(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第3678443号
(P3678443)
(45)発行日　平成17年8月3日(2005.8.3)　(24)登録日　平成17年5月20日(2005.5.20)

(51)Int.Cl.[7]	F I		
G06F 9/38	G06F	9/38	330K
G06F 12/08	G06F	9/38	350X
	G06F	12/08	503C

請求項の数 12　(全 43 頁)

(21)出願番号	特願平6-251989
(22)出願日	平成6年10月18日(1994.10.18)
(65)公開番号	特開平7-152566
(43)公開日	平成7年6月16日(1995.6.16)
審査請求日	平成13年10月17日(2001.10.17)
(31)優先権主張番号	08/138651
(32)優先日	平成5年10月18日(1993.10.18)
(33)優先権主張国	米国(US)
(31)優先権主張番号	08/138652
(32)優先日	平成5年10月18日(1993.10.18)
(33)優先権主張国	米国(US)
(31)優先権主張番号	08/138654
(32)優先日	平成5年10月18日(1993.10.18)
(33)優先権主張国	米国(US)

(73)特許権者　501397942
ヴァイア　サイリックス　インコーポレイテッド
アメリカ合衆国　テキサス州　リチャードソン　ノース　セントラル　エクスプレスウェイ　２７０３
(74)代理人　100059959
弁理士　中村　稔
(74)代理人　100067013
弁理士　大塚　文昭
(74)代理人　100082005
弁理士　熊倉　禎男
(74)代理人　100065189
弁理士　宍戸　嘉一

最終頁に続く

(54)【発明の名称】スーパーパイプライン式スーパースカラーマイクロプロセッサ用の書き込みバッファ

(57)【特許請求の範囲】

【請求項１】

マイクロプロセッサであって、

プログラムオーダにおいて実行されるべき命令によって規定される動作にしたがって、データを処理するための中央処理装置のコア、前記命令の少なくとも１つは条件分岐型であり、

前記中央処理装置のコアに接続され、そこから前記命令の結果にしたがってデータを受け取るための、複数のバッファエントリを有する書き込みバッファ、前記バッファエントリの各々は、少なくとも１つの推論制御ビットがセットされると、前記バッファエントリの各々に書き込まれるべきデータが前記条件分岐型の命令後にプログラムオーダにおいて予想分岐命令の実行からであることを示す少なくとも１つの推論制御ビットを含み、且つ、推論制御ビットのセット数は、推論実行の次数に相当し、

複数のロケーションを有するキャッシュ、前記キャッシュは、前記書き込みバッファからデータを受け取るため、前記書き込みバッファへ結合され、および前記中央処理装置のコアへデータを送るため、前記中央処理装置のコアに結合されており、

前記中央処理装置のコア、前記書き込みバッファ及び前記キャッシュに接続され、前記キャッシュからデータを受け取り、且つ前記キャッシュへデータを送るためのメモリバス、及び

前記推論制御ビットがセットされない場合のみ、前記バッファエントリの内容が前記キャッシュへ送られるように、前記書き込みバッファによって前記キャッシュへのデータの

送りを制御するための制御ロジック、
を有することを特徴とするマイクロプロセッサ。
【請求項２】
前記バッファエントリの各々は、複数の推論制御ビットを有し、各々の推論制御ビットは、１つの予想分岐に対応し、且つ
前記制御ロジックは、また前記バッファエントリへのデータの書き込みが依存する予想分岐に対応するバッファエントリに対するこれらの推論制御ビットをセットするためであることを特徴とする請求項１に記載のプロセッサ。
【請求項３】
前記制御ロジックは、また予想分岐の首尾よい完了に応答する前記バッファエントリの推論制御ビットをクリアするためであり、且つ
前記制御ロジックは、また前記推論制御ビットがセットされた対応するバッファエントリを無効にするために、首尾よくいかない予想分岐のイベントにおいて動作可能であることを特徴とする請求項２に記載のマイクロプロセッサ。
【請求項４】
前記制御ロジックは、さらに、前記キャッシュへ送られる内容を有すべきである次のバッファエントリを示すためのリタイアポインタ、及び　命令の結果を受け取るために割当てられるべきである前記次のバッファエントリを示すための割当てポインタ、
を有し、
前記バッファエントリの各々は、さらに、非動作制御ビットがセットされると、前記リタイアポインタが前記バッファエントリをスキップするための非動作制御ビットを有し、且つ
前記制御ロジックは、また首尾よくいかない予想分岐セットに対応する推論制御ビットを有する前記バッファエントリの各々において前記非動作制御ビットをセットするためであることを特徴とする請求項１に記載のマイクロプロセッサ。
【請求項５】
前記バッファエントリの各々は、アドレス有効制御ビットがセットされると、前記バッファエントリがメモリに割当てられたメモリアドレスを有し、前記中央処理装置のコアによってメモリへのデータの連続した書き込みのために保存さたことを示すアドレス有効制御ビットを有し、且つデータ有効制御ビットがセットされると、前記バッファエントリが前記中央処理装置のコアによってバッファへ書き込まれた有効データを有したことを示すデータ有効制御ビットを有し、
前記中央処理装置のコアは、また例外条件の発生を検出するためであり、且つ　前記制御ロジックは、また例外条件の検出に応答して、アドレス有効制御ビットがセットされ、データ有効制御ビットをクリアする、各書き込みバッファエントリにおける前記アドレス有効制御ビットをクリアするためであることを特徴とする請求項１に記載のマイクロプロセッサ。
【請求項６】
マイクロプロセッサであって、
（ａ）アドレス計算ステージ、後に続く実行ステージ、後に続くライトバックステージを有するパイプライン化したコアユニット、
（ｂ）前記実行ステージ及び前記ライトバックステージに接続した複数のエントリを有する書き込みバッファ、前記複数のエントリの各々は、前記実効ステージにおいて実行され、且つ前記ライトバックステージから書き込まれた書き込み命令に相当する内容を含み、前記内容は、データフィールド、アドレスフィールド、データ、有効制御ビット、及び前記書き込み命令がキャッシュ可能でないか否かを表すキャッシュ可能でない書き込みビットを有し、
（ｃ）前記書き込みバッファから前記データ及びアドレスフィールドを受け取るために接続された複数のエントリを有するキャッシュメモリ、
（ｄ）前記アドレス計算ステージにおける読取り命令のアドレスを前記書き込みバッファ

における前記複数のアドレスフィールドの各々と比較するため、前記アドレス計算ステージ及び前記書き込みバッファに接続された制御論理回路、アドレスマッチに応答して、前記データ有効制御ビットはセットされ、且つ前記書き込みバッファにおける前記データフィールドを前記パイプライン化したコアユニットにおける前記実効ステージへバイパスして、前記キャッシュ可能でない書き込みビットはクリアであることを特徴とするマイクロプロセッサ。

【請求項７】

前記書き込みバッファにおける前記複数のエントリの各々は、さらに前記書き込み命令が前記書き込みバッファにおいて既にエントリのアドレスフィールドをオーバラップするアドレスフィールドを有するか否かを表す読取り可能なビットを有し、かつ前記制御ロジックは、さらに前記書き込みバッファにおける前記データフィールドを前記パイプライン化したコアユニットにおける前記実行ステージにバイパスする前に、セットされている読取り可能なビットに応答することを特徴とする請求項６に記載のマイクロプロセッサ。

【請求項８】

前記書き込みバッファにおける前記複数のエントリの各々は、さらに前記書き込み命令にプログラムオーダにおける前記読み取り命令が続くか否かを表すライト・アフタ・リード・ビットを含み、且つ前記制御ロジックは、さらに、前記書き込みバッファにおける前記データフィールドを前記パイプライン化したコアユニットにおける前記実行ステージへバイパスする前に、クリアである前記ライト・アフタ・リード・ビットに応答することを特徴とする請求項６に記載のマイクロプロセッサ。

【請求項９】

マイクロプロセッサであって、

結果を生成するために、プログラム順序における実行可能な命令によって規定された動作にしたがってデータを処理するための中央処理装置、

複数のバッファエントリを有し、前記中央処理装置に接続された、前記命令の結果にしたがってデータを受け取るための書き込みバッファ、前記複数のバッファは第１と第２のセクションに並べられており、

複数の記憶位置を有するキャッシュ、前記キャッシュは、前記書き込みバッファからデータを受け取るため、前記書き込みバッファに接続され、およびデータを前記中央処理装置のコアへ送るため、前記中央処理装置のコアに接続され、

前記キャッシュに接続され、前記キャッシュからデーを受け取りまた前記キャッシュへデータを送るためのメモリバス、

前記書き込みバッファに記憶された前記命令の結果が前記プログラム順序で前記キャッシュへ送られるように、前記書き込みバッファから前記キャッシュへデータの送りを制御するための制御ロジック、

複数のクロス依存性の制御ビットを含む前記第１と第２のセクションにおける各々のバッファエントリ、前記クロス依存性の制御ビットは、前記第２と第１のセクションにおけるバッファエントリの１つにそれぞれ対応し、前記クロス依存性の制御ビットがセットされると、前記対応するバッファエントリは、予め割当てられ、その内容がまだ前記キャッシュへ送られていないことを示し、且つ

前記制御ビットは、そのクロス依存性の制御ビットがクリアのときだけ、各々の書き込みバッファエントリのデータを前記キャッシュへ送ることを特徴とするマイクロプロセッサ。

【請求項１０】

処理されるべき前記命令の１つは、前記書き込みバッファの内容を前記キャッシュへの送りを有する前記プログラム順序で行われるべきであるメモリからキャッシュ可能でない読取りを含み、さらに、

複数のビット位置を有するキャッシュ可能でない読取りフィールドを有し、各々のビット位置は書き込みバッファにおける前記バッファエントリの１つにそれぞれ対応し、且つビット位置がセットされると、前記対応するバッファエントリは予め割当てられ、その内

容がまだ前記キャッシュへ送られていないことを示し、
前記制御論ロジックは、前記キャッシュ可能でない読取りフィールドがクリアでないならば、前記キャッシュ可能でない読取りの処理を防止することを特徴とする請求項９に記載のマイクロプロセッサ。
【請求項１１】
マイクロプロセッサであって、
（ａ）第１のプログラム命令を処理し、所定の幅の第１の書き込みオペランドを生成する第１の処理装置、
（ｂ）第２のプログラム命令を処理し、前記第１の書き込みオペランドの前記所定の幅より大きい幅を有する第２の書き込みオペランドを生成する第２の処理装置、
（ｃ）前記第２の書き込みオペランドの幅に等しい幅を有し、前記第２の処理装置から前記第２の書き込みオペランドを受取るために接続された余分な広いラッチ、
（ｄ）複数のエントリを有し、前記第１の処理装置に接続された統一された書き込みバッファ、前記複数のエントリの各々は、前記第１の書き込みオペランドを記憶するデータ部分、アドレスを記憶するアドレス部分、及び前記アドレスが前記余分な広いラッチにおける前記第１の書き込みオペランドかまたは前記第２の書き込みオペランドに属するかを識別する制御部分を有し、
（ｅ）前記余分な広いラッチおよび前記統一された書き込みバッファに接続された所定の幅のデータパス、前記余分な広いラッチの幅は、前記データパスの所定の幅に等しいか、またはそれより小さく、且つ
（ｆ）前記統一された書き込みバッファの制御部分に応答する、前記第１の書き込みオペランドか前記第２の書き込みオペランドの何れかを受取るために、前記統一された書き込みバッファからのアドレス部分に、及び前記データパスへのアドレス部分に接続されたメモリ、
を有することを特徴とするマイクロプロセッサ。
【請求項１２】
一連の命令にしたがって、データ処理の動作を実行するための、中央処理装置のコアを有するパイプライン化したマイクロプロセッサであって、前記マイクロプロセッサは、キャッシュへストアする前に前記中央処理装置のコアによって実行さえる命令の結果をバッファするための複数の書き込みバッファエントリを有する書き込みバッファ、
第１の命令の結果を記憶するのための、第１のメモリアドレスを決め、また第１の書き込みバッファエントリにおける前記第１のメモリアドレスに相当する第１の物理アドレスをストアするように動作可能な回路、及び
前記第１の命令の実効前に、例外条件を検出するように構成され、例外条件の前記検出に応答して、前記第１の書き込みバッファエントリを無効にするように構成された回路、
を有するマイクロプロセッサ。
【発明の詳細な説明】
【０００１】
【産業上の利用分野】
本発明は、マイクロプロセッサ型の集積回路の分野に係り、より詳細には、そのメモリアクセス回路に係る。
【０００２】
【従来の技術】
マイクロプロセッサの分野においては、１秒当たりに実行される命令の数が主たる性能尺度である。この技術で良く知られているように、マイクロプロセッサの設計及び製造における多数のファクタがこの尺度に影響を及ぼす。例えば、実行速度は、マイクロプロセッサのクロック周波数によってかなり左右される。しかしながら、マイクロプロセッサに与えられるクロックの周波数は、電力消費の状態、及びマイクロプロセッサ内のトランジスタのスイッチング特性により制限される。
【０００３】

又、マイクロプロセッサのアーキテクチャも、マイクロプロセッサの実行速度において意義のあるファクタである。例えば、多数の近代的なマイクロプロセッサは、それらの命令の多くが実行のために多数のクロックサイクルを必要とする場合にそれらの実行速度を向上するために「パイプライン式」アーキテクチャを使用している。従来のパイプライン技術によれば、各々のマイクロプロセッサ命令は多数の段階にセグメント化され、そして命令の各段階を実行するために個別の回路が設けられる。従って、各クロックサイクルにおいて多数の命令の異なる段階の実行をオーバーラップさせることによりマイクロプロセッサの実行速度が増加される。このように、各クロックサイクルにおいて１つの多サイクル命令を完了することができる。

【０００４】

更に背景として、あるマイクロプロセッサアーキテクチャは、「スーパースカラー」型のものであり、並列実行のために各々のクロックサイクルに多数の命令が発生される。これら命令の間に依存性がないと仮定すれば、スカラー性の程度に比例して命令スループットが増大する。

【０００５】

マイクロプロセッサ及びこれが実施されるシステムの実行速度を改善するための別の公知技術は、キャッシュメモリを使用することである。従来のキャッシュメモリは、選択アルゴリズムによって決定されるように後で命令を実行する際にアクセスされる見込みの高いメモリ位置からのプログラム及びデータを記憶する小型の高速メモリである。キャッシュメモリは、メインシステムメモリに対して少数のクロックサイクル（しばしば単一のサイクル）でアクセスできるので、キャッシュを用いたマイクロプロセッサの有効実行速度は、非キャッシュシステムよりも相当に改善される。多くのキャッシュメモリは、マイクロプロセッサ自体と同じ集積回路チップ上に配置され、更に性能の改善を与える。

【０００６】

【発明が解決しようとする課題】

これらのアーキテクチャ関連の性能改善技術の各々によれば、マイクロプロセッサの性能を低速化する幾つかの事象が生じる。例えば、パイプライン式及びスーパースカラー式の両方のアーキテクチャにおいては、多数の命令が同時に同じ内部回路へのアクセスを要求することがあり、この場合には、優先順位の高い命令が回路によって応対されるまで命令の１つが待機しなければならない（即ち、「ストール」）。

【０００７】

このような競合の１つの形式は、ある命令がメモリ（キャッシュを含む）への書き込みを要求すると同時に、別の命令がメモリからの読み取りを要求する場合にしばしば生じる。命令が「先着順サービス(first-come-first-served)」ベースで応対される場合には、後で到着する命令は、メモリアクセスが許可されるまで、手前の命令の完了を待機しなければならない。これら及び他のストールは、もちろん、マイクロプロセッサの性能に対して不利益となる。

【０００８】

ほとんどの命令シーケンス（即ち、プログラム）に対し、特に、マイクロプロセッサアーキテクチャに多数の汎用レジスタが設けられている場合には、メモリ又はキャッシュからの読み取りの方が、一般に、メモリ又はキャッシュへの書き込みよりも時間的に厳密であることが分かっている。これは、プログラムの実行において、プログラムを効率的に実行するためには、特定の時間に命令及び入力データが必要であるのに対し、メモリへの書き込みはプログラム実行の結果を単に書き込むものでありその後の命令の実行はその結果に左右されないから、書き込みが行われる実際の時間は重要でないためである。

【０００９】

更に背景技術として、マイクロプロセッサには書き込みバッファが設けられており、これらの書き込みバッファは、オンチップのキャッシュメモリと、メインメモリへのバスとの間に論理的に配置されている。これらの従来のキャッシュ後の書き込みバッファは、ライトスルー又はライトバック動作のためにキャッシュからデータを受け取り、このキャッシ

ュ後の書き込みバッファの内容は、バスが使用できるようになったときに、バスコントローラの制御のもとでメインメモリに書き込まれる。

【００１０】

更に背景技術として、多くの近代的なマイクロプロセッサは、必ずしもオペランドサイズのモジュロではないアドレスを用いてメモリ位置をアクセスすることができる。このような形式のマイクロプロセッサタイプの一例は、「Ｘ８６」コンパチブルと一般に称されているものである。それ故、これらのマイクロプロセッサにおいて、あるメモリ書き込みは、最下位バイトアドレスを含むバイトブロックから外れたバイトを含むことがあり、書き込み動作を行うのに多数の書き込みサイクルが必要とされる。これらの書き込みは、「不整列書き込み」としばしば称される。実際に、多くのマイクロプロセッサでは、メモリ書き込みの重要な部分がバイトブロックの境界にオーバーラップする。

【００１１】

更に背景技術として、整数の中央処理ユニットと、浮動小数点の処理ユニットとの両方を含むマイクロプロセッサが知られている。このようなマイクロプロセッサでは、整数結果のデータワード巾の方が、浮動小数点ユニットよりも小さくなり、例えば、整数データは３２ビット巾であるが、浮動小数点データは６４ビット巾である。浮動小数点ユニットに加えて、他の処理ユニットも、中央処理ユニットによって与えられるものよりも巾の広いデータを与える。

【００１２】

更に背景技術として、多くのマイクロプロセッサ用途では、中央処理ユニットのコアからキャッシュ又はメモリへの書き込みを、プログラムの順序で、即ちプログラマによって与えられた命令の順序で行って、適切なプログラム動作を確保することが重要である。例えば、円形バッファ及びカウンタを設けることにより単一ユニットの動作をプログラムの順序に維持するための方法が知られている。しかしながら、２つ以上のバッファ又は動作をプログラムの順序に維持しなければならない場合には、１つのカウンタの使用では不充分である。

【００１３】

更に背景技術として、いわゆる「Ｘ８６」コンパチビリティを有するもののような従来アーキテクチャのマイクロプロセッサは、内部データバスの容量より小さいバイトサイズの書き込み動作を行うことが知られている。

【００１４】

更に背景技術として、パイプライン式マイクロプロセッサは、データ依存性と一般に称される幾つかのハザードに弱いことが知られている。一般的に、データ依存性は、パイプラインにおいて異なる段階にある２つの命令が同じレジスタ又はメモリ位置へのアクセスを要求するときに生じる。というのは、手前の命令でレジスタ又はメモリ位置にデータが書き込まれてしまう前にパイプラインが後の命令のために（プログラム順序で）そこにアクセスすることがあり、動作エラーを生じるからである。従来のパイプライン式マイクロプロセッサにおいてこのようなデータ依存性を検出するための技術が知られており、パターソン及びヘネシー著の「コンピュータアーキテクチャ：定量的アプローチ（モーガン・カウフマン、１９９０年）」、第２５７ないし７８ページに説明されている。従来技術によれば、データ依存性又はハザードの検出は、手前の命令（プログラム順序で）が完了するまでパイプラインをストールし、その後、後続命令を処理できるようにすることにより取り扱われる。もちろん、パイプラインのストールは、マイクロプロセッサの性能のロスを招く。

【００１５】

更に背景技術として、あるパイプライン式アーキテクチャのマイクロプロセッサは、プログラムシーケンスに条件分岐又はジャンプ命令が存在するにも係わりなくパイプラインをいっぱいに維持するために推論的実行に基づいて動作する。この推論的実行は、マイクロプロセッサが条件分岐が行われるかどうかをアルゴリズムに基づいて予想する場所で予想分岐を行い、次いで、その予想した経路をパイプラインにおいて実行することを必要とす

