(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第4610307号
(P4610307)
(45)発行日　平成23年1月12日(2011.1.12)　(24)登録日　平成22年10月22日(2010.10.22)

(51)Int.Cl.　F I
G06F 13/10 (2006.01)　G06F 13/10 330B

請求項の数 20　(全 17 頁)

(21)出願番号　特願2004-329651 (P2004-329651)
(22)出願日　平成16年11月12日 (2004.11.12)
(65)公開番号　特開2005-174307 (P2005-174307A)
(43)公開日　平成17年6月30日 (2005.6.30)
審査請求日　平成19年11月12日 (2007.11.12)
(31)優先権主張番号　10/734,450
(32)優先日　平成15年12月12日 (2003.12.12)
(33)優先権主張国　米国 (US)

(73)特許権者　500046438
マイクロソフト　コーポレーション
アメリカ合衆国　ワシントン州　９８０５２－６３９９　レッドモンド　ワン　マイクロソフト　ウェイ
(74)代理人　100077481
弁理士　谷　義一
(74)代理人　100088915
弁理士　阿部　和夫
(72)発明者　エリック　トラウト
アメリカ合衆国　９８０５２　ワシントン州　レッドモンド　ワン　マイクロソフト　ウェイ　マイクロソフト　コーポレーション内

最終頁に続く

(54)【発明の名称】実際のハードウェアベースのデバイスおよび理想化されたハードウェアベースのデバイスにおけるバイモーダルデバイス仮想化のためのシステムおよび方法

(57)【特許請求の範囲】
【請求項１】
ゲストオペレーティングシステムとホストオペレーティングシステムとの間の通信を可能とする方法であって、
前記ゲストオペレーティングシステムと前記ホストオペレーティングシステムとの間にバイモーダルデバイスが接続されている場合、前記ゲストオペレーティングシステムのドライバが、前記バイモーダルデバイスの動作モードを識別するステップと、
ここで、前記バイモーダルデバイスは、第１の動作モードでは前記ホストオペレーティングシステムに接続されたハードウェアデバイスが有する所定のオリジナルオペレーション機能を実行するためのハードウェア仮想デバイスとして、第２の動作モードでは前記所定のオリジナルオペレーション機能に追加された追加オペレーション機能をも実行するための理想化されたハードウェア仮想デバイスとして選択的に動作し、ここで、該理想化されたハードウェア仮想デバイスは、前記ゲストオペレーティングシステムと前記ホストオペレーティングシステムとの間の通信を可能にする設計された仮想デバイスとして構成され、該設計された仮想デバイスは、所定の実際の又は設計された物理デバイスに対応する論理デバイスであり、該論理デバイスは、ソフトウェアとしての所定のドライバに従って前記実際の又は設計された物理デバイスとして動作するものであり、
前記ゲストオペレーティングシステムのドライバが、前記第２の動作モードで動作する前記バイモーダルデバイスと通信できないときは、前記ハードウェアデバイスが、前記追加オペレーション機能を実行することなく前記所定のオリジナルオペレーション機能を実

