(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第3618821号
(P3618821)
(45)発行日　平成17年2月9日(2005.2.9)　(24)登録日　平成16年11月19日(2004.11.19)

(51)Int.Cl.[7]　F I
G06F 9/38　G06F 9/38　310F
G06F 9/30　G06F 9/30　370

請求項の数 4　(全 20 頁)

(21)出願番号　特願平7-101147
(22)出願日　平成7年4月25日(1995.4.25)
(65)公開番号　特開平7-295813
(43)公開日　平成7年11月10日(1995.11.10)
審査請求日　平成14年3月15日(2002.3.15)
(31)優先権主張番号　233563
(32)優先日　平成6年4月26日(1994.4.26)
(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】複数の型のオペレーションを同時並列で実行するためのプロセッサコア、およびオペレーションに用いられるオペランドデータを処理し通信する方法

(57)【特許請求の範囲】
【請求項１】
オペランドデータを用いかつ整数および浮動小数点型であるオペレーションを実行するためのプロセッサコアであって、
複数のn_1ビットのオペランドバスとn_1ビットの結果バスとを備え、n_1は正の整数であり、前記プロセッサコアはさらに、
オペランドバスに複数のn_2ビットの入力を介して結合されかつ結果バスにn_2ビットの出力を介して結合される整数機能ユニットを備え、n_2はn_1より小さいかまたは等しい正の整数であり、前記プロセッサコアはさらに、
n_1ビットオペランドバスに複数のn_1ビット入力を介して結合されかつ結果バスにn_1ビットの出力を介して結合される浮動小数点機能ユニットを備え、浮動小数点機能ユニットは、
浮動小数点機能ユニットの入力からの複数のn_1ビットのオペランドをn_3ビットのオペランドデータに組合せるための手段と、
n_3ビットのオペランドデータを用いて浮動小数点オペレーションを実行するための手段と、
n_3ビットの結果を発生するための手段と、
n_3ビットの結果を複数のn_1ビットの部分的結果に区分するための手段と、　部分的結果を浮動小数点機能ユニットの出力に送るための手段とを含み、ビットn_3の数はビットn_1の数よりも大きく、前記プロセッサコアはさらに、

n_1ビットのライトバックバスと、
出力がn_1ビットのオペランドバスとn_1ビットのライトバックバスとに結合されかつ入力が結果バスに結合され、予測実行されたオペレーションの結果を記憶するためのリオーダバッファと、
出力がn_1ビットのオペランドバスに結合されかつ入力がn_1ビットのライトバックバスに結合され、予測実行されたオペレーションの結果がもはや予測的でないときそれらの結果を記憶するためのレジスタファイルとをさらに含む、オペランドデータを用いかつ整数および浮動小数点型であるオペレーションを実行するためのプロセッサコア。
【請求項２】
プロセッサコアは実質的に長方形のセグメントの概して平面的な配列で配置され、各セグメントは第１の次元内に延びており、セグメントは第２の次元内に実質的に積み重ねられ、オペランドバスと結果バスとライトバックバスとは複数のセグメントにわたって第２の次元内に延び、
レジスタファイルは第１のセグメントにおいて平面配列の一方の端部に順序付けられ、
リオーダバッファは第１のセグメントに隣接した第２のセグメントに順序付けられ、
複数の整数機能ユニットは第２のセグメントに隣接した第３のセグメントに順序付けられ、
浮動小数点機能ユニットは第３のセグメントに隣接した第４のセグメントに順序付けられる、請求項１に記載のプロセッサコア。
【請求項３】
オペランドバスは第２の次元内で、レジスタファイルを実質的に通過し、かつリオーダバッファと整数機能ユニットとを完全に通過し、かつ浮動小数点機能ユニットを部分的に通過して延び、
結果バスは第２の次元内で、リオーダバッファを実質的に通過し、かつ整数機能ユニットを完全に通過し、かつ浮動小数点機能ユニットを部分的に通過して延び、したがって、機能ユニットを通過して延びるが機能ユニットには結合しないオペランドおよび結果バスデータ線の数、ならびにオペランドおよび結果バスデータ線の長さが減じられ、
ライトバックバスは第２の次元内で、リオーダバッファを実質的に通過し、かつレジスタファイルを実質的に通過して延びる、請求項２に記載のプロセッサコア。
【請求項４】
整数ビットn_2の数は３２であり、浮動小数点ビットn_3の数は８０またはそれより大きく、バスのビットn_1の数は４０またはそれより大きい、請求項１に記載のプロセッサコア。
【発明の詳細な説明】
【０００１】
【発明の背景】
関連出願
この出願は、１９９３年１０月２９日出願の米国特許出願連続番号０８／１４６，３８２の一部係属出願である。
【０００２】
発明の分野
この発明はプロセッサアーキテクチャに関する。より特定的にはこの発明は、浮動小数点機能ユニットと非浮動小数点機能ユニットとを組合せるプロセッサアーキテクチャに関連する。
【０００３】
関連技術の説明
プロセッサは一般的に命令セットのうちの１つの命令をいくつかのステップで処理する。初期の技術によるプロセッサはこれらのステップを直列に行なった。技術の進歩で、スカラプロセッサと呼ばれる、多くの命令の異なるステップを同時並列で行なうパイプラインアーキテクチャプロセッサへと至った。「スーパースカラ」プロセッサもまたパイプライ

ン構造を用いて実現されるが、それはスカラ命令を同時並列で実行することによって性能をさらに向上させる。

【０００４】

スーパースカラプロセッサでは、必要なデータまたは資源が利用可能でないために、発行された命令が実行できないという命令の競合および依存性条件が生じる。たとえば、発行された命令の入力オペランドが実行を完了していない他の命令によって計算されるデータに依存しているときは、その命令は実行できない。

【０００５】

分岐命令の予測実行と、命令を即座に実行できるかどうかにかかわらず命令デコードを継続することとによって、スーパースカラプロセッサの性能は向上される。命令デコードと命令実行との切離しは、プロセッサの命令デコーダと、機能ユニットと呼ばれる、命令を実行する回路との間にバッファを必要とする。

【０００６】

浮動小数点機能はもう何年もの間、非スーパースカラコンピュータおよびプロセッサにおいて利用されてきた。マイクロプロセッサは通常は、別個の浮動小数点回路と整数回路とを活性化することによって、浮動小数点命令と整数命令とを行なう。浮動小数点演算のための標準は、インスティチュート・オブ・エレクトリカル・アンド・エレクトロニック・エンジニアーズ（Ｉｎｓｔｉｔｕｔｅ ｏｆ Ｅｌｅｃｔｒｉｃａｌ ａｎｄ Ｅｌｅｃｔｒｏｎｉｃ Ｅｎｇｉｎｅｅｒｓ）によって、「２進浮動小数点演算のためのＩＥＥＥ規格（ＩＥＥＥ Ｓｔａｎｄａｒｄ Ｆｏｒ Ｂｉｎａｒｙ Ｆｌｏａｔｉｎｇ－Ｐｏｉｎｔ Ａｒｉｔｈｍｅｔｉｃ）」、ＡＮＳＩ／ＩＥＥＥ 規格７５４－１９８５、ＩＥＥＥインコーポレイテッド（ＩＥＥＥ Ｉｎｃ．）、１９８５に発表されている。この標準は広く受入れられており、プロセッサがそのオプションの拡張浮動小数点フォーマットをサポートするのは有利なことである。

【０００７】

いくつかのコンピュータは、別個の主プロセッサチップとコプロセッサチップとを用いる。主プロセッサは浮動小数点レジスタスタックを読出しおよびそれに書込んで浮動小数点オペレーションを行なう。たとえば、スカラマイクロプロセッサである８０３８６主プロセッサおよび８０３８７数値計算コプロセッサがさまざまな製造業者から入手可能である。数値計算コプロセッサは主プロセッサからの要求で開始される浮動小数点オペレーションを制御する。主プロセッサは、２倍に拡張されたフォーマットに記憶される８つまでの浮動小数点値を保持するための８つのレジスタを含むレジスタスタックにアクセスする。３２ビットの単精度値または６４ビットの倍精度値はメモリからロードされ８０ビットの２倍に拡張されたフォーマットに拡張される。逆に、メモリに記憶される際には、２倍に拡張された値は短縮されて単精度値または倍精度値に丸められる。

【０００８】

カリフォルニア州（Ｃａｌｉｆｏｒｎｉａ）、サンタ・クララ（Ｓａｎｔａ Ｃｌａｒａ）のインテル・コーポレイション（Ｉｎｔｅｌ Ｃｏｒｐｏｒａｔｉｏｎ）から入手可能なＰｅｎｔｉｕｍ ＴＭ マイクロプロセッサは、２つの命令パイプラインのオペレーションを制御することによって、混合された浮動小数点および整数命令を実行する、スーパースカラプロセッサである。パイプラインのうちの１つはすべての整数および浮動小数点命令を実行する。第２のパイプラインは単純な整数命令と浮動小数点交換命令とを実行する。

【０００９】

浮動小数点機能ユニットをスーパースカラプロセッサにいくつかの整数機能ユニットとともに組入れることは望ましい。ダブリュー・エム・ジョンソン（Ｗ．Ｍ．Ｊｏｈｎｓｏｎ）は、「スーパースカラプロセッサ設計（Ｓｕｐｅｒｓｃａｌａｒ Ｐｒｏｃｅｓｓｏｒ Ｄｅｓｉｇｎ）」、エンゲルウッド・クリフス（Ｅｎｇｅｌｗｏｏｄ Ｃｌｉｆｆｓ）、Ｎ．Ｊ．、プレンティス・ホール（Ｐｒｅｎｔｉｃｅ Ｈａｌｌ）、１９９１、ｐ．４５で、プロセッサ機能ブロックの２つの組、つまり３２ビットのユニットおよびバス

