(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2004-152318
(P2004-152318A)
(43)公開日　平成16年5月27日(2004.5.27)

(51)Int.Cl.[7]		F I			テーマコード（参考）
G06F	9/30	G06F	9/30	350E	5B022
G06F	7/00	G06F	9/34	330	5B033
G06F	9/34	G06F	7/00	R	
G06F	9/455	G06F	9/44	310A	

審査請求 有　請求項の数 27　OL　（全 65 頁）

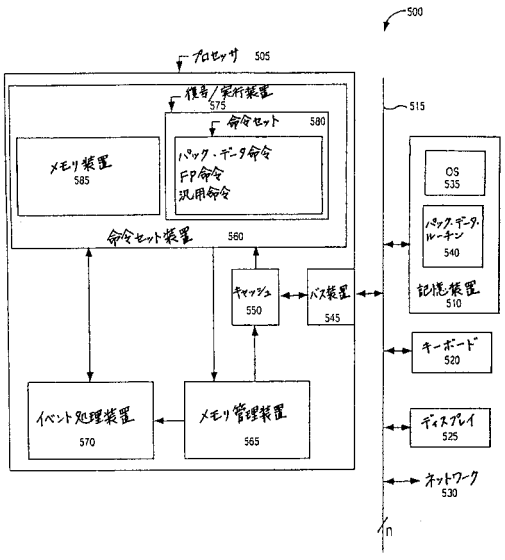
(21)出願番号　特願2003-417823(P2003-417823)
(22)出願日　平成15年12月16日(2003.12.16)
(62)分割の表示　特願平9-523030の分割
原出願日　平成8年12月17日(1996.12.17)
(31)優先権主張番号　08/574,719
(32)優先日　平成7年12月19日(1995.12.19)
(33)優先権主張国　米国(US)

(71)出願人　591003943
インテル・コーポレーション
アメリカ合衆国　９５０５２　カリフォルニア州・サンタクララ・ミッション　カレッジ　ブーレバード・２２００
(74)代理人　100064621
弁理士　山川　政樹
(72)発明者　リン，デリック
アメリカ合衆国・９４４０４・カリフォルニア州・フォスターシティ・バーケンティン　ストリート・１１３
(72)発明者　ヴァッカラガッダ，ラマモハン・アール
アメリカ合衆国・９５１３５・カリフォルニア州・サン　ホゼ・ミラージュ　ウェイ・３２９０

最終頁に続く

(54)【発明の名称】単一のレジスタ・ファイルを使用して浮動小数点命令およびパック・データ命令を実行する方法および装置

(57)【要約】　（修正有）
【課題】既存のソフトウェアおよびハードウェアとの互換性を有するように、パック・データを処理する１組の命令をプロセッサに組み込む。
【解決手段】少なくとも部分的にエイリアス化され、複数のタグが対応している単一の論理レジスタ・ファイルの内容に対して、パック・データ命令のセットの実行を浮動小数点命令のセットの実行前に双方実行し、パック・データ命令セットの第１の命令の実行開始と浮動小数点命令セットの第１の命令の実行の完了との間のある時点に、少なくとも単一の論理レジスタ・ファイル中のエイリアス化したレジスタに対応する複数のタグを非空状態に変更し、タグが、前記単一の論理レジスタ・ファイル中のレジスタが空であるか空でないかを識別する。
【選択図】図５



【特許請求の範囲】
【請求項１】
データ処理装置で命令を実行する方法であって：
少なくとも部分的にエイリアス化され、複数のタグが対応している単一の論理レジスタ・ファイルの内容に対してパック・データ命令のセットと浮動小数点命令のセットとを実行するステップであって、前記パック・データ命令のセットは前記浮動小数点命令のセットより前に実行されるステップと；
前記パック・データ命令セットの第１の命令の実行を試みることと、前記浮動小数点命令セットの第１の命令の実行を完了することとの間のある時点に、少なくとも前記単一の論理レジスタ・ファイル中のエイリアス化したレジスタに対応する前記複数のタグを非空状態に変更するステップであって、前記複数のタグは、前記単一の論理レジスタ・ファイル中のレジスタが空であるか空でないかを識別するステップと
を含むことを特徴とする方法。
【請求項２】
前記実行ステップは：
フラット・レジスタ・ファイルとしての前記単一の論理レジスタ・ファイルに対して前記パック・データ命令のセットを実行するステップと；
スタック参照レジスタ・ファイルとしての前記単一の論理レジスタ・ファイルに対して前記浮動小数点命令のセットを実行するステップと
を含むことを特徴とする請求項１に記載の方法。
【請求項３】
データ処理装置で命令を実行する方法であって：
単一の論理レジスタ・ファイルの内容に対して第１の命令タイプの第１の命令セットを実行するステップであって、前記単一の論理レジスタ・ファイルは、前記第１の命令セットを実行する間はフラット・レジスタ・ファイルとして操作されるステップと；
同様に前記単一の論理レジスタ・ファイルの内容に対して第２の命令タイプの第１の命令を実行するステップであって、前記単一の論理レジスタ・ファイルは、前記第１の命令を実行する間はスタック参照レジスタ・ファイルとして操作されるステップと
を含むことを特徴とする方法。
【請求項４】
フラット・レジスタ・ファイルとしての前記単一の論理レジスタ・ファイルに第１の命令セットを実行する前記ステップはさらに、パック・データに対する操作を実行することを含むことを特徴とする請求項３に記載の方法。
【請求項５】
スタック参照レジスタ・ファイルとしての前記単一の論理レジスタ・ファイルに命令を実行する前記ステップは、スカラー・データに対して操作を実行させることを特徴とする請求項２ないし４のいずれか１項に記載の方法。
【請求項６】
実行ステップはさらに、スカラー浮動小数点演算を行うステップを含むことを特徴とする請求項１、２または５のいずれか１項に記載の方法。
【請求項７】
パック・データ命令を実行するステップはさらに、パック浮動小数点演算を行うステップを含むことを特徴とする請求項１、２または４ないし６のいずれか１項に記載の方法。
【請求項８】
パック・データ命令を実行するステップはさらに、パック整数データに対して操作を行うステップを含むことを特徴とする請求項１、２または４ないし７のいずれか１項に記載の方法。
【請求項９】
前記第２の命令タイプの前記第１の命令を実行する前記ステップはさらに、
前記単一の論理レジスタ・ファイルに含まれるデータをメモリにコピーするステップ

を含むことを特徴とする請求項３または４に記載の方法。
【請求項１０】
フラット・レジスタ・ファイルとしての前記単一の論理レジスタ・ファイルに命令を実行する前記ステップの開始から、スタック参照レジスタ・ファイルとしての前記単一の論理レジスタ・ファイルに前記第１の命令を実行する前記ステップを完了するまでの間のある時点に、前記単一の論理レジスタ・ファイル中の１つのレジスタを現在のトップ・オブ・スタック・レジスタとして識別するトップ・オブ・スタック表示を初期設定値に変更するステップ
をさらに含むことを特徴とする請求項２ないし９のいずれか１項に記載の方法。
【請求項１１】
フラット・レジスタ・ファイルとしての前記単一の論理レジスタ・ファイルに対して命令を実行する前記ステップの開始から、スタック参照レジスタ・ファイルとしての前記単一の論理レジスタ・ファイルに対して前記第１の命令を実行する前記ステップを開始するまでのある時点に、フラット・レジスタ・ファイルとしての前記単一の論理レジスタ・ファイルに対して命令を実行するステップ中に書き込みが行われる前記単一の論理レジスタ・ファイル中の各レジスタの符号および指数のフィールドに、非数値または無限を示す値を書き込むステップ
をさらに含むことを特徴とする請求項２ないし１０のいずれか１項に記載の方法。
【請求項１２】
フラット・レジスタ・ファイルとしての前記単一の論理レジスタ・ファイルに対して命令を実行する前記ステップの開始から、スタック参照レジスタ・ファイルとしての前記単一の論理レジスタ・ファイルに対して前記第１の命令を実行する前記ステップの完了まで間のある時点に、前記単一の論理レジスタ・ファイルに対応し、前記単一の論理レジスタ・ファイル中のレジスタが空であるか、空でないかを識別する複数のタグを非空状態に変更するステップ
をさらに含むことを特徴とする請求項２ないし１１のいずれか１項に記載の方法。
【請求項１３】
前記タグを変更するステップは、フラット・レジスタ・ファイルとしての前記単一の論理レジスタ・ファイルに対して、前記第１の命令の実行を試みることか、または前記命令の第１の命令を実行することに応答して実行されることを特徴とする請求項２または１２に記載の方法。
【請求項１４】
前記タグを変更するステップは、フラット・レジスタ・ファイルとしての前記単一の論理レジスタ・ファイルに対して前記命令の第１の命令の実行を試みることに応答して実行されることを特徴とする請求項２または１２に記載の方法。
【請求項１５】
前記タグを変更するステップは、フラット・レジスタ・ファイルとしての前記単一の論理レジスタ・ファイルに対して前記命令のそれぞれを実行することに応答して実行されることを特徴とする請求項２または１２に記載の方法。
【請求項１６】
前記タグを変更するステップは、フラット・レジスタ・ファイルとしての前記単一の論理レジスタ・ファイルに対して前記命令を実行する前記ステップと、スタック参照レジスタ・ファイルとしての前記単一の論理レジスタ・ファイルに対して前記第１の命令を実行するステップとの間に実行されることを特徴とする請求項２または１２ないし１５のいずれか１項に記載の方法。
【請求項１７】
請求項３ないし１１のいずれか１項に記載の方法を実行するデータ処理装置であって：
複数の物理レジスタと；
前記複数の物理レジスタがソフトウェアには前記単一の論理レジスタ・ファイルとして見えるようにするメモリ装置と；

請求項３ないし１１のいずれか１項に記載の方法により、フラット・レジスタ・ファイルとしての前記単一の論理レジスタ・ファイルに対して対して命令を実行し、スタック参照レジスタ・ファイルとしての前記単一の論理レジスタ・ファイルに対して対して命令を実行する復号／実行装置と
を備えることを特徴とするデータ処理装置。
【請求項１８】
請求項１ないし１６のいずれか１項に記載の方法を実行するデータ処理装置であって：
複数の物理レジスタと；
前記複数の物理レジスタが、ソフトウェアには、スカラー浮動小数点データを保存する前記単一の論理レジスタ・ファイルとして見えるようにするメモリ装置と；
請求項１ないし１６のいずれか１項に記載の方法に従って前記第１のレジスタの小数部フィールドに書き込まれる第１のパック・データ項目を、前記単一の論理レジスタ・ファイル中の第１のレジスタに書き込ませる第１のパック・データ命令を受け取る復号／実行装置と
を備えることを特徴とするデータ処理装置。
【請求項１９】
請求項１ないし１６のいずれか１項に記載の方法を実行するデータ処理装置であって：
複数の物理レジスタと；
前記複数の物理レジスタがソフトウェアには前記単一の論理レジスタ・ファイルとして見えるようにするメモリ装置と；
請求項１ないし１６のいずれか１項に記載の方法により、前記単一の論理レジスタ・ファイルの内容に対して命令を実行する復号／実行装置と；
部分的なコンテキスト切り替えによる前記単一の論理レジスタ・ファイルの使用不可、または使用可能を判定するイベント処理装置であって、使用不可と判定されると、前記データ処理装置は第１のルーチンの実行に割り込み、第２のルーチンを実行して前記単一の論理レジスタ・ファイルの内容をメモリにコピーするイベント処理装置と
を備えることを特徴とするデータ処理装置。
【請求項２０】
エミュレーション状態を示すＥＭフィールドを含む少なくとも１つのステータス・レジスタをさらに含み、エミュレーション状態が示されると、前記プロセッサは前記第１のルーチンの実行に割り込むことを特徴とする請求項１９に記載のデータ処理装置。
【請求項２１】
前記復号／実行装置は、前記ＥＭフィールドがエミュレーション状態を示し、かつ前記復号／実行装置が前記第１のルーチンに属する第１のスカラー・データ命令を受け取った場合に、前記第２のルーチンを実行することを特徴とする請求項２０に記載のデータ処理装置。
【請求項２２】
前記復号／実行装置は、前記ＥＭフィールドがエミュレーション状態を示し、かつ前記復号／実行装置が前記第１のルーチンに属する第１のパック・データ命令を受け取った場合に、第３のルーチンを実行することを特徴とする請求項２０または２１に記載のデータ処理装置。
【請求項２３】
前記データ処理装置は、前記イベント処理装置が、前記ソフトウェアに対して可視の単一のレジスタ・ファイルが使用可能であると判断し、かつ前記復号／実行装置が、前記復号／実行装置に前記第１のパック・データ項目を前記単一の論理レジスタ・ファイルに対して書き込ませる前記第１のルーチンに属する命令を受け取った場合に、第１の小数部フィールドに第１のパック・データ項目を書き込み、前記単一の論理レジスタ・ファイル中の第１のレジスタの第１の符号および指数フィールドに、非数値または無限を表す第１の値を書き込むことを特徴とする請求項１９ないし２２のいずれか１項に記載のデータ処理装置。

【請求項２４】
トップ・オブ・スタック・フィールドを有する少なくとも１つのステータス・レジスタをさらに備え、前記トップ・オブ・スタック・フィールドは、前記イベント処理装置が前記単一の論理レジスタ・ファイルが使用可能であると判定し、かつ
前記復号／実行装置が、前記第１のルーチンに属する第１のパック・データ命令を受け取った；
前記復号／実行装置が、前記第１のルーチンに属する第１のスカラー命令を受け取り、かつ、スタック参照レジスタ・ファイルとしての前記単一の論理レジスタ・ファイルに対して作用する第２のスカラー命令よりも後に、フラット・レジスタ・ファイルとしての前記単一の論理レジスタ・ファイルに対して作用する第２のパック・データ命令を実行した；または
前記復号／実行装置が、前記第１のルーチンに属する前記第１のパック・データ命令を受け取り、かつ、フラット・レジスタ・ファイルとしての前記単一の論理レジスタ・ファイルに対して作用する第３のパック・データ命令よりも後に、スタック参照レジスタ・ファイルとしての前記単一の論理レジスタ・ファイルに対して作用する第３のスカラー命令を実行した
ことからなる第１の条件セットの１つである第１の条件が存在する場合に、初期設定値に変更されることを特徴とする請求項１９に記載のデータ処理装置。
【請求項２５】
請求項２または１２ないし１６のいずれか１項に記載の方法を実行するデータ処理装置であって、
複数の物理レジスタと；
前記複数の物理メモリがソフトウェアには前記単一の論理レジスタ・ファイルとして見えるようにするメモリ装置と；
請求項２または１２ないし１６のいずれか１項に記載の方法により、前記単一の論理レジスタ・ファイルの内容に対して命令を実行する復号／実行装置と；
請求項２または１２ないし１６のいずれか１項に記載の方法により、前記複数のタグのうちのタグを変更するタグ修飾装置と
を備えることを特徴とするデータ処理装置。
【請求項２６】
部分的なコンテキスト切り替えによる前記単一の論理レジスタ・ファイルの使用不可、または使用可能を判定するイベント処理装置であって、使用不可と判定されると、前記データ処理装置は第１のルーチンの実行に割り込み、第２のルーチンを実行して前記単一の論理レジスタ・ファイルの内容をメモリにコピーするイベント処理装置と；
タグ修飾装置であって、前記イベント処理装置が前記単一の論理レジスタ・ファイルが使用可能であると判断し、かつ、
前記復号／実行装置が、前記第１のルーチンに属する非遷移パック・データ命令を受け取った；
前記復号／実行装置が、前記第１のルーチンに属する第１のスカラー命令を受け取り、かつ、スタック参照レジスタ・ファイルとしての前記単一の論理レジスタ・ファイルに対して作用する命令の１つより後に、かつ第１の遷移命令よりも後に、第１のパック・データ命令を実行した；または
前記復号／実行装置が、前記第１のルーチンに属する第２のパック・データ命令を受け取り、かつ、フラット・レジスタ・ファイルとしての前記単一の論理レジスタ・ファイルに対して作用する命令の１つよりも後に、スタック参照レジスタ・ファイルとしての前記単一の論理レジスタ・ファイルに対して作用する命令の１つを実行した；
ことからなる第１の条件セットの１つである第１の条件が存在する場合に、前記単一の論理レジスタ・ファイル中の異なるレジスタに対応する前記複数のタグ中の各タグを非空状態に変更する前記タグ修飾装置と
をさらに備えることを特徴とする請求項２５に記載のデータ処理装置。

【請求項２７】
前記タグ修飾装置は、
前記復号／実行装置が、前記第１のルーチンに属する遷移命令を受け取った；または
前記復号／実行装置が、前記第１のルーチンに属する第２のスカラー命令を受け取り、かつ、スタック参照レジスタ・ファイルとしての前記単一の論理レジスタ・ファイルに対して作用する命令の１つよりも後に、かつフラット・レジスタ・ファイルとしての前記単一の論理レジスタ・ファイルに対して作用する命令のどの他の命令よりも後に、前記遷移命令を実行した
ことからなる第２の条件セットの１つである第２の条件が存在する場合に、前記複数のタグを空状態に変更することを特徴とする請求項２５または２６に記載のデータ処理装置。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、コンピュータ・システムの分野に関する。詳細には、本発明は、プロセッサによって浮動小数点命令およびパック・データ命令を実行することに関する。
【背景技術】
【０００２】
典型的なコンピュータ・システムでは、プログラムされた命令に応答して、１つまたは複数のプロセッサが、多数のビット（たとえば、１６、３２、６４など）で表されたデータ値を処理して結果を生成する。たとえば、加算命令を実行すると、第１のデータ値と第２のデータ値が加えられ、結果が第３のデータ値として記憶される。しかし、マルチメディア・アプリケーション（たとえば、コンピュータ・サポーテッド・コーペレーション（ＣＳＣ－電話会議と混合媒体データ処理を統合したもの）、２Ｄ／３Ｄグラフィックス、画像処理、ビデオ圧縮／圧縮解除、認識アルゴリズム、オーディオ処理を対象とするアプリケーション）では、より小数のビットで表されることが多い大量のデータを処理する必要がある。たとえば、マルチメディア・データは通常、６４ビット数で表されるが、有意の情報を伝達するのは小数のビットだけである。
【０００３】
マルチメディア・アプリケーション（ならびに同じ特性を有するその他のアプリケーション）の効率を向上させるために、従来技術のプロセッサはパック・データ・フォーマットを使用する。パック・データ・フォーマットとは、単一の値を表すために使用されるビットが、それぞれ、別々の値を表す、いくつかの固定サイズのデータ要素に分割されるフォーマットである。たとえば、６４ビット・レジスタ内のデータは、それぞれ、別々の３２ビット値を表す、２つの３２ビット要素に分割することができる。
【０００４】
ヒューレット・パッカード社の基本３２ビット・アーキテクチャ・マシンはこの手法を使用してマルチメディア・データ・タイプを実装していた。すなわち、プロセッサは３２ビット汎用整数レジスタを並行して使用して６４ビット・データ・タイプを実施していた。この簡単な手法の主要な欠点は、利用可能なレジスタ空間が厳しく制限されることである。また、既存のアーキテクチャを拡張するために必要な作業に鑑み、このようにマルチメディア・データを処理することの性能上の利点は最小限のものであるとみなされている。
【０００５】
Ｍｏｔｏｒｏｌａ（登録商標）８８１１０™プロセッサで使用されている同様な手法では、整数レジスタ対が組み合わされる。２つの３２ビット・レジスタを対にするこの方法では、単一の演算または命令に関して指定されたレジスタのランダム組合せが連結される。しかし、この場合も、レジスタ対を使用して６４ビット・マルチメディア・データ・タイプを用いることの主要な欠点は、限られた数のレジスタ対しか使用できないということである。アーキテクチャに追加レジスタ空間を加えずに、マルチメディア・データ・タイ

プを用いる他の技法が必要である。

【０００６】

大規模なソフトウェア・ハードウェア・ベースを有する１つのプロセッサ・ラインは、カリフォルニア州サンタクララのインテル社によって製造されている、Ｐｅｎｔｉｕｍ（登録商標）プロセッサを含むインテル・アーキテクチャ・プロセッサ・ファミリである。図１は、Ｐｅｎｔｉｕｍプロセッサを使用した例示的なコンピュータ・システム１００を示すブロック図である。この図に示したよりも詳しいＰｅｎｔｉｕｍプロセッサの説明については、カリフォルニア州サンタクララのインテル社から入手することのできる「Ｐｅｎｔｉｕｍ Ｐｒｏｃｅｓｓｏｒ’ｓ Ｕｓｅｒｓ Ｍａｎｕａｌ－Ｖｏｌｕｍｅ ３；Ａｒｃｈｉｔｅｃｔｕｒｅ ａｎｄ Ｐｒｏｇｒａｍｍｉｎｇ Ｍａｎｕａｌ」（１９９４年）を参照されたい。例示的なコンピュータ・システム１００は、プロセッサ１０５と、記憶装置１１０と、バス１１５とを含む。プロセッサ１０５はバス１１５によって記憶装置１１０に結合される。また、キーボード１２０やディスプレイ１２５などいくつかのユーザ入出力装置もバス１１５に結合される。ネットワーク１３０をバス１１５に結合することもできる。プロセッサ１０５はＰｅｎｔｉｕｍプロセッサを表す。記憶装置１１０は、データを記憶する１つまたは複数の機構を表す。たとえば、記憶装置１１０には、読取り専用メモリ（ＲＯＭ）や、ランダム・アクセス・メモリ（ＲＡＭ）や、磁気ディスク記憶媒体や、光学記憶媒体や、フラッシュ・メモリ装置や、その他の機械可読媒体を含めることができる。バス１１５は、１つまたは複数のバス（たとえば、ＰＣＩ、ＩＳＡ、Ｘ－Ｂｕｓ、ＥＩＳＡ、ＶＥＳＡなど）およびブリッジ（バス・コントローラとも呼ばれる）を表す。

【０００７】

図１は、プロセッサ１０５上で実行されるオペレーティング・システム１３２が記憶装置１１０に記憶されていることも示している。もちろん、記憶装置１１０は好ましくは、追加ソフトウェア（図示せず）を含む。図１は、プロセッサ１０５が浮動小数点装置１３５と浮動小数点状態レジスタ１５５とを含む（本明細書では、「浮動小数点」を示すために表記「ＦＰ」を使用する）ことも示している。もちろん、プロセッサ１０５は、本発明を理解するうえで必要とされない追加回路を含む。

【０００８】

浮動小数点装置１３５は、浮動小数点データを記憶するために使用され、１組の浮動小数点レジスタ（浮動小数点レジスタ・ファイルとも呼ぶ）１４５と、１組のタグ１５０と、浮動小数点状態レジスタ１５５とを含む。１組の浮動小数点レジスタ１４５は、Ｒ０ないし Ｒ７（本明細書では、浮動小数点レジスタの物理的位置を示すために表記Ｒｎを使用している）で示された８つのレジスタを含む。この８つのレジスタはそれぞれ、８０ビット幅であり、符号フィールド（ビット７９）と、指数フィールド（ビット［７８：６４］）と、小数部フィールド（ビット［６３：０］）とを含む。浮動小数点装置１３５は１組の浮動小数点レジスタ１４５をスタックとして操作する。言い換えれば、浮動小数点装置１３５はスタック参照レジスタ・ファイルを含む。１組のレジスタをスタックとして操作すると、動作は、１組の浮動小数点レジスタ１４５内のレジスタの物理的位置ではなくスタックの１番上の浮動小数点レジスタを参照することによって実行される（本明細書では、スタックの１番上の浮動小数点レジスタに対する論理浮動小数点レジスタｎの相対位置を示すために表記ＳＴｎを使用する）。浮動小数点状態レジスタ１５５は、１組の浮動小数点レジスタ１４５内のどのレジスタが現在、浮動小数点スタックの１番上にあるかを識別するトップ・オブ・スタック・フィールド１６０を含む。図１では、トップ・オブ・スタック表示は、物理位置Ｒ４にあるレジスタ１６５をスタックの１番上のレジスタとして識別している。

【０００９】

１組のタグ１５０は８つのタグを含み、単一のレジスタに格納される。各タグは、それぞれの異なる浮動小数点レジスタに対応し、２つのビットを備える。図１に示したように、タグ１７０はレジスタ１６５に対応する。タグは、そのタグが対応する浮動小数点レジ

スタの現在の内容に関する情報を識別する。すなわち、００＝有効、０１＝零、１０＝特殊、１１＝空である。これらのタグは、空レジスタ位置と非空レジスタ位置を区別するために浮動小数点装置１３５によって使用される。したがって、タグは、１１で示される空と、００、０１、１０のうちの１つで示される非空の２つの状態を識別するともみることができる。

【００１０】

これらのタグを使用してイベントを処理することもできる。「イベント」とは、ハードウェア割込み、ソフトウェア割込み、例外、障害、トラップ、アボート、マシン・チェック、アシスト、デバッグ・イベントとを含め、コンピュータ・システムが応答することのできるアクションまたはオカレンスである。イベントを受け取ると、プロセッサのイベント処理機構によって、プロセッサは現プロセスの実行に割り込み、割り込まれたプロセスの実行環境（すなわち、割り込まれたプロセスの実行を再開するのに必要な情報）を記憶し、適切なイベント・ハンドラを呼び出しイベントを処理する。イベントを処理した後、イベント・ハンドラはプロセッサに、すでに記憶されているプロセスの実行環境を使用して、割り込まれたプロセスを再開させる。イベント・ハンドラのプログラマは、これらのタグを使用してそれぞれの異なる浮動小数点レジスタの内容を検査し、イベントをよりうまく処理することができる。

【００１１】

各タグを２つのビットを含むものとして説明したが、代替実施形態では、各タグごとに１ビットのみを格納することができる。このような１ビット・タグはそれぞれ、空と非空のどちらかを識別する。そのような実施形態では、タグ値が必要なときに適切な２ビット・タグ値を決定することによって、このような１ビット・タグをユーザからは２ビットを備えるように見えるようにすることができる。

【００１２】

状態レジスタ１４０は、それぞれ、ＥＭ表示とＴＳ表示を格納する、ＥＭフィールド１７５とＴＳフィールド１８０を含む。ＥＭ表示が１であり、あるいはＴＳ表示が１であり、あるいはその両方である場合、プロセッサ・ハードウェアにより、浮動小数点命令実行時に「装置使用不能」例外が発生することによってオペレーティング・システムがトラップされる。ソフトウェア規約によれば、ＥＭ表示およびＴＳ表示はそれぞれ、浮動小数点命令をエミュレートし、マルチタスクを実施するために使用される。しかし、これらの表示を使用することはソフトウェア規約に過ぎない。したがって、どちらかの表示または両方の表示を任意の目的に使用することができる。たとえば、ＥＭ表示を使用してマルチタスクを実施することができる。

【００１３】

前述のソフトウェア規約によれば、ＥＭフィールド１７５は、ソフトウェアを使用して浮動小数点装置をエミュレートすべきであるかどうかを識別する浮動小数点エミュレート表示（「ＥＭ表示」）を格納するために使用される。システムのブート時には通常、一連の命令または単一の命令（たとえば、ＣＰＵＩＤ）が実行され、浮動小数点装置が存在するかどうかが判定され、必要に応じてＥＭ表示が変更される。したがって、ＥＭ表示は通常、プロセッサが浮動小数点装置を含まないときに浮動小数点装置をエミュレートすべきであることを示すように変更される。一実施態様では、浮動小数点装置をエミュレートすべきであるときＥＭ表示は１に等しく、代替実施態様では他の値を使用することができる。

【００１４】

オペレーティング・システムを使用することによって、多くのプロセッサは、コオペラティブ・マルチタスクやタイムスライス・マルチタスクなどの技法を使用していくつかのプロセス（本明細書ではタスクと呼ぶ）をマルチタスク化することができる。プロセッサは、一度に１つのタスクしか実行できないので、様々なタスクを切り換えることによって、様々なタスク間で処理時間を分割しなければならない。プロセッサがあるタスクから他のタスクに切り換えたとき、タスク切換（「コンテキスト切換」または「プロセス切換」

とも呼ぶ）が行われたと言う。タスク切換を実行するには、プロセッサはあるタスクの実行を停止し、他のタスクの実行を再開または開始しなければならない。タスク切換の後にタスクの実行を再開するために内容を保存しなければならない（浮動小数点レジスタを含む）いくつかのレジスタがある。タスクの実行時の所与の時間のこれらのレジスタの内容をそのタスクの「レジスタ状態」と呼ぶ。いくつかのプロセスをマルチタスク化する間、タスクの「レジスタ状態」は、他のプロセスの実行時に、プロセッサの外部のメモリに含まれるデータ構造（タスクの「コンテキスト構造」と呼ぶ）に格納されることによって保存される。タスクの実行を再開するときは、タスクのコンテキスト構造を使用してタスクのレジスタ状態が復元される（すなわち、プロセッサにロードし直される）。

【００１５】

タスクのレジスタ状態の保存および復元は、いくつかの異なる技法を使用して行うことができる。たとえば、あるオペレーティング・システムは、前のタスクのレジスタ状態全体を記憶し、各タスク切換時に次のタスクのレジスタ状態全体を復元する。しかし、レジスタ状態全体を記憶し復元するのは時間がかかるので、タスク切換時に不要な部分を記憶および／または復元することを回避することが望ましい。タスクが浮動小数点装置を使用しない場合、浮動小数点レジスタの内容をそのタスクのレジスタ状態の一部として記憶し復元することは不要である。したがって、従来、タスク切換時の浮動小数点レジスタの内容の記憶および復元を回避するために、前述のソフトウェア規約に従って、オペレーティング・システムによってＴＳ表示が使用されている（一般に、「部分コンテキスト切換」または「オンデマンド・コンテキスト切換」と呼ぶ）。

【００１６】

ＴＳ表示を使用して部分コンテキスト切換を実施することは良く知られている。しかし、本発明では、部分コンテキスト切換が実行された（すなわち、浮動小数点装置が「使用不能」であり、あるいは「ディスエーブル」されている）ＴＳ表示が示すときに浮動小数点命令を実行しようすると、「装置使用不能」例外が生じることが重要である。イベント・ハンドラは、この例外に応答して、プロセッサに作用し、現タスクが浮動小数点装置のオーナーであるかどうか（浮動小数点装置に記憶されているデータが現タスクまたはすでに実行されたタスクに属するものであるかどうか）を判定する。現タスクがオーナーではない場合、イベント・ハンドラによって、プロセッサは浮動小数点レジスタの内容を前のタスクのコンテキスト構造に格納し、現タスクの浮動小数点状態（使用可能である場合）を復元し、現タスクをオーナーとして識別する。しかし、現タスクが浮動小数点装置のオーナーである場合、現タスクは浮動小数点装置を最後に使用したタスクであり（現タスクのレジスタ状態の浮動小数点部分はすでに浮動小数点装置に記憶されている）、浮動小数点装置に対して処置を施す必要はなく、ＴＳは設定されず、例外は生じない。ハンドラを実行すると、プロセッサはまた、浮動小数点装置が現タスクによって所有されている（「使用可能」または「イネーブル」とも呼ぶ）ことを示すようにＴＳ表示を変更する。

