(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第3547482号
(P3547482)
(45)発行日　平成16年7月28日(2004.7.28)　(24)登録日　平成16年4月23日(2004.4.23)

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

請求項の数 2　(全 21 頁)

(21)出願番号　特願平6-102275
(22)出願日　平成6年4月15日(1994.4.15)
(65)公開番号　特開平7-281896
(43)公開日　平成7年10月27日(1995.10.27)
審査請求日　平成13年4月11日(2001.4.11)

(73)特許権者　000005108
株式会社日立製作所
東京都千代田区神田駿河台四丁目６番地
(74)代理人　100099298
弁理士　伊藤　修
(72)発明者　伊藤　元久
東京都国分寺市東恋ケ窪１丁目２８０番地
株式会社日立製作所　中央研究所内
(72)発明者　釜田　栄樹
東京都国分寺市東恋ケ窪１丁目２８０番地
株式会社日立製作所　中央研究所内

審査官　後藤　彰

最終頁に続く

(54)【発明の名称】情報処理装置

(57)【特許請求の範囲】
【請求項１】
演算の種類毎に分類されたｎフィールド（ｎは2以上）の演算フィールドを有する長語命令からなる命令流ｍ本（ｍは2以上）の処理を行なう情報処理装置であって、
ｍ個の命令デコード部と、該命令デコード部から発行された演算命令を受ける前記演算の種類毎に設けられたｎ個の命令スケジュール部と、該ｎ個の命令スケジュール部対応にそれぞれ設けられ、対応する命令スケジュール部から発行された演算命令を実行する実行部と、レジスタ部を備え、
前記各命令デコード部は、長語命令を格納する命令バッファと、実行中の長語命令と前記命令バッファ内の長語命令との間における、レジスタのデータ依存関係を判定する判定手段とリソース競合を判定する判定手段と、該両判定手段の判定結果がレジスタのデータ依存関係がなく且つリソース競合がないことを示すとき、実行中の長語命令内の演算命令全ての実行が完了していなくとも命令バッファ内の長語命令を各演算命令に分け対応する命令スケジュール部へ発行制御する手段を備え、
前記各命令スケジュール部は、前記命令デコード部から発行された演算命令を保持する命令スケジュールバッファと、命令スケジュールバッファに保持された演算命令の中から実行部に投入する演算命令を選択し、該選択した演算命令を命令流番号と共に実行部に向け投入制御する手段を備え、
前記各実行部は、前記命令スケジュール部から投入される演算命令を実行する演算命令実行手段と、命令スケジュール部から投入される命令流番号を受け取りこれを保持する手段

と、前記投入された演算命令と命令流番号から、前記演算命令実行手段の動作を制御する手段と、前記演算命令実行手段の状態を前記命令デコード部および命令スケジュール部に通知する手段を備えることを特徴とする情報処理装置。

【請求項２】

請求項１記載の情報処理装置において、

前記実行部の演算命令実行手段は、演算命令の実行により例外発生を検知したとき例外発生信号を前記命令流番号に対応する命令デコード部に通知し、

前記デコード部は、例外発生信号を通知した実行部に対して、前記全ての実行部の演算命令実行手段の状態が受け入れ可能状態のとき、例外発生をした演算命令の再実行を指示する手段を備えることを特徴とする情報処理装置。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】

本発明は、長語命令方式を用いて並列実行性を高め、性能向上を計る情報処理装置に関する。

【０００２】

【従来の技術】

図７に示すのは、長語命令方式（以下、ＶＬＩＷ方式）計算機の一例である。ＶＬＩＷ方式は、長い命令を多数の演算フィールドに分割し、各々の演算フィールドで実行部、レジスタ、メモリ等のリソースを独立して制御し、並列処理を実現する。

ＶＬＩＷ方式では、コンパイル時に並列性の抽出を行なう。コンパイラはプログラムの中から並列実行可能な演算命令を取りだし、１つの長語命令に合成する。

ＶＬＩＷ方式では、コンパイル時にデータ依存関係を始めとした演算命令間の依存関係のチェックを行ない、実行時には依存関係のチェックはしない。そのため、依存関係チェック用のハードウェアが不要となり、ハードウェア量を削減することができる。

【０００３】

しかし、命令間の依存関係をチェックするハードウェアを持たないため、実行部の空きが生じることがある。命令間の依存関係をチェックするハードウェアを持たないＶＬＩＷ方式では、無矛盾性の保障のため、先の長語命令の終了を待ってからでないと次の長語命令を実行できない。

このため、長語命令内に実行時間が他の演算フィールドの演算命令と異なる演算命令が存在すると、最も長くかかる演算命令の終了を待ってから次の長語命令の実行が開始されるため、演算命令が実行されない実行部が生じる。

実行時間に差が生じる例としては、データのロード／ストア命令がある。データのロード／ストア命令は、キャッシュのヒット／ミスヒットにより実行時間が異なる。

また、実行部の空きを生じる別の原因がある。ＶＬＩＷ方式では、演算フィールド数に等しい数の演算命令を１つの長語命令に合成できるとき最高の処理性能を達成できる。どれだけの演算命令を１つの長語命令に合成することができるかは、コンパイラの性能だけではなく、ソースプログラムの性質にも左右される。ソースプログラムに含まれる並列実行できる演算命令の割合が低ければ、１つの長語命令に合成することができる演算命令数は少ない。一般的なプログラムでは、並列実行できる演算命令はそれほど多くない。例えば、日経エレクトロニクス第４８７号によれば、平均して２～３演算命令しか並列に実行できない。

【０００４】

【発明が解決しようとする課題】

上記のように、ＶＬＩＷ方式には、

（ａ）長語命令内に実行時間が異なる演算命令が含まれていると、実行完了
を待ち合わせるため、実行部に空きが生じる。

（ｂ）一般的なプログラムでは、並列性はそれほど高くなく、長語命令内の演算フィールドすべてに演算命令を埋めることができるのはまれで

ある。
といった短所がある。
本発明の目的は、
（１）実行完了待ち合わせにともなう実行部の空きを解消する。
（２）実効並列度を上げる。
手段を提供することにある。
【０００５】
【課題を解決するための手段】
上記目的を達成するため、本発明は、
演算の種類毎に分類されたｎフィールド（ｎは１以上）の演算フィールドを有し、演算フィールド内の各演算命令が他の演算フィールド内の演算命令と互いに依存関係が無いように構成された長語命令からなる命令流ｍ本（ｍは１以上）の処理を行なう情報処理装置であって、ｍ個の命令デコード部と、該命令デコード部から発行された演算命令を受ける前記演算の種類毎に設けられたｎ個の命令スケジュール部と、該ｎ個の命令スケジュール部対応にそれぞれ設けられ、対応する命令スケジュール部から発行された演算命令を実行する実行部と、レジスタ部を備え、前記各命令デコード部は、長語命令を格納する命令バッファと、実行中の長語命令と前記命令バッファ内の長語命令との間におけるレジスタのデータ依存関係およびリソース競合に基づき、実行中の長語命令内の演算命令全ての実行が完了していなくとも前記命令バッファ内の長語命令の前記命令スケジュール部への発行を制御する手段を備えるようにしている。
また、ｍ個の命令デコード部と、該命令デコード部から発行された演算命令を受ける前記演算の種類毎に設けられたｎ個の命令スケジュール部と、該ｎ個の命令スケジュール部対応にそれぞれ設けられ、対応する命令スケジュール部から発行された演算命令を実行する実行部と、レジスタ部を備え、
前記各命令デコード部は、長語命令を格納する命令バッファと、実行中の長語命令と前記命令バッファ内の長語命令との間における、レジスタのデータ依存関係を判定する判定手段とリソース競合を判定する判定手段と、該両判定手段の判定結果がレジスタのデータ依存関係がなく且つリソース競合がないことを示すとき、実行中の長語命令内の演算命令全ての実行が完了していなくとも命令バッファ内の長語命令を各演算命令に分け対応する命令スケジュール部へ発行制御する手段を備え、
前記各命令スケジュール部は、前記命令デコード部から発行された演算命令を保持する命令スケジュールバッファと、命令スケジュールバッファに保持された演算命令の中から実行部に投入する演算命令を選択し、該選択した演算命令を命令流番号と共に実行部に向け投入制御する手段を備え、
前記各実行部は、前記命令スケジュール部から投入される演算命令を実行する演算命令実行手段と、命令スケジュール部から投入される命令流番号を受け取りこれを保持する手段と、前記投入された演算命令と命令流番号から、前記演算命令実行手段の動作を制御する手段と、前記演算命令実行手段の状態を前記命令デコード部および命令スケジュール部に通知する手段を備えるようにしている。
また、前記実行部の演算命令実行手段は、演算命令の実行により例外発生を検知したとき例外発生信号を前記命令流番号に対応する命令デコード部に通知し、前記デコード部は、例外発生信号を通知した実行部に対して、前記全ての実行部の演算命令実行手段の状態が受け入れ可能状態のとき、例外発生をした演算命令の再実行を指示する手段を備えるようにしている。
【０００６】
【作用】
上記手段により、実行完了待ち合わせにともなう実行部の空きを解消することができ、実行部における命令実行を効率よく行なうことができる。
また、複数命令流の命令をスケジューリングすることにより、実効並列度を上げ、データ依存関係に因る実行部の空きを解消できる。その結果、演算器の空きが生じることなく、

