(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号

特許第5078366号
(P5078366)

(45)発行日　平成24年11月21日(2012.11.21)　(24)登録日　平成24年9月7日(2012.9.7)

(51)Int.Cl.		F I		
G06F 13/00	(2006.01)	G06F	13/00	301J
G06F 13/38	(2006.01)	G06F	13/38	350
G06F 11/30	(2006.01)	G06F	11/30	K

請求項の数 15　(全 25 頁)

(21)出願番号	特願2007-7741(P2007-7741)
(22)出願日	平成19年1月17日(2007.1.17)
(65)公開番号	特開2007-200319(P2007-200319A)
(43)公開日	平成19年8月9日(2007.8.9)
審査請求日	平成21年10月30日(2009.10.30)
(31)優先権主張番号	11/340,447
(32)優先日	平成18年1月26日(2006.1.26)
(33)優先権主張国	米国(US)

(73)特許権者　390009531
インターナショナル・ビジネス・マシーンズ・コーポレーション
ＩＮＴＥＲＮＡＴＩＯＮＡＬ　ＢＵＳＩＮＥＳＳ　ＭＡＳＣＨＩＮＥＳ　ＣＯＲＰＯＲＡＴＩＯＮ
アメリカ合衆国１０５０４　ニューヨーク州　アーモンク　ニュー　オーチャード　ロード
(74)代理人　100108501
弁理士　上野　剛史
(74)代理人　100112690
弁理士　太佐　種一
(74)代理人　100091568
弁理士　市位　嘉宏

最終頁に続く

(54)【発明の名称】マルチホスト環境における共用の入出力ファブリックのエラー・メッセージをマスタ制御ルート・ノードに経路指定する方法、装置およびプログラム

(57)【特許請求の範囲】

【請求項１】

複数のホスト・コンピュータ・システムを含むデータ処理環境においてコンピュータ実施される方法であって、

前記複数のホスト・コンピュータ・システムは複数の入出力アダプタにスイッチドファブリックを使用して接続され、

前記スイッチドファブリックは、前記複数のホスト・コンピュータ・システムのうちの、エラー・メッセージによって識別されるエラーにより影響を受けるホスト・コンピュータ・システムのみに前記エラー・メッセージを経路指定し、

前記方法は、

前記複数の入出力アダプタのうちの１つが、前記複数の入出力アダプタのうちの当該１つにおけるエラーを検出するステップと、

前記複数の入出力アダプタのうちの前記１つが、前記エラーについての情報を含み且つ前記複数の入出力アダプタのうちの前記１つを識別する識別子を含むエラー・メッセージを生成するステップと、

前記スイッチドファブリックにおけるハードウェア・スイッチに保存された経路指定テーブルを使用して、前記エラーにより影響を受けるホスト・コンピュータ・システムを識別するステップと、

前記ハードウェア・スイッチが、前記エラー・メッセージを、前記識別されたホスト・コンピュータ・システムのみに経路指定するステップと

を含む、前記方法。
【請求項２】
前記エラーにより影響を受けるホスト・コンピュータ・システムを識別する単一のマスタ制御ノードに前記エラー・メッセージを経路指定するステップをさらに含む、請求項１に記載の方法。
【請求項３】
前記エラー・メッセージを単一のマスタ制御ノードに経路指定するステップと、
前記エラーが解消されないうちは、前記識別されたホスト・コンピュータ・システムのすべてが前記エラー・メッセージの受信を確認するまで、前記マスタ制御ノードを待機させるステップと、
前記エラーが解消されるまで、前記複数の入出力アダプタのうちの前記１つへの伝送トラフィックを中断するステップと
をさらに含む、請求項１に記載の方法。
【請求項４】
前記識別されたホスト・コンピュータ・システムの第１のホスト・コンピュータ・システムが前記エラー・メッセージを受信するステップと、
前記第１のホスト・コンピュータ・システムが前記エラー・メッセージの受信を確認するステップと、
前記第１のホスト・コンピュータ・システムが当該第１のホスト・コンピュータ・システムから前記複数の入出力アダプタのうちの前記１つへの伝送トラフィックを中断するステップと、
前記エラーが解消されるまで前記第１のホスト・コンピュータ・システムが待機し、しかる後前記複数の入出力アダプタのうちの前記１つへの伝送トラフィックを再開するステップと
をさらに含む、請求項１に記載の方法。
【請求項５】
前記ハードウェア・スイッチはレジスタを備えており、
前記方法は、
前記レジスタにおける個々のビットを前記複数のホスト・コンピュータ・システムの各々に割り当てるステップと、
前記エラー・メッセージに応答して、前記複数の入出力アダプタのうちの前記１つへの伝送トラフィックを中断するステップと、
前記第１のホスト・コンピュータ・システムが前記エラー・メッセージを受信するステップと、
前記第１のホスト・コンピュータ・システムに割り当てられているビットをセットすることによって、前記第１のホスト・コンピュータ・システムが前記エラー・メッセージの受信を確認するステップと、
前記識別されたホスト・コンピュータ・システムのすべてが前記エラー・メッセージの受信を確認したかどうかを決定するために、マスタ制御ノードが前記レジスタをポーリングするステップと、
前記識別されたホスト・コンピュータ・システムのすべてが前記エラー・メッセージの受信を確認したという前記マスタ制御ノードによる決定に応答して、前記識別されたホスト・コンピュータ・システムが前記複数の入出力アダプタのうちの前記１つへの伝送トラフィックを再開することを可能にするために前記エラーを解消するステップと、
前記識別されたホスト・コンピュータ・システムのすべてが前記エラー・メッセージの受信を確認したのではないという前記マスタ制御ノードによる決定に応答して、前記エラーの解消を待つステップであって、前記エラーの解消を待つ間、前記複数の入出力アダプタのうちの前記１つへのトラフィックが中断されたままである、前記待つステップと
をさらに含む、請求項１に記載の方法。
【請求項６】

前記マスタ制御ノードが前記スイッチドファブリックを全探索して、前記ハードウェア・スイッチのすべての識別情報、ならびに、前記ハードウェア・スイッチ、前記複数のホスト・コンピュータ・システム、および前記複数の入出力アダプタの相互接続性を含むトポロジを識別するステップと、
前記識別されたトポロジを前記マスタ制御ノードに保存するステップと
をさらに含む、請求項１に記載の方法。
【請求項７】
前記ハードウェア・スイッチの１つに関して、前記ハードウェア・スイッチの前記１つに接続されるすべてのハードウェア・スイッチを識別するデバイス・トポロジを決定するステップと、
前記ハードウェア・スイッチの前記１つに含まれる前記経路指定テーブルに前記デバイス・トポロジを保存するステップと
をさらに含む、請求項１に記載の方法。
【請求項８】
複数のホスト・コンピュータ・システムを含むデータ処理環境における装置であって、
前記複数のホスト・コンピュータ・システムは複数の入出力アダプタにスイッチドファブリックを使用して接続され、
前記スイッチドファブリックは、前記複数のホスト・コンピュータ・システムのうちの、エラー・メッセージによって識別されるエラーにより影響を受けるホスト・コンピュータ・システムのみに前記エラー・メッセージを経路指定し、
前記複数の入出力アダプタのうちの１つが、前記複数の入出力アダプタのうちの当該１つにおけるエラーを検出し、
前記複数の入出力アダプタのうちの前記１つが、前記エラーについての情報を含み且つ前記複数の入出力アダプタのうちの前記１つを識別する識別子を含むエラー・メッセージを生成し、
前記装置は、
前記スイッチドファブリックにおけるハードウェア・スイッチに保存され、前記複数のホスト・コンピュータ・システムのうちの、前記データ処理環境内に含まれる複数の入出力アダプタのうちの１つにおいて生じたエラーにより影響を受けるホスト・コンピュータ・システムを識別するために使用される経路指定テーブルを備えており、
前記ハードウェア・スイッチが、前記エラー・メッセージを、前記識別されたホスト・コンピュータ・システムのみに経路指定する、
前記装置。
【請求項９】
前記エラーにより影響を受けるホスト・コンピュータ・システムを識別する単一のマスタ制御ノードをさらに備えており、
前記エラー・メッセージが前記マスタ制御ノードに経路指定される、
請求項８に記載の装置。
【請求項１０】
単一のマスタ制御ノードをさらに備えており、
前記エラー・メッセージが前記マスタ制御ノードに経路指定され、
前記エラーが解消されないうちは、前記識別されたホスト・コンピュータ・システムのすべてが前記エラー・メッセージの受信を確認するまで、前記マスタ制御ノードが待機し、
前記エラーが解消されるまで、前記複数の入出力アダプタのうちの前記１つへのトラフィックが中断される、
請求項８に記載の装置。
【請求項１１】
前記識別されたホスト・コンピュータ・システムの第１のホスト・コンピュータ・システムが前記エラー・メッセージを受信し、

前記第１のホスト・コンピュータ・システムが前記エラー・メッセージの受信を確認し、
前記第１のホスト・コンピュータ・システムが前記第１のホスト・コンピュータ・システムから前記複数の入出力アダプタのうちの前記１つへの伝送トラフィックを中断し、
前記エラーが解消されるまで前記第１のホスト・コンピュータ・システムが待機し、しかる後前記複数の入出力アダプタのうちの前記１つへの伝送トラフィックを再開する、
請求項８に記載の装置。
【請求項１２】
前記ハードウェア・スイッチはレジスタを備えており、
前記レジスタにおける個々のビットが前記複数のホスト・コンピュータ・システムの各々に割り当てられ、
前記エラー・メッセージに応答して、前記複数の入出力アダプタのうちの前記１つへのトラフィックが中断され、
前記第１のホスト・コンピュータ・システムが前記エラー・メッセージを受信し、
前記第１のホスト・コンピュータ・システムに割り当てられたビットをセットすることによって、前記第１のホスト・コンピュータ・システムが前記エラー・メッセージの受信を確認し、
前記識別されたホスト・コンピュータ・システムのすべてが前記エラー・メッセージの受信を確認したかどうかを決定するために、マスタ制御ノードが前記レジスタをポーリングし、
前記識別されたホスト・コンピュータ・システムのすべてが前記エラー・メッセージの受信を確認したという前記マスタ制御ノードによる決定に応答して、前記識別されたホスト・コンピュータ・システムが前記複数の入出力アダプタのうちの前記１つへの伝送トラフィックを再開することを可能にするために前記マスタ制御ノードが前記エラーを解消し、
前記識別されたホスト・コンピュータ・システムのすべてが前記エラー・メッセージの受信を確認したのではないという前記マスタ制御ノードによる決定に応答して、前記マスタ制御ノードが前記エラーの解消を待ち、前記エラーの解消を待つ間、前記複数の入出力アダプタのうちの前記１つへのトラフィックが中断されたままである、
請求項８に記載の装置。
【請求項１３】
前記スイッチドファブリックを全探索して、前記ハードウェア・スイッチのすべての識別情報、ならびに、前記ハードウェア・スイッチ、前記複数のホスト・コンピュータ・システム、および前記複数の入出力アダプタの相互接続性を含むトポロジを識別するマスタ制御ノードをさらに備えており、
前記識別されたトポロジを前記マスタ制御ノードに保存する、
請求項８に記載の装置。
【請求項１４】
前記ハードウェア・スイッチの１つに関して、前記ハードウェア・スイッチの前記１つに接続されるすべてのハードウェア・スイッチを識別するデバイス・トポロジが決定され、
前記ハードウェア・スイッチの前記１つに含まれる前記経路指定テーブルに前記デバイス・トポロジが保存される、
請求項８に記載の装置。
【請求項１５】
データ処理システムに、請求項１～７のいずれか一項に記載の方法の各ステップを実行させるコンピュータ・プログラム。
【発明の詳細な説明】
【技術分野】
【０００１】