上に構成される整数の組と、８０ビット構造に編成される浮動小数点の組とを与えている。スーパースカラプロセッサにおいて、浮動小数点の組および整数の組は、別個のレジスタファイルとリオーダバッファとオペランドバスと結果バスとを各々必要とする。浮動小数点命令は浮動小数点の組内の命令デコーダによってディスパッチされる。別の命令デコーダはユニットの整数の組に設けられる。このジョンソンのアプローチはスーパースカラアーキテクチャを組込むプロセッサにおける浮動小数点演算をサポートして、命令デコードおよび命令実行と、分岐予測とを切離す。このアプローチのかなりの性能上の利点は、資源の重複を省みないことによって達成される。さらに、性能におけるいくぶんかの低下が、機能ブロックの整数の組と浮動小数点の組との間におけるオペレーションの調整から生ずる。

【００１０】

【発明の概要】

この発明は、スーパースカラアーキテクチャの利点をよりよく実現して、命令デコードおよび命令実行と分岐予測とを切離す、整数および浮動小数点データの改善された処理のための内部マシン編成を有するプロセッサおよびアーキテクチャである。

【００１１】

この発明のスーパースカラプロセッサおよびアーキテクチャは、整数およびＩＥＥＥ拡張浮動小数点フォーマットの標準的演算をサポートしながら、デコーダ、レジスタファイル、リオーダバッファ、ならびに関連のオペランドバスおよび結果バスのような機能ブロックの重複を有利に回避する。資源重複を避けることにより、生産コストおよびオペレーティングコストが減少し、重複する資源間の相互作用から生じる複雑性が回避され、機能ブロックの異なる組によって行なわれるオペレーション間のタイミング関係の制御が容易となる。

【００１２】

この発明のアーキテクチャは、整数データまたは浮動小数点データのいずれもが共通の包括的な方法で共通の制御下において多重データ経路の組の任意のデータ経路に流れるという整数データおよび浮動小数点データの処理を、パイプラインプロセッサまたはスーパースカラプロセッサに有利に組込む。

【００１３】

整数オペレーションおよび浮動小数点オペレーションの両方のための依存性およびタグ情報を記憶するために、１つのリオーダバッファメモリを用いて、解決されないデータ依存性を処理することは、この発明のスーパースカラプロセッサの特定的な利点である。リオーダバッファの１つの機能は、散在する整数および浮動小数点オペレーションの順序を含む、ディスパッチされるオペレーションの順序を維持することである。単一のリオーダバッファはオペレーション順序を単純な態様で維持する。対照的に、別個の整数リオーダバッファと浮動小数点リオーダバッファとを有するプロセッサはオペレーション順序を維持するのに広範囲にわたる相互参照を必要とする。

【００１４】

さらに、この発明のスーパースカラプロセッサの性能は、単一のリオーダバッファを実現して、分岐予測および回復を用いて、混合された整数および浮動小数点オペレーションの予測実行を処理することによって、有利に強化される。単一のリオーダバッファは、整数オペレーション、浮動小数点オペレーション、および分岐オペレーションの順序を維持する。リオーダバッファにおける他のエントリに関する分岐命令の位置はポインタによって示される。したがって、命令シーケンスは１つの分岐ポインタでもってリオーダバッファエントリの１つのシーケンスに反映される。分岐の予測誤り後の予測状態のフラッシングは、記憶されるデータの型に関係なく、および複雑な制御構造または命令の直列化を要求することなく、分岐の予測誤りの後の回復の間にわずか１つのメモリを変えることによって容易に達成されるであろう。

【００１５】

この発明のプロセッサおよびアーキテクチャは、第１のサイズのオペランドデータを用い

る第１の機能ユニットと第２のサイズのオペランドデータを用いる第２の機能ユニットとを含む、混合された整数および浮動小数点オペレーションを同時並列で実行するための新しいプロセッサコアを提供することによってこれらの利点を達成する。データを与えるために、いくつかのオペランドバスが機能ユニットに接続する。オペランドバスの幅は第１または第２のサイズのデータのいずれをも通信するのに十分である。

【００１６】

この発明の別の実施例は、混合された命令セットを実現するためのプロセッサにおいてオペランドデータを通信するための方法である。この命令セットは、第１のサイズ（たとえば３２ビット）のオペランドデータを用いる第１の型（たとえば整数）の機能ユニットで実行するオペレーションと、第１のサイズより大きい第２のサイズ（たとえば８０またはそれより大きいビット）のオペランドデータを用いて第２の型（たとえば浮動小数点）の機能ユニットで実行するオペレーションとを定義する。この方法は、第２の型のオペレーションを、第２のサイズに比して小さいサイズ（たとえば４０またはそれより多いビット）のサブオペランドデータに各々が関連する複数のサブオペレーションに区分するステップを含む。この方法はさらに、第１のサイズまたは区分された第２のサイズのいずれにも対処する第３のサイズ（たとえば４０またはそれより多いビット）のバス上でサブオペランドデータが通信される状態で、複数のサブオペランドおよび関連するサブオペランドデータを第２の型の機能ユニットにディスパッチするステップを含む。この方法のさらなるステップは、ディスパッチされたサブオペランドデータを第２のサイズのオペランドデータに再び組合せ、その区分されたオペレーションを実行して、第２のサイズの結果を発生するステップを含む。

【００１７】

この発明のさらなる実施例は、整数オペランドデータまたは浮動小数点オペランドデータのいずれをも用いて、いくつかのオペレーションを並列して実行するためのプロセッサである。このプロセッサはいくつかのオペレーションをデコードしディスパッチするためのデコーダを含む。デコーダは、並行してディスパッチされる浮動小数点サブオペランドに各々が関連する複数の関連サブオペレーションに、各々の浮動小数点オペレーションを区分するための回路を含む。プロセッサは、ディスパッチされたサブオペレーションを含む制御信号を受取るためにデコーダに結合される浮動小数点機能ユニットをさらに含む。機能ユニットは、サブオペランドデータを再び組合せてその組合されたデータを用いて１つのオペレーションで駆動少数点サブオペレーションを実行するための回路と、その実行結果を複数のサブ結果に区分するための回路とを含む。

【００１８】

同じ参照番号は同じ要素を示す添付の図面に関連させて以下の説明を参照することにより、この発明はよりよく理解され、その利点、目的および特徴はより明らかにされるであろう。

【００１９】

【好ましい実施例の詳細な説明】

整数および浮動少数点オペレーションを実行するための命令セットを有するスーパースカラプロセッサ１０のアーキテクチャおよび機能図が図１に示される。６４ビットの内部アドレスおよびデータバス１１は、プロセッサ１０のさまざまな機能ブロックと外部メモリ１４との間で、アドレス、データ、および制御転送を通信する。命令キャッシュ１６はＣＩＳＣ命令をパージングしプリデコードする。バイト待ち行列３５はプリデコードされた命令を命令デコーダ１８に転送し、命令デコーダ１８はＣＩＳＣ命令をＲＩＳＣ的オペレーション（「ＲＯＰ」）のための命令のそれぞれのシーケンスにマッピングする。命令デコーダ１８は、バイト待ち行列３５にあるプリデコードされたＣＩＳＣ命令に基づいて、すべてのＲＯＰのための型、オペコード、およびポインタ値を発生する。例示の実施例は記憶された複合命令を受取ってその複合（ＣＩＳＣ）命令を実行のためにＲＩＳＣ型ＲＯＰに変換するためのプロセッサ１０を示すが、専らＣＩＳＣ命令でまたはＲＩＳＣ型オペレーションのいずれかのみで動作するプロセッサ、およびデコード時に命令の形式を変更

するプロセッサも想定されていることは当然である。

【００２０】

好適な命令キャッシュ１６は、１９９３年１０月２９日出願の米国特許出願連続番号０８／１４５，９０５（デイビッド・ビー・ウィット（Ｄａｖｉｄ Ｂ．Ｗｉｔｔ）およびマイケル・ディー・ゴッダード（Ｍｉｃｈａｅｌ Ｄ．Ｇｏｄｄａｒｄ ）、「可変バイト長命令に特に好適な、プリデコード命令キャッシュおよびそのための方法（Ｐｒｅ－Ｄｅｃｏｄｅ Ｉｎｓｔｒｕｃｔｉｏｎ Ｃａｃｈｅ ａｎｄ Ｍｅｔｈｏｄ Ｔｈｅｒｅｆｏｒ Ｐａｒｔｉｃｕｌａｒｌｙ Ｓｕｉｔａｂｌｅ ｆｏｒ Ｖａｒｉａｂｌｅ Ｂｙｔｅ－Ｌｅｎｇｔｈ Ｉｎｓｔｒｕｃｔｉｏｎｓ）」）にさらに詳細に記載される。好適なバイト待ち行列３５は、１９９３年１０月２９日出願の米国特許出願連続番号０８／１４５，９０２（デイビッド・ビー・ウィット「可変バイト長命令に特に好適な、予測命令待ち行列およびそのための方法（Ｓｐｅｃｕｌａｔｉｖｅ Ｉｎｓｔｒｕｃｔｉｏｎ Ｑｕｅｕｅ ａｎｄ Ｍｅｔｈｏｄ Ｔｈｅｒｅｆｏｒ Ｐａｒｔｉｃｕｌａｒｌｙ Ｓｕｉｔａｂｌｅ ｆｏｒ Ｖａｒｉａｂｌｅ Ｂｙｔｅ－Ｌｅｎｇｔｈ Ｉｎｓｔｒｕｃｔｉｏｎｓ ）」）にさらに詳細に記載される。好適な命令デコーダ１８は、１９９３年１０月２９日出願の米国特許出願連続番号０８／１４６，３８３（デイビッド・ビー・ウィットおよびマイケル・ディー・ゴッダード「スーパースカラ命令デコーダ（Ｓｕｐｅｒｓｃａｌａｒ Ｉｎｓｔｒｕｃｔｉｏｎ Ｄｅｃｏｄｅｒ ）」）にさらに詳細に記載される。これらの特許出願の各々の全体をここに引用により援用する。

