(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第3833825号
(P3833825)
(45)発行日　平成18年10月18日(2006.10.18)　(24)登録日　平成18年7月28日(2006.7.28)

(51)Int.Cl.　F I
G06F 9/38 (2006.01)　G06F 9/38 310E

請求項の数 3　(全 13 頁)

(21)出願番号　特願平10-125635
(22)出願日　平成10年5月8日(1998.5.8)
(65)公開番号　特開平11-24927
(43)公開日　平成11年1月29日(1999.1.29)
審査請求日　平成17年5月9日(2005.5.9)
(31)優先権主張番号　08/853970
(32)優先日　平成9年5月9日(1997.5.9)
(33)優先権主張国　米国(US)

(73)特許権者　595034134
サン・マイクロシステムズ・インコーポレイテッド
Ｓｕｎ　Ｍｉｃｒｏｓｙｓｔｅｍｓ，　Ｉｎｃ．
アメリカ合衆国　カリフォルニア　９５０５４，　サンタ　クララ，　ネットワーク　サークル　４１５０
(74)代理人　100064621
弁理士　山川　政樹
(72)発明者　イヴァン・イー・サザーランド
アメリカ合衆国・９０４０５・カリフォルニア州・サンタ　モニカ・ワズワース　アヴェニュ・１２５

最終頁に続く

(54)【発明の名称】多重発行／複数逆流パイプライン・プロセッサ

(57)【特許請求の範囲】
【請求項１】
命令フェッチ・ユニットと、
レジスタ・ファイルと、
命令フェッチ・ユニットとレジスタ・ファイルとの間に接続され、複数のステージを含む単一のパイプラインとを含み、
前記単一のパイプライン中の各ステージが、
第１の命令を記憶する第１の命令レジスタ、
第２の命令を記憶する第２の命令レジスタ、および
結果を記憶する第１の結果レジスタを含み、
第１と第２の命令が第１の方向でステージからステージにロックステップさせて転送され、かつ結果が反対の方向でステージからステージに転送されるコンピュータ・システム・アーキテクチャ。
【請求項２】
命令フェッチ・ユニットと、
レジスタ・ファイルと、
命令フェッチ・ユニットとレジスタ・ファイルとの間に接続され、それぞれ複数のステージを含む少なくとも第１のパイプライン、第２のパイプライン、および第３のパイプラインとを含み、
第１のパイプラインの各ステージが第１の命令を記憶する第１の命令レジスタを含み、

第２のパイプラインの各ステージが第２の命令を記憶する第２の命令レジスタを含み、
第３のパイプラインの各ステージが結果を記憶する第１の結果レジスタを含み、
第１及び第２の命令は、第１及び第２の２つのパイプライン中へ交互に発行され、そして、その２つのパイプラインのそれぞれにおいて第１の方向でステージからステージに転送され、かつ結果が第３のパイプライン中で反対の方向でステージからステージに転送されるコンピュータ・システム・アーキテクチャ。
【請求項３】
命令フェッチ・ユニットと、
レジスタ・ファイルと、
命令フェッチ・ユニットとレジスタ・ファイルとの間に接続され、それぞれ複数のステージを含む少なくとも第１のパイプライン、第２のパイプライン、および第３のパイプラインとを含み、
第１のパイプラインの各ステージが第１の命令を記憶する第１の命令レジスタおよび第２の命令を記憶する第２の命令レジスタを含み、
第２のパイプラインの各ステージが第３の命令を記憶する第３の命令レジスタおよび第４の命令を記憶する第４の命令レジスタを含み、
第３のパイプラインの各ステージが結果を記憶する第１の結果レジスタを含み、
第１と第２の命令及び第３と第４の命令がそれぞれ第１のパイプラインおよび第２のパイプライン中で第１の方向でステージからステージにロックステップさせて転送され、かつ結果が第３のパイプライン中で反対の方向でステージからステージに転送されるコンピュータ・システム・アーキテクチャ。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】
本発明は、コンピュータ・アーキテクチャに関し、特に、逆流パイプライン・プロセッサと呼ばれるアーキテクチャの一種に関する。逆流パイプライン・プロセッサは、命令と結果がパイプライン中を反対の方向に流れ、それらが通過するときに相互作用するコンピュータ・システム・プロセッサである。さらに詳細には、本発明は、多重発行パイプラインまたは複数逆流パイプライン、またはその両方が使用されるそのようなアーキテクチャに関する。
【０００２】
【従来の技術】
逆流パイプライン・プロセッサでは、パイプラインは、それぞれ命令部分および結果部分を含むステージから形成される。命令はパイプライン中を一方向に流れ、結果は反対の方向に流れる。命令が上方に流れていると考えられる場合、任意の時刻において命令パイプラインはプログラム・リストのように見えるが、命令流れの実際の方向は重要ではない。
【０００３】
この類推を続けると、命令は、命令フェッチ・ユニットからパイプラインの底部に入る。各命令はパイプライン中のあるステージにおいて実行するが、異なるタイプの命令は異なるステージ中で実行する。命令は、その実行ステージに到達したときに実行の準備ができていない場合、準備ができるまでそこで待つ。パイプラインは、命令の物理的シーケンスを保持する。命令がいつパイプラインの適切なステージに到達したかに基づいて、命令を任意の時間的順序で実行させる。各命令は、分かっている場合、そのソースの名前または識別子および宛先オペランドおよびそれらの値を伴っている。最初に命令がパイプに入ったとき、値は未知であり、反対の方向に流れ過ぎた結果ストリームから最後にコピーされる。
【０００４】
結果は、パイプラインの上部のレジスタ・ファイルからか、または命令が実行するときに結果ストリームに入る。各結果は名前および値を有する。結果の名前は、ソースまたは命令の宛先レジスタのレジスタ名に対応し、オペランドの一意の識別子であることが重要で