る。この推論的に実行した命令の結果はメモリにもキャッシュにも書き込まれないのが重要である。というのは、予想が誤りであった場合に、その誤って実行されたメモリ書き込みから回復することは困難又は不可能だからである。更に、メモリへの書き込みを含む命令がパイプラインにおいて処理される場合であって、プログラムの実行を直ちに停止しなければならない例外条件（例えば、０で除算）を手前の命令が有している場合には、別の形式の状況が生じ得る。

【００１６】

【課題を解決するための手段】

本発明は、マイクロプロセッサのコアとメモリとの間に論理的に配置されたマイクロプロセッサの書き込みバッファであって、メモリへの各書き込みがメモリバス又はキャッシュメモリではなくてこの書き込みバッファに向けられるような書き込みバッファに係る。この書き込みバッファの内容は、次いで、メモリバス又はキャッシュメモリが使用できるときに、キャッシュメモリ又はメインメモリへ非同期で書き込まれる。

【００１７】

本発明の特徴は、不整列書き込みの処理能力を含む。不整列書き込みが検出された場合には、第１の書き込みバッファエントリが第２の書き込みバッファエントリに割り当てられ、上位アドレスが計算されてそこに記憶される。不整列書き込みがリタイアすると、第１の書き込みバッファエントリに記憶されたデータが適当なバイトレーンへシフトされて下位バイトが上位端に来るようにされそしてこれとは逆にもなされる。第１エントリ及び下位データのリタイアの後にシフトされたデータを記憶するためのラッチが設けられる。このラッチは、上位データを第２エントリに対するアドレスと共に与え、不整列書き込みを完了する。

【００１８】

本発明の別の特徴は、過剰な巾をもつ全書き込みバッファを構成する必要なく過剰巾のデータをバッファすることである。浮動小数点ユニットのような二次の処理ユニットは、中央処理ユニットにより発生されるワードの巾よりも巾の広い（ビットで）ワードで結果を発生する。二次処理ユニットの結果を記憶するために二次データラッチが設けられ、そのデータが有効であるときに制御ビットがセットされる。標準的な書き込みバッファエントリには、二次処理ユニットの結果に対応する物理アドレスが割り当てられ、そのデータが二次データラッチに記憶されることを指示するように制御ビットがセットされる。書き込みバッファエントリがリタイアすると、書き込みバッファエントリの内容ではなくて二次データラッチの内容がキャッシュに与えられる。

【００１９】

本発明の別の特徴は、書き込みバッファがレイアウト又は動作効率のために２つの区分に分割される場合に書き込みバッファからのデータの書き込みがプログラム順序となるよう確保することである。プログラム順序は、書き込みバッファエントリにクロス依存性フィールドを含ませることにより維持され、このフィールドは、各書き込みバッファエントリの割り当ての際に、逆の区分にあるどの書き込みバッファエントリが既に割り当てられたかを示すマップがロードされる。各書き込みバッファエントリのクロス依存性フィールドは、その前にプログラム順序にあった各書き込みバッファがリタイアするたびに、ビットごとにクリアされる。書き込みバッファエントリがプログラム順序でリタイアすることは、そのクロス依存性フィールドをクリアすることを必要とすることにより確保される。更に、同様の概念を使用し、読み取りが割り当てられたときに割り当てられた書き込みバッファエントリのマップである読み取りに対してクロス依存性フィールドを与え、動作を同様の形態でゲートすることにより、書き込みバッファからリタイアしながらプログラム順序で非キャッシュ処理読み取りする性能を確保することができる。

【００２０】

本発明の別の特徴は、書き込みバッファからキャッシュへの集合書き込みを実行する構成を含む。書き込みデータエントリの割り当て中に、現在割り当てられたエントリと既に割り当てられているエントリの物理アドレスの比較が行われ、少なくとも割り当てられた物

理アドレスが同じバイトグループ内にあるかどうか決定され、その場合には、多数の書き込みをキャッシュへの単一の書き込み動作に集合又は併合することができる。集合性に対する他の制約は、バイトが互いに相接していること、及び書き込みがプログラム順序で隣接する書き込み命令からのものであることを含む。集合可能な書き込みバッファエントリのリタイアは、データを適切なバイトレーンに入れるようにシフトした後に、書き込みバッファエントリからのデータをラッチにロードすることにより行われ、書き込みは、ラッチの内容と組み合わせてアドレスを与えることにより行われる。

【００２１】

本発明の別の特徴は、リード・アフタ・ライト（ＲＡＷ）依存性のようなデータハザード即ち依存性を、特に書き込みバッファに既に書き込まれているデータに対して検出する構成を含む。書き込みバッファ内容のリタイアは、ＲＡＷ依存性を受けるエントリについては防止され、従って、エラーのある読み取りが回避される。又、書き込みバッファからデータを直接供給するか又は書き込みバッファをバイパスすらさせる能力も設けられ、ＲＡＷハザードによるパイプラインストールの影響が低減される。又、多数の読み取りの最後のものだけをコアへ供給するような更に別の能力も設けられる。偽のＲＡＷハザード検出を回避するために、ライト・アフタ・リード制御も与えられる。

【００２２】

本発明の別の特徴は、推論的実行中に書き込みバッファへの書き込みが許される場合に各書き込みバッファエントリに対する制御ビットの推論的実行フィールドを含むことである。各制御ビットは、予想又は推論分岐に対応し、書き込みについての推論の程度に基づいて書き込みバッファエントリの割り当ての際にセットされる。誤った予想の場合には、失敗予想に対してその推論的制御ビットがセットされている各書き込みバッファエントリがフラッシュされ、書き込みバッファエントリが再割り当てに使用できるようになる。割り当てられているが例外の時にまだリタイアされていない全ての書き込みバッファエントリをクリアすることにより例外取り扱いが実行される。バッファが空であるときにリタイアポインタ及び割り当てポインタが一致できるようにするために各書き込みバッファエントリごとにノーオペレーションビットが設けられる。

【００２３】

本発明の実施例は、次のような効果的な技術的作用の１つ以上を実現するように実施される。

データをキャッシュへリタイアする前に、ＣＰＵコアから書き込みバッファへのデータの書き込みをバッファし、そして不整列書き込みを最小限の性能ロスで容易に取り扱えるようにするマイクロプロセッサアーキテクチャ；

データをキャッシュ又はメモリへリタイアする前に実行結果を書き込みバッファに記憶できるようにし、書き込みバッファは、二次処理ユニットによって与えられるものよりもビット巾の小さい複数の位置を有しているマイクロプロセッサアーキテクチャ；

余計なビット巾を受け入れるように全ての書き込みバッファ位置を構成する必要なく、二次処理ユニットの結果に対してバッファ作用が与えられるようなマイクロプロセッサアーキテクチャ；

データをキャッシュへリタイアする前に、ＣＰＵコアから書き込みバッファへのデータの書き込みをバッファし、書き込みバッファが２つのセクションに分割されているマイクロプロセッサアーキテクチャ；

書き込みバッファからキャッシュ又はメインメモリへプログラム順序でリタイアするよう確保するマイクロプロセッサアーキテクチャ；

書き込みバッファからデータをリタイアしながらプログラム順序で非キャッシュ式読み取りを実行できるようにするマイクロプロセッサアーキテクチャ；

データをキャッシュ又はメモリへリタイアする前に実行結果を書き込みバッファに記憶できるようにし、多数の書き込み動作からの書き込みデータを記憶して書き込みバッファからキャッシュへ１つのサイクルで与えるような能力が設けられたマイクロプロセッサアーキテクチャ；

集合可能又は併合可能な書き込み動作を検出し、書き込みバッファからキャッシュへ集合書き込みを行えるようにするマイクロプロセッサアーキテクチャ；
データをキャッシュ又はメモリへリタイアする前に、データ依存性を検出できるような仕方で実行結果を書き込みバッファに記憶できるようにするマイクロプロセッサアーキテクチャ；
他の明白なデータ依存性が実際にはデータ依存性でないという指示で書き込みバッファ位置を割り当てできるようにするマイクロプロセッサアーキテクチャ；及び
データをキャッシュへリタイアするまえに、ＣＰＵコアから書き込みバッファへのデータの書き込みをバッファし、推論的実行又は例外からの回復を容易に行い得るようにするマイクロプロセッサアーキテクチャ。

【００２４】

【実施例】

本発明の他の目的及び特徴は、添付図面を参照した以下の詳細な説明から当業者に明らかとなろう。本発明のマイクロプロセッサの実施例の詳細な説明は、次のように構成される。

１．模範的プロセッサシステム
２．一般化されたパイプラインアーキテクチャ
３．書き込みバッファアーキテクチャ及び動作
４．リード・アフタ・ライトハザードの検出及び書き込みバッファの動作
５．推論的実行及び例外の取り扱い
６．書き込みバッファからの特殊な書き込みサイクル
７．結論

【００２５】

この構成テーブル及びこの詳細な説明で使用する対応する見出しは、参照の便宜上設けられているものに過ぎない。マイクロプロセッサの従来又は既知の観点の説明は、これを不必要に詳細にすることにより本発明の説明を不明瞭にしない程度に省略する。

【００２６】

１．模範的プロセッサシステム

模範的プロセッサシステムが図１、２及び３に示されている。図１及び２は、各々、模範的なスーパースカラー、スーパーパイプライン型マイクロプロセッサと、２つの実行パイプラインのパイプ段階との基本的な機能ブロックを示している。図３は、マイクロプロセッサを使用する模範的プロセッサシステム（マザーボード）設計を示している。

【００２７】

１．１　マイクロプロセッサ

図１を参照すれば、マイクロプロセッサ１０の主たるサブブロックは、次のものを含む。（ａ）中央処理ユニット（ＣＰＵ）コア２０；（ｂ）プリフェッチバッファ３０；（ｃ）プリフェッチャ３５；（ｄ）分岐処理ユニット（ＢＰＵ）４０；（ｅ）アドレス変換ユニット（ＡＴＵ）５０；及び（ｆ）ＴＡＧ　ＲＡＭ６２を含む単一化した１６Ｋバイトのコード／データキャッシュ６０。２５６バイトの命令ラインキャッシュ６５は、単一化キャッシュへの命令フェッチを減少するための一次命令キャッシュを構成し、単一化キャッシュは、二次命令キャッシュとして働く。オンボード浮動小数点ユニット（ＦＰＵ）７０は、ＣＰＵコア２０によってこれに発せられた浮動小数点命令を実行する。

【００２８】

マイクロプロセッサは、内部３２ビットアドレス及び６４ビットデータバスＡＤＳ及びＤＡＴＡを使用している。単一化キャッシュ６０及び命令ラインキャッシュ６５の３２バイトラインサイズに対応する２５６ビット（３２バイト）プリフェッチバス（ＰＦＢ）は、３２命令バイトの全ラインを単一のクロックにおいて命令ラインキャッシュへ転送できるようにする。外部３２ビットアドレス及び６４ビットデータバスへのインターフェイスは、バスインターフェイスユニット（ＢＩＵ）を経て行われる。

【００２９】

ＣＰＵコア２０は、２つの実行パイプＸ及びＹを有するスーパースカラー設計のものである。これは、命令デコーダ２１と、アドレス計算ユニット２２Ｘ及び２２Ｙと、実行ユニット２３Ｘ及び２３Ｙと、３２個の３２ビットレジスタをもつレジスタファイル２４とを備えている。ＡＣコントロールユニット２５は、レジスタスコアボード及びレジスタ再ネーミングハードウェアをもつレジスタ変換ユニット２５ａを備えている。マイクロシーケンサ及びマイクロＲＯＭを含むマイクロコントロールユニット２６は、実行制御を与える。

【００３０】

ＣＰＵコア２０からの書き込みは、１２個の３２ビット書き込みバッファ２９へ待ち行列され、書き込みバッファの割り当ては、ＡＣコントロールユニット２５によって実行される。これらの書き込みバッファは、単一化キャッシュ６０へ書き込むためのインターフェイスを形成し、非キャッシュ処理書き込みは、書き込みバッファから外部メモリへ直接送られる。書き込みバッファロジックは、オプションの読み取りソース作用及び書き込み集合作用をサポートする。

【００３１】

パイプコントロールユニット２８は、実行パイプを通る命令の流れを制御し、これは、命令が例外を生じないことが決定されるまで命令の順序を保持し、命令流におけるバブルを押しつぶし、そして誤って予想された分岐及び例外を生じる命令の後に実行パイプをフラッシュすることを含む。各段階に対し、パイプコントロールユニットは、どの実行パイプが最も初期の命令を含むかを追跡し、「ストール」出力を与え、そして「遅延」入力を受け取る。

【００３２】

ＢＰＵ４０は、分岐（行われる又は行われない）の方向を予想し、そしてその予想された行われる分岐及び流れ命令の無条件変更（ジャンプ、コール、リターン）に対するターゲットアドレスを与える。更に、このＢＰＵは、分岐及び浮動小数点命令の場合には推論的実行を監視し、即ち予想を誤ることのある分岐後に推論的に発生される命令の実行、及びＦＰＵ７０へ発生される浮動小数点命令であって推論的に発生された命令が実行を完了した後に失敗となることのある浮動小数点命令の実行を監視する。浮動小数点命令が失敗に終わった場合、又は分岐が誤って予想された（これは、分岐に対するＥＸ又はＷＢ段階まで分からない）場合には、実行パイプラインは、その失敗となった又は予想を誤った命令の点まで修理され（即ち、その命令の後に実行パイプラインがフラッシュされ）、そして命令のフェッチが再開されねばならない。

【００３３】

パイプラインの修理は、各パイプ段において浮動小数点又は予想分岐命令がその段に入るときにプロセッサ状態のチェックポイントを形成することにより行われる。これらのチェックポイント検査される命令に対し、その後の推論的に発生される命令によって変更され得る全てのリソース（プログラマが見ることのできるレジスタ、命令ポインタ、条件コードレジスタ）がチェックポイント検査される。チェックポイント検査される浮動小数点命令が失敗に終わるか又はチェックポイント検査される分岐が誤って予想された場合は、そのチェックポイント検査された命令の後に実行パイプラインがフラッシュされ、浮動小数点命令の場合には、これは、典型的に、実行パイプライン全体をフラッシュすることを意味し、一方、誤って予想された分岐の場合には、完了することが許されたＥＸの対命令及びＷＢの２つの命令があることを意味する。

【００３４】

模範的なマイクロプロセッサ１０の場合に、推論の程度についての主たる制約は、次の通りである。（ａ）一度に４つまでの浮動小数点又は分岐命令に対してのみ推論的実行が許される（即ち、推論レベルは最大４である）。（ｂ）書き込み又は浮動小数点の記憶は、それに関連した分岐又は浮動小数点命令が解決する（即ち、予想が正しいか又は浮動小数点命令が失敗に終わらない）までキャッシュ又は外部メモリに対して完了しない。

【００３５】

単一化キャッシュ６０は、４方セット連想（４ｋセットサイズをもつ）のもので、擬似ＬＲＵ置換アルゴリズムを使用し、ライトスルー及びライトバックモードを有している。これは、クロック当たり２つのメモリアクセス（データ読み取り、命令フェッチ又はデータ書き込み）を許すためにデュアルポート式（バンク構成による）にされている。命令ラインキャッシュは、完全連想、ルックアサイド実施（単一化キャッシュに対して）のもので、ＬＲＵ置換アルゴリズムを使用する。

【００３６】

ＦＰＵ７０は、４深さロード及び記憶待ち行列をもつロード／記憶段と、変換段（３２ビットないし８０ビットの拡張フォーマット）と、実行段とを備えている。ロードは、ＣＰＵコア２０により制御され、そしてキャッシュ処理記憶は、書き込みバッファ２９により指示される（即ち、各浮動小数点記憶動作に対して書き込みバッファが割り当てられる）。

【００３７】

図２を参照すれば、マイクロプロセッサは、７段のＸ及びＹ実行パイプラインを有し、即ち、命令フェッチ段（ＩＦ）、２つの命令デコード段（ＩＤ１、ＩＤ２）、２つのアドレス計算段（ＡＣ１、ＡＣ２）、実行段（ＥＸ）、及びライトバック段（ＷＢ）を有している。複合ＩＤ及びＡＣパイプ段は、スーパーパイプライン式であることに注意されたい。

【００３８】

ＩＦ段は、ＣＰＵコア２０に連続的なコード流を与える。プリフェッチャ３５は、（一次）命令ラインキャッシュ６５或いは（二次）単一化キャッシュ６０のいずれかからプリフェッチバッファ３０へ１６バイトの命令データをフェッチする。ＢＰＵ４０は、プリフェットアドレスでアクセスされ、そして予想される流れの変更に対してプリフェッチャへターゲットアドレスを供給し、プリフェッチャが１つのクロック内に新たなコード流へシフトできるようにする。

【００３９】

デコード段ＩＤ１及びＩＤ２は、可変長さのＸ８６命令セットをデコードする。命令デコーダ２１は、各クロックごとにプリフェッチバッファ３０から１６バイトの命令データを検索する。ＩＤ１において、２つの命令の長さがデコードされて（Ｘ及びＹの実行パイプに対して各々１つづつ）、Ｘ及びＹ命令ポインタを得、それに対応するＸ及びＹバイト使用信号がプリフェッチバッファへ返送される（これは、次いで、次の１６バイト転送のために増加する）。又、ＩＤ２においても、流れの変更のような幾つかの命令形式が決定され、即座及び／又は変位オペランドが分離される。ＩＤ２段は、Ｘ及びＹ命令のデコード、マイクロＲＯＭに対するエントリポイントの発生、及びアドレスモード及びレジスタフィールドのデコードを完了する。

【００４０】

ＩＤ段の間に、命令を実行するための最適なパイプが決定され、命令がそのパイプへ発生される。パイプの切り換えにより、ＩＤ２ＸからＡＣ１ＹへそしてＩＤ２ＹからＡＣ１Ｘへ命令を切り換えることができる。模範的な実施例については、流れ変更命令、浮動小数点命令及び排他的命令のような幾つかの命令がＸパイプラインのみへ発生される。排他的命令は、ＥＸパイプ段において失敗となることのある命令、及びある形式の命令、例えば、保護モードセグメントロード、ストリング命令、特殊なレジスタアクセス（制御、デバッグ、テスト）、乗算／除算、入力／出力、プッシュオール／ポップオール（ＰＵＳＨ／ＰＯＰＡ）、及びタスクスイッチを含む。排他的命令は、両パイプのリソースを使用することができる。というのは、これらは、ＩＤ段のみから発生される（即ちこれらは他の命令と対にされない）からである。これらの発生制約を除くと、いかなる命令も対にして、Ｘ又はＹのいずれのパイプへ発生することもできる。

【００４１】

アドレス計算段ＡＣ１及びＡＣ２は、メモリ参照のためのアドレスを計算し、そしてメモリオペランドを供給する。ＡＣ１段は、クロックごとに２つの３２ビットリニア（３オペランド）アドレスを計算する（比較的稀である４つのオペランドアドレスは、２つのクロ

ックを必要とする）。データ依存性もチェックされそしてレジスタ変換ユニット２５ａ（レジスタスコアボード及びレジスタ再ネーミングハードウェア）を用いて分析され、３２個の物理レジスタ２４は、Ｘ８６アーキテクチャで定められた８個の汎用のプログラマから見える論理レジスタをマップするのに使用される（ＥＡＸ、ＥＢＸ、ＥＣＸ、ＥＤＸ、ＥＤＩ、ＥＳＩ、ＥＢＰ、ＥＳＰ）。

【００４２】

ＡＣユニットは、８個のアーキテクチャ（論理）レジスタ（Ｘ８６で定められたレジスタセットを表す）を備えており、これらは、アドレス変換のためのレジスタオペランドをアクセスする前にレジスタ変換ユニットＡＣ１のアクセスに必要な遅延を回避するようにＡＣユニットによって使用される。アドレス変換を必要とする命令については、ＡＣ１は、アーキテクチャレジスタをアクセスする前にこれらアーキテクチャレジスタの所要データが有効になる（リード・アフタ・ライトの依存性がない）まで待機する。ＡＣ２段の間に、レジスタファイル２４及び単一化キャッシュ６０は、物理アドレスでアクセスされ（キャッシュヒットの場合には、デュアルポート式の単一化キャッシュのためのキャッシュアクセス時間は、レジスタのアクセス時間と同じであり、レジスタセットを効果的に拡張する）、物理アドレスは、リニアアドレスであるか、或いはアドレス変換がイネーブルされた場合には、ＡＴＵ５０によって発生された変換されたアドレスである。