【００１７】

イベント・ハンドラの完了時に、装置使用不能例外を生じさせた浮動小数点命令を再開することによって現タスクの実行が再開される。ＴＳ表示は、浮動小数点装置が使用可能であることを示すように変更されているので、後続の浮動小数点命令を実行しても装置使用不能例外は生じない。しかし、次の部分コンテキスト切換時には、部分コンテキスト切換が実行されたことを示すようにＴＳ表示が変更される。したがって、他の浮動小数点命令の実行が試みられた場合、他の装置使用不能例外が発生し、再びイベント・ハンドラが実行される。このように、ＴＳ表示によって、オペレーティング・システムは浮動小数点レジスタ・ファイルの保存およびロードを遅延させ、場合によっては回避することができる。そうすることにより、保存しロードしなければならないレジスタの数を削減することによってタスク切換オーバヘッドが低減される。

【００１８】

タスク切換時には浮動小数点状態が記憶されることも、あるいは復元されることもないあるオペレーティング・システムについて説明したが、代替実施形態では任意の数の他の

技法を使用することができる。たとえば、前述のように、常に各タスク切換時にレジスタ状態全体を記憶し復元するようにオペレーティング・システムを構成することができる。

【００１９】

プロセスの浮動小数点状態を記憶できるいくつかの異なる時間（たとえば、装置使用不能イベントなどに応答したコンテキスト切換時など）だけでなく、浮動小数点状態を記憶するいくつかの異なる技法もある。たとえば、浮動小数点状態全体を記憶するようにオペレーティング・システムを実装することができる（本明細書では「単純タスク切換」と呼ぶ）。別法として、対応するタグが非空状態を示す浮動小数点レジスタのみの内容を記憶するようにオペレーティング・システムを実装することもできる（本明細書では「最小タスク切換」と呼ぶ）。そうする際に、オペレーティング・システムは、有用なデータを含む浮動小数点レジスタのみの内容を記憶する。このように、保存しなければならないレジスタの数を削減することによって、浮動小数点状態の記憶に関するオーバヘッドを低減することができる。

【００２０】

図２は、Ｐｅｎｔｉｕｍプロセッサによる命令の実行を示す流れ図である。この流れ図はステップ２００から開始し、フローはステップ２００からステップ２０５に進む。

【００２１】

ステップ２０５に示したように、１組のビットが命令としてアクセスされ、フローはステップ２１０に進む。この１組のビットは、この命令によって実行される動作を識別する命令コードを含む。

【００２２】

ステップ２１０で、命令コードが有効であるかどうかが判定される。命令コードが有効ではない場合、フローはステップ２１５に進む。そうでない場合、フローはステップ２２０に進む。

【００２３】

ステップ２１５に示したように、無効命令コード例外が発生し、適切なイベント・ハンドラが実行される。このイベント・ハンドラを、プロセッサにメッセージを表示させ、現タスクの実行を中止させ、他のタスクを実行させるようにすることができる。もちろん、代替実施形態では、このイベント・ハンドラを任意の数の方法で実行できるようにすることができる。

【００２４】

ステップ２２０で、命令が浮動小数点命令であるかどうかが判定される。命令が浮動小数点命令ではない場合、フローはステップ２２５に進む。そうでない場合、フローはステップ２３０に進む。

【００２５】

ステップ２２５に示したように、プロセッサは命令を実行する。このステップは本発明を説明するうえで必要とされないものなので、本明細書ではこれ以上は説明しない。

【００２６】

ステップ２３０に示したように、ＥＭ表示が１に等しいかどうか（前述のソフトウェア規約によれば、浮動小数点装置をエミュレートすべきかどうか）と、ＴＳ表示が１に等しいかどうか（前述のソフトウェア規約によれば、部分コンテキスト切換が実行されたかどうか）が判定される。ＥＭ表示またはＴＳ表示、あるいはその両方が１に等しい場合、フローはステップ２３５に進む。そうでない場合、フローはステップ２４０に進む。

【００２７】

ステップ２３５で、「装置使用不能」例外が発生し、対応するイベント・ハンドラが実行される。このイベントに応答して、対応するイベント・ハンドラを、ＥＭ表示およびＴＳ表示をポーリングするようにすることができる。ＥＭ表示が１に等しい場合、プロセッサに、浮動小数点装置をエミュレートすることによって命令を実行させ、かつ次の命令（ステップ２０５で受け取った命令の後に論理的に続く命令）時に実行を再開させるようにイベント・ハンドラを構成することができる。ＴＳ表示が１に等しい場合、部分コンテキ

スト切換に関連して説明したように機能し（浮動小数点装置の内容を記憶し、必要に応じて正しい浮動小数点状態を復元する）、かつプロセッサに、ステップ２０５で受け取った命令の実行を再開することによって実行を再開させるようにイベント・ハンドラを構成することができる。もちろん、代替実施形態では、このイベント・ハンドラを任意の数の方法を利用できるようにすることができる。

【００２８】

浮動小数点命令の実行時にある数値エラーが発生した場合、そのようなエラーは、次の浮動小数点命令の実行が試みられるまで未処理のままに保持され、次の浮動小数点命令の実行が試みられた時点でその実行に割り込み、未処理の浮動小数点数値エラーを処理することができる。ステップ２４０に示したように、そのような未処理のエラーがあるかどうかが判定される。そのような未処理のエラーがある場合、フローはステップ２４５に進む。そうでない場合、フローはステップ２５０に進む。

【００２９】

ステップ２４５で、未処理浮動小数点エラー・イベントが発生する。このイベントに関して、プロセッサは、浮動小数点エラーがマスクされているかどうかを判定する。そうである場合、プロセッサはマイクロコードを使用してイベントを内部で処理することを試み、浮動小数点命令が「マイクロ再開」される。マイクロ再開の語は、非マイクロコード・ハンドラ（オペレーティング・システム・イベント・ハンドラとも呼ぶ）を実行せずにイベントを処理する技法を指す。そのようなイベントは、内部でプロセッサによって処理され、したがって、外部オペレーティング・システム・ハンドラの実行を必要としないので、内部イベントと呼ばれる（ソフトウェアに見えないイベントとも呼ぶ）。これに対して、浮動小数点エラーがマスクされていない場合、このイベントは外部イベントであり（「ソフトウェアに見えるイベント」とも呼ぶ）、イベントの対応するイベント・ハンドラが実行される。このイベント・ハンドラを、エラーを処理し、かつプロセッサに、ステップ２０５で受け取った命令の実行を再開することによって実行を再開させるようにすることができる。命令を再開するこの技法を「マクロ再開」または「命令レベル再開」と呼ぶ。もちろん、代替実施形態では、この非マイクロコード・イベント・ハンドラを任意の数の方法が利用できるようにすることができる。

【００３０】

ステップ２５０に示したように、浮動小数点命令が実行される。そのような実行時には、タグが必要に応じて変更され、このとき処理できる数値エラーが報告され、他の数値エラーは未処理のままで保持される。

【発明の開示】

【発明が解決しようとする課題】

【００３１】

（Ｐｅｎｔｉｕｍプロセッサを含む）インテル・アーキテクチャ・プロセッサ・ファミリならびにその他のある汎用プロセッサの１つの制限は、パック・データを処理する１組の命令を含まないことである。したがって、既存のソフトウェアおよびハードウェアとの互換性を有するように、パック・データを処理する１組の命令をそのようなプロセッサに組み込むことが望ましい。さらに、１組のパック・データ命令をサポートし、オペレーティング・システムを含め既存のソフトウェアとの互換性を有する新しいプロセッサを製造することが望ましい。

【課題を解決するための手段】

【００３２】

本発明は、エイリアス化された単一の物理レジスタ・ファイルを使用して浮動小数点命令およびパック・データ命令を実行する方法および装置を提供する。本発明の一態様によれば、復号装置と、マッピング装置と、記憶装置とを含むプロセッサが設けられる。復号装置は、少なくとも第１の１組の命令と第２の１組の命令とを含む少なくとも１つの命令セットの命令およびそのオペランドを復号するように構成される。記憶装置は、物理レジスタ・ファイルを含む。マッピング装置は、物理レジスタ・ファイルへの第１の１組の命

令によってスタック参照的に使用されるオペランドをマップするように構成される。また、マッピング装置は、同じ物理レジスタ・ファイルへの第２の１組の命令によって非スタック参照的に使用されるオペランドをマップするように構成される。

【００３３】

本発明の他の態様によれば、一般に復号装置と、マッピング装置と、リタイヤメント装置とを含むプロセッサが設けられる。マッピング装置は、浮動小数点およびパック・データ・オペランドを、リタイヤメント装置に含まれる同じ１組のレジスタにマップする。マッピング装置は、浮動小数点オペランドをスタック参照的にマップしながら、パック・データ・オペランドを非スタック参照的にマップする。また、マッピング装置は１組のタグを含み、これらのタグはそれぞれ、マッピング・テーブル内のそれぞれの異なるエントリに対応し、その対応するエントリが空状態であるか、それとも非空状態であるかを識別する。

【発明を実施するための最良の形態】

【００３４】

下記の説明では、本発明を完全に理解していただくために多数の特定の詳細について述べる。しかし、本発明がこれらの特定の詳細なしに実施できるものであることを理解されたい。他の例では、本発明を曖昧にしないように、周知の回路、構造、技法については詳しく示していない。

【００３５】

本発明の一実施形態によれば、本出願は、様々なオペレーティング・システム技法には見えず、良好なプログラミング方法を推進し、既存のソフトウェアには見えないように、それぞれの異なるデータ・タイプ演算をプロセッサに実行させる、いくつかの異なる数組の命令を実行する方法および装置について説明するものである。これを達成するには、少なくとも論理的にはソフトウェアから単一のエイリアス化レジスタ・ファイルとして見えるものに対して、いくつかの異なるデータ・タイプ演算をプロセッサに実行させる異なる数組の命令を実行する。異なる数組の命令を実行した結果として実行されるデータ・タイプ演算はどんなタイプの演算でもよい。たとえば、ある１組の命令では、プロセッサはスカラ演算（浮動小数点または整数、あるいはその両方）を実行し、他の１組の命令ではパック演算（浮動小数点または整数、あるいはその両方）を実行する。他の例を挙げると、ある１組の命令では、プロセッサは浮動小数点演算（スカラまたはパック、あるいはその両方）を実行し、他の１組の命令では整数演算（スカラまたはパック、あるいはその両方）を実行する。他の例を挙げると、単一のエイリアス化レジスタ・ファイルをスタック参照レジスタ・ファイルおよびフラット・レジスタ・ファイルとして動作させることができる。また、本出願では、論理的にはソフトウェアから単一のエイリアス化レジスタ・ファイルとして見える別々の物理レジスタ・ファイルを使用してこれらの異なる数組の命令を実行する方法および装置について説明する。さらに、本出願では、単一の物理レジスタ・ファイルを使用してこれらの異なる数組の命令を実行する方法および装置について説明する。

【００３６】

話を明確にするために、浮動小数点命令およびパック・データ命令（浮動小数点または整数、あるいはその両方）の実行に関連して本発明を説明する。しかし、任意の数の異なるデータ・タイプ演算を実行することができ、本発明が浮動小数点演算やパック・データ演算に限らないことを理解されたい。

【００３７】

図３Ａは、本発明の一実施形態による、パック・データ状態および浮動小数点状態のエイリアス化を示す機能図である。図３Ａは、浮動小数点データ（本明細書では、浮動小数点状態と呼ぶ）を格納する１組の浮動小数点レジスタ３００と、パック・データ（本明細書では、パック・データ状態と呼ぶ）を格納する１組のパック・データ・レジスタ３１０を示す。本明細書では、パック・データ・レジスタの物理的位置を示すために表記ＰＤｎを使用する。図３Ａは、パック・データ状態が浮動小数点状態上にエイリアス化されるこ

とも示す。すなわち、浮動小数点命令およびパック・データ命令は少なくとも、ソフトウェアには同じ１組の論理レジスタ上で実行されるように見える。複数の別々の物理レジスタ・ファイルまたは単一の物理レジスタ・ファイルを使用することを含め、このエイリアス化を行ういくつかの技法がある。そのような技法の例については後で、図４ないし図１３を参照して説明する。

【００３８】

前述のように、既存のオペレーティング・システムは、プロセッサにマルチタスクの結果として浮動小数点状態を記憶させるようになっている。パック・データ状態は浮動小数点状態上にエイリアス化されるので、これらの同じオペレーティング・システムは、プロセッサに、浮動小数点状態上にエイリアス化されたパック・データ状態を記憶させる。その結果、本発明は従来のオペレーティング・システム・タスク切換ルーチンを必要とせず（もちろん、タスク切換ルーチンを１つまたは複数のイベント・ハンドラとして構成することができる）、イベント・ハンドラを修正することも、あるいは新しいオペレーティング・システム・イベント・ハンドラを書き込むことも必要としない。したがって、新しいオペレーティング・システムや修正済みオペレーティング・システムを設計しなくても、マルチタスクを行う際にパック・データ状態を記憶することができる。そのため、そのようなオペレーティング・システムを開発するために必要なコストおよび時間が必要とされない。また、一実施形態では、パック・データ命令を実行することによって生成されたイベントは、内部でプロセッサによって処理され、あるいは既存のイベントにマップされ、既存のイベントの対応するオペレーティング・システム・イベント・ハンドラによって既存のイベントを処理することができる。その結果、オペレーティング・システムに見えないようにパック・データ命令を実行することができる。

【００３９】

図３Ａは、１組の浮動小数点タグ３２０および１組のパック・データ・タグ３３０も示す。浮動小数点タグ３２０は、図１を参照して説明したタグ１５０と同様に働く。したがって、各タグは、対応する浮動小数点レジスタの内容が空であるか、それとも非空であるかを示す２つのビットを含む（たとえば、有効、特殊、または零）。パック・データ・タグ３３０は、パック・データ・レジスタ３１０に対応し、浮動小数点タグ３２０上にエイリアス化される。各タグは２つのビットを使用するようにできるが、代替実施形態では、各タグごとに１ビットのみを格納することができる。これらの１ビット・タグはそれぞれ、空と非空のどちらかを識別する。そのような実施形態では、タグ値が必要なときに適切な２ビット・タグ値を決定することによって、これらの１ビット・タグを、ソフトウェアから２ビットを備えるものとして見えるようにすることができる。最小タスク切換を実施するオペレーティング・システムは、対応するタグが非空状態を示すレジスタのみの内容を記憶する。タグがエイリアス化されるので、そのようなオペレーティング・システムは必要なパック・データ状態および浮動小数点状態を記憶する。これに対して、単純タスク切換を実施するオペレーティング・システムは、タグの状態にかかわらずに、エイリアス化論理レジスタ・ファイルの内容全体を記憶する。

【００４０】

一実施形態では、浮動小数点レジスタ３００は、図１で説明した浮動小数点レジスタ１４５と同様に動作する。したがって、図３Ａは、トップ・オブ・スタック・フィールド３５０を含む浮動小数点状態レジスタ３４０も示す。トップ・オブ・スタック・フィールド３５０は、１つの浮動小数点レジスタ３００を識別するトップ・オブ・スタック表示（ＴＯＳ）を格納するために使用される。浮動小数点レジスタ３００がスタックとして動作すると、レジスタの物理的位置にかかわらず、トップ・オブ・スタック・レジスタに対して演算が実行される。これに対して、パック・データ・レジスタ３１０は固定レジスタ・ファイル（直接アクセス・レジスタ・ファイルとも呼ぶ）として動作する。したがって、パック・データ命令は、使用するレジスタの物理的位置を指定する。パック・データ・レジスタ３１０は浮動小数点レジスタ３００の物理的位置にマップされ、このマッピングは、スタックの１番上のレジスタが変更されても変化しない。そのため、ソフトウェアからは

、少なくとも、スタック参照レジスタ・ファイルまたはフラット・レジスタ・ファイルとして動作できる単一の論理レジスタ・ファイルが存在するように見える。

【００４１】

図３Ｂおよび図３Ｃは、図３Ａに示した、エイリアス化浮動小数点レジスタ３００および浮動小数点タグ３２０のパック・データ・レジスタ３１０およびパック・データ・タグ３３０に対するマッピングを示す。前述のように、浮動小数点環境では、各レジスタｎは、ＴＯＳポインタで識別される浮動小数点レジスタに対して指定される。図３Ｂおよび図３Ｃに２つのケースが示されている。各図は、論理浮動小数点レジスタまたはプログラマに見える浮動小数点レジスタ（スタック）と論理パック・データ・レジスタまたはプログラマに見えるパック・データ・レジスタとの関係を表す。図３Ｂおよび図３Ｃに示した内円３６０は、物理浮動小数点／パック・データ・レジスタおよび対応するタグを表し、外円は、トップ・オブ・スタック・ポインタ３７０によって参照される論理浮動小数点レジスタを表す。図３Ｂに示したように、トップ・オブ・スタック・ポインタ３７０は物理浮動小数点／パック・データ・レジスタ０を指し示す。したがって、論理浮動小数点レジスタと物理浮動小数点／パック・データ・レジスタが対応する。図示したように、プッシュとポップのどちらかを行わせる浮動小数点命令によってトップ・オブ・スタック・ポインタ３７０が修正されると、それに応じてトップ・オブ・スタック・ポインタ３７０も変更される。プッシュは、図中の逆時計回りのトップ・オブ・スタック・ポインタの回転によって示されており、浮動小数点ポップ動作では、トップ・オブ・スタック・ポインタが時計回りに回転する。

【００４２】

図３Ｃに示した例では、論理浮動小数点レジスタＳＴ０と物理レジスタ０は対応しない。したがって、図示した図３Ｃの例では、トップ・オブ・スタック・ポインタ３７０は、論理浮動小数点レジスタＳＴ０に対応する物理浮動小数点／パック・データ・レジスタ２を指し示す。他のすべての論理浮動小数点レジスタはＴＯＳ３７０を基準としてアクセスされる。浮動小数点レジスタがスタックとして動作し、パック・データ・レジスタが固定レジスタ・ファイルとして動作する一実施形態について説明したが、代替実施形態では、これらの数組のレジスタを任意の方法を利用できるようにすることができる。また、浮動小数点演算およびパック・データ演算に関連して一実施形態を説明したが、この技法を使用して、レジスタ・ファイルに対して実行される演算のタイプにかかわらずに、スタック参照レジスタ・ファイル上に固定レジスタ・ファイルをエイリアス化できることを理解されたい。

【００４３】

パック・データ状態は、浮動小数点状態の任意の部分にエイリアス化することも、あるいはすべての浮動小数点状態にエイリアス化することもできる。一実施形態では、パック・データ状態は、浮動小数点状態の小数部フィールドにエイリアス化される。さらに、エイリアス化は完全なものでも、あるいは部分的なものでもよい。全エイリアス化は、レジスタの内容全体がエイリアス化される実施形態を表すために使用される。部分エイリアス化については図６Ａを参照して説明する。

【００４４】

図３Ｄは、本発明の一実施形態による、経時的な浮動小数点命令およびパック・データ命令の実行を示すブロック図である。図３Ｄは、第１の１組の浮動小数点命令３８０と、１組のパック・データ命令３８２と、第２の１組の浮動小数点命令３８４を実行順序に従って示すものである。１組のパック・データ命令３８２の実行は時間Ｔ１から開始し時間Ｔ２で終了し、それに対して、１組の浮動小数点命令の実行は時間Ｔ３から開始する。他の命令は、１組のパック・データ命令３８２が実行されてから第２の１組の浮動小数点命令３８４の実行が開始されるまでの間に実行しても、あるいは実行しなくてもよい。第１の間隔３８６は時間Ｔ１と時間Ｔ３との間の時間を示し、それに対して、第２の間隔３８８は時間Ｔ２と時間Ｔ３との間の時間を示す。

【００４５】

浮動小数点状態およびパック・データ状態がエイリアス化レジスタ・ファイルに格納されるので、タグは、第２の１組の浮動小数点命令３８４が実行される前に空に変更すべきである。そうでない場合、スタック・オーバフロー例外が発生する恐れがある。したがって、場合によっては第１の間隔３８６中にタグが空に変更される。これはいくつかの異なる方法で行うことができる。たとえば、ある実施形態では、１）１組のパック・データ命令３８２中の第１のパック・データ命令を実行させることによってタグを空状態に変更し、２）１組のパック・データ命令３８２中の各パック・データ命令を実行させることによってタグを空状態に変更し、３）実行するとエイリアス化レジスタ・ファイルが修正される第１の浮動小数点命令の実行が試みられたときにタグを空状態に変更することなどによって、これを行うことができる。これらの実施形態は、パック・データ状態がレジスタ状態の残りの部分と共に記憶され復元されるので、単純コンテキスト切換（レジスタ状態全体を各タスク切換時に記憶し復元する）をサポートする既存のオペレーティング・システムからは見えない。

【００４６】

他の実施形態では、単純コンテキスト切換または最小コンテキスト切換、あるいはその両方をサポートするオペレーティング・システムとの互換性を維持するために、１組のパック・データ命令３８２を実行すると、論理ブロック３９０で表された１組の遷移命令が時間Ｔ２から時間Ｔ３（第２の１組の浮動小数点命令３８４が開始される時間）までの間に実行されないかぎり、第１の間隔３８６でタグが非空状態に変更される。たとえば、１組のパック・データ命令３８２がタスクＡに属すると仮定する。また、タスクＡが、１組の遷移命令３９０を実行する前に全タスク切換（すなわち、部分タスク切換ではない）によって割り込まれると仮定する。全タスク切換が実行されるので、タスク切換ハンドラは、浮動小数点／パック・データ状態を記憶するための浮動小数点命令（第２の１組の浮動小数点命令３８４によって示されており、この例では「ＦＰタスク切換ルーチン」と呼ばれる）を含む。１組の遷移命令３９０が実行されていないので、プロセッサは、ＦＰタスク切換ルーチンを実行する前のある時にタグを非空状態に変更する。そのため、ＦＰタスク切換ルーチンは、最小切換ルーチンであるか、それとも単純切換ルーチンであるかにかかわらず、エイリアス化レジスタ・ファイル全体の内容（この例では、タスクＡのパック・データ状態）を記憶する。これに対して、１組の遷移命令３９０が実行された場合、プロセッサは第２の間隔３８８中のある時間にタグを空状態に変更する。したがって、１組の遷移命令３９０が実行された後にタスク切換がタスクＡに割り込むかどうかにかかわらず、プロセッサは、（第２の１組の浮動小数点命令３８４がタスク切換ハンドラに属するか、それともタスクＡに属するか、それとも他のプログラムに属するかにかかわらず）第２の１組の浮動小数点命令３８４を実行する前のある時にタグを空状態に変更する。

【００４７】

他の例として、再び、１組のパック・データ命令３８２がタスクＡに属し、タスクＡが、１組の遷移命令３９０が実行される前にタスク切換によって割り込まれると仮定する。しかし、この場合、タスク切換は部分タスク切換である（すなわち、浮動小数点／パック・データ状態は記憶されることも、あるいは復元されることもない）。浮動小数点命令またはパック・データ命令を使用する他のタスクを実行しない場合、プロセッサは最終的にタスクＡの実行に戻り、１組の遷移命令３９０が実行される。しかし、他のタスク（たとえば、タスクＢ）が浮動小数点命令またはパック・データ命令を使用する場合、このような命令の実行を試みると、オペレーティング・システム・ハンドラ呼出しによってタスクＡの浮動小数点／パック・データ状態が記憶され、タスクＢの浮動小数点／パック・データ状態が復元される。このハンドラは、浮動小数点／パック・データ状態を記憶するためのＦＰタスク切換ルーチン（この例では、第２の１組の浮動小数点命令３８４によって示されている）を含む。１組の遷移命令３９０が実行されていないので、プロセッサは、ＦＰタスク切換ルーチンを実行する前のある時にタグを非空状態に変更する。そのため、ＦＰタスク切換ルーチンは、最小切換ルーチンであるか、それとも単純切換ルーチンであるかにかかわらず、エイリアス化レジスタ・ファイル全体の内容（すなわち、タスクＡのパ

ック・データ状態）を記憶する。このように、この実施形態は、エイリアス化レジスタの状態を記憶するために使用される技法にかかわらず、オペレーティング・システムからは見えない。

【００４８】

１組の遷移命令は、任意の数の方法を利用できるようにすることができる。一実施形態では、この１組の遷移命令は、本明細書ではＥＭＭＳ（空マルチメディア状態）命令と呼ぶ新しい命令を含むことができる。この命令によって、浮動小数点／パック・データ・タグがクリアされ、実行される可能性のある後続の浮動小数点命令にすべての浮動小数点レジスタ３００を使用できることが、これに続いて実行されるコードに対して示される。これによって、普通なら、パック・データ命令の後ではなく浮動小数点命令の実行前にＥＭＭＳ命令が実行される場合に発生するスタック・オーバフロー条件が発生するのが回避される。

【００４９】

インテル・アーキテクチャ・プロセッサを使用した従来技術の浮動小数点プログラミング方法では、一般に、浮動小数点状態をクリアする動作によって浮動小数点コードのブロックを終了する。部分コンテキスト切換を使用するか、それとも最小コンテキスト切換を使用するか、それともその両方を使用するかにかかわらず、浮動小数点コードの第１のブロックの終了時に浮動小数点状態はクリア状態のままである。したがって、ＥＭＭＳ命令は、パック・データ状態をクリアするためにパック・データ・シーケンスで使用されるものである。ＥＭＭＳ命令は、パック・データ・コードのブロックの後に実行すべきである。したがって、本明細書に記載した方法および装置を備えたプロセッサは、インテル・アーキテクチャ・プロセッサを使用した従来技術の浮動小数点プロセッサとの完全な互換性を保持するが、良好なプログラミング技法および適切なハウスキーピング（パック・データ・コードと浮動小数点コードとの間の遷移の前に状態をクリアすること）を用いてプログラムした場合に、浮動小数点状態にも、あるいはパック・データ状態にも悪影響を与えずにパック・データ・コードと浮動小数点コードとの間の遷移を可能にするパック・データ命令を実行する機能も有する。

【００５０】

他の実施形態では、プロセッサにタグを実行時に空状態に変更させる既存の浮動小数点命令を使用して１組の遷移命令を形成することができる。

【００５１】

一実施形態では、パック・データ命令の実行と浮動小数点命令の実行との間の切換に時間がかかる。したがって、良好なプログラミング技法はこれらの遷移の数を最小限に抑えることである。浮動小数点命令とパック・データ命令との間の遷移の数は、パック・データ命令だけでなく浮動小数点命令もグループ化することによって削減することができる。そのような良好なプログラミング技法を推進することが望ましいので、そのような良好なプログラミング技法を無視することを困難にするようにプロセッサを構成することが望ましい。したがって、一実施形態ではまた、第１の間隔３８６中にトップ・オブ・スタック表示が初期設定状態（たとえば、レジスタＲ０を示す零）に変更される。これは、１）第１のパック・データ命令を実行させることによってトップ・オブ・スタック表示を変更し、２）１組のパック・データ命令３８２中の各パック・データ命令を実行させることによってトップ・オブ・スタック表示を変更し、３）ＥＭＭＳ命令を実行させることによってトップ・オブ・スタック表示を設定し、４）図３Ｄの時間Ｔ３に浮動小数点命令の実行が試みられたときにトップ・オブ・スタック表示を変更することなどによって行うことができる。この場合も、これは、パック・データ命令と浮動小数点命令を混合するコードに完全な互換性を維持するために行われる。また、良好なプログラミング技法を推進するために、一実施形態では、第１の間隔３８６中に、パック・データが書き込まれるエイリアス化レジスタの符号フィールドおよび指数フィールドに、数ではないことを示す値も格納される。

【００５２】

図４Ａおよび図４Ｂは、様々なオペレーティング・システム技法からは見えず、かつ効率的なプログラミング技法を推進するように浮動小数点命令およびパック・データ命令を実行する、本発明の一実施形態による方法を示す一般的な流れ図である。流れ図はステップ４００から開始する。フローは、ステップ４００からステップ４０２に進む。

【００５３】

ステップ４０２に示したように、１組のビットが命令としてアクセスされ、フローはステップ４０４に進む。この１組のビットは、この命令によって実行される動作を識別する命令コードを含む。

【００５４】

ステップ４０４で、命令コードが有効であるかどうかが判定される。命令コードが有効ではない場合、フローはステップ４０６に進む。そうでない場合、フローはステップ４０８に進む。パック・データ命令をサポートしないプロセッサに対して、パック・データ命令を含むルーチンの実行が試みられると仮定すると、パック・データ命令の命令コードは有効ではなく、フローはステップ４０６に進む。逆に、プロセッサがパック・データ命令を実行できる場合、このような命令の命令コードは有効であり、フローはステップ４０８に進む。

【００５５】

ステップ４０６に示したように、無効命令コード例外が発生し、適切なイベント・ハンドラが実行される。上記で図２のステップ２１５を参照して説明したように、このイベント・ハンドラは、プロセッサにメッセージを表示させ、現タスクの実行を中止させ、他のタスクを実行させることができる。もちろん、このイベント・ハンドラを任意の数の方法を利用できるようにすることができる。たとえば、このイベント・ハンドラは、プロセッサがパック・データ命令を実行できないかどうかを識別するようにすることができる。この同じイベント・ハンドラを、プロセッサがパック・データ命令を実行できないことを識別する表示を設定するようにすることもできる。プロセッサ上で実行される他のアプリケーションは、この表示を使用して、１組のスカラ・ルーチンを使用して実行すべきか、それとも重複された１組のパック・データ・ルーチンを使用して実行すべきかを判定する。しかし、そのように構成するには、既存のオペレーティング・システムを変更し、あるいは新しいオペレーティング・システムを開発する必要がある。

【００５６】

ステップ４０８で、どんなタイプの命令を受け取ったかが判定される。命令が浮動小数点命令でも、あるいはパック・データ命令でもない場合、フローはステップ４１０に進む。しかし、命令が浮動小数点命令である場合、フローはステップ４１２に進む。これに対して、命令がパック・データ命令である場合、フローはステップ４１４に進む。

【００５７】

ステップ４１０に示したように、プロセッサは命令を実行する。このステップは本発明を理解するうえで必要とされないので、本明細書ではこれ以上説明しない。

【００５８】

ステップ４１２に示したように、ＥＭ表示が１に等しいかどうか（前述のソフトウェア規約によれば、浮動小数点装置をエミュレートすべきかどうか）と、ＴＳ表示が１に等しいかどうか（前述のソフトウェア規約によれば、部分コンテキスト切換が実行されたかどうか）が判定される。ＥＭ表示またはＴＳ表示、あるいはその両方が１に等しい場合、フローはステップ４１６に進む。そうでない場合、フローはステップ４２０に進む。一実施形態は、ＥＭ表示またはＴＳ表示、あるいはその両方が１のときに装置使用不能例外が発生するようにするが、代替実施形態は任意の数の他の値を使用するできるようにすることができる。