ある。結果は、一致する名前を持っている場合にのみ物理的シーケンス中に留まる。異なる名前を有する結果は、結果パイプライン中で互いに追い越す。

【０００５】

最初に結果の値は未知であるが、実行が進行するにつれて、命令は、前の命令から来た結果のストリームからそれが必要とする値をコピーする。ソース値を命令中にコピーするプロセスを示すために「ガーナ（ｇａｒｎｅｒ）」なる語を使用する。命令は、それが遭遇した反対の方向に流れている結果からそれが必要とする値を「ガーナ」する。

【０００６】

また、結果と命令が遭遇したとき結果の値が変化する。命令がすでに結果中に名前付けされたレジスタの新しい値を計算してある場合、結果はこの最新の値をとる。命令がまだ結果中に名前付けされたレジスタの新しい値を計算しておらず、これからそれを計算しようとする場合、結果は無効になる。プロセッサが宛先値を結果中にコピーすることを示すために「レンラグ（ｒｅｎｒａｇ）」（ガーナを逆に綴ったもの）なる語を使用する。命令と結果が遭遇したときはいつでも、ガーナ・プロセスとレンラグ・プロセスは適宜にそれらの間で値を交換する。

【０００７】

逆流パイプライン・プロセッサの従来の研究のより詳細な議論は、リポートＳＭＬＩ　ＴＲ－９４－２５として本発明の譲受人であるＳｕｎ　Ｍｉｃｒｏｓｙｓｔｅｍｓ社から市販されている文献「逆流パイプラインプロセッサアーキテクチャ（Counterflow Pipeline Processor Architecture ）」に記載されている。これはまた、本発明の譲受人に譲渡された米国特許出願第５１８７８００号「非同期パイプライン化データプロセスシステム（Asynchronous Pipelined Data Processing System）」の主題である。

【０００８】

【発明が解決しようとする課題】

現代のマイクロプロセッサの能力が向上するにつれて、また単一の集積回路上に配置できる回路の量が増大するにつれて、逆流プロセッサ中の命令の実行が向上することが望ましい。本発明はそれを実現することを課題とする。

【０００９】

本発明の逆流パイプライン・プロセッサ・アーキテクチャは、命令フェッチ・ユニットおよびレジスタ・ファイルを含む。この２つは、所望の数のステージを有するパイプラインによって接続される。各ステージにおいて、命令中の所望のオペランドが実行される。

【００１０】

命令は、命令フェッチ・ユニットによってフェッチされ、パイプライン中に順にディスパッチされる。命令が上方に流れるとき、命令の実行からの結果または他のデータはパイプライン中を下方に流れる。各ステージにおいて、結果の名前および命令ソースの名前と宛先の名前とを比較する。一致した場合、結果の値を更新するか、または命令のソースまたは宛先を更新する。

【００１１】

【課題を解決するための手段】

一実施態様では、各ステージ中に２つまたはそれ以上の命令用のレジスタを含めることによって、改善された性能が得られる。そのような手法によれば、単一の命令レジスタのみを備えた場合よりも多くの命令がパイプライン中を移動できる。各命令レジスタは、演算フィールド、第１のオペランド・ソース・フィールド、第２のオペランド・ソース・フィールド、および宛先フィールドを含む。（所望の数のオペランド・ソースと宛先が備えられる）。命令がパイプライン中を移動するとき、ソース・フィールドおよび宛先フィールドの一部（レジスタ名）と結果フィールドの対応する部分（これもレジスタ名）とを比較する。レジスタ名が一致した場合、情報を命令と結果の間で転送することができる。

【００１２】

本発明の他の実施態様では、少なくとも３つのパイプラインが命令フェッチ・ユニットとレジスタ・ファイルとの間に備えられる。１つのパイプラインは第１の命令レジスタ用の

ステージを含み、第２のパイプラインは第２の命令レジスタ用のステージを含む。第３のパイプラインは結果レジスタを含む。そのような手法によれば、例えば浮動小数点命令が１つのパイプラインにディスパッチされ、例えば固定小数点命令が他のパイプラインにディスパッチされる。他の考えられる手法は、交互の命令を各パイプラインにディスパッチすることである。第１の実施態様の場合と同様に、命令がパイプライン中を一方向に移動し、結果が反対の方向に移動するとき、様々なレジスタの内容と様々なパイプラインの内容とを互いに比較して、オペランド・ソース・フィールドおよび結果フィールドを更新する。この実施態様では、追加のインタロック回路がパイプライン中の各ステージごとに備えられる。インタロック回路は、様々なパイプライン中を進行する命令が適切なシーケンス中に維持されることを保証する。

【００１３】

【発明の実施の形態】

図１に、従来技術の逆流パイプライン・プロセッサ（本明細書ではＣＦＰＰと呼ぶ）の代表的な構造を示す。本発明の好ましい実施形態は逆流処理の基本概念を使用するので、図１の説明によれば、本明細書の好ましい実施形態がよりよく理解できる。ＣＦＰＰ１は、命令が一方向に流れ、結果が反対の方向に流れる双方向パイプライン２を使用して、以下で論じる正確な動作を保証するいくつかのパイプライン規則に従って、部分的に実行される命令および結果を一定の形で移動させる。