本発明は、一般的に、データ処理システムに関し、特に、複数のアダプタを共用し、ＰＣＩ交換ファブリック・バスを介してそれらのアダプタと通信を実施する複数のホスト・コンピュータ・システムおよび複数のアダプタを含むデータ処理システムにおける通信に関する。さらに詳しく言えば、本発明は、１つのデバイスにおいて生じたエラーを、そのエラーの影響を受け得る、そのファブリック内のパスにおけるすべてのトラフィックが中断され、そしてエラーを解消しないうちは、そのエラーの影響を受け得るすべてのホスト・ノードがそのエラー発生の通知の受信を確認するまで待機する単一のマスタ制御ホスト・ノードにレポートするための、コンピュータにおいて実施される方法、装置、およびコンピュータ・プログラムに関する。

【背景技術】

【０００２】

通常のＰＣＩバスは、拡張カードがパーソナル・コンピュータのような単一のコンピュータ・システム内に設置されることを可能にするローカル・パラレル・バスである。従って、ＰＣＩ準拠のアダプタ・カードは、ディスク・ドライブのような入出力（Ｉ／Ｏ）装置を追加するためにＰＣＩバスを使用することができる。ＰＣＩバスをコンピュータ・システムのシステム・バスに接続するためには、ＰＣＩブリッジ／コントローラが必要である。ＰＣＩバスは、そのＰＣＩバスが設置されるコンピュータ・システムのＣＰＵと、ＰＣＩブリッジ／コントローラを介して通信を行うことができる。単一のコンピュータ・システム内に複数のＰＣＩブリッジが存在することがある。しかし、これらのＰＣＩブリッジは、複数のＰＣＩバスを、それらのＰＣＩバスが配置されたコンピュータ・システムのＣＰＵに接続するように働く。かかるコンピュータ・システムが複数のＣＰＵを含む場合は、複数のＣＰＵがＰＣＩバスを利用するようにすることもできる。

【０００３】

ＰＣＩエクスプレス（ＰＣＩ－Ｅ）バスは標準のＰＣＩコンピュータ・バスの改良版である。ＰＣＩエクスプレスは、より高速のシリアル伝送を実現する。さらに、ＰＣＩエクスプレスは、特に、ルート・コンプレックス（ＲＣ）がホスト・コンピュータ・システム・サブシステムを入出力に接続する入出力階層構造のルートを示すとともに、木構造の入出力相互接続トポロジを意識して体系化されている。

【０００４】

ＰＣＩエクスプレスは、ＰＣＩソフトウェア環境と互換性のあるマイグレーション・パスを提供する。ＰＣＩエクスプレスは、上位帯域幅、性能、ならびに、バス幅およびバス周波数の両方における拡張容易性を提供することに加えて、サービス品質（ＱｏＳ：quality of service）、積極的な電源管理、ネイティブ・ホットプラグ、ピン当たりの帯域幅効率、エラー・レポート、回復、修正、および革新的なフォーム・ファクタを含む他の先進的な特徴を提供し、ピア・ツー・ピア転送および動的再構成のような高度の機能に対する要求に適応する。ＰＣＩエクスプレスは、低いピン・カウントおよびワイヤを通して製品の低コスト設計も可能にする。線形スケールの１６レーンＰＣＩエクスプレス相互接続は、８ギガバイト／秒以上のデータ転送速度を提供することができる。

【０００５】

一般に、ホスト・コンピュータ・システムは、ルート・コンプレックスとして広く知られているＰＣＩ・ツー・ホスト・ブリッジ機能を有する。ルート・コンプレックスは、ハイパ・トランスポートのようなＣＰＵバスとＰＣＩバスとの間をブリッジする。必要な場合、アドレス変換のような他の機能をルート・コンプレックスにおいて実施することできる。１以上のルート機能を含む複数ホスト・コンピュータ・システムは、マルチルート・システムとも呼ばれる。ここで、入出力（Ｉ／Ｏ）ファブリックを共用するマルチルート構成は、これまで十分に検討されていない。

【０００６】

現在では、ＰＣＩエクスプレス・バスは、複数の個別のコンピュータ・システム間におけるＰＣＩアダプタの共用を許容していない。ＰＣＩ標準、あるいは、ファイバ・チャネル、InfiniBand またはイーサネット（登録商標）のような二次的ネットワーク標準に準

拠
した既知の入出力アダプタは、一般に、ブレードおよびサーバ・コンピュータ・システムとして統合され、それらが統合されるブレードまたはシステムに専用のものである。アダプタを専用化してしまうことは、アダプタがかなり高価なものであるので、各システムのコストを高くする。さらに、種々のホスト・コンピュータ・システム間でアダプタを共用できないということが、これらの技術の採用を遅れさせる一因となっている。

【０００７】

コストの問題のほかに、ブレード・システムには、物理的スペースの問題がある。アダプタ用のブレードにおいてのために利用し得るスペースには制約がある。

【０００８】

入出力ファブリックを共用するマルチルート入出力ネットワーク構成は、これまで十分に検討されていなかった。そのため、既知のシステムでは、エラーが検出された時、そのエラーはすべてのホスト・ノードにレポートされ、入出力ファブリックにおいて検出されたエラーは、一般に、そのファブリックを使用している可能性のあるすべてのホスト・ノードを停止させる可能性があった。

【０００９】

すべてのホストに影響を及ぼし、すべてのホストにレポートされなければならないタイプのエラーが存在する。例えば、スイッチが故障した場合、すべてのホストに通知をする必要がある。しかし、別のタイプのエラーは、特定のホストのみに影響を与えるだけであって、すべてのホストに影響を与えるものではないこともある。例えば、アダプタが機能停止する場合、そのアダプタを利用している各ホスト・ノードにはエラーが通知されなければならないが、その他のホストには通知をする必要がないこともあるだろう。

【００１０】

既知のシステムでは、エラーが特定のホスト・ノードに影響を与えるのかまたはすべてのホスト・ノードに影響を与えるのかに関係なくすべてのエラーがすべてのホスト・ノードにレポートされる。なぜなら、そのエラーの影響を受ける可能性があるホスト・ノードのみにエラーのレポートを経路指定する方法がないためである。

【００１１】

従って、デバイス（コンポーネントとも呼ばれる）において生じたエラーを単一のマスタ制御ホスト・コンピュータ・システムにレポートするための方法、装置、およびコンピュータ・プログラムに対する要求が存在することが理解される。具体的には、ファブリックにおけるパスにあってそのエラーの影響を受け得るすべてのトラフィックが一時的に停止され、そのマスタ制御ホスト・コンピュータ・システムがエラーを解消しないうちは、そのエラーの影響を受け得るすべてのホスト・コンピュータ・システムがそのエラー発生の通知の受信を確認するまでそのマスタ制御ホスト・コンピュータ・システムが待機し、従って、そのエラーの影響を受け得るホスト・コンピュータ・システムのみにエラー・メッセージが経路指定されるようにする、方法、装置、およびコンピュータ・プログラムに対する要求が存在することが理解される。

【発明の開示】

【発明が解決しようとする課題】

【００１２】

本発明は、どのホスト・ノードおよび入出力ファブリック・デバイスが特定のエラーの影響を受け得るかを入出力ファブリックに対して定義し、マルチルート環境においてそのエラーの影響を受け得るホスト・ノードのみにエラー・メッセージを送付するための方法、装置、およびコンピュータ・プログラムを提供する。

【課題を解決するための手段】

【００１３】

複数ホスト・コンピュータ・システム環境においてエラー・メッセージを、エラーの影響を受けるホスト・コンピュータ・システムのみに経路指定するためのコンピュータ実施される方法、装置、およびコンピュータ・プログラムが開示される。複数ホスト・コンピ

ュータ・システム環境は、ファブリック（スイッチドファブリック（ｓｗｉｔｃｈｅｄ　ｆａｂｒｉｃ））を利用して複数のデバイスを共用する複数ホスト・コンピュータ・システムを含む。エラーはそれらデバイスの１つにおいて検出される。そのファブリックにおけるファブリック・デバイスに記憶された経路指定テーブルが、ホスト・コンピュータ・システムのうち、そのエラーの影響を受けるホスト・コンピュータ・システムを識別するために使用される。エラーを識別するエラー・メッセージが、ホスト・コンピュータ・システムのうちの識別されたホスト・コンピュータ・システムに経路指定される。

【００１４】

特に、本発明の実施例は、処理のためのＰＣＩエクスプレス入出力ファブリック・エラー・メッセージを適切なファブリック・デバイスおよびホスト・コンピュータ・システムに伝達するための方法、装置、およびコンピュータ・プログラムに関するものである。

【発明を実施するための最良の形態】

【００１５】

本発明の実施例は、複数のホスト・コンピュータ・システムが共通の入出力ファブリックを介して入出力アダプタ（ＩＯＡ）のプールを共用する任意の汎用または特定用途のコンピュータ・システムに適用する。好適な実施例では、ファブリックは、ＰＣＩエクスプレス標準に準拠したデバイスの集合体である。

【００１６】

本発明の実施例では、複数の種々のホスト・コンピュータ・システムが他のホスト・コンピュータ・システムと入出力アダプタを共用し得るように、入出力ファブリックが２つ以上のホスト・コンピュータ・システムに接続される。入出力ファブリックに接続されたアダプタの１つによって検出されたエラーは、その影響を受けるホスト・コンピュータ・システムおよびマスタ制御ルート・ノードに経路指定される。そのホスト・コンピュータ・システムの１つがマスタ制御ルート・ノードとして作用する。

【００１７】

本発明の実施例によれば、ファブリックは、すべてのアダプタ・エラーを、マスタ制御ルート・ノード、およびそのエラーの影響を受け得るすべての他のホスト・コンピュータ・システムおよび他のファブリック・デバイスにレポートする。そこで、マスタ制御ルート・ノードおよび他のホスト・ノードは、影響を受けるファブリックを介したそれらの伝送を中断する。マスタ制御ルート・ノードは、その影響を受けたすべてのホスト・コンピュータ・システムがそのエラーを知っているということをレポートするまで待機し、しかる後、その影響を受けたホスト・コンピュータ・システムがその影響を受けたファブリック・デバイスを介して入出力オペレーションを再開することをホスト・コンピュータ・システムが可能にする。

【００１８】

次に、図面、特に図１を参照すると、本発明の好適な実施例に従って、分散型コンピュータ・システム１００のブロック図が示される。図１に示された分散型コンピュータ・システム１００は、２つ以上のルート・コンプレックス（ＲＣ）１０８、１１８、１２８、１３８、および１３９が入出力リンク１１０、１２０、１３０、１４２、および１４３を介して入出力ファブリック１４４に接続され、ルート・ノード（ＲＮ）１６０～１６３のメモリ・コントローラ１０４、１１４、１２４、および１３４に接続されるという形を取る。ルート・コンプレックスは、ホスト・コンピュータ・システム・サブシステムを入出力に接続する入出力階層構造のルートを表す。ルート・コンプレックスはルート・ノード内に含まれる。ルート・ノードは、サーバ・コンピュータ・システムのような完全なコンピュータ・システムである。ルート・ノードは、本明細書では、ホスト・ノードとも呼ばれる。