高い並列性を得ることができる。
【０００７】
【実施例】
本発明の実施例を図を用いて説明する。以下、二重引用符　”” で囲まれた数字は２進数を、引用符で囲まれていない数字は１０進数を表すとする。
図８は本発明の長語命令の一実施例を示す。本実施例の長語命令は１命令内にＬ／Ｓ１６１、ＦＡｄｄ１６２、ＦＭｕｌｔ１６３、ＦｉｘＯｐ１６４の４演算フィールドを持つ。各演算フィールドには、それぞれ以下の演算命令が入る。Ｌ／Ｓ演算フィールド　　：主記憶とレジスタ間でデータのロード／ストアを行なう演算命令（以下、Ｌ／Ｓ命令）
ＦＡｄｄ演算フィールド　　：浮動小数点加減算を行なう演算命令（以下、ＦＡｄｄ命令）
ＦＭｕｌｔ演算フィールド　：浮動小数点乗算を行なう演算命令（以下、ＦＭｕｌｔ命令）
ＦｉｘＯｐ演算フィールド　：整数演算を行なう演算命令（以下、ＦｉｘＯｐ命令）
本実施例の演算命令は、主記憶とレジスタの間のデータのやり取りをＬ／Ｓ命令のみに限る、ロード／ストア方式である。
【０００８】
各演算フィールドの演算命令の持つオペランドの数は、次のとおりである。
ＦＡｄｄ演算フィールド、ＦＭｕｌｔ演算フィールド、ＦｉｘＯｐ演算フィールド
演算ソース：２
演算ディスティネーション：１
Ｌ／Ｓ演算フィールド
アドレス計算ソース：１
ロードデータディスィネーション：１（データロード命令の場合）
ストアデータソース：１（データストア命令の場合）。
【０００９】
本発明の演算命令の実施例を図９に示す。図９（ａ）はＦＡｄｄ命令、ＦＭｕｌｔ命令、ＦｉｘＯｐ命令の構成を、図９（ｂ）は、Ｌ／Ｓ命令のうちデータロード命令の構成を、図９（ｃ）は、Ｌ／Ｓ命令のうちデータストア命令の構成を示す。
（ａ）図９に示すようにＦＡｄｄ命令、ＦＭｕｌｔ命令、ＦｉｘＯｐ命令は、オペコード、ディスティネーションレジスタ番号、空きフィールド、
ソース１レジスタ番号、ソース２レジスタ番号
の各フィールドから構成される。
命令の実行は、ソース１レジスタ番号とソース２レジスタ番号のフィールドで示されるレジスタの値に、オペコードフィールドで表される演算を施し、ディスティネーションレジスタ番号フィールドで示されるレジスタに格納する。
（ｂ）データロード命令は、
オペコード、ディスティネーションレジスタ番号、オフセット、
アドレス計算レジスタ番号、空きフィールド
の各フィールドから構成される。
命令の実行は、アドレス計算レジスタ番号フィールドで示されるレジスタの値にオフセットフィールドの値を加算し、主記憶のアドレスを求め、このアドレスからデータを、ディスティネーションレジスタ番号フィールドで示されるレジスタに転送する。
（ｃ）データストア命令は、
オペコード、空きフィールド、オフセット、

アドレス計算レジスタ番号、ストアデータレジスタ番号
の各フィールドから構成される。
命令の実行は、アドレス計算レジスタ番号フィールドで示されるレジスタの値にオフセットフィールドの値を加算し、主記憶のアドレスを求め、このアドレスにデータを、ストアデータレジスタ番号フィールドで
示されるレジスタから転送する。
本実施例の長語命令は、コンパイラがデータの依存関係や制御の依存関係の無い演算命令を組み合わせて合成する。コンパイラが長語命令を合成するとき、適切な演算命令が無く、演算命令を埋め込むことができない場合は、その演算フィールドにはＮｏｐ命令が挿入される。Ｎｏｐ命令は何も動作を起こさない演算命令である。

【００１０】
一実施例として、３本の独立した命令流を処理する情報処理装置を図１に示す。
３本の命令流間には、データや制御の依存関係は無い。この３本の命令流を以下、命令流Ａ、命令流Ｂ、命令流Ｃと呼ぶ。
本発明の情報処理装置は、主記憶２１、命令フェッチ部２２、命令デコード部２３、命令スケジュール部２４、実行部２５、レジスタ部２６、その他の装置からなる。このうち、図１では、その他の装置は省略してある。
主記憶２１は、命令流Ａ、命令流Ｂ、命令流Ｃそれぞれのプログラムやデータを格納している。主記憶２１には、キャッシュメモリを含むものとする。
命令フェッチ部２２の働きは、
（１）各命令流毎に、長語命令をフェッチする主記憶のアドレス１３１を主記憶２１に送り、そのアドレスから長語命令１３２を取り出し、命令
デコード部２３に送る
ことである。

【００１１】
命令デコード部２３から命令発行信号１３３が送られてくれば、毎マシンサイクルごと以上の動作を繰り返す。
一実施例として図１に示すような３本の命令流を処理する場合、命令流Ａ、命令流Ｂ、命令流Ｃそれぞれの命令をフェッチするアドレス１３１は、互いに独
立して主記憶２１に送られ長語命令が取り出される。
命令デコード部２３の働きは、
（１）　演算命令発行の可能性を判定する
（２）　命令スケジュール部２４に向け演算命令を発行する
（３）　レジスタの状態の管理する
（４）　命令フェッチ部２２に次命令のフェッチを要求する
ことである。
命令デコード部２３は、毎マシンサイクルごと以上の動作を繰り返す。
一実施例として図１に示すような３本の命令流を処理する場合、命令デコード部２３は、命令流Ａ、命令流Ｂ、命令流Ｃに対応して、命令デコード部Ａ　２３ａ、命令デコード部Ｂ　２３ｂ、命令デコード部Ｃ　２３ｃの３本が設けられている。そして、それぞれの命令デコード部２３は干渉することなく独立して動作する。

【００１２】
命令スケジュール部２４は各命令毎に設けられ、命令スケジュール部２４の働きは、
（１）命令デコード部２３から発行された演算命令１３５のなかから、実行
部２５に投入する演算命令を選択する
ことである。
命令スケジュール部２４は、命令デコード部２３より命令が発行されているならば、毎マシンサイクルごとに以上の動作を繰り返す。
一実施例として図８に示すような４つの演算フィールドを有する長語命令の場合、命令スケジュール部２４は、各演算フィールドに対応して、Ｌ／Ｓ命令用の命令スケジュール部