【００１４】

ＣＦＰＰ１では、任意の長さのパイプライン２が底部の命令フェッチ・ユニット３を上部のレジスタ・ファイル４に接続する。命令は、パイプライン中の命令のシーケンスがコードのそのセクションのリストに似るように命令パイプライン２中を上方に流れるものとして示されている。この実施形態では命令が互いに追い越すことが禁止されているので前の命令は後の命令の上方にあり、この状態が維持される。命令は、パイプ２中をできるだけ遠くかつ迅速に上方に移動し、図１ですぐ上のパイプライン・ステージ（ステージ０、ステージ１、．．．ステージｎで示される）がまだ新しい命令を受けていないとき、または命令がそれを実行するために備えられている最後のパイプライン・ステージに到達したときのみ停止する。停止した命令の上方には、任意のサイズのギャップがパイプライン中に形成される。すべてのステージが命令で「いっぱい」になる必要はない。制御回路（図示せず）は、命令が適切な時刻において移動し、かつ他の命令を上書きしないこと、および結果および命令の動きが調和することを保証する。

【００１５】

ＣＦＰＰアーキテクチャによれば、様々なステージに様々な処理動作を実施させるか、または様々なステージを様々な処理動作に対して最適化することができる。例えば、１つのステージが整数演算命令を実行し、他の整数演算命令が浮動小数点命令を実行することができる。１つのステージが加算を実行し、他のステージが乗算を実行することもできる。乗算器が命令パイプライン中の加算器の下にある場合、内積を形成するために使用される「乗算－加算」シーケンスは、結果パイプラインによって与えられた転送を利用する。言い換えれば、乗算結果は、加算器に迅速に転送される。

【００１６】

命令パイプラインの他の様態は、ステージが部分的実行を実施し、命令がパイプラインを上方に進行するために十分に実行されることを確認するために必要な場合に停止する。命令をフェッチするステージまたは複数のステージはまた、部分的実行を達成するとも言われる。すなわち、ステージは、プログラム・カウンタ値を実行すべき命令を記述するビットに変換する。

【００１７】

図１に示されるようなシステムの性能を改善するためには、追加の並列度を使用すればパイプラインのスループットを改善することができる。２つまたはそれ以上の命令を各ステージ中にパックすることによって、改善された性能が得られる。パイプライン中の各ステージが多数の命令を担持するので、このタイプのアーキテクチャを多重発行システムと呼

ぶ。このタイプの二重発行設計では、命令は、パイプライン中を対になって上方に移動する。並列度を増加する三重またはそれ以上の手法も可能である。これは、複数（２つまたはそれ以上）の命令に対する各移動動作のオーバヘッドを無くすことができる。２つの命令を一時に発行し、パイプ中をロックステップさせて（つめて）上方に移動させることにより、ステップごとに２回動作が起こる。後で示すように、追加の名前比較およびガーナ・パスも設けられている。

【００１８】

図２に、本発明の一実施形態の多重発行逆流パイプライン・プロセッサの代表的な構造を示す。ＣＦＰＰ１０では、パイプライン２０が底部の命令フェッチ・ユニット３０を上部のレジスタ・ファイル４０に接続する。ただし、この場合、パイプライン２０中の各ステージは２つの命令を含む。例えば、ステージ０では、命令は命令Ａおよび命令Ｂで示される。上述のように、命令は、命令パイプライン２０中を上方に流れるものとして示されており、前の命令は後の命令の上方にある。命令Ｃおよび命令Ｄは命令Ａおよび命令Ｂの後で発行され、命令Ｍおよび命令Ｎはさらに後で発行される。命令対は、パイプ２０中をできるだけ遠くかつ迅速に上方に移動し、すぐ上のパイプライン・ステージがまだ新しい命令対を受けることができないとき、または対になった命令の一方がそれを実行するために備えられた最後のパイプライン・ステージに到達したときのみ停止する。

【００１９】

図３に、図２に示されるパイプライン２０中の各命令および結果ステージｎのデータ構造、したがってまたラッチ構造を示す。図３に示される実施形態は二重命令パイプライン用であるが、所望の任意の数、すなわち３つ以上の命令を対応する形で実施できることを理解されたい。図３には、各命令５０、６０がどのようにそのソース・オペランド５２、５５、６２、６５およびその宛先５８、６８用のバインド（フィールド）を備えるか、ならびに結果７０がどのようにその結果値７４、７６用のバインド（フィールド）を備えるか示されている。命令用ならびに結果用の各バインドは、データ値８０を、レジスタ名８４に関連付け、有効ビット８５がその関連が有効であるかどうかを示す。

【００２０】

図３に示されるパイプライン・ステージは、各命令を有する３つのバインド、例えば、２つのソース５２、５５および命令５０に関連する１つの宛先５８、および２つのソース６２、６５および命令６０に関連する１つの宛先６８を示す。もちろん、他の数のソースおよび宛先も使用できる。命令５０または６０が実行されたとき、新しいデータが宛先バインド５８、６８のデータ値８０中に入れられ、次いでビット８５を使用して、有効とマーク付けされる。命令５０、６０がパイプラインの上部に到達したとき、宛先バインド５８、６８中に記憶されたデータ値は、レジスタ・ファイル４０（図２参照）中の対応する位置に書き込まれる。この最後の記録がレジスタ・ファイル４０中に作成されるまで、命令５０、６０は不確実であると考えられ、トラップまたはブランチによって必要とされた場合に必要に応じて取り消される。