【００１９】

入出力ファブリックはリンク１５１～１５８を介して入出力アダプタ１４５～１５０に接続される。入出力アダプタは、入出力アダプタ１４５～１４６および１４９におけるように単一機能の入出力アダプタであってもよく、入出力アダプタ１４７～１４８および１

５０におけるように多重機能の入出力アダプタであってもよい。さらに、入出力アダプタは、入出力アダプタ１４５～１４８におけるように単一のリンクを介して入出力ファブリックに接続されてもよく、入出力アダプタ１４９～１５０におけるように冗長性のための複数のリンクと接続されてもよい。

【００２０】

ルート・コンプレックス１０８、１１８、１２８、１３８、および１３９はルート・ノード１６０～１６３の一部分である。ルート・ノード１６３におけるように、１つのルート・ノード当たり２つ以上のルート・コンプレックスがあってもよい。各ルート・ノードは、ルート・コンプレックスのほかに１以上の中央処理ユニット（ＣＰＵ）１０１～１０２、１１１～１１２、１２１～１２２、１３１～１３２、メモリ１０３、１１３、１２３、および１３３、ＣＰＵ、メモリ、および入出力ファブリックを接続するメモリ・コントローラ１０４、１１４、１２４、および１３４を含み、メモリに対するコヒーレンシ・トラフィックを処理する機能等を実現する。

【００２１】

ルート・ノードは、そのメモリ・コントローラにおけるライン１５９により相互に接続されて１つのコヒーレンシ領域を形成することが可能であり、その場合、単一の対称マルチプロセッサ（ＳＭＰ）システムとして作用し得る。あるいは、ルート・ノードは、ルート・ノード１６２～１６３におけるように個別のコヒーレンシ領域を有する独立したノードとすることもできる。

【００２２】

構成マネージャ１６４は、入出力ファブリック１４４に別個に接続されてもよく、あるいはルート・ノード１６０～１６３の１つの一部分であってもよい。構成マネージャ１６４は、入出力ファブリックの共用リソースを構成し、リソースをルート・ノードに割り当てる。

【００２３】

分散型コンピュータ・システム１００は、種々の商業的に入手可能なコンピュータ・システムを使用して実現することが可能である。例えば、分散型コンピュータ・システム１００は、インターナショナル・ビジネス・マシーンズ・コーポレーションから入手し得るIBM eServer iSeries Model 840 システムを使用して実現することが可能である。そのようなシステムは、インターナショナル・ビジネス・マシーンズ・コーポレーションから入手し得るOS/400 オペレーティング・システムを使用して論理的パーティショニングをサポートすることが可能である。

【００２４】

図１に示されたハードウェアが変更可能であることは当業者には明らかであろう。例えば、光ディスク・ドライブ等のような他の周辺装置が図示のハードウェアに加えてまたは図示のハードウェアの代わりに使用することも可能である。図示の例は、本発明の実施例に関してアーキテクチャ上の限定を示唆することを意味するものではない。

【００２５】

次に、図２を参照すると、本発明の実施例を具現化し得る例示的な論理的にパーティション化されたプラットフォームのブロック図が示される。論理的にパーティション化されたプラットフォーム２００におけるハードウェアは、例えば、図１における分散型コンピュータ・システム１００として具現化することが可能である。論理的にパーティション化されたプラットフォーム２００は、パーティション化されたハードウェア２３０、オペレーティング・システム２０２、２０４、２０６、２０８、およびパーティション管理ファームウェア（プラットフォーム・ファームウェア）２１０を含む。

【００２６】

オペレーティング・システム２０２、２０４、２０６、および２０８は、単一のオペレーティング・システムの多重コピーであってもよく、あるいは論理的にパーティション化されたプラットフォーム２００上で同時に実行される複数の異種のオペレーティング・システムであってもよい。これらのオペレーティング・システムは、ハイパーバイザのよう

なパーティション管理ファームウェア２１０とインターフェースするように設計されたＯＳ／４００を使用して具現化することも可能である。ＯＳ／４００は、これらの実施例では単に１つの例として使用される。特定の具現化方法に依存して、ＡＩＸ（Ｒ）およびＬinux（Ｒ） のような他のタイプのオペレーティング・システムを使用することも可能である。オペレーティング・システム２０２、２０４、２０６、および２０８は、パーティション２０３、２０５、２０７、および２０９に設けられる。パーティション管理ファームウェア２１０を具現化するために使用することが可能なソフトウェアの一例は、インターナショナル・ビジネス・マシーンズ・コーポレーションから入手可能であるハイパーバイザ・ソフトウェアである。ファームウェアは、例えば、リード・オンリ・メモリ（ＲＯＭ）、プログラマブルＲＯＭ（ＰＲＯＭ）、消去可能プログラマブルＲＯＭ（ＥＰＲＯＭ）、電気的消去可能プログラマブルＲＯＭ（ＥＥＰＲＯＭ）、および不揮発性ランダム・アクセス・メモリ（不揮発性ＲＡＭ）のような、電力なしで内容を保持するメモリ・チップに記憶された「ソフトウェア」である。

【００２７】

さらに、これらのパーティションは、パーティション・ファームウェア２１１、２１３，２１５、および２１７も含む。パーティション・ファームウェア２１１、２１３，２１５、および２１７は、初期ブート・ストラップ・コード、ＩＥＥＥ－１２７５標準オープン・ファームウェア、およびインターナショナル・ビジネス・マシーンズ・コーポレーションから入手可能であるランタイム・アブストラクション・ソフトウェア（ＲＴＡＳ）を使用して具現化することが可能である。パーティション２０３、２０５、２０７、および２０９がインスタンス化されるとき、ブート・ストラップ・コードのコピーがプラットフォーム・ファームウェア２１０によってパーティション２０３、２０５、２０７、および２０９上にロードされる。しかる後、ブート・ストラップ・コードがオープン・ファームウェアおよびＲＴＡＳをロードすることによって、制御がそのブート・ストラップ・コードに移される。そこで、それらのパーティションに関連したまたは割り当てられたプロセッサが、そのパーティション・ファームウェアを実行するためにパーティションのメモリにディスパッチされる。

【００２８】

パーティション化されたハードウェア２３０は、複数のプロセッサ２３２～２３８、複数のシステム・メモリ・ユニット２４０～２４６、複数の入出力アダプタ２４８～２６２、ストレージ・ユニット２７０を含む。プロセッサ２３２～２３８、メモリ・ユニット２４０～２４６、ＮＶＲＡＭ２９８、および入出力アダプタ２４８～２６２の各々、またはそれらの一部は、パーティションの１つに割り当てられることによって論理的パーティション・プラットフォームにおける複数のパーティションの１つにパーティション化されてもよい。その場合、パーティション化されたリソースの各々はオペレーティング・システム２０２、２０４、２０６、および２０８の１つに対応する。

【００２９】

パーティション管理ファームウェア２１０は、論理的にパーティション化されたプラットフォーム２００のパーティション化を作成および強化するために、パーティション２０３、２０５，２０７、および２０９に関する多くの機能およびサービスを遂行する。パーティション管理ファームウェア２１０は、基礎的なハードウェアと同じファームウェア実装の仮想マシンである。従って、パーティション管理ファームウェア２１０は、論理的にパーティション化されたプラットフォーム２００のハードウェア・リソースを仮想化することによって、独立したＯＳイメージ２０２、２０４、２０６、および２０８の同時実行を可能にする。

【００３０】

サービス・プロセッサ２９０は、パーティションにおけるプラットフォーム・エラーの処理のような種々のサービスを提供するために使用することが可能である。これらのサービスは、インターナショナル・ビジネス・マシーンズ・コーポレーションのようなベンダにエラーをレポートするためのサービス・エージェントとして作用することも可能である

。種々のパーティションのオペレーションは、ハードウェア管理コンソール２８０のようなハードウェア管理コンソールを通して制御することが可能である。ハードウェア管理コンソール２８０は、別個の分散型コンピューティング・システムであり、そのシステムによって、システム管理者が、種々のパーティションへのリソースの再割り振りを含む種々の機能を遂行することが可能である。論理的にパーティション化された（ＬＰＡＲ）環境では、１つのパーティションにおけるリソースまたはプログラムが他のパーティションにおけるオペレーションに影響を与えることは許されないことである。さらに、有用であるために、リソースの割当てはきめ細かくされる必要がある。

【００３１】

図３は、本発明の実施例に従って本明細書に示されたいずれのデータ処理システムを具現化するためにも使用することが可能であるデータ処理システムのブロック図である。データ処理システム３００は、システム・バス３０６に接続された複数のプロセッサ３０２および３０４を含む対称マルチプロセッサ（ＳＭＰ）システムであってもよい。それとは別に、シングル・プロセッサ・システムを使用することも可能である。図示の例では、プロセッサ３０４はサービス・プロセッサである。システム・バス３０６には、ローカル・メモリ３０９に対するインターフェースを提供するメモリ・コントローラ／キャッシュ３０８が接続される。入出力バス・ブリッジ３１０がシステム・バス３０６に接続され、入出力バス３１２に対するインターフェースを提供する。メモリ・コントローラ／キャッシュ３０８および／Ｏバス・ブリッジ３１０は、図示のように統合されてもよい。

【００３２】

入出力バス３１２に接続された通常の周辺コンポーネント相互接続（ＰＣＩ）バス・ブリッジ３１４は、通常のＰＣＩローカル・バス３１６に対するインターフェースを提供する。モデム３１８のような多くの入出力アダプタがＰＣＩバス３１６に接続されてもよい。典型的なＰＣＩバスの実装体は、４つのＰＣＩ拡張スロットまたはアドイン・コネクタをサポートするであろう。モデム３１８および通信アダプタ３２０を通して他のコンピュータに対する通信リンクをサポートすることも可能である。通信アダプタ３２０は、データ処理システム３００が通信リンク３８０を介して他のコンピュータ・システムとの間でメッセージを送信および受信することを可能にする。

【００３３】

更なるＰＣＩバス・ブリッジ３２２および３２４は、更なるＰＣＩバス３２６および３２８に対するインターフェースを提供し、それらのＰＣＩバスによって、更なるモデムまたはネットワーク・アダプタをサポートすることも可能である。このように、データ処理システム３００は、複数のネットワーク・コンピュータへの接続を可能にする。メモリ・マップされたグラフィックス・アダプタ３３０およびハード・ディスク３３２を、図示のように、入出力バス３１２に直接的にまたは間接的に接続することも可能である。

【００３４】

図４は、本発明の実施例に従って、エラーをレポートするために使用されるメッセージ・リクエスト・パケットの一般的なレイアウトのブロック図を示す。メッセージ・リクエスト・パケット４００は、ヘッダ・フィールド４０２、要求元識別子（ＩＤ）フィールド４０４、データ・フィールド４０６、およびメッセージ・コード・フィールド４０８を含む。メッセージ・リクエスト・パケット４００は、エラーが生じたという通知を送信するために使用される。メッセージ・リクエスト・パケット４００は、本明細書では、エラー・メッセージとも呼ばれる。メッセージ・リクエスト・パケット４００は、エラー・メッセージを送信するために使用されるエラー・メッセージ・パケットである。

【００３５】

要求元ＩＤフィールド４０４には、要求元ＩＤが含まれる。要求元ＩＤは、エラーが生じたデバイスを識別する。メッセージ・コード・フィールド４０８には、メッセージ・コードが保存される。メッセージ・コードは、生じた特定のエラーに関する情報を含む。