【００２１】

命令デコーダ１８は、さまざまなバスを介してプロセッサ１０内の機能ブロックにＲＯＰオペレーションをディスパッチする。プロセッサ１０は、４つのＲＯＰ発行と、５つのＲＯＰ結果と、１６までの予測して実行されるＲＯＰの結果とをサポートする。Ａソースオペランドおよびソースオペランドならびに宛先レジスタへのポインタの４つまでの組は、レジスタファイル２４とリオーダバッファ２６とに、命令デコーダ１８によって、それぞれのＡオペランドポインタ３６とＢオペランドポインタ３７と宛先レジスタポインタ４３とを介して与えられる。レジスタファイル２４およびリオーダバッファ２６は次いで、さまざまな機能ユニットへ、４１ビットのＡオペランドバス３０および４１ビットのＢオペランドバス３１の４つの対上で、ＲＩＳＣオペランドＡおよびＢの適当な「予想されて実行される」バージョンを与える。ＡおよびＢオペランドバス３０および３１に関連するのは、Ａオペランドタグバス４８およびＢオペランドタグバス４９の４つの対を含むオペランドタグバスである。結果がオペランドバス上に置かれるために利用可能になると、結果が利用可能になったときにそれを受取るためのリオーダバッファ２６におけるエントリを識別するタグが、対応するオペランドタグバス上にロードされる。オペランドバスおよびオペランドタグバスの４つの対は４つのＲＯＰディスパッチ位置に対応する。命令デコーダ１８は、リオーダバッファ２６と協働して、ＲＯＰの実行後に機能ユニットから結果を受取るリオーダバッファ２６におけるエントリを識別するための４つの宛先タグバス４０を特定する。機能ユニットは、ＲＯＰを実行し、宛先タグを５つの結果タグバス３９のうちの１つにコピーし、結果が利用可能になるとそれを５つの結果バス３２のうちの対応する１つに置く。結果タグバス３９上に対応するタグが、結果を待つＲＯＰのオペランドタグと一致すると、機能ユニットは結果バス３２上の結果に直接アクセスする。

【００２２】

命令デコーダ１８は、４つのオペコード／型バス５０を介して、オペコードおよびオペコードの型を含む、ＡおよびＢソースオペランド情報に伴うオペコード情報をディスパッチする。

【００２３】

プロセッサ１０は、分岐ユニット２０、整数機能ユニット２１、浮動小数点機能ユニット２２、およびロード／ストア機能ユニット８０のような、いくつかの機能ユニットを含む。整数機能ユニット２１は、一般的なものとして示されたものであり、算術論理ユニット、シフトユニット、および特別なレジスタユニットのような、さまざまなタイプのユニッ

トを表わす。分岐ユニット２０は、分岐があるときに適当な命令フェッチ速度を可能にし、複数の命令発行を伴う動作を達成するのに必要な技術である、分岐予測オペレーションの妥当性を検査する。分岐ユニット２０および命令デコーダ１８を含む好適な分岐予測システムは、米国特許第５，１３６，６９７号（ウィリアム・エム・ジョンソン（Ｗｉｌｌｉａｍ Ｍ．Ｊｏｈｎｓｏｎ ）「キャッシュに命令の各ブロックとともに記憶されるフェッチ情報を用いて正しく予測される分岐命令後の実行のための遅延を低減するためのシステム（Ｓｙｓｔｅｍ ｆｏｒ Ｒｅｄｕｃｉｎｇ Ｄｅｌａｙ ｆｏｒ Ｅｘｅｃｕｔｉｏｎ Ｓｕｂｓｅｑｕｅｎｔ ｔｏ Ｃｏｒｒｅｃｔｌｙ Ｐｒｅｄｉｃｔｅｄ Ｂｒａｎｃｈ Ｉｎｓｔｒｕｃｔｉｏｎ Ｕｓｉｎｇ Ｆｅｔｃｈ Ｉｎｆｏｒｍａｔｉｏｎ Ｓｔｏｒｅｄ ｗｉｔｈ ｅａｃｈ Ｂｌｏｃｋ ｏｆ Ｉｎｓｔｒｕｃｔｉｏｎｓ ｉｎ Ｃａｃｈｅ）」）にさらに詳細に記載され、その全体をここに引用により援用する。

【００２４】

プロセッサ１０は過度の複雑さを避けるために、機能ユニットの単純な組を有して示される。機能ユニットの数および型は、単一の浮動小数点機能ユニット２２と、一般的には整数データでオペレーションを行なう複数の機能ユニット２０と２１と８０とを有する、特定の態様でここに示されるが、整数および浮動少数線ユニットの他の組合せが所望されるように実現されてもよいことが理解される。各機能ユニット２０、２１、２２および８０は、オペランドバス３０および３１ならびにオペコード／型バス５０に入力が接続される、それぞれのリザベーションステーション（図示せず）を有する。リザベーションステーションは機能ユニットへの予測ＲＯＰのディスパッチを可能にする。

【００２５】

レジスタファイル２４は、整数および浮動小数点命令のためのマッピングされたＣＩＳＣのレジスタを含む物理的な記憶メモリである。それは、中間の計算を保持するための一次的整数および浮動小数点レジスタをさらに含む。レジスタファイル２４は浮動小数点データおよび整数データの両方を処理する。整数はレジスタファイル２４の下位３２ビット〈３１：０〉に位置される。上位ビット〈４０：３２〉はレジスタファイル２４の整数レジスタにおいては実現されない。レジスタファイル２４は整数データまたは浮動小数点データに対して同様に機能する。レジスタファイル２４は、結果回収（リタイア）として公知のプロセスで、リオーダバッファ２６から４つのライトバックバス３４を介して、実行された、予測によらないオペレーションの結果を受取る。

【００２６】

リオーダバッファ２６は予測実行されたＲＯＰの相対的な順序を把握するための循環ＦＩＦＯである。リオーダバッファ記憶位置は、レジスタファイル２４へ回収中の結果を送るために、および機能ユニットから結果を受取るために、動的に割り当てられる。命令がデコードされると、その結果値にリオーダバッファ２６にある位置すなわち宛先が割り当てられ、その宛先レジスタ番号はこの位置に関連付けされる。次のオペレーションが依存性を全く有さない場合には、その関連するＡおよびＢオペランドバス３０および３１はレジスタファイル２４から駆動される。しかしながら、次のオペレーションが依存性を有し、そのリネーミングされた宛先レジスタに記憶されると考えられる値を得るためにそれを参照する場合には、リオーダバッファ内でエントリがアクセスされる。結果がそこで利用可能である場合には、それはオペランドバス上に置かれる。結果が利用可能でない場合には、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４８および４９のうちの１つのオペランドタグバス上に与えられる。この結果またはタグは、機能ユニットに、オペランドバス３０、３１またはオペランドタグバス４８、４９を介してそれぞれ与えられる。機能ユニット２０、２１、２２および８０における実行の完了から結果が得られると、その結果およびそれらのそれぞれの結果タグは、リオーダバッファ２６および機能ユニットのリザベーションステーションに、５バス幅の結果バス３２および結果タグバス３９を介して与えられる。

【００２７】

機能ユニットによって発生された結果は、５つの４１ビットの結果バス３２と５つの関連する結果タグおよび状態バス３９とを用いてリオーダバッファ２６に通信される。５つの結果バスおよび結果タグバスならびに状態バスのうち、４つはリオーダバッファに整数および浮動小数点結果を転送するための汎用バスである。付加的な第５の結果バスおよび結果タグバスならびに状態バスは、機能ユニットのいくつかからリオーダバッファへ、転送される結果ではない情報を転送するのに用いられる。たとえば、ロード／ストア機能ユニット８０による記憶オペレーションからまたは分岐ユニット２０による分岐オペレーションから生ずる状態情報はこの付加的なバス上に置かれる。この付加的なバスは、４つの汎用結果バスのバンド幅を保存するために設けられる。

【００２８】

命令デコーダ１８は機能ユニットにＲＯＰを「正順序で」ディスパッチする。順序はリオーダバッファ２６内のエントリの順序で維持される。待ち行列に合った前のＲＯＰがすべて実行を完了し、すべてのソースオペランドがオペランドバスまたは結果バスのいずれかを介して利用可能となり、結果バスが結果を受取るのに利用可能となると、機能ユニットはＲＯＰを発行のために待ち行列に入れる。したがって、機能ユニットは、ＲＯＰを「順序外」で完了する。この態様で、オペレーションのディスパッチは、オペレーションの完了に依存しないので、リザベーションステーションの待ち行列が利用できないことまたはリオーダバッファのエントリが割り当てられないことによってプロセッサがストーリングされない限り、命令デコーダ１８は命令が即座に完了され得るか否かにかかわらず命令をデコードし続ける。