【００２１】

命令５０、６０が実行されるときはいつでも、出力は３つの形で使用される。第一に、出力が命令の宛先バインド５８、６８中に入れられ、最後に上述のようにレジスタ・ファイル４０中に書き込む。第二に、各宛先バインド５８、６８が後続の命令によって観測されるように下方に流れる結果パイプライン中に挿入される。第三に、例えば下位の命令が命令グループ内の上位の命令からの結果を必要とする場合、結果がそのグループ内で使用される。これについては以下でさらに論じる。図３に示すように、この実施形態では、結果パイプラインの各ステージは２つのバインド７４、７６を収容する。

【００２２】

そのレジスタ名８４が結果バインド７４、７６のレジスタ名８４に一致するソース・バインド５２、５５、６２、６５を必要とするパイプライン中の後の命令は、その値を命令パイプ中のバインド中にコピーし、書き込むことによってその値をガーナする。ソース・レジスタ５２、５５、６２、６５の名前、宛先レジスタ５８、６８の名前、および結果レジ

スタ７４、７６の名前の間の比較動作は、図３に楕円形で示されるコンパレータ９０のグループによって実施される。そのようなコンパレータの回路設計は周知である。したがって、結果パイプラインは、従来のプロセッサ設計において「バイパス」または「転送」と呼ばれる機能を実施し、すべてのステージに対して均一である形でその機能を実施する。

【００２３】

比較を実施することによって、下方に流れる結果バインドが後続の命令によって修正される。コンパレータを使用して、パイプライン中の各ステージ５０および６０は、命令バインドと結果バインドとの一致、すなわち、命令バインド中のレジスタ名８４と結果バインド中のレジスタ名８４とが一致する場合を検出する（図３参照）。すでに実行され、かつ結果バインド７４、７６に一致する宛先バインド５８、６８を有する命令５０、６０は、宛先バインド５８、６８からの値を結果バインド７４、７６中にコピーしなければならない。このようにして、後の命令は、任意のレジスタに対して最も新しいバインドをガーナする。

【００２４】

命令５０または６０がまだ実行すべき場合、異なる状況が生じる。この場合、バインドがさらに後の命令に対して有効でなくなるので、命令中の宛先バインド５８、６８に一致する結果バインド７４、７６は結果パイプラインから削除される（本明細書では「キル」されると呼ぶ）。したがって、特定の結果バインド７０は、一般に、命令５０、６０の短いスパン、すなわち値を計算した命令の後およびその値を上書きする次の命令までのセクションのみを通過させる。これらの結果修正規則は、逆流パイプライン中の命令５０、６０に遭遇した結果７０がその命令５０、６０に対して正確であるバインドを保持することを保証する。レジスタ用の異なる複数のバインドが同時に結果パイプラインの異なる部分中を通過している。そのような多数の結果バインドの使用は、従来のプロセッサ設計における「レジスタ名前変更」と同じ役目を果たす。

【００２５】

レジスタ・ファイル４０（図２参照）はまた、命令に対するオペランドのソースである。このため、レジスタ・ファイル４０は、結果パイプ中に挿入されたバインドの主要なソースである。好ましい実施形態では、フェッチし、パイプ中に送るべきレジスタ値を決定するのに適している多数の方針のうち、このシステムは、パイプ中の命令のソースに一致することが分かっているバインドを結果パイプ中に送ることを選択する。これらのレジスタ値は、関連する命令によってガーナされ、したがって命令の実行を可能にする。この方針の一実施形態では、命令復号ステージがソース・レジスタ・アドレスをレジスタ・ファイルに送り、レジスタ・ファイルがレジスタ値をフェッチし、対応するバインドを結果パイプ中に送る。

【００２６】

正確な動作を保証するために、ＣＦＰＰは、下方に流れる各結果中のバインドと上方に流れる各命令中のバインドとの一致を検出しなければならない。各結果は、比較が行われたあるパイプライン・ステージ中の後続のすべての命令に遭遇しなければならない。適切な制御回路を使用することによって、隣接するパイプライン・ステージが、一致の検出を妨げる命令と結果を「同時に」交換することがなくなる。ステージ間の境界は、命令を上方に通過させるか、または結果を下方に通過させるが、両方を通過させることはできない。ＣＦＰＰの非同期実施形態では、ステージの各対間のアービタを使用して、この通信プロトコルを実施する。

【００２７】

パイプライン中の各ステージは、いくつかの規則に従って動作する。命令に関して、

１．命令は、命令パイプライン中でプログラム順でなければならない。

２．命令に対するすべてのソース・バインドが有効であり、かつ命令が適切な計算論理を含むステージ中に保持される場合、命令は実行する。命令が実行を完了したとき、その宛先バインドが有効とマーク付けされ、値が挿入される。

