的が結果バスに駆動され、それにより命令デコーダ１０８からリオーダバッファ１１４への新しい分岐位置の書込みは必要でなくなる。

【００５５】

リオーダバッファ１１４は、リオーダバッファ制御およびステータス回路４００と、リオーダバッファアレイ４０２と、リオーダバッファオペランドバスドライバ４０４とを含む。リオーダバッファアレイ４０２は一時記憶メモリであって、リオーダバッファ制御およびステータス回路４００によって制御される。命令デコーダ１０８がＲＯＰをディスパッ

チすると、命令デコーダ１０８は行先ポインタ（ＤＥＳＴ　ＲＥＧ）バスの１本に信号をもたらす。リオーダバッファ制御およびステータス回路４００はその後、リオーダバッファアレイ４０２のエントリを割当てる。リオーダバッファアレイ４０２の各エントリは、一時行先レジスタを提供するために割当てられ、ＲＯＰが完了するとそこに結果が書込まれる。リオーダバッファおよびステータス回路４００内の実行プログラムカウンタレジスタの中にストアされる実行ＰＣ値は、命令がリタイアされると更新される。複数のリオーダバッファエントリが、レジスタファイル１１２内の単一のレジスタに対応することができ、それによりレジスタの一時的な名前変更に備えることができる。リオーダバッファ制御およびステータス回路４００は、完了されたＲＯＰがいつレジスタ１１２内にリタイアされ得るかを決定する。命令がリタイアされると、実行プログラムカウンタレジスタは更新される。

【００５６】

リオーダバッファ制御およびステータス回路４００は命令デコーダ１０８からのＡおよびＢオペランドポインタを受取る。リオーダバッファ制御およびステータス回路４００はこれらのポインタを用いてリオーダバッファアレイ４０２内のエントリに問合わせをし、ディスパッチされたＲＯＰの実行に必要なオペランドがリオーダバッファアレイ４０２内に含まれているかどうかを判断する。

【００５７】

図６を参照して、リオーダバッファアレイ４０２は、レジスタファイルにライトバックされるべき送られたオペランドの結果を、その結果がもはやプロセッサの推論的状態の一部ではない場合に一時的にストアするための１６個のエントリを含む。１６個のリオーダバッファエントリの各々は、４１ビット結果フィールド（ＲＥＳＵＬＴ）、９ビット行先ポインタフィールド（Ｐ）、４ビット下位プログラムカウンタ値フィールド（ＬＰＣ）および１１ビット浮動小数点オペレーションコードフィールド（ＦＰＯＦ）、１１ビット浮動小数点フラグレジスタフィールド（ＦＰＦＲ）、ならびに２４ビット制御およびステータスフィールド（ＣＳ）を含む。

【００５８】

結果フィールドは機能ユニットからの結果データをストアする。４１ビット結果フィールドは浮動小数点結果を２つのリオーダバッファエントリ内にストアできるようにする。整数結果は４１ビットのうちの３２ビットにストアされる。

【００５９】

各リオーダバッファエントリの行先ポインタフィールドは、ライトバック中に書込まれるべき命令の実行よりの結果のための、レジスタファイル１１２内の行先レジスタアドレスを指定する。リオーダバッファ制御およびステータス回路４００は行先レジスタ（ＤＥＳＴ　ＲＥＧ）バスを介して命令デコーダ１０８から行先ポインタを受取り、そのポインタをリオーダバッファアレイ４０２の行先ポインタフィールドにロードし、ポインタと関連するＲＯＰがリタイアされているときのためにそれをストアさせる。ＲＯＰがディスパッチされると、リオーダバッファ１１４はリオーダバッファアレイ４０２のすべての行先ポインタフィールドをスキャンすることによって依存性チェックを行ない、行先ポインタとＡまたはＢオペランドとの間に整合があるかどうかを判断する。整合があるということは、データ依存性があるということを示す。

【００６０】

浮動小数点オペレーションコードフィールドは、リオーダバッファエントリに割当てられた浮動小数点オペレーションコードのビットのサブセットにセットされる。浮動小数点フラグレジスタフィールドは浮動小数点動作によりもたらされる浮動小数点フラグの状態をストアする。浮動小数点フラグは浮動小数点機能ユニット１３６により検出された、精度、アンダーフロー、オーバーフロー、ゼロ分割、正規化を取止められたオペランド、および無効オペランドエラーに関連の情報をストアする。整数オペランドについては、整数演算によりもたらされるフラグは４１ビット結果フィールドの上位ビット内に保持されるため、対応するフラグフィールドは必要ではない。

【００６１】
ステータスおよび制御フィールドは、ＲＯＢエントリのステータス、たとえばリオーダバッファエントリが割当てられているか、分岐が誤って予測されているか、または命令の実行が例外またはエラー条件をもたらしているのではないか、ということを示すビットを含む。ステータスおよび制御フィールドはまた、実行プログラムカウンタ値を発生することに関連のビットをも含む。より特定的には、ステータスおよび制御フィールドは、各ＲＯＰのためにデコードＰＣ発生器回路によって与えられるキャリービット、セットされるとＲＯＰがある特定のＸ８６命令についてＲＯＰのシーケンスのうちの最後のＲＯＰであるということを示す出口ビット、および結果が有効であることを示して命令が完了していることを表わす完了（または有効）ビットを含む。出口ビットは、実行プログラムカウンタレジスタ内にストアされた実行ＰＣ値をいつ更新すべきかを示す。実行プログラムカウンタ値が更新されるのは、特定のＸ８６命令のためのＲＯＰシーケンスが完了したときのみである。

