(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第3756411号
(P3756411)
(45)発行日　平成18年3月15日(2006.3.15)　(24)登録日　平成18年1月6日(2006.1.6)

(51)Int.Cl.　F I
G06F 9/38 (2006.01)　G06F 9/38 350A

請求項の数 3　(全 25 頁)

(21)出願番号　特願2001-2040(P2001-2040)
(22)出願日　平成13年1月10日(2001.1.10)
(65)公開番号　特開2001-209538(P2001-209538A)
(43)公開日　平成13年8月3日(2001.8.3)
審査請求日　平成16年8月10日(2004.8.10)
(31)優先権主張番号　09/490389
(32)優先日　平成12年1月24日(2000.1.24)
(33)優先権主張国　米国(US)

(73)特許権者　398038580
ヒューレット・パッカード・カンパニー
HEWLETT-PACKARD COMPANY
アメリカ合衆国カリフォルニア州パロアルト　ハノーバー・ストリート　3000
(74)代理人　100081721
弁理士　岡田　次生
(72)発明者　ロニー・リー・アーノルド
アメリカ合衆国80528コロラド州フォート・コリンズ、スティルウォーター・クリーク・ドライブ　2200

最終頁に続く

(54)【発明の名称】データハザードを検出するシステム

(57)【特許請求の範囲】
【請求項１】
コンピュータプログラムの命令を処理するシステムであって、
第１のステージにおける第１の命令と第２のステージにおける第２の命令とを同時に処理するように構成されており、該第１のステージは該第１の命令に関連する第１の属性データのセットを送信し該第１の命令に関連する第１の符号化レジスタ識別子を送信するよう構成され、該第２のステージは該第２の命令に関連する第２の属性データのセットを送信し該第２の命令に関連する第２の符号化レジスタ識別子を送信するよう構成されているパイプラインと、
前記第１のステージに結合され、前記第１の符号化レジスタ識別子を受信するよう構成され、該第１の符号化レジスタ識別子を復号化して第１の復号化レジスタ識別子にし、該第１の復号化レジスタ識別子を送信するよう構成された第１のデコーダと、
前記第２のステージに結合され、前記第２の符号化レジスタ識別子を受信するよう構成され、該第２の符号化レジスタ識別子を復号化して第２の復号化レジスタ識別子にし、該第２の復号化レジスタ識別子を送信するよう構成された第２のデコーダと、
前記第１および第２のデコーダとインタフェースされ、前記第１および第２の復号化レジスタ識別子を受信すると共に、前記第１および第２の属性データのセットを受信するよう構成され、該第１および第２の復号化レジスタ識別子と該第１および第２の属性データのセットとに基づいてデータハザードを検出するよう構成された比較ロジックと、
前記第１のデコーダおよび前記比較ロジックに結合され、前記第１の復号化レジスタ識

別子および前記第１の属性データのセットを受信し、該第１の属性データのセットを該第１の復号化レジスタ識別子と組み合わせるよう構成された第１の属性インタフェースと、
前記第２のデコーダおよび前記比較ロジックに結合され、前記第２の復号化レジスタ識別子および前記第２の属性データのセットを受信し、該第２の属性データのセットを該第２の復号化レジスタ識別子と組み合わせるよう構成された第２の属性インタフェースと、
を含むシステム。
【請求項２】
コンピュータプログラムの命令を処理する方法であって、
命令を処理システムのパイプラインに送信するステップと、
前記命令が前記パイプラインの第１の部分によって処理されている間に、該命令に関連する符号化レジスタ識別子を復号化して第１の復号化レジスタ識別子にするステップと、
該命令が該パイプラインの第２の部分によって処理されている間に、該符号化レジスタ識別子を復号化して第２の復号化レジスタ識別子にするステップと、
該命令に関連する第１の属性データのセットを送信するステップと、
該命令に関連する第２の属性データのセットを送信するステップと、
前記第１の復号化レジスタ識別子および前記第１の属性データのセットを、他の命令に関連する復号化レジスタ識別子および属性データと比較するステップと、
前記第２の復号化レジスタ識別子および前記第２の属性データのセットを、他の命令に関連する復号化レジスタ識別子および属性データと比較するステップと、
前記比較するステップに基づいてデータ従属性ハザードを検出するステップと、
前記第１の属性データのセットを前記第１の復号化レジスタ識別子と組み合わせるステップと、
前記第２の属性データのセットを前記第２の復号化レジスタ識別子と組み合わせるステップと、を含む方法。
【請求項３】
コンピュータプログラムの命令を処理する方法であって、
パイプラインの第１のステージにおいて、第１の符号化レジスタ識別子　に関連する、前記プログラムの第１の命令を処理するステップと、
第１のステージから第１の符号化レジスタ識別子を受け取るステップと、
受け取った第１の符号化レジスタ識別子を、第１の復号化レジスタ識別子に復号化するステップと、
前記第１の復号化レジスタ識別子と、前記第１の命令に関連する第１の属性データのセットを組み合わせて、第１の組み合わせたデータのセットを形成するステップと、
前記プログラムの第２の命令に関連する、第２の符号化レジスタ識別子を受け取るステップと、
受け取った第２の符号化レジスタ識別子を、第２の復号化レジスタ識別子に復号化するステップと、
前記第２の復号化レジスタ識別子と、前記第２の命令に関連する第２の属性データのセットを組み合わせて、第２の組み合わせたデータのセットを形成するステップと、
第１および第２の組み合わせたデータのセットを比較するステップと、
該比較するステップに基づいて、前記第１および第２の命令の中にデータハザードが存在するか否かを検出するステップと、を含む方法。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】
本発明は、一般にデータ処理技術に関し、特に、コンピュータプログラムの命令を処理し、命令に関連するレジスタ識別子と属性データとを比較することにより命令間のデータハザードを検出するシステムおよび方法に関する。
【０００２】
【従来の技術】

多くのプロセッサの性能を向上させるために、パイプライン処理が開発されている。パイプライン処理では、プロセッサは、複数の命令を同時に処理することができる少なくとも１つのパイプラインを備えている。従って、パイプラインにおける１つの命令の実行が、そのパイプラインにおける先行する命令の実行の結果の使用可能になる前に始まる場合があり、その結果、データ従属性ハザードからエラーが発生する可能性がある。

【０００３】

パイプラインによって実行されるある命令が他の命令の実行によって生成されたデータを利用する場合、データ従属性が存在し、他の命令によって生成されたデータをある命令によって使用可能でない場合、そのデータ従属性によりデータ従属性ハザードが発生する。例えば、後の命令が、実行時に、前の命令の実行によって生成されるデータを利用する場合（例えば、後の加算命令が、先のロード命令によって検索されたデータを利用する場合）がある。前の命令の実行からのデータが使用可能となる前に後の命令が実行されると、後の命令は誤ったデータを利用し、その結果データ従属性エラーとなる。従って、後の命令によって利用されるデータが使用可能となるまで、またはデータ従属性エラーが発生するまで、２つの命令間にデータ従属性ハザードが存在する。

【０００４】

【発明が解決しようとする課題】

言うまでもなく、データ従属性エラーを防止できるように、データ従属性ハザードを検出することが重要である。しかしながら、データ従属性ハザードを検出する回路は複雑なことが多く、プロセッサ内で相対的に大容量の領域を利用することが多々ある。これは、特に、命令を同時に実行する複数のパイプラインを含むスーパスカラプロセッサにおいて当てはまる。この場合、あるパイプラインにおける命令は、同じパイプラインにある他の命令との従属性を有するだけでなく、他のパイプラインにある他の命令との従属性をも有する可能性がある。従って、データ従属性ハザードを適切にチェックするために、あるパイプラインの第１の命令は、第１の命令とデータ従属性ハザードを共有する可能性のある各パイプラインにおける各命令と比較されなければならない。従って、プロセッサ内のパイプラインの数が増大するにつれて、データ従属性ハザードを規定するデータ従属性を検出するために必要な回路および複雑さが、大幅に増大する。

【０００５】

さらに、データ従属性ハザードが検出されると、データ従属性ハザードに関連する命令は、データ従属性ハザードが無くなるまで、エラーを防止するためにしばしばストール（stall）される。命令がストールされると、命令の処理は一時的に停止する。一般に、ストールは命令を処理するために必要な時間を増大させるため、可能な限りストールの発生および持続時間を制限することが望ましい。

【０００６】

このように、本業界において、コンピュータプログラムの命令間のデータハザードを検出するための最小限の複雑さおよび回路構成を有する効率的な処理システムに対する要望が存在する。

【０００７】

【課題を解決するための手段】

本発明は、上述したような従来技術の欠点および欠陥を克服する。一般に、本発明は、コンピュータプログラムの命令を処理し、命令間のデータハザードを検出するシステムおよび方法を提供する。

【０００８】

構造上、本発明のシステムは、少なくとも１つのパイプラインと、第１のデコーダと、第２のデコーダと、比較ロジックと、を利用する。パイプラインは、コンピュータプログラムの命令を受信し、同時に処理する。第１のデコーダおよび第２のデコーダは、パイプラインに結合され、パイプラインによって処理されている命令に関連するレジスタ識別子を復号化する。比較ロジックは、第１のデコーダおよび第２のデコーダとインタフェースされ、パイプラインによって処理されている命令のステータスおよび／またはタイプを示す

属性データとともに、復号化レジスタ識別子を受信する。比較ロジックは、復号化レジスタ識別子および属性データを、他の命令に関連する他の復号化レジスタ識別子および属性データと比較することにより、データハザードを検出する。

【０００９】

本発明の他の特徴によれば、属性インタフェースは復号化レジスタ識別子および属性データを受信し、属性データを復号化レジスタ識別子と組み合わせる。

【００１０】

また、本発明は、コンピュータプログラムの命令を処理する方法を提供するものとして見ることもできる。本方法は、以下のステップにより広く概念化することができる。すなわち、処理システムのパイプラインに命令を送信するステップと、命令がパイプラインの第１の部分によって処理されている間に、命令に関連する符号化レジスタ識別子を復号化して第１の復号化レジスタ識別子にするステップと、命令がパイプラインの第２の部分によって処理されている間に、符号化レジスタ識別子を復号化して第２の復号化レジスタ識別子にするステップと、命令に関連する第１の属性データのセットを送信するステップと、命令に関連する第２の属性データのセットを送信するステップと、第１の復号化レジスタ識別子および第１の属性データのセットを、他の命令に関連する復号化レジスタ識別子および属性データと比較するステップと、第２の復号化レジスタ識別子および第２の属性データのセットを、他の命令に関連する復号化レジスタ識別子および属性データと比較するステップと、比較するステップに基づいてデータ従属性ハザードを検出するステップと、である。

【００１１】

本発明の他の特徴および利点は、当業者にとって以下の詳細な説明により明らかとなろう。かかるすべての特徴および利点は、本明細書において本発明の範囲内に含まれ、特許請求の範囲によって保護されるよう意図されている。