【００５９】

ステップ４１６で、「装置使用不能」例外が発生し、対応するイベント・ハンドラが実行される。上記で図２のステップ２３５を参照して説明したように、対応するイベント・ハンドラを、ＥＭ表示およびＴＳ表示をポーリングするようにすることができる。ＥＭ表

示が１に等しい場合、イベント・ハンドラは浮動小数点装置をエミュレートして命令を実行し、プロセッサに次の命令（ステップ４０２で受け取った命令の後に論理的に続く命令）時に実行を再開させる。ＴＳ表示が１に等しい場合、イベント・ハンドラは、プロセッサに、部分コンテキスト切換に関連して説明したように機能し（浮動小数点装置の内容を記憶し、必要に応じて正しい浮動小数点状態を復元する）、かつプロセッサにステップ４０２で受け取った命令の実行を再開することによって実行を再開させる。もちろん、代替実施形態では、このイベント・ハンドラを任意の数の方法を利用できるようにすることができる。たとえば、ＥＭ表示を使用してマルチタスクを構成することができる。

【００６０】

パック・データ状態が浮動小数点状態上にエイリアス化され、ＥＭ表示およびＴＳ表示によって浮動小数点状態が変化するので、プロセッサはまた、パック・データ命令を実行する際にＥＭ表示およびＴＳ表示に応答し、ソフトウェアとの完全な互換性を維持しなければならない。

【００６１】

ステップ４１４で、ＥＭ表示が１に等しいと判定される。前述のように、装置使用不能例外を処理するために実行されるイベント・ハンドラを、ＥＭ表示をポーリングし、ＥＭ表示が１に等しい場合に浮動小数点装置のエミュレーションを試みるようにすることができる。既存のイベント・ハンドラはパック・データ命令をエミュレートするようには書かれていないので、ＥＭ表示が１に等しいときのパック・データ命令の実行の試みをこのイベント・ハンドラによって処理することはできない。さらに、オペレーティング・システムから見えないようにするには、このイベント・ハンドラの変更をプロセッサからは要求しないようにする必要がある。このため、ステップ４１４でＥＭ表示が１に等しいと判定された場合、フローはステップ４１６ではなくステップ４０６に進む。そうでない場合、フローはステップ４１８に進む。

【００６２】

前述のように、ステップ４０６で無効命令コード例外が発生し、対応するイベント・ハンドラが実行される。ＥＭ＝１のときのパック・データ命令の実行の試みを無効命令コード例外に切り換えることによって、この実施形態はオペレーティング・システムから見えなくなる。

【００６３】

オペレーティング・システムから見えないようにＥＭ表示を処理する一実施形態について説明したが、代替実施形態では他の技法を使用することができる。たとえば、代替実施形態は、ＥＭ表示が１に等しいときのパック・データ命令の実行の試みに応答して装置使用不能例外と、異なる既存のイベントと、新しいイベントとのうちのいずれかを発生することができる。さらに、オペレーティング・システムのわずかな修正が受け入れられる場合、この状況に応じて適切とみなされる処置をとるように、選択されたイベント・ハンドラを変更することができる。たとえば、パック・データ命令をエミュレートするようにイベント・ハンドラを書くことができる。他の代替実施形態では、パック・データ命令を実行する際にＥＭ表示を単に無視することができる。

【００６４】

ステップ４１８に示したように、ＴＳ表示が１に等しいかどうか（前述のソフトウェア規約によれば、部分コンテキスト切換が実行されたかどうか）が判定される。ＴＳ表示が１に等しい場合、フローはステップ４１６に進む。そうでない場合、フローはステップ４２２に進む。

【００６５】

前述のように、ステップ４１６で装置使用不能例外が発生し、対応するイベント・ハンドラが実行される。したがって、このイベントに応答して、対応するイベント・ハンドラを、ＥＭ表示およびＴＳ表示をポーリングするようにすることができる。ステップ４１４で、ＥＭ表示が１に等しい状況が無効命令コード例外に切り換えられたので、ＥＭ表示は０に等しくなければならず、ＴＳ表示は１に等しくなければならない。ＴＳ表示が１に等

しいので、イベント・ハンドラは、プロセッサに、部分コンテキスト切換に関連して説明したように機能し（浮動小数点装置の内容を記憶し、必要に応じて正しい浮動小数点状態を復元する）、かつプロセッサにステップ４０２で受け取った命令の実行を再開することによって実行を再開させる。パック・データ状態が浮動小数点状態上にエイリアス化されるので、このイベント・ハンドラは浮動小数点状態とパック・データ状態の両方のために働く。このため、この方法はオペレーティング・システムからは見えない。もちろん、代替実施形態ではこのイベント・ハンドラを任意の数の方法を利用できるようにすることができる。たとえば、パック・データ状態が浮動小数点状態上にエイリアス化されない代替実施形態では、浮動小数点状態とパック・データ状態の両方を記憶する新しいイベント・ハンドラを使用することができる。

【００６６】

オペレーティング・システムから見えないようにＴＳ表示を処理する一実施形態について説明したが、代替実施形態では他の技法を使用することができる。たとえば、代替実施形態ではＴＳ表示を実施しなくてもよい。そのような代替実施形態は、ＴＳ表示を使用して部分コンテキスト切換を実施するオペレーティング・システムとの互換性を有さない。しかし、そのような代替実施形態は、ＴＳ表示を使用した部分コンテキスト切換をサポートしない既存のオペレーティング・システムとの互換性を有する。他の例を挙げると、ＴＳ表示が１に等しいときのパック・データ命令の実行の試みを、新しいイベント・ハンドラまたは修正された既存のイベント・ハンドラに切り換えることができる。このイベント・ハンドラは、この状況に応じて適切とみなされる処置をとるようにすることができる。たとえば、パック・データ状態が浮動小数点状態上にエイリアス化されない実施形態では、このイベント・ハンドラはパック・データ状態または浮動小数点状態、あるいはその両方を記憶することができる。

【００６７】

上記で図２を参照して説明したように、浮動小数点命令の実行時にある数値エラーが発生した場合、このようなエラーは、次の浮動小数点命令の実行が試みられるまで未処理のままに保持され、次の浮動小数点命令の実行が試みられた時点でその実行に割込み、エラーを処理することができる。ステップ４２０とステップ４２２の両方に示したように、次に処理できるそのような未処理のエラーがあるかどうかが判定される。したがって、これらのステップは図２のステップ２４０に類似している。そのような未処理のエラーがある場合、フローはステップ４２０とステップ４２２の両方からステップ４２４へ移る。しかし、ステップ４２０でそのような未処理のエラーがないと判定された場合、フローはステップ４２６に進む。これに対して、ステップ４２２でそのような未処理のエラーがないと判定された場合、フローはステップ４３０に進む。代替実施形態では、パック・データ命令が実行される間そのようなエラーは未処理のままにされる。

【００６８】

ステップ４２４で、未処理浮動小数点エラー例外が発生する。上記で図２のステップ２４５を参照して説明したように、プロセッサはこのイベントに応答して、浮動小数点エラーがマスクされているかどうかを判定する。そうである場合、プロセッサはこのイベントを内部で処理することを試み、浮動小数点命令がマイクロ再開される。浮動小数点エラーがマスクされていない場合、このイベントは外部イベントであり、対応するイベント・ハンドラが実行される。このイベント・ハンドラは、エラーを処理し、かつプロセッサに、ステップ４０２で受け取った命令の実行を再開することによって実行を再開させるようにすることができる。もちろん、代替実施形態では、このイベント・ハンドラを任意の数の方法を利用するようにすることができる。

【００６９】

ステップ４２６に示したように、浮動小数点命令が実行される。オペレーティング・システムから見えないように、一実施形態はまた、タグを必要に応じて変更し、次に処理できる数値エラーを報告し、他の数値エラーを未処理のままにしておく。浮動小数点装置の内容を記憶するための多数のオペレーティング・システム技法があるが、すべてのそのよ

うなオペレーティング・システム技法から見えないようにパック・データ命令および浮動小数点命令を実行することが望ましい。タグを維持することによってこの実施形態は、対応するタグが非空状態を示す浮動小数点レジスタのみの内容を記憶するそのようなオペレーティング・システム技法からは見えないオペレーティング・システムのままとなる。しかし、代替実施形態は、これらのオペレーティング・システム技法のうちのいくつかの技法との互換性を有するようにすることができる。たとえば、既存のオペレーティング・システムがタグを使用しない場合、タグを実装していないプロセッサは依然として、そのオペレーティング・システムとの互換性を有する。さらに、本発明では、数値浮動小数点例外を未処理のままにしておく必要はなく、したがって、数値浮動小数点例外を未処理のままにしておかない代替実施形態も本発明の範囲内である。

【００７０】

ステップ４３０に示したように、パック・データ命令がＥＭＭＳ命令（遷移命令とも呼ぶ）であるかどうかが判定される。パック・データ命令がＥＭＭＳ命令である場合、フローはステップ４３２に進む。そうでない場合、フローはステップ４３４に進む。ＥＭＭＳ命令を使用して、浮動小数点タグが初期設定状態に変更される。したがって、パック・データ状態が浮動小数点状態上にエイリアス化されている場合、この命令は、パック・データ命令の実行から浮動小数点命令の実行へ遷移する際に実行すべきである。このように、浮動小数点装置は、浮動小数点命令を実行できるように初期設定される。パック・データ状態を浮動小数点状態上にエイリアス化しない代替実施形態は、ステップ４３０およびステップ４３２を実行する必要はない。ＥＭＭＳ命令をエミュレートする場合も、ステップ４３０およびステップ４３２は必要とされない。

【００７１】

ステップ４３２に示したように、すべてのタグが空状態に変更され、トップ・オブ・スタック表示が初期設定値に変更される。タグを空状態に変更することによって、浮動小数点装置は初期設定され、浮動小数点命令を実行する準備が完了している。トップ・オブ・スタック表示を初期設定値（一実施形態ではレジスタＲ０を識別する零）に変更すると、浮動小数点命令とパック・データ命令が別々にグループ化され、したがって、良好なプログラミング技法が推進される。代替実施形態では、トップ・オブ・スタック表示を初期設定する必要はない。ステップ４３２が完了すると、システムは次の命令（ステップ４０２で受け取った命令の後に論理的に続く命令）を実行することができる。

【００７２】

ステップ４３４に示したように、（数値例外を発生せずに）パック・データ命令が実行され、トップ・オブ・スタック表示が初期設定値に変更される。数値例外が発生するのを回避するために、一実施形態は、データ値が最大値または最小値で飽和しあるいは固定されるようにパック・データ命令を構成する。数値例外が発生しないことにより、イベント・ハンドラは例外を処理する必要がない。このため、本発明のこの実施形態はオペレーティング・システムからは見えない。別法として、そのような数値例外に応答してマイクロコード・イベント・ハンドラを実行するように実施形態を構成させることができる。オペレーティング・システムに追加イベント・ハンドラが組み込まれ、あるいは既存のイベント・ハンドラがエラーを処理するように変更されるように、オペレーティング・システムから完全に見えないわけではない代替実施形態を構成することができる。トップ・オブ・スタックは上記で述べたのと同じ理由で変更される。代替実施形態は、トップ・オブ・スタックを任意のそれぞれの異なる回数だけ変更するようにすることができる。たとえば、ＥＭＭＳを除くすべてのパック・データ命令の実行時にトップ・オブ・スタック表示を変更するように代替実施形態を形成することができる。他の代替実施形態は、ＥＭＭＳを除くパック・データ命令の実行時にはトップ・オブ・スタック表示を変更しないようにすることができる。パック・データ命令の実行を試みた結果としてメモリ・イベントが発生した場合、実行が割り込まれ、トップ・オブ・スタック表示は変更されず、このイベントが処理される。イベントの処理が完了すると、ステップ４０２で受け取った命令が再開される。フローはステップ４３４からステップ４３６へ移る。

【００７３】
ステップ４３６に示したように、パック・データ命令によってプロセッサがエイリアス化レジスタに書込みを行うかどうかが判定される。そうである場合、フローはステップ４３８へ進む。そうでない場合、フローはステップ４４０に進む。
【００７４】
ステップ４３８で、パック・データ命令によってプロセッサが書込みを行う各エイリアス化レジスタの符号フィールドおよび指数フィールドに１が格納される。フローはステップ４３８からステップ４４０へ移る。このステップを実行すると、浮動小数点命令とパック・データ命令が別々にグループ化されるため、良好なプログラミング技法が推進される。もちろん、この問題に関連のない代替実施形態は、このステップを実施するのを回避することができる。一実施形態では、符号フィールドおよび指数フィールドに１が書き込まれるが、代替実施形態では、ＮＡＮ（非数値）または無限を表す任意の値を使用することができる。
【００７５】
ステップ４４０に示したように、すべてのタグが非空状態に変更される。すべてのタグを非空状態に変更すると、浮動小数点命令とパック・データ命令が別々にグループ化されるため、良好なプログラミング技法が推進される。また、オペレーティング・システム互換性のために、ある種のオペレーティング・システム技法では、対応するタグが非空状態（最小コンテキスト切換）を示す浮動小数点レジスタのみの内容が記憶される。したがって、パック・データ状態が浮動小数点状態上にエイリアス化される実施形態では、すべてのタグを非空状態に等しい値に変更することによって、そのようなオペレーティング・システムが浮動小数点状態の場合と同様にパック・データ状態を保存する。代替実施形態では、対応するレジスタが有効なパック・データ項目を含むタグのみを変更することができる。さらに、代替実施形態は、このようなオペレーティング・システム技法のいくつかとの互換性を有するようにすることができる。たとえば、ある既存のオペレーティング・システムがタグを使用しない（たとえば、レジスタ状態全体を記憶し復元するオペレーティング・システム）場合、タグを実装しない実施形態はそのオペレーティング・システムとの互換性を有する。ステップ４４０が完了すると、システムは次の命令（ステップ４０２で受け取った命令の後に論理的に続く命令）を実行することができる。
【００７６】
したがって、この実施形態では、浮動小数点状態保存（ＦＳＡＶＥ）命令または浮動小数点環境記憶（ＦＳＴＥＮＶ）命令の後のメモリ内のタグの内容を以下に、表１を参照して示す。
【００７７】
【表１】

表１　パック・データ／ＦＰ命令のタグ・ワードに対する影響

命令タイプ	命令	タグ・ビット	ＦＳＡＶＥ／ＦＳＴＥＮＶ後のメモリ内の計算されたタグ・ワード
パック・データ	任意（ＥＭＭＳを除く）	非空（００、０１、または１０）	非空（００、０１、または１０）
パック・データ	ＥＭＭＳ	空（１１）	空（１１）
浮動小数点	任意	００、１１	００、１１、０１、または１０
浮動小数点	ＦＲＳＴＯＲ、ＦＬＤＥＮＶ	００、１１、０１、または１０	００、１１、０１、または１０

【００７８】
図のように、ＥＭＭＳを除くパック・データ命令では、タグ３２０が非空状態（００）に設定される。ＥＭＭＳによって浮動小数点タグ・レジスタは空（１１）に設定される。また、ＥＭＭＳを含むパック・データ命令では、トップ・オブ・スタック・フィールド３５０に記憶されているトップ・オブ・スタック表示も０にリセットされる。
【００７９】
インテル・アーキテクチャ・プロセッサ内の制御ワードや状況ワード（ＴＯＳを除く）など残りの環境レジスタは変更されない。パック・データ読取りまたはＥＭＭＳでは、浮動小数点レジスタ３００の小数部および指数部は変更されない。しかし、一実施形態では、エイリアス化機構のためのパック・データ・レジスタへのパック・データ書込みでは、対応する浮動小数点レジスタの小数部が、実行中の演算に応じて修正される。さらに、この実施形態では、パック・データ・レジスタ３１０を修正することによって浮動小数点レジスタの小数部にデータが書き込まれると、浮動小数点レジスタ３００の符号部および指数部中のすべてのビットが１に設定される。パック・データ命令は浮動小数点レジスタの符号部および指数部を使用しないので（浮動小数点レジスタの符号部および指数部においてパック・データ・レジスタはエイリアス化されない）、これはパック・データ命令には影響を与えない。前述のように、代替実施形態では、浮動小数点状態の任意の部分上にパック・データ状態をエイリアス化することができる。また、代替実施形態では、他の値を書き込むこともでき、また、レジスタの符号部または指数部、あるいはその両方を変更しなくてもよい。
【００８０】
【表２】

表２　パック・データ命令のＦＰＵに対する影響

命令タイプ	タグ・ワード	ＴＯＳ（ＳＷ１３．１１）	ＦＰＵの他環境データ（ＣＷポインタ、ポインタ、そのＳＷフィールド）	パック・データ・レジスタの指数部＋トップビット（パック・データ）	パック・データ・レジスタの小数部（パック・データ）
パック・データ・レジスタからのパック・データ読取り	すべてのフィールドが００に設定される（非空）。	０	変更なし	変更なし	変更なし
パック・データ・レジスタへのパック・データ書込み	すべてのフィールドが００に設定される（非空）。	０	変更なし	１に設定	影響を受ける
ＥＭＭＳ	すべてのフィールドが１１（空）に設定される。	０	変更なし	変更なし	変更なし

【００８１】
パック・データ命令の実行をさらに示すために、書き込まれる浮動小数点レジスタの符号部および指数部がすべて１に設定される。これが行われるのは、浮動小数点レジスタが浮動小数点レジスタの指数部を使用し、パック・データ命令を実行した後にレジスタのこの部分を決定状態のままにしておくことが望ましいからである。インテル・アーキテクチャ・マイクロプロセッサでは、浮動小数点レジスタの指数部がすべて１に設定されている

場合、非数値（ＮＡＮ）として解釈される。したがって、パック・データ・タグ３３０が非空状態に設定されるだけでなく、浮動小数点レジスタの指数部が、すでにパック・データ命令が実行されたことを示すために使用できるすべて１に設定される。このため、データを修正し不適切な結果をもたらすパック・データ命令および浮動小数点命令によるデータの混合がさらに防止される。したがって、浮動小数点コードは、浮動小数点レジスタが浮動小数点データを含むときと、パック・データを含むときとを区別する追加方法を有する。

【００８２】

したがって、既存のオペレーティング・システム（ワシントン州レッドモンドのマイクロソフト社（Ｍｉｃｒｏｓｏｆｔ　Ｃｏｒｐｏｒａｔｉｏｎ）から市販されているＭＳ　Ｗｉｎｄｏｗｓ（登録商標）ブランド・オペレーティング環境など）との互換性を有し、良好なプログラミング技法を推進するパック・データ命令を実行する方法について説明する。パック・データ状態が浮動小数点状態上にエイリアス化されるので、パック・データ状態は、浮動小数点状態の場合と同様に、既存のオペレーティング・システムによって保存され復元される。さらに、パック・データ命令を実行することによって生成されるイベントを既存のオペレーティング・システム・イベント・ハンドラによって処理できるので、このようなイベント・ハンドラを修正する必要がなく、新しいイベント・ハンドラを追加する必要もない。このため、プロセッサは後方互換性を有し、更新時には、オペレーティング・システムを開発または修正するために必要なコストおよび時間が必要とされない。

【００８３】

やはり既存のオペレーティング・システムとの互換性を有するこの方法のいくつかの異なる実施形態について、図７Ａないし図７Ｃ、図８、図９、図１１Ａないし図１１Ｃを参照して説明する。これらの実施形態はそれぞれ異なるが、以下のことはすべてのこれらの実施形態（図４Ａないし図４Ｂに示した実施形態、図７Ａないし図７Ｃ、図８、図９に示した実施形態、図１１Ａないし図１１Ｃに示した実施形態）に共通する。１）ソフトウェアからは少なくとも、浮動小数点状態およびパック・データ状態が単一の論理レジスタ・ファイルに格納されているように見える。２）ＥＭビットが「浮動小数点命令をエミュレートすべきである」ことを示すときにパック・データ命令を実行すると、装置使用不能例外ではなく無効命令コード例外が生じる。３）ＴＳビットが「部分コンテキスト切換が実行された」ことを示すときにパック・データ命令を実行すると装置使用不能例外が生じる。４）パック・データ命令の実行を試みることによって未処理の浮動小数点イベントが処理される。５）パック・データ命令を実行すると、次の浮動小数点命令が実行される前のある時点でトップ・オブ・スタック表示が０に変更される。６）ＥＭＭＳ命令が実行された後に他のパック・データ命令が実行されない場合、ＥＭＭＳ命令を実行すると、次の浮動小数点命令が実行される前のある時点ですべてのタグが空状態に変更される。７）パック・データ命令が実行された後にＥＭＭＳ命令が実行されない場合、次の浮動小数点命令が実行される前のある時点ですべてのタグが非空状態に変更される。８）パック・データ命令の実行に応答してプロセッサによって書き込まれるＦＰ／ＰＤレジスタの符号フィールドおよび指数フィールドに、ＮＡＮ（非数値）または無限を表すある値が記憶される。９）新しい非マイクロコード・イベント・ハンドラは必要とされない。

【００８４】

図４Ａないし図４Ｂに示した実施形態の変形形態のうちのいくつかについて説明したが、これらの実施形態は、そのようなオペレーティング・システムとの全互換性または部分互換性を有し、かつ／あるいは良好なプログラミング技法を推進することができる。たとえば、本発明の代替実施形態は、図４Ａないし図４Ｂに示した流れ図内のあるステップを異なる位置へ移動することができる。本発明の他の実施形態は１つまたは複数のステップを変更または削除することができる。たとえば、代替実施形態はＥＭビットをサポートしないこともある。もちろん、本発明は任意の数のシステム・アーキテクチャに有用であり、本明細書に記載したアーキテクチャに限らない。

【００８５】
本発明の実施形態を使用するプログラマが、浮動小数点命令およびパック・データ命令を実行する上記の方法を使用して、コードを、図３Ｄに示したように浮動小数点命令およびパック・データ命令の別々のブロックを備えるセクションに区画することを推奨する。これによって、浮動小数点演算のシーケンスからパック・データ演算のシーケンスへの遷移の前にパック・データの状態を保存しクリアすることができ、その逆の場合も同様である。この場合も、タスク切換時にコンテキストを保存する機構を含め従来技術のタスク切換機構との互換性が許容される。

【００８６】
パック・データ命令は浮動小数点レジスタ３００（図３Ａ）に影響を与え、単一のパック・データ命令がすべての浮動小数点タグを非空状態に設定するので、適切なブックキーピングを行うにはコードをコード・タイプのブロックとして区画することを推奨する。ブロック中の混合された浮動小数点命令およびパック・データ命令の実行の例を図３Ｄに示す。これには、コオペラティブ・マルチタスク・オペレーティング・システム内の動作、または単一のアプリケーション内の浮動小数点命令パック命令混合アプリケーション・コードを含めることができる。いずれの場合も、浮動小数点レジスタ３００、対応するタグ、トップ・オブ・スタック表示の適切なブックキーピングは、機能を別々の浮動小数点パック・データ・コード・ブロックとして区画することによって保証される。

【００８７】
たとえば、図３Ｄに示したように、実行ストリームには第１の１組の浮動小数点命令３８０を含めることができる。浮動小数点命令３８０のブロックが終了した後、アプリケーションの必要に応じて浮動小数点状態を保存することができる。これは、浮動小数点スタックをポップすることと、インテル・アーキテクチャ・プロセッサでＦＳＡＶＥ／ＦＮＳＡＶＥ命令を使用することを含め、任意の数の既知の従来技術の技法を使用して実行することができる。これは、浮動小数点環境を保存し、対応する浮動小数点レジスタが有効なデータを含むことの表示があるかどうかに関して個別のタグを検査する最小コンテキスト切換時に行うこともできる。対応する浮動小数点データが有効なデータを含むことを示す各タグごとに、対応する浮動小数点レジスタが保存される。また、この場合、浮動小数点レジスタの数の表示も保存する必要がある。

【００８８】
第１の１組の浮動小数点命令３８０が実行された後に、実行ストリーム内の第２の１組のパック・データ命令３８２が実行される。１組の遷移命令３９０が実行されない場合、各パック・データ命令を実行するたびに、間隔３８６中のある時点ですべてのパック・データ・タグ３３０が非空状態に設定されることを想起されたい。

【００８９】
１組のパック・データ命令３８２が実行された後にタスク切換が行われない場合は、１組の遷移命令３９０が実行される。この１組の遷移命令３９０をパック・データ状態を保存するようにすることができる。これは、前述の従来技術の浮動小数点保存命令、またはパック・データ状態のみを保存する専用命令を含む機構を使用して実行することができる。パック・データ状態は、部分コンテキスト切換機構や最小コンテキスト切換機構を含め、任意の従来技術の方法で保存することができる。パック・データ状態を保存するかどうかにかかわらず、１組の遷移命令３９０によってパック・データ状態が空になる。この場合、パック・データ状態はパック・データ・タグ３３０および対応するエイリアス化浮動小数点タグ３２０に影響を与える。前述のように、パック・データ状態を空にすることは、単一の命令ＥＭＭＳを実行し、あるいは以下で図１４を参照して論じるように一連の浮動小数点演算を実行することによって行われる。このため、プロセッサは、間隔３８８中のある時点でパック・データ状態を空にし、浮動小数点命令を実行できるように初期設定される。

【００９０】
１組の遷移命令３９０が実行された後に、第２の１組の浮動小数点命令３８４が実行さ

れる。第２の間隔３８８中にタグが空にされトップ・オブ・スタック表示が第１の物理レジスタ０を指し示すように変更されたので、すべての浮動小数点レジスタを使用することができる。このため、普通なら浮動小数点命令の実行時に起こる浮動小数点スタック・オーバフロー例外の発生が防止される。いくつかのソフトウェア実施態様では、スタック・オーバフロー条件によって、割込みハンドラはパック・データ状態を保存し空にすることができる。したがって、本発明の実施済み実施形態では、パック・データ命令浮動小数点命令混合ブロックが許容される。しかし、遷移時にタスクの状態が破壊されないように、パック・データ命令と浮動小数点命令との間の遷移時に所望の浮動小数点状態またはパック・データ状態を保存するためにアプリケーション・プログラマまたはコオペラティブ・マルチタスク・コードによって適切なブックキーピングを行わなければならない。また、この方法では、本発明の実施済み実施形態を使用して、普通なら、推奨されないプログラミング技法を使用した場合に起こる、不要な例外が回避される。

【００９１】

ＥＭＭＳ命令では、パック・データ命令ストリームと浮動小数点命令ストリームとの間の遷移を円滑に行うことができる。前述のように、ＥＭＭＳ命令は、浮動小数点タグをクリアして、発生する恐れのある浮動小数点オーバフロー条件を回避し、さらにトップ・オブ・スタック・フィールド３５０に記憶されているトップ・オブ・スタック表示をリセットする。これらの動作を実行する専用命令を実行することができるが、既存の浮動小数点命令の組合せを使用してそのような動作を行うことも企図され、かつこの開示の範囲内である。この例を図１４に示す。さらに、パック・データ命令を実行した後の第１の浮動小数点命令の実行にこの機能を組み込むことができる。この実施形態では、パック・データ命令を実行した後に（浮動小数点／パック・データ状態の環境を記憶する命令以外の）第１の浮動小数点命令を実行すると、プロセッサは暗黙的なＥＭＭＳ動作（すべてのタグを空状態に設定する）を実行する。

【００９２】

図５は、本発明の一実施形態による例示的なコンピュータ・システム５００を示すブロック図である。例示的なコンピュータ・システム５００はプロセッサ５０５と、記憶装置５１０と、バス５１５とを含む。プロセッサ５０５はバス５１５によって記憶装置５１０に結合される。また、キーボード５２０やディスプレイ５２５などいくつかのユーザ入出力装置もバス５１５に結合される。ネットワーク５３０をバス５１５に結合することもできる。プロセッサ５０５はＣＩＳＣ、ＲＩＳＣ、ＶＬＩＷ、混成アーキテクチャなど任意のタイプのアーキテクチャの中央演算処理装置を表す。また、プロセッサ５０５は１つまたは複数のチップに実装することができる。記憶装置５１０はデータを記憶する１つまたは複数の機構を表す。たとえば、記憶装置５１０には、読取り専用メモリ（ＲＯＭ）や、ランダム・アクセス・メモリ（ＲＡＭ）や、磁気ディスク記憶媒体や、光学記憶媒体や、フラッシュ・メモリ装置や、その他の機械可読媒体を含めることができる。バス５１５は１つまたは複数のバス（たとえば、ＰＣＩ、ＩＳＡ、Ｘ－Ｂｕｓ、ＥＩＳＡ、ＶＥＳＡなど）およびブリッジ（バス・コントローラとも呼ぶ）を表す。この実施形態は単一のプロセッサ・コンピュータ・システムに関して説明しているが、本発明はマルチプロセッサ・コンピュータ・システムで実施することができる。また、この実施形態は、３２ビット・コンピュータ・システムおよび６４ビット・コンピュータ・システムに関して説明しているが、本発明の実施態様はそのようなコンピュータ・システムに限らない。

【００９３】

図５は、プロセッサ５０５がバス装置５４５と、キャッシュ５５０と、命令セット装置５６０とメモリ管理装置５６５と、イベント処理装置５７０とを含むことも示している。もちろん、プロセッサ５０５は、本発明の実施態様を理解するうえでは必要とされない追加回路を含む。

【００９４】

バス装置５４５はキャッシュ５５０に結合される。バス装置５４５は、プロセッサ５０５の外部で生成された信号を監視し評価すると共に、プロセッサ５０５内の他の装置およ

び機構からの入力信号および内容要求に応答して出力信号を調整するために使用される。

【００９５】

キャッシュ５５０は、プロセッサ５０５によって命令キャッシュおよびデータ・キャッシュとして使用される１つまたは複数の記憶域を表す。たとえば、一実施形態では、キャッシュ５５０は、一方が命令用で一方がデータ用の２つの別々のキャッシュとして実施される。キャッシュ５５０は、命令セット装置５６０およびメモリ管理装置５６５に結合される。

【００９６】

命令セット装置５６０は、少なくとも１つの命令セットを復号し実行するハードウェアまたはファームウェア、あるいはその両方を含む。図５に示したように、命令セット装置５６０は復号／実行装置５７５を含む。復号装置は、プロセッサ５０５が受け取った命令を、制御信号またはマイクロコード・エントリ・ポイント、あるいはその両方に復号するために使用される。このような制御信号またはマイクロコード・エントリ・ポイント、あるいはその両方に応答して、実行装置は適切な動作を実行する。復号装置は、任意の数の異なる機構（たとえば、参照テーブルや、ハードウェア実施態様や、ＰＬＡなど）を使用して構成することができる。復号装置および実行装置による様々な命令の実行は本明細書では一連のｉｆ／ｔｈｅｎ文で表されるが、これらのｉｆ／ｔｈｅｎ文の一連の処理なしでも命令が実行できることを理解されたい。むしろ、このｉｆ／ｔｈｅｎ処理を論理的に実行する機構は本発明の実施態様の範囲内であるとみなされる。