【００３６】

図５は、本発明の実施例に従って、唯一のＰＣＩルート・スイッチを含むＰＣＩスイッ

チのファブリックを利用して、コンピュータ・システムが入出力アダプタのようなアダプタに接続されるデータ処理環境を示す。データ処理環境５００はコンピュータ・システム５０２および５０４を含む。本明細書では、コンピュータ・システムはホスト・ノードとも呼ばれる。従って、コンピュータ・システム５０２は、ホスト・ノード５０２と呼ばれることもある。

【００３７】

コンピュータ・システム５０２～５０４は、物理的アダプタ５１２、５１４、５１６、５１８、５２０、５２２、５２４、および５２６を利用する。コンピュータ・システム５０２～５０４および物理的アダプタ５１２～５２６は、ファブリック５３０を介して相互に通信を行う。ファブリック５３０は、ＰＣＩブリッジ／スイッチ５３２、５３４、５３６、および５３８のような複数のＰＣＩブリッジ／スイッチを含む。ファブリック５３０は、ＰＣＩエクスプレス標準に準拠したデバイスのファブリックである。ＰＣＩスイッチ５３２はＰＣＩルート・スイッチであり、一方、ＰＣＩスイッチ５３４，５３６、および５３８はＰＣＩルート・スイッチではない。コンピュータ・システム５０２または５０４のようなホスト・ノードに直接的に接続されるときのＰＣＩスイッチがＰＣＩルート・スイッチである。

【００３８】

各コンピュータ・システムは、図２に示されるように、論理的にパーティション化されてもよい。例えば、コンピュータ・システム５０２は、論理的パーティション５４０および論理的パーティション５４２を含む。コンピュータ・システム５０４は、論理的パーティション５４４および論理的パーティション５４６を含む。

【００３９】

各物理的アダプタは、１つの物理的アダプタが複数の独立したアダプタであるように見えるよう、仮想化されてもよい。例えば、物理的アダプタ５１２は、２つの個別の仮想アダプタ５４８および５５０であるように見える。物理的アダプタ５１４は、３つの個別の仮想アダプタ５５２、５５４、および５５６であるように見える。物理的アダプタ５１６は、個別の仮想アダプタ５５８であるように見える。

【００４０】

各コンピュータ・システムおよび物理的アダプタは、ＰＣＩスイッチの１つにおけるポートに接続されることによってファブリック５３０に接続される。コンピュータ・システム５０２は、ＰＣＩスイッチ５３２におけるポート５６０に接続される。コンピュータ・システム５０４は、ＰＣＩスイッチ５３２におけるポート５６２に接続される。物理的アダプタ５１２は、ＰＣＩスイッチ５３４におけるポート５６４に接続される。物理的アダプタ５１４は、ＰＣＩスイッチ５３４におけるポート５６６に接続される。物理的アダプタ５１６は、ＰＣＩスイッチ５３６におけるポート５６８に接続される。物理的アダプタ５１８は、ＰＣＩスイッチ５３８におけるポート５７０に接続される。物理的アダプタ５２０は、ＰＣＩスイッチ５３８におけるポート５７２に接続される。物理的アダプタ５２２は、ＰＣＩスイッチ５３８におけるポート５７４に接続される。物理的アダプタ５２４は、ＰＣＩスイッチ５３８におけるポート５７６に接続される。物理的アダプタ５２６は、ＰＣＩスイッチ５３８におけるポート５７８に接続される。

【００４１】

各ＰＣＩスイッチは、ファブリック５３０内の他のＰＣＩスイッチに接続することも可能である。例えば、ＰＣＩスイッチ５３２におけるポート５８０がＰＣＩスイッチ５３４におけるポート５８２に接続される。ＰＣＩスイッチ５３２におけるポート５８４がＰＣＩスイッチ５３６におけるポート５８６に接続される。ＰＣＩスイッチ５３２におけるポート５８８がＰＣＩスイッチ５３８におけるポート５９０に接続される。

【００４２】

各ＰＣＩスイッチには、経路指定テーブルが含まれる。ＰＣＩスイッチ５３２には、経路指定テーブル５９２が含まれる。ＰＣＩスイッチ５３４には、経路指定テーブル５９３が含まれる。ＰＣＩスイッチ５３６には、経路指定テーブル５９４が含まれる。ＰＣＩス

イッチ５３８には、経路指定テーブル５９５が含まれる。

【００４３】

マスタ制御ノード５０２において、マスタ経路指定テーブル５９６が生成され、保存される。マスタ経路指定テーブル５９６は、経路指定テーブル５９２～５９５の組合せであり、経路指定テーブル５９２～５９５のすべての内容を含む。各アダプタは、それ自身のエラー検出および制御ロジックを含む。例えば、アダプタ５２０は、エラー検出および制御ロジック５９８を含む。エラー検出および制御ロジックは、１つのアダプタのみに示されているが、それは、すべてのアダプタにおいて、図示されていないが、存在する。

【００４４】

ＰＣＩルート・スイッチ、すなわち、図５におけるＰＣＩスイッチ５３２はレジスタ５９７を含む。レジスタ５９７は、各ホスト・ノードに対して１つのビットを含む。図示の例では２つのホスト・ノードしか存在しないので、レジスタ５９７は２つのホスト・ノード・ビットを含むであろう。各ビットは、ホスト・ノードの異なる１つと関連付けられる。従って、レジスタ５９７における第１ビットはコンピュータ・システム５０２と関連付けられ、レジスタ５９７における第２ビットはコンピュータ・システム５０４と関連付けられる。

【００４５】

以下は、アダプタ内でエラーが生じたとき、すなわち、アダプタが故障したとき、そのエラーをレポートするという例を説明する。本明細書において説明される方法、装置、およびコンピュータ・プログラムは、システム内のいずれかのコンポーネントにおいてエラーが生じたとき、そのエラーをレポートするために使用することも可能である。この場合、コンポーネントはアダプタ、ブリッジ、スイッチ、または任意の他のデバイスを含む。

【００４６】

本発明の実施例における例として、エラー検出および制御ロジック５９８がアダプタ５２０内のエラーを検出する。そこで、エラー検出および制御ロジック５９８は、図４のパケット４００によって示されたフォーマットのメッセージ・リクエスト・パケットを生成する。エラーが生じたアダプタを識別する要求元ＩＤがそのパケットに含められる。この場合、要求元ＩＤはアダプタ５２０を識別する。メッセージ・リクエスト・パケットは、エラーがアダプタ内で生じたということをホストに知らせるために使用されるエラー・メッセージである。

【００４７】

要求元ＩＤは、ファブリックの初期化時に構成コードによって設定され、ＰＣＩエクスプレスに関しては、アダプタ５２０におけるデバイスのバス番号、デバイス番号、および機能番号である。そこで、メッセージ・リクエスト・パケット４００は、それが第１ＰＣＩスイッチ、すなわち、ＰＣＩスイッチ５３８に到達するまで、ファブリック５３０を通して送られる。ＰＣＩスイッチ５３８は経路指定テーブル５９５を含む。

【００４８】

メッセージ・リクエスト・パケット４００は、まず、アダプタ５２０からＰＣＩスイッチ５３８に送られる。ＰＣＩスイッチ５３８は、受信したメッセージ・リクエスト・パケットにどのような要求元ＩＤが保存されているかを決定することによって、要求元を識別する。そこで、ＰＣＩスイッチ５３８は、経路指定テーブル５９５を使用してその要求元のエントリを探索する。

【００４９】

経路指定テーブルにおける正しいエントリのサーチを任意の多くの方法で遂行することが可能であることは当業者には明らかであろう。さらに、経路指定テーブルは、情報が保存される任意のタイプのデータ構造のものであってもよい。それは、例えば、エラー・メッセージにおける要求元ＩＤに等しいそのテーブル内の要求元ＩＤの値に関してそのテーブルをスキャンすることを伴う内容アドレス・メモリ、経路指定テーブルに対するインデックスとしてエラー・メッセージにおける要求元ＩＤの使用等を含む。

【００５０】

ＰＣＩスイッチ５３８は、パケットの要求元ＩＤフィールドにおいて識別されるアダプタ５２０に関して、中間ポート５８８に対するビットが設定されるということをその経路指定テーブルから決定する。そこで、ＰＣＩスイッチ５３８は、メッセージ・リクエスト・パケットを中間ポート５８８に送る。

【００５１】

次に、ＰＣＩスイッチ５３２がそのポート５８８からこのメッセージ・リクエスト・パケットを受信する。ＰＣＩスイッチ５３２はそれの経路指定テーブル５９２を使用して、そのパケットにおいて識別される要求元をルックアップすることによりどのルート・ポートおよびどの中間ポートが識別されるかを決定する。ＰＣＩスイッチ５３２は、そのパケットの要求元ＩＤフィールドにおいて識別されるアダプタ５２０に関して、ルート（ホスト）ポート５６０および５６２に対するビットがセットされかつ中間ポートに対するビットがセットされないということをその経路指定テーブルから決定する。そこで、ＰＣＩスイッチ５３２は、メッセージ・リクエスト・パケットをルート・ポート５６０および５６２に送る。次に、コンピュータ・システム５０２は、ＰＣＩルート・スイッチ５３２からメッセージ・リクエスト・パケットを受信するであろう。コンピュータ・システム５０４も、ＰＣＩルート・スイッチ５３２からメッセージ・リクエスト・パケットを受信するであろう。

【００５２】

コンピュータ・システム５０２はマスタ制御ノードである。すべてのファブリック・エラーがそのマスタ制御ノードにレポートされる。従って、コンピュータ・システム５０２は、アダプタ５２０においてエラーが生じたということを表すメッセージ・リクエスト・パケットをＰＣＩルート・スイッチ５３２から受信するであろう。そこで、コンピュータ・システム５０２は、そのレポートされたエラーによってどのパスが影響を受けるかを決定する。コンピュータ・システム５０２は、そのコンピュータ・システム５０２に保存されているマスタ経路指定テーブルを調べることによってこの決定を行う。ファブリック全体のトポロジならびにファブリックに接続されたホストおよびアダプタのトポロジに関する情報はマスタ経路指定テーブル５９２に保存されている。コンピュータ・システム５０２は、そのトポロジ情報を使用して、ファブリック５３０を通るどのパスがエラーの影響を受けるかを識別する。エラーがアダプタ５２０において生じたとする上記の例では、影響を受けるパスはＰＣＩスイッチ５３８およびＰＣＩスイッチ５３２を含む。コンピュータ・システム５０２および５０４は、エラーの影響を受けるものとしても識別される。

【００５３】

各影響を受けるコンピュータ・システムは、エラーが特定のアダプタにおいて生じたということを表すメッセージ・パケットをそのコンピュータ・システムが受信するとき、そのエラーに関して通知されるであろう。コンピュータ・システムがそのようなメッセージ・パケットを受信するとき、そのコンピュータ・システムはその特定のアダプタへのトラフィックを中断するであろう。そこで、このコンピュータ・システムは、このコンピュータ・システムがそのエラーに気づきそして特定のアダプタへのトラフィックを中断したということを通知することによって、そのエラーを確認する責任を負う。そこで、コンピュータ・システムは、そのエラーが解消されるまで待機し、しかる後、そのコンピュータ・システムは特定のアダプタへの伝送トラフィックを再び開始する。