【００４３】

変換されたアドレスは、メモリのページテーブル及びチップ上のワークスペース制御レジスタからの情報を用いてリニアアドレスからＡＴＵ５０によって発生される。単一化キャッシュは、仮想インデックスされると共に物理的にタグが付けられていて、アドレス変換がイネーブルされたときには、変換されていないアドレス（ＡＣ１の終わりに得られる）でセットの選択を行うことができ、そして各セットに対し、ＡＴＵ５０からの変換されたアドレス（ＡＣ２において初期に得られる）でタグの比較を行うことができる。セグメント化及び／又はアドレス変換違反のチェックも、ＡＣ２で行われる。

【００４４】

命令は、それらが例外を生じないと決定されるまではプログラム順序に保たれる。ほとんどの命令に対し、この決定は、ＡＣ２の間又はその前に行われ、浮動小数点命令及びある排他的命令は、実行中に例外を生じることがある。命令は、ＡＣ２からＥＸへ順次に通され（又は浮動小数点命令の場合はＦＰＵ７０へ）、ＥＸにおいて依然として例外を生じることのある整数命令は、排他的と示され、それ故、単独で両方の実行パイプへ発生されるので、しかるべき順序での例外の取り扱いが確保される。

【００４５】

実行段ＥＸＸ及びＥＸＹは、命令により定められた動作を実行する。命令は、ＥＸにおいて可変数のクロックを消費し、即ち順序がずれて実行することが許される（順序ずれ完了）。両方のＥＸ段は、加算、論理及びシフト機能ユニットを備え、そして更に、ＥＸＸ段は、乗算／除算ハードウェアを含む。

【００４６】

ＷＢ段は、レジスタファイル２４、条件コード、及びマシン状態の他の部分を既に実行された命令の結果で更新する。レジスタファイルは、ＷＢのフェーズ１（ＰＨ１）に書き込まれ、そしてＡＣ２のフェーズ２（ＰＨ２）に読み取られる。

【００４７】

１．２　システム

図３を参照すれば、模範的な実施例として、マイクロプロセッサ１０は、単一チップのメモリ及びバスコントローラ８２を含むプロセッサシステムに使用される。メモリ／バスコントローラ８２は、マイクロプロセッサと外部メモリサブシステム－－レベル２キャッシュ８４及びメインメモリ８６－－との間のインターフェイスを形成し、６４ビットプロセッサデータバス（ＰＤ）上でのデータの移動を制御する（データ路はコントローラの外部であり、ピンの本数及びコストを低減する）。

【００４８】

コントローラ８２は、３２ビットアドレスバスＰＡＤＤＲに直接インターフェイスし、コントローラ内のレジスタを読み取ったり書き込んだりするための１ビット巾のデータポート（図示せず）を備えている。両方向性の分離バッファ８８は、マイクロプロセッサ１０と、ＶＬ及びＩＳＡバスとの間のアドレスインターフェイスを形成する。

【００４９】

コントローラ８２は、ＶＬ及びＩＳＡバスインターフェイスの制御を行う。ＶＬ／ＩＳＡインターフェイスチップ９１（ＨＴ３２１のような）は、３２ビットＶＬバス及び１６ビットＩＳＡバスへの標準インターフェイスを形成する。ＩＳＡバスは、ＢＩＯＳ９２、キーボードコントローラ９３、Ｉ／Ｏチップ９４及び標準ＩＳＡスロット９５へインターフェイスする。インターフェイスチップ９１は、二重の高／低ワード〔３１：１６〕／〔１５：０〕分離バッファにより形成された両方向性３２／１６マルチプレクサ９６を経て３２ビットＶＬバスへインターフェイスする。ＶＬバスは、標準ＶＬスロット９７へインターフェイスし、そして両方向性分離バッファ９８を経て６４ビットプロセッサデータバスの下位ダブルワード〔３１：０〕へインターフェイスする。

【００５０】

２．一般化されたパイプラインアーキテクチャ

図４は、パイプライン当たり４命令の性能の一例を示すもので、２パイプラインアーキテクチャに対するオーバーラップした命令実行を示している。付加的なパイプラインと、各パイプラインに対する付加的な段とを設けることができる。好ましい実施例において、マイクロプロセッサ１０の内部動作は、外部システムクロック信号１２４の倍数である周波数における内部クロック信号１２２と同期される。図４において、内部クロック信号１２２は、システムクロック信号１２４の周波数の２倍である。第１の内部クロックサイクル１２６の間に、第１段階の命令デコード段ＩＤ１は、各命令Ｘ０及びＹ０に基づいて動作する。第２の内部クロックサイクル１２８の間に、命令Ｘ０及びＹ０は、第２段階の命令デコード段ＩＤ２へ進んでおり、第１段階の命令デコードユニットＩＤ１には新たな命令Ｘ１及びＹ１が入れられる。第３の内部クロックサイクル１３０の間には、第１段階のデコード段ＩＤ１には命令Ｘ２、Ｙ２があり、第２段階の命令デコード段ＩＤ２には命令Ｘ１、Ｙ１があり、そして第１のアドレス計算ユニットＡＣ１には命令Ｘ０、Ｙ０がある。内部クロックサイクル１３２の間には、第１段階の命令デコード段ＩＤ１には命令Ｘ３、Ｙ３があり、第２段階の命令デコード段ＩＤ２には命令Ｘ２、Ｙ２があり、第１のアドレス計算ユニットＡＣ１には命令Ｘ１、Ｙ１があり、そして第２のアドレス計算ユニットＡＣ２には命令Ｘ０、Ｙ０がある。

【００５１】

この説明から明らかなように、Ｘ及びＹパイプラインの各段を経て次々の命令が順次に流れ続ける。クロックサイクル１３４、１４０に示されたように、各命令の実行部分は、順次のクロックサイクルにおいて実行される。これは、個々の命令の実行時間を減少することなく、クロック当たりに完了される命令の数が増加されるという点で、パイプライン式アーキテクチャの主たる利点である。従って、ハードウェアの速度に大きな需要が生じるようにして大きな命令スループットが達成される。

【００５２】

図４に示す命令の流れは、最適な場合である。図示されたように、１クロックサイクルより大きい段階は必要とされない。実際のマシンでは、他のパイプ段を通る命令の流れを変更するように、１つ以上の段が付加的なクロックサイクルの完了を必要とすることがある。更に、一方のパイプラインを通る命令の流れは、他方のパイプラインを通る命令の流れによって左右される。

【００５３】

３．書き込みバッファアーキテクチャ及び動作

図１に示すように、書き込みバッファ２９は、コア２０の出力に論理的に配置されており、ライトバックバスＷＢ＿ｘ、ＷＢ＿ｙによってコア２０に作動的に接続され、そこからデータを受け取る。又、書き込みバッファ２９は、ＡＴＵ５０にも作動的に接続され、そ

こからアドレスバスＰＡｘ、ＰＡｙ（図５）を経て物理アドレスを受け取る。書き込みバッファ２９の出力は、デュアルキャッシュポート１６０によって単一化キャッシュ６０へ与えられると共に、メモリデータバスＤＡＴＡにも与えられる。キャッシュポート１６０は、従来の仕方で、データ、アドレス及び制御ラインを単一化キャッシュ６０へ与え、本発明の好ましい実施例によれば、キャッシュポート１６０と単一化キャッシュ６０との間のライン数は、２つの同時の書き込み要求をサポートするのに充分なものである。

【００５４】

以下で更に明らかとなるように、書き込みバッファ２９の機能は、レジスタファイル２４のレジスタの１つではなくてメモリへ書き込まれるべきアドレス及びデータ情報をコア２０から受け取ることであり、次いで、書き込みバッファ２９に記憶されたアドレス及びデータ情報は、後で、キャッシュ及びメモリサブシステムが優先順位の高い動作においてビジーでないときに、メモリに書き込むことができる。その結果、書き込みバッファ２９は、コア２０がメモリ書き込み動作を迅速に行い（その観点から）そしてパイプラインの次の命令へ進むことができるようにし、しかも、メモリ読み取り動作を妨げることがなく、且つメモリ書き込みを行うためにコア２０の部分に待機状態を要求することもない。更に、コア２０により書き込みバッファ２９に対して行われるメモリ書き込み動作は、メモリ位置が単一化キャッシュ６０にあるかメインメモリ８６にあるかに係わりなく同じ書き込みサイクルタイムを必要とするものである。

【００５５】

図５を参照し、本発明の好ましい実施例による書き込みバッファ２９の詳細な構造及び動作について説明する。以下に述べる書き込みバッファ２９の例は、スーパーパイプライン式スーパースカラーアーキテクチャのマイクロプロセッサ１０に特に効果的であるが、異なるアーキテクチャのマイクロプロセッサに使用したときにも顕著な性能及び他の利点を与えることを理解されたい。

【００５６】

本発明の好ましい実施例によれば、書き込みバッファ２９は、２つの区分１５２ｘ及び１５２ｙに編成された１２個のエントリ$152x_0$ないし$152x_5$、$152y_0$ないし$152y_5$を含んでいる。この例における書き込みバッファ２９のこの分割編成は、書き込みバッファの区分１５２ｘ及び１５２ｙが各々コア２０のＸ及びＹパイプラインに組み合わされた状態で、レイアウト上好ましいと共に、マイクロプロセッサ１０のスーパースカラーアーキテクチャとの通信効率上も好ましいものである。或いは又、書き込みバッファ２９は、各エントリがコア２０のＸ及びＹパイプラインのいずれかによってアクセスできるような単一バンクとして編成することもできる。

【００５７】

書き込みバッファ２９は、更に、書き込みバッファ制御ロジック１５０も備えており、これは、以下に述べるように、書き込みバッファ２９、及びコア２０とのインターフェイスを制御するように特に設計された組み合わせ又は逐次のロジックである。当業者であれば、本明細書を参照したときに、これらの機能を実行するロジックを容易に実現できることが意図され、このような書き込みバッファ制御ロジック１５０が図５にブロック形態で示されている。

【００５８】

ここで、図６を参照し、書き込みバッファ区分１５２ｘの単一エントリ$152x_i$の内容について説明する。もちろん、書き込みバッファ区分１５２ｙの各エントリ$152y_i$も、本発明の好ましい実施例により同様に構成されることを理解されたい。各エントリ$152x_i$は、アドレス部分、データ部分及び制御部分を含んでいる。更に、書き込みバッファ２９の１２のエントリ１５２の各々を独特に識別するには４ビットで充分であるから、各エントリ１５２は、４ビットタグ値（図示せず）で識別される。このタグは、特定のエントリ１５２をアドレスするようにコア２０によって使用され、パイプラインのＥＸ段及びＷＢ段の間にそこにデータを書き込む（又はそこからデータを供給する）ようにする。４ビットタグを使用することにより、コア２０は、パイプラインの残り部分を通して書き

込みの物理メモリアドレスを維持する必要はない。

【００５９】

３２ビット整数アーキテクチャのマイクロプロセッサ１０の場合に、各エントリ１５２ｘ$_i$は、物理メモリアドレス（ＡＴＵ５０から物理アドレスバスＰＡｘを経て受け取った）の記憶のための３２ビットと、４バイトデータワードの記憶のための３２ビットとを含んでいる。又、本発明の好ましい実施例によれば、各エントリ１５２ｘ$_i$は、更に、以下のテーブルＡに示すように定義された２３の種々の制御ビットも含んでいる。これらの制御ビットは、エントリ１５２の割り当て及び発生を制御するために書き込みバッファ制御ロジック１５０により使用される。更に、単一化キャッシュ６０の制御ロジックのようなマイクロプロセッサ１０の他の部分も、それらの特定の機能を実行するために必要に応じてこれらの制御ビットをアクセスすることができる。各制御ビットの特定の機能については、書き込みバッファ２９の動作に対して以下に詳細に説明する。

テーブルＡ

ＡＶ　　アドレス有効；エントリは有効なアドレスを含む。

ＤＶ　　データ有効；エントリは有効なデータを含む。

ＲＤ　　読み取り可能；エントリはその物理アドレスに対しパイプラインにおける最後の書き込みである。

【００６０】

ＭＲＧ　　合併可能；エントリは手前の書き込みバッファエントリに相接しておりオーバーラップしていない。

ＮＣ　　非キャッシュ式書き込み。

ＦＰ　　エントリは浮動小数点データに対応している。

ＭＡＷ　　不整列書き込み。

ＷＢＮＯＰ　　書き込みバッファノーオペレーション。

ＷＡＲ　　ライト・アフタ・リード；エントリはプログラム順序において他のパイプラインにおける同時読み取りよりも後で生じる書き込みである。

ＳＰＥＣ　　エントリに対する推論の順序を示す４ビットフィールド。

ＸＤＥＰ　　書き込みバッファ区分１５２ｙのクロス依存性マップ。

ＳＩＺＥ　　書き込まれるべきデータのサイズ、バイト数。

ＮＣＲＡ　　非キャッシュ式読み取りが既に割り当てられた。

【００６１】

書き込みバッファ区分１５２ｘは、Ｘパイプラインの実行段ＥＸＸ又はＹパイプラインの実行段ＥＸＹのいずれかの結果を、コア２０により駆動されるライトバックバスＷＢ__ｘを経て受け取り、そして同様に、書き込みバッファ区分１５２ｙは、Ｘパイプラインの実行段ＥＸＸ又はＹパイプラインの実行段ＥＸＹのいずれかの結果をライトバックバスＷＢ__ｙを経て受け取る。

【００６２】

書き込みバッファ区分１５２ｘ、１５２ｙは、それらの内容（アドレス及びデータの両区分）を、例えば、データを適切にフォーマットする回路を経てキャッシュポート１６０へ与える。図５に示すように、書き込みバッファ区分１５２ｘは、そのデータをバレルシフタ１６４ｘへ与え、これは、次いで、その出力を不整列書き込みラッチ１６２ｘへ与える。以下で詳細に述べるように、この不整列書き込みラッチ１６２ｘは、キャッシュポート１６０への第２の書き込みのために書き込みバッファ区分１５２ｘからのデータを記憶することができ、この第２の書き込みは、本発明によれば、メモリへの書き込みが８バイト境界にオーバーラップする場合に実行される。この不整列書き込みラッチ１６２ｘは、その出力をキャッシュポート１６０へ与えると共に、書き込み集合ラッチ１６５へも与え、この書き込み集合ラッチ１６５は、以下で詳細に述べるように、多数の書き込みの物理アドレスが同じ８バイトグループ内にある場合に多数の書き込みバッファエントリ１５２からのデータを集合してキャッシュポート１６０へ単一書き込みするように働く。

【００６３】

書き込みバッファ区分１５２ｙは、その出力をマルチプレクサ１６３の一方の入力に与え、該マルチプレクサは、浮動小数点データラッチ１６６の出力をその他方の入力に受け取り、以下で述べるように、浮動小数点データラッチ１６６はＦＰＵ７０からの出力を含み、書き込みバッファエントリ１５２の１つに対応するメモリ書き込みに対し６４ビットの浮動小数点データ記憶を与える。マルチプレクサ１６３は、以下に述べるように、その出力に与えるための適当な入力を選択するように、書き込みバッファ制御ロジック１５０によって制御されると共に単一化キャッシュ６０のためのキャッシュ制御ロジックによって制御される。マルチプレクサ１６３の出力は、シフタ１６４ｙへ送られ、次いで、上記の書き込みバッファ区分１５２ｘの出力と同様に、不整列書き込みラッチ１６２ｙへ送られる。又、不整列書き込みラッチ１６２ｙの出力も、キャッシュポート１６０へ同様に直結され、そして書き込み集合ラッチ１６５にも接続される。

【００６４】

説明を簡単にするために、図５には１つのキャッシュポート１６０が概略的に示されているだけであるが、上記したように、本発明のこの実施例によるキャッシュポート１６０は、２つの書き込み要求を同時に与えることができるデュアルキャッシュポートである。更に、書き込みバッファ２９は、データバスＤＡＴＡにデータを直接通信する。このように、本発明のこの実施例によれば、図５に示されたキャッシュポート１６０の接続は、キャッシュポート１６０に第２の同時の書き込みを与えるように複製されると共に、データバスＤＡＴＡにも直結されて、キャッシュ制御がメインメモリ８６への書き込みを要求する場合にメモリ書き込みを行うようにされる。

【００６５】

又、本発明の好ましい実施例によれば、書き込みバッファ２９は、マルチプレクサ１５４ｘ、１５４ｙを制御する書き込みバッファ制御ロジック１５０の制御のもとで、そのエントリ１５２からソースバスＳＲＣｘ、ＳＲＣｙによりコア２０へデータを直接供給することができる。マルチプレクサ１５４ｘの出力は、パイプラインコントローラ２８の制御のもとで、物理レジスタ２４へのバスｍｅｍ＿ｘ、ｍｅｍ＿ｙを経て、Ｘ又はＹパイプラインのいずれかへ送られ、同様に、マルチプレクサ１５４ｙの出力は、バスｍｅｍ＿ｘ、ｍｅｍ＿ｙを経てＸ又はＹパイプラインのいずれかへ送られる。更に、ライトバックバスＷＢ＿ｘ、ＷＢ＿ｙも、バイパスバスＢＰ＿ｘ、ＢＰ＿ｙを各々経てマルチプレクサ１５４ｘ、１５４ｙに接続され、これにより、以下に述べるように、書き込みバッファ２９のメモリバイパスが容易にされる。

【００６６】

上記したように、マイクロプロセッサ１０は、浮動小数点演算を実行するためのオンチップＦＰＵ７０を備えている。上記したように、ＦＰＵ７０によって行われた計算の結果は、６４ビットデータワードで表される。本発明のこの好ましい実施例によれば、書き込みバッファエントリ１５２のデータ部分を３２ビットに制限し、そしてＦＰＵ７０からデータを受け取るための６４ビット浮動小数点データラッチ１６６を設けることにより、効率が得られる。浮動小数点データラッチ１６６は、更に、浮動小数点データ有効（ＦＰＤＶ）制御ビットを含み、このビットは、セットされたときに、浮動小数点データラッチ１６６の内容が有効データを含むことを指示する。１つの書き込みバッファエントリ１５２のアドレス部分は、浮動小数点データラッチ１６６に記憶されたＦＰＵ７０からの結果を書き込むべきところのメモリアドレスを含み、この書き込みバッファエントリ１５２は、そのＦＰ制御ビットがセットされると、そのデータ部分が有効データを含まないが、それに対応するデータが浮動小数点データラッチ１６６に存在することを指示する。

【００６７】

或いは又、もちろん、浮動小数点データ書き込みバッファ機能は、各書き込みバッファエントリ１５２に対して６４ビットデータ部分を与えることにより得ることができる。しかしながら、本発明のこの実施例では、６４ビット浮動小数点データのキャッシュ前書き込みバッファ機能は与えられるが、レイアウト及びチップ面積は著しく非効率的なものとなる。この非効率性は、各書き込みバッファエントリ１５２が６４ビットデータ部分をもつ

ことを必要とせず、むしろ、浮動小数点ラッチ１６６が書き込みバッファ２９の各エントリ１５２に対して６４ビット能力を与えることにより生じる。ほとんどの用途では、ＦＰＵ７０によって浮動小数点データが与えられる頻度は、浮動小数点データラッチ１６６から浮動小数点データがリタイアされる（即ち、キャッシュ又はメモリへ書き込まれる）のと同じ程度であることが意図される。これは、図５に示された単一の浮動小数点データラッチ１６６で充分なバッファ機能を与えることができるようにする。もとろん、それとは別に、更にバッファ機能が所望される場合には、多数の浮動小数点データラッチ１６６をマイクロプロセッサ１０に設けることができる。