【００９７】

復号／実行装置５７５は、パック・データ命令を含む命令セット５８０を含む装置として示されている。このようなパック・データ命令は、任意の数の演算を実行するようにすることができる。たとえば、このようなパック・データ命令を実行すれば、プロセッサにパック浮動小数点演算またはパック整数演算、あるいはその両方を実行させることができる。一実施形態では、このようなパック・データ命令は、１９９５年８月３１日に出願された出願番号０８／５２１３６０号の「Ａ Ｓｅｔ ｏｆ Ｉｎｓｔｒｕｃｔｉｏｎｓ ｆｏｒ Ｏｐｅｒａｔｉｎｇ ｏｎ Ｐａｃｋｅｄ Ｄａｔａ」に記載された命令である。命令セット５８０には、パック・データ命令だけでなく、新しい命令や、既存の汎用プロセッサで使用される命令と同様な命令、またはそのような命令と同じ命令を含めることができる。たとえば、一実施形態では、プロセッサ５０５は、Ｐｅｎｔｉｕｍプロセッサなど既存のプロセッサで使用されるインテル・プロセッサ・アーキテクチャ命令セットとの互換性を有する命令セットをサポートする。

【００９８】

図５は、メモリ装置５８５を含む命令セット装置５６０も示す。メモリ装置５８５は、浮動小数点データ、パック・データ、整数データ、制御データ（たとえば、ＥＭ表示、ＴＳ表示、トップ・オブ・スタック表示など）を含め情報を記憶するプロセッサ５０５上の１組または複数組のレジスタを表す。そのうちのいくつかを本明細書で詳しく説明するある種の実施形態では、メモリ装置５８５は浮動小数点状態上にパック・データ状態をエイリアス化する。

【００９９】

メモリ管理装置５６５は、ページングやセグメント化など１つまたは複数のメモリ管理技法を実装するハードウェアおよびファームウェアを表す。任意の数のメモリ管理技法を使用することができるが、一実施形態では、インテル・プロセッサ・アーキテクチャとの互換性を有するメモリ管理技法が実装される。イベント処理装置５７０はメモリ管理装置５６５および命令セット装置５６０に結合される。イベント処理装置５７０は、１つまたは複数のイベント処理技法を実装するハードウェアおよびファームウェアを表す。任意の数のイベント処理技法を使用することができるが、一実施形態では、インテル・プロセッサ・アーキテクチャとの互換性を有するイベント処理技法が実装される。

【０１００】

図５は、コンピュータ・システム５００によって実行されるオペレーティング・システ

ム５３５およびパック・データ・ルーチン５４０が記憶されている記憶装置５１０も示す。パック・データ・ルーチン５４０は、１つまたは複数のパック・データ命令を含む命令のシーケンスである。もちろん、記憶装置５１０は好ましくは、本発明を理解するうえでは必要とされない追加ソフトウェア（図示せず）を含む。

【０１０１】

一実施形態では、プロセッサ５０５上のレジスタ内のビットを使用して様々な表示（たとえば、ＥＭ表示、ＴＳ表示など）が実施されるが、代替実施形態では任意の数の技法を使用することができる。たとえば、代替実施形態では、これらの表示をオフチップで（たとえば、記憶装置５１０に）記憶し、かつ／あるいは各表示ごとに複数のビットを使用することができる。記憶域の語は本明細書では、記憶装置５１０内の位置、プロセッサ５０５内の１つまたは複数のレジスタなどを含め、データを記憶する機構を指すために使用される。

【０１０２】

図６Ａは、２つの別々の物理レジスタ・ファイルを使用して浮動小数点状態上にパック・データ・レジスタ状態をエイリアス化する、本発明の一実施形態による装置を示すブロック図である。この２つの物理レジスタ・ファイルは、エイリアス化されるので、論理的にはソフトウェアからはプロセッサ上で単一の論理レジスタ・ファイルとして実行されているように見える。図６Ａは、遷移装置６００と、浮動小数点装置６０５と、パック・データ装置６１０を示す。浮動小数点装置６０５は図１の浮動小数点装置１３５に類似している。浮動小数点装置６０５は、１組の浮動小数点レジスタ６１５と、１組のタグ６２０と、浮動小数点状態レジスタ６２５と、浮動小数点スタック参照装置６３０とを含む。一実施形態では、浮動小数点装置６０５は８つのレジスタ（Ｒ０ないし Ｒ７と呼ぶ）を含む。この８つのレジスタはそれぞれ、８０ビット幅であり、符号フィールドと、指数フィールドと、小数部フィールドとを含む。浮動小数点スタック参照装置６３０は１組の浮動小数点レジスタ６１５をスタックとして操作する。浮動小数点状態レジスタ６２５は、トップ・オブ・スタック表示を記憶するトップ・オブ・スタック・フィールド６３５を含む。前述のように、トップ・オブ・スタック表示は、現在、１組の浮動小数点レジスタ６１５内のどのレジスタが浮動小数点スタックの１番上のレジスタであるかを識別する。図６Ａでは、トップ・オブ・スタック表示は、物理的位置Ｒ４にあるレジスタ６４０をＳＴ（０）、すなわちスタックの１番上のレジスタとして識別する。

【０１０３】

一実施形態では、１組のタグ６２０は、８つのタグを含み、単一のレジスタに記憶される。各タグはそれぞれの異なる浮動小数点レジスタに対応し、２つのビットを備える。別法として、各タグを、エイリアス化の結果として得られる論理レジスタ・ファイル内のそれぞれの異なるレジスタに対応するものとみることができる。図６Ａに示したように、タグ６４５はレジスタ６４０に対応する。前述のように、これらのタグは、空レジスタ位置と非空レジスタ位置を区別するために浮動小数点装置６０５によって使用される。前述のように、実施形態では、空状態と非空状態のどちらかを識別する１ビット・タグを使用することができるが、ソフトウェアからは、タグ値が必要なときに適切な２ビット・タグ値を判定することによって、このような１ビット・タグが２つのビットを備えるように見える。もちろん、代替実施形態では２ビット・タグとすることができる。いずれの場合も、タグを、１１によって示される空状態と００、０１、１０のうちの１つによって示される非空状態の２つの状態を識別するものとみることができる。

【０１０４】

パック・データ装置６１０は、パック・データを記憶するために使用され、１組のパック・データ・レジスタ（パック・データ・レジスタ・ファイルとも呼ぶ）６５０と、パック・データ状態レジスタ６５５と、パック・データ非スタック参照装置６６０とを含む。一実施形態では、１組のパック・データ・レジスタ６５０は８つのレジスタを含む。この８つのレジスタはそれぞれ、１組の浮動小数点レジスタ６１５内の異なるレジスタに対応する。８つのパック・データ・レジスタはそれぞれ、６４ビット幅であり、それが対応す

る浮動小数点レジスタの６４ビット小数部フィールド上にマップされる。パック・データ非スタック参照装置６６０は、パック・データ・レジスタ６５０を固定レジスタ・ファイルとして操作する。したがって、パック・データ命令は、１組のパック・データ・レジスタ６５０内のどのレジスタを使用すべきかを明示的に指定する。

【０１０５】

遷移装置６００は、パック・データ・レジスタ６５０と浮動小数点レジスタ６１５の２つの物理レジスタ・ファイル間でデータをコピーすることによってパック・データ・レジスタ６５０を浮動小数点レジスタ６１５上にエイリアス化する。したがって、遷移装置６００は、物理浮動小数点レジスタ６１５および物理パック・データ・レジスタ６５０がユーザ／プログラマから論理的に単一の論理レジスタ・ファイルとして見えるようにする。このように、ソフトウェアからは、浮動小数点命令およびパック・データ命令を実行する場合に単一の論理レジスタ・ファイルしか使用できないように見える。遷移装置６００は、ハードウェアやマイクロコードを含め任意の数の技法を使用して構成することができる。もちろん、代替実施形態では、遷移装置６００をプロセッサ上の任意の場所に配置することができる。さらに、代替実施形態では、遷移装置６００は、プロセッサの外部に記憶された非マイクロコード・イベント・ハンドラでよい。

【０１０６】

遷移装置６００は、全エイリアス化または部分エイリアス化を行うようにすることができる。パック・データ・モードへの遷移時にすべての物理浮動小数点レジスタの内容をパック・データ・レジスタ・ファイルにコピーする場合、物理浮動小数点レジスタ・ファイルは完全にパック・データ・レジスタ・ファイル上にエイリアス化される。同様に、浮動小数点モードへの遷移時にすべての物理パック・データ・レジスタの内容を浮動小数点レジスタ・ファイルにコピーする場合、物理パック・データ・レジスタ・ファイルは完全に物理浮動小数点レジスタ・ファイル上にエイリアス化される。これに対して、部分エイリアス化では、「有用な」データを含むレジスタのみの内容がコピーされる。どのレジスタが有用なデータを含むかは、任意の数の基準に基づいて判定することができる。たとえば、対応するタグが非空状態を示す物理浮動小数点レジスタのみに記憶されているデータを物理パック・データ・レジスタにコピーすることによって部分エイリアス化を実施することができる。もちろん、実施形態は、パック・データ命令を実行する際に浮動小数点タグを使用することも、あるいは物理パック・データ・レジスタを物理浮動小数点レジスタ上に部分的にエイリアス化するための別々のパック・データ・タグを含むこともできる。別法として、接触された（読取りまたは書込み、あるいはその両方が行われた）パック・データ・レジスタまたは浮動小数点レジスタ、あるいはその両方を、有用なデータを含むレジスタとみることもできる。空状態または非空状態を示すのではなく、あるいは空状態または非空状態を示すだけでなく、この目的のために浮動小数点タグを使用することができる。別法として、どのレジスタが接触されたかを記録するために、浮動小数点レジスタまたはパック・データ・レジスタ、あるいはその両方用の追加表示を含めることができる。部分エイリアス化を実施する際の良好なプログラミング技法は、遷移時にデータがコピーされなかったレジスタを不定値を含むレジスタとみなさなければならないと仮定することである。

【０１０７】

パック・データ状態レジスタ６５５は、１組のパック・データ・ダーティ・フィールド６６５と、スペキュラティブ・フィールド６７０と、モード・フィールド６７５と、例外状況フィールド６８０と、ＥＭＭＳフィールド６８５とを含む。各パック・データ・ダーティ・フィールド６６５はそれぞれの異なるパック・データ・レジスタ６５０に対応し、ダーティ表示を記憶するために使用される。パック・データ・レジスタ６５０と浮動小数点レジスタ６１５との間に対応する関係があるので、各ダーティ表示は、それぞれの異なる浮動小数点レジスタ６１５との対応する関係を有する。１つのパック・データ・レジスタ６５０に値が書き込まれると、そのレジスタの対応するダーティ表示がダーティ状態を示すように変更される。遷移装置６００によってパック・データ装置６１０から浮動小数

点装置６０５への遷移が行われると、対応するダーティ表示がダーティ状態を示す浮動小数点レジスタ６１５の符号フィールドおよび指数フィールドに１が書き込まれる。このように、図４Ｂのステップ４３０を実施することができる。

【０１０８】

モード・フィールド６７５は、プロセッサの現在の動作モード、すなわち、現在浮動小数点装置６０５が使用されている浮動小数点モード、またはパック・データ装置６１０が使用されているパック・データ・モードを識別するモード表示を格納するために使用される。プロセッサが浮動小数点モードでありパック・データ命令を受け取った場合、浮動小数点モードからパック・データ・モードへの遷移を実行しなければならない。逆に、プロセッサがパック・データ・モードであり浮動小数点命令を受け取った場合、パック・データ・モードから浮動小数点モードへの遷移を実行しなければならない。したがって、パック・データ命令と浮動小数点命令のどちらかを受け取ったときに、モード表示をポーリングして、遷移が必要かどうかを判定することができる。遷移が必要である場合は、その遷移が実行され、それに応じてモード表示が変更される。モード表示の動作については、本明細書で図７Ａないし図９を参照して詳しく説明する。

【０１０９】

例外状況フィールド６８０は、例外状況表示を格納するために使用される。例外状況表示は、前の浮動小数点命令を実行した結果として生じた未処理の例外があるかどうかを識別するために、パック・データ命令の実行時に使用される。一実施形態では、そのような例外が未処理であることを例外状況表示が示す場合、そのような例外はパック・データ・モードへの遷移の前に処理される。一実施形態では、この目的のために浮動小数点装置６０５によって使用される表示はコード化され、あるいは例外状況表示として例外状況フィールドに直接コピーされる。

【０１１０】

ＥＭＭＳフィールド６８５は、最後に実行されたパック・データ命令がＥＭＭＳ命令であるかどうかを識別するＥＭＭＳ表示を記憶するために使用される。一実施形態では、ＥＭＭＳ命令を実行したときに、ＥＭＭＳ表示は、最後に実行されたパック・データ命令がＥＭＭＳ命令であることを示す１に変更される。これに対して、他のすべてのパック・データ命令を実行したときに、ＥＭＭＳ表示は零に変更される。遷移装置６００は、パック・データ・モードから浮動小数点モードに遷移する際にＥＭＭＳ表示をポーリングし、最後のパック・データ命令がＥＭＭＳ命令であったかどうかを判定する。最後に実行されたパック・データ命令がＥＭＭＳ命令である場合、遷移装置６００はすべてのタグ６２０を空状態に変更する。しかし、最後に実行されたパック・データ命令がＥＭＭＳではないことをＥＭＭＳが示す場合、遷移装置６００はすべてのタグ６２０を非空状態に変更する。このように、タグは、図４Ｂのステップ４３２および４４０と同様に変更される。

【０１１１】

スペキュラティブ・フィールド６７０は、浮動小数点モードからパック・データ・モードへの遷移がスペキュラティブであるかどうかを識別するスペキュラティブ表示を格納するために使用される。遷移がスペキュラティブである場合、浮動小数点装置６０５に戻る遷移が必要な場合に時間を節約することができる。モード表示の動作については、本明細書で図７Ａないし図９を参照して詳しく説明する。

【０１１２】

図６Ｂは、本発明の実施形態による図６Ａの浮動小数点スタック参照ファイルの一部の拡大図を示すブロック図である。図６Ｂは、１組のタグ６２０内のあるタグを選択的に変更するタグ修飾装置６９０を含む浮動小数点スタック参照装置６３０を示す。図６Ｂに示した実施形態では、１組のタグ６２０はそれぞれ、空状態であるか、それとも非空状態であるかを示す１ビットのみを含む。タグ修飾装置６９０は１組のＴＯＳ調整装置６９６と検査／修正装置６９８とを含む。各ＴＯＳ調整装置６９６は、実施態様に応じて１つまたは複数のマイクロ演算を受け取るためにマイクロ演算回線６９２に結合される（たとえば、１つのマイクロ演算のみを受け取るＴＯＳ調整装置は１つしか存在できない）。少なく

とも、タグの変更を必要とする浮動小数点命令に関するマイクロ演算を、ＴＯＳ調整装置６９６が受け取る。もちろん、すべての各マイクロ演算またはその関連部分のみをＴＯＳ調整装置６９６が受け取るように浮動小数点スタック参照装置６３０を構成することができる。

【０１１３】

マイクロ演算を受け取ったことに応答して、ＴＯＳ調整装置は、少なくとも１）マイクロ演算によって識別される１組のタグ６２０内のあるタグのアドレスと、２）そのタグに対して実行する処置（たとえば、０または１に変更する、ポーリングするなど）を示す信号を検査／修正装置６９８へ送信する。本発明を理解するうえでタグのポーリングは必要とされないので、これについて本明細書ではこれ以上説明しない。各ＴＯＳ調整装置６９６は、現在のＴＯＳ値を受け取りそれに応じてタグ・アドレスを調整するために回線６９４にも結合される。検査／修正装置６９８は少なくとも書込み線によって各タグ６２０に結合される。たとえば、検査／修正装置６９８は書込み線によってタグ６４５に結合される。タグ・アドレスおよび対応する信号を受け取ったことに応答して、検査／修正装置６９８は必要な検査または修正、あるいはその両方を実行する。複数のマイクロ演算を一度に受け取ることのできる実施態様では、検査／修正装置６９８はマイクロ演算間の比較も実行し、それらのマイクロ演算が同じタグを修正するものであるかどうかを判定する（たとえば、マイクロ演算１ではタグ１を１に変更する必要があり、それに対して、マイクロ演算１と同じ時間に受け取ったマイクロ演算２ではタグ１を０に変更する必要があると仮定する）。同じタグを修正する場合、検査／修正装置６９８は、最後に実行すべきマクロ演算はどれかを判定し、そのマイクロ演算に応じてタグを変更する。上記の例では、マイクロ演算１の後にマイクロ演算２を実行すると仮定しているので、検査／修正装置６９８はタグ１を０を示すように変更する。

【０１１４】

たとえば、タグ（たとえば、タグ６４５）を空状態に変更することを必要とする浮動小数点演算を実行する場合、ＴＯＳ調整装置は現在のＴＯＳ値を受け取り、マイクロ演算線６９２上で、タグを識別するマイクロ演算を受け取る。ＴＯＳ調整装置はタグ（たとえば、タグ６４５）のアドレスを判定し、そのアドレスと、そのタグを空状態に変更すべきであることを示す信号とを検査／修正装置６９８へ送信する。これに応答して、検査／修正装置６９８は、タグ６４５に結合された書込み線上で０を送信することによってタグ６４５を空状態に変更する。

【０１１５】

一実施形態では、一度にすべてのタグを修正しなくても済むように浮動小数点命令を構成することができるので、タグ修飾装置６９０は、一度にすべてのタグは修正できないように実施される。回路の複雑さを回避するために、この既存の機構を使用して、浮動小数点モードへの遷移に応答してタグのグローバル変更を行うことができる。なお、遷移装置６００をマイクロコードで形成する場合、１組のマイクロコード命令によって、復号装置は、８つのタグを変更するいくつかの既存のマイクロ演算を発行する。したがって、最後に実行されたパック・データ命令がＥＭＭＳ命令であることをＥＭＭＳ表示が示している間にパック・データ・モードへの遷移が実行されたことに応答して、復号装置は遷移装置６００にアクセスし、いくつかの既存のマイクロ演算を発行する。これらのマイクロ演算に応答して、タグ修飾装置６９０は、対応するタグを空状態に修正する。これに対して、最後に実行されたパック・データ命令がＥＭＭＳ命令ではないことをＥＭＭＳ表示が示している間にパック・データ・モードへの遷移が実行されたことに応答して、復号装置は遷移装置６００にアクセスし、タグ修飾装置６９０に各タグを非空状態に変更させるいくつかの既存のマイクロ演算を発行する。そのような実施形態では、タグのグローバル変更に約４クロック・サイクルないし８クロック・サイクルが必要である。

【０１１６】

パック・データ・モードへの遷移に応答してすべてのタグを変更する一実施形態について説明したが、代替実施形態では任意の数の機構を使用することができる。たとえば、新

しいマイクロ演算を備え、タグ修飾装置６９０を、新しいマイクロ演算に応答してタグをグローバルに変更できるようにすることによって、すべてのタグの空状態または非空状態への変更を単一のクロック・サイクルで完了することができる。この実施形態では、復号装置に、（いくつかの別々のマイクロ演算ではなく）この単一のマイクロ演算を発行してすべてのタグを空状態または非空状態に変更させるように、遷移装置６００を実装することができる。他の例では、復号装置は、タグ６２０に結合することができ、かつＥＭＭＳ命令を受け取ったことに応答してすべてのタグ６２０を変更する追加ハードウェアを含むことができる。

【０１１７】

前述のように、１組のタグ６２０を１ビット・タグを有するものとして説明したが、１組のタグ６２０は各タグごとに２つのビットがあるように見えるようにすることができる。代替実施形態では、タグの変更後の様々な状態（たとえば、００、０１、１０、１１）を示す追加コード化線または非コード化線を備えることによって各タグごとに２つのビットを形成することができる。

【０１１８】

図７Ａ、図７Ｂ、図７Ｃ、図８、図９は、オペレーティング・システムからは見えず、良好なプログラミング方法を推進し、図６Ａのハードウェア構成を使用して実施することができるように、１組の浮動小数点レジスタ上にエイリアス化された１組のレジスタに対してパック・データ命令を実行する、本発明の一実施形態による方法を示す。この流れ図は、図４Ａおよび図４Ｂを参照して説明した流れ図に類似している。図４Ａおよび図４Ｂを参照して、ステップが変更され、移動され、かつ／あるいは削除される多数の代替実施形態について説明した。図４Ａおよび図４Ｂで実行されたステップと同様な、図７Ａ、図７Ｂ、図７Ｃ、図８、図９を参照して説明するステップを少なくとも、そのような代替実施形態を使用して実行することができることを理解されたい。この流れ図はステップ７００から始まる。フローはステップ７００からステップ７０２に進む。

【０１１９】

ステップ７０２に示したように、１組のビットが命令としてアクセスされ、フローはステップ７０４に進む。この１組のビットは、命令によって実行される演算を識別する命令コードを含む。したがって、ステップ７０２は、図４Ａのステップ４０２に類似している。

【０１２０】

ステップ７０４で、命令コードが有効であるかどうかが判定される。命令コードが有効でない場合、フローはステップ７０６に進む。そうでない場合、フローはステップ７０８に進む。ステップ７０４は、図４Ａのステップ４０４に類似している。

【０１２１】

ステップ７０６に示したように、無効命令コード例外が生成され、適切なイベント・ハンドラが実行される。したがって、ステップ７０６は、図４Ａのステップ４０６に類似している。

【０１２２】

ステップ７０８で、どんな種類の命令コードを受け取ったかが判定される。命令が浮動小数点命令でも、あるいはパック・データ命令でもない場合、フローはステップ７１０に進む。しかし、命令が浮動小数点命令である場合、フローはステップ７１２に進む。これに対して、命令がパック・データ命令である場合、フローはステップ７１４に進む。したがって、ステップ７０８は図４Ａのステップ４０８に類似している。

【０１２３】

ステップ７１０に示したように、プロセッサは命令を実行する。このステップは本発明を理解するうえで必要とされないので、本明細書ではこれ以上説明しない。ステップ７１０は図４Ａのステップ４１０に類似している。

【０１２４】

ステップ７１２に示したように、ＥＭ表示が１に等しいかどうか（前述のソフトウェア

規約によれば、浮動小数点装置をエミュレートすべきかどうか）と、ＴＳ表示が１に等しいかどうか（前述のソフトウェア規約によれば、部分コンテキスト切換が行われたかどうか）が判定される。ＥＭ表示またはＴＳ表示、あるいはその両方が１に等しい場合、フローはステップ７１６に進む。そうでない場合、フローはステップ７２０に進む。したがって、ステップ７１２は図４Ａのステップ４１２に類似している。

【０１２５】

ステップ７１６で、装置使用不能例外が発生し、対応するイベント・ハンドラが実行される。したがって、ステップ７１６は図４Ａのステップ４１６に類似している。前述のように、ＥＭ表示およびＴＳ表示を使用して、浮動小数点命令をエミュレートすべきかどうか、あるいは部分コンテキスト切換が行われたかどうか、あるいはその両方を判定するように、このイベント・ハンドラを構成することができる。

【０１２６】

ステップ７１４で、ＥＭ表示が１に等しいかどうかが判定される。したがって、ステップ７１４は図４Ａのステップ４１４に類似している。このため、ステップ７１４で、ＥＭ表示が１に等しいと判定された場合、フローはステップ７１８ではなくステップ７０６に進む。そうでない場合、フローはステップ７１８に進む。

【０１２７】

前述のように、ステップ７０６で、無効命令コード例外が発生し、対応するイベント・ハンドラが実行される。ＥＭ＝１のときのパック・データ命令の実行の試みを無効命令コード例外に切り換えることによって、上記で図４Ａのステップ４０６を参照して説明したように実施形態はオペレーティング・システムから見えなくなる。

【０１２８】

オペレーティング・システムから見えないようにＥＭ表示を処理する一実施形態について説明したが、代替実施形態では他の技法を使用することができる。たとえば、代替実施形態では、ＥＭ表示が１に等しいときにパック・データ命令の実行が試みられたことに応答して装置使用不能例外と、異なる既存のイベントと、新しいイベントのいずれかを発生させることができる。他の例を挙げると、代替実施形態では、パック・データ命令を実行するときにＥＭ表示を無視することができる。

【０１２９】

ステップ７１８に示したように、ＴＳ表示が１に等しいかどうか（前述のソフトウェア規約によれば、部分コンテキスト切換が行われたかどうか）が判定される。ＴＳ表示が１に等しい場合、フローはステップ７１６に進む。そうでない場合、フローはステップ７２２に進む。したがって、ステップ７１８は図４Ａのステップ４１８に類似している。

【０１３０】

前述のように、ステップ７１６で、装置使用不能例外が発生し、対応するイベント・ハンドラが実行される。ステップ７１６は図４Ａのステップ４１８に類似している。ステップ７１４が、ＥＭ表示が１に等しい状況を無効命令コード例外に切り換えているので、ＥＭ表示は０に等しくなければならず、ＴＳ表示は１に等しくなければならない。ＴＳ表示が１に等しいので、イベント・ハンドラはプロセッサに、上記で部分コンテキスト切換に関して説明したように機能させ（浮動小数点装置の内容を記憶し、必要に応じて正しい浮動小数点状態を復元する）、かつプロセッサに、ステップ７０２で受け取った命令の実行を再開することによって実行を再開させる。パック・データ状態は浮動小数点状態上にエイリアス化されているので、このイベント・ハンドラは浮動小数点状態とパック・データ状態の両方のために働く。このため、この方法はオペレーティング・システムからは見えない。もちろん、代替実施形態ではこのイベント・ハンドラを任意の数の方法を利用できるようにすることができる。

【０１３１】

オペレーティング・システムから見えないようにＴＳ表示を処理する一実施形態について説明したが、代替実施形態では他の技法を使用することができる。たとえば、代替実施形態ではＴＳ表示を実施しなくてもよい。そのような代替実施形態は、ＴＳ表示を使用し

て部分コンテキスト切換を実施するオペレーティング・システムとの互換性を有さない。しかし、このような代替実施形態は、ＴＳ表示を使用した部分コンテキスト切換をサポートしない既存のオペレーティング・システムとの互換性を有する。他の例を挙げると、ＴＳ表示が１に等しいときのパック・データ命令の実行の試みを新しいイベント・ハンドラ、または修正された既存のイベント・ハンドラに切り換えることができる。このイベント・ハンドラは、この状況に応じて適切とみなされる処置をとるようにすることができる。たとえば、パック・データ状態が浮動小数点状態上にエイリアス化されない実施形態では、このイベント・ハンドラは、パック・データ状態または浮動小数点状態、あるいはその両方を記憶することができる。

【０１３２】

前述のように、浮動小数点命令を実行する間にある数値エラーが発生した場合、そのようなエラーは、次の浮動小数点命令の実行が試みられるまで未処理のままにされ、次の浮動小数点命令の実行に割り込みこのようなエラーを処理することができる。前述のように、図４のステップ４２０とステップ４２２の両方で、処理できるそのような未処理のエラーがあるかどうかが判定される。図４Ａのステップ４２０と同様に、ステップ７２０で、処理できるそのような未処理のエラーがあるかどうかが判定される。そのような未処理のエラーがある場合、フローはステップ７２０からステップ７２４に移る。しかし、ステップ７２０で、そのような未処理のエラーがないと判定された場合、フローはステップ７２６に進む。これに対して、パック・データ命令の実行を試みるときの、前の浮動小数点命令による未処理のエラーがあるかどうかの判定は、以下で詳しく説明する別のステップで実行される。このため、ステップ７２２はステップ４２２とは異なる。

【０１３３】

ステップ７２４で、未処理浮動小数点エラー・イベントが発生する。したがって、ステップ７２４は図４Ａのステップ４２４に類似している。上記で図４Ａのステップ４２４を参照して説明したように、このイベントを内部イベントまたは外部イベントとみなし、それに応じて処理することができる。

【０１３４】

ステップ７２６に示したように、プロセッサが浮動小数点モードで動作していることをモード表示が示しているかどうかが判定される。したがって、ステップ７２６は図４Ｂのステップ４２６とは異なる。プロセッサは、浮動小数点モードではない場合、パック・データ・モードから浮動小数点モードへ遷移し浮動小数点命令を実行する必要がある。したがって、プロセッサが浮動小数点モードではない場合、フローはステップ７２８に進む。そうでない場合、フローはステップ７３２に進む。

【０１３５】

ステップ７２８で、プロセッサはパック・データ・モードから浮動小数点モードへ遷移し、フローはステップ７３０に進む。ステップ７２８は図６Ａの遷移装置６００によって実行され、このステップについては図９を参照して詳しく説明する。

【０１３６】

ステップ７３０に示したように、「マイクロ再開」を実行することによって、ステップ７０２で受け取った命令が再開される。一実施形態では、マイクロコードを使用してステップ７２８が実行され、命令がマイクロ再開されるので、オペレーティング・システム・イベント・ハンドラを実行する必要はない。このため、プロセッサの外部での処置を施さずに現在のタスクの実行を再開することができる。すなわち、オペレーティング・システム・イベント・ハンドラなどの非マイクロコード・イベント・ハンドラを実行する必要はない。したがって、プロセッサは、オペレーティング・システムを含め、ソフトウェアから見えないようにパック・データ・モードから浮動小数点モードへ遷移することができる。このように、この実施形態は既存のオペレーティング・システムとの互換性を有する。代替実施形態をより低い互換性を有するようにすることができる。たとえば、プロセッサに追加イベントを組み込むことができ、この遷移を実行する追加イベント・ハンドラをオペレーティング・システムに追加することができる。

【０１３７】
ステップ７３２に示したように、浮動小数点命令が実行される。ステップ７３２は図４Ｂのステップ４２６に類似している。オペレーティング・システムから見えないようにするために、一実施形態ではまた、必要に応じてタグを変更し、このとき処理できる数値エラーを報告し、他の数値エラーを未処理のままにしておく。前述のように、タグを変更することによって、対応するタグが非空状態を示す浮動小数点レジスタのみの内容を記憶するオペレーティング・システム技法からはこの実施形態が見えないようにすることができる。しかし、前述のように、代替実施形態をより少しのオペレーティング・システム技法との互換性を有するようにすることができる。たとえば、既存のオペレーティング・システムがタグを使用しない場合でも、タグを実施しないプロセッサは、そのオペレーティング・システムとの互換性を有する。さらに、本発明では数値浮動小数点例外を未処理のままにしておく必要はなく、したがって、数値浮動小数点例外を未処理のままにしない実施形態は本発明の範囲内である。

【０１３８】
ステップ７２２に示したように、プロセッサがパック・データ・モードであることをモード表示が示しているかどうかが判定される。したがって、ステップ７２２は図４Ａのステップ４２２とは異なる。ステップ７２２が実行され、プロセッサがパック・データ命令を実行するのに適切なモードであるかどうかが判定される。プロセッサがパック・データ・モードではない場合、プロセッサは浮動小数点モードからパック・データ・モードに遷移しパック・データ命令を実行する必要がある。したがって、プロセッサがパック・データ・モードではない場合、フローはステップ７３４に進む。そうでない場合、フローはステップ７３８に進む。