【００２９】

ＲＩＳＣコアのための好適なユニットは、１９９３年１０月２９日出願の米国特許出願連続番号０８／１４６，３８２（デイビッド・ビー・ウィットおよびウィリアム・エム・ジョンソン、「高性能スーパースカラマイクロプロセッサ（Ｈｉｇｈ　Ｐｅｒｆｏｒｍａｎｃｅ　Ｓｕｐｅｒｓｃａｌａｒ　Ｍｉｃｒｏｐｒｏｃｅｓｓｏｒ　）」）に開示され、その全体を引用によりここに援用する。

命令デコーダ１８は、１つのＡオペランドと１つのＢオペランドと１つの宛先レジスタとを特定する１つのディスパッチ位置を用いて、整数ＲＯＰをディスパッチする。例示の命令デコーダ１８は、２つの関連するＡオペランドおよび２つの関連するＢオペランドが「半オペランド」または「サブオペランド」として割り当てられるよう、浮動小数点ＲＯＰを２つの関連する「半ＲＯＰ」またはより一般的には「サブＲＯＰ」としてディスパッチする。浮動小数点の半オペランドは図２によって模式的に示される。図２に示される完全浮動小数点レジスタ２５０は、第１および第２の浮動小数点半オペランド２５６および２５８によって示される２つのオペランドの形式で割り当てられる。Ａオペランドの場合、半オペランドの対のうちの１つ（たとえば２５６）は、第１の関連する浮動小数点オペコードに対応するＡオペランド位置に置かれる。半オペランドの対のうちの第２の半オペランド（たとえば２５８）は、第２の関連する浮動小数点オペコードに対応するＡオペランド位置に置かれる。同様に、Ｂ半オペランドの対は、オペコードの対に対応するＢオペランド位置に置かれる。命令デコーダ１８は、レジスタファイル２４およびリオーダバッファ２６へのポインタ３６および３７を介して、ＡおよびＢオペランドへのアクセスを開始する。命令デコーダ１８は、４つのオペコード／型バス５０のうちの２つで「半オペコード」を通信することによって、浮動小数点機能ユニット２２に半ＲＯＰをディスパッチする。浮動小数点命令の場合、デコーダ１８は、４つのディスパッチ位置のうちのそれぞれ２つからのＡおよびＢ半オペランドの２つの組を有する２つのＲＯＰを、１つのディスパッチウィンドウ内で同時並列でディスパッチする。

【００３０】

例示の実施例において、２つの浮動小数点半ＲＯＰのためのオペコードは「ダミー」オペコードおよび関連する「本当の」オペコードとして指定される。たとえば、ダミーの浮動小数点オペコード（たとえばＦＰＦＩＬＬ）は、ディスパッチウィンドウの１つのディスパッチ位置に、完全な８２ビットの浮動小数点ＡおよびＢオペランドビットのビット〈４

０：０〉に各々が関連するＡおよびＢ半オペランドを割り当てられる。ウィンドウのディスパッチ位置の直後に、本当の浮動小数点オペコードが、完全な浮動小数点ＡおよびＢオペランドのビット〈８１：４１〉と各々が関連するＡおよびＢ半オペランドを割り当てられる。浮動小数点機能ユニット２２はＲＯＰを実行すると、それは半オペコードを併せて１つのＲＯＰを実行する。たとえば、浮動小数点加算命令ＦＡＤＤは次の態様でディスパッチされてもよい。

【００３１】

【表１】

ディスパッチ位置	オペコード／型	Ａオペランド	Ｂオペランド
1	ＦＰＦＩＬＬ	〈４０：０〉	〈４０：０〉
2	ＦＡＤＤ	〈８１：４１〉	〈８１：４１〉
3	－	－	－
4	－	－	－

【００３２】

他のＲＯＰはウィンドウ位置３および４でディスパッチされてもよい。半ＲＯＰの対は１つのウィンドウでディスパッチされる。

【００３３】

他の実施例では、プロセッサ１０はオペランドを２つより多いサブオペランドに分けてもよい。たとえば、２つの半オペランドにオペランドフィールドを区分するのではなく、３つまたはそれより多いサブＲＯＰのディスパッチを介して動作される同じ数のサブオペランドにオペランドを分けてもよい。

【００３４】

リオーダバッファ２６と、レジスタファイル２４と、非浮動小数点機能ユニットと、オペランドバスとタグバスと結果バスとライトバックバスとを含むさまざまなバスとは、浮動小数点半ＲＯＰを２つの独立した無関係のＲＯＰとして処理する。しかしながら、半ＲＯＰを区分する命令デコーダ１８と、半ＲＯＰを実行前に再び組合せる浮動小数点機能ユニット２２とは、半ＲＯＰを関連するエンティティとして扱う。この態様で、浮動小数点オペランドは浮動小数点ユニットおよび他の機能ブロックの間を整数データ経路上で通信される。

【００３５】

レジスタファイル２４の詳細な例は図３に示される。レジスタファイル２４は、読出デコーダ６０、レジスタファイルアレイ６２、書込デコーダ６４、レジスタファイル制御６６、およびレジスタファイルオペランドバスドライバ６８を含む。レジスタファイルアレイ６２は、プロセッサの機能ユニットで動作されかつそれにより発生される結果を記憶するための複数のアドレス指定可能な記憶レジスタを含む。図４は、この実施例では一次整数レジスタＥＴＭＰ０からＥＴＭＰ１５と同じ物理上のレジスタ位置にマッピングされる、８つの３２ビットの整数レジスタ（ＥＡＸ、ＥＰＸ、ＥＣＸ、ＥＤＸ、ＥＳＰ、ＥＢＰ、ＥＳＩおよびＥＤＩ）と８つの８２ビットの浮動少数点レジスタＦＰ０からＦＰ７と１６の４１ビットの一次整数レジスタＥＴＭＰ０からＥＴＭＰ１５と８つの８２ビットの一次浮動小数点レジスタＦＴＭＰ０からＦＴＭＰ７とを含む、４０のレジスタを有する例示のレジスタファイルアレイ６２を示す。

【００３６】

図５を参照すると、リオーダバッファ２６は、リオーダバッファ（ＲＯＢ）制御および状態ブロック７０とリオーダバッファ（ＲＯＢ）アレイ７４とリオーダバッファ（ＲＯＢ）オペランドバスドライバ７６とを含む。ＲＯＢ制御および状態ブロック７０は、ＲＯＰのためのソースおよび宛先オペランドを識別する入力を受取るために、ＡおよびＢオペラン

ドポインタ３６および３７ならびに宛先ポインタ（ＤＥＳＴ　ＲＥＧ）バス４３に結合される。ＲＯＢアレイ７４は機能ユニットからの結果を受取るために結果バス３２に結合される。ヘッド信号とテール信号とＡオペランド選択信号とＢオペランド選択信号と結果選択信号とを含む制御信号は、ＲＯＢ制御および状態７０からＲＯＢアレイ７４に伝えられる。これらの制御信号は結果バス３２データでもって書込まれ、ライトバックバス３４と書込ポインタ３３とＡおよびＢオペランドバス３１および３２とＡおよびＢオペランドタグバス４８および４９とに出力される、ＲＯＢアレイ要素を選択する。Ａオペランド選択信号およびＢオペランド選択信号は、オペランドデータがオペランドバス３０および３１上に置かれるよう指定するべく、リオーダバッファアレイ７４に与えられる。データがレジスタファイル２４またはリオーダバッファ２６のいずれにおいても利用可能でない場合には、Ａオペランド選択信号およびＢオペランド選択信号は、ＡおよびＢオペランドタグバス４８および４９を駆動するよう、リオーダバッファオペランドバスドライバ７６に直接さらに与えられる。１つが各リオーダバッファアレイ要素のためのものである１６の宛先ポインタは、依存性チェックを実現するために、ＲＯＢアレイ７４からＲＯＢ制御および状態７０へ伝えられる。

【００３７】

ＲＯＢアレイ７４はＲＯＢ制御および状態ブロック７０の制御下にあるメモリアレイである。命令デコーダ１８はＲＯＰをディスパッチすると、それは４つの宛先ポインタ（ＤＥＳＴ　ＲＥＧ）バス４３上にポインタを置く。ＲＯＢ制御状態７０は次いでＲＯＢアレイ７４のエントリを割り当てて、割り当てられたエントリのＤＥＳＴ　ＰＴＲフィールドに宛先ポインタを書込む。

【００３８】

オペレーションが実行されて結果が結果バス３２上に機能ユニットによって置かれると、ＲＯＢ制御および状態７０は、結果バス３２からデータを受取るように対応するＲＯＢアレイエントリを指定する、結果タグバス３９からのポインタにアクセスする。ＲＯＢ制御７０は、４つの結果選択ポインタを用いて、結果バス３２からＲＯＢアレイ７４への書込みを命ずる。

【００３９】

図６は、結果フィールドと宛先ポインタフィールドと制御情報を記憶する他のフィールドとを各々が含む１６のエントリを含むリオーダバッファアレイ７４の例を示す。４１ビットの結果フィールドは機能ユニットから受取られた結果を記憶するために与えられる。２つのリオーダバッファエントリは浮動小数点結果を記憶するために用いられる。整数結果は４１ビットのうちの３２ビットで記憶され、残りの９ビットは状態フラグを保持するために用いられる。