３．命令が実行を完了したとき、それぞれその宛先バインドの１つまたは複数のコピーが

結果パイプライン中に挿入される。
４．実行されない命令は、それを実行することができるパイプラインの最後のステージを追い越すことはできない。特に、命令は、それが実行されるまでレジスタ・ファイル中に入ることができない。
命令と結果が同じパイプライン・ステージ中に存在し、かつ一致するバインドを有する場合に適用する一致規則に関して、
１．有効な結果バインドが無効なソース・バインドに一致する場合、結果値をソース値にコピーし、ソースを有効とマーク付けする。
２．無効な宛先バインドが有効な結果バインドに一致する場合、結果バインドを無効とマーク付けする。
３．有効な宛先バインドが結果バインドに一致する場合、宛先値を結果値中にコピーし、結果を有効とマーク付けする。
４．グループ内で、宛先およびソースを適切に処理する必要がある。
【００２８】
多重発行システムでは、各命令は、実行の準備ができるまでその実行ステージのところで待たなければならない。１つの問題は、多重発行命令中の別々の動作が異なるステージにおいて実行する必要があることである。複合命令は、複合命令の任意の部分を計算するためにその計算に必要とされるソースをガーナするまで第１のステージにおいて停止する。残念ながら、停止をもたらした動作は、命令シーケンス中の前の方で、パイプライン中の後のステージまで計算されない複合命令のある他の部分からの結果に依存する。複合命令中の別々の動作がパイプライン中のステージの順序に一致しない順序で現れた場合、そのような停止はデッドロックをもたらす。そのようなデッドロックは、以下で説明するように、予定された形でグループ内の命令を実行する機構を備えることによって回避される。
【００２９】
本発明のいくつかの実施形態では、結果パスを同様に広くすることができる。言い換えれば、２つまたはそれ以上の結果レジスタが使用できる。しかしながら、実施した実験的テストでは、実際の命令の統計から、命令はそれぞれ１よりもわずかに大きいソース値のみを使用することが分かる。第二に、レジスタ・キャッシュは、小さいものでも、レジスタ・ファイルへのアクセスを最小限に抑え、したがって結果パイプライン・トラフィックを少なくする。第三に、結果は、結果パイプ中で短い距離、すなわち結果が生成された場所から結果がキルされた場所までしか持続しない。同じ宛先レジスタを有する２つの近い命令は、一緒にただ１つの結果パイプ要素をもたらす。したがって、パイプラインをより長くすることには、結果パイプの帯域幅を広くする効果がある。第四に、一対の命令は、共同で結果を生成し、消費する。命令を対にすることによって、結果パイプへのステージ当たりのアクセスを増加させる。両方の命令が共通のソース・レジスタを必要とする場合、それが結果パイプ中にただ１回現れる必要がある。２つの命令が共通の宛先を有する場合、それらは一緒にただ１つの結果をもたらす。命令のグループが名前を共用する場合、適切な実行を保証するために特に注意しなければならない。
【００３０】
多重発行パイプラインを処理するために追加の演算装置が備えられる。例えば、命令パイプの各側に１つずつ、２つのＡＬＵを使用し、異なる複数のステージにおいて対のＡＬＵを繰り返す。好ましい実施形態では、命令パイプラインを半分に分けてその２つの半分が単一のＡＬＵ（図示せず）を共用し、対になった動作が異なっているという統計的な可能性を利用する。そのような実施形態では、両方の動作がそれらのすべてのソース値をガーナし、かつ両方が同じＡＬＵを必要とする場合、それは順にまず一方の命令によって使用され、次いで他方の命令によって使用される。好ましい実施形態では、単一のＡＬＵが任意の１つのステージ中で使用される。そのような単一のＡＬＵの競合は２つの理由で最小になる。第一に、統計的に両方の命令が同じ種類のＡＬＵを必要とする可能性は低い。第二に、両方が同じ種類のＡＬＵを必要とする場合でも、対の一方の命令は、一般に他方の命令の前にその資源をガーナする。まだ準備ができていない命令は、まだＡＬＵの競合者

ではない。他の実施形態では、複数のＡＬＵが備えられるが、命令対によって共用されるべき別々のステージ中に配置される。そのような手法では、部分的に完了した対が完了のために他のステージに前進することができる。他の実施形態では、ステージパイプごとに多重命令を処理するより多くの演算装置が備えられる。そのような実施形態では、命令パイプの各側に１つずつ、２つのＡＬＵが備えられる。

【００３１】

命令対は、単一の命令よりも所与のステージ中で処理を必要とする可能性が高い。対によっては、２倍の処理時間を必要とする。したがって、単一発行パイプラインに比較して、二重発行パイプラインはより広範な処理時間を有する。必要な場合、高速バッファステージを可変遅延ステージの作業負荷の前ならびに後に配置すれば、作業負荷を平滑化できる。

【００３２】

上述の多重発行手法は、プログラマの連続的実行モデルを保持する。このモデルは、各ステージが各対を適切に処理する場合、すなわち結果の処理に対して必ずしも連続的な発行順序でではなく各対を処理する場合、二重命令パイプライン中に保存される。対の前の命令が結果をキルした場合、その結果は、対の後の命令に見えてはならない。パイプラインのフレキシビリティは、設計に大きい利点をもたらす。実行順序は対の中でも保持されない。対のどちらかがまず実行される。命令の対が結果をガーナし、キルするためにどのように一緒に動作するかに関してのみ注意が必要である。好ましい実施形態では、これは、グループ内の命令の宛先を比較する余分のコンパレータを備えることによって達成される。例えば、図３のコンパレータ８８を参照されたい。追加の帯域幅が望まれる場合、レジスタ・ファイル・キャッシュが備えられる。そのようなキャッシュは、本発明の譲受人に譲渡された米国特許出願第０８／７５８８０２号「コンピュータ・プロセッサへレジスタ値を与えるレジスタ・キャッシュ（A Register Cache for Providing Resister Values to a Computer Processor）」に記載されている。