【０１３９】
ステップ７３４で、プロセッサは浮動小数点モードからパック・データ・モードへ遷移し、フローはステップ７３６に進む。ステップ７３４は図６Ａの遷移装置６００によって実行され、このステップについては図８を参照して詳しく説明する。

【０１４０】
ステップ７３６に示したように、マイクロ再開を実行することによって、ステップ７０２で受け取った命令が再開される。したがって、ステップ７３６はステップ７３０に類似している。

【０１４１】
ステップ７４０で、パック・データ命令がＥＭＭＳ命令であるかどうかが判定される。パック・データ命令がＥＭＭＳ命令である場合、フローはステップ７４２に進む。そうでない場合、フローはステップ７４４に進む。パック・データ命令は独立の装置（すなわち、パック・データ装置）上で実行されるので、ある動作（たとえば、ＥＭＭＳ命令の実行に応答してタグを空状態に変更し、他のパック・データ命令の実行に応答してタグを非空状態に変更すること）を実際に実行するよりも、浮動小数点モードに戻る遷移時に、ステップ７２８で何を行わなければならないかを識別する表示（たとえば、ＥＭＭＳ表示）を記憶する方が効率的である。ＥＭＭＳ表示とその他の表示の使用については、図９で詳しく説明するパック・データ・モードから浮動小数点モードへの遷移のステップを参照して説明する。

【０１４２】
ステップ７４２に示したように、ＥＭＭＳ表示は、最後のパック・データ命令がＥＭＭＳ命令であったことを示すように変更される。ステップ７４２の完了時に、プロセッサは次の命令（ステップ７０２で受け取った命令の後に論理的に続く命令）を実行することができる。

【０１４３】
ステップ７４４から明らかなように。最終パック・データ命令はＥＭＭ命令でなかったことを示すようにＥＭＭ表示が変更される。フローはステップ７４４からステップ７４６へと進む。表示動作の説明は図８との関連で後に説明する。

【０１４４】
ステップ７４６に示したように、パック・データ命令がプロセッサにエイリアス化レジスタに書込みを行わせるかどうかが判定される。そうである場合、フローはステップ７４８に進む。そうでない場合、フローはステップ７５０に進む。したがって、ステップ７４６は図４Ｂのステップ４３６に類似している。
【０１４５】
ステップ７４８で、エイリアス化レジスタの対応するダーティ表示がダーティ状態に変更され、フローはステップ７５０に進む。このようなダーティ表示は、ステップ７２８で、パック・データ・モードから浮動小数点モードへ遷移する際に使用される。前述のように、このようなダーティ表示は、符号フィールドおよび指数フィールドを１に設定する必要のある浮動小数点レジスタを識別するために使用される。一実施形態では符号フィールドおよび指数フィールドに１が書き込まれるが、代替実施形態ではＮＡＮ（非数値）または無限を表す任意の値を使用することができる。符号フィールドおよび指数フィールドが変更されない代替実施形態ではステップ７４６および７４８は必要とされない。
【０１４６】
ステップ７５０に示したように、ＥＭＭ表示は、最後のパック・データ命令がＥＭＭ命令でなかったことを示すように変更される。ステップ７５０の完了時に、システムは次の命令を実行することができる。もちろん、ＥＭＭＳ命令を使用しない実施形態はステップ７４０、ステップ７４２、ステップ７４４を必要としない。
【０１４７】
したがって、（ワシントン州レッドモンドのマイクロソフト社（Ｍｉｃｒｏｓｏｆｔ Ｃｏｒｐｏｒａｔｉｏｎ）から市販されているＭＳ－ＤＯＳ Ｗｉｎｄｏｗｓブランド動作環境など）既存のオペレーティング・システムとの互換性を有し、良好なプログラミング技法を推進するパック・データ命令を実行する方法および装置について説明した。パック・データ状態は、浮動小数点状態上にエイリアス化されるので、浮動小数点状態の場合と同様に既存のオペレーティング・システムによって保存され復元される。さらに、パック・データ命令を実行することによって発生したイベントを既存のオペレーティング・システム・イベント・ハンドラによって処理することができるので、このようなイベント・ハンドラを修正する必要がなく、新しいイベント・ハンドラを追加する必要がない。このため、プロセッサは後方互換性を有し、更新時に、オペレーティング・システムを開発または修正するために必要なコストおよび時間が必要とされない。
【０１４８】
そのうちのいくつかについて説明したこの実施形態の変形形態は、そのようなオペレーティング・システムとの全互換性または部分互換性を有し、かつ／あるいは良好なプログラミング技法を推進することができる。たとえば、本発明の代替実施形態では、あるステップを流れ図の異なる位置へ移動することができる。本発明の他の代替実施形態では、１つまたは複数のステップを変更または削除することができる。図７Ａまたは図７Ｂまたは図７Ｃ、あるいはそれらの組合せからあるステップを削除する場合、図６Ａ中のあるハードウェアは必要とされない。たとえば、ＥＭＭＳ命令を使用しない場合、ＥＭＭＳ表示は必要とされない。もちろん、本発明は任意の数のシステム・アーキテクチャに有用であり、本明細書で説明するアーキテクチャに限らない。
【０１４９】
さらに、２つの物理レジスタ・ファイルをエイリアス化する方法および装置について説明したが、代替実施形態では任意の数の物理レジスタ・ファイルをエイリアス化し任意の数の異なる種類の命令を実行することができる。また、浮動小数点命令を実行する物理スタック・レジスタ・ファイルと、パック・データ命令を実行する物理フラット・レジスタ・ファイルに関してこの実施形態を説明したが、本明細書の教示を使用して、レジスタ・ファイル上の実行される命令の種類にかかわらず、少なくとも１つの物理スタック・レジスタ・ファイルと少なくとも１つの物理フラット・レジスタ・ファイルをエイリアス化することができる。

【０１５０】
また、浮動小数点命令およびパック・データ命令を実行する方法および装置について説明したが、代替実施形態を、いくつかの異なる種類の命令を実行するようにすることができる。たとえば、前述のように、プロセッサにパック整数演算またはパック浮動小数点演算、あるいはその両方を実行させるようにパック・データ命令を形成することができる。他の例を挙げると、代替実施形態では、パック・データ命令ではなく、あるいはパック・データ命令の他に、スカラ浮動小数点命令およびスカラ整数命令を実行できるように物理レジスタ・ファイルにエイリアス化することができる。他の例を挙げると、代替実施形態では、パック・データ命令を浮動小数レジスタ上にエイリアス化するのではなく、パック・データ命令を整数レジスタ上にエイリアス化することができる。他の例を挙げると、代替実施形態では、スカラ浮動小数点命令、スカラ整数命令、パック命令（整数または浮動小数点、あるいはその両方）の実行を単一の論理レジスタ・ファイル上にエイリアス化することができる。したがって、本明細書の教示を使用して、ソフトウェアから論理的に、いくつかの異なるデータ・タイプに作用する命令を単一の論理レジスタ・ファイルを用いて実行できるように見えるようにすることができる。
【０１５１】
図８は、本発明の一実施形態によって図７Ｃのステップ７３４を実行する方法を示す流れ図である。前述のように、ステップ７３４で、プロセッサは浮動小数点モードからパック・データ・モードへ遷移する。フローはステップ７２２からステップ８００へ移る。
【０１５２】
ステップ８００に示したように、前の浮動小数点命令による未処理のエラーがあるかどうかが判定される。そうである場合、フローはステップ７２４に移る。そうでない場合、フローはステップ８０４に進む。したがって、ステップ８００は第７図のステップ７２０および図４Ａのステップ４２２に類似している。
【０１５３】
前述のように、ステップ７２４で、未処理浮動小数点エラー例外が発生し、適切なイベント・ハンドラが実行される。上記で図４Ａのステップ４２４を参照して説明したように、このイベントを内部イベントまたは外部イベントとみなし、それに応じて処理することができる。代替実施形態では、パック・データ命令の実行時にそのようなエラーは未処理のままにされる。
【０１５４】
ステップ８０４に示したように、浮動小数点レジスタの小数部フィールドに格納されているデータは、パック・データ・レジスタにコピーされる。そうする際に、浮動小数点レジスタに記憶されていたデータをパック・データとして処理することができる。全エイリアス化を行う場合、すべての浮動小数点レジスタの小数部フィールドに格納されているデータが、対応するパック・データ・レジスタにコピーされる。これに対して、部分エイリアス化を行う場合、対応するタグが非空状態を示す浮動小数点レジスタのみの小数部フィールドに格納されているデータが、適切な対応するパック・データ・レジスタにコピーされるように、実施形態を形成することができる。浮動小数点レジスタに記憶されているデータをパック・データとして処理することを許容しない代替実施形態は、ステップ８０４を実行する必要はない。フローはステップ８０４からステップ８０６へ移る。
【０１５５】
ステップ８０６で、ＥＭＭＳ表示が、最後のパック・データ命令がＥＭＭＳ命令ではなかったことを示すように変更され、フローはステップ８０８に進む。このステップは、パック・データ・モードを初期設定するために実行される。
【０１５６】
ステップ８０８に示したように、各ダーティ表示は、クリーン状態を示すように変更され、フローはステップ８１０に進む。ステップ８０６およびステップ８０８は、パック・データ・モードを初期設定するために実行される。
【０１５７】

ステップ８１０に示したように、スペキュラティブ表示は、浮動小数点からパック・データへの遷移がスペキュラティブであることを示すように変更される。ステップ８０４で、浮動小数点レジスタに記憶されているデータがパック・データ・レジスタにコピーされたが、浮動小数点装置の状態は変更されていない。したがって、浮動小数点状態は依然として現在値である（たとえば、浮動小数点レジスタの小数部フィールドに記憶されているデータが、パック・データ・レジスタに記憶されているデータと等価であり、タグが変更されておらず、トップ・オブ・スタック表示が変更されていない）。これに続いてパック・データ命令を実行する場合、パック・データ・レジスタに記憶されているデータが変更され、浮動小数点状態はもはや現在値ではなくなる。このため、パック・データ・モードから浮動小数点モードへの遷移では、浮動小数点状態を更新する必要がある（たとえば、パック・データ・レジスタに記憶されているデータを、浮動小数点レジスタの小数部フィールドにコピーする必要があり、トップ・オブ・スタック表示を０に変更する必要があり、タグを空状態に変更する必要がある）。しかし、パック・データ命令を実行する前に浮動小数点命令の実行を試みる場合（浮動小数点モードからパック・データ・モードへの遷移を生じさせるパック・データ命令を実行する前にイベントが発生した場合、たとえば、パック・データ命令の実行を試みている間にメモリ障害が起こった場合に、こうなる可能性がある）、浮動小数点状態は、依然として現在値であるので更新する必要はない。この更新を回避することによって、パック・データ・モードから浮動小数点モードへの遷移に関するオーバヘッドが著しく低減される。このことを利用して、このステップで、スペキュラティブ表示は、浮動小数点装置からパック・データ装置への遷移がスペキュラティブであることを示すように変更される。浮動小数点状態は依然として現在値である。これに続いてパック・データ命令を実行する場合、スペキュラティブ表示は、上記で第７図のステップ７３８を参照して説明したように、遷移がもはやスペキュラティブではないことを示すように変更される。スペキュラティブ表示の使用については、図９を参照して詳しく説明する。スペキュラティブ表示を使用した一実施形態について説明したが、代替実施形態では、そのようなスペキュラティブ表示の実施を回避することができる。

【０１５８】

ステップ８１２で、プロセッサが現在パック・データ・モードであることを示すようにモード表示が変更される。フローは、ステップ８１２からステップ７３６へ移る。

【０１５９】

図９は、本発明の一実施形態による第７図のステップ７２８を実行する方法を示す流れ図である。前述のように、プロセッサはステップ７２８で、パック・データ・モードから浮動小数点モードへ遷移する。フローはステップ７２６からステップ９００へ移る。

【０１６０】

ステップ９００で、パック・データ・モードへの遷移が依然としてスペキュラティブであることをスペキュラティブ表示が示しているかどうかが判定される。前述のように、スペキュラティブ表示を使用して、パック・データ・モードから浮動小数点モードへの遷移に関するオーバヘッドを低減することができる。ステップ９００で、浮動小数点からパック・データへの遷移がスペキュラティブであると判定され、ステップ９０２ないしステップ９１２が回避され、フローが直接ステップ９１４に進み、遷移オーバヘッドが低減される。そうでない場合、フローはステップ９０２に進む。

【０１６１】

ステップ９０２に示したように、最後のパック・データ命令がＥＭＭＳ命令であったことをＥＭＭＳ表示が示しているかどうかが判定される。そうである場合、フローはステップ９０４に進む。そうでない場合、フローはステップ９０６に進む。前述のように、パック・データ命令が独立の装置（すなわち、パック・データ装置）上で実行されるので、ある種の動作（たとえば、タグを変更すること）を実行するよりも、浮動小数点モードに戻る遷移時に、何を行わなければならないかを識別する表示（たとえば、ＥＭＭＳ表示）を記憶する方が効率的である。したがって、ＥＭＭＳ命令に応答してタグを変更するのでなく、ＥＭＭＳ表示が変更されている。その場合、浮動小数点モードに戻る遷移を実行する

際、それに応じて、本明細書に示したようにタグが変更される。
【０１６２】
ステップ９０４で、すべてのタグが空状態に変更され、フローはステップ９０８に進む。このように、タグは図４Ｂのステップ４３２と同様に変更される。
【０１６３】
ステップ９０６で、すべてのタグが非空状態に変更され、フローはステップ９０８に進む。このように、タグは図４Ｂのステップ４４０と同様に変更される。
【０１６４】
ステップ９０８に示したように、パック・データ・レジスタの内容が浮動小数点レジスタの小数部フィールドにコピーされ、フローがステップ９１０に進む。このように、パック・データ・レジスタに記憶されているデータを浮動小数点データとして処理することができる。さらに、既存のオペレーティング・システムが、マルチタスクを実行する際にすでに浮動小数点状態を記憶しているので、パック・データ状態は、浮動小数点状態の場合と同様に、様々なコンテキスト構造に記憶され、そのような構造から復元される。このように、物理パック・データ・レジスタは物理浮動小数点レジスタ上にエイリアス化され、プロセッサは論理的に、単一の論理レジスタ・ファイルを有するように見える。このため、この実施形態は、オペレーティング・システムを含め、ソフトウェアからは見えない。全エイリアス化を行う場合、すべてのパック・データ・レジスタに記憶されているデータが、対応する浮動小数点レジスタの小数部フィールドにコピーされる。これに対して、部分エイリアス化が実装される場合、接触されたパック・データ・レジスタのみに記憶されているデータが適切な対応する浮動小数点レジスタの小数部フィールドにコピーされるように実施形態を実装することができる。
【０１６５】
ステップ９１０に示したように、トップ・オブ・スタックは初期設定値に変更される。一実施形態では、この値は零である。代替実施形態では、パック・データ命令を実行すると、トップ・オブ・スタック表示が初期設定値に設定される。フローはステップ９１０からステップ９１２へ移る。
【０１６６】
ステップ９１２に示したように、対応するダーティ表示がダーティ状態である浮動小数点レジスタの符号フィールドおよび指数フィールドに１が格納される。このように、図４Ｂのステップ４３８が実行される。フローはステップ９１２からステップ９１４へ移る。
【０１６７】
ステップ９１４で、プロセッサが浮動小数点モードで動作していることを示すようにモード表示が変更され、フローがステップ７３０に進む。このように、パック・データ・モードから浮動小数点モードへの遷移が実行される。
【０１６８】
図１０は、本発明の他の実施形態による、単一の物理レジスタ・ファイルを使用してパック・データ状態を浮動小数点状態上にエイリアス化する装置内のデータ・フローを示すブロック図である。図１０に示した装置は、図５の命令セット装置５６０として使用することができる。一実施形態では、図１０の装置は少なくとも命令セット５８０を実行することができる。図１０は、復号装置１００２と、名前変更装置１００４と、リタイヤメント装置１００６と、発行装置１００８と、実行装置１０１０と、１組の状態レジスタ１０１２と、マイクロコードＲＯＭ１０１４を示す。
【０１６９】
復号装置１００２は、プロセッサが受け取った命令を制御信号またはマイクロコード・エントリ・ポイント、あるいはその両方に復号するために使用される。このようなマイクロコード・エントリ・ポイントは、復号装置１００２によってプロセッサ内の様々な装置へ送られるマイクロ演算（「ｕｏｐｓ」とも呼ぶ）のシーケンスを識別する。ある種のマイクロ演算は復号装置１００２に記憶することができるが、一実施形態では、マイクロ演算の大部分がマイクロコードＲＯＭ１０１４に記憶される。この実施形態では、復号装置

１００２はマイクロコード・エントリ・ポイントをマイクロコードＲＯＭ１０１４へ送り、マイクロコードＲＯＭは、必要なマイクロ演算を復号装置１００２へ送り返すことによって応答する。

【０１７０】

復号装置１００２が受け取る命令の大部分は、命令の演算を実行すべき１つまたは複数のオペランド（データ、レジスタ位置、メモリ内の位置のいずれか）を含む。レジスタを識別するオペランドは名前変更装置１００４へ送られる。

【０１７１】

名前変更装置１００４およびリタイヤメント装置１００６は、レジスタ名前変更を実施するために使用される。レジスタ名前変更技法は良く知られており、いくつかの異なる命令がレジスタなど限られた数の記憶位置を使用することを試みるために生じる記憶域競合を回避するために実行される。記憶域競合は、競合する命令どうしが他の点では独立しているにもかかわらず互いに干渉するときに起こると考えられている。記憶域競合は、レジスタと値との間の対応を再確立するために使用される追加レジスタ（本明細書ではバッファ・レジスタと呼ぶ）を設けることによって除去することができる。レジスタ名前変更を実施するために、プロセッサは通常、生成されるあらゆる新しい値、すなわちレジスタに書込みを行うあらゆる命令に対してそれぞれの異なるバッファ・レジスタを割り振る。値を読み取るために最初のレジスタを識別する命令はその代わりに、割り振られたバッファ・レジスタ内の値を得る。したがって、ハードウェアは、最初のレジスタの名前を変更して、命令を識別しバッファ・レジスタおよび正しい値を識別する。いくつかの異なる命令中の同じレジスタ識別子は、レジスタ割当てに対するレジスタ参照の位置に応じて、それぞれの異なるハードウェア・レジスタにアクセスすることができる。レジスタ名前変更の詳細な説明については、Ｊｏｈｎｓｏｎ，Ｍｉｋｅ著「Ｓｕｐｅｒｓｃａｌａｒ Ｍｉｃｒｏ Ｐｒｏｃｅｓｓｏｒ Ｄｅｓｉｇｎ」（１９９１年、ＰＴＲ Ｐｒｅｎｔｉｃｅ－Ｈａｌｌ，Ｉｎｃ．、ニュージャージー）、「Ｆｌａｇ Ｒｅｎａｍｉｎｇ ａｎｄ Ｆｌａｇ Ｍａｓｋ Ｗｉｔｈｉｎ Ｒｅｇｉｓｔｅｒ Ａｌｉａｓ Ｔａｂｌｅ」（米国特許出願第０８／２０４５２１号、Ｃｏｌｗｅｌｌ等）、「Ｉｎｔｅｇｅｒ ａｎｄ Ｆｌｏａｔｉｎｇ Ｐｏｉｎｔ Ｒｅｇｉｓｔｅｒ Ａｌｉａｓ Ｔａｂｌｅ Ｗｉｔｈｉｎ Ｐｒｏｃｅｓｓｏｒ Ｄｅｖｉｃｅ」（米国特許出願第０８／１２９６７８号、Ｃｌｉｆｔ等）、「Ｐａｒｔｉａｌ Ｗｉｄｔｈ Ｓｔａｌｌｓ Ｗｉｔｈｉｎ Ｒｅｇｉｓｔｅｒ Ａｌｉａｓ Ｔａｂｌｅ」（米国特許出願第０８／１７４８４１号、Ｃｏｌｗｅｌｌ等）を参照されたい。ある命令が（未処置のままにされないイベントを発生せずに）実行を首尾良く完了すると、命令に割り振られたバッファ・レジスタが「リタイア」される。すなわち、値が、バッファ・レジスタから、命令中で識別される最初のレジスタへ転送される。代替実施形態では、インタロック、部分名前変更など記憶域競合を解消する任意の数の技法を利用することができる。

【０１７２】

リタイヤメント装置１００６は、１組のバッファ・レジスタ１０２０と、１組のＦＰ／ＰＤレジスタ１０２２と、１組の整数レジスタ１０２４とを含む。１組のバッファ・レジスタ１０２０は、レジスタ名前変更に使用される追加レジスタを備える。一実施形態では、１組のバッファ・レジスタ１０２０は４０個のレジスタを含み、代替実施形態では、任意の数のレジスタを使用することができる。この実施形態では、１組のバッファ・レジスタ１０２０を順序変更バッファとして操作することができる。

【０１７３】

一実施形態では、ＦＰ／ＰＤレジスタ１０２２および整数レジスタ１０２４はソフトウェアから見える。すなわち、これらは命令中で識別されるレジスタであり、したがって、ソフトウェアからは、これらが浮動小数点データ、パック・データ、整数データを実行する唯一のレジスタであるように見える。これに対して、バッファ・レジスタ１０２０はソフトウェアから見えない。したがって、ＦＰ／ＰＤレジスタ１０２２は、ソフトウェアから単一の論理レジスタ・ファイルとして見える単一の物理レジスタ・ファイルである。一

実施形態では、１組のＦＰ／ＰＤレジスタ１０２２および１組の整数レジスタ１０２４はそれぞれ、既存のインテル・アーキテクチャ・ソフトウェアとの互換性を維持するために８つのレジスタを含む。しかし、代替実施形態では任意の数のレジスタを使用することができる。

【０１７４】

名前変更装置１００４は、ＦＰ／ＰＤマッピング装置１０３０と、ＦＰ／ＰＤマッピング・テーブル１０３２と、１組のタグ１０３４と、整数マッピング装置１０４０と、整数マッピング・テーブル１０４２とを含む。オペランドが名前変更装置１００４によって受信されると、それが浮動小数点オペランドであるか、それともパック・データ・オペランドであるか、それとも整数オペランドであるかが判定される。

【０１７５】

整数オペランドは整数マッピング装置１０４０によって受信される。整数マッピング装置１０４０は整数マッピング・テーブル１０４２を制御する。一実施形態では、整数マッピング・テーブル１０４２は、整数レジスタ１０２４内のレジスタと同じ数のエントリを含む。整数マッピング・テーブル１０４２内の各エントリはそれぞれの異なる整数レジスタ１０２４に対応する。図１０では、エントリ１０５０は整数レジスタ１０５２に対応する。プロセッサに整数レジスタ（たとえば、整数レジスタ１０５２）への書込みを行わせる命令を受け取ると、整数マッピング装置１０４０は、１組のバッファ・レジスタ１０２０内の使用可能なレジスタ（たとえば、バッファ・レジスタ１０５４）を識別するポインタを、整数マッピング・テーブル１０４２内の整数レジスタの対応するエントリ（たとえば、エントリ１０５０）に格納することによって、１つのバッファ・レジスタ１０２０を割り振る。データは、選択されたバッファ・レジスタ（たとえば、バッファ・レジスタ１０５４）に書き込まれる。このオペランドを生成した命令の実行が割込みなしに（イベントを発生せずに）完了すると、リタイヤメント装置１００６は、データを、選択されたバッファ・レジスタ（たとえば、バッファ・レジスタ１０５４）から適切な整数レジスタ（たとえば、整数レジスタ１０５２）にコピーすることによってこのデータを「コミット」し、整数マッピング装置１０４０に、このデータがエントリ（たとえば、エントリ１０５０）の対応する整数レジスタに記憶されたことを示すようにエントリの内容を更新させる。

【０１７６】

プロセッサに整数レジスタを読み取らせる命令を受信すると、プロセッサは、整数マッピング装置１０４０を使用して、整数マッピング・テーブル１０４２内の整数レジスタの対応するエントリ（たとえば、エントリ１０５０）の内容にアクセスする。エントリがバッファ・レジスタ（たとえば、バッファ・レジスタ１０５４）へのポインタを含む場合、プロセッサはそのバッファ・レジスタの内容を読み取る。しかし、このエントリの対応する整数レジスタ（たとえば、整数レジスタ１０５２）にデータが記憶されていることをこのエントリの内容が示している場合、プロセッサは、エントリの対応する整数レジスタの内容を読み取る。したがって、本発明のこの実施形態では、整数レジスタ１０２４は固定レジスタ・ファイルとして実装される。

【０１７７】

ＦＰ／ＰＤマッピング装置１０３０は、ＦＰ／ＰＤマッピング・テーブル１０３２およびタグ１０３４を制御する。前述のように、各タグは任意の数のビットを使用することができる。整数マッピング装置１０４０と同様に、ＦＰ／ＰＤマッピング・テーブル１０３２は、ＦＰ／ＰＤレジスタ１０２２内のレジスタと同じ数のエントリを含む。ＦＰ／ＰＤマッピング・テーブル１０３２内の各エントリはそれぞれの異なるＦＰ／ＰＤレジスタ１０２２に対応する。浮動小数点オペランドおよびパック・データ・オペランドは、ＦＰ／ＰＤマッピング装置１０３０が受け取り、バッファ・レジスタ１０２０にマップされ、ＦＰ／ＰＤレジスタ１０２２にリタイアされる。したがって、浮動小数点状態およびパック・データ状態は、ユーザに見える単一のレジスタ・ファイル上にエイリアス化される。既存のオペレーティング・システムは、プロセッサにマルチタスク時に浮動小数点状態を記

憶させるように実施されるので、この同じオペレーティング・システムは、プロセッサに、浮動小数点レジスタ上にエイリアス化されたパック・データ状態を記憶させる。

【０１７８】

一実施形態では、パック・データ・オペランドは整数オペランドと同様に処理され、パック・データ・レジスタは固定レジスタ・ファイルとして実装される。したがって、プロセッサにＦＰ／ＰＤレジスタへの書込みを行わせるパック・データ命令を受け取ると、ＦＰ／ＰＤマッピング装置１０３０は、１組のバッファ・レジスタ１０２０内の使用可能なレジスタを識別するポインタを、ＦＰ／ＰＤマッピング・テーブル１０３２内のＦＰ／ＰＤレジスタの対応するエントリに格納することによって、１つのバッファ・レジスタ１０２０を割り振る。データは、選択されたバッファ・レジスタに書き込まれる。このオペランドを生成した命令の実行が割込みなしに（イベントを発生せずに）完了すると、リタイヤメント装置１００６は、データを、選択されたバッファ・レジスタから適切なＦＰ／ＰＤレジスタ（ＦＰ／ＰＤマッピング・テーブル１０３２内のエントリに対応するＦＰ／ＰＤレジスタ）にコピーすることによってこのデータを「コミット」し、ＦＰ／ＰＤマッピング装置１０３０に、このデータがエントリの対応するＦＰ／ＰＤレジスタに記憶されたことを示すようにＦＰ／ＰＤマッピング・テーブル１０３２内のエントリを更新させる。

【０１７９】

パック・データ命令を実行する際、レジスタは固定レジスタ・ファイルとして実現されるが、本発明の一実施形態では、浮動小数点命令を実行する際、レジスタは、（オペレーティング・システムを含め）既存のインテル・アーキテクチャ・ソフトウェアとの互換性を有するようにスタック参照レジスタ・ファイルとして実装される。このため、ＦＰ／ＰＤマッピング装置１０３０はＦＰ／ＰＤマッピング・テーブル１０３２をパック・データ・オペランド用の固定レジスタ・ファイルと浮動小数点オペランド用のスタックの両方として操作できなければならない。この目的のために、ＦＰ／ＰＤマッピング装置１０３０は、トップ・オブ・スタック・フィールド１０７２を有する浮動小数点状態レジスタ１０７０を含む。トップ・オブ・スタック・フィールド１０７２は、現在浮動小数点スタックの１番上に配置されているレジスタを表すＦＰ／ＰＤマッピング・テーブル１０３２内のエントリを識別するトップ・オブ・スタック表示を記憶するために使用される。もちろん、代替実施形態では、浮動小数点命令を実行する際にレジスタをフラット・レジスタ・ファイルとして操作することができる。

【０１８０】

プロセッサにＦＰ／ＰＤレジスタへの書込みを行わせる浮動小数点命令を受け取ると、ＦＰ／ＰＤマッピング装置１０３０は、トップ・オブ・スタック表示を変更し、１組のバッファ・レジスタ１０２０内の使用可能なレジスタを識別するポインタを、ＦＰ／ＰＤマッピング・テーブル１０３２内のトップ・オブ・スタック・レジスタの対応するエントリに格納することによって、１つのバッファ・レジスタ１０２０を割り振る。データは、選択されたバッファ・レジスタに書き込まれる。このオペランドを生成した命令の実行が割込みなしに（イベントを発生せずに）完了すると、リタイヤメント装置１００６は、データを、選択されたバッファ・レジスタから適切なＦＰ／ＰＤレジスタ（ＦＰ／ＰＤマッピング・テーブル１０３２内のエントリに対応するＦＰ／ＰＤレジスタ）にコピーすることによってこのデータを「コミット」し、ＦＰ／ＰＤマッピング装置１０３０に、このデータがエントリの対応するＦＰ／ＰＤレジスタに記憶されたことを示すようにＦＰ／ＰＤマッピング・テーブル１０３２内のエントリを更新させる。

【０１８１】

プロセッサにＦＰ／ＰＤレジスタを読み取らせる浮動小数点命令を受信すると、プロセッサは、ＦＰ／ＰＤマッピング・テーブル１０３２内のトップ・オブ・スタック・レジスタの対応するエントリの内容にアクセスし、それに応じてスタックを変更する。バッファ・レジスタを指し示すポインタがそのエントリに記憶されている場合、プロセッサはそのバッファ・レジスタの内容を読み取る。しかし、ＦＰ／ＰＤレジスタ１０２２内のそのエントリの対応するＦＰ／ＰＤにデータが記憶されていることをエントリの内容が示してい

