(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第4608099号
(P4608099)
(45)発行日　平成23年1月5日(2011.1.5)　(24)登録日　平成22年10月15日(2010.10.15)

(51)Int.Cl.　F I
G06F 9/52 (2006.01)　G06F 9/46 475A

請求項の数 24 (全 16 頁)

(21)出願番号	特願2000-582883 (P2000-582883)
(86)(22)出願日	平成11年11月12日 (1999.11.12)
(65)公表番号	特表2002-530735 (P2002-530735A)
(43)公表日	平成14年9月17日 (2002.9.17)
(86)国際出願番号	PCT/SE1999/002061
(87)国際公開番号	WO2000/029940
(87)国際公開日	平成12年5月25日 (2000.5.25)
審査請求日	平成18年10月26日 (2006.10.26)
(31)優先権主張番号	9803901-9
(32)優先日	平成10年11月16日 (1998.11.16)
(33)優先権主張国	スウェーデン(SE)
(31)優先権主張番号	9901145-4
(32)優先日	平成11年3月29日 (1999.3.29)
(33)優先権主張国	スウェーデン(SE)

(73)特許権者　598036300
テレフオンアクチーボラゲット　エル　エム　エリクソン(パブル)
スウェーデン国　ストックホルム　エス－１６４　８３
(74)代理人　100066692
弁理士　浅村　皓
(74)代理人　100072040
弁理士　浅村　肇
(74)代理人　100091339
弁理士　清水　邦明
(74)代理人　100094673
弁理士　林　鈺三

最終頁に続く

(54)【発明の名称】ジョブ信号を処理する多数の処理ユニットを有する処理システムにおけるジョブ信号処理方法および処理システム

(57)【特許請求の範囲】

【請求項１】

ジョブ信号を記憶するジョブ信号キュー（３２）および前記ジョブ信号キュー内のジョブ信号を並列に処理する多数の処理装置（３４Ａ－Ｄ）を有する処理システム（１０）におけるジョブ信号処理方法であって，各ジョブ信号は，対応するジョブを定義する命令のストリームへのポインターおよび該命令の実行に必要なオペランドを含み，前記処理システムはジョブの投機的実行に適合しており，前記ジョブ信号処理方法は，

コミット順に従って一時に１ジョブ信号づつ，前記ジョブ信号キュー内のジョブ信号にコミット優先度を連続的に割り当てるステップであり，コミット優先度を有するジョブ信号に対応するジョブは選択された処理装置により非投機的に実行されて該ジョブがコミットされるようにし，一方，コミット優先度を有さないジョブ信号に対応するジョブは投機的に実行され，もしデータ依存性が検出されるならフラッシュされる，ステップと，

コミット優先度を有するジョブ信号が割り当てられていない少なくとも１つの前記処理装置（３４）に対して，コミット優先度を待つ間，

（１）前記処理装置に第１のジョブ信号を割り当て，対応する第１のジョブを投機的に実行するステップと，

（２）もう１つのジョブ信号を前記処理装置に割り当て，対応するもう１つのジョブの投機的実行に備えるステップと，

（３）前記処理装置が前記第１のジョブの実行を完了した時，前記もう１つのジョブの投機的実行を開始するステップと，

を実行するステップとを含むことを特徴とする前記ジョブ信号処理方法。
【請求項２】
請求項１記載のジョブ信号処理方法であって，さらに，各ジョブ信号に，ジョブ信号に対して一意的な識別子（U_L＆D_L；ＪＱＰ）を関連づけるステップを含む方法。
【請求項３】
請求項２記載のジョブ信号処理方法であって，さらに，
各投機的に実行されたジョブについて，ジョブの結果を関連する識別子により検索できるように一時的に記憶するステップと，
コミット優先度を有するジョブ信号について，対応するジョブの一時的に記憶された結果を関連する識別子により検索して，検索した結果のメモリシステム（３７）へのライトバックを実施するステップと，
を含む方法。
【請求項４】
請求項３記載のジョブ信号処理方法であって，さらに，
投機的ジョブについてコミット優先度を有するジョブとの依存性が検出される時に，前記投機的ジョブの一時的に記憶された結果を関連する識別子により識別するステップと，
前記識別された結果を無効として，前記投機的ジョブがフラッシュされるステップと，
を含む方法。
【請求項５】
請求項４記載のジョブ信号処理方法であって，さらに，前記フラッシュされたジョブの後ろに置かれる全ての投機的ジョブをコミット順にフラッシュするステップを含む方法。
【請求項６】
請求項４記載のジョブ信号処理方法であって，さらに，
フラッシュされたジョブに関連づけられた識別子をクリアするステップと，
フラッシュされたジョブに対応するジョブ信号を処理装置に再度割り当てて実行するステップと，
前記再度割り当てたジョブに新しい識別子を関連付けるステップと，
を含む方法。
【請求項７】
請求項２記載のジョブ信号処理方法であって，前記識別子はジョブ信号キュー内のジョブ信号へのポインター（ＪＱＰ）である方法。
【請求項８】
請求項２記載のジョブ信号処理方法であって，前記識別子は前記ジョブ信号が割り当てられる処理装置のアイデンティティを表わす装置ラベル（U_L）および前記処理装置内のジョブ信号が与えられるアイデンティティを表わすジョブ信号識別ラベル（D_L）を含む方法。
【請求項９】
請求項８記載のジョブ信号処理方法であって，さらに，
投機的に実行した各ジョブについて，ジョブの結果を関連するジョブ信号識別ラベル（D_L）によりもしくはジョブ信号識別ラベル（D_L）と共に関連する装置ラベル（U_L）により検索できるように一時的に記憶するステップと，
コミット優先度を有するジョブ信号について，対応するジョブの一時的に記憶された結果を関連するジョブ信号識別ラベルによりもしくはジョブ信号識別ラベルと共に関連する装置ラベルにより検索して，検索した結果のメモリシステム（３７）へのライトバックを実施するステップと，
を含む方法。
【請求項１０】
請求項８記載のジョブ信号処理方法であって，さらに，フラッシュされている投機的に実行されたたジョブを関連する装置ラベルにより識別された処理装置に再度割り当ててジョブの実行を再開するステップを含む方法。