【００６２】
再び図５を参照して、ＲＯＰを実行して結果を発生する機能ユニットは、その結果を行先タグを用いて適切なリオーダバッファエントリに導く。ディスパッチの際、機能ユニットに接続される４つの行先タグバスのうちの１つに行先タグがセットされる。機能ユニットは結果バス１４０のうち１本にその結果を位置づけ、対応する結果タグおよびステータスバス上に行先タグを位置づける。リオーダバッファ制御およびステータス回路４００は結果タグバスからの結果タグを受取り、このタグをリオーダバッファアレイ４０２内のエントリをアドレス指定するのに用いる。結果バスおよびステータスバスのうち１つに対応する結果バスの１本の上の信号が、結果タグ値により指定されるリオーダバッファアレイエントリにロードされる。

【００６３】
リオーダバッファ制御およびステータス回路４００はまた、データ依存性についてもチェックし、未解決の依存性が起こると、オペランドバスと関連のＡおよびＢオペランドタグバスを用いてオペランドにタグをつける。タグは機能ユニットの予約ステーションによって用いられるために、依存性が基づいている結果を識別する。リオーダバッファ１１４はデータ依存性が検出されるとレジスタファイル読出を無効にする。リオーダバッファ制御およびステータス回路４００は、ディスパッチの際にデータ依存性を検出した場合、レジスタファイルオペランドバスドライバ４０４に与えられる無効化バスのビットをセットすることによって、リオーダバッファアレイ４０２のエントリ内におけるいかなる依存性オペランドの読出動作をも無効にする。無効化バスは、各オペランドバスのための無効化信号を含む。

【００６４】
リオーダバッファ１１４が、ソースオペランドデータが利用可能でないデータには依存しておらず、レジスタファイル１１２またはリオーダバッファ１１４から得ることができると判断した場合、オペランドデータはオペランドバス１１６を介して機能ユニット予約ステーションに送られる。読出ポインタのレジスタアドレスがエントリの行先ポインタアドレスに一致した場合、データはリオーダバッファ１１４内にアドレス指定される。ＡまたはＢオペランド読出ポインタ（Ａ－ＲＤＰＴＲまたはＢ－ＲＤＰＴＲ）の読出ポインタは、リオーダバッファ制御およびステータス回路４００を介してリオーダバッファアレイ４０２をアドレス指定し、オペランドデータをリオーダバッファオペランドバスドライバ４０４に与える。リオーダバッファオペランドバスドライバ４０４は、オペランドデータをＡおよびＢオペランドバス１１６に駆動する。

【００６５】
ＲＯＰは、リオーダバッファ制御およびステータス回路４００がリオーダバッファアレイ４０２からの結果をレジスタファイル１１２に書込み、その結果をライトバックバスの１つに位置づけて行先ポインタをライトバックバスに対応する書込ポインタに書込むにつれ、リタイアされる。書込ポインタはレジスタファイル１１２内のレジスタアドレスを指定

し、リタイアされた結果を受取る。ＲＯＰがリタイアされると、リオーダバッファオペランドバスドライバ４０４はリオーダバッファアレイ４０２のＬＰＣフィールドを用いて実行プログラムカウンタ値を発生する。

【００６６】

リオーダバッファステータスおよび制御回路４００はリオーダバッファ１１４の様々な条件を検出し、リオーダ条件バスを用いてプロセッサ１００における様々な回路へ条件信号をリレーする。リオーダ条件バスの信号は４つのＡおよびＢオペランドの各々のためにレジスタファイル１１２によって受取られ、リオーダバッファ１１４がいつ送られたオペランドを実行のために与えるかを示す。充満、空、または単一エントリのリオーダバッファ条件を反映するリオーダバッファ条件バス上の他の信号は、命令デコーダ１０８によって受取られる。

【００６７】

図７を参照して、ＲＯＢ制御およびステータス回路４００内に含まれる実行プログラムカウンタ発生器回路５９８は、下位プログラムカウンタ発生器６００と、上位プログラムカウンタ発生器下位６０２と、実行プログラムカウンタ制御装置６０４とを含む。下位プログラムカウンタ発生器６００は、エントリマルチプレクサ６１０と、分岐マルチプレクサ６１２と、レジスタ６１４とを含む。エントリマルチプレクサ６１０は、最も古い４つのリオーダバッファエントリ４０２にストアされる４つの命令のためのプログラムカウンタ値の下位４ビットの部分を受取る。エントリマルチプレクサ６１０は、これらの下位プログラムカウンタ値の１つを分岐マルチプレクサ６１２に与え、分岐マルチプレクサ６１２はまた、分岐プログラムカウンタ値の下位ビットをも受取る。分岐マルチプレクサ６１２はレジスタ６１４に４ビット下位実行プログラムカウンタ値を与え、レジスタ６１４はこの値を次の実行プログラムカウンタ値として提供する。エントリマルチプレクサ６１０、分岐マルチプレクサ６１２、およびレジスタ６１４は、実行プログラムカウンタ制御回路６０４により制御される。