２４ａ、ＦＡｄｄ命令用の命令スケジュール部２４ｂ、ＦＭｕｌｔ命令用の命令スケジュール部２４ｃ、ＦｉｘＯｐ命令用の命令スケジュール部２４ｄの４本が設けられている。そして、それぞれの命令スケジュール部２４は干渉することなく独立して動作する。

【００１３】

実行部２５は各命令毎に設けられ、実行部２５の働きは、

（１）命令スケジュール部２４から投入される演算命令を実行する

（２）次の演算命令を受け入れ可能であるかを判定する

ことである。

【００１４】

実行部２５は、演算パイプライン化されており、先行演算命令の実行が完了せずとも、次の演算命令実行を開始できる。しかし、演算命令の種類により、また、実行状態により、次の演算命令がいつ実行開始できるかは異なる。そのため、実行部２５が、演算命令の実行状態をみて、次の演算命令を受け入れ可能であるか、あるいは、不可能であるかを判定する。

実行部２５は、命令スケジュール部２４から演算命令が投入されている間は以上の動作を毎マシンサイクルごと繰り返す。

一実施例として図８に示すような４つの演算フィールドを有する長語命令の場合、実行部２５は、各演算フィールドに対応して、Ｌ／Ｓ命令用の実行部２５ａ、ＦＡｄｄ命令用の実行部２５ｂ、ＦＭｕｌｔ命令用の実行部２５ｃ、ＦｉｘＯｐ命令用の実行部２５ｄの４本が設けられている。そして、それぞれの命令実行部２５は干渉することなく独立して動作する。

【００１５】

レジスタ部２６の働きは、

（１）データを保持する

（２）レジスタの状態を保持する

（３）レジスタの状態を更新する

ことである。

レジスタの状態は、

（ａ）未定

（ｂ）確定

の２値をとる。

（ａ）未定状態とは、レジスタの値を書き換える演算命令が命令デコード部２３から発行されたが、未だ実行が完了しておらず、したがって、レジスタ

の値が未定な状態である。

（ｂ）確定状態とは、演算命令が完了し、レジスタの値がすでに確定している

状態である。

一実施例として図１に示すような３本の命令流を処理する場合、レジスタ部２６は、各命令流に対応して、２６ａ、２６ｂ、２６ｃの３本が設けられている。そして、それぞれのレジスタ部２６は干渉することなく独立して動作する。

【００１６】

命令フェッチ部２２の動作をさらに詳しく説明する。

ある命令に注目してみた場合、命令フェッチ部２２は、主記憶２１に命令フェッチアドレス１３１を与え、主記憶２１から長語命令を取り出し、信号線１３４を使って命令デコード部２３内の命令バッファ３１（図２）に送る。

先行長語命令が命令デコード部２３から発行されると次の長語命令を主記憶２１からフェッチする。命令デコード部２３から長語命令が発行されたかどうかは、命令発行信号１３３に示される。

命令発行信号１３３が”１”

のとき、命令デコード部２３から長語命令が発行され、命令フェッチ部１１は命令フェッチを行なう。

命令発行信号１３３が”０”
ならば、命令デコード部２３から長語命令が発行されず、したがって、命令フェッチも行なわない。以上は、各命令流ごと独立して行なわれる。
【００１７】
次に、命令デコード部２３の動作をさらに詳しく説明する。図２に命令デコード部２３の一実施例を示す。
図２中、命令デコード部２３は、命令バッファ３１、競合判定部３２、依存関係解明部３３、実行制御部３４、受け入れ可能フラグ３５、命令発行器３６から構成される。
命令デコード部２３を構成する各部の動作を以下に説明する。
命令バッファ３１は、命令フェッチ部２２から送られてくる長語命令を各命令スケジュール部２４に向け発行するまでの間保持する。
【００１８】
競合判定部３２は実行部２５の競合状態を調べる。図１０に競合判定部３２の一実施例を示す。
実行部２５の競合状態とは、
（ａ）命令バッファ３１内の長語命令にＮｏｐ命令でない演算フィールドが有り、
かつ、
（ｂ）その、演算フィールドに対応する実行部２５に演算命令の受け入れが不可能なものが１以上存在する状態である。
各実行部２５の演算命令受け入れ可能であるか不可能であるかは、受け入れ可能信号１４３が書き込まれた受け入れ可能フラグ３５に示される。
競合判定部３２は、受け入れ可能フラグ３５（受入れ可能なら”１”、そうでなければ”０”）と、命令バッファ３１の各演算命令とＮｏｐ命令とを比較した比較結果（演算命令がＮｏｐ命令なら”１”、そうでなければ”０”）から、競合状態を調べ、信号線１５１に結果を出力する。
競合判定出力１５１の値は、
実行部４本全てが競合状態でない：競合判定出力１５１　＝　”１”
実行部４本のうち１本でも競合状態：競合判定出力１５１　＝　”０”
である。
【００１９】
依存関係解明部３３は、実行中の演算命令と命令バッファ３１内の長語命令間で、レジスタのデータ依存関係を調べる。図１１に依存関係解明部３３の一実施例を示す。
データ依存関係には、
（ａ）　ＲＡＷ　の依存関係
（ｂ）　ＷＡＲ　の依存関係
（ｃ）　ＷＡＷ　の依存関係
がある。まず、以上３種類のデータ依存関係について説明するＲ１～Ｒ５はレジスタを表すとする。Ｒ１＜－Ｒ２＋Ｒ３　は、Ｒ２とＲ３の値を加算して、その結果をＲ１に代入することを表す。
（ａ）ＲＡＷ（Ｒｅａｄ　Ａｆｔｅｒ　Ｗｒｉｔｅ）の依存関係
１：Ｒ１＜－Ｒ２＋Ｒ３
２：Ｒ４＜－Ｒ１＋Ｒ５
１、２の順で命令を実行した場合、Ｒ１の値が確定するまで、２の命令は
実行開始することができない。
（ｂ）ＷＡＲ（Ｗｒｉｔｅ　Ａｆｔｅｒ　Ｒｅａｄ）の依存関係
１：Ｒ２＜－Ｒ１＋Ｒ３
２：Ｒ１＜－Ｒ４＋Ｒ５
１、２の順で命令を実行した場合、１の命令が実行開始した後でないと、
２の命令の結果をＲ１に書き込めない。
（ｃ）ＷＡＷ（Ｗｒｉｔｅ　Ａｆｔｅｒ　Ｗｒｉｔｅ）の依存関係