【００１２】

【発明の実施の形態】

本発明は、一般に、コンピュータプログラムの命令を実行する処理システム内のデータハザードを効率的に検出することによって、データハザードからのエラーを防止することができるシステムおよび方法に関する。本発明の原理を説明するために、従来のスーパスカラ処理システム15（以下、単に処理システム15とする）を示す図１を参照する。処理システム15は、コンピュータプログラムの命令を受信し各命令を複数のパイプライン21のうちの１つに割当てる命令分配ユニット18を含んでいる。各パイプライン21は、そのパイプライン21が受信した各命令を実行するよう構成されている。

【００１３】

通常、各パイプライン21は、特定のタイプの命令（例えば、整数演算、浮動小数点演算、メモリ演算等）のみを処理するように構成されている。従って、命令分配ユニット18は、各命令をその命令と適合したパイプライン21にのみ割当てるよう構成されている。さらに図１では、簡単にするために比較ロジック22または24が１つのパイプライン21に結合されているように示されているが、各パイプライン21が比較ロジック22または24に同様に結合されている、ということに留意するべきである。

【００１４】

図１に示すように、パイプライン21は典型的に命令を複数ステージで処理する。本明細書で使用されているように、「ステージ」とは、命令を処理しまたそのステージに入力されるデータのタイミングをクロック信号のエッジに応答して制御できるようにその入力でラッチを含む、パイプライン21の任意の部分である。図１に示すパイプライン21は、命令を４つのステージで処理する。すなわち、レジスタステージ25、実行ステージ28、例外検出ステージ32および書込みステージ35である。他の実施の形態では、処理システム15は、他のタイプのステージおよびそのステージの組合せで命令を処理することが可能である。

【００１５】

図１に示す処理システム15において、パイプライン21の１つによって受信される命令は、

まずレジスタステージ25において処理される。レジスタステージ25では、命令の実行に必要なあらゆるオペランドが取得される。オペランドが取得されると、命令は、命令が実行される実行ステージ28に入る。実行ステージ28で命令が実行された後、その命令は、例外検出ステージ32に入る。例外検出ステージ32は、例えばデータの低信頼性を示す実行中のオーバラン等のステータスをチェックする。例外検出ステージ32が完了した後、命令は書込みステージ35に入る。書込みステージ35は、レジスタファイル39内のレジスタ37かまたはメモリロケーションに実行ステージ28の結果を書込む。

【００１６】

典型的に、パイプライン21の各ステージ25、28、32、35は、一度に１つの命令のみを処理し、ステージ25、28、32、35がそれぞれの命令を同時に処理することで各パイプライン21は複数の命令を処理することができる。例えば、図１に示す処理システム15において、パイプライン21の１つは４つの命令を同時に処理することができ、そのときパイプライン21の各ステージ25、28、32、35は４つの命令のうちの１つを実行している。さらに、各パイプライン21は、他のパイプライン21が他の命令を処理している時に命令を処理することができる。従って、図１に示す処理システム15により、相対的に大量の命令を同時に処理することが可能である。

【００１７】

タイミングを制御するために、命令は、一般的にクロック信号のエッジに応答してステージ25、28、32、35を通ってステップする。例えば、書込みステージ35における命令は、レジスタステージ25、実行ステージ28および例外検出ステージ32における命令が、それぞれ同じパイプライン21の実行ステージ28、例外検出ステージ32および書込みステージ35にステップインするのと同じクロックエッジで、パイプライン21からステップアウトする。しかしながら、典型的に、ステージがデータ従属性エラー無しに命令を完全に処理することができるまで、その命令はステージ25、28、32または35からステップアウトしないようになっている。

【００１８】

この点で、ステージ25、28、32または35における命令の処理がクロック信号の次のアクティブなエッジより前にデータ従属性エラー無しに完了することができない場合は、ステージ25、28、32または35において命令をストールさせる必要があるときがある。例えば、例外検出ステージ32および書込みステージ35における命令の処理がデータ従属性エラー無しに完了し、これらステージ32、35の命令が、それぞれクロック信号の次のエッジでそれぞれのステージ32、35からステップアウトする場合がある。しかしながら、同じパイプライン21の実行ステージ28における命令が、次のクロックエッジが発生する前に、使用できない（例えば、先のロード命令によってまだ検索されているような）データを利用する場合がある。従って、命令の処理が次のクロックエッジの発生する前にエラー無しに完了することができないので、実行ステージ28における命令は、次のクロックエッジにおいて実行ステージ28からステップアウトされない。言い換えれば、実行ステージ28における命令はストールされる。この命令は、実行ステージ28がデータ従属性エラーなしに命令を完全に処理することができるまで、ストールされたままでなければならない。1999年9月７日に出願された「Superscalar Processing System and Method for Efficiently Performing In-Order Processing of Instructions」と題する、出願番号第09/390,199号の米国特許出願は、データ従属性ハザードを防止するために命令をストールする適切な処理をより詳細に述べており、参照により本明細書に援用される。

【００１９】

各ステージ25、28、32、35は、通常１度に１つの命令のみを処理するよう構成されているため、上述した例において同じパイプライン21のレジスタステージ25における命令もまた、少なくとも実行ステージ28における命令が実行ステージ28からステップアウトすることができるようになるまで、レジスタステージ25においてストールされていなければならない。従って、ステージ25、28、32または35において命令がストールされているとき、同じパイプライン21によって処理されている後の各命令もまた、該後の命令が次のステップ28

、32または35にステップする用意ができている場合であっても、ストールされていなければならない。

【００２０】

命令がパイプライン21を通ってステップするにつれて、命令のうちの少なくとも１つをストールさせることで、データ従属性ハザードを検出してデータ従属性エラーを防止することが通常望ましい。通常、データ従属性ハザードは、データ従属性を検出することによって、およびデータ従属性に関連するデータがまだ使用可能でないと判断することによって検出される。従って、２つの命令間にデータ従属性ハザードが存在するか否かを検出するために、以下の判断が通常行われる。すなわち、１）２つの命令間にデータ従属性が存在するか、および２）データ従属性に関連するデータ（すなわち、命令のうちの１つによって生成され他の命令によって利用されるデータ）が使用可能であるか、である。

【００２１】

２つの命令間のデータ従属性は、通常、命令に関連するレジスタ識別子を比較することによって検出される。この点で、各命令は通常、その命令によりいずれのレジスタ37が使用されるかを示す少なくとも１つのレジスタ識別子を含む。例えば、データを処理する（例えば、書込む）命令（「プロデューサ」と呼称する）は、プロデューサを実行する時にデータが書込まれるべきレジスタを識別するレジスタ識別子を含み、格納されたデータまたはプロデューサによって生成されるデータを利用する（例えば、検索する）命令（「コンシューマ」と呼称する）は、コンシューマを実行する時にデータが検索されるべきレジスタ37を識別するレジスタ識別子を含む。パイプライン21のうちの１つにおける後のステージ28、32または35が、前のステージ25、28または32によって処理されているコンシューマと同じレジスタ識別子を有するプロデューサを処理している場合、２つの命令間にはデータ従属性が存在する。さらに、プロデューサが生成しコンシューマが使用するデータが、コンシューマにはまだ使用不可能である場合、２つの命令間のデータ従属性はデータ従属性ハザードを生成する。

【００２２】

本明細書において、命令が他のステージによって処理された後にあるステージで処理された場合、そのステージはその他のステージより「後」である。例えば、図１において、実行ステージ28はレジスタステージ25より「後」であり、レジスタステージ25は実行ステージ28より「前」である。

【００２３】

大部分の命令は、実行時にデータの検索とデータの格納を共に行うという点で、コンシューマでもプロデューサでもあることに留意するべきである。本明細書で使用されているように、「プロデューサ」と呼称する命令のレジスタ識別子は、命令が別のレジスタ37からデータを検索し、そのため別のレジスタ識別子に関連付けられる場合があっても、その命令がデータを格納するレジスタ37を識別する。さらに、「コンシューマ」と呼称する命令のレジスタ識別子は、その命令が別のレジスタ37にデータを格納し、そのため別のレジスタ識別子に関連付けられる場合があっても、その命令がデータを検索するレジスタ37を識別する。

【００２４】

コンシューマのオペランドはレジスタステージ25で取得されるので、レジスタステージ25における各コンシューマのレジスタ識別子は、通常、後のステージ28、32、35における各プロデューサのレジスタ識別子と比較されて、データ従属性ハザードを生成するあらゆるデータ従属性がレジスタステージ25におけるコンシューマに対して存在するか否かが判断される。このようなレジスタ識別子の比較を可能にするために、各命令に関連するレジスタ識別子は、命令と共にパイプライン21内を通ってステップすることもある。

【００２５】

ここで、レジスタ識別子は通常ｎビットの符号化された値であるが、図２の従来のシステムによって示すように、まずデコーダ42によってｍビット値（ｍは通常2^nである）に復号化される。値ｍは、処理システム15に関連するレジスタ37の数に対応し、ｍビットレジ

スタ識別子の各ビットは、特定のレジスタ37に対応する。デコーダ42は、ｎビットレジスタ識別子がいずれのレジスタ37を識別するかを判断し、識別したレジスタ37に対応するｍビットレジスタ識別子のビットをアサート（assert）するよう構成されている。ｍビットレジスタ識別子の残りのビットはデアサート（deassert）される。従って、あるレジスタ37に関連するビットがアサートされたとき、ｍビットレジスタ識別子はレジスタのうちのそのレジスタを識別する。

【００２６】

ｍビットレジスタ識別子は、そのレジスタ識別子に関連する命令がラッチ56、58、61、63によってそれぞれステージ25、28、32、35にラッチされるのと同じエッジで、ラッチ44、46、48、52によってそれぞれステージ25、28、32、35にラッチされる。従って、レジスタ識別子とその関連する命令は、同時に同じステージ25、28、32または35に存在しなければならない。図２に示すように、レジスタ識別子に関連する命令は、ステージ25、28、32、35にそれぞれラッチされると、処理回路66、69、72、75によってそれぞれ処理される。処理回路66、69、72、75は、ステージ25、28、32、35について以下に述べる機能をそれぞれ実行する。すなわち、処理回路66はオペランドを取得し、処理回路69は命令を実行し、処理回路72は例外に対してチェックを行い、処理回路75は命令の実行によって生成されたデータをレジスタ37またはメモリのロケーションに書込む。