る場合、プロセッサはそのＦＰ／ＰＤレジスタの内容を読み取る。

【０１８２】

したがって、ＦＰ／ＰＤマッピング装置１０３０が浮動小数点オペランドをスタック参照レジスタ・ファイル上にマップするので、スタックの１番上のレジスタを基準としてＦＰ／ＰＤマッピング・テーブル１０３２内のエントリにアクセスしなければならない。これに対して、ＦＰ／ＰＤマッピング装置１０３０がパック・データ・オペランドを固定レジスタ・ファイル上にマップするので、レジスタＲ０を基準としてＦＰ／ＰＤマッピング・テーブル１０３２内のエントリにアクセスしなければならない。プロセッサに、レジスタＲ０を基準としてＦＰ／ＰＤマッピング・テーブルのエントリにアクセスさせるには、トップ・オブ・スタック表示をレジスタＲ０を示すように変更しなければならない。したがって、トップ・オブ・スタック表示は、プロセッサがパック・データ命令を実行している間はレジスタＲ０を示すように変更しなければならない。これは、浮動小数点モードからパック・データ・モードへの遷移時にはレジスタＲ０を示すようにトップ・オブ・スタック表示を変更し、パック・データ命令の実行時にはトップ・オブ・スタック表示を変更しないようにすることによって行うことができる。このように、浮動小数点スタックをマップするために使用されるのと同じ回路を使用して固定パック・データ・レジスタ・ファイルをマップすることができる。このため、図６Ａを参照して説明した実施形態と比べて、回路の複雑さが低減され、ダイ面積が節約される。同じ回路を使用してパック・データ・オペランドと浮動小数点オペランドの両方をマップする一実施形態について説明したが、代替実施形態では別々の回路を使用することができる。

【０１８３】

実行中の命令の種類にはかかわらず、一実施形態では、バッファ・レジスタの割振りと割振り解除が同様に処理される。リタイヤメント装置１００６は、割振りフィールド１０６２とリタイヤメント・フィールド１０６４とを有する制御レジスタ１０６０を含む。割振りフィールド１０６２は、次に使用するバッファ・レジスタを識別する割振りポインタを格納する。ＦＰ／ＰＤマッピング装置１０３０と整数マッピング装置１０４０のどちらかがレジスタを必要とするとき、現在の割振りポインタが適切なマッピング・テーブル（すなわち、ＦＰ／ＰＤマッピング装置１０３０または整数マッピング・テーブル１０４２）に記憶され、割振りポインタが増分される。また、名前変更装置１００４は、命令がパック・データ命令であるかどうかと、プロセッサがパック・データ・モードであるかどうかを示す信号をリタイヤメント装置１００６へ送信する。

【０１８４】

割り振られたバッファ・レジスタ内で、リタイヤメント装置１００６は準備完了フィールド１０８２に準備完了表示を格納する。準備完了表示は最初、バッファ・レジスタがリタイアする準備が完了していないことを示すように変更される。しかし、バッファ・レジスタのデータ・フィールド１０８０にデータが書き込まれると、バッファ・レジスタの準備完了表示が、バッファ・レジスタがリタイアする準備が完了していることを示すように変更される。

【０１８５】

制御レジスタ１０６０のリタイヤメント・フィールド１０６４は、次にリタイアするバッファ・レジスタを識別するリタイヤメント・ポインタを格納する。このバッファ・レジスタの準備完了表示が準備完了状態に変更されると、リタイヤメント装置１００６は、このバッファ・レジスタ内のデータをコミットできるかどうかを判定しなければならない。以下で詳しく説明するように、リタイヤメント装置１００６の一実施形態は、例外を発生させなければならない場合（たとえば、装置使用不能例外、未処理浮動小数点エラー例外、無効命令コード例外など）、あるいはパック・データ・モードと浮動小数点モードとの間の遷移が必要とされる場合にはデータをコミットしない。データをコミットできる場合、そのデータが適切なＦＰ／ＰＤレジスタまたは整数レジスタにコピーされ、リタイヤメント・ポインタが次のバッファ・レジスタに増分される。リタイヤメント・ポインタおよび割振りポインタを制御レジスタに記憶されるものとして説明したが、代替実施形態では

、これらのポインタと、本明細書に記載のその他の情報（たとえば、ＥＭＭＳ表示やモード表示など）を１組のフリップフロップなどある種の順次要素の形で記憶することができる。

【０１８６】

リタイヤメント装置１００６が別々の３組のレジスタを含み、データがバッファ・レジスタからＦＰ／ＰＤレジスタまたは整数レジスタにコミットされる一実施形態について説明したが、代替実施形態は、任意の数の異なる組のレジスタを含むようにすることができる。たとえば、１つの代替実施形態には、１組のレジスタを含めることができる。この実施形態では、この１組のレジスタ内の各レジスタは、それに記憶されているデータがコミットされているかどうかを識別する表示を含む。

【０１８７】

一実施形態では、プロセッサが浮動小数点モードとパック・データ・モードのどちらかである。プロセッサは、パック・データ・モードではない場合、パック・データ命令を適切に実行することはできず、この逆の場合も同様である。このため、リタイヤメント装置１００６は、バッファ・レジスタに記憶されているデータをコミットする前に、データがパック・データであるかどうかと、プロセッサがパック・データ・モードであるかどうかを判定しておく。データがパック・データであり、プロセッサがパック・データ・モードではない場合、マイクロコードＲＯＭ１０１４に含まれている遷移装置１０３６が呼び出され、パック・データ・モードへの遷移を実行する。一実施形態では、トップ・オブ・スタック表示が初期設定値に（たとえば、レジスタＲ０を示すように）変更されており、かつすべてのタグ１０３４が非空状態であるかどうかを判定することによってプロセッサがパック・データ・モードであるかどうかが判定される。

【０１８８】

プロセッサに、トップ・オブ・スタック表示およびタグ１０３４をポーリングしてプロセッサがパック・データ・モードであるかどうかを判定させる、いくつかの技法がある。たとえば、前述のように、復号装置１００２はマイクロコードＲＯＭ１０１４からマイクロ演算にアクセスする。このようなマイクロ演算は、ＦＰ／ＰＤマッピング装置１０３０が実行する適切なマッピング（たとえば、トップ・オブ・スタック表示を増分することや、トップ・オブ・スタック表示を減分することなど）を識別するコード化フィールドを含む。一実施形態には、パック・データ命令に関するマッピングを識別する少なくとも１つの追加コード化ビット・パターン（本明細書では「パック・データ・ビット・パターン」と呼ぶ）が含まれる。したがって、復号装置１００２がパック・データ命令を受信しマイクロコードＲＯＭ１０１４にアクセスする際、復号装置１００２へ送信される少なくとも１つのマイクロ演算にはパック・データ・ビット・パターンが含まれる。

【０１８９】

ＦＰ／ＰＤマッピング装置１０３０は、パック・データ・ビット・パターンを含むマイクロ演算を受け取ると、１）タグ１０３４およびトップ・オブ・スタック表示の状態を判定し、２）パック・データ・モードへの遷移が必要であるかどうかを示す信号をリタイヤメント装置１００６へ送信する（一実施形態では、プロセッサのモードおよび命令の種類が送信される）。これに応答して、リタイヤメント装置１００６は、命令によって割り振られたバッファ・レジスタに遷移フィールド１０８４内の遷移表示を格納する（一実施形態では、遷移表示は、プロセッサのモードを示す第１のビットと命令の種類を示す第２のビットとを含む）。したがって、命令がパック・データ命令であり、プロセッサがパック・データ・モードではない場合、適切なバッファ・レジスタのモード表示が、遷移が必要であることを示すように変更される。そうでない場合、モード表示は、遷移が必要ではないことを示すように変更される。リタイヤメント・ポインタによって識別されるバッファ・レジスタの準備完了表示が準備完了状態に変更されると、リタイヤメント装置１００６は遷移表示を検査する。遷移表示が、遷移が必要ではないことを示しており、データを他の方法でリタイアできる（たとえば、処理しなければならないイベントがない）場合、データがリタイアされる。逆に、遷移表示が、遷移が必要であることを示している場合、リ

タイヤメント装置１００６は、遷移装置１０３６に関するマイクロコード・エントリ・ポイントをマイクロコードＲＯＭ１０１４へ送信する。これに応答して、マイクロコードＲＯＭ１０１４は、プロセッサをパック・データ・モードへ遷移させるのに必要なマイクロコード演算を送信する。

【０１９０】

このように、パック・データ・モードへの遷移を組み込むには、複雑さをわずかに増大させるだけでよい。もちろん、代替実施形態では、１）復号装置１００２に、名前変更装置１００４にタグおよびトップ・オブ・スタック表示をポーリングさせるパック・データ命令を受け取ったときに特殊な信号を送信させることと、２）タグおよびトップ・オブ・スタックをポーリングすべきであるかどうかを示すビットをすべてのマイクロ演算に追加することと、３）ＦＰ／ＰＤマッピング装置１０３０に、バッファ・レジスタが割り振られるたびにタグおよびトップ・オブ・スタック表示をポーリングさせることと、４）リタイヤメント装置１００６に、パック・データ項目をコミットする準備が完了したときにそのことをＦＰ／ＰＤマッピング装置１０３０に示させ、ＦＰ／ＰＤマッピング装置１０３０に、プロセッサがパック・データ・モードではない場合に遷移装置１０３６を呼び出させることとなどを含め、任意の数の方法でこの機能を実現することができる。一実施形態では、トップ・オブ・スタック表示およびタグ１０３４に基づいて、プロセッサがパック・データ・モードであるかどうかが判定されるが、代替実施形態では、前述のモード表示を含め、任意の数の技法を使用することができる。

【０１９１】

前述のように、遷移装置１０３６は、プロセッサを浮動小数点モードからパック・データ・モードへ遷移させるために使用される。遷移装置１０３６は、プロセッサにトップ・オブ・スタック表示を初期設定値に変更させ、かつすべてのタグ１０３４を非空状態に変更させる。このように、名前変更装置１００４は、パック・データ命令を実行できるように初期設定される。遷移が完了すると、浮動小数点モードからパック・データ・モードへの遷移を生じさせた命令がマイクロ再開される。このため、（オペレーティング・システム・イベント・ハンドラを含む）非マイクロコード・イベント・ハンドラは必要とされず、この実施形態はオペレーティング・システムから見えない。遷移装置１０３６はマイクロコードＲＯＭ１０１４内に配置されているように示されているが、代替実施形態では、遷移装置１０３６をプロセッサ上の任意の位置に配置することができる。他の代替実施形態では、浮動小数点モードからパック・データ・モードへの遷移を実行するように遷移装置１０３６を構成することができる。この遷移時に、遷移装置１０３６は現在のトップ・オブ・スタック表示を記憶域に保存し、トップ・オブ・スタック表示を初期設定値に変更する。遷移装置１０３６は、浮動小数点モードへ遷移するために再び呼び出されると、前のトップ・オブ・スタック表示を復元する。さらに、代替実施形態では、遷移装置１０３６をハードウェア内に実装することも、あるいはプロセッサの外部に記憶された非マイクロコード・イベント・ハンドラとして実施することもできる。

【０１９２】

上記で一実施形態を参照して説明したように、パック・データ命令の各群はＥＭＭＳ命令で終わる。実行装置１０１０は、ＥＭＭＳ命令を実行したことに応答して、名前変更装置１００４にタグ１０３４を空状態に変更させる。したがって、ＥＭＭＳ命令を実行した後、プロセッサは浮動小数点モードになる。すなわち、すべてのタグ１０３４が空状態になり、トップ・オブ・スタック表示が初期設定状態になる（前述のように、トップ・オブ・スタック表示は、パック・データ・モードへの遷移時に初期設定値に変更され、パック・データ命令の実行時には変更されていない）。このため、パック・データ・モードから浮動小数点モードへの遷移を実行するうえで遷移装置は必要とされない。これは、プロセッサを浮動小数点モードとパック・データ・モードとの間で遷移させるために呼び出さなければならない、図６Ａを参照して説明した遷移装置とは異なる。また、浮動小数点状態およびパック・データ状態用に単一のエイリアス化レジスタ・ファイルが使用されるので、２つの別々のレジスタ・ファイル間でデータをコピーするときにこの遷移は必要とされ

ない。このため、回路の複雑さが低減され、プロセッサ上のダイ面積が節約される。

代替実施形態では、タグおよびトップ・オブ・スタック表示の変更は、パック・データ命令の実行時に完全または部分的に実行される。たとえば、遷移装置は、１）ＥＭＭＳ命令ではない各パック・データ命令を実行させ、トップ・オブ・スタック表示を初期設定値に変更し、かつタグを非空状態に変更し、２）ＥＭＭＳ命令を実行させ、タグを空状態に変更することによって不要にすることができる。他の実施形態では、ＥＭＭＳ命令は実施されず、以下で図１４を参照して説明するように浮動小数点命令を使用してエミュレートされる。

【０１９３】

発行装置１００８は、命令およびそのオペランドを記憶するバッファを表す。発行装置１００８は、一連の予約ステーション、中央命令ウィンドウ、またはこの２つを混成させたものとすることができる。予約ステーションを使用する際、各機能装置（たとえば、ＡＬＵ）は、命令とそれに対応するオペランドを識別する情報とを記憶する機能装置自体のバッファを有する。これに対して、中央命令ウィンドウを使用すると、すべての機能装置に共通する中央バッファを使用して、命令とそれに対応するオペランドを識別する情報とが記憶される。命令に対応するオペランドは、どんな情報が使用できるかに応じていくつかの異なる形式であってよい。実データが得られない場合、命令に対応するオペランドは、データの種類と、データがコミットされているかどうかに応じて、１組のＦＰ／ＰＤレジスタ１０２２内のレジスタと、１組の整数レジスタ１０２４内のレジスタと、１組のバッファ・レジスタ１０２０内のレジスタのいずれかを識別する。実データが得られるようになると、そのデータはバッファに記憶される。一実施形態では、発行装置１００８は名前変更装置１００４からの情報も受け取る。しかし、この情報は本発明を理解するうえでは必要とされない。発行装置１００８は、必要な情報が得られたときに実行装置１０１０に命令を発行する。

【０１９４】

実行装置１０１０は命令を実行する。実行装置１０１０は、前述のように、記憶しなければならないオペランド情報を記憶するためにリタイヤメント装置１００６へ送信する。一実施形態では、オペランド情報がないために発行装置１００８内で命令が遅延することがある。実行装置１０１０はまた、任意のオペランド情報を発行装置１００８に送信する。このように、オペランド情報をリタイヤメント装置１００６へ送信し、次いで発行装置１００８へ送信することによって生じる追加遅延が回避される。実行装置１０１０は状態レジスタ１０１２に結合される。状態レジスタ１０１２は、実行装置１０１０が使用する制御情報を記憶する。前述のように、そのような制御情報には、ＥＭ表示とＴＳ表示とを含めることができる。実行装置１０１０は、リタイヤメント装置１００６からアクセスされる様々な種類のデータを整列させるデータ・アラインメント装置１０９０（「ロード／ストア変換装置」とも呼ぶ）を含む。データ・アラインメント装置の動作については、第１２図および図１３を参照して詳しく説明する。

【０１９５】

タグ１０３４の変更は、任意の数の異なる機構を使用して実施することができる。たとえば、図１０は、タグを変更するタグ修飾装置１０９２も含むＦＰ／ＰＤマッピング装置１０３０を示す。タグ修飾装置１０９２は、図６Ｂを参照して説明した方法を含め、任意の数の方法を利用することができる。

【０１９６】

たとえば、一実施形態では、すべてのタグを一度に修正しなくても済むように浮動小数点命令を構成できるので、タグ修飾装置１０９２は、一度にすべてのタグを修正することができないようにされる（そのような１つの実施形態は、上記で図６Ｂを参照して説明したような実施形態である）。回路の複雑さを回避するには、この既存の機構を使用して、パック・データ状態への遷移に応答し、あるいはＥＭＭＳ命令の実行に応答してタグのグローバル変更を実施することができる。なお、ＥＭＭＳ命令を実装するために、ＥＭＭＳ装置１０９４によって表される１組のマイクロコード命令をマイクロコードＲＯＭ１０１

４に記憶することができる。ＥＭＭＳ装置１０９４および遷移装置１０３６内のマイクロコード命令によって、復号装置１００２は、８つのタグのそれぞれを変更するいくつかの既存のマイクロ演算を発行する。したがって、復号装置１００２は、ＥＭＭＳ命令を受け取ったことに応答して、ＥＭＭＳ装置１０９４にアクセスし、いくつかの既存のマイクロ演算を発行する。タグ修飾装置１０９２は、これらのマイクロ演算のそれぞれに応答して、対応するタグを空状態に修正する。これに対して、復号装置１００２は、遷移装置１０３６にアクセスしたことに応答して、タグ修飾装置１０９２に各タグを非空状態に変更させるいくつかの既存のマイクロ演算を発行する。そのような実施形態では、タグをグローバル変更するのに約４クロック・サイクルないし８クロック・サイクルが必要である。

【０１９７】

遷移またはＥＭＭＳ命令に応答してすべてのタグを変更する一実施形態について説明したが、代替実施形態では任意の数の機構を使用することができる。たとえば、すべてのタグの空状態または非空状態への変更は、新しいマイクロ演算を備え、タグ修飾装置１０９２を、新しいマイクロ演算に応答してタグをグローバルに変更できるようにする（タグ修飾装置１０９２のそのような１つの実施形態について図６Ｂを参照して説明する）ことによって単一のクロック・サイクルで完了することができる。この実施形態では、ＥＭＭＳ装置１０９４は、復号装置１００２に、（いくつかの別々のマイクロ演算ではなく）単一のマイクロ演算を発行してすべてのタグを空状態に変更させるようにする。これに対して、遷移装置１０３６は、復号装置１００２に、（いくつかの別々の既存のマイクロ演算ではなく）この単一のマイクロ演算を発行してすべてのタグを非空状態に変更させるようにする。他の例を挙げると、代替実施形態は、実行装置１０１０をタグ１０３４およびリタイヤメント装置１００６に結合するバスを含むことができる。この代替実施形態は、ＥＭＭＳ命令に応答して、プロセッサが直列化され（これは名前変更装置１００４によって実行することができる）、タグを変更させる信号がバス上で送信され（これは、実行装置１０１０によって実施することができる）、プロセッサが再び直列化される（これは名前変更装置１００４によって実行することができる）ようにすることができる。このような実施形態は、すべてのタグを変更するのに約１０クロック・サイクルないし２０クロック・サイクルを必要とする。これに対して、この代替実施形態は、この事前直列化または事後直列化、あるいはその両方が別の装置によって実行され、あるいは不要になるようにすることができる。他の例を挙げると、復号装置１００２は、タグ１０３４に結合することができ、かつＥＭＭＳ命令を受け取ったことに応答してすべてのタグ１０３４を変更する追加ハードウェアを含むことができる。

【０１９８】

したがって、図１０に示した実施形態は、上記で図６Ａを参照して説明したのとは異なり、別々の浮動小数点装置およびパック・データ装置ではなく、浮動小数点命令およびパック・データ命令を実行する１組のレジスタを使用する。また、図６Ａの実施形態は、浮動小数点レジスタにスタックとしてアクセスしパック・データ・レジスタに固定レジスタ・ファイルとしてアクセスする別々の回路を必要とし、それに対して、ＦＰ／ＰＤマッピング装置１０３０は同じ回路を使用する。さらに、プロセッサを浮動小数点モードとパック・データ・モードとの間で遷移させるために呼び出さなければならない、図６Ａを参照して説明した遷移装置とは異なり、図１０を参照して説明した遷移装置は、プロセッサを浮動小数点モードからパック・データ・モードへ遷移させるだけでよい。また、浮動小数点状態およびパック・データ状態用に単一のエイリアス化レジスタ・ファイルが使用されるので、２つの別々のレジスタ・ファイル間でデータをコピーするときにこの遷移は必要とされない。このため、図１０に示した実施形態では、必要とされる回路の複雑さが低減され、プロセッサ上のダイ面積が節約される。

【０１９９】

前述のように、浮動小数点演算およびパック・データ演算を実行する命令を含む一実施形態について説明しているが、代替実施形態は、プロセッサにいくつかの異なるデータ・タイプ演算を実行させる異なる数組の命令を用意することができる。たとえば、ある１組

の命令は、プロセッサにスカラ演算（浮動小数点または整数、あるいはその両方）を実行させることができ、他の１組の命令は、プロセッサにパック演算（浮動小数点または整数、あるいはその両方）を実行させることができる。他の例を挙げると、ある１組の命令は、プロセッサに浮動小数点演算（スカラまたはパック、あるいはその両方）を実行させることができ、他の１組の命令は、プロセッサに整数演算（スカラまたはパック、あるいはその両方）を実行させることができる。他の例を挙げると、単一のエイリアス化レジスタ・ファイルをスタック参照レジスタ・ファイルおよびフラット・レジスタ・ファイルとして操作することができる。また、全エイリアス化を行う一実施形態について説明したが、単一の物理レジスタ・ファイルを有する代替実施形態を、部分エイリアス化されたものとして動作するようにすることができる。この場合、単一のエイリアス化物理レジスタ・ファイルにどんなデータを格納すべきかを追跡する何らかの機構（たとえば、テーブル）が必要である。

【０２００】

図１１Ａ、図１１Ｂ、図１１Ｃは、オペレーティング・システムから見えず、良好なプログラミング方法を推進し、図１０のハードウェア構成を使用して実施することができるように、パック・データ命令および浮動小数点命令を単一のエイリアス化レジスタ・ファイル上で実行する、本発明の他の実施形態による方法を示す。この流れ図は、図４Ａないし図４Ｂ、図７Ａないし図７Ｃ、図９、図１０を参照して説明した流れ図に類似している。これらの前述の流れ図を参照して、ステップが変更、移動、かつ／あるいは削除される多数の代替実施形態について説明した。前述の流れ図で実行されるステップと同様な、図１１Ａないし図１１Ｃを参照して説明したステップを、そのような代替実施形態を使用して実施できることを理解されたい。この流れ図はステップ１１００から開始する。フローは、ステップ１１００からステップ１１０２へ移る。

【０２０１】

ステップ１１０２に示したように、１組のビットが命令としてアクセスされ、フローはステップ１１０４に進む。この１組のビットは、命令によって実行される演算を識別する命令コードを含む。したがって、ステップ１１０２は、図４Ａのステップ４０２に類似している。

【０２０２】

一実施形態では、以下のステップはパイプラインの復号段で実行される。

【０２０３】

ステップ１１０４で、命令コードが有効であるかどうかが判定される。命令コードが有効でない場合、フローはステップ１１０６に進む。そうでない場合、フローはステップ１１０８に進む。ステップ１１０４は、図４のステップ４０４に類似している。

【０２０４】

ステップ１１０６で、無効命令コード例外を発生させるべきであることを示す１つまたは複数のイベント信号マイクロ演算が挿入される。イベント信号マイクロ演算は、パイプラインのリタイヤメント段までエラーの処理を回避するために使用される。命令がイベント信号マイクロ演算である場合、その命令は復号段、レジスタ名前変更段、実行段内を流れる。しかし、イベント信号マイクロ演算をリタイヤメント段で受け取ると、バッファ・レジスタの状態はコミットされず、適切なイベントが発生する。イベント信号マイクロ演算は、イベントを生じさせている命令の前に、あるいはその命令の代わりに挿入される。マイクロ演算の使用については、「Ｍｅｔｈｏｄ ａｎｄ Ａｐｐａｒａｔｕｓ ｆｏｒ Ｓｉｇｎａｌｉｎｇ ａｎ Ｏｃｃｕｒｒｅｎｃｅ ｏｆ ａｎ Ｅｖｅｎｔ ｉｎ ａ Ｐｒｏｃｅｓｓｏｒ」（米国特許出願第０８／２０３７９０号、Ｄａｒｒｅｌ Ｄ. Ｂｏｇｇｓ等）に詳しく説明されている。フローは、ステップ１１０６からステップ１１０８へ移る。

【０２０５】

ステップ１１０８で、どんな種類の命令が受信されたかが判定される。命令が浮動小数点命令でも、あるいはパック・データ命令でもない場合、フローはステップ１１１０に進

む。したがって、ステップ１１０６で１つまたは複数のイベント信号マイクロ演算が挿入された場合、フローはステップ１１１０に進む。しかし、命令が浮動小数点命令である場合、フローはステップ１１１２に進む。これに対して、命令がパック・データ命令である場合、フローはステップ１１１４に進む。したがって、ステップ１１０８は、図４Ａのステップ４０８に類似している。

【０２０６】

ステップ１１１０に示したように、プロセッサが命令を実行する。ステップ１１０６で、無効命令コード例外を発生させるべきであることを示す１つまたは複数のマイクロ演算が挿入された場合、マイクロ演算は復号段、レジスタ名前変更段、実行段内を流れる。しかし、イベント信号マイクロ演算がリタイヤメント段に達すると、バッファ・レジスタの状態はコミットされず、無効命令コード例外が発生する。上記で図２のステップ２１５を参照して説明したように、このイベント・ハンドラは、プロセッサにメッセージを表示させ、現在のタスクの実行を中止させ、次いで他のタスクを実行させるようにすることができる。もちろん、代替実施形態では、前述の任意の数の方法でこのハンドラを形成することができる。他の命令の実行は本発明を理解するうえで必要とされないので、本明細書ではこのことについては詳しく説明しない。

【０２０７】

ステップ１１１２に示したように、ＥＭ表示が１に等しいかどうか（前述のソフトウェア規約によれば、浮動小数点装置をエミュレートすべきかどうか）と、ＴＳ表示が１に等しいかどうか（前述のソフトウェア規約によれば、部分コンテキスト切換が行われたかどうか）が判定される。ＥＭ表示またはＴＳ表示、あるいはその両方が１に等しい場合、フローはステップ１１１６に進む。そうでない場合、フローはステップ１１２０に進む。したがって、ステップ１１１２は図４Ａのステップ４１２に類似している。

【０２０８】

ステップ１１１６で、装置使用不能例外を発生させるべきであることを示す１つまたは複数のイベント信号マイクロ演算が挿入される。フローは、ステップ１１１６からステップ１１２０へ移る。

【０２０９】

ステップ１１１４とステップ１１２０の両方に示したように、レジスタ名前変更が実行される。フローは、ステップ１１２０からステップ１１２２へ移る。これに対して、フローはステップ１１１４からステップ１１３４へ移る。一実施形態では、ステップ１１１４およびステップ１１２０はパイプラインの名前変更段で実行される。

【０２１０】

一実施形態では、以下のステップがパイプラインの実行段で実行される。

【０２１１】

ステップ１１２２に示したように、浮動小数点命令が実行される。ステップ１１２２は図４Ｂのステップ４２６に類似している。オペレーティング・システムから見えないようにするために、一実施形態ではまた、必要に応じてタグを変更し、このとき処理できる数値エラーを報告し、他の数値エラーを未処理のままにしておく。前述のように、タグを変更することによって、この実施形態は、対応するタグが非空状態を示す浮動小数点レジスタのみの内容を記憶するオペレーティング・システム技法からは見えないオペレーティング・システムのままとなるようにすることができる。しかし、代替実施形態は、ある種のとの互換性を有するようにすることができる。たとえば、既存のオペレーティング・システムがタグを使用しない場合、タグを実装していないプロセッサでもそのオペレーティング・システムとの互換性を有する。さらに、本発明では数値浮動小数点例外を未処理のままにしておく必要はなく、したがって、数値浮動小数点例外を未処理のままにしない代替実施形態は本発明の範囲内である。フローはステップ１１２２からステップ１１２４へ移る。

【０２１２】

ステップ１１３４で、パック・データ命令がＥＭＭＳ命令であるかどうかが判定される

。したがって、ステップ１１３４は図４Ｂのステップ４３０に類似している。パック・データ命令がＥＭＭＳ命令である場合、フローはステップ１１３６に進む。そうでない場合、フローはステップ１１３８に進む。前述のように、ＥＭＭＳ命令は、浮動小数点タグを初期設定状態に変更するために使用されるが、パック・データ命令を実行した後、または浮動小数点命令を実行してプロセッサを浮動小数点モードに遷移させる前、あるいはその両方に実行すべきである。

【０２１３】

ステップ１１３６に示したように、すべてのタグは空状態に変更される。このように、タグは初期設定されており、浮動小数点命令を実行することが可能である。ステップ１１３６が完了した後、フローはステップ１１４４に進む。ＥＭＭＳ命令を実施しない実施形態では、ステップ１１３４およびステップ１１３６が存在せず、フローはステップ１１１４からステップ１１３８へ移る。

【０２１４】

ステップ１１３８に示したように、パック・データ命令が実行される。このステップ中に、ＦＰレジスタ、またはパック・データが書き込まれるＦＰ／ＰＤレジスタとして働くバッファ・レジスタの符号フィールドおよび指数フィールドに１が格納される。したがって、ステップ１１３８は図４Ｂのステップ４３４、４３６、４３８に類似している。そうすることにより、浮動小数点命令とパック・データ命令の分離を推進することによって良好なプログラミング技法が推進される。しかし、前述のように、代替実施形態ではこの機能の実施を回避することができる。一実施形態では符号フィールドおよび指数フィールドに１が書き込まれるが、代替実施形態では、ＮＡＮ（非数値）または無限を表す値を使用することができる。また、このステップは数値例外を発生させずに実行される。パック・データ命令の実行を試みたためにメモリ・イベントが発生した場合、実行が割込みを受け、イベントが処理される。フローはステップ１１３８からステップ１１４４へ移る。

【０２１５】

一実施形態では、以下のステップはパイプラインのリタイヤメント段で実行される。

【０２１６】

ステップ１１２４で、命令が、装置使用不能例外を示すイベント信号マイクロ演算であるかどうかが判定される。そうである場合、ステップ１１１２で、ＴＳ表示とＥＭ表示のどちらかまたは両方が１に等しいと判定されている。したがって、命令が、装置使用不能例外を示すイベント信号マイクロ演算である場合、フローはステップ１１２６に進む。そうでない場合、フローはステップ１１２８に進む。このように、レジスタ名前変更を使用するプロセッサに装置使用不能例外を組み込むことができる。

【０２１７】

ステップ１１２６で、装置使用不能例外が発生し、対応するイベント・ハンドラが実行される。したがって、ステップ１１２６は図４Ａのステップ４１６に類似している。前述のように、ＥＭ表示およびＴＳ表示を使用して、浮動小数点命令をエミュレートすべきかどうか、あるいは部分コンテキスト切換が行われたかどうか、あるいはその両方を判定するように、このイベント・ハンドラを形成することができる。やはり前述のように、ＥＭ表示およびＴＳ表示を使用することはソフトウェア規約であり、したがって、他の目的に使用することができる。

【０２１８】

ステップ１１４４で、ＥＭ表示が１に等しいかどうかが判定される。したがって、ステップ１１４４は図４Ａのステップ４１４に類似している。ステップ１１４４で、ＥＭ表示が１に等しいと判定された場合、フローはステップ１１２６ではなくステップ１１４６に進む。そうでない場合、フローはステップ１１４８に進む。

【０２１９】

ステップ１１４６で、無効命令コード例外が発生し、適切なイベント・ハンドラが実行される。これは、図１１Ａのステップ１１１０を参照して説明したのと同じ無効命令コード例外である。無効命令コード例外の発生は、図４Ａのステップ４０６で発生した無効命

令コード例外に類似している。上記で図２のステップ２１５を参照して説明したように、このイベント・ハンドラは、プロセッサにメッセージを表示させ、現在のタスクの実行を中止させ、次いで他のタスクを実行させるように構成することができる。もちろん、代替実施形態では、前述の任意の数の方法でこのハンドラを構成することができる。ＥＭが１に等しいときのパック・データ命令の実行の試みを無効命令コード例外に切り換えることによって、この実施形態はオペレーティング・システムからは見えない。