【００６８】

本発明の好ましい実施例による書き込みバッファ２９の動作を以下に詳細に説明する。この動作は、以下に述べる機能を実行するように構成された組み合わせ又は逐次ロジックである書き込みバッファ制御ロジック１５０の制御のもとにある。上記したように、当業者であれば、以下の説明に基づいて書き込みバッファ制御ロジック１５０の機能を達成するように、このようなロジックを容易に実施することができよう。

【００６９】

より詳細には、本発明のこの実施例によれば、書き込みバッファ制御ロジック１５０は、Ｘ及びＹ割り当てポインタ１５６ｘ、１５６ｙと、Ｘ及びＹリタイアポインタ１５８ｘ、１５８ｙを各々備えており、これらポインタ１５６、１５８は、各々次に割り当て又はリタイアされるべき書き込みバッファ２９のエントリ１５２を追跡する。従って、書き込みバッファ２９の区分１５２ｘ、１５２ｙの各々は、割り当て及びリタイアの目的で円形バッファとして動作し、そしてデータを発生する目的でアドレス可能なレジスタのファイルとして動作する。或いは又、書き込みバッファ２９は、もし所望ならば、完全に連想的な一次データキャッシュとして実施されてもよい。

【００７０】

一般に、第２のアドレス計算段ＡＣ２が、命令の実行中にメモリ書き込みが行われることを決定すると、この段において物理アドレスが計算されたときに、書き込みバッファエントリ１５２の１つが「割り当てられ」、その物理アドレスがエントリ１５２のアドレス部分に記憶され、そしてそのアドレス有効制御ビット（ＡＶ）及び他の適当な制御ビットがセットされる。命令の実行後であって且つＷＢＸ、ＷＢＹ（図１）の間に、コア２０は、その結果をその書き込みバッファエントリ１５２のデータ部分に書き込んで、書き込みバッファエントリを「発生(issue)」し、データ有効制御ビット（ＤＶ）をセットする。書き込みバッファエントリ１５２は、選択されたエントリ１５２のＡＶ及びＤＶビットに質問し、そして両方がセットされた場合は、エントリ１５２のアドレス及びデータ部分の内容が場合によってキャッシュポート１６０又はシステムバスに現れるようにすることにより、非同期でプログラム順序で「リタイア」される。

【００７１】

３．１　書き込みバッファエントリの割り当て

図７を参照し、本発明の好ましい実施例による書き込みバッファエントリ１５２の割り当てプロセスを詳細に説明する。本発明のこの実施例では、割り当てプロセスは、Ｘ及びＹの両パイプラインにおいて第２のアドレス計算段ＡＣ２の部分として実行される。図７のプロセス１７０で示されたように、割り当てプロセスは、命令の結果を書き込むべき（即ち、メモリ書き込み）物理メモリアドレスを計算する際に開始される。

【００７２】

説明を容易にするために、図７のシーケンスは、書き込みバッファ２９の区分１５２ｘ、１５２ｙの一方について説明する。書き込みバッファ２９の反対の区分における書き込みバッファエントリ１５２の割り当ては、図７に示したものと同じである。

【００７３】

物理アドレスが計算されると、プロセス１７２は、割り当てポインタ１５６が指している書き込みバッファエントリ１５２から制御ビットＡＶを検索する。本発明のこの実施例によれば、書き込みバッファ２９の各側は、円形バッファとして動作し、割り当てポインタ

１５６ｘ、１５６ｙは、Ｘ及びＹパイプラインに対して各々割り当てられるべき次の書き込みバッファエントリ１５２を指示し、この説明上、適当な割り込みポインタ１５６ｘ、１５６ｙが指す書き込みバッファエントリ１５２を、１５２_{n}と称する。判断ステップ１７３は、制御ビットＡＶがセットされている（１）かクリアされている（０）かを判断する。制御ビットＡＶが既にセットされている場合には、書き込みバッファエントリ１５２_{n}は、有効アドレスを既に記憶しているので、既に割り当てられているか又は保留中である。従って、エントリ１５２_{n}は、このときには割り当てに使用できず、待機状態１７４に入り、その後、プロセス１７２及び判断１７３において次のエントリ１５２_{n+1}に対して制御ビットＡＶが繰り返し検索されてチェックされる。

【００７４】

判断ステップ１７３で、エントリ１５２_{n}の制御ビットＡＶがクリアされていると判断された場合に、そのエントリ１５２_{n}は、既に割り当てられてもいないし保留中でもないから、割り当てに使用できる。この場合、プロセス１７６は、プロセス１７０で計算された物理アドレスをエントリ１５２_{n}のアドレス部分に記憶する。

【００７５】

図７に示すプロセス１７６ないし１８８の特定の順序は、一例に過ぎない。これらのプロセスは、当業者による特定の実現化にとって効果的又は適当であると思われるいかなる順序で行われてもよいことが意図される。

【００７６】

３．１．１　リード・アフタ・マルチプルライトハザードの処理

本発明のこの実施例によれば、あるデータ依存性が検出されそして書き込みバッファアクセスに対して処理される。この技術分野で良く知られているように、データ依存性とは、パイプライン式アーキテクチャのマイクロプロセッサにおいてプログラム結果にエラーを生じることのある一種のハザードである。これらの依存性は、スーパースカラー、スーパーパイプライン式アーキテクチャのマイクロプロセッサ１０において、特に幾つかの命令が性能向上のためにプログラム順序から外れて実行される場合に、より顕著なものとなる。より詳細には、図５について上記しそして以下で詳細に述べるように、書き込みバッファ２９は、パイプラインにおいてその後の命令に対してデータが必要となる場合にはエントリをリタイアする前にバスＳＲＣｘ、ＳＲＣｙを経てコア２０へデータを送ることができる。書き込みバッファエントリ１５２の読み取り可能な制御ビット（ＲＤ）は、同じ物理アドレスへの多数の書き込みの後であって且つこのアドレスに指定された書き込みバッファエントリ１５２のリタイアの前に実行されるべき物理メモリアドレスの読み取りをパイプラインが含むような特殊な形式のリード・アフタ・ライト（ＲＡＷ）依存性の処理を助成する。本発明の好ましい実施例によれば、制御ビットＲＤがセットされた書き込みバッファエントリ１５２のみを使用して、バスＳＲＣｘ、ＳＲＣｙを経てコア２０へデータを送ることができる。これは、同じ物理アドレスに対して後で割り当てられたがまだ実行されていない書き込み動作からではなくて、完了したその前の書き込みからコア２０へ誤ったデータが送られるおそれを回避する。

【００７７】

プロセス１７８において、書き込みバッファ制御ロジック１５０は、既に割り当てられた各書き込みバッファエントリ１５２のアドレスフィールドを検査し、エントリ１５２_{n}に割り当てられるべき物理アドレスに一致するかどうかを決定する。本発明の好ましい実施例によれば、各読み取り又は書き込み動作のサイズが８バイト程度であり（浮動小数点データを書き込むべき場合。本発明のこの実施例では、整数データの場合は、４バイト）、そして各物理アドレスが１バイトに対応することを考慮すれば、プロセス１７８において物理アドレス値を比較するだけでなく、各動作のメモリスパンを考慮しなければならない。この構成のために、異なる物理アドレスを有する書き込み動作が、その動作のサイズに基づいて同じバイトにオーバーラップする。

【００７８】

図８を参照し、本発明の好ましい実施例により異なるメモリアクセス命令の物理アドレス

をプロセス１７８において比較する方法について詳細に説明する。２つの書き込み動作の書き込みスパンを比較するために、パイプライン制御ロジック２８は、第１スパンマップＳＰＡＮ$_0$に、古い書き込み命令の書き込み動作が作用するところのバイトの相対位置に対応するビットがセットされたビットマップをロードすると共に、第２スパンマップＳＰＡＮ$_1$に、新たな書き込み命令の書き込み動作が作用するところのバイトの位置に対応するビットがセットされたビットマップをロードする。図８は、２つの例示的な書き込み動作に対するスパンマップＳＰＡＮ$_0$、ＳＰＡＮ$_1$の例を示している。プロセス１７８は、次いで、スパンマップＳＰＡＮ$_0$とＳＰＡＮ$_1$のビットごとの論理積をとり、両方の書き込み動作によって書き込まれるバイトの位置をセットされたビットで指示するマップＡＮＤＳＰＡＮを形成する。図８の例では、マップＡＮＤＳＰＡＮにおいて２つのビットがセットされており、２つの例示的な書き込み動作の両方が２つのバイトに書き込むことを示している。

【００７９】

プロセス１７８は、次いで、マップＡＮＤＳＰＡＮにおけるビットの論理和をとり、いずれかのビットがセットされたかどうかを判断する。エントリ１５２$_n$に対する読み取り可能な制御ビットはセットされ（一致するエントリが見つかるかどうかに係わりなく）そしてマップＡＮＤＳＰＡＮにおけるビットの論理和の結果を真とさせた既に割り当てられた書き込みバッファエントリ１５２に対する制御ビットＲＤはクリアされる。従って、以下に述べるように、書き込みバッファ２９を後で読み取るべきである（即ち、リタイアの前に書き込みバッファ２９からデータを供給する）場合には、最後に書き込まれた書き込みバッファエントリ１５２$_n$のみがその制御ビットＲＤをセットし、従って、そのデータをソースバスＳＲＣｘ、ＳＲＣｙを経てコア２０へ与えることができる。有効データを有する（制御ビットＤＶがセットされた）がそれらの制御ビットＲＤがクリアされた書き込みバッファエントリ１５２は、書き込みバッファ制御ロジック１５０により、それらのデータをバスＳＲＣｘ、ＳＲＣｙへ送らないよう防止される。

【００８０】

３．１．２　クロス依存性及びプログラム順序でのリタイア

上記したように、書き込みバッファエントリ１５２は、プログラム順序でリタイア（即ち、単一化キャッシュ６０又はメインメモリ８６へ書き込み）されねばならない。書き込みバッファエントリ１５２の単一のバンクしか使用されない本発明のこれらの実施については、単一のリタイアポインタ１５８によってプログラム順序が容易に維持される。しかしながら、マイクロプロセッサ１０はスーパースカラーアーキテクチャであるので、書き込みバッファ２９の実現においてレイアウトの効率を得るためには、上記したように、本発明のこの例では、書き込みバッファエントリ１５２が、Ｘ及びＹパイプラインの各々に対して１つづつの２つのグループに分割され、その各々がそれ自身のリタイアポインタ１５８ｘ、１５８ｙを各々有する。本発明のこの好ましい実施例は、Ｘ区分の書き込みバッファエントリ１５２ｘとＹ区分の書き込みバッファエントリ１５２ｙとの間にプログラム順序でのリタイアを確保する技術を与える。

【００８１】

図９を参照し、割り当て時における選択された書き込みバッファエントリ１５２ｘ$_i$に対するクロス依存性制御ビットＸＤＥＰのマップを説明する。図９に示すように、書き込みバッファ２９のＸ区分における各書き込みバッファエントリ１５２ｘ$_i$は、書き込みバッファ２９のＹ区分１５２ｙにおける書き込みバッファエントリ１５２ｙ$_i$の１つに各々対応する６個のクロス依存性制御ビットＸＤＥＰ$_0$ないしＸＤＥＰ$_5$を有し、同様に（図９には示さないが）、各書き込みバッファエントリ１５２ｙ$_i$は、書き込みバッファ２９のＸ区分１５２ｘにおける書き込みバッファエントリ１５２ｘ$_i$の各々に対して１つづつ６個のクロス依存性制御ビットＹＤＥＰ$_0$ないしＹＤＥＰ$_5$を有する。図９に示すように、書き込みバッファエントリ１５２ｘ$_i$に対する各クロス依存性ビットＸＤＥＰの内容は、割り当て時において、書き込みバッファ２９のＹ区分１５２ｙの対応する書き込みバッファエントリ１５２ｙ$_i$のための制御ビットＡＶの状態に対応する。

【００８２】
図７の割り当てプロセスにおけるプロセス１８０は、現在割り当てられている書き込みバッファエントリ１５２$_n$ に対するクロス依存性制御ビットＸＤＥＰ$_0$ ないしＸＤＥＰ$_5$ に、割り込み時に、書き込みバッファ２９のＹ区分１５２ｙにおける６個の書き込みバッファエントリ１５２ｙ$_i$ のためのアドレス有効制御ビットＡＶの状態をロードする。以下に詳細に述べるように、各書き込みバッファエントリ１５２がリタイアされるときには、書き込みバッファ２９の反対の部分における書き込みバッファエントリ１５２の各々の対応するクロス依存性制御ビットＸＤＥＰがクリアされる。更に、書き込みバッファエントリ１５２が割り当てシーケンスのプロセス１８０においてそのクロス依存性制御ビットＸＤＥＰをセットした後には、それ自身のいずれのクロス依存性制御ビットＸＤＥＰも付加的にセットされることはない。

【００８３】
従って、書き込みバッファエントリ１５２をリタイアするために、６個全部のクロス依存性制御ビットＸＤＥＰ$_0$ ないしＸＤＥＰ$_5$ をクリアしなければならない（即ち、０に等しくする）ことを必要とすることにより、プログラム順序が維持される。従って、プロセス１８０においてクロス依存性制御ビットＸＤＥＰをセットすると、既に割り当てられた書き込みバッファ２９の反対部分の書き込みバッファエントリ１５２（即ち、プログラムシーケンスにおいて割り当てられた書き込みバッファエントリ１５２$_n$ の前方）の「スナップショット」が得られる。クロス依存性制御ビットＸＤＥＰとリタイアポインタ１５８ｘ、１５８ｙの組み合わせは、書き込みバッファエントリ１５２がプログラム順序でリタイアされるように確保する。

【００８４】
同様に、以下に詳細に述べるように、マイクロプロセッサ１０は、プログラム順序で行わねばならないメインメモリ８６からの非キャッシュ式読み取りを実行する構成を含むことができる。非キャッシュ式読み取り割り当て制御ビット（ＮＣＲＡ）をセットすることにより各書き込みエントリに対し既に割り当てられた非キャッシュ式読み取りの存在が指示され、非キャッシュ式読み取りを実行すると、全ての書き込みバッファエントリ１５２に対して制御ビットＮＣＲＡがクリアされる。制御ビットＮＣＲＡのセット及びクリアは、上記のクロス依存性制御ビットＸＤＥＰと同様に行われ、非キャッシュ式読み取りが適切なプログラム順序で実行されるよう確保する。

【００８５】
３．１．３　割り当てプロセスの完了
次いで、プロセス１８２が書き込みバッファエントリ１５２$_n$ の割り当てにおいて実行され、行われるべきメモリ書き込みの特定の属性に基づいて書き込みバッファエントリ１５２$_n$ のある制御ビットがセットされる。書き込みサイズ制御ビット（ＳＩＺＥ）は、命令において指示されたように、書き込みバッファエントリ１５２$_n$ に書き込まれるべきデータのバイト数でセットされる（８バイトまで。従って、３つの書き込みサイズ制御ビットＳＩＺＥを必要とする）。

【００８６】
書き込みバッファエントリ１５２$_n$ の使用においてマイクロプロセッサ１０の動作を制御するために、プロセス１８２では書き込みバッファエントリ１５２$_n$ の他の制御ビットもセットされる。これらビットの状態に基づいて本発明のこの実施例で行われる特定の制御を以下に詳細に述べるが、これら制御ビットの性質について以下に要約する。非キャッシュ式書き込み制御ビット（ＮＣ）は、メモリ書き込み動作を非キャッシュ式とすべき場合にセットされる。合併可能な制御ビット（ＭＲＧ）は、書き込みバッファエントリ１５２$_n$ に対し、それに対応する物理メモリ位置が、既に割り当てられた書き込みバッファエントリ１５２$_n$ に対応するメモリ位置に相接しそしてオーバーラップしていない場合にセットされて、集合書き込み動作が行えるようにされる。ライト・アフタ・リード制御ビット（ＷＡＲ）は、書き込みバッファエントリ１５２$_n$ への書き込み動作を他のパイプラインにおける同時読み取りの後に行うべきである場合にセットされる。不整列書き込み制御ビ

ット（ＭＡＷ）は、書き込みバッファエントリ１５２$_n$に記憶された物理アドレスへ書き込まれるべきデータの長さが８バイト境界に交差する場合にセットされる（この場合は、書き込みバッファエントリ１５２$_n$をリタイアするのに２つの書き込みサイクルが必要である）。制御ビットＮＣＲＡは、非キャッシュ式読み取りが既に割り当てられていてまだ実行されていない場合にセットされる。

【００８７】

書き込みバッファエントリ１５２$_n$において物理アドレスの記憶と制御ビットのセットが完了すると、プロセス１８４において書き込みバッファエントリ１５２$_n$に対する制御ビットＡＶがセットされる。更に、手前のリタイア動作によってまだクリアされていない場合には、制御ビットＤＶがこのときクリアされる。制御ビットＡＶをセットすることは、書き込みバッファエントリ１５２$_n$をその後の動作に割り当てることを指示し、これは、書き込みバッファ２９の反対区分において書き込みバッファエントリ１５２を割り当てる際にクロス依存性制御ビットＸＤＥＰをセットすることを含む。

【００８８】

プロセス１８６において、書き込みバッファ制御ロジック１５０は、現在割り当てられた書き込みバッファエントリ１５２$_n$のタグ値をコア２０に返送する。次いで、コア２０は、プロセス１７０で計算された全３２ビットの物理アドレスではなくて、この４ビットタグ値を命令の実行に使用する。短いタグ値を使用することにより、命令の実行が容易になり、従って、マイクロプロセッサ１０の性能が向上される。

【００８９】

割り当てシーケンスはプロセス１８８で完了し、割り当てポインタ１５６ｘ、１５６ｙ（書き込みバッファエントリ１５２$_n$が書き込みバッファ２９のＸ区分１５２ｘにあるかＹ区分１５２ｙにあるかに基づく）は、割り当てられるべき次の書き込みバッファエントリ１５２を指すように増加される。次いで、制御は、書き込みに関連した命令が他の何らかの理由でパイプラインにおいて前方に進むことが禁止されなければ、パイプラインにおける関連ＥＸ段であるプロセス１９０へ進む。

【００９０】

３．２　書き込みデータバッファへのデータの発生

図１０を参照し、書き込みバッファエントリ１５２へデータを発生するプロセスを、選択された書き込みバッファエントリ１５２$_i$について詳細に説明する。上記のように、書き込みバッファ２９へのデータの発生は、命令のＥＸ段の完了後であって且つ動作がＸパイプラインであるかＹパイプラインであるかに基づいてＷＢ段の１つの間に、コア２０によって実行される。

【００９１】

発生シーケンスはプロセス１９２で始まり、コア２０は、書き込みバッファ２９に書き込まれるべきデータを、Ｘ又はＹパイプラインのいずれが命令を実行するかに基づいてライトバックバスＷＢ＿ｘ、ＷＢ＿ｙの適当な１つに出す。又、コア２０は、行き先書き込みバッファエントリ１５２のタグを書き込みバッファ制御ロジック１５０へ通信する。次いで、書き込みバッファ制御ロジック１５０は、プロセス１９４において、与えられたタグ値に関連した書き込みバッファエントリの１つである書き込みバッファエントリ１５２$_i$をイネーブルし、その関連ライトバックバスＷＢ＿ｘ、ＷＢ＿ｙに与えられたデータをラッチする。書き込みバッファエントリ１５２$_i$におけるデータの記憶又はラッチ動作が完了すると、プロセス１９６において制御ビットＤＶがセットされ、発生シーケンスが終わりとなる。

【００９２】

書き込みバッファエントリ１５２$_i$がその制御ビットＡＶ及びその制御ビットＤＶをセットすると、書き込みバッファエントリ１５２$_i$は、「保留」状態となり、リタイアすることができる。上記したように、書き込みバッファエントリ１５２のリタイアは、単一化キャッシュ６０を動作するのに使用されるキャッシュロジックの制御のもとで、非同期で行われ、単一化キャッシュ６０又はメインメモリ８６への書き込みバッファエントリ１５２