【請求項１１】
処理システム（１０）であって，
ジョブ信号を記憶するジョブ信号キュー（３２）と，
前記ジョブ信号キュー内のジョブ信号を並列に処理する多数の処理装置（３４Ａ－Ｄ）であって，各ジョブ信号は，対応するジョブを定義する命令のストリームへのポインターおよび該命令の実行に必要なオペランドを含み，前記処理システムはジョブの投機的実行に適合している，前記処理装置と，
コミット順に従って一時に１ジョブ信号づつ，前記ジョブ信号キュー内のジョブ信号にコミット優先度を連続的に割り当てる手段であり，コミット優先度を有するジョブ信号に対応するジョブは選択された処理装置により非投機的に実行されて該ジョブがコミットされるようにし，一方，コミット優先度を有さないジョブ信号に対応するジョブは投機的に実行され，もしデータ依存性が検出されるならフラッシュされる，前記手段と，
コミット優先度を有するジョブ信号が割り当てられていない少なくとも１つの前記処理装置（３４）に対して次の操作手順：
前記処理装置に第１のジョブ信号を割り当て，対応する第１のジョブを投機的に実行し，
もう１つのジョブ信号を前記処理装置に割り当て，対応するもう１つのジョブの投機的実行に備え，
前記処理装置が前記第１のジョブの実行を完了した時，前記もう１つのジョブの投機的実行を開始する，
を実行する手段とを含むことを特徴とした前記システム。
【請求項１２】
請求項１１記載の処理システムであって，さらに，各ジョブ信号にジョブ信号に対して一意的な識別子を関連付ける手段を含む処理システム。
【請求項１３】
請求項１２記載の処理システムであって，さらに，
各投機的に実行されたジョブについて，ジョブの結果を関連する識別子により検索できるように一時的に記憶する手段（３６）と，
コミット優先度を有するジョブ信号について，対応するジョブの一時的に記憶された結果を関連する識別子により検索して，検索した結果のメモリシステム（３７）へのライトバックを実施する手段と，
を含む処理システム。
【請求項１４】
請求項１３記載の処理システムであって，さらに，
コミット優先度を有するジョブとのデータ依存性が検出される時に投機的ジョブを中断する手段と，
前記中断した投機的ジョブの一時的に記憶された結果を関連する識別子により識別する手段と，
前記識別した結果を無効として，データ依存性を保持する前記ジョブがフラッシュされる手段と，
を含む処理システム。
【請求項１５】
請求項１４記載の処理システムであって，さらに，前記フラッシュしたジョブの後に置かれた全ての投機的ジョブをコミット順でフラッシュする手段を含む処理システム。
【請求項１６】
請求項１４記載の処理システムであって，さらに，
フラッシュしたジョブに関連する識別子をクリアする手段と，
フラッシュしたジョブに対応するジョブ信号を処理装置に再度割り当てて実行する手段（３２）と，
前記再度割り当てたジョブ信号を新しい識別子に関連づける手段と，

を含む処理システム。
【請求項１７】
請求項１３もしくは１４記載の処理システムであって，前記識別子は前記記憶手段内のジョブ信号へのポインター（ＪＱＰ）である処理システム。
【請求項１８】
請求項１２記載の処理システムであって，前記識別子は前記ジョブ信号が割り当てられる処理装置のアイデンティティを表わす装置ラベル（U_L）および前記処理装置内のジョブ信号が与えられるアイデンティティを表わすジョブ信号識別ラベル（D_L）を含む処理システム。
【請求項１９】
請求項１８記載の処理システムであって，さらに，
投機的に実行された各ジョブについて，ジョブの結果を関連するジョブ信号識別ラベルによりもしくはジョブ信号識別ラベルと共に関連する装置ラベルにより検索できるように一時的に記憶する手段（３６）と，
コミット優先度を有するジョブ信号について，対応するジョブの一時的に記憶された結果を関連するジョブ信号識別ラベルによりもしくはジョブ信号識別ラベルと共に関連する装置ラベルにより検索して，検索した結果のメモリシステムへのライトバックを実施する手段と，
を含む処理システム。
【請求項２０】
請求項１８記載の処理システムであって，さらに，
処理装置の投機的に実行された各ジョブについて，ジョブの結果を対応するジョブ信号に関連するジョブ信号識別ラベル（D_L）により検索できるように一時的に記憶する，各処理装置に１つずつ専用とされたいくつかのライトキュー（ＷＱ１－ＷＱ４）と，
コミット優先度を有するジョブ信号について，ジョブ信号に関連する装置ラベル（U_L）に基づいてどの処理装置にジョブ信号が割り当てられるかを決定する手段と，
決定した処理装置に専用のライトキュー（ＷＱ）から対応するジョブの結果を検索する手段と，
検索した結果のメモリシステム（３７）へのライトバックを実施する手段と，
を含む処理システム。
【請求項２１】
処理装置に割り当てられたジョブ信号の並列処理のための多数の処理装置（３４Ａ－Ｄ）を有する処理システム（１０）のための，物理的メモリで実現されたジョブ信号キュー（３２）であって，各ジョブ信号は，対応するジョブを定義する命令のストリームへのポインターおよび該命令の実行に必要なオペランドを含み，前記処理システムは対応するジョブの投機的実行に適合しており，ジョブ信号は連続的にコミット優先度を割り当てられ，コミット優先度を有するジョブ信号に対応するジョブは選択された処理装置による非投機的実行が可能にされて該ジョブがコミットされるようにし，一方，コミット優先度を有さないジョブ信号に対応するジョブは投機的に実行され，もしデータ依存性が検出されるならフラッシュされ，前記ジョブ信号キューは多くの記憶位置を有しており，各記憶位置は，
ジョブ信号を記憶する信号フィールド（ＳＩＧＮＡＬ）と，
前記ジョブ信号が割り当てられる処理装置のアイデンティティを表わす第１のラベルを記憶する第１の識別フィールド（U_L）と，
前記第１のラベルにより識別される処理装置内でジョブ信号が与えられるアイデンティティを表わす第２のラベルを記憶する第２の識別フィールド（D_L）であり，コミット優先度を有するジョブ信号が割り当てられるのを待っている処理装置において，それぞれの第２ラベルにより識別された，複数のジョブの投機的実行を可能にする，前記第２の識別フィールドと，
を含むことを特徴としたジョブ信号キュー。