【００５４】

マスタ制御ノードは、そのマスタ制御ノードがエラーを解消しないうちは、そのエラーの影響を受けた各ホスト・コンピュータ・システムがそのエラーを確認するまで待機する。各ルートＰＣＩスイッチは、各ホスト・コンピュータ・システムに対して１つのビットが割り当てられているレジスタを含む。コンピュータ・システムは、このレジスタにおける関連するビットをセットすることによってエラー・メッセージの受信を確認する。そこで、マスタ制御ノードはそのレジスタをポーリングして、特定のホスト・コンピュータ・システムがエラーの受信を確認したかどうかを判定することができる。

【００５５】

すべての影響を受けたコンピュータ・システムがエラーの受信を確認した後、マスタ制御ノードはレジスタにおけるビットをクリアするであろう。マスタ制御ノードは、レジスタにおけるコンピュータ・システムのビットをクリアすることによってそのエラーが解消されたことをコンピュータ・システムに示す。コンピュータ・システムがエラー・メッセージを受信したことを表すためにそのコンピュータ・システムがレジスタにおけるそれのビットをセットした後、そのコンピュータ・システムは、そのコンピュータ・システムのビットが今でもセットされているかどうかを決定するためにそのレジスタをポーリングし続けるであろう。コンピュータ・システムのビットがクリアされるとき、そのコンピュータ・システムは、それが特定のアダプタへの伝送トラフィックを再び開始し得るということを通知される。

【００５６】

コンピュータ・システム５０２がメッセージ・リクエスト・パケットを受信するとき、コンピュータ・システム５０２は、この場合、アダプタ５２０である故障コンポーネントを介したトラフィックを中断するであろう。コンピュータ・システム５０２がメッセージ・リクエスト・パケットを受信するとき、コンピュータ・システム５０２はレジスタ５９７をポーリングして、コンピュータ・システム５０４に対するビットがセットされているかどうか決定するであろう。コンピュータ・システム５０４に対するビットがセットされていることをコンピュータ・システム５０２が決定するまで、コンピュータ・システム５０２はレジスタ５９７をポーリングし続けるであろう。コンピュータ・システム５０４に対するビットがセットされているとき、コンピュータ・システム５０２はレジスタ５９７のビットをクリアするであろう。この時点で、コンピュータ・システム５０２は、可能であれば、アダプタ５２０への伝送トラフィックを再開するであろう。

【００５７】

コンピュータ・システム５０２がエラー・メッセージを受信するときに生じる上記のプロセスと同時に、コンピュータ・システム５０４がメッセージ・パケットを受信するとき、コンピュータ・システム５０４はアダプタ５２０へのそれのトラフィックを中断するであろう。そこで、コンピュータ・システム５０４はレジスタ５９７におけるビットをセットするであろう。コンピュータ・システム５０４がセットするビットは、そのコンピュータ・システム５０４と関連するビットである。このビットがセットされるとき、それは、コンピュータ・システム５０４がエラーの通知を受信したということ、すなわち、それがメッセージ・リクエスト・パケットを受信したということを表す。そのビットがクリアされているとき、それは、コンピュータ・システム５０４がエラーの通知を受信していないということを表す。そこで、コンピュータ・システム５０４はレジスタ５９７をポーリングして、そのコンピュータ・システム５０４と関連するビットが今でもセットされているかどうかを決定する。そのビットがセットされている間、コンピュータ・システム５０４はアダプタ５２０へのそれのトラフィックを中断し続け、そのレジスタをポーリングし続ける。そのビットがクリアされるとき、コンピュータ・システム５０４はアダプタ５２０への伝送トラフィックを再開するであろう。

【００５８】

図６は、本発明の実施例に従って、複数のＰＣＩルート・スイッチを含むＰＣＩスイッチのファブリックを利用して、コンピュータ・システムが入出力アダプタのようなアダプタに接続されるデータ処理システムを示す。データ処理システム６００はコンピュータ・システム６０２、６０４、６０６、６０８、および６１０を含む。コンピュータ・システム６０２～６１０は、物理的アダプタ６１２、６１４、６１６、６１８、６２０、６２２、６２４、および６２６を利用する。コンピュータ・システム６０２～６１０および物理的アダプタ６１２～６２６は、ファブリック６３０を介して相互に通信を行う。ファブリック６３０は、ＰＣＩブリッジ／スイッチ６３２、６３３、６３４、６３５、６３６、および６３８のような複数のＰＣＩブリッジ／スイッチを含む。ＰＣＩスイッチ６３２、６３３、および６３４はＰＣＩルート・スイッチであり、一方、ＰＣＩスイッチ６３５、６３６、および６３８はＰＣＩルート・スイッチではない。ＰＣＩスイッチは、そのＰＣＩ

スイッチがコンピュータ・システム６０２～６１０の１つのようなホスト・ノードに直接的に接続されるとき、ＰＣＩルート・スイッチである。

【００５９】

各コンピュータ・システムは、図２に示されるように、論理的にパーティションすることが可能である。例えば、コンピュータ・システム６０２は、論理的パーティション６４０および論理的パーティション６４２を含む。コンピュータ・システム６０４は、論理的パーティション６４４および論理的パーティション６４６を含む。

【００６０】

各物理的アダプタは、１つの物理的アダプタが複数の、個別の、および独立したアダプタであるように見えるように仮想化されることも可能である。例えば、物理的アダプタ６１２は２つの個別の仮想アダプタ６４８および６５０であるように見える。物理的アダプタ６１４は３つの個別の仮想アダプタ６５２、６５４、および６５６であるように見える。物理的アダプタ６１６は仮想アダプタ６５８であるように見える。

【００６１】

各コンピュータ・システムおよび物理的アダプタは、ＰＣＩスイッチの１つにおけるポートに接続されることによってファブリック６３０に接続される。コンピュータ・システム６０２がＰＣＩスイッチ６３２におけるポート６６０、スイッチ６３３におけるポート６６１、およびスイッチ６３４におけるポート６６５に接続される。コンピュータ・システム６０４がＰＣＩスイッチ６３３におけるポート６６２に接続される。コンピュータ・システム６０６がＰＣＩスイッチ６３３におけるポート６６４に接続される。コンピュータ・システム６０８がＰＣＩスイッチ６３４におけるポート６６６に接続される。コンピュータ・システム６１０がＰＣＩスイッチ６３４におけるポート６６８に接続される。

【００６２】

物理的アダプタ５１２がＰＣＩスイッチ６３５におけるポート６７０に接続される。物理的アダプタ６１４がＰＣＩスイッチ６３５におけるポート６７１に接続される。物理的アダプタ６１６がＰＣＩスイッチ６３６におけるポート６７２に接続される。物理的アダプタ６１８がＰＣＩスイッチ６３８におけるポート６７３に接続される。物理的アダプタ６２０がＰＣＩスイッチ６３８におけるポート６７４に接続される。物理的アダプタ６２２がＰＣＩスイッチ６３８におけるポート６７５に接続される。物理的アダプタ６２４がＰＣＩスイッチ６３８におけるポート６７６に接続される。物理的アダプタ６２６がＰＣＩスイッチ６３８におけるポート６７７に接続される。

【００６３】

各ＰＣＩスイッチは、ファブリック６３０内の他のＰＣＩスイッチに接続することも可能である。例えば、ＰＣＩスイッチ６３２におけるポート６７８がＰＣＩスイッチ６３５におけるポート６７９に接続される。ＰＣＩスイッチ６３３におけるポート６８０がＰＣＩスイッチ６３６におけるポート６８２に接続される。ＰＣＩスイッチ６３４におけるポート６８３がＰＣＩスイッチ６３８におけるポート６８４に接続される。ＰＣＩスイッチ６３５におけるポート６８５がＰＣＩスイッチ６３６におけるポート６８６に接続される。

【００６４】

各ＰＣＩスイッチには、経路指定テーブルが含まれる。ＰＣＩスイッチ６３２には、経路指定テーブル６８７が含まれる。ＰＣＩスイッチ６３３には、経路指定テーブル６８８が含まれる。ＰＣＩスイッチ６３４には、経路指定テーブル６８９が含まれる。ＰＣＩスイッチ６３５には、経路指定テーブル６９０が含まれる。ＰＣＩスイッチ６３６には、経路指定テーブル６９１が含まれる。ＰＣＩスイッチ６３８には、経路指定テーブル６９２が含まれる。

【００６５】

マスタ制御ノード６０２は、そのマスタ制御ノード６０２内にマスタ経路指定テーブル６９３を生成および保存する。マスタ経路指定テーブル６９３は、経路指定テーブル６８７～６９２の結合体を含む。

【００６６】
図６におけるＰＣＩスイッチ６３２～６３４はＰＣＩルート・スイッチである。従って、各ＰＣＩスイッチはレジスタを含む。ＰＣＩスイッチ６３２はレジスタ６９４を含む。ＰＣＩスイッチ６３３はレジスタ６９５を含む。ＰＣＩスイッチ６３４はレジスタ６９６を含む。
【００６７】
これらのレジスタの各々は、各ホスト・ノードに対して１つのビットを含む。したがって、各レジスタは、マスタ制御ノード６０２と関連した第１ビット、コンピュータ・システム６０４と関連した第２ビット、コンピュータ・システム６０６と関連した第３ビット、コンピュータ・システム６０８と関連した第４ビット、およびコンピュータ・システム６１０と関連した第５ビットを含むであろう。５つのホスト・ノードが存在するので、レジスタ６８７～６８９はそれぞれ５つのホスト・ノード・ビットを含むであろう。各ビットは、それらのホスト・ノードの異なる１つと関連する。
【００６８】
各アダプタは、エラー検出および制御ロジックを含む。例えば、アダプタ６２０は、エラー検出および制御ロジック６９７を含む。
【００６９】
本発明の１つの実施例として、エラー検出および制御ロジック６９７がアダプタ６２０内のエラーを検出する。しかる後、エラー検出および制御ロジック６９７は、アダプタ６２０を識別する要求元ＩＤが書込まれるメッセージ・リクエスト・パケット４００を生成する。このメッセージ・リクエスト・パケットは、アダプタ６２０内でエラーが生じたということを各ホスト・ノードに通知するために使用されるエラー・メッセージである。そこで、メッセージ・リクエスト・パケット４００は、それが第１ＰＣＩスイッチ、すなわち、ＰＣＩスイッチ６３８に達するまで、ファブリック６３０を通して送られる。
【００７０】
ＰＣＩスイッチ６３８は、どのような要求元ＩＤがメッセージ・リクエスト・パケットに保存されているか決定することによって要求元を識別する。そこで、ＰＣＩスイッチ６３８は、経路指定テーブル６９２を使用してテーブル６９２における要求元のエントリをルックアップする。ＰＣＩスイッチ６３８は、そのパケットの要求元ＩＤフィールドにおいて識別されるアダプタ６２０に関して、中間ポート６８３に対するビットがセットされているということをそれの経路指定テーブルから決定する。そこで、ＰＣＩスイッチ６３８はメッセージ・リクエスト・パケットを中間ポート６８３に送る。
【００７１】
次に、ＰＣＩスイッチ６３４が、それのポート６８３からこのメッセージ・リクエスト・パケットを受信する。ＰＣＩスイッチ６３４は、それの経路指定テーブル６８９を使用して、そのパケットにおいて識別される要求元をルックアップすることによってどのルート・ポートおよびどの中間ポートが識別されるかを決定する。ＰＣＩスイッチ６３４は、パケットの要求元ＩＤフィールドにおいて識別されるアダプタ６２０に関して、ルート（ホスト）ポート６６５、６６６、および６６８に対するビットがセットされ、いずれの中間ポートに対するビットもセットされてないということをその経路指定テーブルから決定する。次に、ＰＣＩスイッチ６３４は、メッセージ・リクエスト・パケットをルート・ポート６６５、６６６、および６６８に送る。
【００７２】
コンピュータ・システム６０２、６０８、および６１０は、それぞれ、ＰＣＩルート・スイッチ６３４からメッセージ・リクエスト・パケットを受信する。コンピュータ・システム６０８がメッセージ・パケットを受信するとき、コンピュータ・システム６０８はアダプタ６２０へのそれのトラフィックを中断するであろう。次に、コンピュータ・システム６０８はレジスタ６９６における１つのビットをセットする。コンピュータ・システム６０８がセットするビットは、コンピュータ・システム６０８と関連するビットである。このビットがセットされるとき、それは、コンピュータ・システム６０８がエラーの通知