【０２２０】

オペレーティング・システムから見えないようにＥＭ表示を処理する一実施形態について説明したが、代替実施形態では他の技法を使用することができる。たとえば、代替実施形態では、ＥＭ表示が１に等しいときのパック・データ命令の実行の試みに応答して装置使用不能例外と、異なる既存のイベントと、新しいイベントのいずれかを発生させることができる。他の例を挙げると、代替実施形態では、パック・データ命令を実行するときにＥＭ表示を無視することができる。

【０２２１】

ステップ１１４８に示したように、ＴＳ表示が１に等しいかどうか（前述のソフトウェア規約によれば、部分コンテキスト切換が行われたかどうか）が判定される。部分コンテキスト切換が実行された場合、フローはステップ１１２６に進む。そうでない場合、フローはステップ１１５０に進む。

【０２２２】

前述のように、ステップ１１２６で、装置使用不能例外が発生し、対応するイベント・ハンドラが実行される。したがって、このイベントに応答して、ＥＭ表示およびＴＳ表示をポーリングするように、この対応するイベント・ハンドラを構成することができる。しかし、パック・データ命令を実行すると、フローがステップ１１４４に進み、ＥＭ表示が１に等しい状況が無効命令コード例外に切り換えられる。このため、パック・データ命令を実行中であり、ステップ１１２６に達したときに、ＥＭ表示は０に等しくなければならず、ＴＳ表示は１に等しくなければならない。ＴＳ表示が１に等しいので、イベント・ハンドラは、上記で部分コンテキスト切換に関して説明したように機能し、プロセッサに、ステップ１１０２で受け取った命令の実行を再開することによって実行を再開させる。パック・データ状態は浮動小数点状態上にエイリアス化されているので、このイベント・ハンドラは浮動小数点状態とパック・データ状態の両方のために働く。このため、この方法はオペレーティング・システムからは見えない。もちろん、代替実施形態ではこのイベント・ハンドラを前述の任意の数の方法で実行できるようにすることができる。オペレーティング・システムから見えないようにＴＳ表示を処理する一実施形態について説明したが、代替実施形態では前述の他の技法を使用することができる。

【０２２３】

前述のように、浮動小数点命令の実行時にある数値エラーが発生した場合、そのようなエラーは、次の浮動小数点命令の実行が試みられるまで未処理のままにされ、次の浮動小数点命令の実行に割り込みこのようなエラーを処理することができる。ステップ１１２８とステップ１１５０の両方に示したように、処理できるそのような未処理のエラーがあるかどうかが判定される。したがって、これらのステップは、図４Ａのステップ４２０およびステップ４２２に類似している。そのような未処理のエラーがある場合、フローはステップ１１２８とステップ１１５０の両方からステップ１１３０へ移る。しかし、ステップ１１２８で、そのような未処理のエラーがないと判定された場合、フローはステップ１１３２に進む。逆に、ステップ１１５０で、そのような未処理のエラーがないと判定された場合、フローはステップ１１５２に進む。代替実施形態では、ステップ１１５０は実行されず、パック・データ命令を実行する間、浮動小数点エラーは未処理のままにされる。

【０２２４】

ステップ１１３０で、未処理浮動小数点エラー・イベントが発生する。したがって、ステップ１１３０は図４Ａのステップ４２４に類似している。上記で図２のステップ４２４を参照して説明したように、このイベントを内部イベントまたは外部イベントとみなし、

それに応じて処理することができる。

【０２２５】

ステップ１１５２に示したように、プロセッサがパック・データ・モードであるかどうかが判定される。プロセッサがパック・データ・モードである場合、パック・データ命令の実行は首尾良く完了しており、フローはステップ１１３２に進む。しかし、プロセッサがパック・データ・モードではない場合、パック・データ命令は浮動小数点モードで実行されている。このため、このパック・データ命令の実行は正確なものではない。これを補正するには、プロセッサを浮動小数点モードからパック・データ・モードに切り換え、パック・データ命令を再実行しなければならない。これを行うために、プロセッサがパック・データ・モードではない場合、フローはステップ１１５４に進む。ステップ１１５２の判定は任意の数の方法で行うことができる。たとえば、上記で図６Ａを参照して説明したモード表示を使用することができる。他の例を挙げると、トップ・オブ・スタック表示およびタグをポーリングすることができる。トップ・オブ・スタック表示が初期設定状態であり、すべてのタグが非空状態である場合、プロセッサはパック・データ・モードである。しかし、トップ・オブ・スタック表示が初期設定状態ではなく、あるいはすべてのタグが非空状態であるわけではない場合、プロセッサはパック・データ・モードではない。

【０２２６】

ステップ１１５４で、プロセッサは浮動小数点モードからパック・データ・モードへ遷移し、フローはステップ１１５６に進む。ステップ１１５４で、プロセッサは、すべてのタグを非空状態に変更し、トップ・オブ・スタック表示を初期設定値に変更することによって浮動小数点モードからパック・データ・モードへ遷移する。すべてのタグを非空状態に変更すると、浮動小数点命令とパック・データ命令が別々にグループ化されるので、良好なプログラミング技法が推進される。また、オペレーティング・システムとの互換性の点では、ある種のオペレーティング・システム技法は、対応するタグが非空状態を示す浮動小数点レジスタのみの内容を記憶する。したがって、パック・データ状態が浮動小数点状態上にエイリアス化される実施形態では、すべてのタグを非空状態に変更することによって、そのようなオペレーティング・システムは浮動小数点状態の場合と同様にパック・データ状態を保存する。代替実施形態は、このようなオペレーティング・システム技法のいくつかとの互換性を有するようにすることができる。たとえば、あるオペレーティング・システムがタグを使用しない場合、タグを実装しない実施形態でもそのオペレーティング・システムとの互換性を有する。トップ・オブ・スタック表示を零に変更することを利用して、前述のように効率的なプログラミング技法が行われる。また、トップ・オブ・スタック表示を初期設定値に変更し、パック・データ命令の実行時にはトップ・オブ・スタック表示を変更しないことにより、上記で図１０を参照して説明したように、同じ回路を使用してＦＰ／ＰＤレジスタを浮動小数点スタックおよび固定レジスタ・ファイルとして操作することができる。浮動小数点状態およびパック・データ状態が単一のレジスタ・ファイル上にエイリアス化されるので、遷移時に、別々の浮動小数点レジスタ・ファイルとパック・データ・レジスタ・ファイルとの間でデータをコピーする必要はない。これにより、浮動小数点モードとパック・データ・モードとの間の遷移に必要な時間が短縮する。前述のように、浮動小数点からパック・データへの遷移はマイクロコード内に備えることができる。代替実施形態では、各パック・データ命令を実行することによって、トップ・オブ・スタック表示が初期設定値に変更される。

【０２２７】

ステップ１１５６に示したように、マイクロ再開を実行することによって、ステップ１１０２で受け取った命令が再開される。マイクロ再開を使用するので、プロセッサの外部での処置を施さずに現在のタスクの実行を再開することができる。すなわち、非マイクロコード・イベント・ハンドラを実行する必要はない。このように、この実施形態は既存のオペレーティング・システムとの互換性を有する。代替実施形態は、より低い互換性を有するようにすることができる。たとえば、プロセッサに追加イベントを組み込むことができ、この遷移を実行する追加イベント・ハンドラをオペレーティング・システムに追加す

ることができる。

【０２２８】

ステップ１１３２で、バッファ・レジスタの状態が、バッファ・レジスタに対応するＦＰ／ＰＤレジスタまたは整数レジスタにコミットされる。ステップ１１３２が完了すると、プロセッサは実行を継続することができる。

【０２２９】

したがって、既存のオペレーティング・システムとの互換性を有し、良好なプログラミング技法を推進するパック・データ命令を実行する方法について説明した。パック・データ状態は、浮動小数点状態上にエイリアス化されるので、浮動小数点状態の場合と同様に既存のオペレーティング・システムによって保存され復元される。さらに、パック・データ命令を実行することによって発生したイベントを既存のオペレーティング・システム・ハンドラによって処理することができるので、このようなイベント・ハンドラを修正する必要がなく、新しいイベント・ハンドラを追加する必要がない。このため、プロセッサは後方互換性を有し、更新時に、オペレーティング・システムを開発または修正するために必要なコストおよび時間が必要とされない。

【０２３０】

そのうちのいくつかについて説明したこの実施形態の変形形態は、そのようなオペレーティング・システムとの全互換性または部分互換性を有し、かつ／あるいは良好なプログラミング技法を推進することができる。たとえば、代替実施形態では、この流れ図の１つまたは複数のステップを移動し、変更し、かつ／あるいは削除することができる。図１１Ａまたは図１１Ｂまたは図１１Ｃ、あるいはそれらの組合せからあるステップを削除する場合、図１０中のあるハードウェアは必要とされない。たとえば、ＴＳ表示を使用しない場合、ＴＳ表示は必要とされない。もちろん、本発明は任意の数のシステム・アーキテクチャに有用であり、本明細書で説明するアーキテクチャに限らない。

【０２３１】

図１２Ａ、図１２Ｂ、図１２Ｃは、図１０を参照して説明した実施形態によって浮動小数点データ、パック・データ、整数データを記憶するための記憶フォーマットを示す。もちろん、代替実施形態では、浮動小数点データ、パック・データ、整数データを記憶するためにいくつかの異なる記憶フォーマットを使用することができる。

【０２３２】

図１２Ａは、図１０を参照して説明した本発明の一実施形態による浮動小数点記憶フォーマットを示す。図１２Ａは、ビット８５からなる符号フィールド１２０２と、ビット［８４：６８］からなる指数フィールド１２０４と、ビット［６７：３］からなる小数部フィールド１２０６と、ビット［２：０］からなる丸めフィールド１２０８とを含む浮動小数点記憶フォーマット１２００を示す。前述のように、タスク切換を実行する際にメモリに浮動小数点状態を記憶するために使用されるのと同じ浮動小数点命令を使用して、浮動小数点レジスタ上にエイリアス化されたパック・データ状態も記憶しなければならない。一実施形態では、プロセッサは丸めビットを丸めフィールド１０２８には記憶しない。このため、パック・データは、浮動小数点記憶フォーマット１２００の小数部フィールド１２０６内のどこかに記憶しなければならない。

【０２３３】

図１２Ｂは、図１０を参照して説明した本発明の一実施形態によるパック・データ用の記憶フォーマットを示す。図１２Ｂは、ビット［８５：６８］からなる符号／指数フィールド１２１２と、ビット［６７］からなる第１の予約フィールド１２１４と、ビット［６６：３］からなるパック・データ・フィールド１２１６と、ビット［２：０］からなる第２の予約フィールド１２１８とを含むパック・データ記憶フォーマット１２１０を示す。前述のように、パック・データがレジスタに書き込まれる際、符号／指数フィールド１２１２にすべて１が格納される。やはり前述のように、パック・データ・フィールド１２１６は、既存の浮動小数点命令によってパック・データ状態が記憶されるように小数部フィールド１２０６上にエイリアス化される。一実施形態では、パック・データがレジスタに

書き込まれる際、第１および第２の予約フィールド１２１４および１２１８に零が書き込まれる。パック・データ記憶フォーマット１２１０のパック・データ・フィールド１２１６が浮動小数点記憶フォーマット１２００の小数部フィールド１２０６と同じビット位置から始まる本発明の一実施形態について説明したが、代替実施形態では、この関係を変更することができる。

【０２３４】

図１２Ｃは、図１０を参照して説明した本発明の実施形態による整数データ用の記憶フォーマットを示す。図１２Ｃは、ビット［８５：３２］からなる予約フィールド１２２２と、ビット［３１：０］からなる整数データ・フィールド１２２４とを含む整数データ記憶フォーマット１２２０を示す。整数データが３２ビットとして記憶される一実施形態について説明したが、代替実施形態は、任意の数のビットを使用した１つまたは複数のフォーマットで整数データを記憶するようにすることができる。たとえば、代替実施形態では６４ビット・フォーマットをサポートすることができる。一実施形態では、ソフトウェアから見える各整数レジスタ１０２４は３２ビットのみを含む。このため、整数記憶フォーマット１２２０はバッファ・レジスタ１０２０でのみ使用される。

【０２３５】

図１３は、図１２Ａ、図１２Ｂ、図１２Ｃを参照して説明した記憶フォーマットが使用されたときに、図１１Ｂのステップ１１３８を実行する、本発明の一実施形態による方法を示す。フローは、ステップ１１３４からステップ１３００へ移る。

【０２３６】

ステップ１３００で、パック・データ命令が、ＦＰ／ＰＤレジスタ、またはＦＰ／ＰＤレジスタとして働くバッファ・レジスタからパック・データを検索するかどうかが判定される。そうである場合、フローはステップ１３０２に進む。そうでない場合、フローはステップ１３０４に進む。

【０２３７】

ステップ１３０２に示したように、エイリアス化バッファ・レジスタまたはＦＰ／ＰＤレジスタのビット［６６：３］が検索され、フローはステップ１３０４に進む。このステップが必要であるのは、図１２Ｂに示したように、パック・データがビット零からではなくビット３から記憶されるからである。このため、ビット［２：０］を破棄しなければならない。一実施形態では、このステップは図１０のデータ・アラインメント装置１０９０によって実行される。この実施形態では、データは、図１２Ｂに示したフォーマットで、リタイヤメント装置１００６から発行装置１００８を通して実行装置１０１０へ転送される。したがって、データは、実行装置１０１０によって、図１２Ｂに示したフォーマットで受信され、データ・アラインメント装置１０９０がイネーブルされ、ビット［６６：３］を抽出する。図１０は単一のデータ・アラインメント装置を示しているが、一実施形態では、パック・データを処理する実行装置１０１０内の各機能単位が、ビット［６３：３］を抽出するデータ・アラインメント装置を含む。データは実行装置１０１０内で整列するので、パック・データ・フォーマットの使用はプロセッサの残りの部分には透過的である。データ・アラインメント装置は、任意の数の技法を使用してビット［６６：３］にアクセスするようにすることができる。たとえば、一実施形態では、データ・アラインメント装置は、ＦＰ／ＰＤレジスタ、またはＦＰ／ＰＤレジスタとして働くバッファ・レジスタから検索されたすべてのパック・データを３ビットだけ右にシフトさせるように設計される。代替実施形態では、リタイヤメント装置または発行装置を、ビット［２：０］またはビット［８５：６７］、あるいはその両方を除去するようにすることができる。他の例を挙げると、パック・データがビット零から記憶されるように代替実施形態を構成することができる。

【０２３８】

ステップ１３０４で、パック・データ命令が整数レジスタ、または整数レジスタとして働くバッファ・レジスタからパック・データを検索するかどうかが判定される。そうである場合、フローはステップ１３０６に進む。そうでない場合、フローはステップ１３０８

に進む。

【０２３９】

ステップ１３０６に示したように、エイリアス化バッファ・レジスタまたは整数レジスタのビット［３１：０］が検索され、フローはステップ１３０８に進む。このステップが必要であるのは、データがビット零から記憶されるからである。前述のように、一実施形態では、このステップは図１０のデータ・アラインメント装置１０９０によって実行される。この実施形態では、データはリタイヤメント装置１００６から発行装置１００８を通して実行装置１０１０へ転送される。データは、バッファ・レジスタ１０２０からアクセスされた場合は、実行装置１０１０によって、図１２Ｃに示したフォーマットで受信され、データ・アラインメント装置がイネーブルされ、ビット［３１：０］を抽出する。しかし、データは、整数レジスタ１０２４が３２ビット・レジスタである実施形態で整数レジスタ１０２４からアクセスされる場合は、実行装置１０１０によって３２ビット・フォーマットで受信される。いずれの場合も、３２ビット・データを６４ビットのパック・データ項目の一部とみることができる。たとえば、第１の移動命令を実施して整数レジスタの３２ビットをパック・データ項目の上位ビットへ移動し、同時に、第２の移動命令を実施して整数レジスタの３２ビットをパック・データ項目の下位３２ビットへ移動することができる。

【０２４０】

ステップ１３０８に示したように、命令が必要とする動作が実行され、フローはステップ１３１０に進む。

【０２４１】

ステップ１３１０で、パック・データ命令によって、プロセッサが、ＦＰ／ＰＤレジスタ、またはＦＰ／ＰＤレジスタとして働くバッファ・レジスタに書込みを行うかどうかが判定される。そうである場合、フローはステップ１３１２に進む。そうでない場合、フローはステップ１３１４に進む。

【０２４２】

パック・データ命令によって、プロセッサが、ＦＰ／ＰＤレジスタ、またはＦＰ／ＰＤレジスタとして働くバッファ・レジスタに書込みを行う場合、データを固有のフォーマットで記憶しなければならない。したがって、ステップ１３１２で、パック・データは、ＦＰ／ＰＤレジスタまたはバッファ・レジスタのビット［６６：３］に記憶される。一実施形態では、図１０のデータ・アラインメント装置１０９０が再び使用される。この場合も、この機能を実行するいくつかの技法がある。たとえば、データを３ビットだけ左にシフトさせ、ビット［２：０］に零を埋め込み、ビット［６７］に零を埋め込み、ビット［８５：６８］に１を記憶するように、データ・アラインメント装置を構成することができる。代替実施形態では、データをこのフォーマットで記憶するようにリタイヤメント装置を構成することができる。

【０２４３】

ステップ１３１４で、パック・データ命令によって、プロセッサが、整数レジスタ、または整数レジスタとして働くバッファ・レジスタに書込みを行うかどうかが判定される。そうである場合、フローはステップ１３１６に進む。そうでない場合、フローはステップ１１４４に進む。

【０２４４】

パック・データ命令によって、プロセッサが、整数レジスタ、または整数レジスタとして働くバッファ・レジスタに書込みを行う場合、パック・データを固有の整数記憶フォーマットで記憶しなければならない。したがって、ステップ１３１６で、データは、整数レジスタ内ではビット［３１：０］として存在し、バッファ・レジスタ内では（実施態様に応じて）ビット［６３：０］または［３１：０］として存在する。６４ビットのデータがあるので、３２ビット分のデータをこれらのレジスタに記憶することができる。たとえば、第１の移動命令は、パック・データ項目の上位ビットを整数レジスタへ移動し、同時に、第２の移動命令を実施してパック・データ項目の下位３２ビットを整数レジスタへ移動

するように構成しても良い。一実施形態では、このステップはやはり図１０のデータ・アラインメント装置１０９０によって実行される。もちろん、前述の技法を含め、任意の数の技法を使用してステップ１３１６を実施することができる。

【０２４５】

このように、いくつかの異なる種類のデータによって使用される記憶フォーマットはプロセッサのレジスタ内で固有に整列される。一実施形態では、ＦＰ／ＰＤレジスタ１０２２で使用されるバッファ・レジスタ１０２０と整数レジスタ１０２４で使用されるバッファ・レジスタ１０２０で同じ記憶フォーマットが使用される。もちろん、代替実施形態では、任意の数の異なる記憶フォーマットを使用することができ、したがって、そのような代替実施形態は依然として本発明の範囲内である。たとえば、一代替実施形態は、これらのデータ記憶フォーマットを１組のバッファ・レジスタ１０２０で使用し、ソフトウェアから見えるレジスタ（たとえば、ＦＰ／ＰＤレジスタ１０２２および整数レジスタ１０２４）では異なるデータ記憶フォーマットを使用する。

【０２４６】

前述のように、浮動小数点モードとパック・データ・モードとの間の遷移は時間がかかることがあり、効率的なプログラミング方法ではない。プログラマが多くのそのような遷移を実行するかどうかを判定するのを助けるには、異なる性能監視技法を使用することができる。たとえば、一実施形態では、性能モニタ・カウンタが使用される。性能モニタ・カウンタは、プログラマから見ることができ、プロセッサにおいていくつかの異なる条件が満たされる回数をカウントする。本発明の一実施形態では、これらの条件の１つは浮動小数点モードとパック・データ・モードとの間の遷移である。このように、プログラマはプログラムにいくつの遷移が必要であるかを知ることができる。プログラム・カウンタに関する詳細については、「Ａｐｐａｒａｔｕｓ ｆｏｒ Ｍｏｎｉｔｏｒｉｎｇ ｔｈｅ Ｐｅｒｆｏｒｍａｎｃｅ ｏｆ ａ Ｐｒｏｃｅｓｓｏｒ」（米国特許出願第０７／８８３８４５号、Ｒｏｂｅｒｔ Ｓ．Ｄｒｅｙｅｒ等）を参照されたい。

【０２４７】

従来技術の浮動小数点プロセッサでは浮動小数点タグを直接処理することができないので、浮動小数点命令を使用したＥＭＭＳ命令のエミュレーションを実行することができる。

【０２４８】

図１４は、本発明の一実施形態による、タグをクリアする方法を示す流れ図である。この流れ図は、ステップ１４０２で浮動小数点環境をメモリ内の所定の位置に記憶することによって開始する。これは、インテル・アーキテクチャ・プロセッサではＦＮＳＡＶＥ命令またはＦＳＡＶＥ命令を使用して行われる。これを実行した後、ステップ１４０４で、環境が記憶された所定のメモリ位置のタグまたはＴＯＳ部分、あるいはその両方をその空状態に修正することができる。これは、このタグおよびＴＯＳビットの適切なビット・パターンに関する即値オペランドを有するＭＯＶ命令を含め任意の数の従来型の命令を使用して行われる。所定のメモリ位置のタグおよびＴＯＳ部分を空状態に設定する他の適切な命令を使用することができる。これに続いてステップ１４０６で、修正された所定のメモリ位置から環境を再ロードすることができる。環境の他の部分（制御語、状態語など）は未修整のままにすべきであり、浮動小数点タグが修正されるに過ぎないので、環境の残りの部分は環境記憶動作１４０２から変わらない。さらに、予想されない割込みが行われるのを防止するために、割込みをディスエーブルする命令（たとえば、ＦＮＳＴＥＮＶ）の使用を含む既知の従来型の技法を使用してプロセスのこの実施形態を実行することができる。いずれにしても、この場合、ＦＲＳＴＯＲやＦＬＤＥＮＶなど従来技術の技法を使用して環境を再ロードするので、環境には、空状態に修正された浮動小数点タグしか再ロードされていない。さらに、トップ・オブ・スタック・フィールド３５０に格納されたトップ・オブ・スタック表示を含む浮動小数点環境の部分をクリアする追加ステップをステップ１４０４に含めることができることに留意されたい。

【０２４９】

他の代替実施形態では、すべてのタグ・ビットが空になるまで浮動小数点レジスタを十分な回数だけポップすることによってＥＭＭＳ命令をエミュレートすることができる。いずれの場合も、ＥＭＭＳは専用命令として実行することも、あるいはエミュレートすることもでき、どちらの方法も本開示の教示の範囲内である。

【０２５０】

図１５Ａは、エイリアス化された別々の物理レジスタ・ファイルを更新することのできる時間間隔を示すために、パック・データ命令と浮動小数点命令とを含む実行ストリームを示している。図１５Ａは、浮動小数点命令１５００と、その後に続く１組のパック・データ命令１５１０を示している。図１５Ａはさらに、浮動小数点命令１５００が時間Ｔ１に実行され、それに対して、１組のパック・データ命令１５１０の実行が時間Ｔ２から開始することも示している。浮動小数点命令１５００を実行すると、プロセッサは浮動小数点レジスタに値を書き込む。間隔１５２０は、この値をエイリアス化しなければならない時間Ｔ１と時間Ｔ２との間の時間を示す。たとえば、別々の物理レジスタ・ファイルを使用して浮動小数点命令とパック・データ命令が実行される、図６Ａないし図９を参照して説明した一実施形態では、浮動小数点状態は、時間Ｔ２になって初めて、物理浮動小数点レジスタから対応する物理パック・データ・レジスタにコピーされる（時間Ｔ２よりも前に同じ浮動小数点レジスタに他の値が書き込まれることがないと仮定する）。これに対して、単一物理レジスタ・ファイルを使用すると（図１０ないし図１１Ｃを参照して説明した実施形態）、浮動小数点値は時間Ｔ１にエイリアス化レジスタに記憶される。

【０２５１】

したがって、間隔１５２０の２つの極値について説明した。しかし、代替実施形態は、間隔１５２０中の任意の時間にレジスタをエイリアス化するようにすることができる。たとえば、別々の物理レジスタ・ファイルを使用して浮動小数点命令およびパック・データ命令を実行する代替実施形態は、時間Ｔ１に、浮動小数点物理レジスタ・ファイルに書き込まれるデータがパック・データ物理レジスタ・ファイルにも書き込まれるようにすることができる。値を両方の物理レジスタ・ファイルに同時に（たとえば、時間Ｔ１に）書き込む一実施形態では、データを浮動小数点レジスタからパック・データ・レジスタにコピーする遷移の部分をハードウェアとして実施することができる（もちろん、代替実施形態はソフトウェア、またはファームウェア、またはハードウェア、あるいはそれらの組合せを使用することができる）。他の例を挙げると、別々の物理レジスタ・ファイルを使用して浮動小数点命令およびパック・データ命令を実行する代替実施形態は、間隔１５２０中に（ただし、時間Ｔ２よりも前のある時点に）自由な処理時間があるときに、浮動小数点物理レジスタ・ファイルに書き込まれたデータがパック・データ物理レジスタ・ファイルに書き込まれるようにすることができる。このように、これらの実施形態は遷移時間を短縮させることができる。

【０２５２】

図１５Ｂは、エイリアス化された別々の物理レジスタ・ファイルを更新することのできる時間間隔を示すために、パック・データ命令と浮動小数点命令とを含む実行ストリームを示している。図１５Ａは、パック・データ命令１５３０の後に１組の浮動小数点命令１５４０が続くことを除いて図１５Ｂに類似している。図１５Ｂは、パック・データ命令１５３０が時間Ｔ１に実行され、それに対して、１組の浮動小数点命令１５４０の実行が時間Ｔ２から開始することを示している。パック・データ命令１５３０を実行すると、プロセッサはパック・データ・レジスタに値を書き込む。間隔１５５０は、この値をエイリアス化しなければならない時間Ｔ１と時間Ｔ２との間の時間を示す。（浮動小数点命令とその後に続くパック・データ命令に関して）図１５Ａを参照して説明したすべての代替実施形態を図１５Ｂに関して（パック・データ命令とその後に続く浮動小数点命令に関して）実施することもできる。

【０２５３】

本発明をいくつかの実施形態に関して説明したが、当業者には、本発明が前述の実施形態に限らないことが認識されよう。本発明の方法および装置は、添付の請求の範囲の趣旨

および範囲内で修正および変更を加えて実施することができる。したがって、この説明は本発明を制限するものではなく例示的なものとみるべきである。

【図面の簡単な説明】

【０２５４】

【図１】Ｐｅｎｔｉｕｍプロセッサを使用した例示的なコンピュータ・システムを示すブロック図である。

【図２】Ｐｅｎｔｉｕｍプロセッサによる命令の実行を示す流れ図である。

【図３】Ａ：本発明の一実施形態によるパック・データ状態および浮動小数点状態のエイリアス化を示す機能図、ＢおよびＣ：物理浮動小数点レジスタおよびパック・データ・レジスタの論理浮動小数点レジスタに対するマッピングを示す図、Ｄ：パック・データ命令と浮動小数点命令とを含む実行ストリームを示す図である。

【図４Ａ】既存のソフトウェアとの互換性を有し、様々なオペレーティング・システム技法には見えず、効率的なプログラミング技法を推進するように、浮動小数点命令およびパック・データ命令を実行する、本発明の一実施形態による方法の一部を示す流れ図である。

【図４Ｂ】図４Ａに部分的に示した方法の残りの部分を示す流れ図である。

【図５】本発明の一実施形態による例示的なコンピュータ・システムを示すブロック図である。

【図６Ａ】本発明の一実施形態による、２つの物理レジスタ・ファイルを使用して浮動小数点状態上にパック・データ・レジスタ状態をエイリアス化する装置を示すブロック図である。

【図６Ｂ】本発明の一実施形態による、図６Ａの浮動小数点スタック参照ファイルの一部の拡大図を示すブロック図である。

【図７Ａ】既存のソフトウェアとの互換性を有し、様々なオペレーティング・システム技法には見えず、良好なプログラミング方法を推進し、図６Ａのハードウェア構成を使用して実行することができるように、１組の浮動小数点レジスタ上にエイリアス化された１組のレジスタ上でパック・データ命令を実行するための、本発明の一実施形態による方法の一部を示す流れ図である。

【図７Ｂ】図７Ａに部分的に示した方法の他の部分を示す流れ図である。

【図７Ｃ】図７Ａおよび図７Ｂに部分的に示した方法の残りの部分を示す流れ図である。

【図８】本発明の一実施形態による、図７Ｃのステップ７３４を実行する方法を示す流れ図である。

【図９】本発明の一実施形態による、図７Ｂのステップ７２８を実行する方法を示す流れ図である。

【図１０】本発明の他の実施形態による、単一のレジスタ・ファイルを使用して浮動小数点状態上にパック・データ状態をエイリアス化する装置内のデータ・フローを示すブロック図である。

【図１１Ａ】既存のソフトウェアとの互換性を有し、様々なオペレーティング・システム技法には見えず、良好なプログラミング方法を推進し、図１０のハードウェア構成を使用して実行することができるように、単一のエイリアス化レジスタ・ファイルに対してパック・データ命令および浮動小数点命令を実行するための、本発明の他の実施形態による方法の一部を示す図である。

【図１１Ｂ】図１１Ａに部分的に示した方法の他の部分を示す流れ図である。

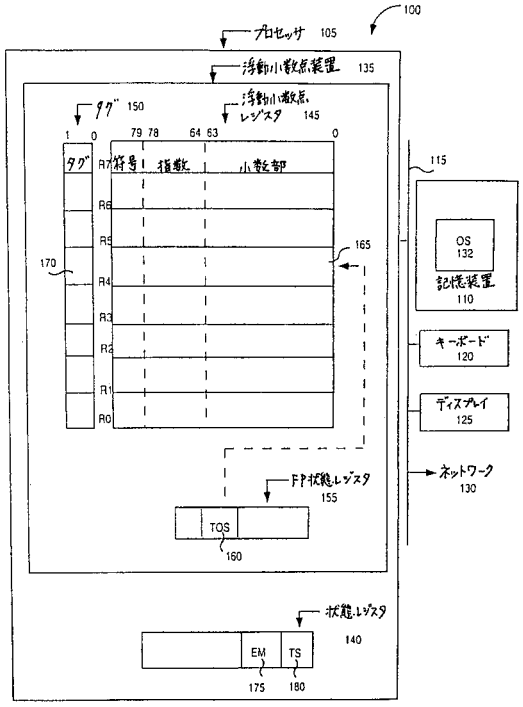
【図１１Ｃ】図１１Ａおよび図１１Ｂに部分的に示した方法の残りの部分を示す流れ図である。

【図１２】Ａ：図１０を参照して説明する本発明の一実施形態による浮動小数点格納フォーマットを示す図、Ｂ：図１０を参照して説明する本発明の一実施形態によるパック・データの格納フォーマットを示す図、Ｃ：図１０を参照して説明する本発明の一実施形態による整数データの格納フォーマットである。