【請求項２２】
請求項２１記載のジョブ信号キューであって，前記第１の識別フィールド（U_L）および前記第２の識別フィールド（D_L）は単一のフィールド（U_L＆D_L）へ連接されるジョブ信号キュー。
【請求項２３】
請求項２１記載のジョブ信号キューであって，各記憶位置はさらに前記信号フィールド内に記憶されたジョブ信号に対応するジョブが完了まで投機的に実行されているかどうかを示すフラグを記憶するフィールド（Ｆ）を含むジョブ信号キュー。
【請求項２４】
請求項２１記載のジョブ信号キューであって，各記憶位置はさらに，
信号フィールドが有効なジョブ信号を含むかどうかを示すフラグを記憶する有効フィールド（Ｖ）と，
前記信号フィールド内に記憶されたジョブ信号が処理装置に割り当てられているかどうかを示すフラグを記憶するフィールド（Ｔ）と，
を含むジョブ信号キュー。
【発明の詳細な説明】
【０００１】
（発明の技術的分野）
本発明は一般的に処理システムに関し，特に多数の処理装置を有する処理システム，およびこのような処理システムにおけるジョブ信号の処理方法に関する。
【０００２】
（発明の背景）
Telefonaktiebolaget LM Ericssonからの既知のデジタル交換システムにおけるＡＰＺ処理システム等の多くの従来の中央処理システムは，ＡＸＥシステム内の実行パイプラインと呼ばれる，単一処理装置の周りに形成されている。しかしながら，単一処理装置に基づく中央処理システムには容量に関する制約がある。
【０００３】
処理能力を高める１つの方法は処理システムをマルチプロセッサシステム，すなわち並列に動作する多数のプロセッサすなわち処理装置，として形成することである。従来のマルチ処理システムでは，各処理装置が入力信号を処理して対応する一連の命令を実行し，一時に１つの入力信号が各処理装置に割り当てられる。
【０００４】
近隣命令間で見つかる細粒並列度を精査するいわゆるスーパースカラープロセッサでは，プロセッサ内の機能的ユニットはいくつかの命令を並列に同時に実行するようにされている。
【０００５】
しかしながら，まだ，より効率的な処理システムに対する全般的な要求がある。
【０００６】
（発明の概要）
本発明は従来技術のマルチ処理システムのさらなる進展を構成する。
【０００７】
本発明はジョブの投機的（speculative）実行を行うようにされたマルチ処理システムに向けられている。マルチ処理システムでは，処理装置は異なるジョブを並列に独立して実行する。しかしながら，投機的実行を行うようにされた処理システム内でジョブを並列に実行している間は常に，１つの処理装置しかコミット優先度を持たずその現在のジョブを非投機的に実行して，メモリシステムへのライトバックを実施して信号送出をコミットすることができるのはそれだけである。他の処理装置におけるジョブは投機的に実施され依存性が検出される場合にはフラッシュされることがある。
【０００８】
このような処理システムの開発中に，特有の問題に遭遇した。コミット優先度を有する

ジョブが他の投機的ジョブよりも実行時間が長ければ，投機的に実行する処理装置はコミットジョブが完了するよりも遥かに前に割り当てられたジョブの実行を完了している。したがって，投機的に実行されたジョブはコミットジョブが完了まで実行されてその１つがコミット優先度を得られるまで待機しなければならない。それは一般的に，投機的に実行する処理装置がコミット優先度を単純に待機して貴重な実行時間が浪費され，処理システムの性能がひどく劣化することを意味する。

【０００９】

したがって，より柔軟で効率的な投機的マルチ処理システムおよびこのようなマルチ処理システムにおけるジョブ信号のより効率的な処理方法を提供することが本発明の一般的な目的である。

【００１０】

マルチ処理システムに使用するジョブキューを提供することが本発明のもう１つの目的である。

【００１１】

これらおよびその他の目的は添付する特許請求の範囲に明記された本発明により満たされる。

【００１２】

本発明に従った一般的なアイデアは，マルチ処理システム内の少なくとも１つの処理装置に対して，第１のジョブ信号を対応する第１のジョブを投機的に実行する処理装置へ割り当て，もう１つのジョブ信号を対応するもう１つのジョブを投機的に実行する処理装置へ割り当て，処理装置が第１のジョブの投機的実行を完了している時に前記もう１つのジョブの投機的実行を開始することに基づいている。所望により，より多くのジョブ信号を処理装置へ割り当てて投機的実行を行うことができ，対応するジョブの実行は処理装置が前に割り当てられたジョブの投機的実行を完了するとすぐに開始される。コミット優先度を有する処理装置にはさらにもう１つのジョブ信号を割り当てることができ，その実行はコミットジョブが完了するとすぐに開始される。

【００１３】

多数のジョブ信号を処理装置により投機的に実行するために割り当てることにより，ジョブ間の実行時間のばらつきの影響が相殺もしくは少なくとも著しく低減され，処理装置はコミット優先度を待機しながら複数のジョブを投機的に実行できるため処理システムの全体性能が実質的に改善される。

【００１４】

一般的に，ジョブ信号を処理装置に割り当て，ジョブ信号を追跡しコミット優先度を処理するのに必要なプロトコルは適切な制御ソフトウェアもしくはハードウェアと組み合わせてジョブ信号キューにより管理される。

【００１５】

ジョブ信号，その対応するジョブもしくは処理システムの操作におけるその結果を識別するために，各ジョブ信号には識別子が関連付けられる。識別子はジョブ信号キュー内の対応するジョブ信号の記憶位置へのポインターの形とすることができる。あるいは，識別子はジョブ信号が割り当てられる処理装置のアイデンティティを表わす単一ラベル，および処理装置内のジョブ信号が与えられるアイデンティティを表わすジョブ信号識別ラベルを含む。

【００１６】

投機的実行を行うようにされた処理システムでは，投機的に実行されたジョブの結果はライトキュー構成内に一時的に格納されてコミットされるのを待つ。本発明に従って，投機的に実行されたジョブがコミット優先度を得るとそのジョブの結果が識別子により検索される。

【００１７】

本発明は下記の利点を提供する。

－　処理システムの全体性能が実質的に改善される。

－　柔軟で効率的な投機的実行が提供される。
－　ジョブ間の実行時間のばらつきの影響が相殺される。
【００１８】
本発明の実施例の下記の説明を読めば本発明により提供される他の利点も理解できるであろう。
【００１９】
（発明の実施例の詳細な説明）
図面全体をとおして，同じ参照文字は対応するすなわち同じ要素のために使用される。
【００２０】
図１は本発明に従ったジョブ信号処理方法の略フロー図である。ジョブ信号処理用の多数の処理装置を有する処理システムでは，ジョブ信号は対応するジョブを実行する処理装置に割り当てられる。一般的に，少なくとも１つの処理装置に対して下記のステップが実施される。ステップ１において，第１のジョブ信号が対応する第１のジョブの投機的実行を行う処理装置へ割り当てられる。コミット優先度を有する処理装置の場合には，ジョブは非投機的に実行される。ステップ２において，もう１つのジョブ信号が対応するもう１つのジョブの投機的実行を行う処理装置へ割り当てられる。ステップ３において，処理装置が第１のジョブの実行を完了しておれば，もう１つのジョブの投機的実行が開始される。もちろん，さらに多くのジョブ信号に対してステップ２および３を繰り返すことができる。
【００２１】
このようにして，処理装置はコミット優先度を待機しながら複数のジョブを投機的に実行することができる。理論上，同じ処理装置に割り当てられるジョブ数は制限されない。
【００２２】
各ジョブ信号はその対応するジョブおよび前記ジョブの結果に密接に相関されている。
【００２３】
以下に，投機的実行を行うようにされたマルチ処理システムの実現例を参照して本発明の説明を行う。しかしながら，それに限定されるものではない。
【００２４】
図２は本発明に従った処理システムの略図である。処理システム１０は基本的にジョブスケジューラ２０および処理コア３０を含んでいる。処理コア３０はジョブの投機的実行を行うようにされており，処理コア３０の処理装置３４Ａ－Ｄは好ましくは実行パイプライン等の特殊化されたハードウェアの形であるが，在庫からすぐ入手できる標準マイクロプロセッサも適している。
【００２５】
ジョブスケジューラ２０は処理コア３０からだけでなく外部装置からも，イベントを表わす，ジョブ信号を受信して処理コア３０による処理に対してジョブ信号をスケジュールする。特定タイプのジョブスケジューラの例はTelefonaktiebolaget LM ericssonからの既知のＡＸＥデジタル交換システムにおける信号処理装置（ＳＰＵ）である。
【００２６】
処理コア３０は基本的には，単純にジョブキューと呼ばれる，ジョブ信号キュー３２，複数の実行パイプライン３４Ａ－Ｄ，データ依存性を処理する依存性チェック装置と一時的ライトキュー３６の組合せ，およびプログラム記憶３８およびデータ記憶３９に分割されたメモリシステム３７を含んでいる。
【００２７】
ジョブスケジューラ２０からのジョブ信号はジョブキュー３２内にバッファされ，それはジョブ信号を記憶するためのいくつかの記憶位置を有する。各ジョブ信号は付加情報と共にジョブキュー３２の各記憶位置に記憶される。一般的に，各ジョブ信号はヘッダーおよびデータを含んでいる。管理情報の他に，ヘッダーは通常プログラム記憶３８内のソフトウェアコードへのポインターを含み，ジョブ信号のデータは対応するジョブを実行するのに必要な入力オペランドを含んでいる。このデータは地域プロセッサやもう１つのプロ