【００３３】

命令の並列発行を実施する他の方法は、上述の発行手法の他に、複数の別々のパイプラインを使用することである。図４にこの実施形態を示す。図示のように、命令用の２つのパイプライン１００、１２０があり、結果用の１つのパイプライン１３０がある。もちろん、追加の命令パイプラインおよび結果パイプラインが備えられる。命令パイプライン１００は、上述のように任意の数の別々のステージを含む。他方の命令パイプライン１２０も、一般に７６パイプライン１００と同じ数の別々のステージから構成される別々のステージを含む。底部の命令フェッチ・ユニット１４０は、パイプライン１００とパイプライン１２０の両方に接続され、それぞれに命令を与える。上部のレジスタ・ファイル１５０は、パイプライン１００とパイプライン１２０の両方に接続され、それらから結果を受ける。上述のように、命令は、命令パイプライン１００、１２０中を上方に流れるものとして示されている。単一のパイプライン１００または１２０中の前の命令は、後の命令の上方にある。命令は、それぞれパイプ１００、１２０中をできるだけ遠くかつ迅速に上方に移動し、すぐ上のパイプライン・ステージがまだ新しい命令を受けることができないとき、または命令がそれを実行するために備えられた最後のパイプライン・ステージに到達したときのみ停止する。また、上述のように、停止した命令の上方には、任意のサイズのギャップがパイプライン中に生じる。パイプラインの命令が異なるレジスタ組を使用するのでパイプラインは独立している。パイプラインは、底部において、命令を適切なパイプライン中に送る命令発行機構１４０によって結合される。好ましい実施形態では、一方のそのようなパイプは主として浮動小数点演算用に使用され、他方のパイプは主として固定小数点演算用に使用される。

【００３４】

図４に示されるアーキテクチャでは、命令は、２つのパイプライン中に交互に発行される。命令が同じ命令シーケンスから来た場合、一方のパイプライン中の命令が他方のパイプライン中の命令を追い越すのを防ぐ追加の機構を備えて、それらの相互のシーケンスを保

持することが望ましい。２つのパイプ間の命令の順序は、インタロック機構１６０によって維持される。この回路は、一方のパイプライン中の後で発行された命令が他方のパイプライン中の前に発行された命令を追い越すのを防ぐ。したがって、命令は、それらが命令フェッチ１４０によって発行されたのと同じ順序でレジスタ・ファイル１５０中に入れられる。所要のインタロック機構を実施するのに適した回路は、本発明の譲受人に譲渡された１９９６年４月２３日出願の米国特許出願第０８／６３６２６０号「インターロックされたＦＩＦＯ制御回路（Interlocked FIFO Control Circuits ）」に記載されている。

【００３５】

また、２つのパイプラインが異なる機能を有する場合、それらの間で連続的な動作を維持することができる。例えば、プログラムが浮動小数点演算のグループと固定小数点演算のグループとを交互に組み合わせると仮定する。固定小数点演算は、浮動小数点演算用の索引値を与える。複数の固定小数点演算の結果は、単一のそのような索引値である。そのような命令の交互に組み合わせた組を２つのパイプラインに分割できる。

【００３６】

浮動小数点パイプ中では、固定小数点演算の各グループが「要約」演算に置き換えられる。そのような要約演算は、実際に「レジスタＸの値がこの点において命令ストリーム中に現れる」という。同様に、固定小数点パイプ中では、浮動小数点演算の各グループが、実際に「この点において以下のレジスタが計算される」という「要約」演算に置き換えられる。

【００３７】

パイプ間の通信手段は、要約演算を使用して、結果を他方の結果パイプ中に導入する。要約演算はまた、結果パス中の古くなった結果をキルする役目をする。要約演算は結果をキルするので、宛先値を生成する演算が単一のパイプライン中の結果ストリームを分割する場合と同様に、要約演算は結果パスをその適切な部分に分割する。

【００３８】

図４に示される実施形態の命令／結果バインド機構は、図３に示されるシステムと同じである。命令が機能的に別々のパイプライン中にあることは、上述のように、単一の多重発行パイプラインに比較して動作を変更しない。

【００３９】

上述の多重パイプライン実施形態では、両方のパイプライン中の命令は、単一結果パイプラインからの値と相互作用する。ガーナとレンラグの両方の動作が行われる。適切なアービトレーション回路および適切な名前比較回路をそれぞれ命令パイプラインと結果パイプラインの間に複製しなければならない。この手法は、単位時間当たり発行する命令が、各結果が２つのパイプラインのどちらか一方に入る場合の２倍であるので大きい利点をもたらす。

【００４０】

いくつかの状況では、そのようなシステムは、結果パイプラインに非常に重い負担をかけることがある。それらの状況では、複数の結果パイプラインを同様に備えることができる。このアーキテクチャによれば、結果が異なる名前を有することを条件として結果が互いに追い越すことができる。したがって、異なる種類の名前を異なる結果パイプ中に入れることが有用である。例えば、２つの結果パイプを有するシステムでは、偶数の名前を有する結果は一方のパイプ中に入り、奇数の名前を有する結果は他方のパイプ中に入る。あるいは、結果名は、ある他の条件によっていくつかのグループに分割され、例えばアルファベットの後のほうの名前は一方のパイプ中に入り、アルファベットの前のほうの名前は他方のパイプ中に入る。

【００４１】

所要の制御回路の数および比較回路の数は、一方向に流れるパイプラインの数と反対の方向に流れるパイプラインの数との積である。グループ間比較のために追加のコンパレータ８８が必要である。例えば、上方に流れる３つの命令パイプおよび下方に流れる２つの結果パイプを有するシステムは、１ステージ当たり６つのアービタおよび６つの比較回路を