、すなわち、メッセージ・リクエスト・パケットを受信したことを表す。そのビットがクリアされているとき、それは、コンピュータ・システム６０８がエラーの通知を受信していないことを表す。次に、コンピュータ・システム６０８は、レジスタ６９６をポーリングして、そのコンピュータ・システム６０８と関連するビットがセットされているかどうかを決定する。そのビットがセットされている間、コンピュータ・システム６０８はアダプタ６２０へのそれのトラフィックを中断し、そのレジスタをポーリングし続ける。そのビットがクリアされるとき、コンピュータ・システム６０８はアダプタ６２０への伝送トラフィックを再開するであろう。

【００７３】

コンピュータ・システム６１０がメッセージ・リクエスト・パケットを受信するとき、コンピュータ・システム６１０はアダプタ６２０へのそれのトラフィックを中断するであろう。次に、コンピュータ・システム６１０はレジスタ６９６における１つのビットをセットする。コンピュータ・システム６１０がセットするビットは、コンピュータ・システム６１０と関連するビットである。このビットがセットされるとき、それは、コンピュータ・システム６１０がエラーの通知を受信したことを表す。そのビットがクリアされているとき、それは、コンピュータ・システム６１０がエラーの通知を受信していないことを表す。次に、コンピュータ・システム６１０は、レジスタ６９６をポーリングして、そのコンピュータ・システム６１０と関連するビットがセットされているかどうかを決定する。そのビットがセットされている間、コンピュータ・システム６１０はアダプタ６２０へのそれのトラフィックを中断し、そのレジスタをポーリングし続ける。そのビットがクリアされるとき、コンピュータ・システム６１０はアダプタ６２０への伝送トラフィックを再開するであろう。

【００７４】

次に、コンピュータ・システム６０２がレジスタ６９６をポーリングし、コンピュータ・システム６０８および６１０と関連するビットがセットされているかどうかを決定する。コンピュータ・システム６０２は、コンピュータ・システム６０８および６１０に対するビットがセットされていることをそのコンピュータ・システム６０２が決定するまで、レジスタ６９６をポーリングし続けるであろう。コンピュータ・システム６０８および６１０に対するビットがセットされていたとき、コンピュータ・システム６０２はレジスタ６９６のビットをクリアするであろう。この時点で、コンピュータ・システム６０８および６１０はアダプタ６２０への伝送トラフィックを再開するであろう。

【００７５】

図７は、本発明の実施例に従って、経路指定テーブル・エントリ７００を示す。エントリ７００は、エラーが生じたデバイスを識別する要求元識別子（ＩＤ）７０２を含む。エントリ７００はさらにルート・ポート・ビット・アレイ７０４を含む。アレイ７０４における各ビットは、メッセージ・リクエスト・パケットが経路指定される必要のあるルート・ポートに対応する。エントリ７００は中間ポート・ビット・アレイ７０６を含む。アレイ７０６における各ビットは、メッセージ・リクエスト・パケットが経路指定される必要のある中間ポートに対応する。

【００７６】

図８は、本発明の実施例に従って、マスタ制御ノードがファブリック・トポロジを用いて経路指定テーブルを生成しかつそれらのテーブルを移植する操作を示す高レベルのフローチャートである。プロセスはブロック８００によって示されるように開始し、ブロック８０２に進む。ブロック８０２は、マスタ制御ノードがファブリックを全探索してそのファブリックおけるすべてのデバイスを識別し、ファブリック・トポロジを識別する。ファブリック・トポロジは、各デバイスを識別することおよびこれらのデバイスの相互接続性、すなわち、それらのデバイスすべてを相互に接続する方法を識別することによって識別される。次に、ブロック８０４は、マスタ制御ノードがマスタ経路指定テーブルを生成することを示す。そこで、マスタ制御ノードはマスタ経路指定テーブルをそのマスタ制御ノードに保存する。

【００７７】
次に、プロセスはブロック８０６に進む。ブロック８０６は、マスタ制御ノードがファブリック全体のトポロジをマスタ経路指定テーブルに保存することを示す。次に、ブロック８０８は、マスタ制御ノードがファブリックにおける各ＰＣＩスイッチに対する経路指定テーブルを生成することを示す。そこで、マスタ制御ノードは、各ＰＣＩスイッチの経路指定テーブルをそのＰＣＩスイッチに保存する。しかる後、ブロック８１０は、マスタ制御ノードが各可能な要求元に対するエントリを有する各ＰＣＩスイッチのテーブルをそのＰＣＩスイッチに移植する。各エントリは、すべての可能なルート・ポートおよび中間ポートを識別する。そこで、プロセスは、ブロック８１２に示されるように終了する。
【００７８】
図９は、本発明の実施例に従って、エラーを検出し、そのエラーを記述したメッセージ・リクエスト・パケットを生成および伝送するエラー検出および制御ロジックを示す高レベルのフローチャートである。プロセスは、ブロック９００によって示されるように開始し、しかる後、ブロック９０２に進む。ブロック９０２は、エラー検出ロジックがエラーを検出することを示す。ブロック９０４は、そのエラーを生じたデバイスをエラー検出ロジックが識別することを示す。そのデバイスの識別標識は、メッセージ・リクエスト・パケットにおける要求元識別子（ＩＤ）として使用される。
【００７９】
次に、ブロック９０６は、エラー検出ロジックがメッセージ・リクエスト・パケットを生成することを示す。エラー検出ロジックは、要求元ＩＤをそのパケットの要求元ＩＤフィールドに挿入し、この特定のエラーに関する情報をそのパケットのメッセージ・コード・フィールドに挿入する。そのエラーに関する情報はそのエラーのタイプを識別する。しかる後、ブロック９０８は、エラー検出ロジックがファブリックへのメッセージ・パケットをそれの最も近いＰＣＩスイッチまで伝送すること示す。そこで、プロセスは、ブロック９１０によって示されるように終了する。
【００８０】
図１０は、本発明の実施例に従って、ＰＣＩスイッチがそれの経路指定テーブルを利用してメッセージ・リクエスト・パケットを送り、故障した入出力アダプタに接続されたポートを介して入出力オペレーションを中断することを示す高レベルのフローチャートを示す。プロセスは、ブロック１０００によって示されるように開始し、しかる後、ブロック１００２に進む。ブロック１００２は、ＰＣＩスイッチがメッセージ・リクエスト・パケットを受信することを示す。次に、ブロック１００４は、ＰＣＩスイッチがそれの内部経路指定テーブルを使用して、メッセージ・リクエスト・パケットにおいて識別される要求元ＩＤをルックアップすることを示す。そこで、プロセスはブロック１００６に進む。ブロック１００６は、この要求元ＩＤに対する経路指定テーブルのエントリを使用してすべてのルート・ポートおよび中間ポートを識別することを示す。エントリが要求元ＩＤを使用して位置指定される。一旦この要求元ＩＤに対するエントリが位置指定されると、そのエントリにおいて識別されたルート・ポートおよび中間ポートが識別される。次に、ブロック１００８は、ＰＣＩスイッチが、経路指定テーブルにおいて識別された各ルート・ポートおよび各中間ポートにそのメッセージ・リクエスト・パケットを伝送することを示す。
【００８１】
次に、ブロック１０１０は、ＰＣＩスイッチが、故障したアダプタに直接的にまたは間接的に接続されているその経路指定テーブルにおいて識別されたすべてのポートを介してその故障したアダプタに対するすべてのオペレーションを中断することを示す。そこで、プロセスは、ブロック１０１２によって示されるように終了する。
【００８２】
図１１は、本発明の実施例に従って、エラーが解消されないうちは、ファブリック内の潜在的に影響を受けるパスにおけるすべてのトラフィックが中断され、すべての潜在的に影響を受けるホスト・ノードが、エラーが生じたという通知の受信を確認するまで、マス

タ制御ノードが待機することを示す高レベルのフローチャートを示す。プロセスは、ブロック１１００によって示されるように開始し、しかる後、ブロック１１０２に進む。ブロック１１０２は、マスタ制御ノードがメッセージ・リクエスト・パケットを受信することを示す。次に、ブロック１１０４は、マスタ制御ノードが、その影響を受けるファブリックを通してそのパケットにおける識別されたアダプタに送られるトラフィックを中断することを示す。その影響を受けるファブリックは、マスタ経路指定テーブルにおいて識別されたポートを含む。

【００８３】

次に、プロセスはブロック１１０６に進む。ブロック１１０６は、マスタ制御ノードがそれのマスタ経路指定テーブルを使用して、どのホスト・ノードがこのエラーの影響を受けるかを決定する。次に、ブロック１１０８は、各ＰＣＩルート・スイッチの各レジスタにおける各影響を受けるホスト・ノードに対するビットがセットされているかどうかを決定するために、マスタ制御ノードがすべてのＰＣＩルート・スイッチにおけるレジスタをポーリングする。次に、ブロック１１１０は、各ＰＣＩルート・スイッチの各レジスタにおける各影響を受けるホスト・ノードに対するすべてのビットがセットされているかどうかの決定を示す。各ＰＣＩルート・スイッチの各レジスタにおける各影響を受けるホスト・ノードに対するすべてのビットがセットされているわけではないという決定が行われる場合、プロセスはブロック１１０８に戻る。各ＰＣＩルート・スイッチの各レジスタにおける各影響を受けるホスト・ノードに対するすべてのビットがセットされているという決定が行われる場合、プロセスはブロック１１１２に進む。ブロック１１１２は、マスタ制御ノードが各ＰＣＩルート・スイッチにおける各レジスタをクリアすることを示す。そこで、プロセスはブロック１１１４によって示されるように終了する。

【００８４】

図１２は、本発明の実施例に従って、エラーの影響を受けるすべてのホスト・ノードがエラーを確認したことをマスタ制御ノードが信号するまで、ホスト・ノードが、エラーの影響を受けるファブリックの部分を介するそれのトラフィックを中断することを示す高レベルのフローチャートを示す。プロセスは、ブロック１２００によって示されるように開始し、しかる後、ブロック１２０２に進む。ブロック１２０２は、ホスト・ノードがメッセージ・リクエスト・パケットを受信することを示す。次に、ブロック１２０４は、ホスト・ノードが、エラー・メッセージにおける影響を受けるアダプタへの、その影響を受けるポートを介するそのトラフィックを中断すること、およびＰＣＩルート・スイッチのレジスタにおけるビットをセットすることを示す。これは、この特定のホスト・ノードと関連するビットである。