【００６８】

上位プログラムカウンタ発生器６０２は、エントリマルチプレクサ６２０と、分岐マルチプレクサ６２２と、発生上位増分回路６２４と、加算器回路６２６と、レジスタ６２８とを含む。エントリマルチプレクサ６２０は、下位の４つのリオーダバッファエントリにストアされる４つの命令からの分岐目的値を受取り、分岐目的マルチプレクサがこれらのエントリのうち１つを実行プログラムカウンタ制御回路６０４の制御下にある次の分岐目的として与える。次の分岐目的値は、２８ビットの上位分岐目的値と、４ビットの下位分岐目的値とを含む。上位分岐目的値は分岐マルチプレクサ６２２に与えられ、分岐マルチプレクサ６２２は、レジスタ６２８からの２８ビットの上位実行プログラムカウンタ値をも受取る。分岐マルチプレクサ６２２はマルチプレクスされた予備的な上位実行プログラムカウンタ値を加算器回路６２６に与え、加算器回路６２６は、発生上位増分回路６２４からの増分信号をも受取る。加算器回路６２６はこの加算に基づき、上位実行プログラムカウンタ値をレジスタ回路６２８に与える。レジスタ回路６２８はクロックド上位実行プログラムカウンタ値を制御回路６０４の制御下で提供する。

【００６９】

制御装置６０４は下位の４つのリオーダバッファエントリの各々からの制御情報を受取る。この制御情報は各エントリのための分岐発生信号および各エントリからの更新プログラムカウンタ信号、ならびに各エントリのための有効信号および更新実行プログラムカウンタ信号を含む。この制御情報に基づき、実行プログラムカウンタ制御装置６０４は上位および下位実行プログラムカウンタ値の発生を制御する。分岐発生信号は、分岐が行なわれたかどうかを示すものであって、分岐が行なわれたときに活性であり、それにより分岐アドレスが用いられるべきであることを示す。有効信号は、ＲＯＰの結果が機能ユニットから返されたこと、すなわち有効結果が存在するということを示し、分岐結果が返されたときに活性である。更新実行プログラムカウンタ値信号は、実行プログラムカウンタ値をいつ更新すべきかを示し、ＲＯＰがリタイアされたときに活性である。

【００７０】
下位プログラムカウンタ発生器６０２は新しい下位実行プログラムカウンタ値を発生するのにマルチプレクサを用いるので、より大きいパイプラインのための実行プログラムカウンタ値を、下位プログラムカウンタ値が与えられるマルチプレクサの幅を広くするだけで発生することが可能である。
【００７１】
図８を参照して、Ｘ８６バイトの順次的ストリームを実行するプロセッサ１００のタイミングが示される。この例では、予測実行された経路は実際にとられており、命令キャッシュ１０４から直接得ることができる。動作中、マイクロプロセッサパイプラインの効果的な実行のパイプライン段は５つある。
【００７２】
フェッチサイクルは実行の第１段である。フェッチクロックサイクルは命令キャッシュ１０４の中で費やされる。命令キャッシュ１０４はクロックサイクルのＰＨ１の間に新しいフェッチプログラムカウンタ値を形成し、次にフェッチクロックサイクルのＰＨ２においてキャッシュアレイ１８０および１８２にアクセスする。フェッチプログラムカウンタ値（ＦＰＣ（３１：０））は、ストアアレイと並行して線形命令キャッシュタグアレイにアクセスする。フェッチサイクルのＰＨ２の終わりの方で、線形タグがフェッチプログラムカウンタの線形アドレスに一致するかどうかの判断が行なわれる。一致していれば、予測実行されたバイトはバイトキュー１０６に送られる。
【００７３】
タグおよびストアアレイへのアクセスに加えて、フェッチプログラムカウンタ値はまた、命令キャッシュ１０４の分岐予測アレイ１８４にもアクセスする。分岐予測アレイ１８４はＸ８６バイトのうちどれが予測実行されるかを識別し、かつ次の予測実行されるブロックが順次的であるか非順次的であるかを識別する。フェッチサイクルのＰＨ２においてもアクセスされるこの情報は、現在フェッチされているブロックのどのバイトが有効なバイトとしてバイトキュー１０６に駆動されるかを決定する。
【００７４】
加えて、フェッチクロックサイクルの間、バイトキュー１８６は前にフェッチされているがまだ機能ユニットに発行されてはいないＸ８６バイトを保持していてもよい。その場合、命令キャッシュ１０４にバイト充填位置が示され、現在のサイクルの第１の予測されたバイトが、保持されているＸ８６バイトを超えて現在フェッチされているバイトを充填するためにバイトキュー１０６が保持しているバイトの数に等しい量だけシフトされる。
【００７５】
分岐予測情報はフェッチサイクルのＰＨ２において発生するため、プリフェッチされるべき次のブロックは順次的でも非順次的でもあり得る。いずれの場合も、再びアレイにアクセスするクロックサイクルは１つある。したがって、分岐予測アレイにより、ブロック外の分岐は次の順次的ブロックにアクセスするのと同じ相対的な性能を有する。
【００７６】
第１のデコードサイクルの始まりで、プリフェッチされ予測実行されたバイトは指定された充填位置においてバイトキュー１８６に駆動される。これは図８では第１のデコードサイクルのＰＨ１において起こるバイトキュー充填として示される。これらのバイトは次に、バイトキュー１０６の中の未決定のバイトのいずれとでもマージされる。
【００７７】
バイトキュー１０６はプリデコード状態の５ビットに加えて生のＸ８６バイトを、命令の境界がどこにあるかを示すために含む。バイトキューの先頭は常に次の予測実行されるＸ８６命令の始まりである。第１のデコード段のＰＨ１の中途で、バイトキュー１０６は命令キャッシュ１０４からのバイトの次のストリームを、バイトキュー１０６における既存のバイトとマージし、マージされたバイトのストリームを命令デコーダ１０８に提示する。命令デコーダ１０８は各命令がとるＲＯＰの数およびオペレーションコードの位置を決定し、これらのオペレーションコードを対応するＲＯＰ発行位置０～３（ここで０は次に