必要とする。非同期デバイス中でのこの種類の信頼できる相互作用にはアービトレーションが必要である。各パイプの各ステージは、各逆流パイプラインごとに１つのアービタを必要とする。同様に、各パイプラインの各ステージは、その名前を逆流パイプライン中の名前に比較する名前比較回路を必要とする。したがって、回路の数は、２つの方向に流れるパイプの数と、適切なグループ間協働を保証するために追加された追加の回路との積に関連する。

【００４２】

上述のように、所与のグループ内での命令の実行を調整するために必要とされる追加のコンパレータ８８がある。必要とされる特定の調整は、命令パイプラインおよび結果パイプライン用に選択された特定の実施形態によって異なる。

【００４３】

命令パイプラインに関して、３つの可能な構成がある。第１の実施形態では、命令の各対はパイプライン中を「ロックステップ」の形で移動する。これは、図３に示される実施形態であり、図３では、実行中の上位の命令からの結果が対（またはより大きいグループ）内の適切な下位の命令に送られて、その下位の命令が実行できることを保証するために追加のコンパレータ８８が必要である。

【００４４】

命令パイプライン中の命令の他の可能な関係は、それらが別々の組を形成することである。この状況では、命令は、命令の各対（またはより大きいグループ）が互いに異なる文字のみの命令を含むようにコンパイルされる。例えば、一対の命令の場合、１つの命令が浮動小数点命令になり、他方の命令が固定小数点命令になる。そのような場合、結果が対の一方の命令から他方の命令に送られないので追加のコンパレータ８８は不要である。

【００４５】

命令パイプライン用の命令の他の可能な構成は、命令がそれらの実行順序を保存するような形でインタロックされることである。この構成では、各対の１つの命令は、より最近命令フェッチ・ユニットによって発行された上位命令、下位命令と考えられる。ただし、このシステムは、次の対の上位命令が実行する前に各対の下位命令が必ず実行するような形で動作するようになされる。言い換えれば、次のグループの第１の命令が実行する前にグループ内のすべての命令が実行することを保証するインタロック機構が必要である。

【００４６】

パイプラインの各ステージ中に多数の結果が存在する場合、結果パイプライン用の同様の可能な構成がある。もちろん、最も簡単な場合は、単一の結果のみがパイプラインの各ステージのところに存在し、その結果がパイプラインのすべての命令によって共用されることである。他の場合は、命令ステージの場合と同じである。言い換えれば、結果は、結果パイプラインに沿ってロックステップの形または互いに素な集合の形で移動する。互いに素な集合は、奇数または偶数のステージからの結果、またはアルファベットのある部分中の名前対他の部分中の名前に対する結果などの結果から構成される。結果パイプラインの最後の場合は、結果がパイプライン中に存在するが、必ず同じ名前または識別が同じ名前または識別を有する他の命令を通さないような形で調整されることである。

【００４７】

上述の本発明は、多数の態様を有する。第１に、複数のパイプラインが所与の方向に流れることができる。例えば、３つのパイプが一方向に流れ、２つのパイプが反対の方向に流れることができる。パイプライン間の相互作用の規則は、単一のパイプが各方向に流れる場合とまったく同様に保持される。言い換えれば、一方向に流れるデータ要素は、反対の方向に流れるすべてのデータ要素と相互作用する機会を有する。

【００４８】

第２に、そのような構造を使用してパイプライン・アキーテクチャを実施することによって、多数の命令パイプが単位時間当たりより多くの命令を発行することができる。第３に、多数の結果パイプを使用することによって、性能がさらに改善される。識別可能な名前を有する結果が別々の結果パイプ中に入れられた場合、結果の順序を保存するインタロッ

クが省略できる。

【００４９】

本発明の他の実施形態では、複数の多重発行パイプラインが使用される。そのような実施形態では、システムの編成は、図２および図４に示されるものの組合せである。言い換えれば、図４の複数の各パイプラインは、図２に示すようにそれ自体多重発行パイプラインである。制御機構は、２つの実施形態それぞれについて個々に不変である。

【００５０】

本明細書に記載の本発明は、同期プロセッサまたは非同期プロセッサ内で実施される。逆流パイプライン・プロセッサの同期実施形態は、隣接するステージの状態に対して組合せ関数によって与えられたステージの局所制御を有する。制御は、固定の方針を使用して、いずれか一方が可能な場合、命令パケットを通すか、または結果パケットを通すかを決定することができる。一実施形態では、命令は優先的に通される。

【００５１】

パイプラインの非同期進行が実施される本発明の実施形態では、アービトレーションと呼ばれる技法が使用される。パイプライン各ステージに関連するアービタは、命令（または結果）がいつ進行の準備ができるかを決定する。アービタ回路は、前進を許可する前に３つの条件が存在するかどうかを確認する。３つの条件とは、（１）例えば、命令が実行されている場合、実行が必ず完了するように項目が進行の準備ができていなければならない、（２）次のステージにスペースが存在しなければならない、（３）命令と結果が交換できない、すなわちそれらが隣接するステージ間で交換できない。これらの条件では、ステージおよびその隣接するステージの状態が局所的に分かっているだけでよい。

【００５２】

以上、逆流プロセッサの性能を改善するコンピュータ・システムについて説明した。本発明の範囲から逸脱することなくこのシステムに修正または変更を加えることができることを理解されたい。本発明の範囲は、首記の請求の範囲に記載されている。