【００２７】

ステージ25、28、32、35におけるｍビットレジスタ識別子は、比較ロジック22に送信される。比較ロジック22は、ｍビットレジスタ識別子を比較して、ｍビットレジスタ識別子に関連する命令のいずれかの間にデータ従属性が存在するか否かを判断するよう構成されている。この場合、レジスタステージ25におけるコンシューマが後のステージ28、32、35におけるプロデューサのいずれかとのデータ従属性を有しているか否かを判断するために、比較ロジック22は、レジスタステージ25におけるコンシューマのｍビットレジスタ識別子を、実行ステージ28、例外検出ステージ32および書込みステージ35におけるプロデューサのｍビットレジスタ識別子の各々と比較する。ステージ28、32または35のいずれかにおけるプロデューサの比較したｍビットレジスタ識別子のいずれかが、レジスタステージ25におけるコンシューマのｍビットレジスタ識別子と一致する場合、比較ロジック22は、２つの一致するレジスタ識別子に関連する命令間にデータ従属性が存在すると判断する。さらに、レジスタステージ25におけるコンシューマとのデータ従属性を有するステージ28、32または35における命令のうちの１つによって生成されたデータが、レジスタステージ25におけるコンシューマによってまだ使用可能でない場合、比較ロジック22は、２つの命令間にデータ従属性ハザードが存在すると判断する。

【００２８】

図３は、パイプライン21の１つにおいて命令を処理し、データ従属性ハザードを生成するデータ従属性をチェックするために従来から使用されている回路の別の実施の形態である。ｎビットの符号化レジスタ識別子に関連する命令は、上述したように、ステージ25、28、32、35にラッチされ、処理回路66、69、72、75によって処理される。しかしながら、ｎビットレジスタ識別子は、デコーダ42（図２）によって復号化されない。代りに、ｎビットレジスタ識別子は、命令がそれぞれステージ25、28、32、35にラッチされるのと同じエッジで、それぞれラッチ87、89、91、93によってステージ25、28、32、35にラッチされる。従って、ｎビットレジスタ識別子は、ｎビットレジスタ識別子に関連する命令と共に、ステージ25、28、32、35を通ってステップする。

【００２９】

各ステージ25、28、32、35におけるｎビットレジスタ識別子は比較ロジック24に送信される。比較ロジック24は、レジスタ識別子を比較してｎビットレジスタ識別子に関連する命令のいずれかの間にデータ従属性ハザードを規定するデータ従属性が存在するか否かを判断するよう構成されている。この場合、レジスタステージ25におけるコンシューマが、後のステージ28、32および／または35におけるプロデューサのいずれかとのデータ従属性を有するか否かを判断するために、比較ロジック24は、レジスタステージ25におけるコンシ

ューマのｎビットレジスタ識別子を、実行ステージ28、例外検出ステージ32および書込みステージ35におけるプロデューサのｎビットレジスタ識別子の各々と比較する。ｎビットレジスタ識別子が符号化されているため、比較ロジック24は、ｎビットコンパレータを使用して、レジスタステージ25におけるレジスタ識別子が、ステージ28、32および／または35におけるレジスタ識別子のいずれかと一致するか否かを判断する。ステージ28、32または35のいずれかにおけるプロデューサの比較されたｎビットレジスタ識別子がレジスタステージ25におけるコンシューマのｎビットレジスタ識別子と一致する場合、比較ロジック24は、２つの一致するレジスタ識別子に関連する命令間にデータ従属性が存在すると判断する。さらに、レジスタステージ25におけるコンシューマとデータ従属性を有する、ステージ28、32または35における命令のうちの１つによって生成されるデータが、レジスタステージ25におけるコンシューマによって使用可能でない場合、比較ロジック24は、２つの命令間にデータ従属性ハザードが存在すると判断する。

【００３０】

なお、図２および図３に示す回路は、簡単のために、単一のパイプライン21における命令のレジスタ識別子のみをステージ25、28、32、35を通ってステップさせていることに留意されたい。さらに、上述した回路は、パイプライン21によって処理される各命令に対する単一のレジスタ識別子のみを処理する。多くの命令は複数のレジスタ識別子を含むので、同じ命令に対する複数のレジスタ識別子を扱うために、必要に応じて追加の回路を実装しなければならない。さらに、レジスタステージ25におけるコンシューマのレジスタ識別子は、同じパイプライン21のステージ28、32および／または35におけるプロデューサのレジスタ識別子と比較されなければならないだけでなく、任意の他のパイプライン21における後のステージ28、32および／または35のいずれかにおける各プロデューサのレジスタ識別子とも比較されなければならない。従って、パイプライン21の数が増加するにつれて、データ従属性ハザードを検出するための配線および他の回路は大幅に増加する。

【００３１】

特に、プロデューサが生成したデータがコンシューマに使用可能となるために必要な待ち時間が一様でも一定でもない場合、データ従属性ハザードを検出しデータ従属性エラーを防止するために必要な回路は、相対的に複雑である。さらに、図２において、処理システム15の性能が向上するに従い、通常、レジスタ37の数（ｍ）は相対的に大きくなる。従って、ｍビットレジスタ識別子を各ステージ25、28、32、35にラッチする（すなわち、ステージからステージにｍビットレジスタ識別子を送信する）ために必要な配線の数は、相対的に大きくなる（例えば数百）。この追加の配線は、処理システム15内の貴重な空間を利用し、さらに、処理システム15の配線設計全体を複雑にする。

【００３２】

加えて、図３の符号化レジスタ識別子を比較するｎビットコンパレータ（ｎビットコンパレータ用の配線を含む）もまた、処理システム15内の貴重な空間を利用し処理システム15の配線設計の複雑さを増大させる。さらに、ステージ25、28、32、35を通してレジスタ識別子に関連するｍビットレジスタ識別子および命令を別々にラッチすることにより、ラッチの数が増大し、その結果、処理システム15を実現するために必要な回路および空間の量が増大する。複雑さおよび空間に関連する問題は、パイプライン21の数が増加するに従って悪化する。

【００３３】

一般に、本発明は、データ従属性ハザードを効率的に検出するシステムおよび方法を提供する。図４は、本発明を実現するために利用することができる処理システム100を示す。図４に示すように、処理システム100は、メモリ109に格納されたコンピュータプログラム107からの命令を実行するために、コンピュータシステム105内で使用することができる。

【００３４】

処理システム100は、１つまたは複数のバスを含むことができるローカルインタフェース112を介して、コンピュータシステム105内の他の要素と通信しかつそれらを駆動する。さらに、例えばキーボードまたはマウス等の入力装置114を用いて、コンピュータシステム1

05のユーザからデータを入力することができ、スクリーンディスプレイ116またはプリンタ118を用いて、ユーザにデータを出力することができる。ディスク格納機構121はローカルインタフェース112に接続されて、不揮発性ディスク（例えば、磁気ディスク、光ディスク等）に対しおよび不揮発性ディスクからデータを転送することができる。コンピュータシステム105は、コンピュータシステム105をネットワーク125とデータを交換することができるようにするネットワークインタフェース123に接続されることが可能である。

【００３５】

データ従属性、ひいてはデータ従属性ハザードを検出する回路以外は、処理システム100の構成は従来の処理システム15の構成と同じであることが好ましい。従って図５に示すように、処理システム100は、従来の処理システム15について上述したように、パイプライン132を介してレジスタステージ133、実行ステージ136、例外検出ステージ139および書込みステージ142において命令を処理する。望ましい場合、他のステージおよびステージの他の組合せを介して、パイプライン132によって実行される処理を分割することも可能であることに留意するべきである。さらに、図５において、比較ロジック144は簡単のためにデコーダ155、157、159、161を介して１つのパイプライン132に結合されているように示されているが、好ましい実施の形態では、各パイプライン132が、同様のデコーダを介して同様に比較ロジック144に結合されていることに留意するべきである。

【００３６】

図６に示すように、処理システム100は、処理システム100のタイミングを制御するために使用されるラッチ145、147、149、151を含む。ここで、本技術分野における周知の技術により、ラッチ145、147、149、151は、それぞれ処理回路66、69、72、75に命令をラッチする。従来の処理システム15と同様に、ラッチ145、147、149、151は好ましくはステージ133、136、139、142の各々の処理回路66、69、72、75の各々が１度に１つの命令のみを処理するよう制御される。さらに、図６に示すパイプライン132は、処理回路66、69、72、75の各々について１命令ずつ、４つまでの命令を同時に処理することができる。しかしながら、他の実施の形態では、処理回路66、69、72および／または75のいずれもが１度に複数の命令を同時に処理することができてもよい。

【００３７】

ラッチ145、147、149、151は、ラッチ145、147、149、151によってラッチされている命令により各々定義される符号化ｎビットレジスタ識別子を、デコーダ155、157、159、161にそれぞれ送信するよう構成されている。従って、ラッチ145、147、149、151の各々が、処理回路66、69、72、75にそれぞれ命令を送信する時、ラッチ145、147、149、151の各々はまた、各命令によって定義される符号化ｎビットレジスタ識別子をデコーダ155、157、159、161に送信する。

【００３８】

デコーダ155、157、159、161の各々は、受信したｎビットレジスタ識別子を、受信したｎビットレジスタ識別子と同じレジスタ37を識別するｍビットレジスタ識別子に復号化するよう設計されている。値ｍは、好ましくは処理システム100に関連するレジスタ37の数に対応しており、各ｍビットレジスタ識別子の各ビットは、特定のレジスタ37に対応する。各デコーダ155、157、159、161は、好ましくは受信したｎビットレジスタ識別子によっていずれのレジスタ37が識別されているかを判断し、識別されたレジスタ37に対応するｍビットレジスタ識別子のビットをアサートするよう構成されている。ｍビットレジスタ識別子の残りのビットはデアサートされる。その結果、各ｍビットレジスタ識別子は、アサートされた１つのビットとデアサートされたビットの残りとを有していることになる。アサートされたビットは、受信したｎビットレジスタ識別子を含む命令によって利用されるレジスタ37に対応する。しかしながら、ｎビットレジスタ識別子を復号化するために他の技術を採用してもよく、また、命令に関連するレジスタ37を識別するために、他の識別子を使用してもよい。

【００３９】

各デコーダ155、157、159、161は、復号化ｍビットレジスタ識別子を比較ロジック144に

送信する。比較ロジック144は、受信したｍビットレジスタ識別子の各々を、少なくとも１つの他の受信したｍビットレジスタ識別子と比較して、データ従属性および／またはデータ従属性ハザードが存在するか否かを判断する。２つの命令間のデータ従属性ハザードは、２つの命令間にデータ従属性が無ければ存在しないことに留意されたい。従って、データ従属性ハザードの検出は、データ従属性の検出としても考慮しなければならない。

【００４０】

比較ロジック144の機能を説明するために、従来の処理システム15と同様に、レジスタステージ133におけるコンシューマが、他のステージ136、139または142のいずれかにおける任意のプロデューサとデータ従属性ハザードを有するか否かを判断することが望ましいものとする。この例では、コンシューマはレジスタステージ133にあり、プロデューサは他のステージ136、139、142にあるものとすると、デコーダ155によって復号化されたｍビットレジスタ識別子は、デコーダ157、159、161によって復号化されたｍビットレジスタ識別子の各々と比較されなければならない。