セッサ等の外部装置からの信号メッセージとすることができる。ジョブは信号ヘッダーにより指定される命令ストリームとして定義することができ，このジョブはジョブ信号の受信で開始してエンドジョブルーチンの呼出しにより終了する。しかしながら，ジョブ信号自体は通常いかなる命令も含まないことをお判り願いたい。ジョブ信号は通常プログラム記憶３８内に記憶されたソフトウェアコード内の命令へのポインター，および命令の実行に必要なオペランドを含み，一般的にジョブ信号を自立型とする。

【００２８】

好ましくは，実行パイプライン３４Ａ－Ｄはジョブキュー３２内の異なる記憶位置からジョブ信号を独立に“フェッチ”して異なるジョブを独立して並列に実行する。実行パイプラインが新しいジョブの実行を開始できる時は常に，ジョブキュー３２は未割当てジョブ信号を見つけるために調べられ，未割当てジョブ信号がパイプラインに割り当てられる。次に，ジョブ信号は実行パイプラインにおいて処理されて対応するジョブが実行される。この特定の例では，４つのパイプラインが４つの異なるジョブを，互いに独立して，並列に実行するように動作することができる。並列ジョブ実行中は常に，ジョブキュー３２内の１つのジョブ信号だけがコミット位置にあり，ジョブ信号が割り当てられる実行パイプラインは対応するジョブをコミットする，すなわちメモリシステム３７へのライトバックを実施して信号送出をコミットすることができる。他の実行パイプライン内のジョブは投機的に実行され，依存性チェック装置３６によりデータ依存性が検出される場合にはフラッシュすることができる。

【００２９】

情報フローがプロトコルにより支配されるシステムに対する一般的な要求条件はある関連イベントを受信順に処理しなければならないことである。それはシステムがどのように実現されようとも，システムによって変わることはない。ジョブ間のコミット順は処理コアへの到来により定義され，一般的には変更されない。しかしながら，異なる優先度レベルのジョブ信号を処理する処理システムでは，優先度の高いジョブ信号を優先度の低いジョブ信号の前に置くのが有用であることがある。

【００３０】

一般的に，各実行パイプラインはプログラム記憶から命令をフェッチし，命令を復号し，命令を実行してメモリライトバックを実施する回路を含んでいる。本発明により使用することができる特定の実行パイプラインはエリクソンＡＸＥデジタル交換システム内のパイプラインである。

【００３１】

依存性チェック装置３６は一般的に実行パイプライン３４Ａ－Ｄに関連する１つ以上のリードバッファを使用して実現される。パイプラインがデータ記憶３９からデータをフェッチすると，リードアドレスがリードバッファ内にバッファされる。コミット優先度を有する実行パイプラインがデータ記憶３９へのライトバックを実施すると，データ記憶へのライトアドレスがリードバッファ内のリードアドレスと比較されてジョブ間にデータ依存性があるか調べられる。ジョブを投機的に実行して読み出されたデータが引続きコミットされたジョブにより修正される場合には，データ依存性が存在し投機的に実行されたジョブはフラッシュして再開しなければならない。フラッシュしたジョブはジョブキュー３２から再開することができる。コミットされているジョブに対応するジョブ信号はジョブキュー３２から除去され，したがってジョブスケジューラ２０からの新しいジョブ信号をジョブキュー３２内にバッファーすることができる。

【００３２】

投機的に実行されたジョブにより提案されるデータ記憶修正は一時ライトキュー３６内にログインされるが，ジョブがコミット優先度を得るまではデータ記憶３９には書き込まれない。投機的ジョブがコミット優先度を得ると，問題とするジョブに属する一時的ライトキュー３６内へのエントリーは即座にデータ記憶３９に書き込まれる。しかしながら，実行中のコミットジョブは一般的にそのデータをライトキュー内には記憶せず，対応するメモリアドレスを依存性チェック装置へ転送して依存性チェックをできるようにするだけ

である。コミット優先度を有するジョブにより発生されたジョブ信号はジョブスケジューラ２０へ転送され，そこで後の処理のためにスケジュールされるかあるいは入出力装置（図示せず）へ送られて外部装置へ配送される。

【００３３】

特定タイプの依存性チェック装置および一時的ライトキューが国際特許出願ＷＯ８８／０２５１３に開示されている。

【００３４】

図２の処理システムの開発およびテスト中に，特有の問題に遭遇した。コミット優先度を有するジョブが投機的ジョブよりも長い実行時間を有する場合には，投機的実行処理装置はコミットジョブの完了よりも遥かに前にその割り当てられたジョブの実行を完了している。投機的に実行されたジョブはコミットジョブが完了まで実行され最後にコミットされて投機的に実行されたジョブがコミット優先度を得られるまで待機しなければならない。一般的に，それは投機的実行処理装置が単純にコミット優先度を待ち貴重な実行時間が浪費されて，処理システムの性能がひどく劣化することを意味する。電気通信応用に対するシミュレーションにより，４つの等価パイプラインのマルチパイプを有する処理システムにより僅か２つの等価シングルパイプ処理システムの容量しか得られず，その比は２：４で最適からは程遠いことが判った。