【図面の簡単な説明】

【図１】従来技術の逆流パイプライン・プロセッサの代表的な構造を示す図である。

【図２】多重発行逆流パイプライン・プロセッサの好ましい実施形態を示す図である。

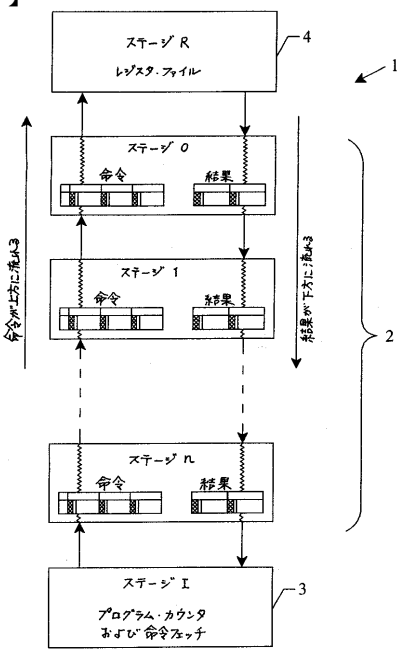
【図３】図２に示されるパイプライン中の各命令および結果ステージのデータ（およびラッチ）構造を示す図である。

【図４】複数の別々のパイプラインを使用して多数の命令を発行する逆流パイプライン・プロセッサの他の実施形態を示す図である。

【符号の説明】

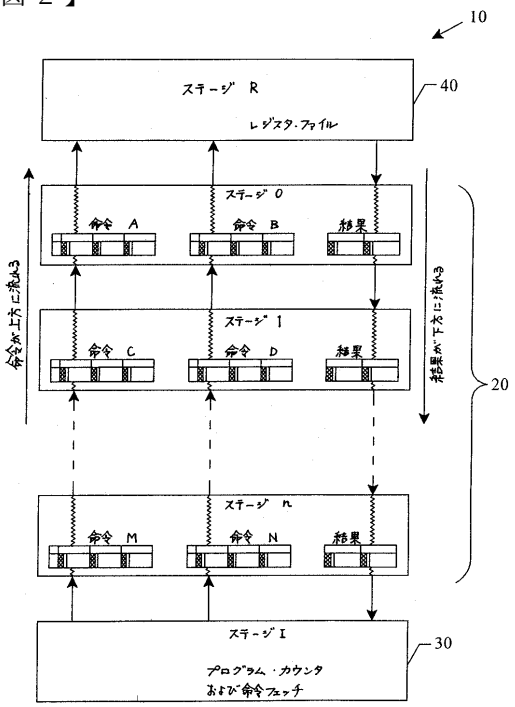
１０　逆流パイプライン・プロセッサ（ＣＦＰＰ）
２０　パイプライン
３０　命令フェッチ・ユニット
４０　レジスタ・ファイル
５０　命令
５２、５５　ソース・オペランド
５８、６０　宛先
６０　命令
６２、６５　ソース・オペランド
６８　宛先
７０　結果
７４、７６　結果値
８０　データ値
８４　レジスタ名
８５　有効ビット

【図１】

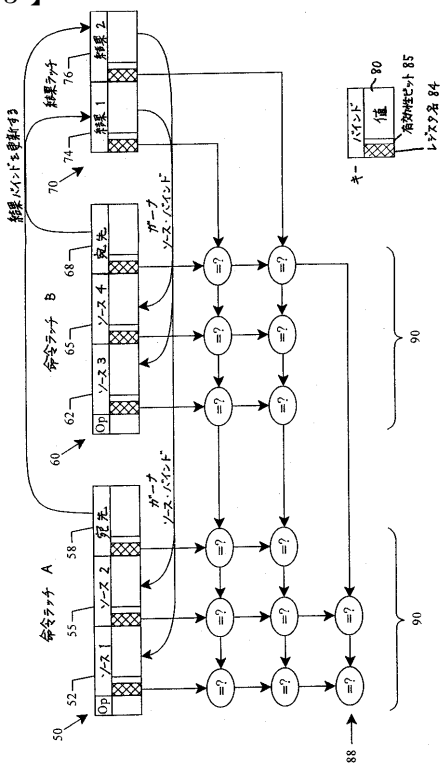


（従来技術）

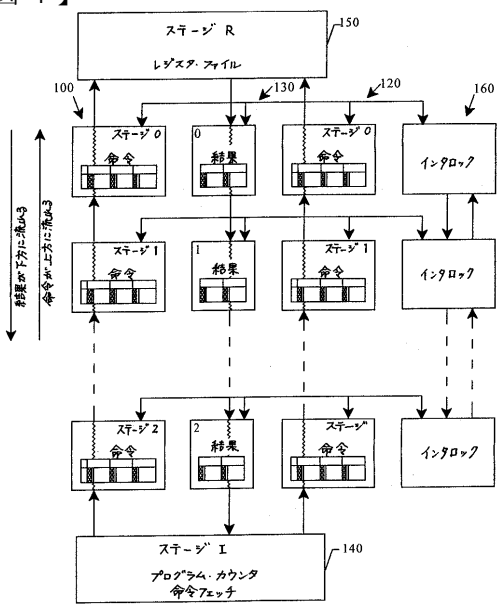
【図２】



【図３】



【図４】



フロントページの続き

審査官　後藤　彰

(56)参考文献　特開平７－２７１５７９（ＪＰ，Ａ）
特開平５－５３８０３（ＪＰ，Ａ）
特開平８－３６４９２（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
G06F 9/38