行できるように、前記バイモーダルデバイスは、前記ハードウェア仮想デバイスとして選択的に動作するステップと、
前記ゲストオペレーティングシステムのドライバが、前記第２の動作モードで動作する前記バイモーダルデバイスと通信できるときは、前記ハードウェアデバイスが、前記所定のオリジナルオペレーション機能の実行と共に前記追加オペレーション機能の実行も同時にできるように、前記バイモーダルデバイスは、前記理想化されたハードウェア仮想デバイスとして選択的に動作するステップと
を具えたことを特徴とする方法。
【請求項２】
前記第２の動作モードの機能は、前記第１の動作モードのオリジナルオペレーション機能を拡張することを特徴とする請求項１記載の方法。
【請求項３】
前記第２の動作モードの機能は、前記第１の動作モードのオリジナルオペレーション機能から独立していることを特徴とする請求項１記載の方法。
【請求項４】
前記第２の動作モードの機能は、前記第１の動作モードのオリジナルオペレーション機能をディスエーブルにすることを特徴とする請求項３記載の方法。
【請求項５】
前記第２の動作モードの機能は、前記第１の動作モードのオリジナルオペレーション機能の一部をディスエーブルにすることを特徴とする請求項３記載の方法。
【請求項６】
前記第２の動作モードは、前記バイモーダルデバイスにおける前記ハードウェア仮想デバイスのレジスタ内の少なくとも１つのビットを使用することによりイネーブルにされることを特徴とする請求項１記載の方法。
【請求項７】
前記第２の動作モードは、前記バイモーダルデバイスにおける１つまたは複数の前記ハードウェア仮想デバイスによる利用のため特に作成されたレジスタ内の少なくとも１つのビットを使用することによりイネーブルにされることを特徴とする請求項１記載の方法。
【請求項８】
前記第２の動作モードは、前記バイモーダルデバイスにおける少なくとも１つのレジスタ内の値を変更する所定のコマンドまたはデータのシーケンスを使用することによりイネーブルにされることを特徴とする請求項１記載の方法。
【請求項９】
前記第２の動作モードは、ゲストオペレーティングシステム環境内にインストールされた第２の動作モードのドライバを使用することによりイネーブルにされ、
前記第２の動作モードのドライバが存在していない場合、第１の動作モードのドライバが代わりにイネーブルにされることを特徴とする請求項１記載の方法。
【請求項１０】
コンピュータに、請求項１ないし９のいずれかに記載の方法を実行させることが可能な命令を有するコンピュータプログラム。
【請求項１１】
請求項１０記載のコンピュータプログラムを有するコンピュータ読取り可能な記録媒体。
【請求項１２】
ゲストオペレーティングシステムとホストオペレーティングシステムとの間の通信を可能とするシステムであって、
前記ゲストオペレーティングシステムと前記ホストオペレーティングシステムとの間にバイモーダルデバイスが接続されている場合、前記ゲストオペレーティングシステムのドライバが、前記バイモーダルデバイスの動作モードを識別する手段と、
ここで、前記バイモーダルデバイスは、第１の動作モードでは前記ホストオペレーティ

ングシステムに接続されたハードウェアデバイスが有する所定のオリジナルオペレーション機能を実行するためのハードウェア仮想デバイスとして、第２の動作モードでは前記所定のオリジナルオペレーション機能に追加された追加オペレーション機能をも実行するための理想化されたハードウェア仮想デバイスとして選択的に動作し、ここで、該理想化されたハードウェア仮想デバイスは、前記ゲストオペレーティングシステムと前記ホストオペレーティングシステムとの間の通信を可能にする設計された仮想デバイスとして構成され、該設計された仮想デバイスは、所定の実際の又は設計された物理デバイスに対応する論理デバイスであり、該論理デバイスは、ソフトウェアとしての所定のドライバに従って前記実際の又は設計された物理デバイスとして動作するものであり、

前記ゲストオペレーティングシステムのドライバが、前記第２の動作モードで動作する前記バイモーダルデバイスと通信できないときは、前記ハードウェアデバイスが、前記追加オペレーション機能を実行することなく前記所定のオリジナルオペレーション機能を実行できるように、前記バイモーダルデバイスは、前記ハードウェア仮想デバイスとして選択的に動作する手段と、

前記ゲストオペレーティングシステムのドライバが、前記第２の動作モードで動作する前記バイモーダルデバイスと通信できるときは、前記ハードウェアデバイスが、前記所定のオリジナルオペレーション機能の実行と共に前記追加オペレーション機能の実行も同時にできるように、前記バイモーダルデバイスは、前記理想化されたハードウェア仮想デバイスとして選択的に動作する手段と

を具えたことを特徴とするシステム。

【請求項１３】

前記第２の動作モードの機能は、前記第１の動作モードのオリジナルオペレーション機能を拡張することを特徴とする請求項１２記載のシステム。

【請求項１４】

前記第２の動作モードの機能は、前記第１の動作モードのオリジナルオペレーション機能から独立していることを特徴とする請求項１２記載のシステム。

【請求項１５】

前記第２の動作モードの機能は、前記第１の動作モードのオリジナルオペレーション機能をディスエーブルにすることを特徴とする請求項１４記載のシステム。

【請求項１６】

前記第２の動作モードの機能は、前記第１の動作モードのオリジナルオペレーション機能の一部をディスエーブルにすることを特徴とする請求項１４記載のシステム。

【請求項１７】

前記第２の動作モードは、前記バイモーダルデバイスにおける前記ハードウェア仮想デバイスのレジスタ内の少なくとも１つのビットを使用することによりイネーブルにされることを特徴とする請求項１２記載のシステム。