の内容の書き込みが使用可能なバスにおいて行われ、キャッシュ又はメインメモリ読み取り動作の実行を中断したり遅延したりすることはない。メモリからのプログラム又はデータの検索の際に実行されるプログラムの依存性のために、メモリ読み取りはメモリ書き込みよりも一般に優先順位が高いことを考えると、書き込みバッファ２９は、従来の技術に対して著しい性能改善を与える。

【００９３】

３．３　書き込みバッファエントリのリタイア動作

図１１を参照し、単一化キャッシュ６０内に含まれるか又はこれと共に設けられたキャッシュ制御ロジックの制御のもとで書き込みバッファエントリ１５２をリタイアするシーケンスについて説明する。幾つかの特殊な又は複雑な書き込み動作を以下に特に詳細に説明する。従って、図１１のリタイアシーケンスは一般化されたシーケンスである。

【００９４】

３．３．１　整数書き込みバッファデータのリタイア動作

上記したように、図１１のリタイアシーケンスは、単一化キャッシュ６０内に含まれるか又はこれと共に設けられたキャッシュ制御ロジックの制御のもとで行われ、これは、Ｘ及びＹパイプラインの動作に対して非同期である。上記のように、書き込みバッファエントリ１５２はプログラム順序でリタイアされるのが重要である。従って、書き込みバッファ２９は、円形バッファとして動作し、書き込みバッファ２９の２つの部分に対しリタイアポインタ１５８ｘ、１５８ｙによってシーケンスが決定される。これらリタイアポインタ１５８ｘ、１５８ｙは、書き込みバッファ２９の対応区分１５２ｘ、１５２ｙにおける書き込みバッファエントリ１５２のプログラム順序を維持し、そしてクロス依存性制御ビットＸＤＥＰは、以下の説明から明らかなように、区分１５２ｘ、１５２ｙ間のエントリ１５２の順序を維持する。

【００９５】

説明を容易にするために、上記の割り当てシーケンスの場合と同様に、図１１のシーケンスは、書き込みバッファ２９の区分１５２ｘ、１５２ｙの一方について説明する。書き込みバッファ２９の反対の区分についてのシーケンスも同一である。

【００９６】

リタイアシーケンスはプロセス２００で始まり、リタイアポインタ１５８がリタイアされるべき次のエントリ１５２として指示する書き込みバッファエントリ１５２の１つである書き込みバッファエントリ１５２_rから制御ビットＡＶが検索される。判断ステップ２０１において、制御ビットＦＰ及び制御ビットＡＶがテストされて、書き込みバッファエントリ１５２_rが浮動小数点データラッチ１６６に関連される（従って、ＦＰＵ７０からの浮動小数点結果をバッファする）かどうかを決定する。制御ビットＦＰ及び制御ビットＡＶの両方がセットされた場合には、書き込みバッファエントリ１５２_rが浮動小数点データに関連され、データは、以下のセクション３．３．２に述べるプロセスに基づいてリタイアされる。

【００９７】

制御ビットＡＶがセットされそして浮動小数点制御ビットＦＰがクリアされた場合には、書き込みバッファエントリ１５２_rは整数データに向けられる。次いで、判断ステップ２０２が実行され、キャッシュ制御ロジックは、制御ビットＡＶ及び制御ビットＤＶの両方がセットされたかどうかを決定する。もしそうでなければ（ＡＶ及びＤＶのいずれかがクリアされた）、エントリ１５２_rはリタイアされる準備がされず、制御はプロセス２００へと進んで、検索及び判断プロセスが繰り返される。その両方がセットされた場合には、有効な整数データが書き込みバッファエントリ１５２_rのデータ部分に現れ、エントリはリタイア可能となる。

【００９８】

次いで、判断ステップ２０４が実行されて、クロス依存性制御ビットＸＤＥＰが書き込みバッファエントリ１５２_rに対して全てクリアであるかどうか判断される。上記したように、クロス依存性制御ビットＸＤＥＰは、書き込みバッファエントリ１５２_rの割り当て

において開始して各書き込みバッファエントリ１５２のリタイアの際に更新される書き込みバッファ２９の反対区分における書き込みバッファエントリ１５２に対する制御ビットＡＶのスナップショットである。全てのクロス依存性制御ビットＸＤＥＰが書き込みバッファエントリ152_rに対してクリアされ（そしてリタイアポインタ１５８がそれを指し）た場合には、書き込みバッファエントリ152_rがプログラム順序において次にリタイアされるべきものとなり、制御はプロセス２０８へ進む。

【００９９】

クロス依存性制御ビットＸＤＥＰが全てクリアされない場合には、書き込みバッファ２９の反対区分における付加的な書き込みバッファエントリ１５２は、プログラム順序を維持するように、エントリ１５２ｙがリタイアされる前にリタイアされねばならない。待機状態２０６が作用され、その後、書き込みバッファエントリ152_rの割り当ての前に割り当てられた反対区分における書き込みバッファエントリ１５２が先にリタイアされるまで、判断ステップ２９４が繰り返される。

【０１００】

以下で詳細に述べるように、マイクロプロセッサ１０は、メインメモリ８６から非キャッシュ式読み取りを行う構成を含んでもよく、これはプログラム順序で行われねばならない。既に割り当てられた非キャッシュ式読み取りの存在は、各書き込みエントリごとに、制御ビットＮＣＲＡがセットされることにより指示され、非キャッシュ式読み取りを実行する際には、制御ビットＮＣＲＡが全ての書き込みバッファエントリ１５２に対してクリアされる。この特徴が実施される場合には、判断ステップ２０４は、制御ビットＮＣＲＡの状態もテストし、全てのクロス依存性制御ビットＸＤＥＰ及び制御ビットＮＣＲＡの両方がクリアされるときまで書き込みバッファエントリ152_rのリタイアを防止する。

【０１０１】

次いで、プロセス２０８が実行され、書き込みバッファエントリ152_rのデータ部分は、キャッシュポート１６０又はメモリバスへ送るように適当なビット又はバイト位置と整列される。この整列は、物理メモリアドレスは特定のバイト位置に対応しているが、データは６４ビット（８バイト）までのワードで与えられることを考慮して必要とされる。従って、適切なメモリ書き込み動作を確保するには、データと適当なビット位置との整列が重要である。更に、集合書き込み及び不整列書き込みに必要とされるような特殊な整列動作がプロセス２０８において達成される。これら整列特徴及びシーケンスの詳細は、以下で説明する。

【０１０２】

次いで、プロセス２１０は、書き込みバッファエントリ152_rのデータ部分を、直接的に又は図５に示された特殊な書き込み回路を経てキャッシュポート１６０へ送る。これが生じると、書き込みバッファエントリ152_rに対応するクロス依存性制御ビットＸＤＥＰの１つが、書き込みバッファ２９の反対区分の各書き込みバッファエントリ152_iにおいてクリアされる（プロセス２１２）。これは、シーケンスにおける次の書き込みバッファエントリ１５２（即ち反対のリタイアポインタ１５８により指示された書き込みバッファエントリ152_i）を次の動作においてリタイアできるようにする。プロセス２１４は、現在リタイアされている書き込みバッファエントリ152_rに対し制御ビットＡＶ及び制御ビットＤＶをクリアする。次いで、プロセス２１６は、その区分に対するリタイアポインタ１５８を増加し、シーケンスにおける次の書き込みバッファエントリ１５２をリタイアできるようにすると共に、書き込みバッファエントリ152_rを再割り当てできるようにする。リタイアシーケンスの制御は、プロセス２００へ戻り、適当な制御ビットを検索する。

【０１０３】

上記したように、単一キャッシュポート１６０は、図５に概略的に示されて、上記のプロセス２１０に対して説明したが、このキャッシュポート１６０はデュアルキャッシュポートとして働き、そして図１のマイクロプロセッサ１０の書き込みバッファ２９は、データバスＤＡＴＡと直接通信する。従って、この場合には、キャッシュ制御ロジックは、書き

込みバッファ２９がプロセス２１０においてエントリ１５２$_r$からデータを与えるところの適当なポートを選択する。

【０１０４】

更に、デュアルキャッシュポート１６０を設けたことにより、図５に示したように書き込みバッファ２９の２つの区分が設けられる場合に、データが２つの書き込みバッファエントリ１５２（書き込みバッファ２９のＸ及びＹ区分１５２ｘ及び１５２ｙの各々に１つ）からデュアルキャッシュポート１６０を経て同時に与えられるときに、付加的な流線型化を与えることができる。このように同時にデータが与えられる場合は、クロス依存性の判断ステップ２０４は、同時に与えられた書き込みバッファエントリ１５２が、セットされたＸＤＥＰビットに対応する限り、書き込みバッファエントリ１５２の１つが単一のセットされたクロス依存性制御ビットＸＤＥＰをもつことを許さねばならない。従って、リタイアプロセスは、書き込みバッファ２９の２つの区分１５２ｘ、１５２ｙを使用することによりその出力速度を倍増できる。

【０１０５】

３．３．２　浮動小数点書き込みバッファデータのリタイア動作

判断ステップ２０１で制御ビットＡＶ及び制御ビットＦＰの両方がセットされたと判断した場合は、リタイアポインタ１５８が指す書き込みバッファエントリ１５２$_r$がＦＰＵ７０からの浮動小数点結果に関連している。本発明のこの実施例によれば、エントリ１５２$_r$の制御ビットＤＶは、以下で述べる例外処理の目的で、有効な整数データが存在しなくてもセットされる。

【０１０６】

次いで、判断ステップ２０３が実行され、キャッシュ制御ロジックが浮動小数点データラッチ１６６の制御ビットＦＰＤＶに質問して、ＦＰＵ７０がそこにデータを書き込んだかどうかを調べ、もしそうであれば、制御ビットＦＰＤＶがセットされる。この制御ビットＦＰＤＶは、セットされたときに、ＦＰＵ７０が有効なデータを書き込んだことを指示するので、書き込みバッファエントリ１５２の制御ビットＤＶと同様である。これに対し、制御ビットＦＰＤＶがクリアされた場合には、ＦＰＵ７０は浮動小数点データラッチ１６６にまだデータを書き込んでおらず、この場合には、判断ステップ２０４が図１１のリタイアシーケンスのプロセス２００に制御を戻す。

【０１０７】

制御ビットＦＰＤＶがセットされた場合に、判断ステップ２０５が実行され、書き込みバッファエントリ１５２$_r$のクロス依存性制御ビットＸＤＥＰが質問されて、全てのビットＸＤＥＰがクリアされたかどうか調べられる。もしそうでなければ、プログラム順序でエントリ１５２$_r$の前に割り当てられていてエントリ１５２$_r$とは反対の書き込みバッファ２９の区分に存在する付加的な書き込みバッファエントリ１５２を、そのエントリ１５２$_r$のリタイアの前にリタイアしなければならない。次いで、待機状態２０７が実行され、判断１０５が繰り返される。エントリ１５２$_r$の全てのクロス依存性制御ビットＸＤＥＰがクリアになると、判断２０５は制御をプロセス２０８へ進め、浮動小数点データラッチ１６６の内容を整列してキャッシュポート１６０へ与える。上記したように、２つの書き込みバッファエントリ１５２の同時の付与がデュアルキャッシュポート１６０を経て許された場合には、エントリ１５２の１つは、それが同時に与えられたエントリの対に対応する限り、１つのセットされたＸＤＥＰビットを有する。

【０１０８】

次いで、上記した整数データの場合と同様に、反対区分のエントリ１５２におけるクロス依存性制御ビットＸＤＥＰがクリアされ（プロセス２１２）、制御ビットＡＶ及び制御ビットＦＰＤＶがクリアされ（プロセス２１４）、そしてリタイアポインタ１５８が増加される（プロセス２１６）。

【０１０９】

３．４　非キャッシュ式読み取りの順序付け

上記した書き込みバッファエントリ１５２の割り当てに使用したクロス依存性概念は、マ

イクロプロセッサ１０の他の機能にも使用できる。以下に述べる非キャッシュ式書き込みの場合と同様に、マイクロプロセッサ１０は、メモリからの非キャッシュ式読み取りを必要とする命令をそのプログラムシーケンスに有している。定義によれば、非キャッシュ式読み取りは、定義により単一化キャッシュ６０からのものではないメインメモリ８６からの読み取りであり、非キャッシュ式読み取りは、説明上、メインメモリ８６へ読み取りアクセスを要求するための保持ラッチとして働く単一エントリ読み取りバッファと考える。適切なパイプライン動作を確保するためには、非キャッシュ式読み取りはプログラム順序で実行されねばならない。従って、特に、ここに述べるスーパーパイプライン式スーパースカラーアーキテクチャのマイクロプロセッサにおいては、非キャッシュ式読み取りのプログラム順序を維持する方法が必要となる。

【０１１０】

図１９を参照し、本発明の好ましい実施例による非キャッシュ式読み取りのクロス依存性フィールド３１０について説明する。非キャッシュ式読み取りのクロス依存性フィールド３１０は、好ましくは、単一化キャッシュ６０のキャッシュ制御ロジックの維持され、そして割り当てられた制御ビットＮＣＲＶを含んでいる。これは、セットされると、非キャッシュ式読み取りが割り当てられたことを指示する。上記したクロス依存性制御ビットＸＤＥＰと同様に、そして上記したように、各書き込みバッファエントリ１５２の制御ビットＮＣＲＡは、その割り当て時に、割り当てられた制御ビットＮＣＲＶがセットされた場合にセットされて、非キャッシュ式読み取りが既に割り当てられたことを指示する。制御ビットＮＣＲＡは、各書き込みエントリ１５２のリタイア中にテストされ、メインメモリ８６への要求の適切な順序を確保する。

【０１１１】

更に、非キャッシュ式読み取りのクロス依存性フィールド３１０は、各書き込みバッファエントリ１５２の制御ビットＡＶの各々にマップされた１ビット位置を含み、これは、非キャッシュ式読み取りの割り当て時にどの書き込みバッファエントリ１５２が既に割り当てられているかを指示すると共に、これらの既に割り当てられた書き込みバッファエントリ１５２のリタイアを指示する。非キャッシュ式読み取りのクロス依存性フィールド３１０は、クロス依存性制御ビットＸＤＥＰと同様に動作し、ビットは、非キャッシュ式読み取りの割り当て時にのみセットされそして各書き込みバッファエントリのリタイア時にクリアされる。

【０１１２】

図２０及び２１を参照し、本発明の好ましい実施例による非キャッシュ式読み取り動作の割り当て及びリタイアのプロセスについて詳細に説明する。図２０に示された非キャッシュ式読み取りの割り当てにおいて、プロセス３１２は、先ず命令が非キャッシュ式読み取りを含むことを決定する。次いで、プロセス３１４が実行され、制御ビットＡＶのスナップショットが非キャッシュ式読み取りクロス依存性フィールド３１０にロードされる。次いで、プロセス３１６が実行されて、非キャッシュ式読み取りのクロス依存性フィールド３１０の割り当てられた制御ビットＮＣＲＶがセットされ、その後の割り当てられた書き込みバッファエントリ１５２に、非キャッシュ式読み取り動作が既に割り当てられたことを指示する。次いで、アドレス計算段ＡＣ２に続く（プロセス３１８）。

【０１１３】

図２１は、単一化キャッシュ６０の制御ロジックの制御のもとで非キャッシュ式読み取りを実行するプロセスを示している。判断ステップ３１９は、非キャッシュ式読み取りのクロス依存性フィールド３１０が完全にクリアであるかどうかを判断する。非キャッシュ式読み取りのクロス依存性フィールド３１０のいずれかのビットがセットされた場合には、非キャッシュ式読み取りに既に割り当てられている書き込みバッファエントリ１５２の１つ以上がまだリタイアされておらず、次いで、待機状態３２１に入り、そして既に割り当てられた全ての書き込みバッファエントリがリタイアされるまで判断ステップ３１９が繰り返される。

【０１１４】

非キャッシュ式読み取りのクロス依存性フィールド３１０が完全にクリアされた際には、非キャッシュ式読み取りがプログラム順序において次に実行されるべきものとなる。次いで、プロセス３２０が実行されて、メインメモリ８６からの読み取りが従来の仕方で行われる。読み取りの完了時に、非キャッシュ式読み取りのクロス依存性フィールド３１０の割り当てられた制御ビットＮＣＲＶがプロセス３２２においてクリアされ、従って、書き込みバッファエントリ１５２のその後の割り当ては、それらの制御ビットＮＣＲＡをセットしない。次いで、プロセス３２４は、書き込みバッファエントリ１５２の制御ビットＮＣＲＡをクリアし、非キャッシュ式読み取りの完了を指示すると共に、その後の書き込みバッファエントリ１５２をプログラム順序でリタイアできるようにする。

【０１１５】

１組とみなされる書き込みバッファエントリ１５２の制御ビットＮＣＲＡが非キャッシュ式読み取りのクロス依存性フィールド３１０に対応していることを考えると、非キャッシュ式読み取りのプログラム順序の実行を制御するのに１組のこれらインジケータを使用すれば充分であることが意図される。例えば、非キャッシュ式読み取りのクロス依存性フィールド３１０のみが使用される場合には、書き込みバッファエントリ１５２の割り当て及びリタイアは、フィールド３１０をテストすることにより非キャッシュ式読み取りが割り当てられたかどうかを決定するように制御され、そしてフィールド３１０の対応ビット位置をテストすることにより非キャッシュ式読み取りの前又は後に特定の書き込みバッファエントリ１５２が割り当てられたかどうかを決定するように制御される。

【０１１６】

それ故、本発明の好ましい実施例によれば、非キャッシュ式読み取り動作は、書き込みバッファエントリ１５２のリタイアに対してプログラム順序で実行されるように制御することができる。

【０１１７】

４．リード・アフタ・ライトハザードの検出及び書き込みバッファ動作

上記したように、パイプライン式アーキテクチャのマイクロプロセッサ、特にマイクロプロセッサ１０のようなスーパーパイプライン式スーパースカラーマイクロプロセッサにおいては幾つかのハザードが本来存在する。このようなハザードの重要な種類は、所与の時間にパイプラインにおいて同じレジスタ又はメモリ位置に対して多数の動作が存在するときに生じるデータ依存性である。

【０１１８】

第１の形式のデータ依存性は、同じメモリ位置への書き込み及び読み取りがパイプラインに存在して読み取り動作が書き込みよりも新しい命令であるようなＲＡＷ即ちリード・アフタ・ライトのデータ依存性である。このような場合は、プログラマは、読み取りを実行する前に書き込みが完了すると仮定する。しかしながら、パイプライン動作であるために、特に読み取り動作が加算又は乗算のような別の命令に含まれる場合には、読み取り動作のためのメモリアクセスが書き込みの実行前に行われることがある。このような場合には、メモリ位置への書き込みがまだ行われていないので、読み取りが誤ったデータをコアに返送する。このハザードは、スーパースカラー、スーパーパイプライン式アーキテクチャーのマイクロプロセッサ１０において生じ易く、そして上記したように、命令をプログラム順序から外れて実行できる場合には更に生じ易いものである。

【０１１９】

図１２を参照し、本発明の好ましい実施例によりマイクロプロセッサ１０においてＲＡＷハザードを検出しそして処理するシーケンスを詳細に説明する。この例において、ＲＡＷハザードの検出は、各読み取り命令に対しＸ及びＹパイプラインの第２アドレス計算段ＡＣ２で実行された物理アドレス計算プロセス２１８の結果として生じる。判断ステップ２１９において、書き込みバッファ制御ロジック１５０は、プロセス２１８で計算された読み取り物理アドレスを、パイプラインの関連性に係わりなく、全ての書き込みバッファエントリ１５２における物理アドレス値の各々と比較する。この比較は、読み取りアクセスの物理アドレスを既に割り当てられたアドレスと比較するだけでなく、図７及び８のプロ

セス１７８について上記したように、動作のスパンも考慮する。又、この比較は、反対のＸ又はＹパイプラインの第２のアドレス計算段に現在ある命令に対しても行われる。読み取り動作と、既に割り当てられているか又は同時に割り当てられるがプログラム順序において早期であるいずれかの書き込みとのオーバーラップがない場合には、その特定の読み取り動作についてＲＡＷハザードは存在せず、プロセス２２２において実行が続けられる。判断ステップ２１９において、読み取り動作に対して計算された物理アドレスと、古い命令に割り当てられ且つそのアドレス有効制御ビットＡＶがセットされているか或いは古い命令の同時に割り当てられた書き込みに対して割り当てられている１つ以上の書き込みバッファエントリ１５２$_W$に対する物理アドレスとが一致すると決定された場合には、ＲＡＷハザードが存在することがあり、図１２に示すハザード処理シーケンスへと続けられる。