発行すべきＲＯＰである）と整列させる。

【００７８】

命令デコーダ１０８は命令の境界間のバイト数をカウントするか、命令キャッシュ１０４内に分岐を検出して、プログラムカウンタ値をその位置からフェッチされた第１のＸ８６バイトに付与することによって、バイトキュー１０８内のＸ８６命令の各々に対応するＲＯＰのためのデコードプログラムカウンタ値を維持する。オペレーションコードおよびＲＯＰ位置決め情報をバイトキュー１０６内にストアされた即値フィールドと同様に用いて、命令デコーダ１０８は第１のデコードサイクルにおけるＰＨ２および第２のデコードサイクルにおけるＰＨ１の間にデコードＰＣ値を静的に決定する。第２のデコードサイクルのＰＨ１の終わりまでには、レジスタ読出および書込ポインタはすべて解決され、動作が決定される。これは、読出書込ポインタ値のアサートとして図８に示される。

【００７９】

加えて、第２のデコードサイクルにおけるＰＨ１の間のタイミングは、次のクロック位相において発行するかもしれない対応するＲＯＰのためのリオーダバッファエントリを割当てている。したがって、最大４つまでの付加的なＲＯＰが各々ＰＨ１の間にリオーダバッファ１１４内の１６個のエントリのうちの１つに割当てられる。ＰＨ２の間、割当てられたすべてのＲＯＰのためのソース読出ポインタが、レジスタファイル１１２から読出され、一方で同時にリオーダバッファ１１４における推論的ＲＯＰのキューにアクセスする。

【００８０】

この、双方のアレイに対する同時のアクセスは、実際のレジスタファイルを用いるかリオーダバッファ１１４からオペランドまたはタグを進めるかについて、遅い選択をできるようにする。まずＰＨＩにおいて４つのＲＯＰエントリを割当て、次にＰＨ２においてリオーダバッファ１１４をスキャンすることによって、ディスパッチされている現在のＲＯＰ内の読出依存性はまだ推論的状態にある以前のＲＯＰすべてとともに同時に調査されてよい。

【００８１】

実行サイクルは標準的なスカラプロセッサの実行サイクルに比類するものである。ＲＯＰは上述の読出オペランドバスだけでなく専用のオペレーションコードバスをも介して発行される。これらのオペランドバスはタイミング図において実行サイクルのＰＨ１でディスチャージしているのを見ることができる。これは信号Ａ／Ｂ読出オペランドバスにおいて示される。

【００８２】

実行サイクルのＰＨ１の終わりの方で、機能ユニットはそれらに対してどのＲＯＰが発行されたかということと、それらにそれらの局所予約ステーションから発行される準備のできた未決定のＲＯＰがあるかどうかということを判断する。機能ユニットの予約ステーション内では、ＦＩＦＯが常に維持されており、最も古い命令が確実に最初に実行されるようになっている。機能ユニット内である命令の実行される準備が整っていれば、機能ユニットはＰＨ１の最後の方で実行を開始し、ＰＨ２を通じて実行を継続する。このタイミングはＡＬＵ１２２に関連している。

【００８３】

ＰＨ２の終わりでは、待ち時間１のすべての機能ユニットについて、結果が準備されており、機能ユニットは５つの結果バスのうちの１つのために仲裁を行なう。これはタイミング図では結果バス仲裁信号のアサートとして示される。実行サイクルのＰＨ２においてアクセスが許可されると、アクセスを許可された機能ユニットは結果サイクルのＰＨ１において割当てられた結果バスを駆動する。

【００８４】

結果サイクルはＲＩＳＣの４段パイプライン上での書込に比類される。結果サイクルは値を直接他の機能ユニットに送って実行させる。これはＲＩＳＣシステムにおけるＡＬＵの送信に大変似ている。マイクロプロセッサ１００は発行されているＲＯＰの推論的性質のために結果サイクルとリタイアサイクルとの双方を含んでおり、その命令がリタイアされ