１：Ｒ１＜－Ｒ２＋Ｒ３
２：Ｒ１＜－Ｒ４＋Ｒ５
１、２の順で命令を実行した場合、１の命令が結果を書き込んだ後でない
と、２の命令の結果をＲ１に書き込めない。
以上、３通りの依存関係のうち、ＷＡＲの依存関係は生じない。それは、先行する長語命令内の演算命令全てが実行開始した後、すなわち、ソースレジスタの値を読み出した後でないと、次の長語命令を命令デコード部２３から発行しないためである。
よって、依存関係解明部３３が解明しなければならないデータ依存関係は、
（ａ）ＲＡＷの依存関係、
（ｃ）ＷＡＷの依存関係
である。
この、２種類の依存関係は、レジスタ値が書き込まれる前に、
（ａ）レジスタの読みだし、
（ｃ）レジスタの書き込み
を行なうと生じる。すなわち、未定状態のレジスタを使用しようとすると、
（ａ）ＲＡＷの依存関係、
（ｃ）ＷＡＷの依存関係
は生じる。
よって、依存関係解明部３３では、レジスタの状態が未定状態であるかを調べれば、データ依存関係を解明できる。
レジスタの状態は、レジスタ部２６が持つ予約ビット１７４（図６）に示される。未定状態のレジスタの予約ビット１７４の値は”１”である。
命令バッファ３１に保持された演算命令のオペランドの各レジスタの予約ビット１７４を調べ、依存関係解明出力１５２に出力する。
依存関係解明出力１５２の値は、
全ての予約ビットが”０”：データ依存関係無、
依存関係解明出力１５２＝”１”
”１”の予約ビットが存在：データ依存関係有、
依存関係解明出力１５２＝”０”
である。
【００２０】
実行制御部３４は、各実行部２５の実行状態を監視し、自命令流の演算命令により例外が発生したならば、例外処理を行なう。他の命令流の演算命令が例外を発生された場合、何もしない。
図１２に実行制御部３４の一実施例を示す。
実行制御部３４は、例外発生フラグ５３と再実行指示信号生成論理５４から構成される。
各実行部２５で例外（０で除算した場合、オーバーフローが発生した場合等）が発生すると、実行部２５は、対応する命令流の命令デコード部２３内の実行制御部３４に、例外が発生したことを例外発生信号１４１を通じ報告する。
例外発生信号１４１を受け取った実行制御部３４は、例外発生フラグ５３に書き込む。例外発生フラグ５３は一実施例として図１に示すような４つの実行部を有する場合、４ｂｉｔで構成され、それぞれのビットが各実行部２５に対応する。
例外処理は、演算例外を起こした演算命令を含む長語命令内の全ての演算命令の実行が全て終了した後、開始する。全ての演算命令の実行が終了したか／否かは、受け入れ可能フラグ３５の全ビットのＡＮＤ（論理積）をとった信号線１４３－１の値を基に検出する。
再実行指示信号生成論理５４は、実行部２５に対し演算命令の再実行を指示する信号１４２を生成する。例外発生フラグ５３の値と、信号線１４３－１の値のＡＮＤ（論理積）をとることで、再実行指示信号１４２を生成する。再実行開始時、対応する演算フィールドの再実行指示信号１４２は、

再実行指示信号１４２＝”１”
になる。
例外処理開始待機中あるいは例外処理実行中であるか、正常状態であるかは、実行状態出力１５３に出力される。実行状態出力１５３の値は、
正常状態：実行状態出力１５３＝”１”
例外処理開始待機中／例外処理実行中：実行状態出力１５３＝”０”
である。
実行状態出力１５３は、例外発生フラグ５３全ビットのＮＯＲ（否定論理和）を取ることで生成される。例外発生の報告を受けた後、例外処理開始を待っている間、また、例外処理中
は、
信号線１５３＝”０”
になり、例外処理開始待機中／例外処理実行中を示す。その結果、発行条件が整っていても命令バッファ３１内の長語命令は発行されない。
【００２１】
受け入れ可能フラグ３５は、実行部２５が演算命令を受け入れ可能であるかどうかを示すフラグで、一実施例として図１に示すような４つの実行部を有する場合、４ｂｉｔで構成され、各ビットがそれぞれの実行部に対応している。受け入れ可能フラグ３５の各ビットの値は、
受け入れ可能　：”１”
受け入れ不可能：”０”
である。
受け入れ可能フラグ３５は、各実行部２５が、演算命令の実行状態に応じ書き換える。
【００２２】
命令発行信号１３３は、命令バッファ３１から長語命令が発行されることを示す信号で、長語命令が発行される場合”１”
になる。長語命令が発行される条件、すなわち、命令発行信号１３３が”１”になる条件は、
（ａ）競合関係がなく（競合判定出力１５１＝”１”）
かつ、
（ｂ）データ依存関係がなく（依存関係解明出力１５２＝”１”）
かつ、
（ｃ）正常状態（実行状態出力１５３＝”１”）
である。
命令発行信号１３３が”１”のとき、命令発行器３６が開き、命令バッファ３１内の長語命令が演算命令に分けられて命令スケジュール部２４に向け発行される。また、命令発行信号１３３の値は命令フェッチ部２２にも送られ、次の長語命令のフェッチを指示する。
命令発行器３６は、命令発行信号１３３の値に基づき、命令バッファ３１内の長語命令を命令スケジュール部２４に向け発行する。命令発行器３６は、命令発行信号１１４が”１”のとき、命令バッファ３１内の長語命令を命令スケジュール部２４に向け発行する。
信号線１４０は、発行された演算命令の結果格納先レジスタの予約ビットを、
予約ビット＝”１”
にすることを、レジスタ部２６に指示する信号である。信号線１４０は、演算命令発行時に、
信号線１４０＝”１”
になる。信号線１４０は、命令発行信号１３３と同じ条件で生成される。
【００２３】
次に、命令スケジュール部２４の働きについて詳しく説明する。図３に命令スケジュール部の一実施例を示す。
命令スケジュール部２４は、スケジュールバッファ３７、Ｏｐｅｃｏｄｅコンパレータ４

０、スレッド選択論理３８、２ｂｉｔカウンタ（一実施例として図１に示すように３命令流の場合）３９、投入演算命令セレクタ４１からなる。
スケジュールバッファ３７は、各命令デコード部２３から発行された演算命令を、実行部２５に投入するまで保持する。スケジュールバッファ３７は、一実施例として図１に示すように３命令流の場合、３エントリから構成される。例えば、命令スケジュール部がＬ／Ｓ命令用の命令スケジュール部２４ａの場合、各エントリには各命令流のＬ／Ｓ命令あるいはＮｏｐ命令が格納される。
Ｏｐｅｃｏｄｅ　コンパレータ４０は、スケジュールバッファ内の演算命令がＮｏｐ命令であるかどうかを判定し、信号線１５５に出力する。信号線１５５の値は、
Ｎｏｐ命令　　：信号線１５５＝”１”
Ｎｏｐ命令以外：信号線１５５＝”０”
である。
一実施例として図１に示すような３本の命令流を処理する場合、Ｏｐｅｃｏｄｅ　コンパレータ４０は各命令流に対応して３本設けられている。
【００２４】
スレッド選択論理３８は、
（ａ）　投入する演算命令の選択
を行なう。
図１３にスレッド選択論理の一実施例を示す。
（ａ）投入する演算命令の選択は、２ｂｉｔカウンタ３９の値を基に行なわれ、投入演算命令セレクタ４１を制御するスレッド選択信号１５４を、２ｂｉｔカウンタ３９の値から生成する。
一実施例として図１に示すような３本の命令流を処理する場合、各命令流を識別する命令流番号をそれぞれ、
”００”（命令流Ａ）、
”０１”（命令流Ｂ）、
”１０”（命令流Ｃ）
とする。”１１”は、選択する命令流がないことを表す。
【００２５】
スレッド選択信号１５４の生成手順を図１３に示す。図１３中の記号の意味は次ぎのとおりである。
【００２６】
ＴＮＣ　　：命令流番号カウンタ３９の値
ＳＴＮ　　：開始時の命令流番号カウンタ３９の値を保持する作業用のレジスタ
Ｓｉｇｎａｌ１４１　　：例外発生信号１４１の値
Ｓｉｇｎａｌ１４３　　：受入れ可能信号１４３の値
Ｓｉｇｎａｌ１５４　　：スレッド選択信号１５４の値
Ｓｉｇｎａｌ１５５（ｎ）　：命令流番号ｎの命令流の信号線１５５の値
Ａ＜－Ｂ　　：ＡにＢの内容を転送する動作を示す
Ａ＝＝Ｂ　　：ＡとＢの値が等しいかどうかを判定
図１３のスレッド選択信号１５４生成手順を説明する。左端カッコ付きの数字は、図１３のカッコ付きの数字に対応する。
【００２７】
（１）：　Ｓｉｇｎａｌ１４３＝＝”０”（実行部が新たな演算命令を受入れ不可能）
あるいは
Ｓｉｇｎａｌ１４１＝＝”０”（例外発生中）
ならば、スレッド選択信号の生成を開始せず、開始できるまで待ち合わせる。