【０１２０】

上記のように、各書き込みバッファエントリ１５２に対する制御ビットの１つは、ライト・アフタ・リード制御ビットＷＡＲである。この制御ビットは、書き込みバッファエントリ１５２が割り当てられた書き込み動作は、割り当て時に反対のパイプラインの第２アドレス計算段ＡＣ２にある古い（プログラム順序で）読み取り命令の後に行うべき書き込み動作であるという点で、ライト・アフタ・リードであることを指示する。もしこの場合には、制御ビットＷＡＲは、割り当てシーケンスにおいてセットされる（図７のプロセス１８２）。これは、新しい書き込み動作が古い読み取り動作の前に実行する場合にマイクロプロセッサ１０がロックするのを防止する。というのは、古い読み取り動作は、実行時に、それ自身、書き込みがクリアされるまで待機するリード・アフタ・ライト動作と考えるからである。更に、書き込み動作は、読み取りよりも新しく、読み取りがクリアされるのを待機するので、読み取りも書き込みも実行されない。制御ビットＷＡＲを使用することにより、マイクロプロセッサ１０は、明白なＲＡＷハザードが実際にはＷＡＲ状態であるかどうかを判断することができ、この場合は書き込みを処理できる。

【０１２１】

従って、図１２へ戻ると、判断ステップ２２１は、判断ステップ２１９で決定されたように読み取りと一致する物理アドレスを有する各書き込みバッファエントリ１５２$_W$に対して制御ビットＷＡＲがセットされたかどうかを判断する。ＷＡＲビットがセットされた各エントリ１５２$_W$については、ＲＡＷの競合が存在しない。従って、一致するエントリ１５２$_W$がいずれもクリアなＷＡＲビットを有していない場合には、プロセス２２２において読み取りの実行が継続される。しかしながら、書き込み制御ビットＷＡＲがセットされていない各一致する書き込みバッファエントリ１５２$_W$については、ＲＡＷハザードが存在し、図１２のハザード処理シーケンスがそのエントリ１５２$_W$に対して実行される。当然、判断ステップ２２１では、書き込みバッファノーオペレーション制御ビット（ＷＢＮＯＰ）のクリア状態や、本発明の特定の具現化において実施される他の制御ビット及び機能の状態のような他の適当な状態もチェックされる。

【０１２２】

次いで、判断ステップ２２３が実行され、制御ビットＡＶ即ちアドレス有効性が各ＲＡＷエントリ１５２$_W$についてテストされる。判断ステップ２２３は、主として、読み取り動作について待機状態を生じるＲＡＷエントリ１５２$_W$（以下に述べる）がリタイアされたかどうかを決定するように行われる。残りのＲＡＷエントリ１５２$_W$で制御ビットＡＶがセットされているものがない場合には、ＲＡＷハザードがクリアされており、読み取り動作を続けることができる（プロセス２２２）。

【０１２３】

残りの一致するＲＡＷエントリ１５２$_W$の各々に対し、プロセス２２４が次に実行されて、エントリをバイパスできるかどうか、又はハザードを生じた書き込みを読み取り動作を続ける前に完了しなければならないかどうかを判断する。本発明の好ましい実施例によれば、コア２０における読み取り動作へデータを供給する前に書き込みからのデータを単一化キャッシュ６０及びある場合には書き込みバッファ２９に書き込む必要がないようにす

る技術が使用できる。

【０１２４】

しかしながら、このようなバイパス動作は、全ての書き込みに使用できるのではない。この例においては、非キャッシュ式書き込み（非キャッシュ式制御ビットＮＣがエントリ１５２においてセットされることにより指示される）の結果をメインメモリ８６から供給しなければならない。第２に、上記したように、ＲＡＷハザードの特殊な場合が、同じ物理位置に対するリード・アフタ・マルチプルライト（多数の書き込み後の読み取り）である。図７に示すように、割り当てシーケンスのプロセス１７８は、書き込みバッファエントリ１５２の制御ビットＲＤをセットし、そして同じ物理アドレスに既に割り当てられている全ての書き込みバッファエントリの制御ビットＲＤをクリアする。これに対し、読み取り可能でない（即ち、それらの制御ビットＲＤがクリアされた）書き込みバッファエントリ１５２は、それらのデータにエラーがあるので、コア２０へデータを供給するのに使用できない。第３に、後続の読み取りが書き込み動作で書き込まれなかったバイトを包含する場合は書き込み動作からデータを供給することができず、読み取りを完了するためにキャッシュ６０又はメインメモリ８６へのアクセスが依然として必要とされる。

【０１２５】

図１２のＲＡＷ処理シーケンスにおいては、各々の一致する書き込みバッファエントリ１５２$_W$ に対してプロセス２２４が実行され、エントリ１５２$_W$ に対する制御ビットＲＤがセットされた（エントリ１５２$_W$ が読み取りの物理アドレスに割り当てられた最後のエントリ１５２であることを指示する）かどうかを判断し、制御ビットＮＣがクリアである（書き込みが非キャッシュ式でないことを指示する）かどうかを判断し、そして読み取られるべきバイトがメモリに書き込まれるべきバイトのサブセットであるという点で、読み取りの物理アドレスが書き込みバッファエントリ１５２$_W$ への書き込みの物理アドレスに「厳密」に一致するかどうかを判断する。これら３つの全ての条件を満たすエントリ１５２$_W$ は、「バイパス可能」であると言え、そして制御は以下に述べる判断ステップ２２５へ進む。上記条件の１つ以上（非キャッシュ式、非読み取り可能又は不存在物理アドレス）を満足しないことにより、バイパス可能なエントリ１５２$_W$ が存在しない場合には、待機状態２２９が行われ、そして制御は判断ステップ２２３へ戻り、この状態は、制御ビットＡＶがクリアされることにより示されるように全ての非バイパスエントリ１５２$_W$ がリタイアされるまで保持され、その後に、読み取り動作が続けられる（プロセス２２２）。

【０１２６】

本発明のこの実施例においては、各バイパス可能なエントリ１５２$_W$ に適用できるバイパス方法が判断ステップ２２５において決定され、制御ビットＤＶ即ちデータ有効性がテストされて、書き込みバッファエントリ１５２$_W$ が保留中であり（即ち、有効なデータを含み）まだリタイアされていないかどうか判断する。保留中である各バイパス可能なエントリ１５２$_W$ については、プロセス２３０が書き込みバッファ制御ロジック１５０によって実行されて、書き込みバッファエントリ１５２$_W$ のデータ部分の内容を、最初にメモリへ書き込まずに、コア２０へ直接供給できるようにする。図５を参照すると、プロセス２３０が行われて、書き込みバッファ制御ロジック１５０が書き込みバッファエントリ１５２$_W$ をイネーブルし、読み取り動作の際に、そのデータをソースバスＳＲＣ（即ち、エントリ１５２$_W$ を含む書き込みバッファ２９の区分に対するバスＳＲＣｘ、ＳＲＣｙの一方）に出すと共に、適当なマルチプレクサ１５４を制御して、ソースバスＳＲＣを、データを要求しているコア２０のＸ又はＹの一方のパイプラインに適用する。従って、この場合に、ＲＡＷハザードの検出は、書き込みバッファ２９からコア２０へデータを供給することにより処理され、読み取り動作の実行時間がスピードアップされる。

【０１２７】

しかしながら、制御ビットＤＶがセットされていないことが判断ステップ２２５で分かることにより示されるようにまだ保留になっていないバイパス可能な書き込みバッファエントリ１５２$_W$ については、有効なデータがエントリ１５２$_W$ に存在せず、そこからコア２０へ供給することができない。これらエントリ１５２$_W$ に対してプロセス２３２が実行さ

れ、コア２０により書き込みバッファエントリ１５２$_W$への書き込みが行われるときに、ライトバックバスＷＢ＿ｘ又はＷＢ＿ｙ上の有効データ（対応するバイパスバスＢＰ＿ｘ、ＢＰ＿ｙにも存在し、マルチプレクサ１５４ｘ、１５４ｙの適当な一方に送られる）が、コア２０の要求を発しているＸ又はＹパイプラインへ送られる。このように、ＲＡＷハザードは、有効なデータをもつ書き込みバッファ２９をバイパスすることによって取り扱われ、データをコア２０へ供給する前に、キャッシュ６０、メインメモリ８６又は書き込みバッファエントリ１５２$_W$からの有効データの記憶及び検索が必要とされないので、読み取り動作の実行が更にスピードアップする。

【０１２８】

５．推論的実行及び例外処理

５．１　推論的実行

上記したように、本発明の好ましい実施例によるスーパーパイプライン式スーパースカラーマイクロプロセッサ１０は、推論的に命令を実行できる。この推論は、条件分岐又はジャンプステートメントの後であって且つジャンプ又は分岐の基礎となる条件の状態を決定する前に１つ以上の命令を実行することにより生じる。推論的実行を行わない場合には、マイクロプロセッサは、後続命令の実行の前に、条件の状態を決定する命令の実行を待機しなければならず、パイプライン「ストール（停止）」状態を招く。推論的実行では、マイクロプロセッサ１０は条件の状態を推論し、その推論に基づいて命令を実行する。行われる推論的実行の数及び推論が正確である率にもよるが、パイプラインストールの影響は著しく減少される。

【０１２９】

本発明のこの実施例によるマイクロプロセッサ１０は、首尾よくいかなかった推論の影響を速やかにクリアして、特に、推論的書き込みの結果がメモリへリタイアされないよう確保すると共に、推論的に書き込まれたデータを書き込みバッファ２９から除去するための回路を備えている。図１３及び１４を参照し、推論的書き込みを実行すると共に不首尾な推論を取り扱う方法について詳細に説明する。図１３及び１４は、この方法を一般的ではなく一例として示すものであり、当業者であれば、この例についての以下の説明を参照することにより、図１３及び１４の方法をマイクロプロセッサにおいて容易に実現することができよう。

【０１３０】

図１３の例示的シーケンスはプロセス２４０で始まり、コア２０は、条件がまだ分からない条件分岐の１つの結果に命令が対応するという点で推論的に実行されるべき一連の命令を選択する。どの条件分岐を選択すべきか（即ち、条件分岐又はジャンプを行うべきかどうか）の判断は、従来の予想分岐機構に基づいて行われる。プロセス２４２において、２つの書き込みバッファエントリ１５２ａ、１５２ｂの割り当て（推論的分岐は、この例ではメモリへの２つの書き込み動作を含む）が、上記のように、パイプラインの第２のアドレス計算段ＡＣ２において実行される。しかしながら、書き込みバッファエントリ１５２ａ、１５２ｂへの書き込み動作は推論的であるから、プロセス２４２の割り当て中に書き込みの推論の程度に基づいて推論制御ビット（ＳＰＥＣ）の少なくとも１つがセットされる。

【０１３１】

本発明のこの実施例では、４次の推論的実行が許される。推論の次数即ち程度は、各書き込みバッファエントリ１５２ごとに、４つｊ、ｋ、ｌ、ｍのＳＰＥＣ制御ビット（ＳＰＥＣビット）によって指示され、各ビット位置は、書き込みバッファエントリ１５２が選択された条件分岐の１つに対し推論的書き込みであるかどうかに対応する。図１３は、プロセス２４２の割り当て後の４つの書き込みバッファエントリ１５２ａ、１５２ｂ、１５２ｃ及び１５２ｄの条件を示している。図１３に示すように、プロセス２４２で割り当てられた書き込みバッファエントリ１５２ａ、１５２ｂは、それらのｊＳＰＥＣビットがセットされている。プロセス２４２の割り当ては、第１次の推論である（即ち、この例で行われる第１の推論である）から、エントリ１５２ａ、１５２ｂに対して単一のｊＳＰＥＣ制

御ビットのみがセットされる。書き込みバッファエントリ１５２ｃ、１５２ｄは、まだ割り当てられておらず、従って、それらの推論制御ビットはクリアである。

【０１３２】

プロセス２４２の割り当ての後に、選択された条件分岐において推論的命令の実行がプロセス２４４で開始される。これら命令の実行は、それが完了した場合に、割り当てられた書き込みバッファエントリ１５２ａ、１５２ｂへの書き込みに影響を与え、それらの制御ビットＤＶがセット状態になる。しかしながら、これら書き込みの実行は推論的であるから、図１１について述べたリタイアシーケンスは、（推論的実行が組み込まれる場合に）制御ビットＳＰＥＣが全てクリアされない限り書き込みバッファエントリ１５２のリタイアを防止するゲート判断も含んでいなければならない。これは、推論的実行の結果がメモリに達するのを防止する。というのは、推論的予想が誤りであった（即ちプロセス２４０で選択されたものから別の分岐をとっていなければならなかった）場合には回復するのが非常に困難であり、もしできたとしても、非常に時間がかかるからである。

【０１３３】

図１３の例では、２次の推論も行われ、プロセス２４０で選択された分岐における命令の１つが、別の条件分岐又はジャンプを含み、これに対して予想分岐選択が再びプロセス２４６で行われて、パイプラインがストールしないようにされる。２次の推論は、プロセス２４６で選択された分岐に対する命令の実行を首尾良く行うために、プロセス２４６での選択を正しく行うだけでなく、プロセス２４０での選択も正しくなければならないことを意味する。プロセス２４６は、図１３では、プロセス２４４における命令の実行後に生じるものとして示されているが、上記したマイクロプロセッサ１０のスーパーパイプラインアーキテクチャにより、プロセス２４６の予想分岐は、プロセス２４４で開始される実行の完了よりも前に生じることがしばしばある。プロセス２４６での分岐の選択の後に、プロセス２４８で書き込みバッファエントリ152_Cが割り当てられる（これも第２のアドレス計算パイプライン段の間に）。プロセス２４６のこの割り当てにおいては、書き込みバッファエントリ152_Cへのいかなる書き込みも２次の推論であるから、ｊ及びｋの両ＳＰＥＣ制御ビットがセットされる。プロセス２４６の後の書き込みバッファエントリ１５２ａ、１５２ｂ、１５２ｃ、１５２ｄに対する制御ビットＳＰＥＣの状態が図１３に示されている。プロセス２４６で選択された分岐における推論的命令の実行は、プロセス２５０で開始される。

【０１３４】

図１３の例では、３次の推論も行われ、これは、プロセス２４６で選択された分岐における命令のシーケンスも別の条件分岐又はジャンプを含むことを意味する。プロセス２５２は、予想分岐選択に基づいて分岐の１つを選択するが、この３次の選択を首尾良く行うためには、プロセス２４０、２４６及び２５２の３つの選択が全て首尾良く行われねばならない。この場合も、前記のように、プロセス２５２は、マイクロプロセッサ１０がスーパーパイプライン式アーキテクチャであることを考えれば、プロセス２５０での命令の実行が完了する前に分岐の選択を行うことがある。この例では、プロセス２５４において書き込みバッファエントリ１５２ｄが割り当てられ、この書き込みバッファエントリ１５２ｄにおいて３つのｊ、ｋ及びｌのＳＰＥＣビットがセットされる。プロセス２５４の後の書き込みバッファエントリ１５２ａないし１５２ｄに対する制御ビットＳＰＥＣの状態がプロセス２５４に示されている。次いで、プロセス２５６は、プロセス２５２で選択された分岐の命令を実行し、これは、書き込みバッファエントリ１５２ｄへの書き込み動作を含む。

【０１３５】

図１４を参照し、書き込みバッファ２９による成功及び不成功の推論的実行の例を説明する。図１３の例と同様に、図１４のシーケンスは、一般的な場合ではなくて、一例に過ぎず、当業者であれば、マイクロプロセッサアーキテクチャにおいてこの方法を容易に実現できることが意図される。

【０１３６】

プロセス２６０において、コア２０は、プロセス２４０の第１選択が首尾良く行われ、プロセス２４４で実行される命令へ分岐（又は非分岐）させるに必要な条件がその前の命令で満足されたことを検出する。従って、プロセス２４２で割り当てられそしてプロセス２４４で書き込まれた書き込みバッファエントリ１５２ａ、１５２ｂのデータ部分の内容がメモリへリタイアされる。というのは、それらの内容は、実行されたプログラムの正確な結果だからである。それ故、プロセス２６２において、全ての推論的書き込みバッファエントリ１５２ａ、１５２ｂ、１５２ｃ、１５２ｄのｊＳＰＥＣビットがクリアされ、プロセス２６２の後の書き込みバッファエントリ１５２ａないし１５２ｄに対する制御ビットＳＰＥＣの状態が図１４に示されている。書き込みバッファエントリ１５２ａ、１５２ｂは、ここで、それらの全ての推論的制御ビットＳＰＥＣがクリアされているので（そしてそのデータ有効性制御ビットＤＶが既にセットされているので）、書き込みバッファエントリ１５２ａ、１５２ｂは、場合によって単一化キャッシュ６０又はメインメモリ８６へリタイアされる。

【０１３７】

図１４の例では、第２の分岐選択（プロセス２４６でなされた）が不首尾であると検出される。というのは、プロセス２４８で実行された命令に必要な条件がその前の命令によって満足されていないからである。更に、プロセス２５２で行われる分岐の選択も、プロセス２４６の首尾良い選択に基づいているので、プロセス２５６で実行されるべき命令に必要な条件も満足されない。書き込みバッファエントリ１５２ｃ、１５２ｄへの書き込みがまだ行われていない程度までは、上記の不首尾の予想選択のために、これらの書き込みは決して実行されない。これらの書き込みが生じた（即ち、書き込みバッファエントリ１５２ｃ、１５２ｄは保留中）程度までは、データは、それがエラーであるからメモリに書き込まれてはならない。従って、書き込みバッファエントリ１５２ｃ、１５２ｄは、それらの内容をリタイアせずに、更に使用するためにクリアされねばならない。

【０１３８】

図１４のシーケンスは、プロセス２６６で始まって不首尾な推論的実行を取り扱い、ＳＰＥＣビットｋがセットされた書き込みバッファエントリ１５２が書き込みバッファ制御ロジック１５０により識別される。この例では、これらの識別された書き込みバッファエントリ１５２は、エントリ１５２ｃ（２次推論）及び１５２ｄ（３次推論）である。プロセス２６８において、書き込みバッファ制御ロジック１５０は、各エントリ１５２ｂ、１５２ｃに対するアドレス有効制御ビットＡＶをクリアし、エントリ１５２ｃ、１５２ｄは再割り当てされリタイアされない（リタイアを行うためにＡＶビットをセットしなければならない図１１のリタイアシーケンスを参照されたい）。

【０１３９】

上記したように、リタイアポインタ１５８ｘ、１５８ｙは、次にリタイアされるべき書き込みバッファエントリ１５２の１つを指す。本発明の好ましい実施例によれば、制御ビットＷＢＮＯＰが書き込みバッファエントリ１５２ｃ、１５２ｄに対してセットされ、それに関連するリタイアポインタ１５８がエントリ１５２ｃ、１５２ｄを指すときに、これらのエントリがスキップされる（あたかもそれらが割り当てられなかったかのように）。これは、書き込みバッファ２９の該当区分が空である場合に、リタイアポインタ１５８が割り当てポインタ１５６を「捕らえる」ことを許す。リタイアプロセスにおいてアドレス有効制御ビットＡＶを繰り返しチェックすることにより、空条件に合致したときに安全停止することができる。

【０１４０】

適切な条件分岐の実行は、図１４に示すプロセス２７０で再開することができる。

【０１４１】

５．２　例外処理

推論的実行に加えて、命令の実行により一般に例外と称されているエラー状態が返送される場合にはパイプラインのストール及びバブルが生じることがある。例外の一例は、コア２０が０で除算する状態を検出する場合である。パイプラインの実行段においてこのよう