るべき次の命令となるまでは、レジスタファイルへの直接の書込は行なえない。リオーダバッファ１１４および結果バス１４０の送信は、このサイクルをいかなる実行プログラムにとってもトランスペアレントなものとし、リオーダバッファ１１４への書込のオーバーヘッドについては立ち往生は全く起こらない。結果サイクルのＰＨ１では、リオーダバッファ１１４内の推論的ＲＯＰの位置は行先結果およびいかなるステータスとともにでも書込まれる。リオーダバッファ１１４内のこのエントリは次に、そのエントリが有効でありかつ割当てられているということを示すべくセットされる。したがって、このエントリが要求された場合、要求される読出アクセスにタグを送る代わりに、リオーダバッファ１１４は直接データを送る。

【００８５】

結果サイクルのＰＨ２では、新しく割当てられたタグはそれがそのソースオペランドの１つであることを要求する後続のＲＯＰによって検出され得る。これはタイミング図では、読出および書込ポインタを用いてのソースＡ／ＢオペランドバスへのＲＯＢタグ送信を介しての結果の直接送信として示される。

【００８６】

リタイアサイクルはパイプラインの最終段である。これは、実行プログラムカウンタ値が実行プログラムカウンタレジスタ内にストアされるサイクルである。リタイアサイクルのＰＨ１では、動作の結果はレジスタファイルに書込まれ、実行プログラムカウンタレジスタ内の実行プログラムカウンタ値は次にリタイアされるべき命令を指し示すべく更新される。リタイアサイクルのＰＨ１では、リオーダバッファ１１４内のエントリは割当を解除され、リオーダバッファ１１４から書込まれる。エントリが割当を解除されているため、結果に対する後に続く参照は、リオーダバッファ１１４からの推論的な読出よりもむしろレジスタファイル１１２からの読出において誘発される。レジスタファイル１１２からの読出は、プロセッサ１００の実際の状態を示す。

【００８７】

したがって、マイクロプロセッサ１００はパイプラインの異なった段において複数個のプログラムカウンタ値を用いる５段パイプラインを含む。このようなパイプラインとともにこれらのプログラムカウンタ値を用いることで、マイクロプロセッサ１００はクロックサイクル１つあたり４つまでの単純なＸ８６命令を発行できるようになる。

【００８８】

【その他の実施例】

その他の実施例は前掲の特許請求の範囲内に含まれる。

【００８９】

たとえば図９を参照して、フェッチプログラムカウンタ値を発生するには、下位プログラムカウンタ値をキャリー情報とともに提供するものである限り他の回路を用いてもよい。より特定的には、代替的なフェッチプログラムカウンタ回路７００は、キュー７０２、上位部分プログラムカウンタラッチ７０４、下位部分プログラムカウンタラッチ７０６、上位部分プログラムカウンタ発生器回路７０８、下位部分プログラムカウンタ発生器回路７１０、およびプログラムカウンタ値セレクタ回路７１２を含む。

【００９０】

キュー７０２は、３２ビット非順次プログラムカウンタ値を有効ビット値とともに保持しており、かつ非順次プログラムカウンタ値のビット４：３１を上位部分プログラムカウンタ発生器回路７０８に与え、非順次プログラムカウンタ値のビット０：３を下位部分プログラムカウンタ発生器回路７１０に与える。非順次プログラムカウンタ値はまた、下位部分プログラムカウンタラッチ７０６にも与えられる。

【００９１】

下位プログラムカウンタラッチは次のプログラムカウンタ値のビット０：３を下位プログラムカウンタ発生器回路７１０に与える。上位プログラムカウンタラッチ７０４は、次のプログラムカウンタ値のビット４：３１を上位プログラムカウンタ発生器回路７０８に与える。下位プログラムカウンタ発生器回路７１０は、キャリービットを上位プログラムカ

ウンタ発生器回路７０８およびプログラムカウンタ値選択回路７１２に与える。上位プログラムカウンタ発生器回路７０８は、プログラムカウンタ値のビット４：３１をプログラムカウンタ値選択回路７１２に与える。

【００９２】

下位プログラムカウンタ発生器回路７１０は、加算器７２０～７２３と、マルチプレクサ７３０～７３３と、マルチプレクサ７４０～７４３およびマルチプレクサ７５０とを含む。加算器７２０～７２３の１つ、マルチプレクサ７３０～７３３の１つ、およびマルチプレクサ７４０～７４３の１つの組合せが、次に来るかもしれないデコードプログラムカウンタ値における下位の４ビットを提供する。次に来るかもしれないデコードプログラムカウンタ値の各々は、キュー７０２からのＲＯＰに対応する。マルチプレクサ７３０～７３３は、以前のデコードＰＣ値が分岐であったかどうかに基づき、それぞれの加算器７２０～７２３への入力の１つを提供する。マルチプレクサ７４０～７４３は、Ｘ８６命令のバイト長およびＸ８６命令にいくつのＲＯＰが対応するかに基づき、それぞれの加算器７２０～７２３への他の入力を提供する。各加算器７２０～７２３は、次に来るかもしれないデコードＰＣ値の下位４ビットとキャリービットとを与える。次に来るかもしれないデコードＰＣ値は、これらの値の１つを次のデコードＰＣ値として与えるマルチプレクサ７５０へ、およびこれらの値をリオーダバッファ１１４に与えるプログラムカウンタ値選択回路７１２に提供される。