【００４０】

各ＲＯＢアレイ７４エントリの宛先ポインタフィールド（ＤＥＳＴ＿ＰＴＲ〈８：０〉）は、レジスタファイル２４にある宛先レジスタを指定する。結果フィールドからのデータは、ＲＯＢアレイ７０のエントリからレジスタファイル２４へライトバックバス３４の１つを介して通信され、ライトバック中に、書込ポインタ３３の１つに宛先ポインタフィールドを置くことによって、指定される宛先レジスタに駆動される。ＲＯＢ制御および状態７０は命令デコーダ１８からオペランドポインタおよび宛先ポインタを、それぞれＡおよびＢオペランドポインタ３６および３７ならびに宛先レジスタ（ＤＥＳＴ　ＲＥＧ）バス４３を介して受取り、ＲＯＢアレイ７４の宛先ポインタ（ＤＥＳＴ　ＰＴＲ〈８：０〉）フィールドに宛先ポインタを書込む。ＲＯＰがディスパッチされると、リオーダバッファ２６は、リオーダバッファアレイ７４の１６の要素すべての宛先ポインタ（ＤＥＳＴ　ＰＴＲ〈８：０〉）フィールドをＡおよびＢオペランドポインタ３６および３７に対して同時にテストして、データ依存性を識別する一致が宛先ポインタとＡまたはＢオペランドポインタとの間に存在するかどうかを判断することによって、依存性チェックを達成する。

【００４１】

ＲＯＢ制御および状態７０はディスパッチでデータ依存性を検知すると、レジスタファイ

ルオペランドバスドライバ６８に与えられる、Ａオペランド無効化バス５７およびＢオペランド無効化バス５８のビットをセットすることによって、レジスタファイルアレイ６２にある任意の依存オペランドの読出しに無効化する。無効化バス５７および５８は各オペランドバスのための無効化信号を含む。ソースオペランドがレジスタファイル２４またはリオーダバッファ２６のいずれもでも利用可能であることを、ＲＯＢ制御および状態７０が判断した場合には、ソースオペランドは、機能ユニットによるアクセスのために、オペランドバス３０または３１のうちの１つのバスの上に置かれる。

【００４２】

ライトバックバス３４のうちの１つの上にＲＯＢアレイ７４要素の結果フィールドを置き、ライトバックバスに対応する書込ポインタ３３を宛先ポインタで駆動することによって、ＲＯＢ制御および状態７０はＲＯＰを回収する。回収された結果を受取るよう、書込ポインタ３３はレジスタファイル２４内のレジスタアドレスを指定する。整数データのライトバックでは、結果の下位３２ビット〈３１：０〉は整数データを保持し、上位ビット〈３７：３２〉は状態フラグ２５を更新するのに用いられるエラーフラグＥＦＬＧＳ７１である。浮動小数点データでは、別々の状態バス３８がリオーダバッファ２６にフラグを通信し、そこで、フラグは、浮動小数点ＲＯＰが回収されるときに浮動小数点状態レジスタ（図示せず）に伝えられるまで記憶される。

【００４３】

図７（ａ）に示される、各リオーダバッファアレイ７４の要素２２０は、４１ビットの結果フィールド１０１と、９ビットの宛先ポインタフィールド１０２と、４ビットのデルタＰＣフィールド１０３と、１１ビットの浮動小数点オペコードフィールド１０４と、１１ビットの浮動小数点フラグレジスタフィールド１０５と、２４ビットの状態／制御フィールド１０６とを含む。浮動小数点オペランドでは、図７（ｂ）に示される結果フィールド２２２は浮動小数点オペレーションの４１ビットの「半分の結果」を保持する。整数オペレーションでは、４１ビットの結果フィールドのビット〈４０：３２〉は、図７（ｃ）、図７（ｄ）、および図７（ｅ）の結果フィールド２２４、２２６および２２８にそれぞれ示されるように、整数フラグレジスタを保持する。８または１６ビットの幅を有する結果をもたらす整数オペレーションでは、図７（ｄ）および図７（ｅ）の結果フィールド２２８および２２６にそれぞれ示される形式で結果を生ずる機能ユニットによって、付加的なビットはクリアされる。

【００４４】

分岐オペレーションでは、図７（ｆ）に示される結果フィールド２３０が、分岐ユニット２０による分岐ＲＯＰの実行によって判断されるように、プログラムカウンタの論理アドレスを保持する。

【００４５】

図７（ａ）を参照すると、宛先ポインタフィールド１０２はレジスタファイル２４の宛先レジスタを特定する。浮動小数点オペコードフィールド１０４は、リオーダバッファエントリに割り当てられる浮動小数点オペ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１０５は、浮動小数点オペレーションから生ずる浮動小数点フラグの状態を保持する。浮動小数点フラグは、浮動小数点機能ユニット２２によって検知される、精度、アンダーフロー、オーバーフロー、ゼロ除算、非正規化オペランド、および無効オペランドエラーに関する情報を保持する。整数オペランドでは、整数オペレーションから生ずるフラグは４１ビットの結果フィールド１０１のビット〈４０：３２〉に保持されるため、対応するフラグフィールドは不要である。状態／制御フィールド１０６は、たとえば、ＲＯＢエントリは割り当てられているか、分岐が間違って予測されているか、またはオペレーションの実行が例外またはエラー条件を引起こしたか、といったオペランドの状態を示す。

【００４６】

図８は、レジスタファイル２４とリオーダバッファ２６と浮動小数点機能ユニット２２とを通る４１ビットのデータ経路を有する、混合された浮動小数点／整数コアの概略的なレ

イアウト図を示す。このデータ経路は、ＡおよびＢオペランドバス３０および３１と、結果バス３２と、ライトバックバス３４とを含む。ＡおよびＢオペランドバス３０および３１のビット〈３１：０〉のみが、ユニット２１のような整数機能ユニットと相互接続される。データ経路が、他の機能ブロックのバスおよびメモリのデータラインのための、ビットごとの対応を有することを示すために、バスのラインはレジスタファイル２４とリオーダバッファ２６と整数ユニット１１０との上に重ねられるようにして図示される。バスのビット構造は浮動小数点リザベーションステーションおよび結果ドライバにのみ延びることを示すために、バスのラインは浮動小数点機能ユニット２２に部分的にのみ延びるように図示される。浮動小数点機能ユニット２２の動作内部では、４１ビットのオペランドの対は、処理のための８２ビットの拡張形式に組合せられる。

【００４７】

４１ビットのＡおよびＢオペランドバス３０および３１の４つのそれぞれの対は、機能ユニットとリオーダバッファ２６とレジスタファイル２４との間を相互接続し、浮動小数点機能ユニット２２のリザベーションステーションから整数ユニットとリオーダバッファ２６とを通り実質的にレジスタファイル２４を通って延びる。整数ユニットを含みかつオペランドバスのビット〈３１：０〉の下部にあるコア領域は、オペランドデータを保持するためのリザベーションステーションレジスタを含む。ＡおよびＢオペランドバス３０および３１のビット〈４１：３２〉は整数ユニットと相互接続することなくそのユニットを通過する。

【００４８】

５つの４１ビットの結果バス３２は、機能ユニットとリオーダバッファ２６との間を相互接続し、かつ浮動小数点機能ユニット２２の結果バスドライバから整数ユニットを通り実質的にリオーダバッファ２６を通って延びる。結果バス３２の４１ビットの実質的にすべてが整数ユニットと相互接続するので、結果データはビット〈３１：０〉で通信され、整数状態フラグはビット〈４１：３２〉で通信される。マイクロプロセッサのこの例示的な実施例においては、定義される状態ビットの数は結果バス３２のビット〈３７：３２〉のみを利用する。

【００４９】

４つの４１ビットのライトバックバス３４は、リオーダバッファ２６からレジスタファイル２４へ結果データを通信し、実質的にリオーダバッファ２６の一方の側から実質的にレジスタファイル２４の反対側へと延びる。

【００５０】

わずかな例外はあるが（たとえばオペランドビット〈４１：３２〉および結果バスビット〈４１：３８〉）、データ経路の各バスの各ビットが、相互接続する機能ブロックのみを通過するということは有利なことである。

【００５１】

このレイアウトは、データ経路のすべてのバスが整数および浮動小数点機能ユニットの全体を通して実質的に一定である、混合された浮動小数点および整数コアを表わす。

【００５２】

図８に示されるレイアウトの実施例は、高密度のコアをもたらすため有利である。浮動小数点機能ユニット２２とレジスタファイル２４とは好ましくはコアの端部に位置される。浮動小数点機能ユニット２２はそれ自体の内部バス構造を有するため、Ａオペランドバス３０とＢオペランドバス３１と結果バス３２とはそれを通過する必要はない。さらに、浮動小数点機能ユニット２２は大きく、コアの約２０％を占める。したがって、Ａオペランドバス３０とＢオペランドバス３１と結果バス３２とが浮動小数点機能ユニット２２を通過して経路付けられるような、浮動小数点機能ユニット２２の位置決めは、ダイスペースを不必要に占めてしまうであろう。同様に、レジスタファイル２４は結果バス３２を使用しないため、結果バス２４はそれを通過する必要はない。さらに、レジスタファイル２４は大きく、コアの約３０％を占める。したがって、結果バス３２がレジスタファイル２４を通って経路付けられるような、レジスタファイル２４の位置決めは、ダイスペースを不