【００３５】

もちろん，前記した推論は他の投機的ジョブよりも長い実行時間を有する投機的ジョブについても言えることである。長い実行時間を有する投機的ジョブがコミット優先度を得ると，他の投機的ジョブは既に完了まで実行されており，コミット優先度を待っているだけである。

【００３６】

本発明に従って，多数のジョブ信号が少なくとも１つの実行パイプラインへ割り当てられ，前のジョブが完了まで実行されているとすぐに次のジョブの投機的実行が開始される。このようにして，実行パイプラインはコミット優先度を待機しながら複数のジョブを投機的に実行することができる。パイプライン当たりいくつかのジョブを実行することにより，ジョブ間の実行時間のばらつきの影響が相殺されあるいは少なくとも低減され，前記したシングルパイプ対マルチパイプ比が実質的に改善される。

【００３７】

一般的に，ジョブ信号を実行パイプライン３４Ａ－Ｄに割り当て，完了まで投機的に実行されているジョブの追跡を継続し，ジョブ信号に連続的にコミット優先度を割り当て，コミットされているジョブ信号を除去するのに必要なプロトコルは適切な制御ソフトウェアもしくはハードウェアと組み合わせたジョブ信号キュー３２により管理される。

【００３８】

ジョブキュー３２は通常はいくつかの記憶位置を有するオーディナリキューである。例として，ジョブキュー３２はメモリの予め定められた部分を，各々がそれ自体の個別のメモリアドレスを有する，いくつかの記憶位置へ論理的に分割することにより共通メモリ内に実現することができる。一般的に，ジョブキュー３２内の各記憶位置は，例えば下記の表Ｉに記載された，いくつかのフィールドへ分割される。

表Ｉ

フレーム名	定義
Valid	セットされると、記憶位置は有効なジョブ信号を含む
Taken	セットされると、ジョブ信号は実行パイプラインに割り当てられている
Finished	セットされると、対応するジョブが完了まで実行されているが、コミットされていない
U_L&D_L	U_Lはジョブ信号を処理するパイプのアイデンティティを表わし、 D_Lはパイプ内のジョブ信号が与えられるアイデンティティを表わす。フィールドは通常U_LおよびD_Lの連接である。
Signal	信号ヘッダーおよびデータ

【００３９】
フィールド‘Valid’は記憶位置が有効なジョブ信号を含むかどうかを示すのに使用される。ジョブ信号はスケジューリング装置２０から受信されると，ジョブキュー３２の最初の空き位置，すなわちValidフラグがセットされない位置に置かれる。次に，Validフラグは‘Signal’フィールドが現在有効なジョブ信号を含んでおり，位置は占有されていることを示すようにセットされる。
【００４０】
フィールド‘Taken’はジョブ信号がパイプラインに割り当てられているかどうかを示すのに使用される。ジョブキュー３２の記憶位置におけるジョブ信号が実行パイプラインに割り当てられる場合には，その位置に対するTakenフラグはジョブ信号が割り当てられていることを示すのに使用される。
【００４１】
フィールド‘Finished’はジョブが完了まで投機的に実行されていることを示すのに使用され，フィールド‘U_L&D_L’はパイプのアイデンティティおよびそのパイプ内でジョブ信号および／もしくは対応するジョブが与えられるアイデンティティを示すのに使用される。パイプラインが第１のジョブに対するエンド－ジョブルーチンを実行している場合には，Finishedフラグはジョブが完了まで実行されていてコミットされる準備が完了していることを知らせるようにセットされる。ここで，パイプラインは第２の投機的ジョブ信号をフェッチする準備が完了している。第１のジョブには問題とするパイプのジョブid‘１’が割り当てられ，そのパイプ内で実行される第２のジョブにはジョブid‘２’が割り当てられる。パイプがコミット優先度を受信する前に第２のジョブを終了する場合には，第３の投機的ジョブを実行することができ，以下同様である。一般的に，パイプ当たりのジョブ数と複雑さとの間でトレードオフがなされ，実際上パイプ当たりジョブ数は制限される。
【００４２】
投機的実行を行うようにされた処理システムでは，ジョブは一般的に次の状態とすることができる，Not_Started，StartedおよびFinished。これらの状態はジョブキュー３２内でさまざまな方法で符号化することができ，表ＩのフィールドValid，TakenおよびFinishedはこれらのジョブ状態を符号化する方法の一例にすぎないことをお判り願いたい。
【００４３】

好ましくは，ジョブキュー３２にはコミット位置にあるのはジョブキュー３２内のどの記憶位置であるかを指摘するポインターが関連付けられる。コミット位置にあるジョブ信号はコミット優先度を有し，このジョブ信号を処理する実行パイプラインはメモリシステム３７へのライトバックを実施してジョブスケジューラ２０にジョブ信号を送られるようにされる。

【００４４】

図３は第１の好ましい実施例に従ったジョブ信号キュー，処理装置およびライトキュー構成間の相互関係を示す略図である。この実施例では，並列に動作する４つの異なるパイプライン３４Ａ－Ｄがあるものと仮定する。各パイプラインには一意的な装置ラベル（１－４）が割り当てられる。また，各パイプラインは２つのジョブを処理できるものとする。したがって，明瞭かつ単純にするために，ジョブキュー３２は８つの記憶位置を有するものとして例示される。各記憶位置はフィールドV(Valid)，T(Taken)，F(Finished)，U_L(Pipe id)，D_L(Job id)およびSIGNALを含んでいる。図３において，フィールド‘U_L&D_L’は単一の連接フィールドではなく２つの別々のフィールドU_LおよびD_Lとして図示されている。両方を実現することができる。

【００４５】

ジョブ開始

次に，新しいジョブをどのように開始するかの例について説明する。最初に，実行準備完了している実行パイプラインがジョブキュー３２から新しいジョブ信号を要求する。好ましくは，パイプラインはパイプ内でどのアイデンティティに新しいジョブ信号を与えるべきかをも示す。次に，ジョブキュー３２が従来の手段によりトラバースされ，有効な未割当てジョブ信号を含む次の位置を見つけるためにジョブキュー３２内の記憶位置のValidフラグおよびTakenフラグが調べられる。未割当てジョブ信号の記憶位置へのポインターが要求している実行パイプラインのジョブキューレジスタ（ＪＱＰ）へ転送される。ポインターは記憶位置へのメモリアドレスの形とすることができる。問題とするパイプラインは次のジョブ信号へのポインターに対するそのＪＱＰレジスタを自動的に読み出し，続いてパイプライン内へのポインターにより与えられるジョブ信号をコピーして対応するジョブ実行開始する。ジョブキュー３２のその位置に対するTakenフラグがセットされ，パイプラインのアイデンティティおよびパイプ内でジョブが与えられるアイデンティティが，それぞれ，U_LおよびD_L内に書き込まれる。このようにして，ジョブキュー３２内に記憶されたジョブ信号は実行パイプライン３４Ａ－Ｄに割り当てられ，記憶位置の信号フィールドがセットされる。