【００４１】

従って、図６に示すように、デコーダ155によって復号化されたｍビットレジスタ識別子は、ＡＮＤロジック164、166、168に送信され、デコーダ157、159、161によって復号化されたｍビットレジスタ識別子は、各々ＡＮＤロジック164、166、168に送信される。そして、ＡＮＤロジック164、166、168の各々は、受信したｍビットレジスタ識別子を比較して、レジスタステージ133において処理されているコンシューマと他のステージ136、139、142のいずれかにおいて処理されているプロデューサのいずれかとの間に、データ従属性が存在するか否かを判断する。レジスタステージ133におけるコンシューマと、まだ自身のデータを生成していない（すなわち、コンシューマに対して使用可能なデータを作成していない）他のステージ136、139、142におけるプロデューサのいずれか１つとの間にデータ従属性がある場合、比較ロジック144は、コンシューマと１つのプロデューサとの間にデータ従属性ハザードが存在すると判断する。そして、制御回路は（図示せず）は、かかる判断に応答してレジスタステージ133においてコンシューマをストールする（または他の処置をとる）ことにより、データ従属性エラーを防止する。

【００４２】

図７は、ＡＮＤロジック164、166、168を実現するために適したロジックを示す。この場合、各ＡＮＤロジック164、166、168は、各々が受信したｍビットレジスタ識別子のうちの１つからのビットと、他の受信したｍビットレジスタ識別子からのビットとを受信する、ｍ個のＡＮＤゲート172を含む。各ＡＮＤゲート172によって受信される両ビットは、好ましくは同じレジスタ37に対応する。従って、２つの受信したｍビットのレジスタ識別子が一致すると、ＡＮＤゲート172の１つは、アサートされた出力を生成しなければならない。従って、ＡＮＤゲート172の出力を分析することができ、ＡＮＤゲート172の出力のいずれかがアサートされると、一致しているｍビットレジスタ識別子に関連する命令間にデータ従属性が存在すると判断することができる。ＡＮＤゲート172の出力を迅速に分析するために、出力の各々をＯＲゲート175（図８）に送信することができる。ＯＲゲート175は、２つの比較されたｍビットレジスタ識別子が一致する場合にのみ、アサートされた出力を生成する。

【００４３】

処理システム100の設計の結果として、データ従属性、さらにはデータ従属性ハザードを検出するために必要な、配線を含む回路および空間の量が、従来の処理システム15と比較して低減される。特に、図２においてステージからステージへレジスタ識別子を送信するために使用される配線の数を大幅に低減することができ、比較ロジック144の実現を、図３の比較ロジック24について述べたようなｎビットコンパレータによる実現より大幅に単純かつ小さくすることができる。加えて、デコーダ155、157、159、161および比較ロジック144の実現に使用される装置は、空間および配線の複雑さを最小化するように容易に配置することができる。さらに、デコーダ155、157、159、161が使用する空間は、ラッチ44、46、48、52（図２）より少なくすることができる。その結果、データ従属性および／ま

たはデータ従属性ハザードを検出する処理システム100の回路を、従来の処理システム15の回路より大幅にコンパクトかつ効率的にすることができる。

【００４４】

図６は、パイプライン132における各命令からの１つのレジスタ識別子を他のレジスタ識別子と比較することができる回路を示していることに留意するべきである。しかしながら、命令が複数のレジスタ識別子を含むことも可能である。従って、図６の回路と同様に、各命令の各レジスタ識別子を確実に検査できるようにするために、追加の回路を実装してもよい。例えば、レジスタステージ133におけるコンシューマは、実行時に２つの異なるレジスタ37からデータを検索してもよく、そのため、２つの異なるレジスタ識別子を含んでもよい。かかる命令に適応するために、レジスタステージ133は、好ましくは比較ロジック144によって他のステージ136、139、142における他のｍビットレジスタ識別子と比較することができるｍビットレジスタ識別子に、他のレジスタ識別子を復号化する、別のデコーダ155を含む。従って、図６に示す設計を必要に応じて拡張することにより、複数のレジスタ識別子に関連する命令に適応させることができるということは当業者には明らかであろう。

【００４５】

さらに、図６は、単一のパイプライン132の回路を示す。スーパスカラシステム100では、異なるパイプライン132における命令間のデータ従属性ハザードを、ゆえにデータ従属性をチェックすることが望ましい。従って、本明細書で述べられている技法に従って、あるパイプライン132における命令のｍビットレジスタ識別子を、同じまたは他のパイプライン132における命令のｍビットレジスタ識別子と比較することによって、異なるパイプライン132の命令間のデータ従属性および／またはデータ従属性ハザードをチェックすることができることは当業者には明らかであろう。例えば、あるパイプライン132のデコーダ155によって復号化されるｍビットレジスタ識別子を、ＡＮＤロジック164、166、168と同様のロジックを介して他のパイプライン132のデコーダ157、159、161によって復号化されるｍビットレジスタ識別と比較して、あるパイプライン132のレジスタステージ133における命令と他のパイプライン132のステージ136、139、142における命令との間に、データ従属性および／またはデータ従属性ハザードがあるか否かを判断することができる。

【００４６】

さらに、本明細書に述べられているように、レジスタステージ133における各コンシューマに関連するレジスタ識別子を、ステージ136、139、142における各プロデューサのレジスタ識別子と比較することが一般的に望ましい。しかしながら、他の実施の形態において、任意の１つのパイプライン132の任意の１つのステージ133、136、139または142からのレジスタ識別子を、任の１つのパイプライン132の任意の１つのステージ133、136、139または142からのレジスタ識別子と比較して、比較されたレジスタ識別子に関連する２つの命令の間にデータ従属性が存在するか否かを判断することができる。

【００４７】

属性データ

システム100の効率をさらに向上させるために、不要なストールを防止する追加の回路を実装して、パイプライン132の命令を処理する際に発生する遅延を低減することができる。この点に関して、パイプラインによって処理されている命令をイネーブル（enable）およびディスエーブル（disable）する述語（predicate）技術が開発されている。イネーブルされた命令はパイプライン132によって実行され、ディスエーブルされた命令は実行されることなくパイプライン132を通過する。2000年1月24日に出願された「System and Method for Providing Predicate Data」と題する、出願番号第09/490,395号の米国特許出願は、プロセッサ性能を向上させるために述語データを使用するプロセスを述べており、この参照をもって本明細書に援用される。

【００４８】

ストールの悪影響を最小限にするために、述語データを分析することにより命令がパイプライン132によって実行されるべきか否かを判断することができる。述語データから命令

が実行されるべきではないと判断することができる場合、該命令と別の命令との間のデータハザードのせいで、該命令がデータエラーを引き起こすことはあり得ない。従って、命令のレジスタ識別子が別の命令のレジスタ識別子と一致したとしても、命令の１つが述語データによってディスエーブルされている場合、実際には命令間にデータハザードは存在しない。従って、命令の一方が述語データによってディスエーブルされている場合、不要なストールを防止するため、比較ロジック144は２つの命令間のデータハザードを検出しないことが望ましい。

【００４９】

さらに、以下で詳細に説明するように、２つの命令のレジスタ識別子が一致する場合であっても、パイプライン132によって処理されている命令のタイプに基づいて、２つの命令間にデータハザードは存在しないと判断することができる場合もある。従って、不要なストールの発生を防止するため、（パイプライン132によって処理されている命令の述語ステータスおよび／またはタイプを示すデータ等の）属性データを分析することができる追加の回路を処理システム100に含めることが望ましい。

【００５０】

図９は、システム100がデータハザードの検出の際に属性データ（すなわち、命令のタイプおよび／または述語ステータスを示すデータ）を分析するために利用することができる、追加の回路183、185、187、189を示す。図９に示すように、デコーダ155、157、159、161によって生成されるｍビットレジスタ識別子は、それぞれ属性インタフェース183、185、187、189に入力され、属性インタフェース183、185、187、189は、それぞれラッチ145、147、149、151から属性データを受信する。そして、属性インタフェース183、185、187、189の各々は、受信した属性データを受信したｍビットレジスタ識別子とインタフェースすることにより、受信したｍビットレジスタ識別子に関連する命令を実行するためにいずれのレジスタ37（図５）が使用されるかを示すと共に命令のタイプおよび／または命令の述語ステータスを示すデータを生成する。従って、比較ロジック191は、命令を実行するために利用されるレジスタ37に基づいて、および命令のタイプおよび／または命令の述語ステータスにも基づいて、先行する命令と他の命令との間にデータハザードが存在するか否かを検出することができる。その結果、比較ロジック191は、比較ロジック144よりさらに正確にデータハザードを検出するよう構成される。

【００５１】

例えば、例示の目的で、レジスタステージ133における非マルチメディア（非ｍｍｕ）のコンシューマが、プロデューサがマルチメディア（ｍｍｕ）命令である場合にのみ例外検出ステージ139におけるプロデューサとのデータハザードを規定すると仮定する。さらに、レジスタステージ133には現在非ｍｍｕコンシューマが存在していると仮定する。比較ロジック144（図６）が属性データを分析せず、従ってパイプライン132によって処理されている命令のタイプおよび命令の述語ステータスを考慮しない場合、１つのパイプライン132のデコーダ159によって受信されるレジスタ識別子が１つのパイプライン132のデコーダ155によって同時に受信されるレジスタ識別子と一致する場合はいつでも、データハザードを検出してしまう。

【００５２】

従って、レジスタステージ133におけるコンシューマと同じレジスタ識別子を有する、パイプライン132の例外検出ステージ139におけるすべてのプロデューサが少なくとも書込みステージ142に進むまで、レジスタステージ133における上述したコンシューマはストールされていなければならない。言い換えれば、少なくとも１つのタイプのプロデューサが少なくとも書込みステージ142に到達するまでは、レジスタステージ133における少なくとも１つのタイプのコンシューマにとって使用不可能であるデータを該プロデューサが生成するので、該コンシューマより前であって該コンシューマと同じレジスタ識別子を有するすべてのプロデューサが少なくとも書込みステージ142に到達するまで、該コンシューマはレジスタステージ133においてストールされていなければならない。そうでなければ、１つのパイプライン132の例外検出ステージ139におけるプロデューサがｍｍｕ命令でありレ

ジスタステージ133におけるコンシューマが非ｍｍｕ命令である場合に、データエラーが発生する可能性がある。

【００５３】

しかし、比較ロジック191は、命令のレジスタ識別子のみならず命令のタイプおよび／または述語ステータスを示す属性データも受信し分析するよう設計されている。従って、２つの命令が同じレジスタ識別子を有する場合であっても、比較ロジック191は、例外検出ステージ139におけるプロデューサとレジスタステージ133におけるコンシューマとの間にデータハザードが存在しないと検出することができる。