な例外が検出されたときは、例外状態を従来の仕方で適切に処理するために、まだパイプラインにある命令をクリアしなければならない。特に、書き込みバッファ２９については、例外を生じた命令の後に割り当てられた書き込みバッファエントリ１５２をフラッシュしなければならない。パイプラインからの書き込み命令が除去されるためにこれらのエントリ１５２への書き込みは決して行われない（そしてデータ有効制御ビットＤＶは決してセットされない）ので、エントリ１５２は、フラッシュされなくても書き込みバッファ２９から決してリタイアせず、マイクロプロセッサ１０は不定にハングし、決して到着することのないデータを待機する。

【０１４２】

図１５を参照し、書き込みバッファ２９に対する例外を取り扱うシーケンスの一例を詳細に説明する。プロセス２７２において、コア２０は、例外状態を検出する。プロセス２７４は、書き込みバッファ制御ロジック１５０によって実行され、制御ビットＡＶ及び制御ビットＤＶが書き込みバッファ２９の各書き込みバッファエントリ１５２から検索される。次いで、判断２７３が行われて、制御ビットＡＶのいずれかが書き込みバッファ２９においてセットされたかどうか判断する。制御ビットＡＶがセットされた各書き込みバッファエントリ１５２に対して、判断２７５でその制御ビットＤＶ即ちデータ有効性がテストされ、それがセットされたかどうか判断される。もしそうでなければ（そのエントリ１５２への書き込みが例外の時点でまだ生じていないことを意味する）、制御ビットＡＶがクリアされ、そして書き込みバッファノーオペレーションビットＷＢＮＯＰがそのエントリ１５２に対してセットされる。上記のように、ＷＢＮＯＰビットは、リタイアポインタ１５８がこのエントリ１５２をスキップできることを指示し、割り当てポインタ１５６ｘ、１５６ｙがそれらの各々のリタイアポインタ１５８ｘ、１５８ｙに等しいところの空条件を達成することができる。次いで、制御は以下に述べるようにプロセス２７４へ戻される。

【０１４３】

制御ビットＡＶ及び制御ビットＤＶの両方がセットされた（判断２７３、２７５により決定される）保留中の書き込みバッファエントリについては、データがコア２０により例外状態の前に書き込まれている。従って、これらの位置に書き込まれたデータは有効であり、図１１について述べた通常の非同期リタイアシーケンスでメモリに書き込むことができる。しかしながら、マイクロプロセッサ１０により例外を処理する前に、書き込みバッファ２９の全てのエントリをリタイアし、割り当てに使用できるようにしなければならない（即ち、書き込みバッファ２９を空にしなければならない）。従って、シーケンスの制御はプロセス２７４へ戻り、全ての書き込みバッファエントリ１５２に対する制御ビットＡＶがクリアされるときまで、制御ビットＡＶ及び制御ビットＤＶが再び検索され質問される。全ての制御ビットＡＶがクリアされたときは、ＷＢＮＯＰビットの作用を考慮して、割り当てポインタ１５６ｘ、１５６ｙの両方がそれらの各々のリタイアポインタ１５８ｘ、１５８ｙと同じエントリ１５２を指す。この空条件が達成されると、プロセス２７８を開始することができ、例外状態が通常のやり方で処理される。

【０１４４】

６．書き込みバッファからの特殊な書き込みサイクル

図１１について上記したように、リタイアプロセスは、書き込みバッファ２９からキャッシュポート１６０への又は直接的にデータバスＤＡＴＡへ至る特殊な書き込み動作を含む。本発明の好ましい実施例によれば、これら特殊な書き込みサイクルは、不整列書き込み及び集合書き込みの取り扱いを含むことができる。本発明の好ましい実施例によりこれらの特殊な書き込みサイクルを取り扱うシーケンスを以下に詳細に説明する。

【０１４５】

６．１　不整列書き込み

上記したように、マイクロプロセッサ１０内に与えられた物理メモリアドレスは、メモリのバイトアドレスに対応し、一方、データバスＤＡＴＡは、６４ビットを並列に通信することができる（本発明のこの実施例では主としてバスインターフェイスユニットＢＩＵの

データ入力／出力から単一化キャッシュ６０へ）。Ｘ８６適合型のマイクロプロセッサにおける物理アドレスはオペランドサイズのモジュロではないので、メモリ書き込みの相当の部分が８バイト境界にオーバーラップし、これらの書き込みを「不整列」書き込みと称する。本発明の好ましい実施例によるマイクロプロセッサ１０の書き込みバッファ２９は、書き込みバッファエントリ１５２が割り当て時に不整列であることを指示し、書き込みの第２部分を与える第２の書き込みバッファエントリ１５２を割り当て、そして不整列書き込みを考慮するようにリタイアプロセスにおいて特殊なルーチンを開始することにより、このような不整列書き込みを考慮するものである。これらのシーケンスを図１６及び１７について詳細に説明する。

【０１４６】

図１６は、図７の割り当てシーケンスのプロセス１８２の一部分であって、不整列書き込みを検出しそしてそれを割り当てられている書き込みバッファエントリ１５２に対して指示するための部分を示すフローチャートである。図１６のプロセス２８０において、書き込みバッファ制御ロジック１５０は、割り当てられている書き込みバッファエントリ１５２$_n$ への書き込み動作の物理アドレス（下位バイトアドレス）に、書き込み動作のサイズ（バイト単位）を加える。書き込み動作のサイズに関する情報は、Ｘ８６型マイクロプロセッサ命令にとって通常そうであるように、命令内に含まれている。判断２８１において、書き込みバッファ制御ロジックは、プロセス２８０の加算がビット３への桁上げを生じたかどうか判断し、これは、割り当てられている書き込みバッファエントリ１５２$_n$ への書き込み動作が８バイト境界に交差することを指示する。判断２８１で桁上げが生じないと決定されると、エントリ１５２$_n$ への書き込みは不整列ではなく、次いで、プロセス２８２が実行され、不整列書き込み制御ビットＭＡＷがエントリ１５２$_n$ においてクリアされ、そして割り当てシーケンスが続けられる（プロセス２８８）。

【０１４７】

しかしながら、桁上げが生じた場合には、エントリ１５２$_n$ への書き込みが８バイトの境界に交差し、この場合には、プロセス２８４が実行されて、エントリ１５２$_n$ の制御ビットＭＡＷをセットする。次いで、プロセス２８６において、割り当てられるべき次の書き込みバッファエントリ１５２$_{n+1}$ が不整列書き込みの目的で割り当てられるが、これは、エントリ１５２$_{n+1}$ のアドレス部分に次の８バイトグループ（即ち、プロセス２８１の検出された桁上げの後の８バイトアドレス）への書き込みに対する物理スタートアドレスをロードし、そしてエントリ１５２$_{n+1}$ に対する制御ビットＡＶをセットすることにより行われる。プロセス２８６では、上位物理アドレスが異なる物理ページに存在するかもしれないことを考慮して、新たな物理アドレスの計算（パイプライン段ＡＣ２）が必要とされる。しかしながら、エントリ１５２$_{n+1}$ は、メモリへの第２のオペランド書き込みを行うためのリタイアプロセスに使用されるものに過ぎないから、エントリ１５２$_{n+1}$ のデータ部分は空のままである。次いで、割り当てプロセスの残りの部分へ続く（プロセス２８８）。

【０１４８】

書き込みバッファエントリ１５２$_n$ が不整列書き込みであるかどうかに係わりなく、エントリ１５２$_n$ へのデータの発生は、図１０について上記したように行われる。本発明のこの実施例によれば、書き込みバッファエントリ１５２$_n$ のデータ部分の特殊なロードは行われないが、不整列書き込みの場合には、エントリ１５２$_{n+1}$ へのデータの発生は行われない。

【０１４９】

図１７を参照し、書き込みバッファエントリ１５２のリタイアにおいて不整列書き込みを処理するシーケンスについて説明する。前記のリタイアシーケンスと同様に、図１７のシーケンスは、書き込みバッファ制御ロジック１５０から助成されるキャッシュ制御ロジックの制御のもとで実行されるのが好ましい。図１７のシーケンスは、上記した図１１のプロセス２０８及び２１０の一部分として行われる。このシーケンスは判断２８９で始まり、エントリ１５２$_n$ の制御ビットＭＡＷがテストされ、もしクリアであれば、リタイアシ

ーケンスは上記のように続けられる（図１７のプロセス２９０）。しかしながら、エントリ１５２$_n$ に対して制御ビットＭＡＷがセットされた場合には、プロセス２９２が次に行われ、エントリ１５２$_n$ のデータ部分が適当な不整列データラッチ１６２ｘ、１６２ｙにラッチされる。

【０１５０】

エントリ１５２$_n$ からのデータの付与は、書き込みの不整列特性を考慮して、２つのメモリサイクルで行わねばならない。しかしながら、書き込み動作を２つのサイクルに分割する場合に、エントリ１５２$_n$ に記憶されたデータは、キャッシュメモリ１６０へ付与するための適切な「バイトレーン」にない。図５に戻ると、シフタ１６４は、対応する書き込みバッファ区分１５２ｘ、１５２ｙから与えられたデータを、その不整列書き込みラッチ１６２ｘ、１６２ｙに記憶する前にシフトするための従来のバレルシフタである。従って、シフタ１６４は、対応する書き込みバッファ区分１５２$_n$ におけるデータの単一シフトを行って、下位データが上位ビットレーンに現れ（第１の下位アドレスの書き込み動作においてキャッシュポート１６０へ与えるために）そして上位データが下位ビットレーンに現れる（第２の上位アドレスの書き込み動作においてキャッシュポート１６０へ与えるために）ようにすることができる。このシフト動作は、図１７に示すシーケンスのプロセス２９２で行われる。

【０１５１】

次いで、プロセス２９４が行われ、エントリ１５２$_n$ の物理アドレスは、下位アドレス８バイトグループに対応するバイトレーンに整列された（プロセス２９２においてシフタ１６４により）下位アドレス８バイトグループに対応するデータの部分と共にキャッシュポート１６０へ与えられる。これは、不整列書き込みに必要な第１の書き込み動作を行う。次いで、プロセス２９６において、不整列書き込みの第２オペランドに対するアドレス及びデータが与えられる。物理アドレスは、次の書き込みバッファエントリ１５２$_{n+1}$ のアドレス部分に記憶されたものであり、そしてデータは、シフタ１６４によりポート１６０への第２アクセスのために適切なバイトレーンにシフトされてエントリ１５２$_n$ から不整列書き込みラッチ１６２に保持されたものである。次いで、リタイアプロセスの残り部分へ続く（プロセス２９８）。

【０１５２】

上記のように、本発明のこの実施例によるマイクロプロセッサ１０の例外処理機能は、制御ビットＤＶの状態を使用して、エントリ１５２が例外検出後にフラッシュされたか否かを判断する。しかしながら、不整列書き込みの場合には、第２の書き込みエントリ１５２$_{n+1}$ は、たとえ書き込みが行われていてもその制御ビットＤＶをセットしていない。というのは、手前の（プログラム順序で）書き込みバッファエントリ１５２$_n$ 内に有効なデータが含まれているからである。従って、不整列書き込み処理機能と、ここに述べる例外処理との両方が与えられた場合には、例外処理シーケンスは、エントリ１５２$_n$ に対して制御ビットＭＡＷ及び制御ビットＤＶの両方をテストし、そして両方がセットされた場合に、次の書き込みエントリ１５２$_{n+1}$ （プログラム順序で）もその制御ビットＤＶをセットしてエントリ１５２$_{n+1}$ がフラッシュされないようにすることを考慮しなければならない。

【０１５３】

この構成の結果、不整列書き込みは、本発明によるマイクロプロセッサ１０により、コア２０の動作に影響を与えないやり方で処理され、しかも、非同期で行われる厳密でないリタイアシーケンス中に付加的なラッチ及び整列ステップを含むだけである。

【０１５４】

６．２　集合書き込み

本発明のこの実施例によるマイクロプロセッサ１０によって実行することのできる別の形式の特殊な書き込み動作は、次々の書き込み動作内に含まれたデータがメモリへの単一書き込みアクセスに集合される集合書き込みである。上記したように、各物理アドレスはバイト位置に対応する。同時にデータバスに出される同じバイトブロック内の１つ又は幾つ

かのバイトに対して一連の書き込みを行うべき場合には、マイクロプロセッサ１０は、データを適切なバイトレーンに保持して、次々の小さな書き込みアクセスではなくてキャッシュポート１６０又はメモリへの単一の書き込みアクセスを行うようにすることができる。例えば、マイクロプロセッサ１０のメモリデータバスＤＡＴＡは６４ビット巾であるから、８バイトのデータを同時に書き込むことができ、本発明の集合書き込み特徴によれば、これらの８バイトは、多数の書き込みバッファエントリ１５２から以下に述べるように集合することができる。

【０１５５】

書き込みバッファ２９の割り当てシーケンスについて上記したように、併合制御ビットＭＲＧは、プログラム順序で直前のメモリ書き込み命令に対し既に割り当てられている別の書き込みバッファエントリ１５２と相接するがオーバーラップしない物理メモリアドレスへの書き込みを実行している各書き込みバッファエントリ１５２に対し割り当て時にセットされる。相接し隣接するという制約は、マイクロプロセッサ１０のＸ８６適合性を考慮して本発明の好ましい実施例により実施されるものであるが、集合書き込みは、同じバイトブロックにおけるデータのメンバーシップが併合書き込みにとって唯一必要な制約であるように他のアーキテクチャでも実施できることが意図される。割り当て後に、併合可能な書き込みバッファエントリ１５２へのデータの発生が、上記した通常のやり方で続けられる。

【０１５６】

図１８を参照し、本発明の好ましい実施例による集合書き込み動作を以下に詳細に説明する。判断ステップ２９９は、リタイアされている現在書き込みバッファエントリ152_nの制御ビットＭＲＧがセットされたかどうかを判断し、もしそうでなければ、通常のリタイアシーケンスが続く（プロセス３００）。現在エントリ152_nの制御ビットＭＲＧがセットされた場合には、プロセス３０２が行われ、エントリ152_nのデータ部分が、適当なシフタ１６４ｘ、１６４ｙにより、集合書き込みを受け入れるための適切なバイトレーンへシフトされる。次いで、プロセス３０４が実行され、シフトされたデータは、手前の相接しているがオーバーラップしない書き込みから集合書き込みラッチ１６５に既にロードされたデータを妨げることなく、適切なバイトレーン位置において集合書き込みラッチ１６５に記憶される。

【０１５７】

次いで、判断３０５は、次の書き込みバッファエントリ152_{n+1}に質問し、その制御ビットＭＲＧがセットされたかどうか判断する。もしそうであれば、制御はプロセス３０２へ戻り、該次のエントリ152_{n+1}に対するデータがシフトされそしてプロセス３０４において集合書き込みラッチ１６５へラッチされる。次のエントリ１５２に対し制御ビットＭＲＧ又は制御ビットＡＶがクリアされることによりそれ以上の併合可能なエントリ１５２が存在しないことが示されると（判断３０５において）、ラッチ１６５の内容が、適当な物理アドレスと共にポート１６０へ与えられ、場合によってキャッシュ６０又はメインメモリ８６への集合書き込み動作が行われる。次いで、リタイアプロセスが前記のように続けられる（プロセス３０８）。

【０１５８】

それ故、本発明の好ましい実施例によれば、相接するメモリ位置への多数のアクセスに代わって単一のメモリアクセスで書き込み動作を行えるようにしたことにより、データをキャッシュ又はメモリへリタイアする効率が相当に改善された。

【０１５９】

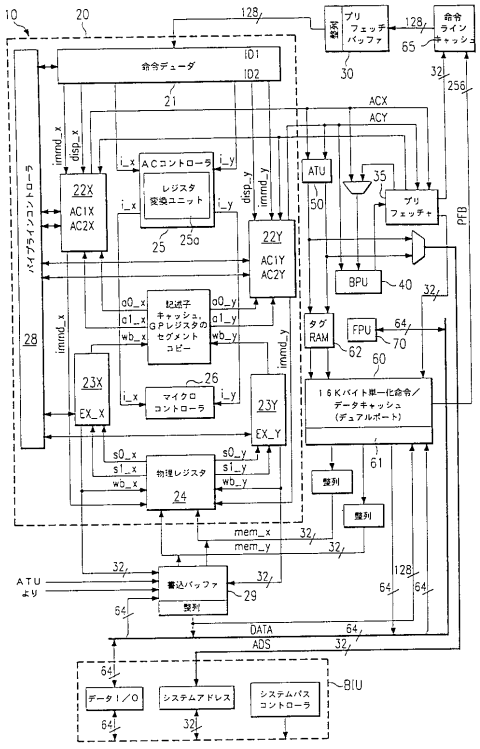
７．結論

本発明の好ましい実施例によれば、ＣＰＵコアとメモリシステム（キャッシュメモリを含む）との間に、実行された命令シーケンスの結果をバッファするための書き込みバッファが設けられた。これは、時間的に厳密でない書き込み動作がバス又はメモリシステムを占有することによりキャッシュ及びメモリの読み取りを最小の待機状態で高い優先順位で行うことができるようにする。

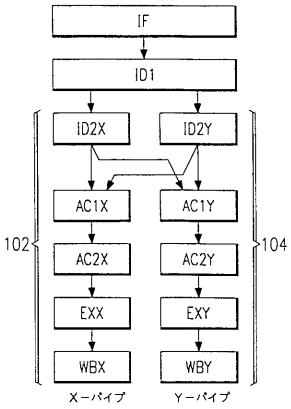
【０１６０】
更に、本発明の好ましい実施例は、特定のマイクロプロセッサアーキテクチャに対して特に有益な多数の特徴を含む。このような特徴は、スーパースカラープロセッサに対して書き込みバッファの２つの区分を設けることと、バッファの分割にも係わりなくプログラム順序でメモリにデータが書き込まれるよう確保する技術とを含む。本発明の好ましい実施例の付加的な特徴は、データ依存性及び例外のようなハザードの検出及び処理と、予想が不首尾であった場合に書き込みバッファを迅速且つ正確にフラッシュして命令の推論的実行を行うことを含む。
【０１６１】
本発明をその好ましい実施例について説明したが、当業者であれば、以上の説明及び添付図面から、本発明の種々の変更や置き換えや修正が明らかとなろう。このような変更や修正は、本発明の範囲内に全て包含されるもので、本発明は、特許請求の範囲のみによって限定されることを理解されたい。
【図面の簡単な説明】
【図１】マイクロプロセッサ全体を示すブロック図である。
【図２】命令パイプライン段の一般化されたブロック図である。
【図３】マイクロプロセッサを用いたプロセッサシステムのブロック図である。
【図４】パイプライン段を通る命令の流れを示すタイミング図である。
【図５】本発明の好ましい実施例による図１のマイクロプロセッサの書き込みバッファのブロック図である。
【図６】図５の書き込みバッファの１つのエントリの内容を示す図である。
【図７】図１のパイプラインのアドレス計算段ＡＣ２の間の書き込みバッファエントリの割り当てを示すフローチャートである。
【図８】図７の割り当てにおける物理アドレス比較プロセスを示す図である。
【図９】図１のマイクロプロセッサの一方のパイプラインに対する書き込みバッファエントリと、図１のマイクロプロセッサの他方のパイプラインに対する書き込みバッファエントリとのクロス依存性フィールドのアドレス有効ビットを示すマップである。
【図１０】本発明の好ましい実施例による書き込みバッファエントリの発生を示すフローチャートである。
【図１１】本発明の好ましい実施例による書き込みバッファエントリのリタイアを示すフローチャートである。
【図１２】本発明の好ましい実施例による依存性ハザードの検出及び処理方法を示すフローチャートである。
【図１３】本発明の好ましい実施例により推論的実行を処理する方法を示すフローチャートである。
【図１４】本発明の好ましい実施例により推論欠陥を処理する方法を示すフローチャートである。
【図１５】本発明の好ましい実施例による例外処理方法を示すフローチャートである。
【図１６】本発明の好ましい実施例により不整列書き込み動作に対する書き込みバッファ位置を割り当てる方法を示すフローチャートである。
【図１７】本発明の好ましい実施例により不整列書き込み動作に対する書き込みバッファ位置をリタイアするシーケンスを示すフローチャートである。
【図１８】本発明の好ましい実施例により集合書き込み動作に対する書き込みバッファ位置をリタイアするシーケンスを示すフローチャートである。
【図１９】本発明の好ましい実施例により図１のマイクロプロセッサに使用される非キャッシュ式読み取りのクロス依存性フィールドを示す図である。
【図２０】本発明の好ましい実施例による非キャッシュ式読み取り動作の割り当てシーケンスを示すフローチャートである。
【図２１】本発明の好ましい実施例による非キャッシュ式読み取り動作のリタイアシーケンスを示すフローチャートである。