必要に占有するであろう。コアの約２５％を占めるロード／ストア機能ユニット８０は、ＡおよびＢオペランドバス３０および３１と結果バス３２とによって通過されるため、浮動小数点機能ユニット２２とリオーダバッファ２６との間に位置される。分岐ユニット２０を含む整数ユニット１１０は、結果バス３２と、ＡおよびＢオペランドバス３０および３１のビット〈３１：０〉とによって通過されるため、浮動小数点機能ユニット２２とリオーダバッファ２６との間に位置される。オペランドバス３０および３１のビット〈４０：３２〉は整数ユニット１１０をバイパスするが、整数ユニット１１０は比較的小さく、コアの約１０％しか占有しないため、コア密度への悪影響は小さい。

【００５３】

図８は、整数ユニット１１０をバイパスしてリオーダバッファ２６に到達する浮動小数点フラグバス３８をさらに示す。バス３８は１１ビットの１つのバスにすぎないため、コア密度におけるバス３８の悪影響はわずかなものである。整数フラグは浮動小数点データ転送のためにも用いられるさまざまな４１ビットのバス上で通信されるため、このようにしない場合であれば８つのオペランドバス３０および３１ならびに５つの結果バスと関連するであろう１３の別個の６ビットの整数フラグバスを設けることを回避できる効果があることに注目されたい。

【００５４】

図８は、一定の４１ビット幅のデータ経路が浮動小数点機能ユニット２２と整数ユニット１１０とロード／ストア機能ユニット８０とリオーダバッファ２６とレジスタファイル２４とを通過するコアを示すが、他のコアのレイアウトは全体にわたって一定のデータ経路幅を有さず、それでいて、浮動小数点データと整数データおよび整数フラグの組合せとの両方に対処するのに十分な幅のバス構造を有することにより利益を得てもよい。たとえば、整数ユニットがレジスタファイルよりも大きい代替実施例（図示せず）においては、レジスタファイルはコア内に位置され、整数ユニットはコアの一方の端部に位置されるかもしれない。この場合、結果バス３２の４１ビットのピッチは拡張されるであろうが、ＡおよびＢオペランドバスのビット〈３１：０〉のみが拡張される必要があるだけである。この代替レイアウトにおいては、ＡおよびＢオペランドバス３０および３１ならびに結果バス３２はレジスタファイルを通過するかまたはバイパスする必要があるだろう。しかしながら、再配置された整数ユニットへの、結果バス３２のビット〈４０：０〉の拡張が、浮動小数点データと整数データおよび整数フラグの組合せとの両方に有利に対処するであろう。

【００５５】

図９は、データ経路幅が浮動小数点機能ユニット２２の内部データ経路幅よりも小さい、プロセッサ実施例の浮動小数点接続を示すブロック図である。好ましくは、浮動小数点機能ユニット２２の内部データ経路幅はデータバス幅の整数倍である。

【００５６】

例示のアーキテクチャでは、オペランドバス３０および３１、結果バス３２、ならびにライトバックバス３４は、８２ビットの拡張された精度の浮動小数点変数の半分に対処するために、３２ビットデータ経路から４１ビットデータ経路に拡張される。データ経路が拡張されると、整数ＲＯＰは、同じ態様で、純粋に整数用の方式としてディスパッチされる。しかしながら、ＲＯＰに関連するオペランドは、ＡおよびＢオペランドバス３０および３１の４つの対のビット〈３１：０〉に割り当てられる。整数オペランドがバス上にロードされると、上位９ビット〈４０：３２〉は整数データを通信するのには用いられず、下位３２ビット〈３１：０〉が整数データをエンコードしてそれを整数機能ユニットに通信する。浮動小数点ＲＯＰがディスパッチされると、４１ビット全部が浮動小数点リザベーションステーション４４にデータを通信し、それは２つの半ＲＯＰを組合せ、４１ビットのオペランドを局所的に併せて、８２ビットのオペランドを形成する。８２ビットのオペランドは、内部で浮動小数点演算回路１００へ２つの８２ビットの浮動小数点オペランドバス９０および９１を介して送られ、演算回路１００から浮動小数点結果ドライバ４５へ８２ビットの浮動小数点結果バス９２上で通信される。標準的な拡張精度数は８０ビット

幅であるが、指数部にある予備の２ビットを与えて内部浮動小数点数のより大きなデータ範囲に対処してもよい。

【００５７】

浮動小数点フラグはＦＰフラグバス１１２上で通信する。プロセッサ１０内には浮動小数点機能ユニット２２は１つしかないので、１つのＦＰフラグバス１１２で十分であり、浮動小数点機能ユニット２２および浮動小数点フラグバス１１２の経路付けおよび配置は改善される。整数機能ユニット１１０もフラグをさらに発生するが、他の場合には利用されないであろう結果バス３２のビット〈４０：３２〉が整数フラグを通信するのに用いられるため、対応する整数フラグバスは不要である。４１ビットのバスを介して整数フラグを通信することは、プロセッサのレイアウトを複雑にしかつダイスペースを使用する付加的な専用整数フラグバスを含むことを回避するのに有利である。

【００５８】

ソースオペランドが利用可能になり、２つの結果バスが利用可能になると、リザベーションステーション４４は、浮動小数点乗算、除算、またはＲＯＰで特定される他のオペレーションのようなＲＯＰを発行し、演算回路１００は８２ビットの結果を計算する。浮動小数点結果ドライバ４５は８２ビットの結果を４１ビットのセグメントに分け、そのセグメントを４つの汎用結果バス３２のうちの２つに置き、状態バス３８上にフラグをセットする。浮動小数点機能ユニットは、浮動小数点結果の上位ビット〈４０：３２〉を、上位のビット位置に番号付けられてはいるが、浮動小数点数の最下位の９ビットに対応するよう割り当てる。これが行なわれるのは、この態様でのデータフィールドのシフトによって、浮動小数点データの上位４１ビットの符号および指数が３２ビットの整数データアクセスにおいて読まれることが可能になるからである。

【００５９】

整数ユニット１１０として組合せて示されるさまざまな整数機能ユニットは３２ビットのデータで動作する。整数ユニット１１０は、結果バス３２のビット〈３１：０〉に結果を書込むことに加えて、結果バス３２のビット〈４０：３２〉に結果フラグを書込む。結果バス３２はリオーダバッファ２６に接続され、それは結果データを一時的に記憶する。リオーダバッファ２６は、データが浮動小数点であるかまたは整数であるかにかかわらず、レジスタファイル２４への４１ビットの結果を回収する。

【００６０】

図１０は、標準的な８２ビットの浮動小数点数のビットフィールドを示す。最上位ビット〈８１：４２〉は、符号ビット２４１と、１９ビットの指数部２４２と６２ビットの仮数部の上位２１ビットとを表わす。浮動小数点数の仮数部は、指数部で指定される２のべき乗によって乗算される整数を表わす。好ましい実施例のプロセッサ１０は、８２ビットのフィールド２５０が２つの４１ビットのフィールドに分けられる図１１に示される態様で浮動小数点数を構成する。上位４１ビットフィールドでは、仮数ビット〈５９：４１〉のうちの９つの最下位ビットは仮数部〈８１：７３〉２５４にシフトされる。符号部２５１および指数部２５２は、重みにおいて９ビット下位にシフトされる。浮動小数点ビット２５０の下位４１ビットフィールドでは、仮数部〈８：０〉は４１ビットフィールドの上位ビット〈４０：３２〉にシフトされ、仮数部〈４０：９〉は下位ビット〈３１：０〉にシフトされる。

【００６１】

８２ビットの浮動小数点レジスタ２５０は、第１の浮動小数点半オペランド２５６および第２の浮動小数点半オペランド２５８に区分される。第２の浮動小数点半オペランド２５８は右に９ビット分シフトされ、仮数ビット〈４９：４１〉２５４は最上位ビット位置に転送される。したがって、符号２５１および指数〈１６：０〉２５２は、３２ビットの整数アクセスにおいて読出および書込が可能となる。

【００６２】

ロード／ストア機能ユニット８０は、整数および浮動小数点ロードおよび記憶オペレーションを制御する。ロード／ストアユニット８０は、データキャッシュ８６からのデータに

アクセスしそのデータを結果バス３２へ転送するための、２つまでのロードオペレーションを同時に実行することができる。図１２を参照すると、ロード／ストア機能ユニット８０は、リザベーションステーション８２と記憶バッファ８４とロード／ストア制御８７とを含む。ロード／ストアリザベーションステーション８２はデュアルポートである。各ポートは、記憶バッファ８４およびデータキャッシュ８６に、４１ビットのデータと好適な数のアドレスビットとを含むチャネルによって接続される。ロード／ストアリザベーションステーション８２は、ロード／ストア要求を待ち行列に入れるＦＩＦＯバッファである。それは、オペコード／型バス５０からオペレーションコードを受取り、ロード／ストアリザベーションステーション８２への入力で多重化されるＡおよびＢオペランドバス３０および３１上の結果で動作する。