（２）：ＳＴＮにＴＮＣを転送する。ＳＴＮはスレッド選択信号１５４生成を終了させるのに使用する。
（３）：Ｓｉｇｎａｌ１５５（ＴＮＣ）＝”０”ならば、ＴＮＣが示す命令流番号のオペコードはＮｏｐ命令ではない。従って、ＴＮＣの示す命令流を選択する。
（４）：スレッド選択信号１５４の値（Ｓｉｇｎａｌ１５４）をＴＮＣの値にし、ＴＮＣの値を更新する。
（５）：ＴＮＣを更新した結果、ＴＮＣ＝”１１”ならば、ＴＮＣを”００”にして終了。ＴＮＣ＝”１１”でなければ、そのまま終了。
（６）：（３）で、Ｓｉｇｎａｌ１５５（ＴＮＣ）＝”０”でなければ、ＴＮＣが示す命令流番号のオペコードはＮｏｐ命令。ＴＮＣを更新して、再びオペコードを調べる。
（７）：ＴＮＣを更新した結果、ＴＮＣ＝”１１”ならば、ＴＮＣを”００”にする。
（８）：ＳＴＮ＝ＴＮＣならば、ＴＮＣの値が一回りした。

【００２８】

このときは、
ＴＮＣを”１１”（選択命令流無し）
スレッド選択信号１５４の値を”１１”（選択命令流無し）
にして終了。
投入演算命令セレクタ４１は、スレッド選択論理３８からの命令流選択信号１５４により実行部２５に投入する演算命令を選択する。
命令流選択信号１５４が示す命令流番号の演算命令を実行部２５に投入するようにセレクタが開く。

【００２９】

図４～図５に実行部２５の一実施例を示す。
図４に示すのは、メモリアクセスを伴わない演算を実行する実行部２５の実施例である。また、図５に示すのは、メモリアクセスをともなう演算命令を実行する実行部２５の実施例である。
メモりアクセスを伴わない演算を実行する実行部２５から説明する。図４に示す実行部２５は、演算器４２、命令流番号レジスタ４４、演算制御器４３からなる。
演算器４２は、実行部２５に投入された演算命令１３７にしたがいレジスタ部２６とデータをやり取りし、演算を行なう。一実施例として図４に示す実行部２５の場合、演算器４２は２ｓｔａｇｅの演算パイプライン化されている。
演算器４２は、演算の段階ごとに実行状態を信号線１５６を通じ演算制御器４３に送る。また、演算の各段階の制御も信号線１５６を通じ演算制御器４３から送られる。
命令流番号レジスタ４４は、実行中の演算命令の命令流番号を保持している。
命令流番号レジスタ４４に保持された命令流番号は、
（ａ）演算実行に伴うレジスタ部２６のアクセス
（ｂ）命令デコード部２３に対する例外発生の通知
に使用される。
演算制御器４３は、
（ａ）演算器４２の制御
（ｂ）例外発生の検出と通知
（ｃ）例外処理の制御
を行なう。
（ａ）演算制御器４３は、命令スケジュール部２４から投入された演算命令１３７を実行するために、信号線１５６を使って演算器４２を制御する。演算命令実行の各段階の演算器４２の状態を信号線１５６を通じ、演算制御器４３は知る。
演算命令１３７の実行が進み、新たな演算命令を受け入れることが可能になったならば、

受け入れ可能信号１４３＝”１”
にして、命令スケジュール部２４、命令デコード部２３に知らせる。命令デコード部２３に送られた信号線１４３は、命令デコード部２３内の受け入れ可能フラグ３５の値を書き換える。
（ｂ）演算命令を実行中、演算制御器４３が例外を検出すると、
例外発生信号１４１＝”１”
にして、命令デコード部２３に例外の発生を知らせる。演算制御部４３が例外を通知する命令デコード部２３は、命令流レジスタ４４に保持された命令流番号により特定される。
（ｃ）命令デコード部２５から再実行指示信号１４２を受け取ると、演算制御部４３は例外を発生させた演算命令の再実行をする。再実行が完了したならば、
受け入れ可能信号１４３＝”１”
にして、命令スケジュール部２４、命令デコード部２３に再実行の完了を知らせる。
【００３０】
次に、図５に示す実行部について説明する。
図４に示す実行部の実施例と違い、メモリアクセスのために、メモリアクセス制御器４７が設けられている。また、アドレス演算器４５は、図４に示す実行部４２の実施例とは異なり、アドレス演算を行なう。
アドレス演算器４５はメモリアクセスアドレス１４６を計算するために使用される。
アドレス演算器４５は、演算命令１３７のアドレス計算オフセット１６９（図９）の値と、アドレス計算ソース１４４から、メモリアクセスアドレス１４６を演算する。アドレス演算器の出力（メモリアクセスアドレス１４６）は、メモリアクセス制御器４７に入力する。
実行制御器４６は、メモリアドレスを計算するよう信号線１５６を通じてアドレス演算器４５に指示を出し、同時に、信号線１５７を通じ、メモリアクセス制御器４７に、メモリアクセスの種類を知らせる。
メモリアクセスの状態は、信号線１５７を通じ実行制御器４６に知らされる。実行制御器４６に伝えられるメモリアクセスの状態には、例として、キャッシュメモリのヒット／ミスヒット、がある。
（ａ）　演算命令１３７がデータロード命令の場合のメモリアクセス制御器４７の動作を説明する。
１）メモリアクセスアドレス１４６を信号線１２１を通じ主記憶２１に出力
２）データが転送されてくるのを待つ
３）障害が発生したならば、信号線１５７を通じ実行制御部４６に知らせる
。障害の一例としてキャッシュメモリのミスヒットがある。
４）データが信号線１２０を通じ転送されてきたならば、信号線１４５を通じ、転送されてきたレジスタ部２６に書き込む
５）レジスタ部２６にデータを書き込んだ段階で、
受け入れ可能信号１４３＝”１”
にする。
（ｂ）演算命令１３７がデータストア命令の場合のメモリアクセス制御器４７の動作を説明する。
１）メモリアクセスアドレス１４６を信号線１２１を通じ主記憶２１に出力
２）続いて、ストアデータ１４４を信号線１２１を通じ主記憶２１に出力
３）障害が発生したならば、信号線１５７を通じ実行制御器４６に知らせる
。障害の一例としてキャッシュメモリのミスヒットがある。
４）主記憶２１にデータを書き込んだ段階で、
受け入れ可能信号１４３＝”１”
にする。
【００３１】
レジスタ部２６についてさらに詳しく説明する。図６にレジスタ部２６の一実施例を示す