【００８５】

次に、プロセスはブロック１２０６に進む。ブロック１２０６は、このホスト・ノードと関連するビットがこのＰＣＩルート・ノードのレジスタにおいてセットされているかどうかを決定するために、ホスト・ノードがすべてのＰＣＩルート・スイッチにおけるレジスタをポーリングする。次に、ブロック１２０８は、ビットが現在クリアされているすべてのＰＣＩスイッチを介した伝送トラフィックを再開することを示す。しかる後、ブロック１２１０は、ＰＣＩルート・スイッチにおけるいずれか１つのレジスタにおけるこのホスト・ノードと関連するいずれかのビットが依然としてセットされているか否かの決定を示す。ＰＣＩルート・スイッチの１つにおけるレジスタにおいて少なくとも１つのビットが依然としてセットされているという決定が行われる場合、プロセスはブロック１２０６に戻る。すべてのＰＣＩルート・スイッチにおけるすべてのビットがクリアされているという決定が行われる場合、プロセスはブロック１２１２によって示されるように終了する。

【００８６】

本発明は、全体的にハードウェアの実施例、全体的にソフトウェアの実施例、またはハードウェアの要素およびソフトウェアの要素の両方を含む実施例の形を取り得る。好適な実施例では、本発明は、ファームウェア、常駐のソフトウェア、マイクロコード等を含む

がそれに限定されないソフトウェアによって具現化される。

【００８７】

さらに、本発明は、コンピュータまたは任意の命令実行システムによってあるいはそれらに関連して使用するためのプログラム・コードを提供するコンピュータ使用可能媒体またはコンピュータ可読媒体からアクセスし得るコンピュータ・プログラムの形を取ることができる。この説明のためには、コンピュータ使用可能媒体またはコンピュータ可読媒体は、命令実行システム、装置、またはデバイスによってあるいはそれらと関連して使用するためのプログラムを含み、保存、通信、伝播、または搬送することができる任意の実態的な装置であることも可能である。

【００８８】

その媒体は、電子的、磁気的、光学式、電磁的、赤外線、または半導体システム（または装置、またはデバイス）あるいは伝送媒体であってもよい。コンピュータ可読媒体の例は、半導体またはソリッド・ステート・メモリ、磁気テープ、取外し可能フロッピ・ディスク、ランダム・アクセス・メモリ（ＲＡＭ）、リード・オンリ・メモリ（ＲＯＭ）、固定磁気ディスク、および光ディスクを含む。光ディスクの現在の例は、コンパクト・ディスク・リード・オンリ・メモリ（ＣＤ－ＲＯＭ）、コンパクト・ディスク・リード／ライト（ＣＤ－Ｒ／Ｗ）、およびＤＶＤを含む。

【００８９】

プログラム・コードを保存し、実行するに適したデータ処理システムは、システム・バスを介してメモリ素子に直接的または間接的に接続された少なくとも１つのプロセッサを含むであろう。メモリ素子は、プログラム・コードの実際の実行中に使用されるローカル・メモリ、大容量記憶装置、および、実行中にコードが大容量記憶装置から検索されなければならない回数を減らすために、プログラム・コードの一時的記憶装置を提供するキャッシュ・メモリを含み得る。

【００９０】

入出力装置（キーボード、ディスプレイ、ポインティング装置等を含むが、それに限定されない）は、直接的にまたは入出力コントローラを介してシステムに接続することが可能である。

【００９１】

ネットワーク・アダプタは、そのシステムに接続してデータ処理システムが介在のプライベート・ネットワークまたは公衆ネットワークを介して他のデータ処理システム、リモート・プリンタ、記憶装置などに接続することを可能する。モデム、ケーブル・モデム、およびイーサネット(登録商標)は、現時点で使用し得るタイプのネットワーク・アダプタである。

【００９２】

本発明の実施例に関する記述は、図解および説明を目的として与えられたものであり、網羅的であることおよび開示された形に発明を限定することを意図するものではない。当業者にとって、多くの修正および変更が明らかであろう。実施例は、本発明の原理を最もよく説明し、考えられる特定の用途に適するような種々の修正を伴う種々の実施例に関して当業者が本発明を理解することを可能にするために、選択され説明された。

【図面の簡単な説明】

【００９３】

【図１】本発明の好適な実施例に従って示された分散型コンピュータ・システムの概略的なブロック図である。

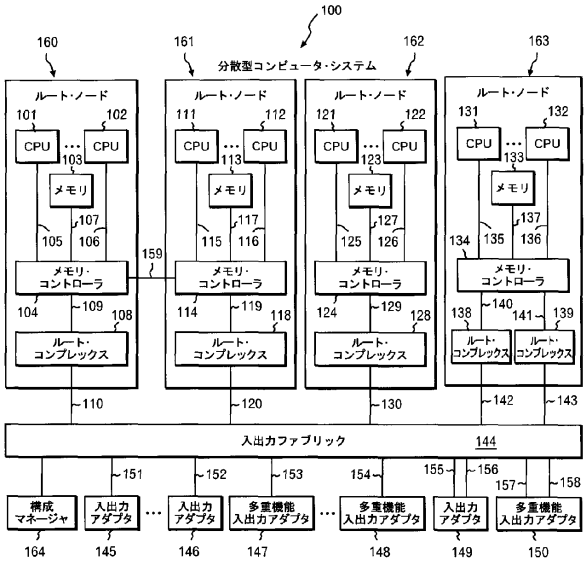
【図２】本発明の実施例を具現化し得る例示的な論理的にパーティション化されたプラットフォームのブロック図である。

【図３】本発明の実施例に従って示された任意のデータ処理システムを具現化するために使用されるデータ処理システムのブロック図である。

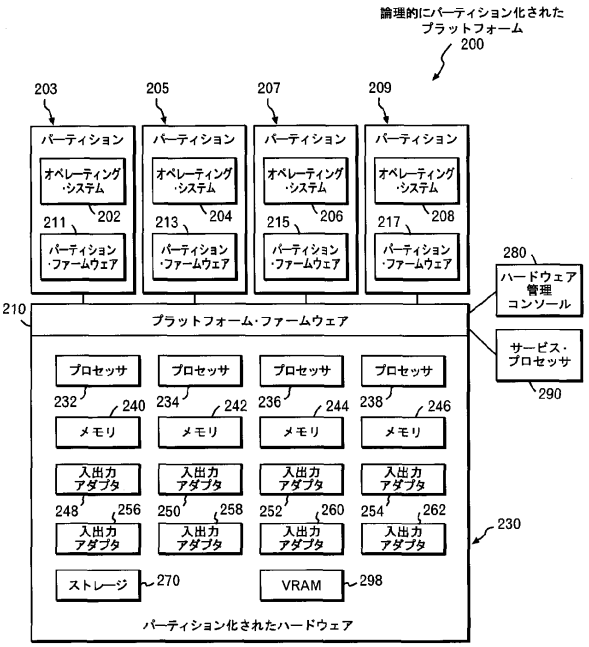
【図４】本発明の実施例に従って、エラーをレポートするために使用されるメッセージ・レポート・パケットの一般的なレイアウトのブロック図である。

【図５】本発明の実施例に従って、１つのＰＣＩルート・スイッチしか含まないＰＣＩスイッチのファブリックを利用して、入出力アダプタのようなアダプタにコンピュータ・システムが接続されるデータ処理環境を示すブロック図である。
【図６】本発明の実施例に従って、複数のＰＣＩルート・スイッチを含むＰＣＩスイッチのファブリックを利用して、入出力アダプタのようなアダプタにコンピュータ・システムが接続されるデータ処理環境を示すブロック図である。
【図７】本発明の実施例に従って、経路指定テーブル・エントリを示すブロック図である。
【図８】本発明の実施例に従って、マスタ制御ノードが経路指定テーブルを生成しかつそれらのテーブルにファブリック・トポロジを移植することを示す高レベルのフローチャートである。
【図９】本発明の実施例に従って、エラー検出ロジックがエラーを検出しかつそのエラーを記述したメッセージ・リクエスト・パッケージを生成および伝送することを示す高レベルのフローチャートである。
【図１０】本発明の実施例に従って、ＰＣＩスイッチがメッセージ・リクエスト・パケットを送るためにおよび故障した入出力アダプタに接続されたポートを介して入出力オペレーションを中断するために経路指定テーブルを利用することを示す高レベルのフローチャートである。
【図１１】本発明の実施例に従って、エラーを解消しないうちは、ファブリック内の潜在的に影響を受けるパスにおけるすべてのトラフィックが中断されかつすべての潜在的に影響を受けるホスト・ノードが、そのエラーが生じたという通知の受信を確認するまで、マスタ制御ノードが待機することを示す高レベルのフローチャートである。
【図１２】本発明の実施例に従って、エラーの影響を受けるすべてのホスト・ノードがそのエラーを確認したということをマスタ制御ノードが信号するまで、ホスト・ノードが、そのエラーの影響を受けるファブリックの部分を通してそれのファブリックを中断することを示す高レベルのフローチャートである。

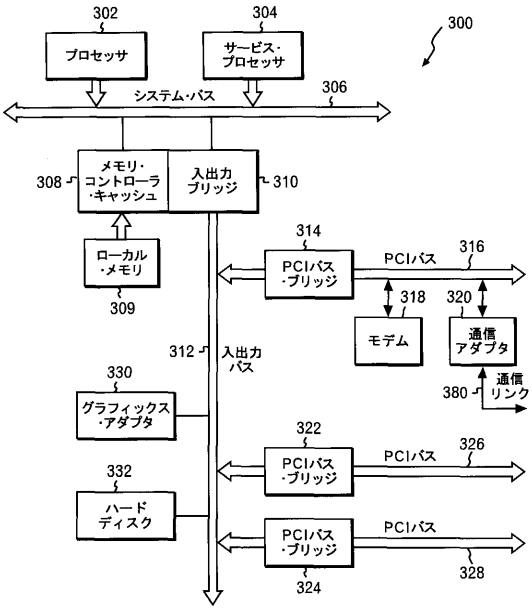
【図１】


100
分散型コンピュータ・システム
160
161
162
163
ルート・ノード
CPU
メモリ
メモリ・コントローラ
ルート・コンプレックス
159
入出力ファブリック
144
構成マネージャ
入出力アダプタ
多重機能入出力アダプタ

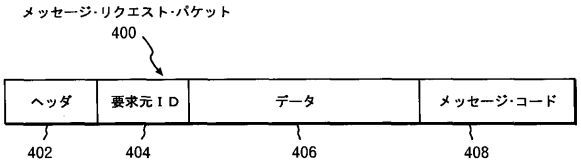
【図２】


論理的にパーティション化されたプラットフォーム
200
パーティション
オペレーティング・システム
パーティション・ファームウェア
プラットフォーム・ファームウェア
ハードウェア管理コンソール
サービス・プロセッサ
プロセッサ
メモリ
入出力アダプタ
ストレージ
VRAM
パーティション化されたハードウェア
230
290