【請求項１８】

前記第２の動作モードは、前記バイモーダルデバイスにおける１つまたは複数の前記ハードウェア仮想デバイスによる利用のため特に作成されたレジスタ内の少なくとも１つのビットを使用することによりイネーブルにされることを特徴とする請求項１２記載のシステム。

【請求項１９】

前記第２の動作モードは、前記バイモーダルデバイスにおける少なくとも１つのレジスタ内の値を変更する所定のコマンドまたはデータのシーケンスを使用することによりイネーブルにされることを特徴とする請求項１２記載のシステム。

【請求項２０】

前記第２の動作モードは、ゲストオペレーティングシステム環境内にインストールされた第２の動作モードのドライバを使用することによりイネーブルにされ、

前記第２の動作モードのドライバが存在していない場合、第１の動作モードのドライバが代わりにイネーブルにされることを特徴とする請求項１２記載のシステム。

【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、一般に、コンピュータシステム内の仮想デバイス(virtual devices)に関するものであり、より詳細には、必要に応じてハードウェア仮想デバイス(hardware virtual device)または理想化された仮想デバイス(idealized virtual device)として選択的に動作するコンピュータシステム内の機能強化された仮想デバイスに関するものである。
【背景技術】
【０００２】
仮想デバイスは、ソフトウェアで実現され、ある種の実際の(actual)または理想化された(idealized)物理デバイス(physical device)に対応する論理デバイス(logical device)である。仮想デバイスをモデル化する方法としては、一般に、既存のハードウェアを直接モデル化する「ハードウェア仮想デバイス(hardware virtual device)」手法(approach)と、物理ハードウェアの単なる反映(reflection)ではなく、ＶＭ環境(environment)向けに最適化(optimize)された「理想化された仮想デバイス(idealized virtual device)」手法の２つがある。
【０００３】
ハードウェア仮想デバイス手法には互換性に関して利点がある。すなわち、仮想デバイスはあらゆる面で実デバイス(real device)とまったく同じように動作するため、そのデバイスと相互作用するように設計されているソフトウェア（例えば、ドライバ）は変更なしでハードウェア仮想デバイスと共に動作する。しかし、ハードウェア仮想デバイスは、パフォーマンスに関しては不利である。すなわち、一般的にハードウェア設計者は仮想化に関する問題を考慮してないため物理ハードウェア(physical hardware)は著しいオーバーヘッドコスト（および不効率）を被ることなく仮想デバイスでエミュレート(emulate)することが困難なことが多く、したがって、ハードウェア仮想デバイスは、しばしば、対応するハードウェア実デバイス(real hardware counterpart)に比べて目立って低速(slower)である。
【０００４】
一方、理想化された仮想デバイス(idealized virtual device)では、開発者は、実施しやすくかつ使用効率のよい仮想デバイスをかなり自由に設計することができる。理想化された仮想デバイスの設計は物理ハードウェア設計により課される制限に従う必要がないため、理想化された仮想デバイスはＶＭ環境内での使用に合わせて最適化できる。さらに、理想化された仮想デバイスの開発者は、正しく動作するために既存のソフトウェアが依拠する可能性のある微妙な副作用（タイミング、状態変化(state change)など）を懸念しなくてもよい。その上、開発者は、実際には存在していないハードウェアに類似する理想化された仮想デバイス、例えば、ゲストシステムとホストシステムとの間の通信を可能にする仮想デバイスを作成することもできる。しかし、欠点は、仮想デバイス(virtual device)が実際にはあらゆる点で実デバイス(real device)とまったく同じように動作するわけではないため、理想化された仮想デバイス手法(idealized virtual device approach)に関する互換性問題(compatibility issues)が発生する可能性があり、またその物理デバイスと相互作用するように設計されているソフトウェア（例えば、ドライバ(driver)）が、変更(modification)なしでは理想化された仮想デバイスと共に正常にまたはまったく動作しない可能性があるという点である。
【発明の開示】
【発明が解決しようとする課題】
【０００５】
従って、上述した２つの既存の手法の利点を備え、ほとんど制限がない、仮想化デバイスの手法が求められている。
【課題を解決するための手段】
【０００６】