。
レジスタ部内の各レジスタは、
（１）　データ領域１７３
（２）　予約ビット１７４
（３）　ソース１レジスタ番号保持ラッチ１７５
（４）　ソース２レジスタ番号保持ラッチ１７６
（５）　ディスティネーションレジスタ番号保持ラッチ１７７
から構成される。
データ領域１７３は、レジスタが持つ値を格納しておく領域である。
予約ビット１７４は、レジスタの状態を表すビットで、図６に示す実施例では、１ｂｉｔで構成されている。
予約ビット１７４の値は、
未定：予約ビット＝”１”
確定：予約ビット＝”０”
である。
予約ビット１７４の値は、データの依存関係解明のため命令デコード部２３内の依存関係解明部３３から読み出される。
【００３２】
ソース１レジスタ番号保持ラッチ１７５は、レジスタ番号１３９に示されるレジスタ番号のうち、ソース１レジスタ番号フィールド１６７、あるいは、アドレス計算レジスタ番号フィールド１７０のレジスタ番号を保持する。
ソース１レジスタ番号フィールド１６７、あるいは、アドレス計算レジスタ番号フィールド１７０が演算命令にない場合は、’Ｉｎｖａｌｉｄ　ｎｕｍｂｅｒ’が保持される。’Ｉｎｖａｌｉｄ　ｎｕｍｂｅｒ’はレジスタ番号としては無意味な数である。
ソース２レジスタ番号保持ラッチ１７６は、レジスタ番号１３９に示されるレジスタ番号のうち、ソース２レジスタ番号フィールド１６８、あるいは、ストアレジスタ番号フィールド１７１のレジスタ番号を保持する。ソース１レジスタ番号フィールド１６７、あるいは、アドレス計算レジスタ番号フィールド１７０が演算命令にない場合は、’Ｉｎｖａｌｉｄ　ｎｕｍｂｅｒ’が保持される。
ディスティネーションレジスタ番号保持ラッチ１７７は、レジスタ番号１３９に示されるレジスタ番号のうち、ディスティネーションレジスタ番号フィールド１６６のレジスタ番号を保持する。ディスティネーションレジスタ番号フィールド１６６が演算命令にない場合は、’Ｉｎｖａｌｉｄ　ｎｕｍｂｅｒ’が保持される。
レジスタ部２６を構成する各１本のレジスタは、ソース１レジスタ番号保持ラッチ１７５、ソース２レジスタ番号保持ラッチ１７６、ディスティネーションレジスタ番号保持ラッチ１７７に保持されたレジスタ番号により特定される。
【００３３】
演算命令が発行されると信号線１４０が有効になる。信号線１４０が有効になったならば、レジスタ部２６はディスティネーションレジスタ番号保持ラッチ１７７に保持されたレジスタの予約ビット１７４の値を、
予約ビット１７４＝”１”
にして、レジスタの状態を未定状態にする。
演算命令の実行完了に伴い、演算結果が信号線１４５を通じ書き込まれたならば、レジスタ部２６はディスティネーションレジスタ番号保持ラッチ１７７に保持されたレジスタの予約ビットの値を、
予約ビット＝”０”
にして、レジスタの状態を確定状態にする。
ソース１レジスタ番号保持ラッチ１７５、ソース２レジスタ番号保持ラッチ１７６、ディスティネーションレジスタ番号保持ラッチ１７７は、一実施例として図８に示すような４演算フィールドからなる長語命令を処理する場合、それぞれ４組づつ設けられている。

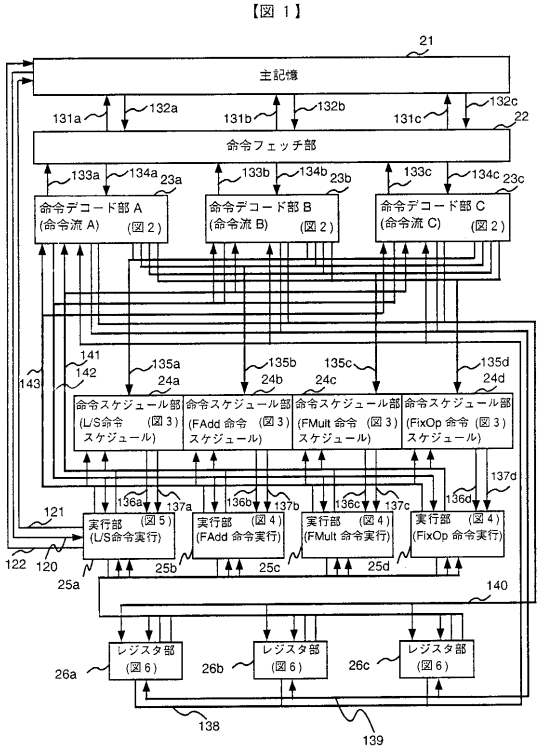
【００３４】
次に長語命令がどのように処理されるかを説明する。一実施例として図１に示すような、命令流Ａ、命令流Ｂ、命令流Ｃの３本の命令流を処理する場合を例に取り上げる。
以下の説明で、左端の数字は、命令デコード部２３ａが命令発行信号１３３ａを命令フェッチ部２２に送った時点から計ったマシンサイクル数を表す。
００　　命令デコード部２３ａが命令発行信号１３３ａを命令フェッチ部２２に送る。
０１　　命令発行信号１３３ａを受け取った命令フェッチ部は２２は、命令フェッチアドレス１３１ａを主記憶２１に送り、長語命令がフェッチされてくるのを待つ。
０２　　キャッシュにヒットしたならば、この時刻に長語命令が命令フェッチ部２２に取り込まれる。
キャッシュがミスヒットしたならば、長語命令が取り込まれるまで待合せを行なう。
以下の処理は、キャッシュのヒット／ミスヒットに関わりなく同じであるため、キャッシュにヒットしたとして説明を続ける。
０３　　命令フェッチ部２２に取り込まれた長語命令は、命令デコード部２３ａ内の命令バッファ３１に転送される。
【００３５】
０４　　命令バッファ３１内の長語命令は、競合判定部３２、依存関係解明部３３に送られ、実行部２５の競合判定と、レジスタの依存関係が調べられる。
依存関係解明部３３に送られると同時に、長語命令から、レジスタ番号が取り出され、信号線１３９を通じ、レジスタ部２６ａに送られる。レジスタ部２６ａは、ソース１レジスタ番号保持ラッチ１７５、ソース２レジスタ番号保持ラッチ１７６、ディスティネーションレジスタ番号保持ラッチ１７７にそれぞれの演算フィールドごと分けて記憶する。
さらに、レジスタ部２６ａ信号線１３９を通じ送られてきた前レジスタの予約ビット１７４の値を、信号線１３８を通じ、命令デコード部２３ａに送る。
この時刻に、
（ａ）競合関係がなく（競合判定出力１５１＝”１”）
かつ、
（ｂ）データ依存関係がなく（依存関係解明出力１５２＝”１”）
かつ、
（ｃ）正常状態（実行状態出力１５３＝”１”）
であれば、
命令発行信号１３３＝”１”
になる。
０５　　命令発行信号１３３＝”１”
ならば、命令発行器３６が開き、命令バッファ３１内の長語命令が、演算命令に分割され、各命令スケジュール部２４内のスケジュールバッファ３７に向け発行される。
同時に、命令発行信号１３３ａは命令フェッチ部１１に送られ、次のマシンサイクルで、命令流Ａの次の長語命令をフェッチする。
さらに、予約化信号１４０がレジスタ部２６ａに送られ、ディスティネーションレジスタ番号保持ラッチに記憶されたレジスタの予約ビット１７４を未定状態にする。