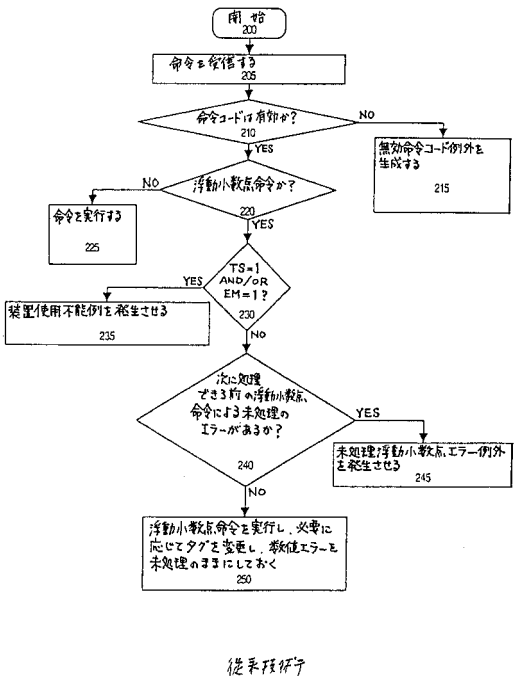
【図１３】図１２Ａ、図１２Ｂ、図１２Ｃを参照して説明する格納フォーマットを実施す

る際に図１１Ｂのステップ１１３８を実行する、本発明の一実施形態による方法を示す図である。
【図１４】本発明の一実施形態による、タグをクリアする方法を示す流れ図である。
【図１５】Ａ：エイリアス化された別々の物理レジスタ・ファイルを更新することのできる時間間隔を示すために、パック・データ命令と浮動小数点命令とを含む実行ストリームを示す図で、Ｂ：エイリアス化された別々の物理レジスタ・ファイルを更新することのできる時間間隔を示すために、パック・データ命令と浮動小数点命令とを含む他の実行ストリームを示す図である。

【図１】

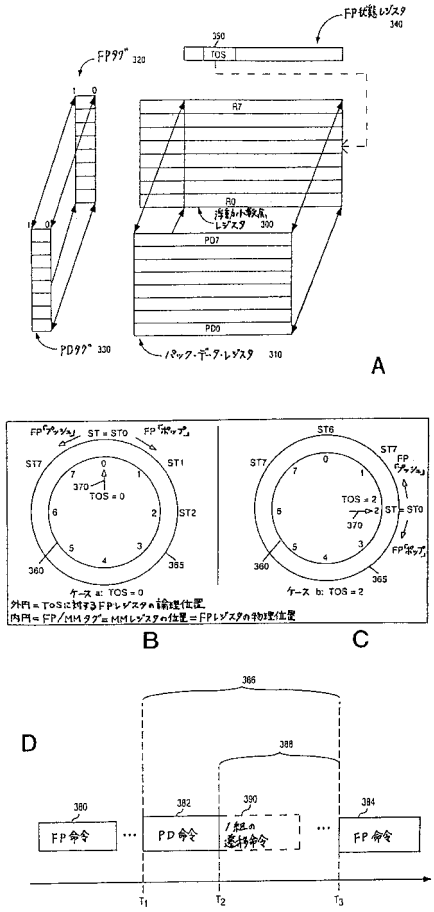


【図２】

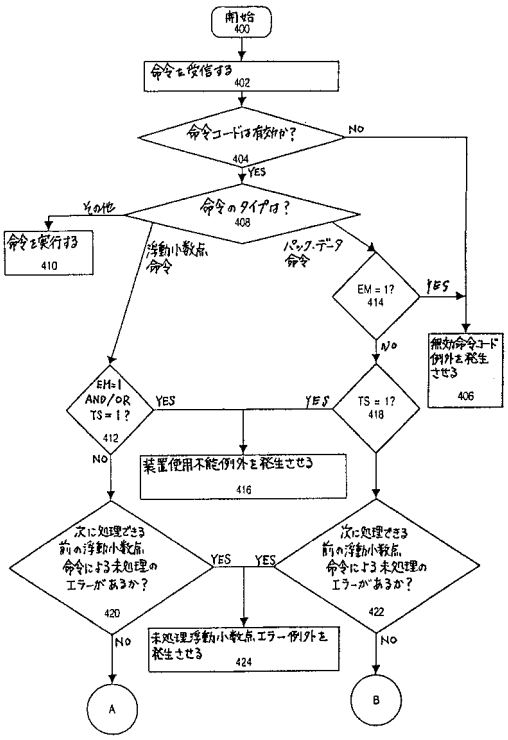


従来技術

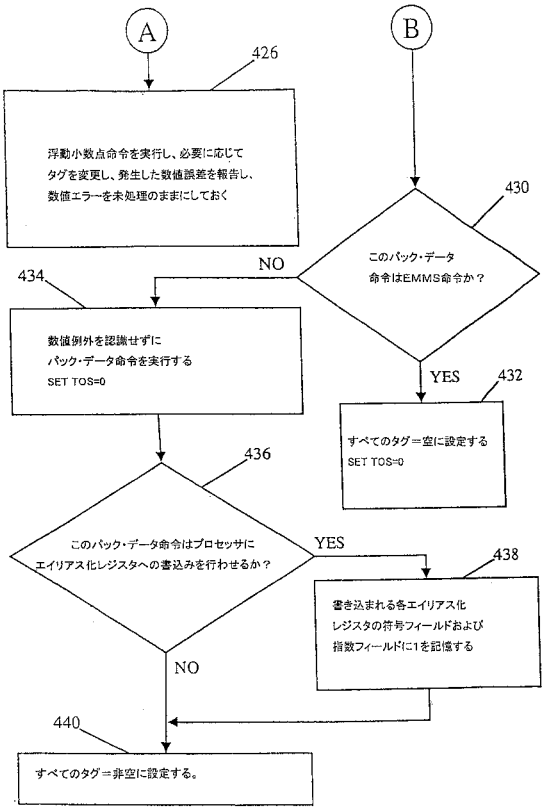
【図３】


FP状態レジスタ
340
350
TOS
FPタグ
320
R7
R0
浮動小数点
レジスタ
300
PD7
PD0
PDタグ
330
パック・データ・レジスタ
310
A
FP「プッシュ」
ST = ST0
FP「ポップ」
ST7
ST1
ST2
ST6
TOS = 0
TOS = 2
370
360
365
ケース a: TOS = 0
ケース b: TOS = 2
外円 = TOSに対するFPレジスタの論理位置
内円 = FP/MMタグ = MMレジスタの位置 = FPレジスタの物理位置
B
C
D
366
388
380
382
390
384
FP命令
PD命令
1組の
選択命令
T1
T2
T3

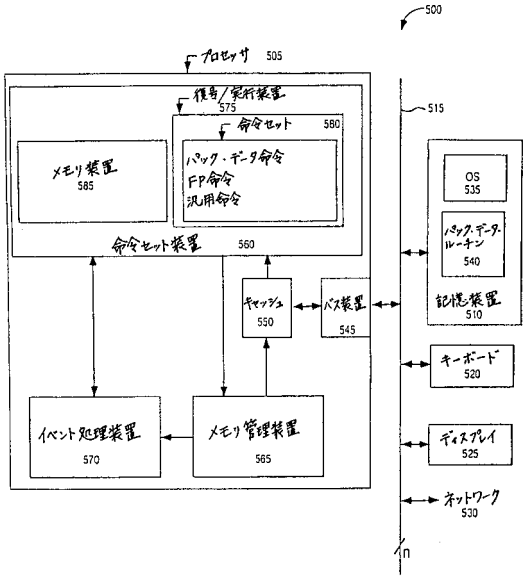
【図４Ａ】


開始
400
命令を受信する
402
命令コードは有効か?
404
No
Yes
命令のタイプは?
408
その他
命令を実行する
410
浮動小数点命令
パック・データ命令
EM = 1?
414
Yes
No
無効命令コード例外を発生させる
406
EM=1 AND/OR TS=1?
412
Yes
TS = 1?
418
装置使用不能例外を発生させる
416
次に処理できる前の浮動小数点命令による未処理のエラーがあるか?
420
次に処理できる前の浮動小数点命令による未処理のエラーがあるか?
422
未処理浮動小数点エラー例外を発生させる
424
A
B

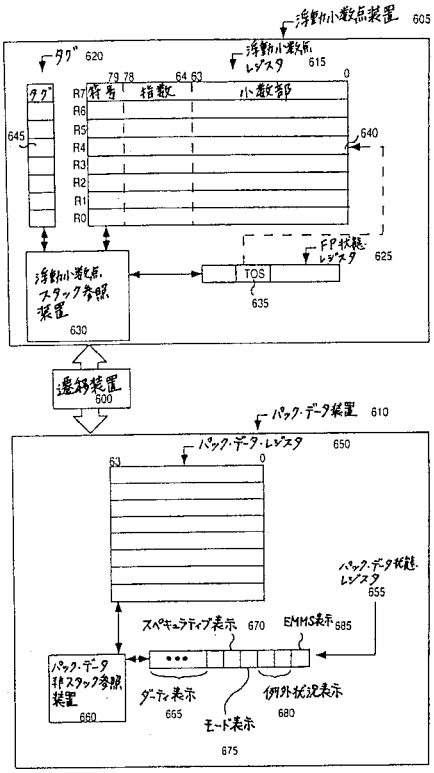
【図４Ｂ】


A
B
426
浮動小数点命令を実行し、必要に応じて
タグを変更し、発生した数値誤差を報告し、
数値エラーを未処理のままにしておく
430
このパック・データ
命令はEMMS命令か?
NO
YES
434
数値例外を認識せずに
パック・データ命令を実行する
SET TOS=0
432
すべてのタグ=空に設定する
SET TOS=0
436
このパック・データ命令はプロセッサに
エイリアス化レジスタへの書込みを行わせるか?
YES
438
書き込まれる各エイリアス化
レジスタの符号フィールドおよび
指数フィールドに1を記憶する
NO
440
すべてのタグ=非空に設定する。

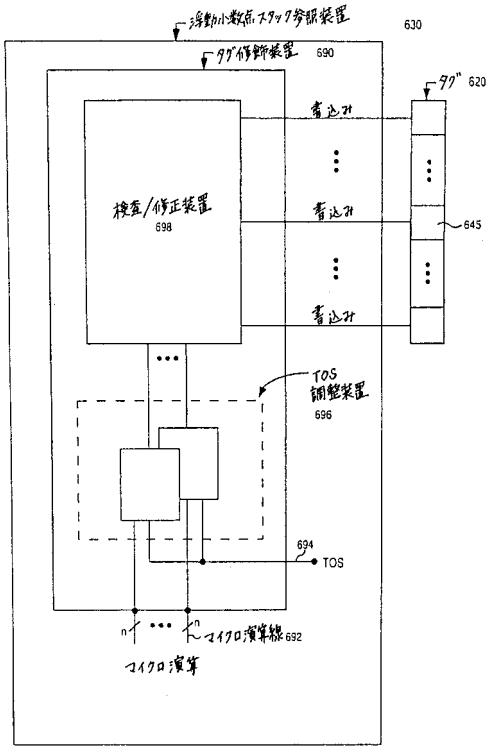
【図５】


500
プロセッサ
505
515
複号/実行装置
575
命令セット
580
メモリ装置
585
パック・データ命令
FP命令
汎用命令
命令セット装置
560
OS
535
パック・データ・ルーチン
540
記憶装置
510
キャッシュ
550
バス装置
545
キーボード
520
イベント処理装置
570
メモリ管理装置
565
ディスプレイ
525
ネットワーク
530
n

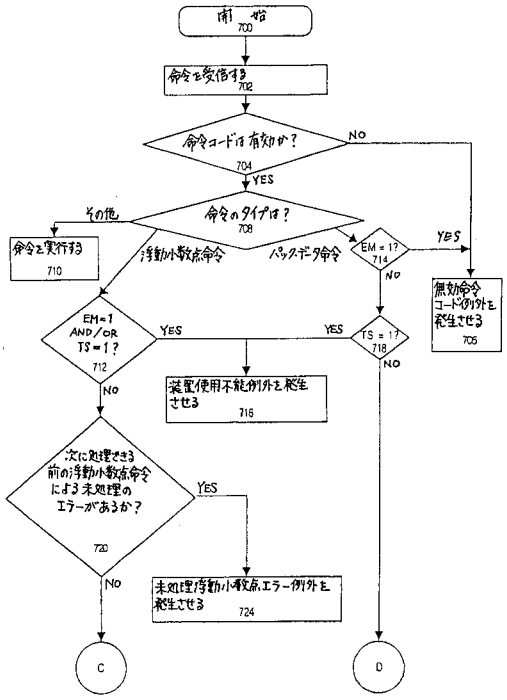
【図６Ａ】



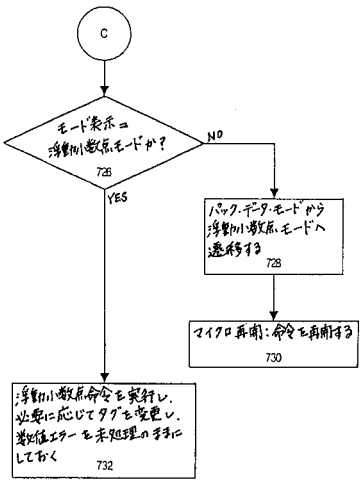
【図６Ｂ】



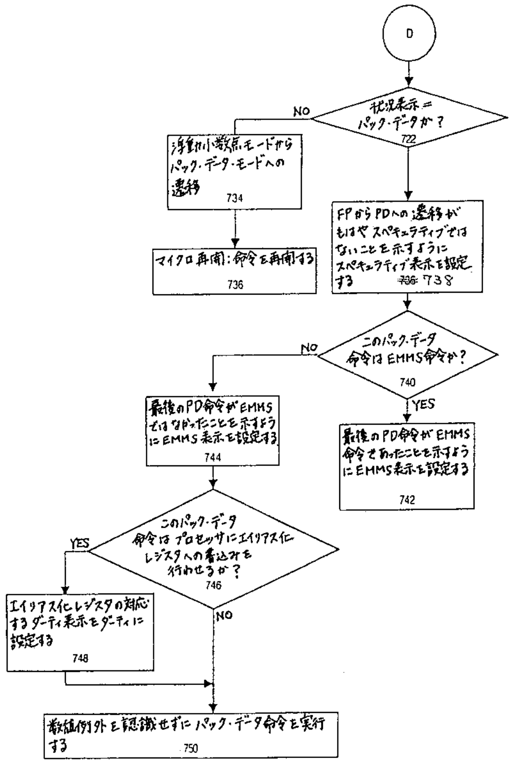
【図７Ａ】



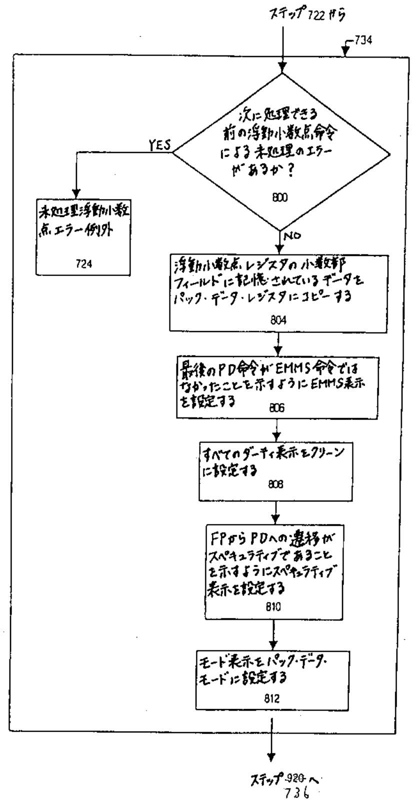
【図７Ｂ】



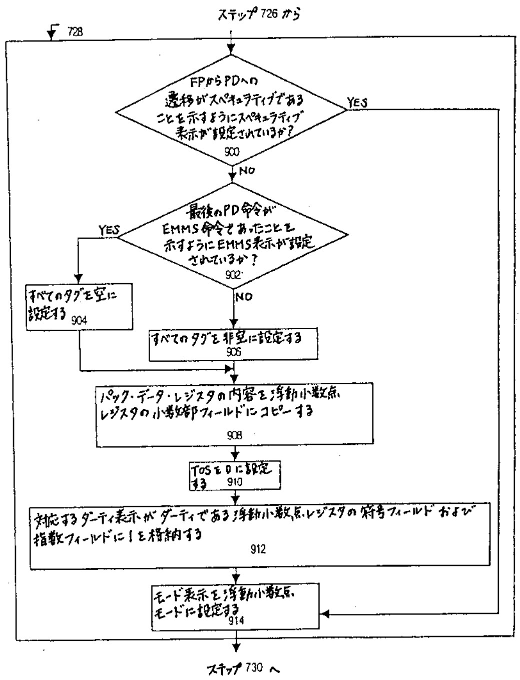
【図７Ｃ】


D
状況表示＝パック・データか？
722
NO
浮動小数点モードからパック・データ・モードへの遷移
734
マイクロ再開：命令を再開する
736
FPからPDへの遷移がもはやスペキュラティブではないことを示すようにスペキュラティブ表示を設定する
738
このパック・データ命令はEMMS命令か？
740
YES
最後のPD命令がEMMSではなかったことを示すようにEMMS表示を設定する
744
最後のPD命令がEMMS命令であったことを示すようにEMMS表示を設定する
742
このパック・データ命令はプロセッサにエイリアス化レジスタへの書込みを行わせるか？
746
エイリアス化レジスタの対応するダーティ表示をダーティに設定する
748
数値例外を認識せずにパック・データ命令を実行する
750

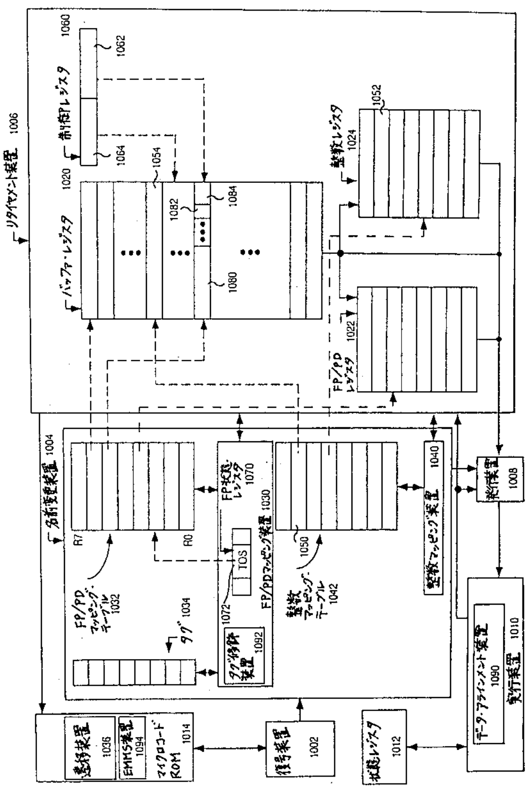
【図８】


ステップ722から
734
次に処理できる前の浮動小数点命令による未処理のエラーがあるか？
800
YES
NO
未処理浮動小数点エラー例外
724
浮動小数点レジスタの小数部フィールドに記憶されているデータをパック・データ・レジスタにコピーする
804
最後のPD命令がEMMS命令ではなかったことを示すようにEMMS表示を設定する
806
すべてのダーティ表示をクリーンに設定する
808
FPからPDへの遷移がスペキュラティブであることを示すようにスペキュラティブ表示を設定する
810
モード表示をパック・データ・モードに設定する
812
ステップ736へ

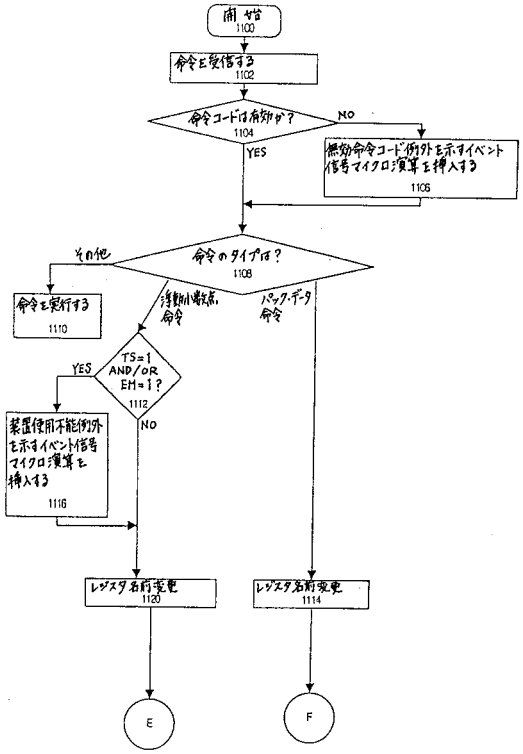
【図９】


ステップ726から
728
FPからPDへの遷移がスペキュラティブであることを示すようにスペキュラティブ表示が設定されているか？
900
YES
NO
最後のPD命令がEMMS命令であったことを示すようにEMMS表示が設定されているか？
902
すべてのタグを空に設定する
904
すべてのタグを非空に設定する
906
パック・データ・レジスタの内容を浮動小数点レジスタの小数部フィールドにコピーする
908
TOSを0に設定する
910
対応するダーティ表示がダーティである浮動小数点レジスタの符号フィールドおよび指数フィールドに1を格納する
912
モード表示を浮動小数点モードに設定する
914
ステップ730へ

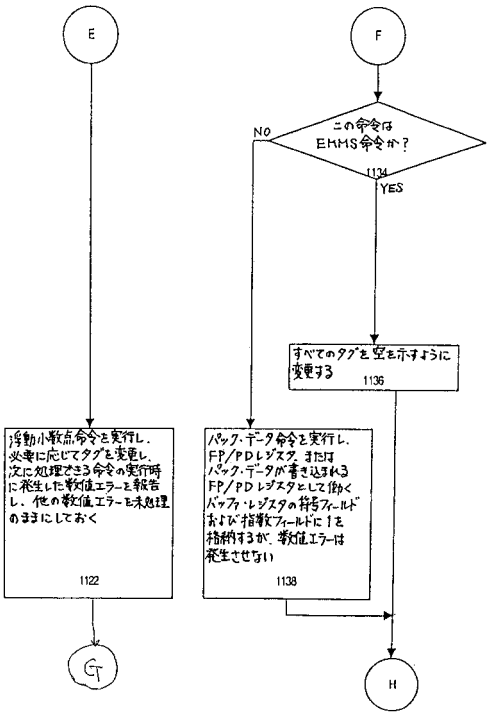
【図１０】


リタイヤメント装置
1006
制御レジスタ
1060
1062
1064
バッファ・レジスタ
1020
1054
1082
1084
1080
整数レジスタ
1024
1052
FP/PDレジスタ
1022
名前変更装置
1004
R7
R0
FP/PDマッピング・テーブル
1032
FP状況レジスタ
1070
1030
TOS
1072
1050
整数マッピング・テーブル
1042
整数マッピング装置
1040
発行装置
1008
タグ
1034
タグ修飾装置
1092
データ・アラインメント装置
1090
実行装置
1010
遷移装置
1036
EMMS装置
1094
マイクロコードROM
1014
復号装置
1002
状態レジスタ
1012

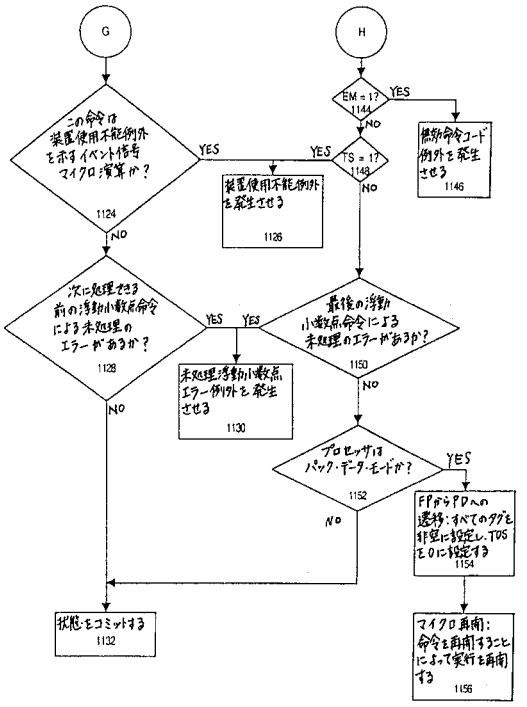
【図１１Ａ】



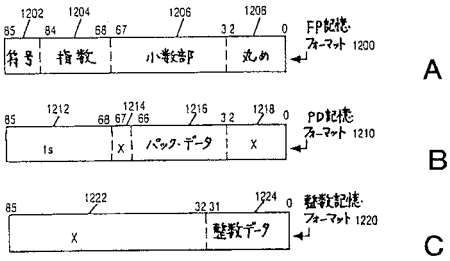
【図１１Ｂ】



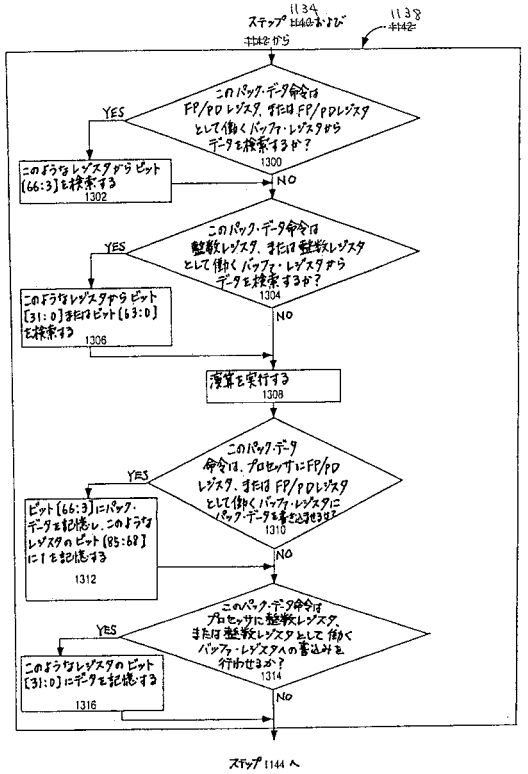
【図１１Ｃ】



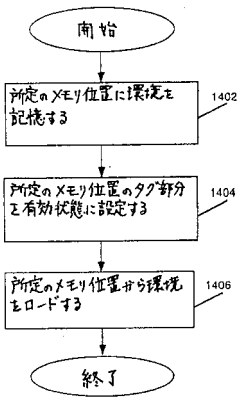
【図１２】



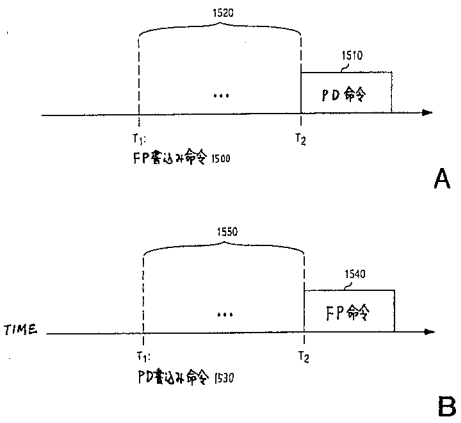
【図１３】


ステップ1134および1138から
このパック・データ命令はFP/PDレジスタ、または FP/PDレジスタとして働くバッファ・レジスタからデータを検索するか？
1300
YES
このようなレジスタからビット[66:3]を検索する
1302
NO
このパック・データ命令は整数レジスタ、または整数レジスタとして働くバッファ・レジスタからデータを検索するか？
1304
YES
このようなレジスタからビット[31:0]またはビット[63:0]を検索する
1306
NO
演算を実行する
1308
このパック・データ命令は、プロセッサにFP/PDレジスタ、またはFP/PDレジスタとして働くバッファ・レジスタにパック・データを書き込むか？
1310
YES
ビット[66:3]にパック・データを記憶し、このようなレジスタのビット[85:68]に1を記憶する
1312
NO
このパック・データ命令はプロセッサに整数レジスタ、または整数レジスタとして働くバッファ・レジスタへの書込みを行わせるか？
1314
YES
このようなレジスタのビット[31:0]にデータを記憶する
1316
NO
ステップ1144へ

【図１４】


開始
所定のメモリ位置に課題を記憶する
1402
所定のメモリ位置のタグ部分を有効状態に設定する
1404
所定のメモリ位置から課題をロードする
1406
終了

【図１５】


1520
1510
PD命令
T1:
T2
FP書込み命令 1500
A
1550
1540
FP命令
TIME
T1:
T2
PD書込み命令 1530
B

【手続補正書】
【提出日】平成15年12月17日(2003.12.17)
【手続補正１】
【補正対象書類名】明細書
【補正対象項目名】０１４６
【補正方法】変更
【補正の内容】
【０１４６】
ステップ７５０に示したように、パック・データ命令は、何らの数値例外を発生させることなく実行される。それ故ステップ７５０はトップ・オブ・スタック表示が変更される以外は。図４Ｂのステップ４４０と類似している。前述したように、オペレーティング・システムに追加のイベント・ハンドラが組み込まれ、あるいは既存のイベント・ハンドラがエラーを処理できるように、完全ではないがオペレーティング・システムを見ることができる別の装置が得られる。もし、パック・データ命令を実行しようと試みた結果、あるメモリ・イベントが発生すると、その実行は中断され、このイベントが処理される。

フロントページの続き

(72)発明者　グルー，アンドリュー・エフ
アメリカ合衆国・９７１２４・オレゴン州・ヒルズボロー・ノースイースト　キャスリン・８２５
(72)発明者　メネマイヤー，ラリー・エム
アメリカ合衆国・９５００６・カリフォルニア州・ブルダー　クリーク・番地なし・ピイオー　ボックス　５８７
(72)発明者　ペレグ，アレギザンダー・ディ
イスラエル国・３１０１５・ハイファ・カーメリア・ハンナ　ストリート・３８
(72)発明者　ビストリー，デイヴィッド
アメリカ合衆国・９５０１４・カリフォルニア州・カッパチーノ・パークウッド　ドライブ・１０２５３・６番
(72)発明者　ミタル，ミリンド
アメリカ合衆国・９４０８０・カリフォルニア州・サウス　サンフランシスコ・ヒルサイド　ブルバード・１１４９
(72)発明者　デュロング，キャロル
アメリカ合衆国・９５０７０・カリフォルニア州・サラトガ・ハーレイ　ドライブ・１８９８３
(72)発明者　小鷲　英一
日本国　３０１　茨城県竜ヶ崎市久保台２丁目８番９号
(72)発明者　エイタン，ベニー
イスラエル国・３１０１５・ハイファ・スティーブン　ワイズ・２５

Ｆターム(参考) 5B022 AA00　BA01　CA01　DA05　EA03　EA10　FA01　FA12
5B033 AA03　AA06　BD00　BE04　DD04