本発明を適用した様々な実施形態は、それぞれの弱みを補いながらハードウェアおよび理想化されたデバイス仮想化の手法(idealized device virtualization approaches)の相対的な強み(relative strength)を組み合わせたバイモーダル仮想デバイス手法(bimodal virtual device approach)（すなわち、「バイモーダルデバイス(bimodal devices)」）を対象とする。いくつかの実施形態では、バイモーダルデバイスは、実ハードウェア(real piece of hardware)に主に基づく仮想デバイスであり、ゲスト環境(guest environment)で実行(running)中のソフトウェアと広範(broad degree)な互換性(compatibility)を実現する（ハードウェアデバイス仮想化手法と類似）。しかし、ハードウェア仮想デバイスにつきまとう不十分なパフォーマンスという問題を克服するために、これらの実施形態では、オリジナルのハードウェアベースのデバイスには見られない理想化された(idealized)「高性能モード(high-performance mode)」をも提供する。オリジナルのハードウェアデバイス(original hardware device)と相互作用(interact)するように開発され、高性能モードを認識(unaware)しない（および使用(use)できない）ソフトウェアドライバ(software driver)（およびその他のソフトウェア）では、そのまま「レガシーモード(legacy mode)」（ハードウェア仮想化(hardware virtualization)）を使用し続けるが、ゲストソフトウェアの機能強化版(enhanced version)は高性能モード（理想化された仮想化）を認識して利用することができる。

【発明を実施するための最良の形態】

【０００７】

好ましい実施形態の詳細な説明は、付属の図面と併せて読むことより、よく理解できる。本発明は、以下に開示されている特定の方法および手段に限定されるものではない。すなわち、以下の説明自体は特許の範囲を制限することを意図していない。むしろ、異なるステップまたは本書で説明されているステップと類似のステップの組み合わしを含むことも可能である。さらに、「ステップ」という用語は、本明細書では、採用されている方法の異なる要素を意味するために使用される場合があるが、この用語は、個々のステップの順序が明示的に説明されていない限り、かつ説明されている場合を除き、本明細書で開示されている様々なステップの間に特定の順序があることを暗示するものではない。

【０００８】

コンピュータ環境(computer environment)

本発明の様々な実施形態は、コンピュータ上で実行可能である。図１および以下の説明は、本発明を実施することができる適切なコンピューティング環境について簡潔に述べた一般的な説明である。本発明の一実施形態として、クライアントワークステーションまたはサーバなどのコンピュータによって実行されるプログラムモジュールなどコンピュータ実行可能命令の一般的状況において説明してある。一般に、プログラムモジュールは、特定のタスクを実行する、または特定の抽象データ型を実施するルーチン、プログラム、オブジェクト、コンポーネント、データ構造などを含む。さらに、当業者であれば、本発明は、ハンドヘルドデバイス、マルチプロセッサシステム、マイクロプロセッサベースのまたはプログラム可能な家電、ネットワークＰＣ、ミニコンピュータ、メインフレームコンピュータなどをはじめとする、他のコンピュータシステム構成でも実施することができることを理解するであろう。また、本発明は、通信ネットワークを通じてリンクされているリモート処理デバイスによりタスクが実行される分散コンピューティング環境で実施することもできる。分散コンピューティング環境では、プログラムモジュールは、ローカルとおよびリモートの両方のメモリ記憶デバイス内に配置することができる。

【０００９】

図１に示されているように、汎用コンピューティングシステムの一例は、処理ユニット２１、システムメモリ２２、およびシステムメモリを含む様々なシステムコンポーネントを処理ユニット２１に結合するシステムバス２３を備える、従来のパーソナルコンピュータ２０などを含む。システムバス２３には、メモリバスまたはメモリコントローラ、周辺機器バス、および様々なバスアーキテクチャを使用するローカルバスを含む数種類のバス構造がある。システムメモリは、読み取り専用メモリ（ＲＯＭ）２４およびランダムアク