【００５４】

この場合、レジスタステージ133のラッチ145は、レジスタステージ133における命令の述語ステータスを示す１ビットの属性データを送信し、レジスタステージ133における命令が特定のタイプの命令（例えば、上述した実施の形態におけるｍｍｕ命令）であるか否かを示す１ビットの属性データを送信する。さらに、例外検出ステージ139におけるラッチ149は、例外検出ステージ139における命令の述語ステータスを示す１ビットの属性データを送信し、例外検出ステージ139における命令が特定のタイプの命令（例えば、上述した実施の形態におけるｍｍｕ命令）であるか否かを示す１ビットの属性データを送信する。属性インタフェース183、187は、それぞれラッチ145、149から属性データを受信し、受信した属性データを示すデータおよびデコーダ155、159からの復号化レジスタ識別子を、ハザード検出回路196に送信する。

【００５５】

ハザード検出回路196は、以下の場合にのみデータハザードを検出するよう設計されている。つまり、１）レジスタステージ133におけるコンシューマが、例外検出ステージ139におけるプロデューサと同じレジスタ識別子を有し、２）レジスタステージ133における命令と例外検出ステージ139における命令とが、共に述語がイネーブルされており、３）レジスタステージ133における命令と例外検出ステージ139における命令のタイプが、２つの命令の間にデータハザードが存在する可能性のあるタイプである、場合である。例示の目的で、（ａ）レジスタステージ133におけるコンシューマが非ｍｍｕ命令であり、例外検出ステージ139におけるプロデューサがｍｍｕ命令である場合、または（ｂ）レジスタステージ133におけるコンシューマがｍｍｕ命令であり、例外検出ステージ139におけるプロデューサが非ｍｍｕ命令である場合にのみ、条件３が満足されるものと仮定する。しかしながら、以下でより詳細に説明する本発明の上述した実施の形態を実現するために使用される回路の修正を行えば、他の命令のタイプおよび／または命令のタイプの他の組合せが条件３を満たすこともある、ということは、当業者には明らかであろう。

【００５６】

上述した３つの条件（すなわち、条件１～３）のいずれかが満たされない場合、ハザード検出回路196はデータハザードを検出しない。その結果、状況によっては、レジスタステージ133におけるコンシューマと同じレジスタ識別子を有するプロデューサが書込みステージ142に到達する前に該コンシューマのストールを防止するかまたは取除くことができ、従って該コンシューマがパイプライン132によってより早く処理されることができる。

【００５７】

図１０は、属性インタフェース183に結合されたデコーダ155の例示的な実現を示す。ここにおいて、デコーダ155は、処理システム100のレジスタ37（図５）とそれぞれ対応するｍ行のデコードロジック202を有する１列のロジックである。デコードロジック202の各行は、デコーダ155に送信されるｎビットレジスタ識別子を受信し、そのｎビットレジスタ識別子がデコードロジック202の行に対応するレジスタ37を識別する場合は、アサートされた出力を送信し、ｎビットレジスタ識別子が別のレジスタ37を識別する場合は、デアサートされた出力を送信するように設計されている。言い換えれば、デコードロジック202の各行は、デコーダ155によって生成されるｍビットレジスタ識別子の１ビットを出力する。

【００５８】

属性インタフェース183は、デコードロジック202の行、すなわち処理システム100のレジスタ37とそれぞれ対応するm行のインタフェースロジック204を含む。インタフェースロジック204の各行は、デコードロジック202の行のうちの１つの出力を受信し、ラッチ145から送信される属性データを受信するよう構成されている。他の属性インタフェース185、187、189によって処理される属性データは、ラッチ147、149、151からそれぞれ受信されることに留意されたい。

【００５９】

ラッチ145は、レジスタステージ133における命令の述語ステータスを示す（すなわち、レジスタステージにおける命令がイネーブルされているか否かを示す）１ビット値と、レジスタステージ133における命令が特定の一つあるいは複数のタイプの命令であるか否かを示す少なくとも１ビットである値と、を属性インタフェース183に送信するよう構成されてもよい。例えば、好ましい実施の形態では、ラッチ145は、レジスタステージ133における命令が述語によりイネーブルされている場合にのみ、ビットのうちの１つ（すなわち、図１１において接続207を介して送信されるビット）をアサートし、レジスタステージ133における命令がｍｍｕ命令である場合にのみ、他のビット（すなわち、接続208を介して送信されるビット）をアサートする。

【００６０】

属性インタフェース183におけるインタフェースロジック204の各行は、デコーダ155から受信されるビットがアサートされているか否かを示すのみでなく、レジスタステージ133における命令の述語ステータスおよび／またはタイプをも示す出力を生成するよう構成されている。例えば、図１０および図１１に示す実施の形態におけるインタフェースロジック204の各行は、２ビット出力を生成してもよい。インタフェースロジック204の単一の行によって出力されるビットのいずれかが、インタフェースロジック204のその行がデコーダ155によって現在受信されているｎビットレジスタ識別子によって識別されるレジスタ37に対応する場合、およびレジスタステージ133における命令が述語によりイネーブルされている場合にのみ、アサートされてもよい。言い換えれば、インタフェースロジック204の各行は、デコーダ155から受信されるビット値がデアサートされている場合か、または接続207を介して送信されるビット値がデアサートされている場合に、両出力ビットをデアサートするように構成される。

【００６１】

さらに属性インタフェース183におけるインタフェースロジック204の各行は、レジスタステージ133における命令は特定のタイプの命令であることをラッチ145から受信される属性データが示す場合にのみ、各出力ビットをそれぞれアサートするよう構成される。例えば、属性インタフェース183についての図１１のＡＮＤゲート209は、レジスタステージ133における命令がｍｍｕ命令である場合にのみアサートされた出力を生成することができ、属性インタフェース183についての図１１のＡＮＤゲート210は、レジスタステージ133における命令が非ｍｍｕ命令である場合にのみアサートされた出力を生成することができる。

【００６２】

ここで、ＡＮＤゲート209の出力がアサートされている場合、レジスタステージ133における命令が、１）イネーブルされており、２）ｍｍｕ命令であり（すなわち、例外検出ステージ139における命令が非ｍｍｕ命令である場合にのみ、例外検出ステージ139における命令とのデータハザードを規定するタイプであり）、３）ＡＮＤゲート209を含むインタフェースロジック204の行に対応するレジスタ37を利用する、ということが分かる。さらに、ＡＮＤゲート210の出力がアサートされている場合、レジスタステージ133における命令が、１）イネーブルされており、２）非ｍｍｕ命令であり（すなわち、例外検出ステージ139における命令がｍｍｕ命令である場合にのみ、例外検出ステージ139における命令とのデータハザードを規定するタイプであり）、３）上記ＡＮＤゲート210を含むインタフェースロジック204の行に対応するレジスタ37を利用する、ということが分かる。

【００６３】

デコーダ159および属性インタフェース187の回路構成は、上述したデコーダ155および属性インタフェース183と同様であってもあるいは同一であってもよい。さらに、ラッチ145と同様、ラッチ149は、例外検出ステージ139における命令がイネーブルされる場合にのみ、属性インタフェース187に送信される属性ビットの１つ（すなわち、図１１において接続207を介して送信されるビット）をアサートし、ラッチ149は、例外検出ステージ139における命令が特定のタイプの命令（例えば、ｍｍｕ命令）である場合にのみ、属性インタフェース187に送信される他の属性ビット（すなわち、図１１において接続208を介して送信されるビット）をアサートする。

【００６４】

属性インタフェース183の出力と同様、属性インタフェース187のインタフェースロジック204（図１０）の各行の出力は、２ビット出力としてもよい。この場合、好ましい実施の形態では、属性インタフェース187についての図１１のＡＮＤゲート209の出力は、例外検出ステージ139における命令が、１）イネーブルされており、２）ｍｍｕ命令であり（すなわち、レジスタステージ133における命令が非ｍｍｕ命令である場合にのみ、レジスタステージ133における命令とのデータハザードを規定するタイプであり）、３）ＡＮＤゲート209を含むインタフェースロジックの行と対応するレジスタ37を利用する場合にのみ、アサートされる。さらに、属性インタフェース187についての図１１のＡＮＤゲート210の出力は、例外検出ステージ139における命令が、１）イネーブルされており、２）非ｍｍｕ命令であり（すなわち、レジスタステージ133における命令がｍｍｕ命令である場合にのみ、レジスタステージ133における命令とのデータハザードを規定するタイプであり）、３）上述したＡＮＤゲート209を含むインタフェースロジック204の行と対応するレジスタ37を利用する場合にのみ、上記の実施の形態ではアサートされる。

【００６５】

ハザード検出ロジック196は、属性インタフェース183、187からの出力を受信して比較し、ハザード検出ロジック196に入力される情報に基づいてデータハザードが存在するか否かを検出するよう構成されている。この場合、ハザード検出ロジック196は、レジスタステージ133におけるコンシューマのレジスタ識別子と例外検出ステージ139におけるプロデューサのレジスタ識別子とが一致し、コンシューマとプロデューサの両方がイネーブルされており、コンシューマとプロデューサとがデータハザードを規定するタイプである（例えば、（ａ）レジスタステージ133におけるコンシューマが非ｍｍｕ命令であり、例外検出ステージ139におけるプロデューサがｍｍｕ命令である、または（ｂ）レジスタステージ133におけるコンシューマがｍｍｕ命令であり、例外検出ステージにおけるプロデューサが非ｍｍｕ命令である）場合にのみ、データハザードを検出する。

【００６６】

図１２は、上述した実施の形態におけるハザード検出ロジック196を実現するために使用することができる回路構成を示す。ハザード検出ロジック196は、ｍ行の比較ロジック211を含む。比較ロジック211の各行は、属性インタフェース183のインタフェースロジック204の単一の行からの出力と、属性インタフェース187のインタフェースロジック204の単一の行からの出力と、を受信するよう構成されている。ハザード検出回路196における比較ロジック211の同じ行に出力を送信する属性インタフェース183、187のインタフェースロジック204の行は、好ましくは同じレジスタ37に対応する。言い換えれば、ハザード検出回路196における比較ロジック211の同じ行に結合されている属性インタフェース183、187のインタフェースロジック204の両方の行は、デコーダ155、159によって同時に受信されるｎビットレジスタ識別子が一致する場合にのみ、アサートされた出力を生成することができる。従って、比較ロジック211の行のいずれもが、属性インタフェース183からのアサートされた出力の少なくとも１ビットと属性インタフェース187からのアサートされた出力の１ビットとを同時に受信しない場合、レジスタステージ133と例外検出ステージ139とにおける命令に関連するｎビットレジスタ識別子は、一致しないか、あるいは、レジスタステージ133かまたは例外検出ステージ139における命令の少なくとも１つがディスエーブルされている。従って、比較ロジック211の１行が、属性インタフェース183からの少なく