【００６３】

整数と浮動小数点との混合構造により、プロセッサは、整数データおよび浮動小数点データの両方のためのロードおよび記憶オペレーションを、同じロード／ストア機能ユニット８０を用いて行なうことが可能となる。ロード／ストア機能ユニット内において、データオペランドは４１ビットであり、整数、１つの精度数、倍精度数の一部、または拡張精度数の一部を表わす。整数データでは、最上位の８ビットは使用されない。ロード／ストア機能ユニット８０は、整数オペランドおよび浮動小数点オペランドの両方に対し同じ態様で機能する。ゆえに、整数データ経路と浮動小数点データ経路とを混合することによって、１つのロード／ストア機能ユニット８０のみを使用して、プロセッサ回路の量および複雑性を低減する。好適なロード／ストアユニットは１９９３年１０月２９日出願の米国特許出願連続番号０８／１４６，３７６（ウィリアム・エム・ジョンソンら、「高性能ロード／ストア機能ユニットおよびデータキャッシュ（Ｈｉｇｈ　Ｐｅｒｆｏｒｍａｎｃｅ　Ｌｏａｄ／Ｓｔｏｒｅ　Ｆｕｎｃｔｉｏｎａｌ　Ｕｎｉｔ　ａｎｄ　Ｄａｔａ　Ｃａｃｈｅ）」）に開示され、その全体を引用によりここに援用する。

【００６４】

この発明は特定の実施例を参照して記載されてきたが、その実施例は例示的なものであって、発明の範囲はそのように限定されるものではないと理解される。多くの変形、修正、追加および改良が、記載される実施例に対して可能である。たとえば、オペランドフィールドを２つの半オペランドに区分するのではなく、実行前に１つのオペレーションに統合される、３つまたはそれより多いサブオペレーションのディスパッチを介して操作される同数のサブオペランドに、オペランドを分割してもよい。さらに、さまざまな構造およびバスにおけるビット数は例示的なものであり、変更されてもよい。レジスタファイルおよびリオーダバッファのサイズ、オペランドバスおよびオペランドタグバスの数、結果バスの数、ライトバックバスの数、ならびに浮動小数点機能ユニットの数のような機能ユニットの型および数は、例示的なものであり、変更されてもよい。これらのおよび他の変形、修正、追加および改良は、前掲の特許請求の範囲に定義されるようなこの発明の範囲内に入るであろう。

【図面の簡単な説明】

【図１】混合された整数／浮動小数点コアを実現するためのプロセッサのアーキテクチャレベルのブロック図である。

【図２】拡張精度数、および半オペランドに区分された拡張精度数のフォーマットをそれぞれ示す模式図である。

【図３】図１のプロセッサ内のレジスタファイルのアーキテクチャレベルのブロック図である。

【図４】図３に示されるレジスタファイルにあるメモリフォーマットを示す模式図である。

【図５】図１のプロセッサ内にあるリオーダバッファのアーキテクチャレベルのブロック図である。

【図６】図５のリオーダバッファ内にあるメモリフォーマットを示す模式図である。

【図７】（ａ）は図５のリオーダバッファ内にある複数のリオーダバッファアレイ要素の

１つのリオーダバッファアレイ要素内の多重ビットフィールドを示す模式図である。（ｂ）、（ｃ）、（ｄ）、（ｅ）、および（ｆ）は、半オペランドの拡張精度浮動小数点数、倍長語整数、語整数、バイト整数、および分岐命令のためのアドレスのためのフォーマットをそれぞれ含む、（ａ）のリオーダバッファアレイ要素の結果ビットフィールドのフォーマットを示す模式図である。

【図８】混合された小数点／整数プロセッサコアのレイアウトの概略図である。

【図９】図１のプロセッサ内のデータ経路の、データの流れおよび幅を示す、概略ブロック図である。

【図１０】図１のプロセッサの機能ユニットによって動作される、拡張精度数のフォーマットを示す模式図である。

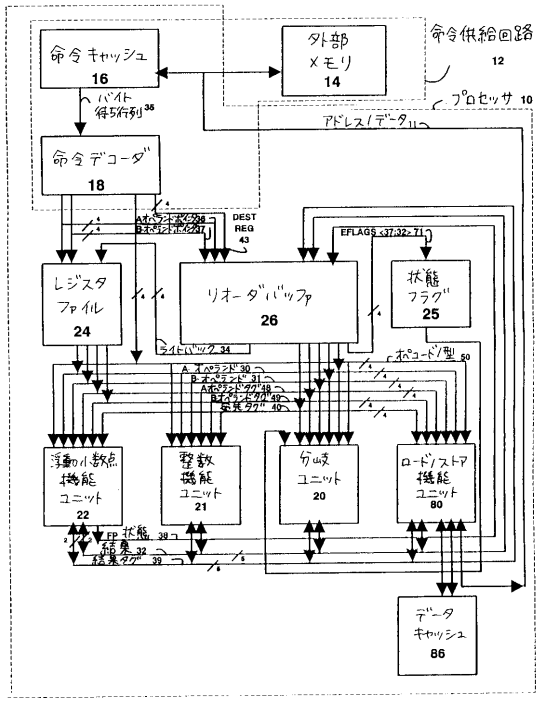
【図１１】図１のプロセッサの機能ユニットによって動作される、半オペランドに分解された、拡張精度数のフォーマットと、下位の半オペランド内のビットフィールドとを示す模式図である。

【図１２】図１のプロセッサのロード／ストア機能ユニットのアーキテクチャレベルのブロック図である。

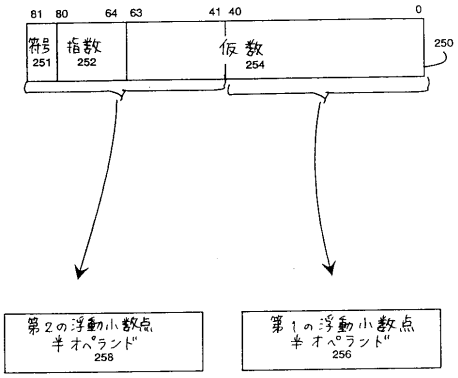
【符号の説明】

１８　命令デコーダ
２０　分岐ユニット
２１　整数機能ユニット
２２　浮動小数点機能ユニット
２４　レジスタファイル
２６　リオーダバッファ
３０　オペランドバス
３１　オペランドバス
３２　結果バス
８０　ロード／ストア機能ユニット

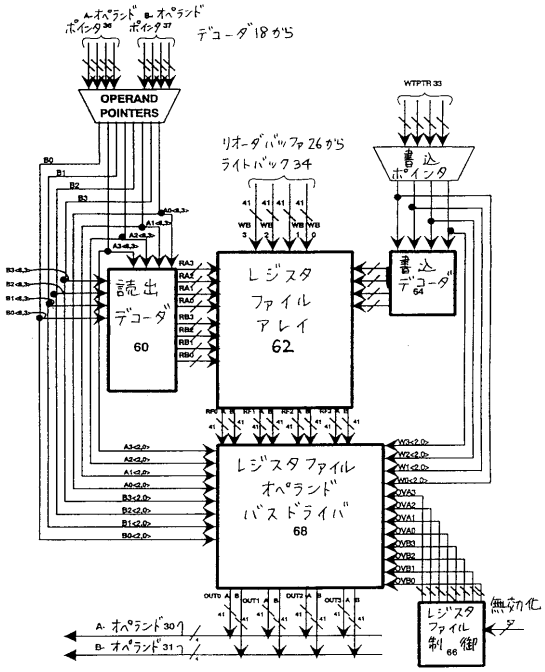
【図１】



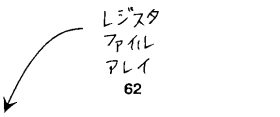
【図２】



【図３】



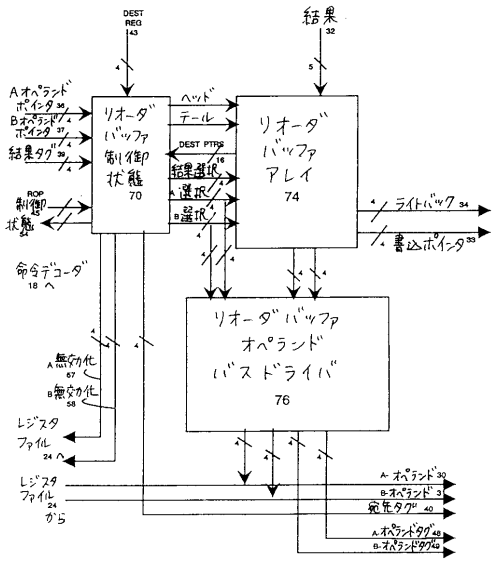
【図４】



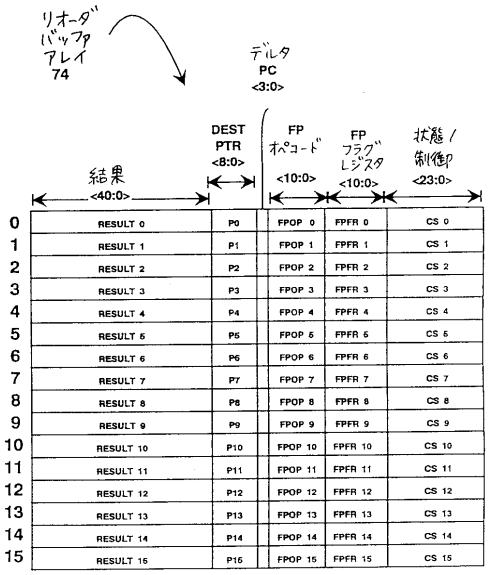
ビット 8:7 / ビット 6:3	00 <31:0>	01 <40:0>	10 <40:0>	11 <40:0>
0000	EAX	—	(L)	FP0 (H)
0001	ECX	—	(L)	FP1 (H)
0010	EDX	—	(L)	FP2 (H)
0011	EBX	—	(L)	FP3 (H)
0100	ESP	—	(L)	FP4 (H)
0101	EBP	—	(L)	FP5 (H)
0110	ESI	—	(L)	FP6 (H)
0111	EDI	—	(L)	FP7 (H)
1000	ETMP0	ETMP8	(L)	FTMP0 (H)
1001	ETMP1	ETMP9	(L)	FTMP1 (H)
1010	ETMP2	ETMP10	(L)	FTMP2 (H)
1011	ETMP3	ETMP11	(L)	FTMP3 (H)
1100	ETMP4	ETMP12	(L)	FTMP4 (H)
1101	ETMP5	ETMP13	(L)	FTMP5 (H)
1110	ETMP6	ETMP14	(L)	FTMP6 (H)
1111	ETMP7	ETMP15	(L)	FTMP7 (H)