セスメモリ（ＲＡＭ）２５を含む。起動時などにパーソナルコンピュータ２０内の要素間の情報伝送を助ける基本ルーチンを含む基本入出力システム２６（ＢＩＯＳ）は通常、ＲＯＭ２４に格納される。パーソナルコンピュータ２０は、さらに、図に示されていないハードディスクへの読み書きを行うためのハードディスクドライブ２７、取り外し可能磁気ディスク２９への読み書きを行うための磁気ディスクドライブ２８、およびＣＤ－ＲＯＭまたはその他の光媒体などの取り外し可能光ディスク３１への読み書きを行うための光ディスクドライブ３０を備えることができる。ハードディスクドライブ２７、磁気ディスクドライブ２８、および光ディスクドライブ３０は、ハードディスクドライブインターフェース３２、磁気ディスクドライブインターフェース３３、および光ドライブインターフェース３４によりそれぞれシステムバス２３に接続される。ドライブおよび関連コンピュータ可読媒体は、コンピュータ可読命令、データ構造体、プログラムモジュール、およびパーソナルコンピュータ２０用のその他のデータを格納する不揮発性ストレージを備える。本発明で説明されている環境例ではハードディスク、取り外し可能磁気ディスク２９、および取り外し可能光ディスク３１を採用しているが、当業者であれば、磁気カセット、フラッシュメモリカード、デジタルビデオディスク、ベルヌーイカートリッジ、ランダムアクセスメモリ（ＲＡＭ）、読み取り専用メモリ（ＲＯＭ）などのコンピュータからアクセス可能なデータを格納できる他のタイプのコンピュータ可読媒体もこの動作環境で使用できることを理解するであろう。

【００１０】

オペレーティングシステム３５、１つまたは複数のアプリケーションプログラム３６、その他のプログラムモジュール３７およびプログラムデータ３８を含む多くのプログラムモジュールは、ハードディスク、磁気ディスク２９、光ディスク３１、ＲＯＭ２４、またはＲＡＭ２５に格納されることができる。ユーザはキーボード４０およびポインティングデバイス４２などの入力デバイスを通じてパーソナルコンピュータ２０にコマンドおよび情報を入力することができる。他の入力デバイス（図示せず）としては、マイク、ジョイスティック、ゲームパッド、衛星パラボラアンテナ、スキャナなどがある。これらの入力デバイスやその他の入力デバイスは、システムバスに結合されているシリアルポートインターフェース４６を介して処理ユニット２１に接続されることが多いが、パラレルポート、ゲームポート、またはユニバーサルシリアルバス（ＵＳＢ）などの他のインターフェースにより接続することもできる。モニタ４７またはその他の種類の表示デバイスも、ビデオアダプタ４８などのインターフェースを介してシステムバス２３に接続される。パーソナルコンピュータは、通常、モニタ４７の他に、スピーカおよびプリンタなど、他の周辺出力デバイス（図示せず）を備える。図１のシステム例は、さらに、ホストアダプタ５５、ＳＣＳＩ(Small Computer System interface)バス５６、およびＳＣＳＩバス５６に接続されている外部記憶デバイス６２を含む。

【００１１】

パーソナルコンピュータ２０は、リモートコンピュータ４９などの１つまたは複数のリモートコンピュータへの論理接続を使用することによりネットワーク環境で動作することも可能である。リモートコンピュータ４９は、他のパーソナルコンピュータ、サーバ、ルータ、ネットワークＰＣ、ピアデバイス、またはその他の共通ネットワークノードでもよく、通常は、パーソナルコンピュータ２０に関係する上述の要素の多くまたはすべてを含むが、メモリ記憶デバイス５０だけが図１に例示されている。図１で説明されている論理接続は、ローカルエリアネットワーク（ＬＡＮ）５１とワイドエリアネットワーク（ＷＡＮ）５２を含む。このようなネットワーキング環境は、オフィス、企業全体にわたるコンピュータネットワーク、イントラネット、およびインターネットでは一般的なものである。

【００１２】

ＬＡＮネットワーキング環境で使用する場合は、パーソナルコンピュータ２０はネットワークインターフェースまたはアダプタ５３を介してＬＡＮ５１に接続される。ＷＡＮネットワーキング環境で使用する場合、パーソナルコンピュータ２０は、通常、モデム５４