とも１つのアサートされたビットと属性インタフェース187からの１つのアサートされたビットとを受信しない限り、ハザード検出ロジック196はデータハザードを検出せず、現時点でデータハザードが検出されないことを示す信号を送信する。

【００６７】

しかしながら、比較ロジック211の行のうちの１つが、属性インタフェース183からのアサートされた出力の少なくとも１ビットと属性インタフェース187からのアサートされた出力の少なくとも１ビットとを同時に受信する場合、レジスタステージ133と例外検出ステージ139とにおける命令に関連するｎビットレジスタ識別子が一致する。従って、レジスタステージ133と例外検出ステージ139との両方における命令がイネーブルされこれら２つの命令間にデータハザードが存在するか否かを検出するために、さらなる分析が実行される。この場合、属性インタフェース183、187からの上記出力が、レジスタステージ133と例外検出ステージ139とにおける命令がデータハザードを生成しないタイプである、ということを示さない限り、アサートされた出力を受信する比較ロジック211の行は、データハザードを検出する。

【００６８】

例えば、上述した例示的な実施の形態において、ハザード検出回路196は、（ａ）属性インタフェース187からの出力が例外検出ステージ139におけるプロデューサがｍｍｕ命令であることを示し、属性インタフェース183からの出力がレジスタステージ133におけるコンシューマが非ｍｍｕ命令であることを示す場合、または（ｂ）属性インタフェース187からの出力が例外検出ステージ139におけるプロデューサが非ｍｍｕ命令であり、属性インタフェース183からの出力がレジスタステージ133におけるコンシューマがｍｍｕ命令であることを示す場合にのみ、データハザードを検出するようにしてもよい。

【００６９】

図１３は、上述した実施の形態についての１行の比較ロジック211を実現するために使用することができる回路を示す。図１３の接続221は、属性インタフェース183（図９）におけるインタフェースロジック204の行に含まれるＡＮＤゲート209（図１１）の出力に結合され、接続223は、属性インタフェース183（図９）におけるインタフェースロジック204の上記行に含まれるＡＮＤゲート210（図１１）の出力に結合されている。さらに、図１３の接続227は、属性インタフェース187（図９）におけるインタフェースロジック204の行に含まれるＡＮＤゲート210（図１１）の出力に結合され、接続229は、属性インタフェース187（図９）におけるインタフェースロジック204の上記行に含まれるＡＮＤゲート209（図１１）の出力に結合されている。

【００７０】

従って、接続231における値は、（１）レジスタステージ133におけるコンシューマのレジスタ識別子が、例外検出ステージ139におけるプロデューサのレジスタ識別子と一致し、（２）レジスタステージ133および例外検出ステージ139における命令が共に述語によってイネーブルされており、（３）レジスタステージ133におけるコンシューマが非ｍｍｕ命令であり、（４）例外検出ステージ139におけるプロデューサがｍｍｕ命令である場合にのみ、アサートされることになる。さらに、接続233における値は、（１）レジスタステージ133におけるコンシューマのレジスタ識別子が、例外検出ステージ139におけるプロデューサのレジスタ識別子と一致し、（２）レジスタステージ133および例外検出ステージ139における命令が共に、イネーブルされた述語であり、（３）レジスタステージ133におけるコンシューマがｍｍｕ命令であり、（４）例外検出ステージ139におけるプロデューサが非ｍｍｕ命令である場合にのみ、アサートされることになる。

【００７１】

その結果、ハザード検出回路196における比較ロジック211の行は、接続231または233における値がアサートされているとき、レジスタステージ133におけるコンシューマと例外検出ステージ139におけるプロデューサとの間のデータハザードを検出する。この場合、ＯＲゲート235によって出力される値は、比較ロジック211の行がデータハザードを検出するか否かを示す。ＯＲゲート235の出力の値がアサートされている場合、比較ロジック211の

行は、レジスタステージ133と例外検出ステージ139とにおける２つの命令間にデータハザードが存在することを示す。逆に、ＯＲゲート235の出力の値がデアサートされている場合、比較ロジック211の行は、比較ロジック211の行によりレジスタステージ133と例外検出ステージ139とにおける２つの命令間にデータハザードが検出されていないことを示す。

【００７２】

図１４に示すように、ハザード検出回路196における比較ロジック211の各行のＯＲゲート235の出力をＯＲゲート252によって組み合わせることにより、ＯＲゲート252の出力が、ハザード検出回路196がデータハザードを検出したか否かを示すようにしてもよい。ＯＲゲート252の出力の値がアサートされている場合、ハザード検出回路196は、レジスタステージ133と例外検出ステージとにおける２つの命令間にデータハザードが存在することを示す。逆に、ＯＲゲート252の出力の値がデアサートされている場合、ハザード検出回路196は、レジスタステージ133と例外検出ステージ139との２つの命令間にデータハザードが存在しないことを示す。

【００７３】

なお、図６に示すシステム100におけるように、レジスタステージ133および例外検出ステージ139以外のステージにおける命令のレジスタ識別子および属性データを比較して、ハザードを検出することもできることに留意するべきである。この場合、デコーダ157、161、属性インタフェース185、189およびハザード検出ロジック194、198の構成は、それぞれデコーダ155、159、属性インタフェース183、187およびハザード検出ロジック196の構成と同様であってよい。さらに比較ロジック144と同様に、比較ロジック191は、本発明に従って、あるパイプライン132の命令のレジスタ識別子および属性データを、他のパイプライン132における別の命令のレジスタ識別子および属性データと比較することにより、データハザードを検出してもよい。

【００７４】

また、本発明の原理を逸脱することなく、図９に示す回路に変更を行うことができる、ということは当業者には明らかであろう。例えば、（述語ステータスを示すビットやまたは命令タイプを示すビット等の）属性データの１ビットのみが属性インタフェース183、185、187、189に送信され、データハザードを正確に検出するために使用されてもよい。この場合、比較ロジック191は、２つの命令のレジスタ識別子が一致した場合であり両方の命令がイネーブルされている場合にのみデータハザードを検出するよう構成されていてもよく、あるいは、２つの命令のレジスタ識別子およびタイプに基づいて、データハザードを検出するよう構成されていてもよい。

【００７５】

加えて、パイプライン132から属性インタフェース183、185、187、189に送信される属性データによって異なるタイプの命令を示すことができる。また、異なる数のビット値が、レジスタ識別子、述語ステータスおよび／または属性情報を表すために送信されてもよい。特に、本明細書で述べられている命令のタイプに付け加えるかまたはそれ以外の命令のタイプがデータハザードを規定してもよく、ステージ133、136、139および／または142の任意の属性データが異なるタイプの命令を示してもよい。属性インタフェース183、185、187および／または189の各々にそれぞれ送信される属性データのセットは、１つまたは複数のビットの情報であってよい。さらに、属性インタフェース183、185、187および／または189およびハザード検出回路194、196および／または198の構成を、他のタイプの命令間のデータハザードを示すように特に変更する必要がある場合もある。

【００７６】

また、データハザードを生成する可能性のある命令のタイプがステージ毎に変化する可能性もある。従って、属性インタフェース183、185、187および／または189の各々に送信される属性データによって示される命令のタイプが異なっている可能性もある。例えば、書込みステージ142にあるときに、レジスタステージ133における特定のタイプの命令とデータハザードを生成する命令が２タイプある場合がある。その結果、属性インタフェース18

9は、属性データの３つのビット、すなわち、書込みステージ142における命令の述語ステータスを示す１ビットと、該命令がレジスタステージ133における命令とのデータハザードを規定する可能性のある２タイプの命令であるか否か、を示す他の２ビットと、を受信する。そして、本発明の原理に従ってこのデータを分析してデータハザードを検出することができる。他のステージ133、136、139および／または142における他のタイプの命令とのデータハザードを規定する可能性のある各ステージ133、136、139および／または142における命令のタイプ（単数および複数）を示す各ステージ133、136、139および／または142からの属性データのみを送信することによって、本発明を実現する回路の大きさを最小限にすることができる。

【００７７】

さらに、望ましければ、属性インタフェース183、185、187、189を図２および／または図３に示す従来のシステム15に組込むことにより、比較回路22および／または24が、レジスタ識別子と同様に属性情報に基づいてデータハザードを検出できるようにすることも可能である。

【００７８】

本発明の原理を利用して、ライトアフタライト（ＷＡＷ）ハザードを検出することもできる。ＷＡＷハザードは、（１）前の命令および後の命令の両方が同じレジスタに書込む場合、および（２）前の命令が実際にレジスタに書込む前、に存在する。データ従属性ハザードの検出と同様、２つの書込み命令が同じレジスタ識別子を含むか否かを判断することにより、ＷＡＷハザードを検出することができる。従って、上述した回路構成を用いて、２つの書込み命令が同じレジスタを利用するときを検出することができる。そして、追加の回路構成を用いて、前の命令からのデータがレジスタに書込まれているか否かを判断することができる。そして、上述した情報を使用して、ＷＡＷハザードが存在するか否かを判断することができる。

【００７９】

本発明の精神および原理から実質的に逸脱することなく、多くの変形および変更を本発明の上記実施の形態に対して行うことができる。

【００８０】

本発明は例として以下の実施形態を含む。

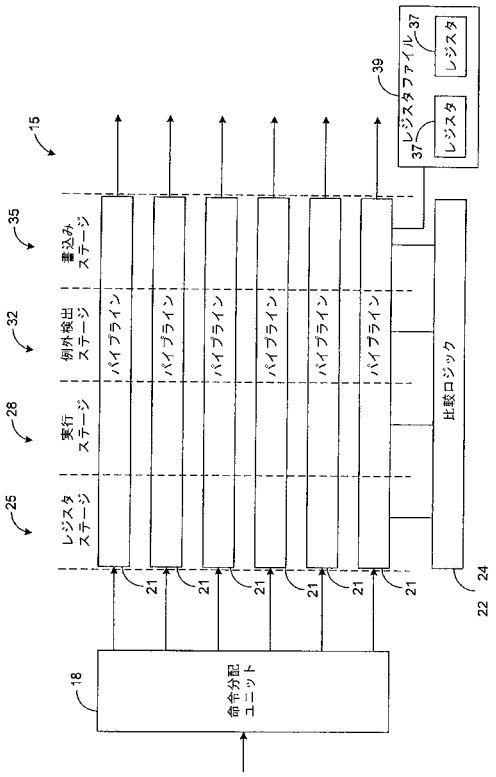
【００８１】

（１）　コンピュータプログラム（107）の命令を処理するシステム（100）であって、
第１のステージ（133、136、139、142）における第１の命令と第２のステージ（133、136、139、142）における第２の命令とを同時に処理するように構成されており、該第１のステージ（133、136、139、142）は該第１の命令に関連する第１の属性データのセットを送信し該第１の命令に関連する第１の符号化レジスタ識別子を送信するよう構成され、該第２のステージ（133、136、139、142）は該第２の命令に関連する第２の属性データのセットを送信し該第２の命令に関連する第２の符号化レジスタ識別子を送信するよう構成されているパイプライン（132）と、
前記第１のステージ（133、136、139、142）に結合され、前記第１の符号化レジスタ識別子を受信するよう構成され、該第１の符号化レジスタ識別子を復号化して第１の復号化レジスタ識別子にし、該第１の復号化レジスタ識別子を送信するよう構成された第１のデコーダ（155、157、159、161）と、
前記第２のステージ（133、136、139、142）に結合され、前記第２の符号化レジスタ識別子を受信するよう構成され、該第２の符号化レジスタ識別子を復号化して第２の復号化レジスタ識別子にし、該第２の復号化レジスタ識別子を送信するよう構成された第２のデコーダ（155、157、159、161）と、
前記第１および第２のデコーダ（155、157、159、161）とインタフェースされ、前記第１および第２の復号化レジスタ識別子を受信すると共に、前記第１および第２の属性データのセットを受信するよう構成され、該第１および第２の復号化レジスタ識別子と該第１および第２の属性データのセットとに基づいてデータハザードを検出するよう構成された比