【００９３】

上位プログラムカウンタ発生器回路７１２は、インクリメンタ７６０、マルチプレクサ７６１、インクリメンタ７６２、およびキャリーセレクタ７６４を含む。インクリメンタ７６０は、ＰＣラッチ７０４内にストアされる現在のデコードＰＣ値の上位部分を増分し、この増分された値をマルチプレクサ７６１に与える。マルチプレクサ７６１はまた、キュー７０２から非順次ＰＣ値をも受取る。マルチプレクサ７６１は、分岐が起こったかどうかに基づき、これらの値の１つをインクリメンタ７６２に与える。インクリメンタ７６２はこの値を受取り、この値をＰＣラッチ７０４に与える。インクリメンタ７６０および７６２は、キャリーセレクタ７６４によって与えられるキャリービットに基づき、増分を行なう。キャリーセレクタ７６４はリオーダバッファ１１４にディスパッチされるべき最後の命令におけるキャリービットを提供する。インクリメンタ７６０は次のデコードＰＣ値のビット４：３１を分岐セクション１３５に与える。

【図面の簡単な説明】

【図１】本発明に従うマイクロプロセッサのブロック図である。

【図２】本発明に従う線形にアドレス指定された命令キャッシュのブロック図である。

【図３】本発明に従う命令デコーダのブロック図である。

【図４】本発明に従う命令デコーダプログラムカウンタ回路のブロック図である。

【図５】本発明に従うリオーダバッファのブロック図である。

【図６】本発明に従うリオーダバッファアレイのブロック図である。

【図７】本発明に従うリオーダバッファプログラムカウンタ回路のブロック図である。

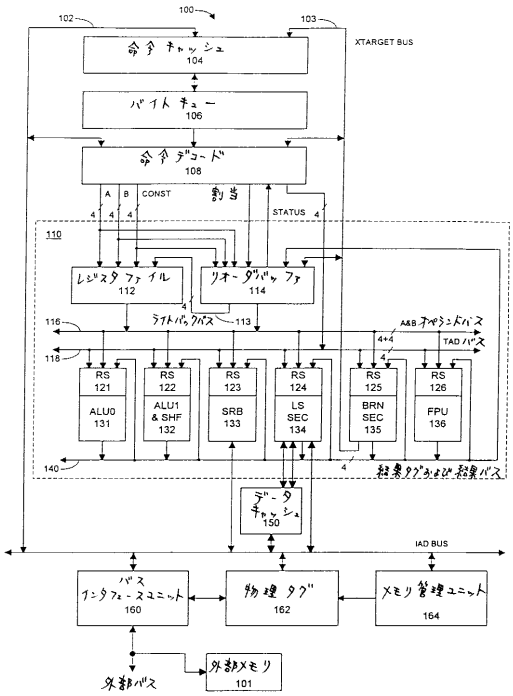
【図８】本発明に従う図１のマイクロプロセッサにおける動作のタイミング図である。

【図９】本発明に従う代替的なフェッチプログラムカウンタのブロック図である。

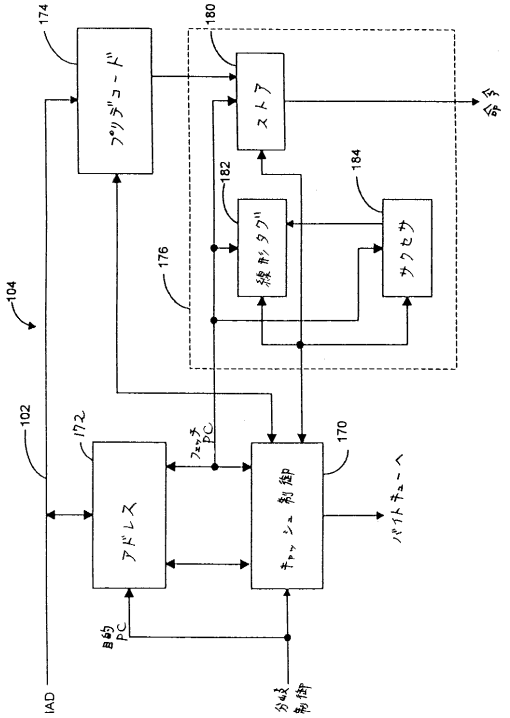
【符号の説明】

４００　ＲＯＰ制御およびステータス回路
５９８　実行プログラムカウンタ発生器回路
６００　下位プログラムカウンタ発生器
６０２　上位プログラムカウンタ発生器
６０４　実行プログラムカウンタ制御装置