【符号の説明】
１０　マイクロプロセッサ
２０　中央処理ユニット（ＣＰＵ）コア
２１　命令デコーダ
２４　レジスタファイル
２５　ＡＣ制御ユニット
２６　マイクロコントロールユニット
２８　パイプコントロールユニット
２９　書き込みバッファ
３０　プリフェッチバッファ
３５　プリフェッチャ
４０　分岐処理ユニット（ＢＰＵ）
５０　アドレス変換ユニット（ＡＴＵ）
６０　単一化キャッシュ
６２　ＲＡＭ
６５　命令ラインキャッシュ
７０　浮動小数点ユニット（ＦＰＵ）

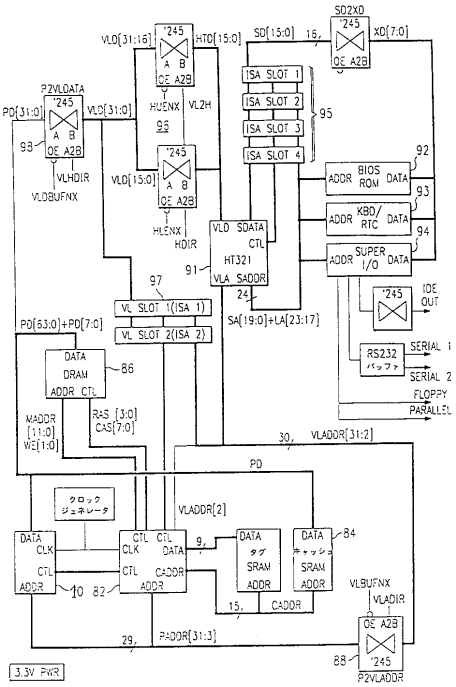
【図１】



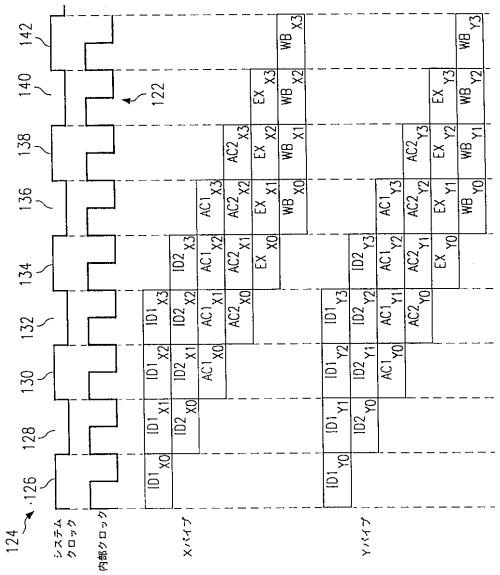
【図２】



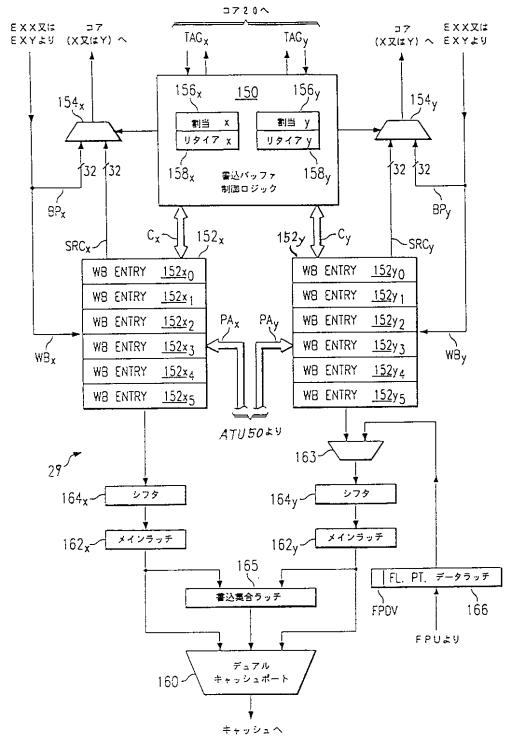
【図３】


SD2XD
'245
OE A2B
VLD[31:16]
HTD[15:0]
SD[15:0]
XD[7:0]
P2VLDATA
PD[31:0]
VLD[31:0]
HUENX
VL2H
ISA SLOT 1
ISA SLOT 2
ISA SLOT 3
ISA SLOT 4
VLHDIR
VLDBUFNX
VLD[15:0]
HLENX
HDIR
BIOS ROM
KBD/ RTC
SUPER I/O
ADDR
DATA
VLD SDATA
CTL
HT321
VLA SADDR
IDE OUT
VL SLOT 1(ISA 1)
VL SLOT 2(ISA 2)
SA[19:0]+LA[23:17]
PD[63:0]+PD[7:0]
RS232 バッファ
SERIAL 1
SERIAL 2
FLOPPY
PARALLEL
DATA DRAM ADDR CTL
MADDR [11:0] WE[1:0]
RAS [3:0] CAS[7:0]
VLADDR[31:2]
PD
クロック ジェネレータ
VLADDR[2]
DATA CLK CTL ADDR
CTL CTL CLK DATA CTL CADDR ADDR
タグ SRAM ADDR
キャッシュ SRAM ADDR
VLBUFNX
VLADIR
CADDR
PADDR[31:3]
P2VLADDR
3.3V PWR

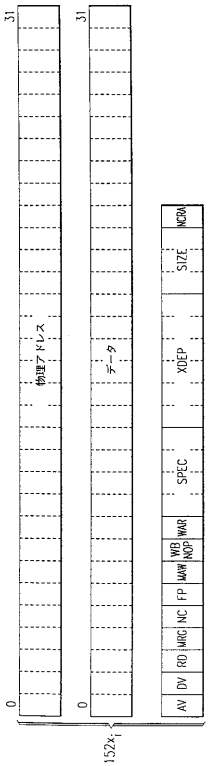
【図４】


システム クロック
内部クロック
Xパイプ
Yパイプ

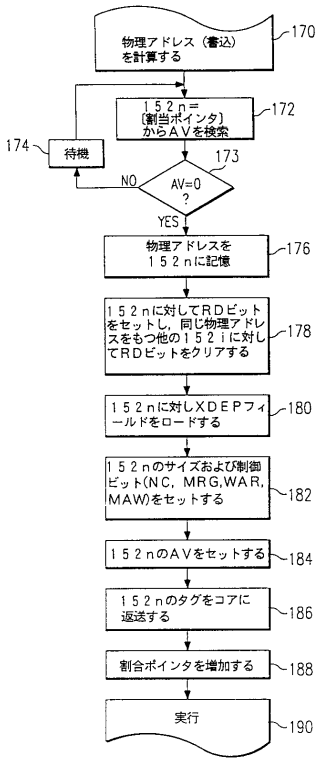
【図５】


コア２０へ
EXX又は EXYより
コア (X又はY) へ
TAGx
TAGy
割当 x
リタイア x
割当 y
リタイア y
書込バッファ 制御ロジック
BPx
BPy
SRCx
SRCy
WB ENTRY
ATU50より
シフタ
メインラッチ
書込集合ラッチ
FL. PT. データラッチ
FPDV
FPUより
デュアル キャッシュポート
キャッシュへ

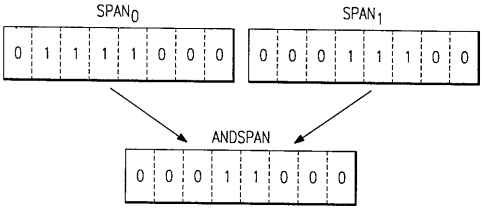
【図６】


物理アドレス
データ
NCRA
SIZE
XDEP
SPEC
WAR
WB NOP
MAW
FP
NC
MRG
RD
DV
AV

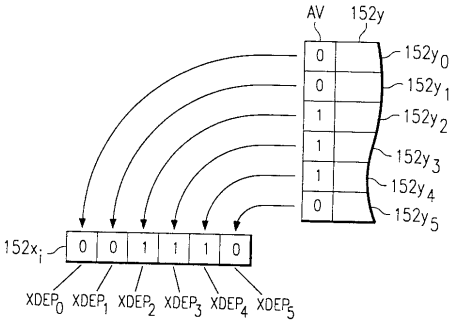
【図７】



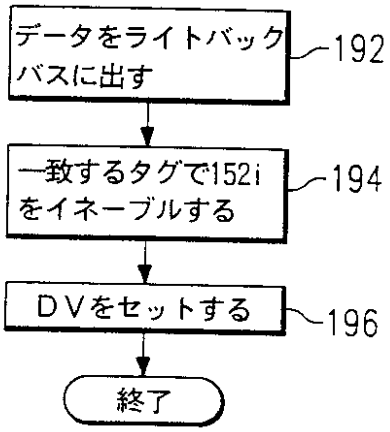
【図８】



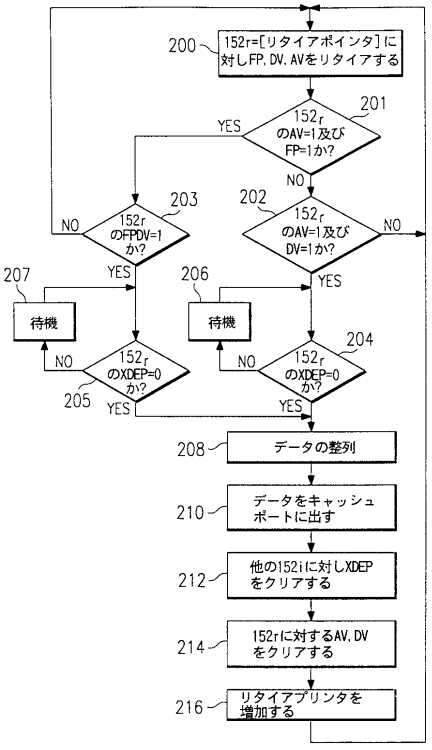
【図９】



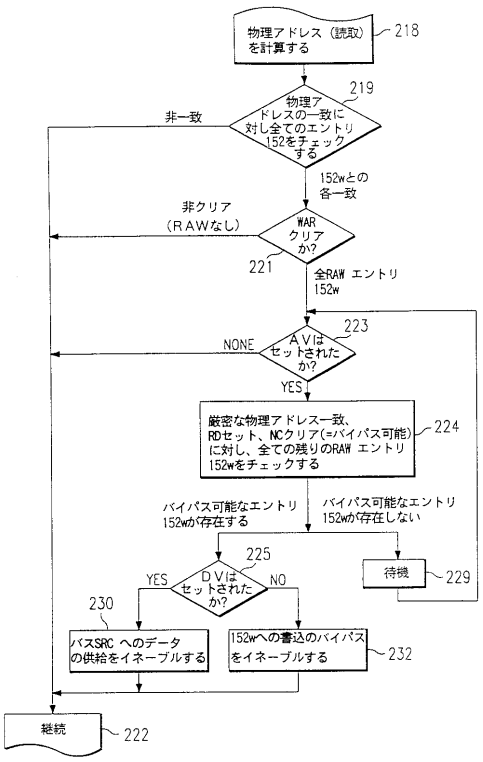
【図１０】



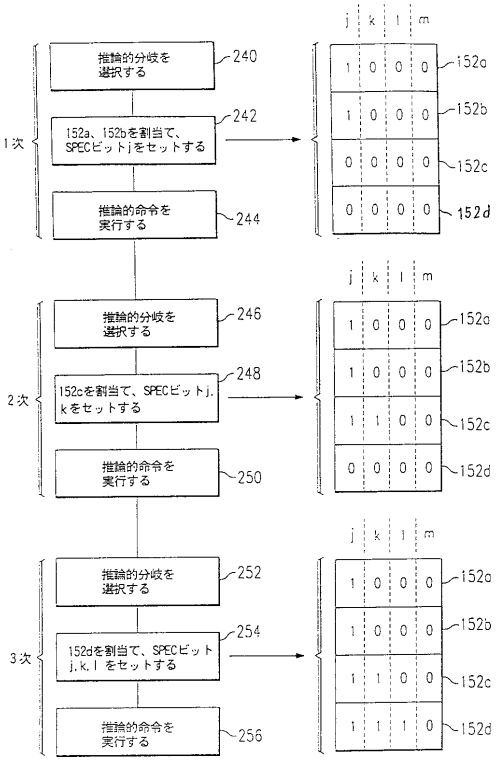
【図１１】



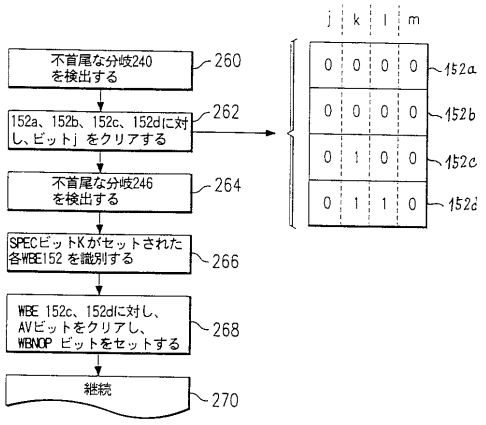
【図１２】



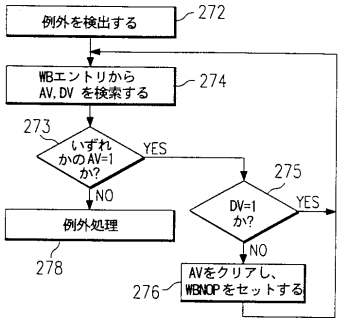
【図１３】



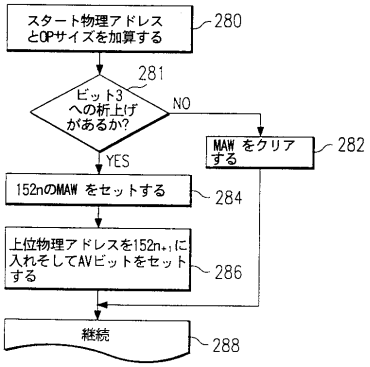
【図１４】



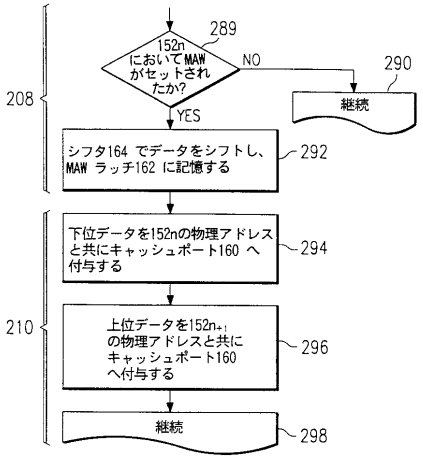
【図１５】



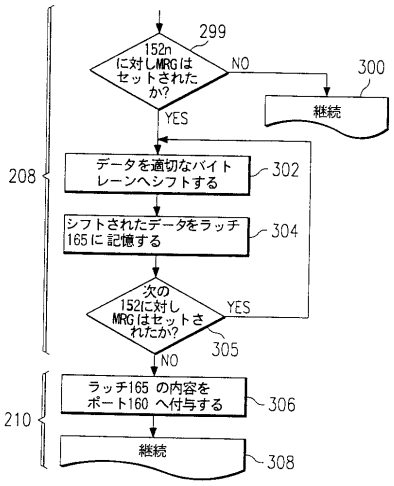
【図１６】



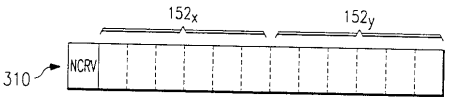
【図１７】



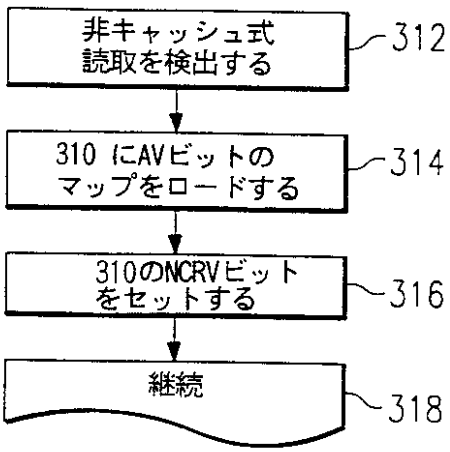
【図１８】



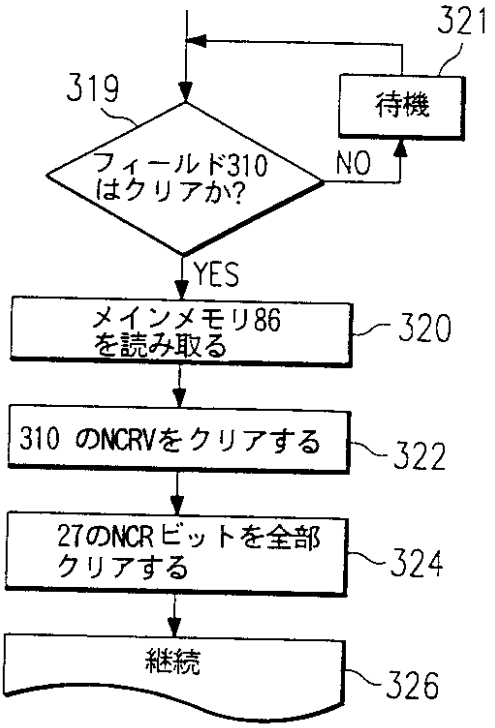
【図１９】



【図２０】



【図２１】



フロントページの続き

(31)優先権主張番号 08/138790
(32)優先日 平成5年10月18日(1993.10.18)
(33)優先権主張国 米国(US)
(31)優先権主張番号 08/139596
(32)優先日 平成5年10月18日(1993.10.18)
(33)優先権主張国 米国(US)
(31)優先権主張番号 08/139598
(32)優先日 平成5年10月18日(1993.10.18)
(33)優先権主張国 米国(US)

(74)代理人 100096194
弁理士 竹内 英人
(74)代理人 100074228
弁理士 今城 俊夫
(74)代理人 100084009
弁理士 小川 信夫
(74)代理人 100082821
弁理士 村社 厚夫
(74)代理人 100086771
弁理士 西島 孝喜
(74)代理人 100084663
弁理士 箱田 篤
(72)発明者 マーク ブルーム
アメリカ合衆国 テキサス州 ７５００７ キャロルトン ブルックシア ドライヴ ２７１１
(72)発明者 ロール エイ ガリベイ ジュニア
アメリカ合衆国 テキサス州 ７５０８０ リチャードソン ３ビー メルローズ ３３３
(72)発明者 マルク エイ コートロマニ
アメリカ合衆国 テキサス州 ７５００２ アレンウイロウ ブルック ７１０
(72)発明者 マーク ダブリュー ハーヴィン
アメリカ合衆国 テキサス州 ７５２５２ ダラス１５６ プレストン ロード １７６０１
(72)発明者 ニタル パトワ
アメリカ合衆国 テキサス州 ７５０９３ プラノ１５１７ オールド シェパード プレイス ５０００

審査官 後藤 彰

(56)参考文献 特開昭６１－１０７４３４（ＪＰ，Ａ）
米国特許第４７６０５２０（ＵＳ，Ａ）
特開昭５９－ ９１５４７（ＪＰ，Ａ）
特開昭６３－２０４３２９（ＪＰ，Ａ）
特開平１－２５５９３３（ＪＰ，Ａ）
特開平３－２５６１９（ＪＰ，Ａ）
特開平３－２１８５２４（ＪＰ，Ａ）
特開平４－１０７６３４（ＪＰ，Ａ）
特表平６－５０１１２２（ＪＰ，Ａ）
国際公開第９３／０１５４５（WO，Ａ１）

(58)調査した分野(Int.Cl.[7]，ＤＢ名)

G06F 9/38
G06F 12/08