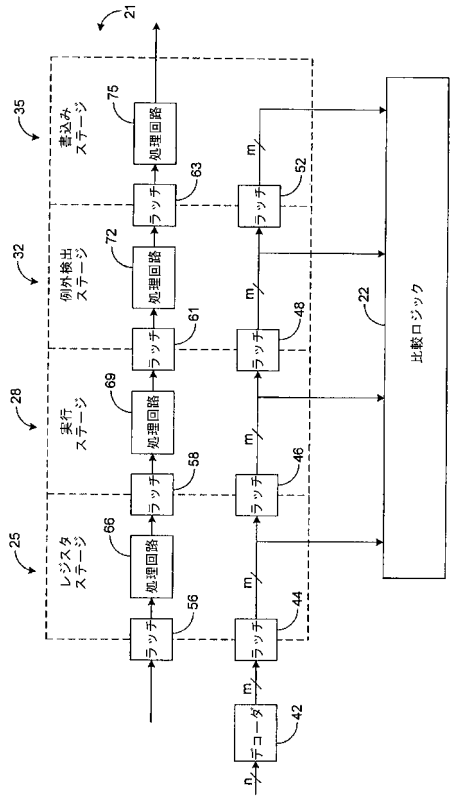
較ロジック（144、191）と、
を含むシステム。
【００８２】
（２）　前記第１の属性データのセットは前記第１の命令の述語ステータスを示し、前記第２の属性データのセットは前記第２の命令の述語ステータスを示す、前記（１）に記載のシステム（100）。
【００８３】
（３）　前記第１の属性データのセットは前記第１の命令が第１のタイプであるか否かを示し、前記第２の属性データのセットは前記第２の命令が第２のタイプであるか否かを示す、前記（１）に記載のシステム（100）。
【００８４】
（４）　前記第１のデコーダ（155、157、159、161）および前記比較ロジック（144、191）に結合され、前記第１の復号化レジスタ識別子および前記第１の属性データのセットを受信し、該第１の属性データのセットを該第１の復号化レジスタ識別子と組み合わせるよう構成された第１の属性インタフェース（183、185、187、189）と、
前記第２のデコーダ（155、157、159、161）および前記比較ロジック（144、191）に結合され、前記第２の復号化レジスタ識別子および前記第２の属性データのセットを受信し、該第２の属性データのセットを該第２の復号化レジスタ識別子と組み合わせるよう構成された第２の属性インタフェース（183、185、187、189）と、
をさらに含む前記（１）に記載のシステム（100）。
【００８５】
（５）　前記第１の復号化レジスタ識別子は複数のビットを含み、前記第１のデコーダは前記第１の符号化レジスタ識別子によりいずれのレジスタが識別されているかに基づいて該ビットのうちの１つをアサートし、該ビットのうちの他のビットをデアサートするよう構成されている前記（４）に記載のシステム。
【００８６】
（６）　コンピュータプログラム（107）の命令を処理する方法であって、
命令を処理システム（100）のパイプライン（132）に送信するステップと、
前記命令が前記パイプライン（132）の第１の部分によって処理されている間に、該命令に関連する符号化レジスタ識別子を復号化して第１の復号化レジスタ識別子にするステップと、
該命令が該パイプライン（132）の第２の部分によって処理されている間に、該符号化レジスタ識別子を復号化して第２の復号化レジスタ識別子にするステップと、
該命令に関連する第１の属性データのセットを送信するステップと、
該命令に関連する第２の属性データのセットを送信するステップと、
前記第１の復号化レジスタ識別子および前記第１の属性データのセットを、他の命令に関連する復号化レジスタ識別子および属性データと比較するステップと、
前記第２の復号化レジスタ識別子および前記第２の属性データのセットを、他の命令に関連する復号化レジスタ識別子および属性データと比較するステップと、
前記比較するステップに基づいてデータ従属性ハザードを検出するステップと、
を含む方法。
【００８７】
（７）　前記第１の属性データのセットを介して前記命令の述語ステータスを示すステップと、
前記第２の属性データのセットを介して前記命令の述語ステータスを示すステップと、
をさらに含む前記（６）に記載の方法。
【００８８】
（８）　前記第１の属性データのセットを介して、記命令が第１のタイプであるか否かを示すステップと、
前記第２の属性データのセットを介して前記命令が第２のタイプであるか否かを示すステ

ップと、
をさらに含む前記（６）に記載の方法。
【００８９】
（９） 前記第１の属性データのセットを前記第１の復号化レジスタ識別子と組み合わせるステップと、
前記第２の属性データのセットを前記第２の復号化レジスタ識別子と組み合わせるステップと、
をさらに含む前記（６）に記載の方法。
【００９０】
（１０） 前記第１の復号化レジスタ識別子は複数のビットを含み、
前記符号化レジスタ識別子に基づいて該ビットのうちの１つをアサートするステップと、
該符号化レジスタ識別子に基づいて該ビットのうちの他のビットをデアサートするステップと、
をさらに含む前記（９）に記載の方法。
【００９１】
【発明の効果】
本発明によれば、コンピュータプログラムの命令間のデータハザードを検出するシステムにおいて、不要なストールの発生を防止する効率的なシステムが提供される。
【図面の簡単な説明】
【図１】 従来技術による処理システムを示すブロック図である。
【図２】 図１に示すパイプラインのより詳細な図を示すブロック図である。
【図３】 図２に示すパイプラインの別の実施の形態を示すブロック図である。
【図４】 本発明の原理に従う処理システムを採用するコンピュータシステムを示すブロック図である。
【図５】 図４に示す処理システムを示すブロック図である。
【図６】 図５に示すパイプラインおよび比較ロジックのより詳細な図を示すブロック図である。
【図７】 図６に示すＡＮＤロジックのより詳細な図を示すブロック図である。
【図８】 図７に示すＡＮＤロジックの別の実施の形態を示すブロック図である。
【図９】 属性データを利用してデータハザードを検出する本発明による、パイプラインおよび比較ロジックの別の実施の形態を示すブロック図である。
【図１０】 図９に示すデコーダおよび属性インタフェースのより詳細な図を示すブロック図である。
【図１１】 図１０に示す１列のインタフェースロジックのより詳細な図を示すブロック図である。
【図１２】 図９に示すハザード検出回路のより詳細な図を示すブロック図である。
【図１３】 図１２に示す１列の比較ロジックのより詳細な図を示すブロック図である。
【図１４】 図１２に示すハザード検出回路における比較ロジックの各列の出力を組み合わせるために使用されるＯＲゲートを示すブロック図である。
【符号の説明】
100 処理システム
107 コンピュータプログラム
132 パイプライン
133 レジスタステージ
136 実行ステージ
139 例外検出ステージ
142 書込みステージ
144、191 比較ロジック
155、157、159、161 デコーダ
183、185、187、189 属性インタフェース

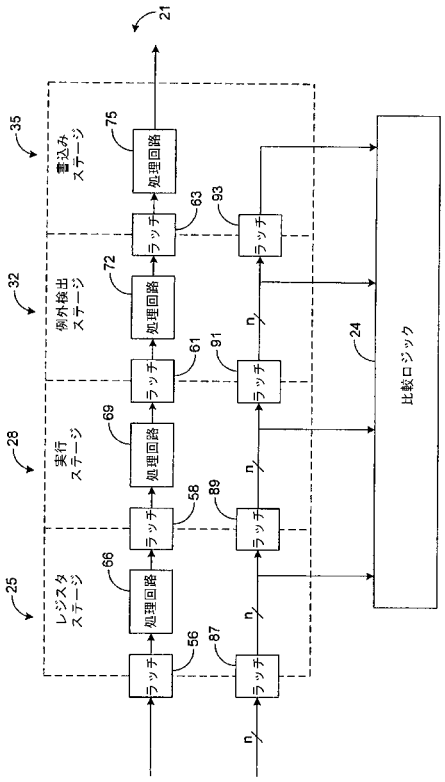
【図１】


15
35
書込みステージ
32
例外検出ステージ
28
実行ステージ
25
レジスタステージ
パイプライン
レジスタファイル
37
レジスタ
39
比較ロジック
24
22
21
命令分配ユニット
18

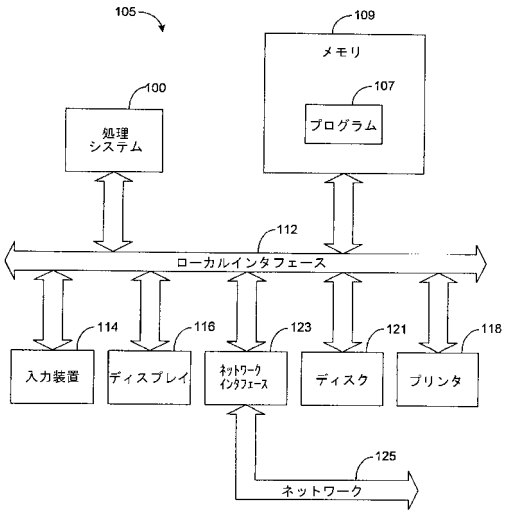
【図２】


21
35
書込みステージ
処理回路
75
ラッチ
63
52
32
例外検出ステージ
72
61
48
28
実行ステージ
69
58
46
25
レジスタステージ
66
56
44
m
デコーダ
42
n
22
比較ロジック