【００３６】
０６　命令スケジュール部２４ａ～ｄの処理はどれでも同じである。そこで、命令スケジュール部２４ｂの動作について述べる。
図１３に示す手順により生成されるスレッド選択信号１５４を基に、スケジュールバッファ３７内の演算命令は選択される。
図１３中の変数の値を以下のように仮定する。
命令流カウンタ３９の値（ＴＮＣ）　：”１０”
例外発生信号１４１の値（Ｓｉｇｎａｌ１４１）　：　”０”
受け入れ可能信号１４３の値（Ｓｉｇｎａｌ１４３）　：　”１”
Ｓｉｇｎａｌ１５５（”１０”）　：　”１”
Ｓｉｇｎａｌ１５５（”００”）　：　”０”
以下、左端のカッコ付きの数字は図１３中のカッコ付きの数字に対応する。
（１）Ｓｉｇｎａｌ（１４１）＝”０”
かつ、Ｓｉｇｎａｌ（１４３）＝”１”であるから、
（２）ＳＴＮにＴＮＣの値を転送する。
（３）Ｓｉｇｎａｌ１５５（”１０”）＝”１”より、
（６）ＴＮＣをＴＮＣ＋１　とする。
（７）ＴＮＣ＝”１１”であるから、ＴＮＣの値は、”００”
（８）ＳＴＮ（”１０”）の値とＴＮＣ（”００”）の値が違うので、
（３）Ｓｉｇｎａｌ１５５（”００”）を評価
Ｓｉｇｎａｌ１５５（”００”）＝”００”より、
（４）Ｓｉｇｎａｌ１５４にＴＮＣ（”００”）を転送
ＴＮＣの値は”０１”に更新される。
（５）ＴＮＣ＝”１１”でないので、ＴＮＣの値はそのまま。
スレッド選択信号１５４の値は、”００”であるので命令流Ａが選択される。
【００３７】
０７　投入演算命令セレクタ４１が開き、命令流Ａの演算命令が実行部２５ｂ内の演算制御器４５に向け送られる。同時に命令流番号（”００”）も実行部２５ｂ内の命令流番号レジスタ４４に送られる。
０８　命令流番号レジスタ４４に記憶された命令流番号（”００”）をもとに、レジスタ部２６ａを選択し、演算に必要なデータを実行部２５ｂに転送する。
０９　演算を開始する。
演算器４２は演算パイプライン化されているため、新たな命令が受け入れ可能になり、信号線１４３＝”１”になる。
１０　演算実行中。新たな命令が命令スケジュール部２４ｂから送られてくる。
１１　演算終了。演算結果をレジスタ部２６に戻す。
レジスタ部２６ａはディスティネーションレジスタ番号保持ラッチに記憶されたレジスタのデータ領域に演算結果を書き込むと同時に予約ビットを確定状態にする。
一実施例として図１に示すような３本の命令流を処理する場合、以上のように長語命令は実行される。
【００３８】
【発明の効果】
本発明によれば、命令実行の並列性低下の原因となる、データ依存関係および演算器の競合を始めとしたリソース競合を回避でき、その結果実行部が演算命令を実行していない空き時間を減少させ、高い効率で実行部を使用することができる。

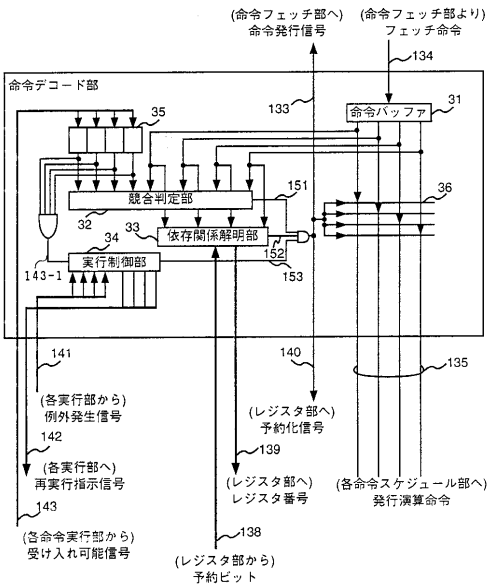
【図面の簡単な説明】
【図１】本発明の一実施例を示す図である。
【図２】命令デコード部の一実施例を示す図である。
【図３】命令スケジュール部の一実施例を示す図である。
【図４】実行部の一実施例を示す図である。
【図５】メモリアクセスを行なう実行部の一実施例を示す図である。
【図６】レジスタ部の一実施例を示す図である。
【図７】従来のＶＬＩＷ方式の情報処理装置を示す図である。
【図８】本発明の長語命令の一実施例を示す図である。
【図９】本発明の長語命令を構成する演算命令の一実施例を示す図である。
【図１０】競合判定部の一実施例を示す図である。
【図１１】依存関係解明部の一実施例を示す図である。
【図１２】実行制御部の一実施例を示す図である。
【図１３】スレッド選択論理がスレッド選択信号を生成する手順を示す図である。
【符号の説明】
２１　主記憶
２２　命令フェッチ部
２３ａ　命令流Ａ命令デコード部
２３ｂ　命令流Ｂ命令デコード部
２３ｃ　命令流Ｃ命令デコード部
２４ａ　Ｌ／Ｓ命令スケジュール部
２４ｂ　ＦＡｄｄ命令スケジュール部
２４ｃ　ＦＭｕｌｔ命令スケジュール部
２４ｄ　ＦｉｘＯｐ命令スケジュール部
２５ａ　Ｌ／Ｓ命令実行部
２５ｂ　ＦＡｄｄ命令実行部
２５ｃ　ＦＭｕｌｔ命令実行部
２５ｄ　ＦｉｘＯｐ命令実行部
２６ａ　命令流Ａレジスタ部
２６ｂ　命令流Ｂレジスタ部
２６ｃ　命令流Ｃレジスタ部
３１　命令バッファ
３２　競合判定部
３３　依存関係解明部
３４　実行制御部
３５　受け入れ可能フラグ
３６　命令発行器
３７　スケジュールバッファ
３８　スレッド選択論理
３９　命令流番号カウンタ
４０ａ～４０ｄ　ｏｐｅｃｏｄｅ　コンパレータ
４１　投入演算命令セレクタ
４２　演算器
４３　演算制御器
４４　命令流番号レジスタ
４５　アドレス演算器
４６　実行制御器
４７　メモリアクセス制御器
５０　レジスタ番号バッファ
５１ａ　Ｌ／Ｓ演算フィールドレジスタ番号バッファ

５１ｂ　ＦＡｄｄ演算フィールドレジスタ番号バッファ
５１ｃ　ＦＭｕｌｔ演算フィールドレジスタ番号バッファ
５１ｄ　ＦｉｘＯｐ演算フィールドレジスタ番号バッファ
５２　依存関係解明論理
５３　例外発生フラグ
５４　再実行指示信号生成論理
７１　従来のＶＬＩＷ方式情報処理装置の主記憶
７２　従来のＶＬＩＷ方式情報処理装置の命令フェッチ部
７３　従来のＶＬＩＷ方式情報処理装置の命令デコード部
７５ａ～ｄ　従来のＶＬＩＷ方式情報処理装置の演算器
７６　従来のＶＬＩＷ方式情報処理装置のレジスタ部
８１　従来のＶＬＩＷ方式情報処理装置の命令バッファ
８５　従来のＶＬＩＷ方式情報処理装置の受け入れ可能ビット
８６　従来のＶＬＩＷ方式情報処理装置の命令発行器
１７５　ソース１レジスタ番号保持ラッチ
１７６　ソース２レジスタ番号保持ラッチ
１７７　ディスティネーションレジスタ番号保持ラッチ