<40:0> <81:0>

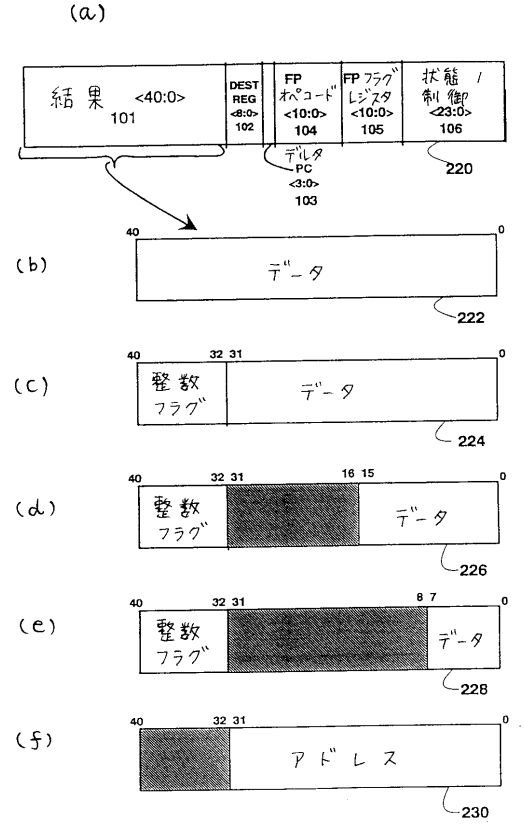
【図５】


DEST REG 43
結果 32
Aオペランド ポインタ 36
Bオペランド ポインタ 37
結果タグ 39
リオーダ バッファ 制御/状態 70
ヘッド
テール
DEST PTRS
結果選択
A選択
B選択
ROP 制御/状態 61
リオーダ バッファ アレイ 74
ライトバック 34
書込ポインタ 33
命令デコーダ 18へ
リオーダ バッファ オペランド バスドライバ 76
A無効化 67
B無効化 58
レジスタ ファイル 24
レジスタ ファイル 24 から
A-オペランド 30
B-オペランド 31
宛先タグ 40
A-オペランドタグ 48
B-オペランドタグ 49

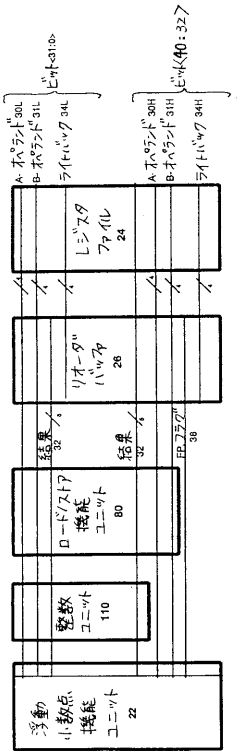
【図６】


リオーダ バッファ アレイ 74
デルタ PC <3:0>
結果 <40:0>
DEST PTR <8:0>
FP オペコード <10:0>
FP フラグ レジスタ <10:0>
状態/制御 <23:0>
0 RESULT 0 P0 FPOP 0 FPFR 0 CS 0
1 RESULT 1 P1 FPOP 1 FPFR 1 CS 1
2 RESULT 2 P2 FPOP 2 FPFR 2 CS 2
3 RESULT 3 P3 FPOP 3 FPFR 3 CS 3
4 RESULT 4 P4 FPOP 4 FPFR 4 CS 4
5 RESULT 5 P5 FPOP 5 FPFR 5 CS 5
6 RESULT 6 P6 FPOP 6 FPFR 6 CS 6
7 RESULT 7 P7 FPOP 7 FPFR 7 CS 7
8 RESULT 8 P8 FPOP 8 FPFR 8 CS 8
9 RESULT 9 P9 FPOP 9 FPFR 9 CS 9
10 RESULT 10 P10 FPOP 10 FPFR 10 CS 10
11 RESULT 11 P11 FPOP 11 FPFR 11 CS 11
12 RESULT 12 P12 FPOP 12 FPFR 12 CS 12
13 RESULT 13 P13 FPOP 13 FPFR 13 CS 13
14 RESULT 14 P14 FPOP 14 FPFR 14 CS 14
15 RESULT 15 P15 FPOP 15 FPFR 15 CS 15

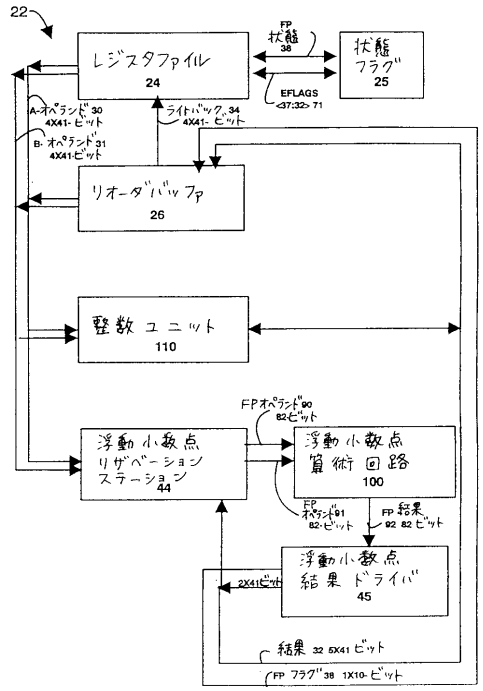
【図７】


(a)
結果 <40:0> 101
DEST REG <8:0> 102
FP オペコード <10:0> 104
FP フラグ レジスタ <10:0> 105
状態/制御 <23:0> 106
220
デルタ PC <3:0> 103
(b)
データ
222
(c)
整数フラグ
データ
224
(d)
整数フラグ
データ
226
(e)
整数フラグ
データ
228
(f)
アドレス
230

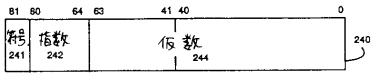
【図８】


Aオペランド 30L
Bオペランド 31L
ライトバック 34L
ビット<31:0>
Aオペランド 30H
Bオペランド 31H
ライトバック 34H
ビット<40:32>
レジスタ ファイル 24
リオーダ バッファ 28
結果 32
結果 32
FPフラグ 38
ロード/ストア 機能 ユニット 80
整数 ユニット 110
浮動小数点 機能 ユニット 22

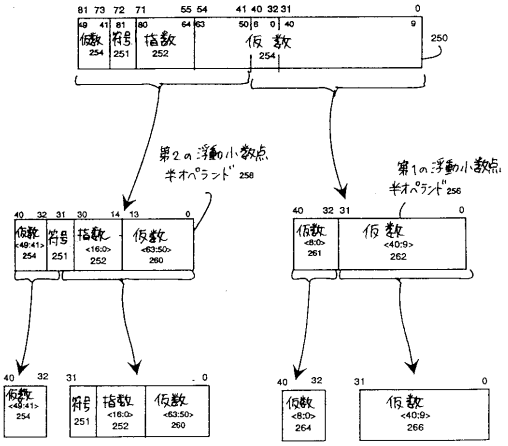
【図９】



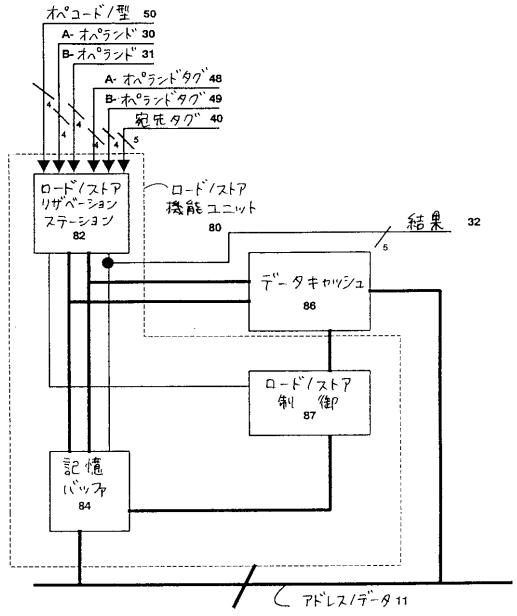
【図１０】



【図１１】



【図１２】



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 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
(72)発明者 ウィリアム・エム・ジョンソン
アメリカ合衆国、７８７４６ テキサス州、オースティン、クリスティー・ドライブ、１０２

審査官 後藤 彰

(56)参考文献 特開平６－１０３１６７（ＪＰ，Ａ）
特表平６－５０１８０５（ＪＰ，Ａ）
特表平６－５０１１２４（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.[7]，ＤＢ名)
G06F 9/30 - 9/38