【００４６】

図３の例では，ジョブキュー３２の各記憶位置が実行パイプライン３４Ａ－Ｄの１つに割り当てられている（Ｔ＝１）有効な（Ｖ＝１）ジョブ信号を含んでいる。最初の４つの位置のジョブ信号に対応するジョブは完了まで実行されており（Ｆ＝１），コミット優先度を待機している。最後の４つの位置のジョブ信号に対応するジョブは投機的に実行されている最中であり，どのジョブも終了していない（Ｆ＝０）。ジョブが完了まで投機的に実行されるとすぐに，対応するFinishedフィールドがセットされる。

【００４７】

投機的に実行されたジョブの結果はライトキュー構成３６にログインされるが，ジョブがコミットされるまではデータ記憶３９には書き込まれない。図３に例示したように，本発明の第１の好ましい実施例ではライトキュー構成３６は全ての実行パイプラインに共通するライトキューを含んでいる。次に，ライトキュー内の各エントリにはそれがどのパイプラインおよび表示されたパイプラインのどの投機的ジョブに属するかを示すpipe-idタグおよびjob-idタグが与えられる。pipe-idタグおよびjob-idタグは，それぞれ，U_LおよびD_Lに対応する。したがって，ライトキュー内の各エントリはデータ記憶３９へのアドレス（ＡＤＤＲ），関連するデータ（ＤＡＴＡ），pipe-idタグ（U_L），job-idタグ（D_L）および有効フラグ（Ｖ）を含んでいる。

【００４８】

図３のジョブキュー３２を指す矢符により示されるコミット位置におけるジョブ信号はコミット優先度を有し，このジョブ信号を処理する実行パイプラインはデータ記憶３９へ書き込めるようにされる。ジョブ信号がコミット位置へ移されると，その記憶位置の装置ラベルはジョブ信号に責任のあるパイプはどれであるかを確認するために調べられ，そのパイプはジョブに対する全てのライト操作および信号送出をコミット開始する。ジョブキュー３２のコミット位置に記憶された装置ラベルU_Lおよびジョブ信号識別ラベルD_Lを使用して，関連する結果をライトキューから検索してデータ記憶３９に書き込むことができる。これが行われると，Validフラグをクリアすることにより位置が開放され従来の手段によりジョブキュー３２がステップされて新しい記憶位置をコミット位置内に移し，ジョブキュー３２のジョブ信号に連続的にコミット優先度を割り当てる。スケジューリング装置２０はジョブが完了まで実行されておりジョブキュー３２がスケジューリング装置から新しいジョブ信号を受信する準備が完了していることを知らされる。

【００４９】

フラッシュ時のジョブ再開

次に，フラッシュをどのように処理するかの例について簡単に説明する。フラッシュ時に，データ依存性が検出されると，フラッシュされるジョブの実行は通常パイプライン内で割り込まれ，ジョブは再度Not-Started状態とされ，フラッシュされたジョブに属するライトキュー内のエントリは無効とされる。

【００５０】

より詳細には，依存性が検出されているライトキュー３６内のエントリのpipe-idタグ（U_L）はフラッシュされるジョブを処理するパイプラインを識別する。ジョブが実行中であれば，パイプラインは通常割込みを得る。識別された実行パイプラインのＪＱＰレジスタはジョブキュー３２内のジョブ信号の記憶位置へのポインターを保持する。装置ラベルU_Lおよびジョブキューポインターにより識別された記憶位置の識別ラベルD_Lを使用して，ライトキュー内の対応するエントリを無効とすることができる。ジョブキュー３２内の対応する記憶位置のFinishedフラグおよびTakenフラグが除去され，次にジョブを再開することができる。ジョブキュー３２内の当該記憶位置のフィールドU_LおよびD_Lがクリアされると，ジョブ信号は実行を再開できる任意の実行パイプラインへ割り当てることができ，新しい装置ラベルおよび新しいジョブ信号識別ラベルに関連付けることができる。あるいは，フラッシュされたジョブに対応するジョブ信号がジョブ信号に関連づけられた装置ラベルにより識別されるパイプラインへ再度割り当てられ，ジョブ信号は新しいジョブ信号識別ラベルD_Lに関連づけられる。

【００５１】

フラッシュされるジョブは実行中であれば必ずしも割り込まれる必要はない。しかしながら，性能的にはジョブを割り込むほうが有利である。

【００５２】

図４は第１の好ましい実施例に従ったジョブ信号キュー，処理装置およびライトキュー構成間の相互関係を示す略図である。ライトキュー構成３６を除けば，図４は図３と同じである。図４において，ライトキュー構成３６は各実行パイプライン３４Ａ－Ｄについて１つずつのいくつかの独立したライトキューＷＱ１－ＷＱ４を含んでいる。各パイプラインに対する専用ライトキューを有することにより，各ライトキュー内の各エントリにpipe-idタグおよびjob-idタグの替わりにjob-idタグを与えれば十分である。したがって，ライトキューＷＱ１－ＷＱ４内の各エントリはデータ記憶３９へのアドレス（Ａ），関連するデータ（ＤＡＴＡ），job-idタグ（D_L）および有効フラグ（Ｖ）を含んでいる。ジョブキュー３２内のジョブ信号がコミット位置内へ移されると，そのジョブ信号に責任のあるパイプラインはどであるかを確認するためにその記憶位置の装置ラベルU_Lが調べられる。同時に，対応するライトキューも識別される。コミット位置に記憶されたジョブ信号識別ラベルD_Lを使用することにより，コミットされるジョブの結果を当該ライトキューから検索してデータ記憶３９へ書き込むことができる。それに伴って，ジョブのフラッシュ時に，ジョブを処理するパイプライン専用のライトキューを関連する装置ラベルU_Lに

より識別することができる。続いて，関連する識別ラベルD_Lを使用することにより，ライトキュー内の当該エントリが識別されその中の結果が無効とされる。

【００５３】

図４の実現において，フラッシュされたジョブの後にコミット順で置かれる全ての投機的ジョブも，好ましくはフラッシュされてライトキュー３６内にスペースを作り出す。その理由はフラッシュされたジョブの新しい実行，すなわちもう１つのジョブの実行，にはフラッシュされたジョブよりも多くのスペースをライトキュー内に必要とすることであるためである。したがって，フラッシュされたジョブのジョブキュー内の位置は問題とするパイプラインのＪＱＰレジスタ内のポインターを調べて確認される。次に，フラッシュされたジョブよりも“若い”全ての投機的ジョブを見つけるためにジョブキュー３２がトラバースされる。見つけられた各ジョブについて，ジョブに関連する装置ラベルU_Lおよびジョブ信号識別ラベルD_Lを使用することにより，ジョブを処理するパイプラインを見つけて実行を中断することができ，当該ライトキュー内のフラッシュされたジョブに属するこれらのエントリを無効とすることができる。