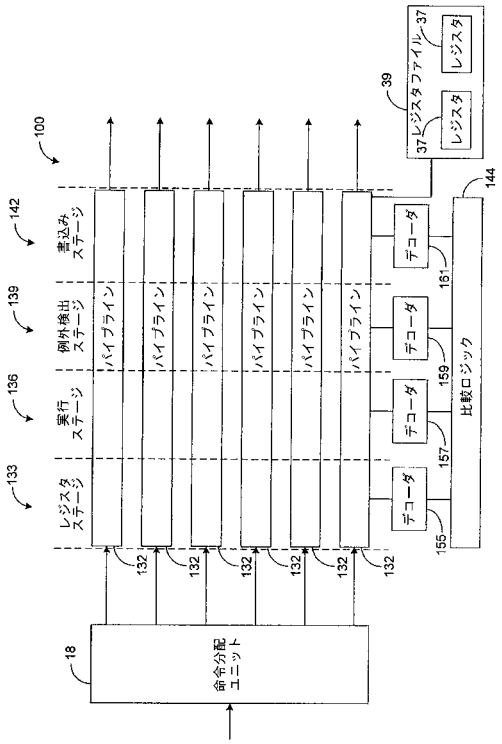
【図３】


21
35
書込みステージ
処理回路
75
ラッチ
63
93
32
例外検出ステージ
72
61
91
28
実行ステージ
69
58
89
25
レジスタステージ
66
56
87
n
24
比較ロジック

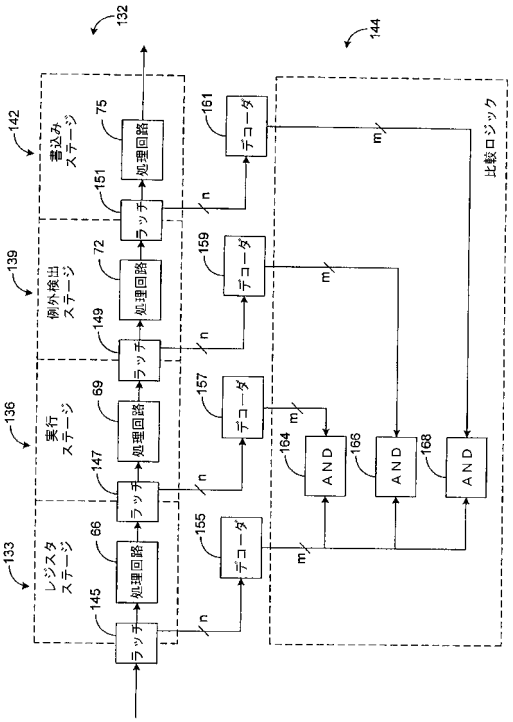
【図４】


105
109
メモリ
107
プログラム
100
処理システム
112
ローカルインタフェース
114
入力装置
116
ディスプレイ
123
ネットワークインタフェース
121
ディスク
118
プリンタ
125
ネットワーク

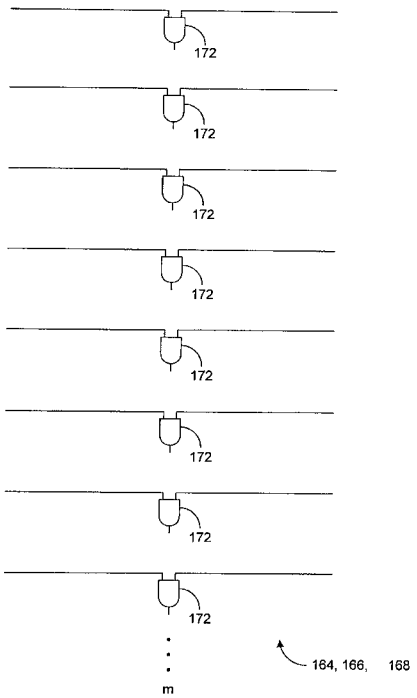
【図５】


100
142
書込みステージ
139
例外検出ステージ
136
実行ステージ
133
レジスタステージ
パイプライン
132
39
37
レジスタファイル
レジスタ
デコーダ
151
159
157
155
144
比較ロジック
18
命令分配ユニット

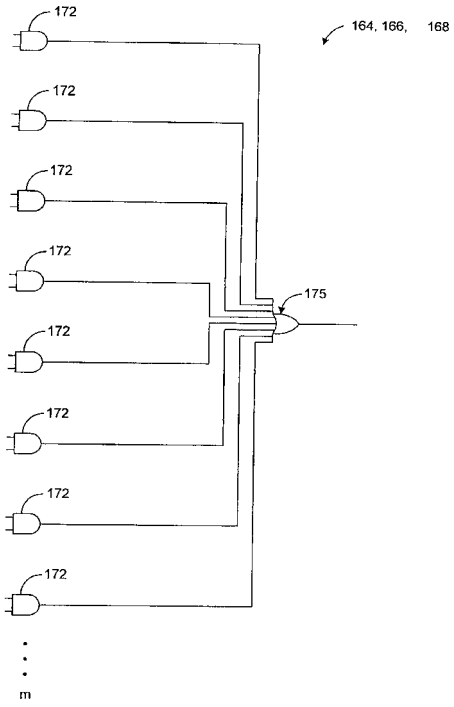
【図６】


132
144
142
書込みステージ
75
処理回路
151
ラッチ
139
例外検出ステージ
72
149
136
実行ステージ
69
147
133
レジスタステージ
66
145
デコーダ
161
159
157
155
n
m
AND
164
166
168
比較ロジック

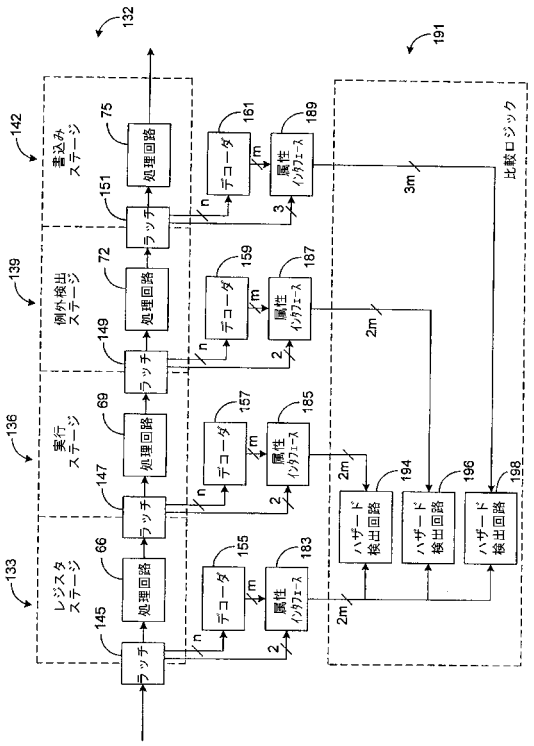
【図７】


172
164, 166, 168
m

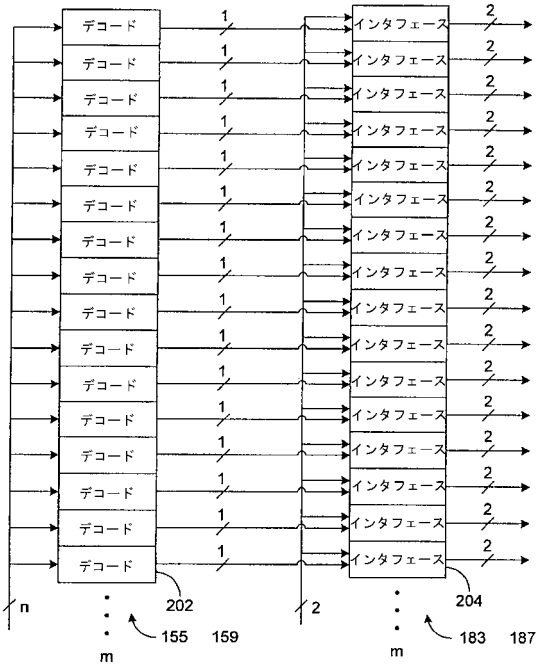
【図８】


172
164, 166, 168
175
m

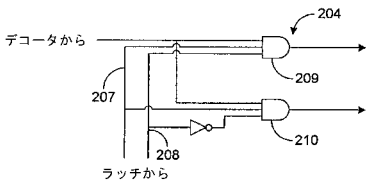
【図９】



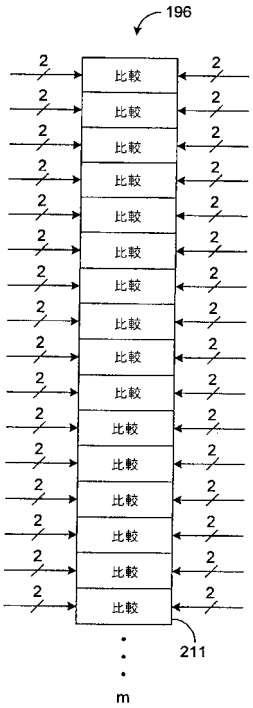
【図１０】



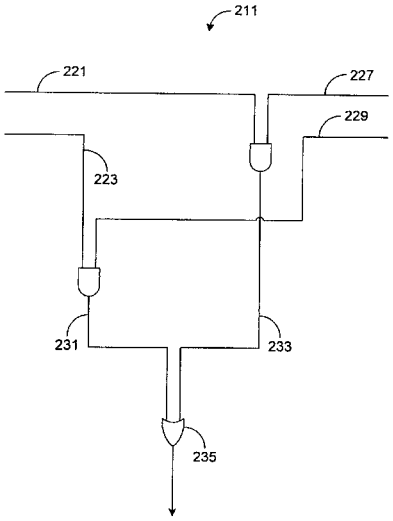
【図１１】



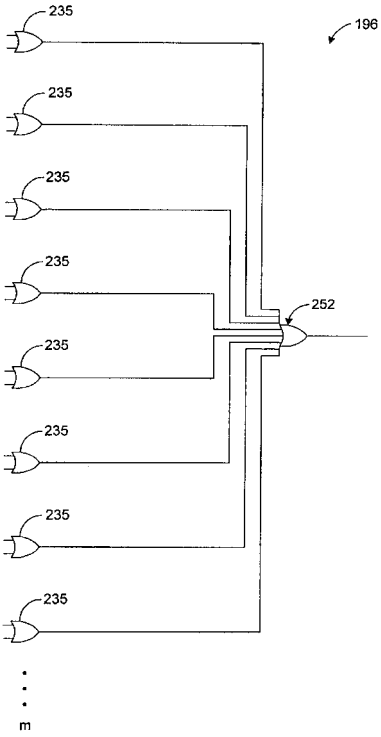
【図１２】



【図１３】


211
221
227
229
223
231
233
235

【図１４】


235
196
235
235
235
252
235
235
235
235
m

フロントページの続き

(72)発明者　ドナルド・チャールズ・ソルティス・ジュニア
アメリカ合衆国８０５２６コロラド州フォート・コリンズ、ローズゲート・コート　４４１４

審査官　後藤　彰

(56)参考文献　特開平１１－２８８３７３（ＪＰ，Ａ）
特開平９－２８２１６１（ＪＰ，Ａ）
特開平５－１０８３４８（ＪＰ，Ａ）
特開平４－３２８６３５（ＪＰ，Ａ）
特開平１－２３１１２６（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
G06F 9/38