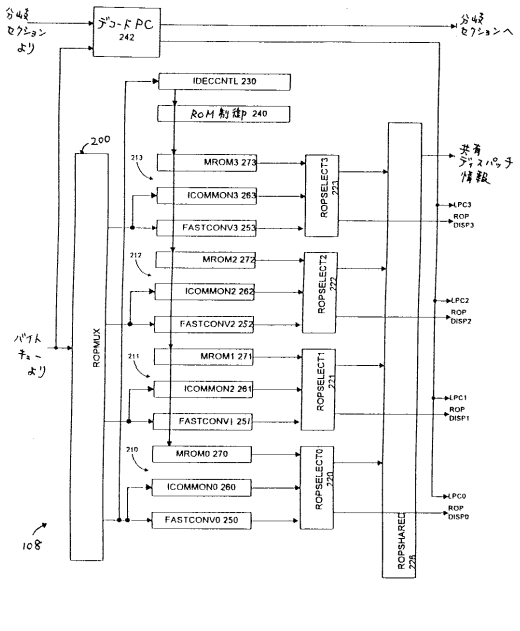
【図１】


100
102
103
XTARGET BUS
命令キャッシュ
104
バイトキュー
106
命令デコード
108
A
B
CONST
割当
STATUS
110
レジスタファイル
112
リオーダバッファ
114
ライトバックバス
113
116
118
A&B オペランドバス
TAD バス
RS 121
ALU0 131
RS 122
ALU1 & SHF 132
RS 123
SRB 133
RS 124
LS SEC 134
RS 125
BRN SEC 135
RS 126
FPU 136
140
結果タグおよび結果バス
データキャッシュ
150
IAD BUS
バスインタフェースユニット
160
物理タグ
162
メモリ管理ユニット
164
外部バス
外部メモリ
101

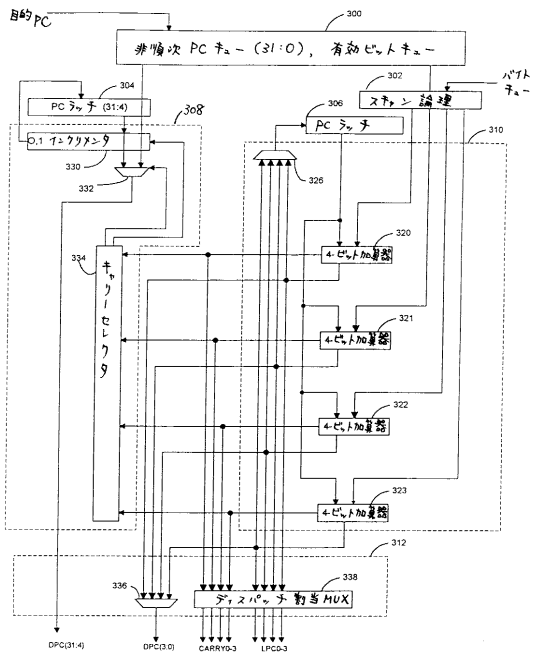
【図２】


174
プリデコード
180
ストア
命令
182
線形タグ
184
サクセサ
176
104
102
172
アドレス
フェッチPC
170
キャッシュ制御
バイトキューへ
目的PC
IAD
分岐制御

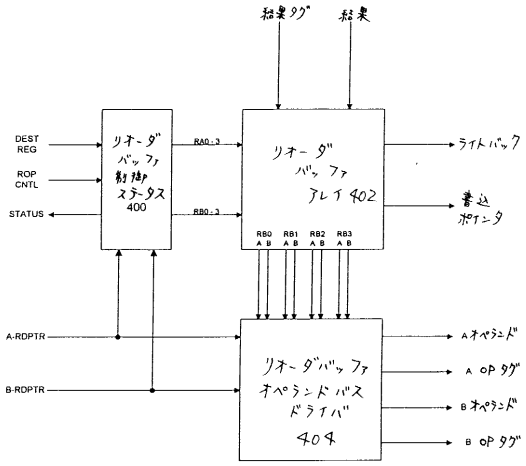
【図３】


分岐セクションより
デコードPC
242
分岐セクションへ
IDECCNTL 230
ROM制御 240
200
MROM3 273
ICOMMON3 263
FASTCONV3 253
ROPSELECT3 223
共有ディスパッチ情報
LPC3
ROP DISP3
MROM2 272
ICOMMON2 262
FASTCONV2 252
ROPSELECT2 222
LPC2
ROP DISP2
MROM1 271
ICOMMON2 261
FASTCONV1 251
ROPSELECT1 221
LPC1
ROP DISP1
MROM0 270
ICOMMON0 260
FASTCONV0 250
ROPSELECT0 220
LPC0
ROP DISP0
ROPSHARED 226
ROPMUX
バイトキューより
108

【図４】


目的PC
300
非順次PCキュー(31:0), 有効ビットキュー
304
PCラッチ(31:4)
308
0,1インクリメンタ
330
332
302
スキャン論理
バイトキュー
306
PCラッチ
310
326
320
4-ビット加算器
321
322
323
334
キャリーセレクタ
312
336
338
ディスパッチ割当MUX
DPC(31:4)
DPC(3:0)
CARRY0-3
LPC0-3