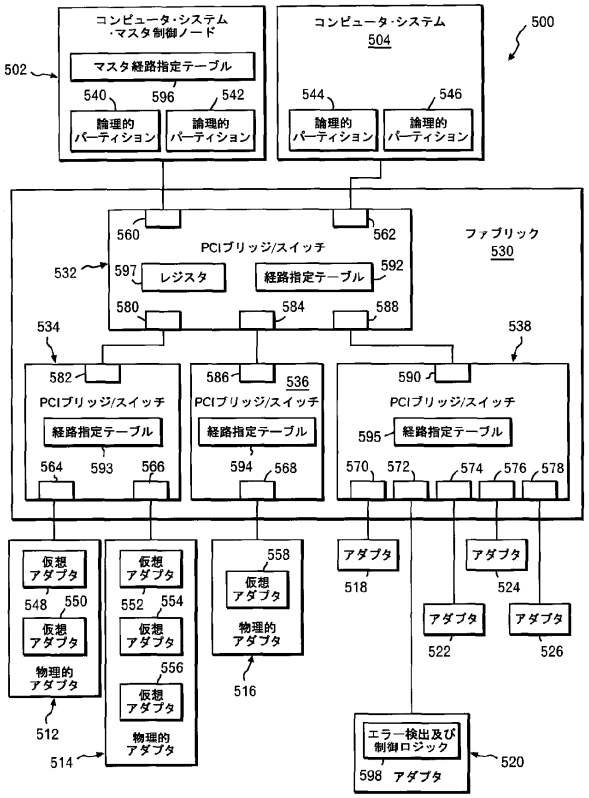
【図３】


300
プロセッサ
サービス・プロセッサ
システム・バス
メモリ・コントローラ・キャッシュ
入出力ブリッジ
ローカル・メモリ
PCIバス・ブリッジ
PCIバス
モデム
通信アダプタ
通信リンク
入出力バス
グラフィックス・アダプタ
ハードディスク

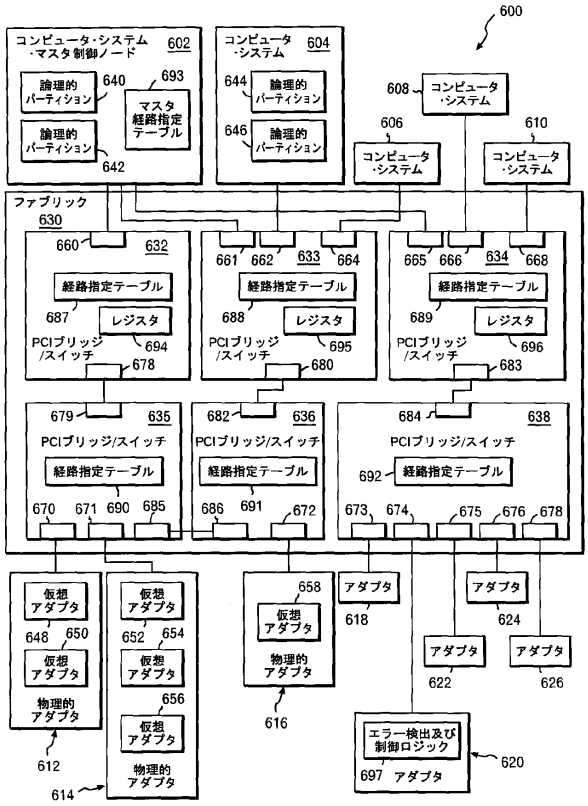
【図４】


メッセージ・リクエスト・パケット
400
ヘッダ
要求元ＩＤ
データ
メッセージ・コード

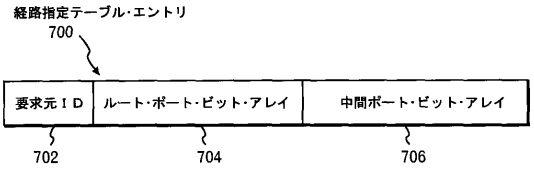
【図５】


500
コンピュータ・システム・マスタ制御ノード
コンピュータ・システム
504
マスタ経路指定テーブル
論理的パーティション
PCIブリッジ/スイッチ
ファブリック
530
レジスタ
経路指定テーブル
仮想アダプタ
物理的アダプタ
アダプタ
エラー検出及び制御ロジック

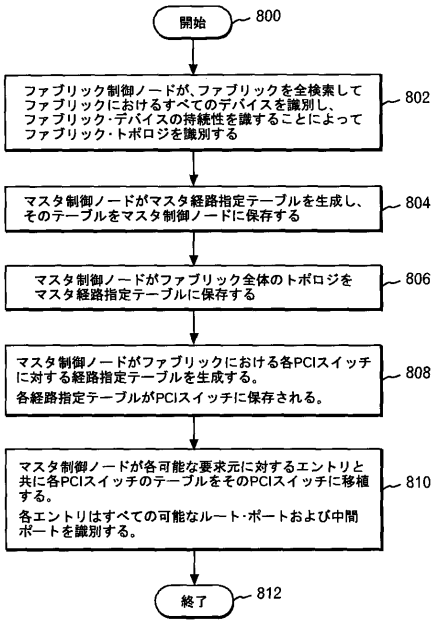
【図６】



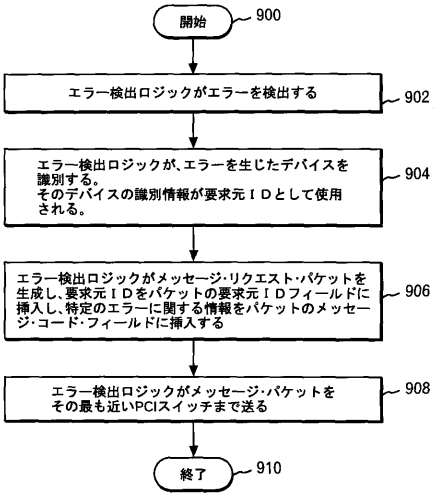
【図７】



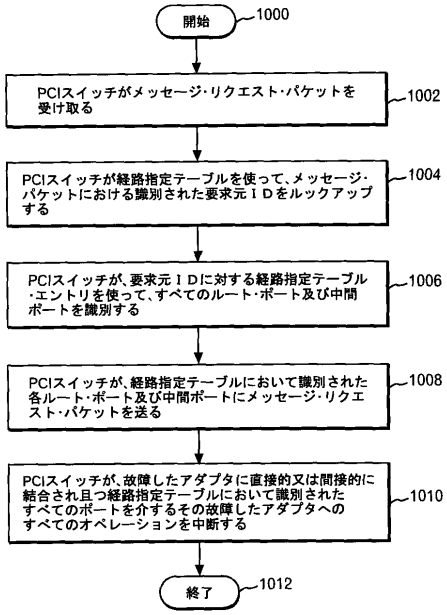
【図８】



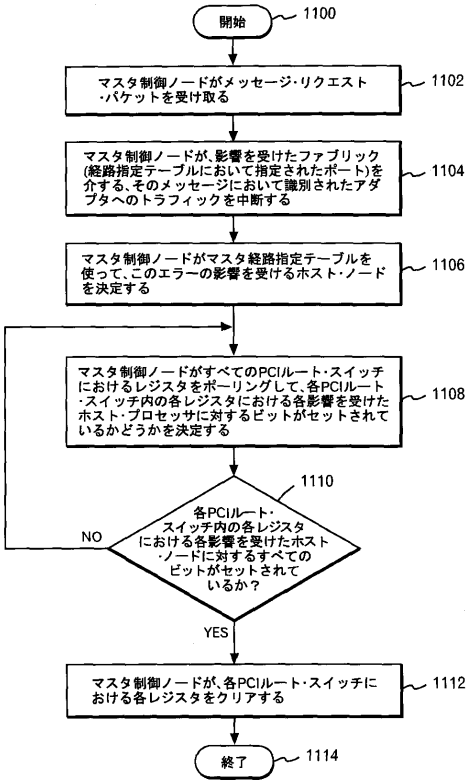
【図９】



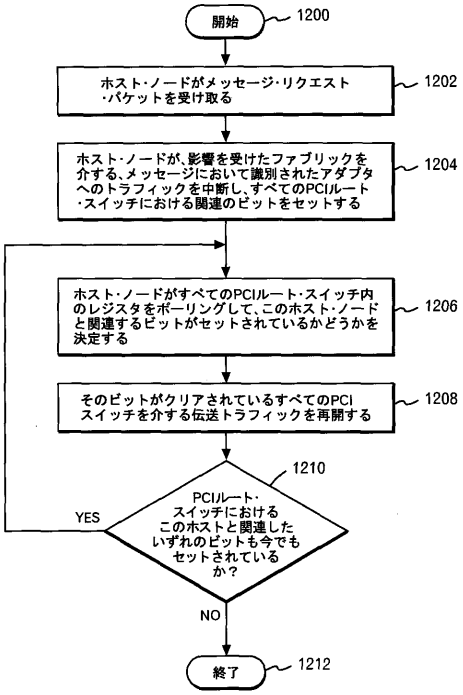
【図１０】



【図１１】


開始
1100
マスタ制御ノードがメッセージ・リクエスト・パケットを受け取る
1102
マスタ制御ノードが、影響を受けたファブリック(経路指定テーブルにおいて指定されたポート)を介する、そのメッセージにおいて識別されたアダプタへのトラフィックを中断する
1104
マスタ制御ノードがマスタ経路指定テーブルを使って、このエラーの影響を受けるホスト・ノードを決定する
1106
マスタ制御ノードがすべてのPCIルート・スイッチにおけるレジスタをポーリングして、各PCIルート・スイッチ内の各レジスタにおける各影響を受けたホスト・プロセッサに対するビットがセットされているかどうかを決定する
1108
1110
各PCIルート・スイッチ内の各レジスタにおける各影響を受けたホスト・ノードに対するすべてのビットがセットされているか？
NO
YES
マスタ制御ノードが、各PCIルート・スイッチにおける各レジスタをクリアする
1112
終了
1114

【図１２】


開始
1200
ホスト・ノードがメッセージ・リクエスト・パケットを受け取る
1202
ホスト・ノードが、影響を受けたファブリックを介する、メッセージにおいて識別されたアダプタへのトラフィックを中断し、すべてのPCIルート・スイッチにおける関連のビットをセットする
1204
ホスト・ノードがすべてのPCIルート・スイッチ内のレジスタをポーリングして、このホスト・ノードと関連するビットがセットされているかどうかを決定する
1206
そのビットがクリアされているすべてのPCIスイッチを介する伝送トラフィックを再開する
1208
1210
PCIルート・スイッチにおけるこのホストと関連したいずれのビットも今でもセットされているか？
YES
NO
終了
1212

フロントページの続き

(74)代理人 100086243
弁理士 坂口 博
(72)発明者 レナト・ジェイ・レシコ
アメリカ合衆国７８７５９、テキサス州オースチン、ウィネペグ・コーブ ６７０７
(72)発明者 スティーブン・ウェイド・ハンター
アメリカ合衆国２７６０６、ノース・カロライナ州ローリー、ダッチ・クリーク・ドライブ ５７０９
(72)発明者 ウィリアム・トッド・ボイド
アメリカ合衆国１２６０３、ニューヨーク州ポーキプシー、カーメン・ドライブ ３７
(72)発明者 スティーブン・マーク・サーバー
アメリカ合衆国７８７１７、テキサス州オースチン、エフライム・ロード ８３０８
(72)発明者 マダリン・ヴェガ
アメリカ合衆国７８７２７、テキサス州オースチン、メイズ・ベンド・ドライブ １９０１
(72)発明者 ウィリアム・ガーヴィン・ホランド
アメリカ合衆国２７５１１、ノース・カロライナ州ケアリー、トリデント・コート １０５
(72)発明者 ダグラス・エム・フライムス
アメリカ合衆国１００２５、ニューヨーク州ニューヨーク、セントラル・パーク・ウェスト アパートメント・ナンバー９エフ ４００

審査官 木村 貴俊

(56)参考文献 特開２００４－３４２１０９（ＪＰ，Ａ）
特開平１０－３２６２６１（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ ３／０６－３／０８
Ｇ０６Ｆ １１／３０
Ｇ０６Ｆ １３／００－１３／４２