【００５４】

ラベルをジョブ信号に関連づけ，ジョブキューフィールド内に情報を書き込みジョブ信号に連続的にコミット優先度を割り当てる等のジョブキューを処理する機能は，好ましくは，例えばマイクロコード命令プログラムもしくはアセンブラープログラム等の形のソフトウェアにより制御される。フラッシュを処理するおよびライトキューを制御する等の処理システムの他の機能もソフトウェアで実現することができる。もちろん，前記した機能はハードウェアで実現することもできる。

【００５５】

図５は第３の好ましい実施例に従ったジョブ信号キュー，処理装置およびライトキュー構成間の相互関係を示す略図である。ジョブキューおよびライトキュー構成を除けば，図５は図３と同様である。図５では，ジョブキュー３２内の記憶位置には一般的にＵＬおよびＤＬ等の識別フィールドはない。替わりにジョブキューポインター（ＪＱＰ）が各ジョブ信号に関連する唯一の識別子として使用される。ジョブのコミットおよびフラッシュにおいて，ジョブに責任があるのはどのパイプラインであるかを識別するためにジョブキューポインターが探索手順において使用される。ライトキュー内の各エントリはデータ記憶アドレス（ＡＤＤＲ），関連データ（ＤＡＴＡ），ジョブキューポインタータグ（ＪＱＰ）および有効フラグ（Ｖ）を含む。コミットにおいて，ジョブの結果をジョブキューポインタータグにより検索することができる。フラッシュにおいて，ジョブの結果をジョブキューポインタータグにより識別して無効とすることができる。

【００５６】

しかしながら，好ましくは，ジョブに責任があるのはどのパイプラインであるかをより容易に見つけるために，図５のライトキューには装置ラベルタグ用フィールドが与えられる。

【００５７】

処理装置にジョブ信号を割当て，ジョブ信号の追跡を続けコミット優先度を管理するプロトコルがジョブ信号キューおよびその関連する制御機能により実施される観点から本発明を説明してきたが，このプロトコルを実行パイプライン側から管理することもできることをお判り願いたい。

【００５８】

“処理装置”という用語は特殊化されたハードウェアに限定はされず，在庫からすぐ入手できる標準マイクロプロセッサ等の他のタイプの処理装置も含まれることをお判り願いたい。標準マイクロプロセッサの場合には，依存性チェックは通常アプリケーションソフトウェア内でリードおよびライト命令に特別なコードを取り付けることにより実現される。好ましくは，フラッシングのサポートはアプリケーションソフトウェア内で実現されオペレーティングシステムもしくは仮想機械により実施される。シングルプロセッサシステム用に書かれたアプリケーションソフトウェアがマイグレートされ標準マイクロプロセッ

サ環境で再使用される場合には，アプリケーションソフトウェアを再コンパイル等により変換することができる。例えば，依存性チェックコードおよび修正された変数のコピーを記憶してジョブの適切なフラッシュもしくはロールバックを可能とするコードを含む，投機的実行をサポートする適切なコードを付加し，次にソフトウェアを再コンパイルすることにより逐次プログラムされたアプリケーションソフトウェアを自動的に変換することができる。標準マイクロプロセッサのオペレーティングシステムや仮想機械も投機的実行をサポートするように修正することができる。例えば，依存性チェックコードを実行している時に依存性が検出されたら，当該ジョブをフラッシュするオペレーティングシステム／仮想機械に制御を転送することができる。

【００５９】

前記した実施例は単なる例にすぎず，本発明はそれに限定されるものではないことをお判り願いたい。ここに開示され特許請求された基本的原理を保有する修正，変更および改良は本発明の範囲および精神に含まれるものとする。

【図面の簡単な説明】

【図１】　本発明に従ってジョブ信号を処理する方法の略フロー図である。

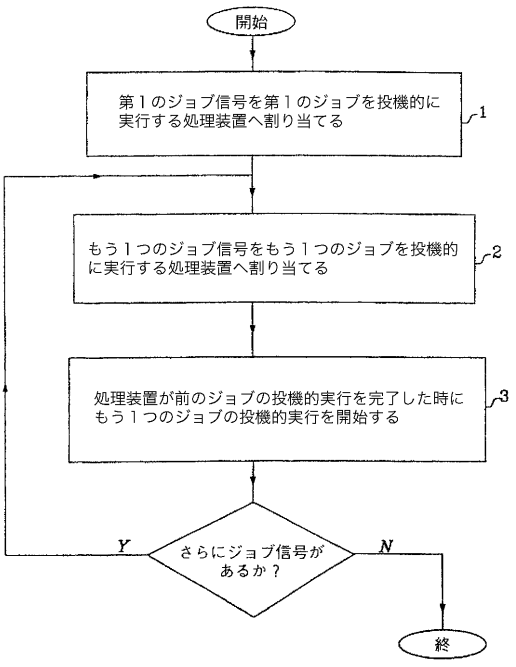
【図２】　本発明に従った処理システムの略図である。

【図３】　第１の好ましい実施例に従ったジョブ信号キュー，処理装置およびライトキュー構成間の相互関係を示す略図である。

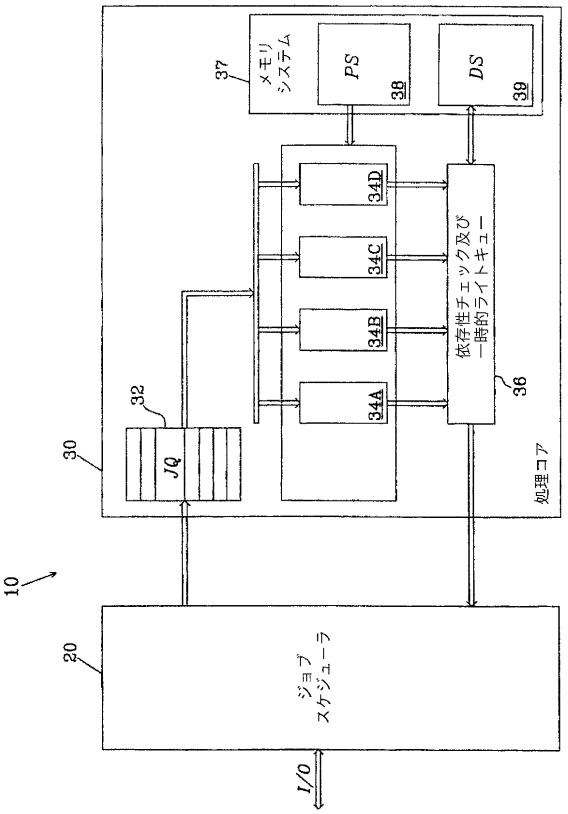
【図４】　第２の好ましい実施例に従ったジョブ信号キュー，処理装置およびライトキュー構成間の相互関係を示す略図である。