【図１】

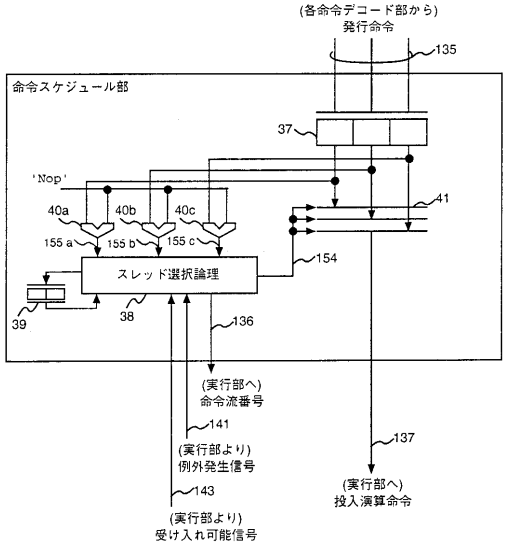


【図２】



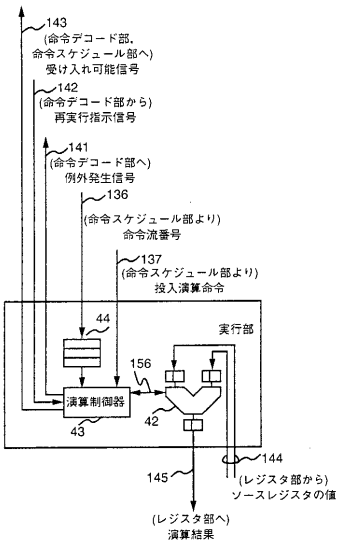
【図３】

【図３】



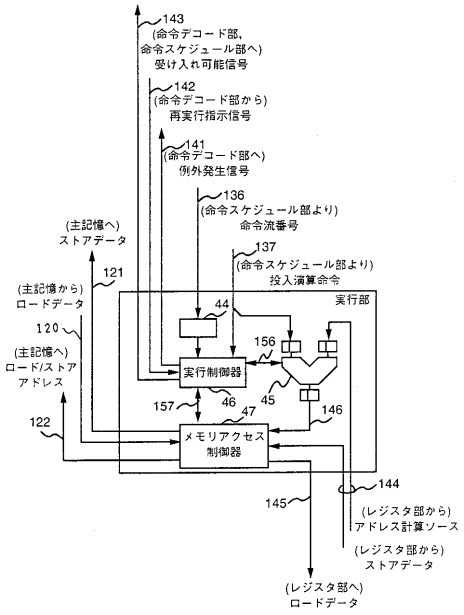
【図４】

【図４】



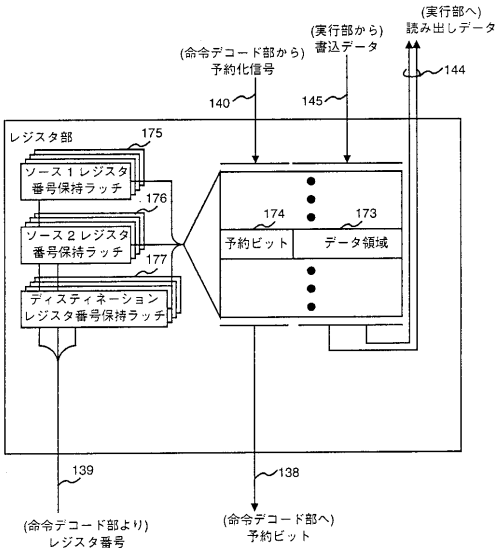
【図５】

【図５】



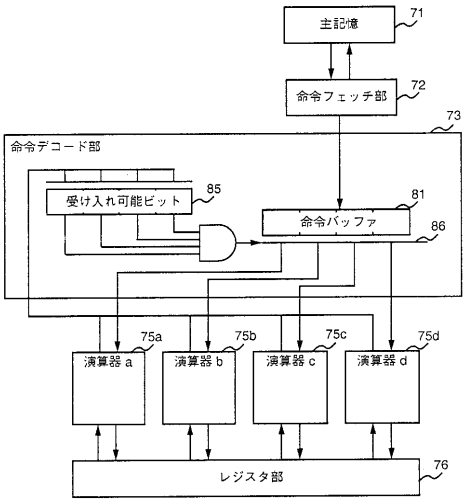
【図６】

【図６】



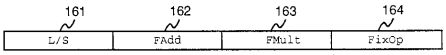
【図７】

【図7】



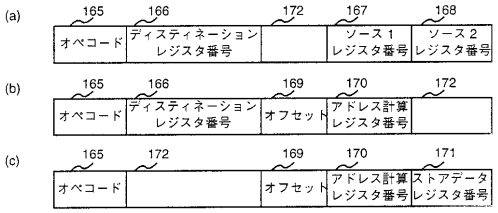
【図８】

【図8】



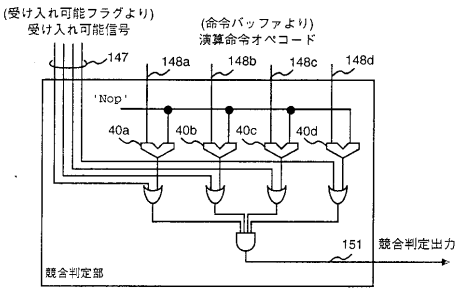
【図９】

【図9】



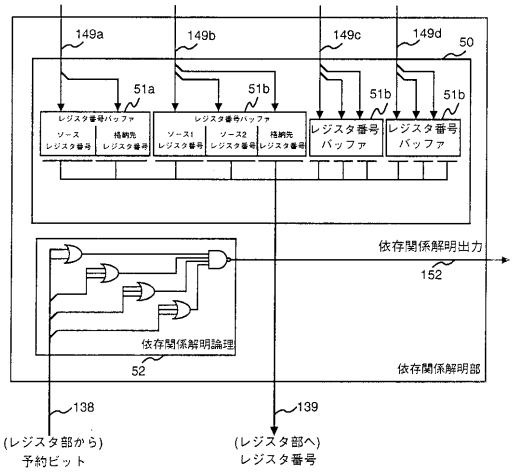
【図１０】

【図１０】



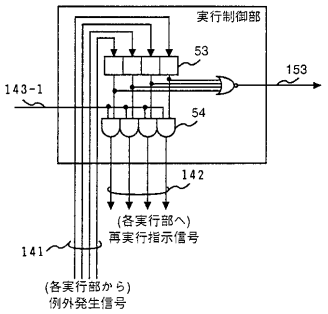
【図１１】

【図１１】



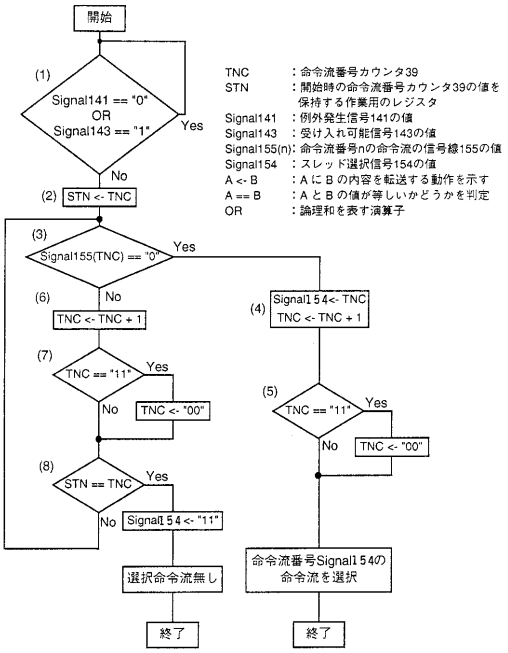
【図１２】

【図１２】



【図１３】

【図１３】


開始
(1)
Signal141 == "0"
OR
Signal143 == "1"
Yes
No
(2) STN <- TNC
(3)
Signal155(TNC) == "0"
Yes
No
(6)
TNC <- TNC + 1
(7)
TNC == "11"
Yes
No
TNC <- "00"
(8)
STN == TNC
Yes
No
Signal154 <- "11"
選択命令流無し
終了
(4)
Signal154 <- TNC
TNC <- TNC + 1
(5)
TNC == "11"
Yes
No
TNC <- "00"
命令流番号Signal154の
命令流を選択
終了
TNC ：命令流番号カウンタ39
STN ：開始時の命令流番号カウンタ39の値を
保持する作業用のレジスタ
Signal141 ：例外発生信号141の値
Signal143 ：受け入れ可能信号143の値
Signal155(n): 命令流番号nの命令流の信号線155の値
Signal154 ：スレッド選択信号154の値
A <- B ：A に B の内容を転送する動作を示す
A == B ：A と B の値が等しいかどうかを判定
OR ：論理和を表す演算子

フロントページの続き

(56)参考文献 特開平５－１９７５４７（ＪＰ，Ａ）
特開平７－１９１８４７（ＪＰ，Ａ）
特開平７－１０５００３（ＪＰ，Ａ）
特開平４－１２３２３０（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.[7]，ＤＢ名)
G06F 9/38