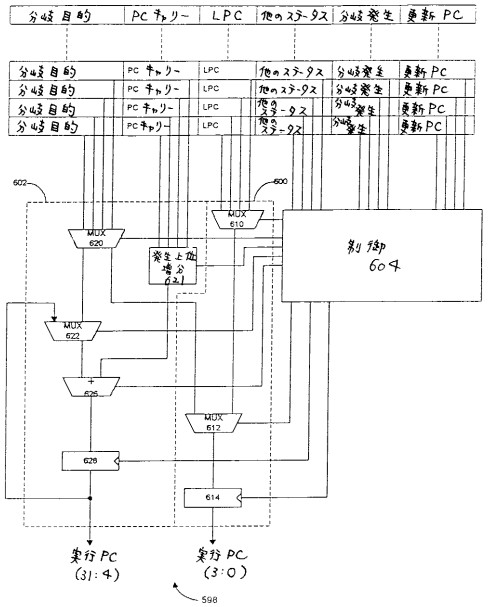
【図５】



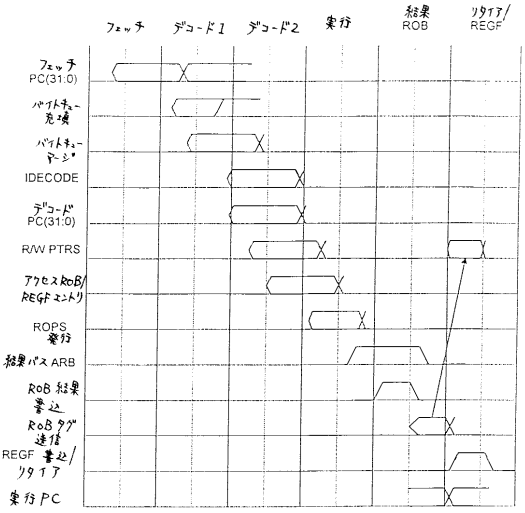
【図６】

CS 0	CS 1	CS 2	CS 3	CS 4	CS 5	CS 6	CS 7	CS 8	CS 9	CS 10	CS 11	CS 12	CS 13	CS 14	CS 15
FPFR 0	FPFR 1	FPFR 2	FPFR 3	FPFR 4	FPFR 5	FPFR 6	FPFR 7	FPFR 8	FPFR 9	FPFR 10	FPFR 11	FPFR 12	FPFR 13	FPFR 14	FPFR 15
FPOP 0	FPOP 1	FPOP 2	FPOP 3	FPOP 4	FPOP 5	FPOP 6	FPOP 7	FPOP 8	FPOP 9	FPOP 10	FPOP 11	FPOP 12	FPOP 13	FPOP 14	FPOP 15
LPC0	LPC1	LPC2	LPC3	LPC4	LPC5	LPC6	LPC7	LPC8	LPC9	LPC10	LPC11	LPC12	LPC13	LPC14	LPC15
P0	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
RESULT 0	RESULT 1	RESULT 2	RESULT 3	RESULT 4	RESULT 5	RESULT 6	RESULT 7	RESULT 8	RESULT 9	RESULT 10	RESULT 11	RESULT 12	RESULT 13	RESULT 14	RESULT 15

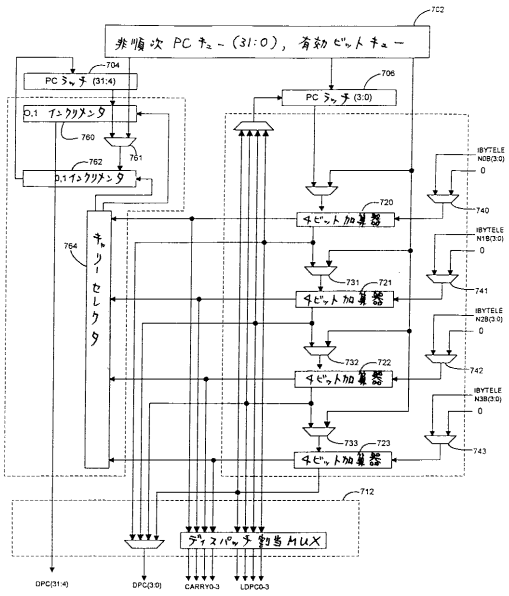
【図７】



【図８】



【図９】


非順次 PC ﾗｯﾁ (31:0), 有効ﾋﾞｯﾄﾗｯﾁ
PC ﾗｯﾁ (31:4)
PC ﾗｯﾁ (3:0)
0,1 ｲﾝｸﾘﾒﾝﾀ
0,1 ｲﾝｸﾘﾒﾝﾀ
ｷｬﾘｰｾﾚｸﾀ
ﾋﾞｯﾄ加算器
ﾋﾞｯﾄ加算器
ﾋﾞｯﾄ加算器
ﾋﾞｯﾄ加算器
ﾃﾞｰﾀﾊﾞｯﾌｧ割当MUX
IBYTELE N0B(3:0)
IBYTELE N1B(3:0)
IBYTELE N2B(3:0)
IBYTELE N3B(3:0)
DPC(31:4)
DPC(3:0)
CARRY0-3
LDPC0-3

フロントページの続き

(74)代理人 100091409
弁理士 伊藤 英彦
(74)代理人 100096781
弁理士 堀井 豊
(72)発明者 デイビッド・エス・クリスティー
アメリカ合衆国、７８７３９ テキサス州、オースティン、ニードハム・レーン、６２０１
(72)発明者 スコット・エイ・ホワイト
アメリカ合衆国、７８７４８ テキサス州、オースティン、ペレンニアル・コート、１１３０３
(72)発明者 マイケル・ディー・ゴッダード
アメリカ合衆国、７８７３９ テキサス州、オースティン、オールド・ハーバー・レーン、６４３４

審査官 後藤 彰

(56)参考文献 特開平４－３４３１３４（ＪＰ，Ａ）
特開平２－２２４０５５（ＪＰ，Ａ）
特表平７－５０５９６８（ＪＰ，Ａ）
国際公開第９３／２０５０７（WO，Ａ２）

(58)調査した分野(Int.Cl.[7]，ＤＢ名)
G06F 9/38