【図５】　第３の好ましい実施例に従ったジョブ信号キュー，処理装置およびライトキュー構成間の相互関係を示す略図である。

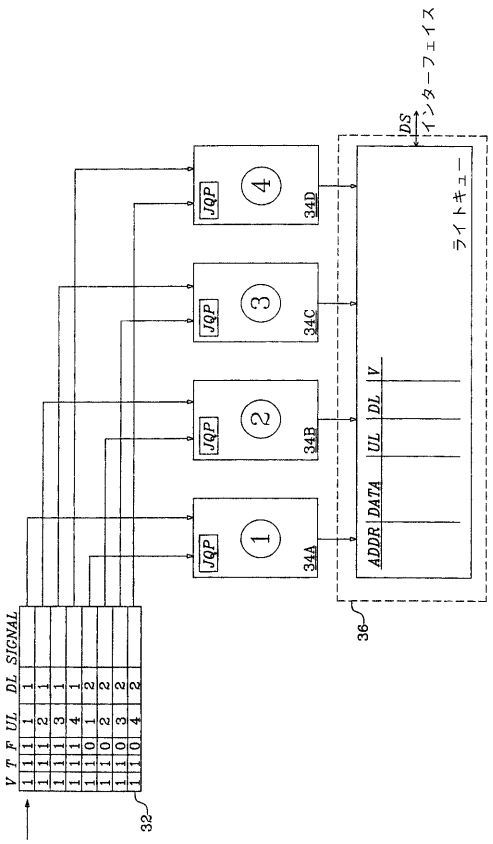
【図１】



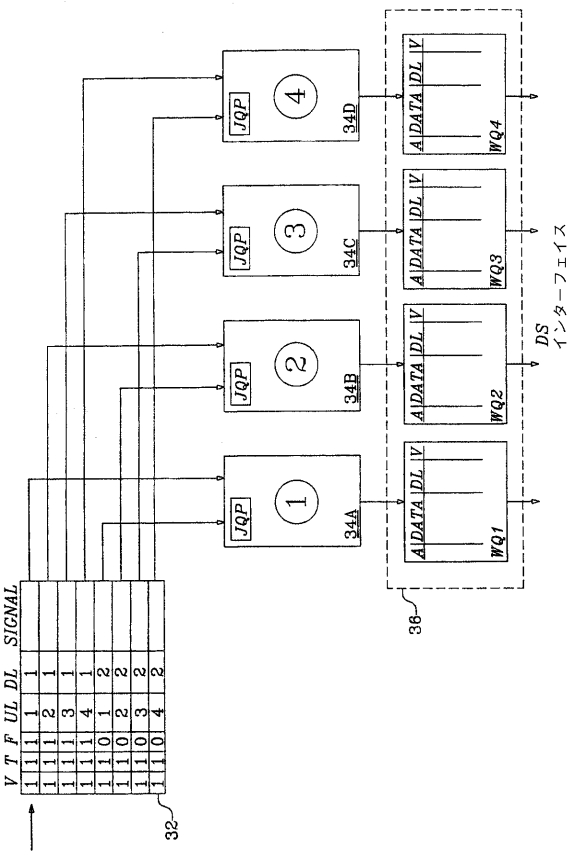
【図２】



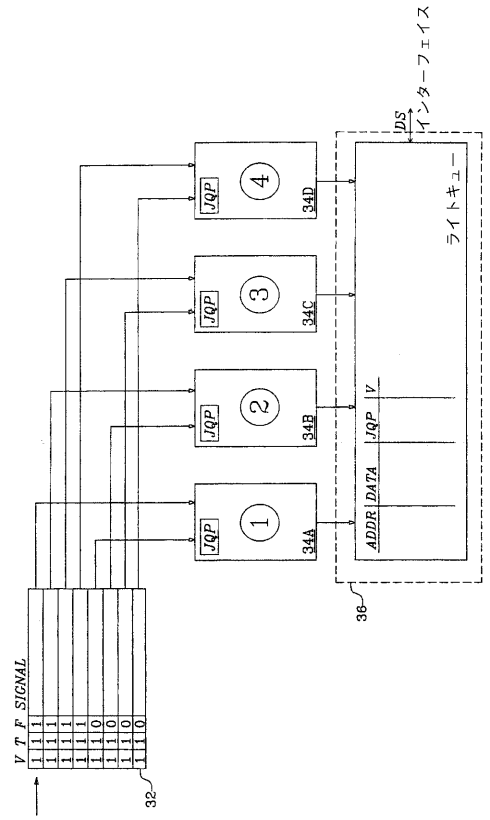
【図３】


V T F UL DL SIGNAL
JQP
ADDR DATA UL DL V
ライトキュー
DS
インターフェイス
32
34A
34B
34C
34D
36

【図４】


V T F UL DL SIGNAL
JQP
A DATA DL V
WQ1
WQ2
WQ3
WQ4
DS
インターフェイス
32
34A
34B
34C
34D
36

【図５】


V T F SIGNAL
JQP
ADDR DATA JQP V
ライトキュー
DS
インターフェイス
32
34A
34B
34C
34D
36

フロントページの続き

(72)発明者　ホルムベルク、ペル、アンデルス
スウェーデン国　ストックホルム、フリントバケン　１８
(72)発明者　リンネルマルク、ニルス、オラ
スウェーデン国　ハニンゲ、トウブベーゲン　１１、３　トル
(72)発明者　ストロムベルグソン、カルル、オスカル、ヨアヒム
スウェーデン国　モルンダル、アクスガタン　６
(72)発明者　エゲランド、テルイエ
スウェーデン国　バルムド、ブヨルクビクス　アレベグ　２
(72)発明者　カルルソン、マグヌス
スウェーデン国　カレレド、リラ　リベレドスベーゲン　１１

審査官　漆原　孝治

(56)参考文献　Morry Katz, PARATRAN: A TRANSPARENT, TRANSACTION BASED RUNTIME MECHANISM FOR PARALLEL EXECUTION OF SCHEME, 1986 S.M. Thesis, 米国, Massachusetts Institute of Technology, Department of Electrical Engineering & Computer Science, １９８９年　７月, pp. 4 - 6, 11 - 34, ［平成２１年２月２６日検索］, ＵＲＬ, http://publications.csail.mit.edu/lcs/pubs/ps/MIT-LCS-TR-454.ps
James S. Mattson Jr., An Effective Speculative Evaluation Technique for Parallel Supercombinator Graph Reduction, A dissertation submitted in partial satisfaction of the requirements for the degree Doctor of Philosophy in Computer Science, 米国, UNIVERSITY OF CALIFORNIA, SAN DIEGO, １９９３年, pp. 1 - 17, 39 - 82, 206 - 211, ［平成２１年２月２６日検索］, ＵＲＬ, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.56.8221

(58)調査した分野(Int.Cl., ＤＢ名)
G06F　9/46 - 9/54