またはインターネットなどのワイドエリアネットワーク５２上で通信を確立するためのその他の手段を備える。モデム５４は、内蔵でも外付けでもよく、シリアルポートインターフェース４６を介してシステムバス２３に接続される。ネットワーク環境では、パーソナルコンピュータ２０またはその一部に関して述べたプログラムモジュールは、リモートメモリ記憶デバイスに格納されることができる。図に示されているネットワーク接続は例であり、コンピュータ間に通信リンクを確立するのに他の手段が使用できることは理解されるであろう。さらに、本発明の多数の実施形態は、コンピュータ化システムに特によく適しているが、本明細書いずれの内容も、本発明をそのような実施形態に制限することを意図されていない。

【００１３】

仮想マシン(virtual machines)

概念的な観点からは、コンピュータシステムは、一般に、ハードウェアの基礎となる層で実行されるソフトウェアの１つまたは複数の層を含む。この層化(layering)は、抽象化(abstraction)のため行われる。ソフトウェアの所定の層に対しインターフェースを定義すると、その層はその上の他の層により異なる方式で実施することができる。適切に設計されたコンピュータシステムであれば、それぞれの層はその下の中間層のみ認識する（およびそれのみに依存する）。これにより、層すなわち「スタック(stack)」（複数の隣接する層）は前記層またはスタックの上の層に悪影響を及ぼすことなく置き換えられることができる。例えば、ソフトウェアアプリケーション（上位層）は、通常、ファイルを何らかの形の恒久的ストレージに書き込むために、オペレーティングシステムの下位のいくつかのレベル（下位層）に依存しており、これらのアプリケーションでは、データを磁気ディスクに書き込むのか、ハードドライブに書き込むのか、またはネットワークフォルダに書き込むのかの違いを理解する必要はない。このような下位の層がファイルの書き込みを行う新しいオペレーティングシステムコンポーネントと置き換えられても、上位層のソフトウェアアプリケーションのオペレーションは影響を受けない。

【００１４】

層化されたソフトウェア(layered software)の持つ柔軟性(flexibility)は、仮想マシン（ＶＭ:virtual machine）が実際には別のソフトウェア層である仮想ハードウェア層を提示することを可能にする。このため、ＶＭはその上のソフトウェア層が専用のプライベートコンピュータシステム上で実行中だと前記ソフトウェア層に思いこませることができ、したがってＶＭを使用すると、複数の「ゲストシステム(guest system)」を単一の「ホストシステム(host system)」上で同時に実行することができる。

【００１５】

図２は、コンピュータシステム内でエミュレートされる動作環境のハードウェアおよびソフトウェアアーキテクチャの論理的層化を表す図である。エミュレーションプログラム９４は、ホストオペレーティングシステムおよび／またはハードウェアアーキテクチャ９２上で実行される。エミュレーションプログラム９４は、ゲストハードウェアアーキテクチャ９６およびゲストオペレーティングシステム９８をエミュレートする。さらに、ソフトウェアアプリケーション１００は、ゲストオペレーティングシステム９８上で実行される。図２のエミュレートされている動作環境では、エミュレーションプログラム９４が動作するので、ソフトウェアアプリケーション１００は、ホストオペレーティングシステムおよびハードウェアアーキテクチャ９２と一般に互換性のないオペレーティングシステム上で実行するように設計されているとしてもコンピュータシステム９０上で実行することができる。

【００１６】

図３Ａは、物理(physical)コンピュータハードウェア１０２のすぐ上で実行されるホストオペレーティングシステムソフトウェア層１０４を備える仮想化されたコンピューティングシステムを例示しており、ホストオペレーティングシステム（ホストＯＳ）１０４は、ホストＯＳが仮想化しようとしているハードウェアと同じであるインターフェースを公開することによりマシンのすべてのリソースを仮想化する（このため、その上で実行され

ているオペレーティングシステム層によりホストＯＳが気づかないままにできる）。
【００１７】
他の実施形態として、仮想マシンモニタ、すなわちＶＭＭは、ソフトウェア層１０４’は、ホストオペレーティングシステム１０４”の代わりに、またはそれと共に、実行中の可能性があり、後者のオプションは図３Ｂに示されている。簡単のため、これ以降のすべての考察（特に、ホストオペレーティングシステム１０４については）は、図３Ａに例示されている実施形態を対象とするものとする。しかし、このような考察のあらゆる面が、図３Ｂの実施形態に等しく当てはまるものとし、図３ＢのＶＭＭ１０４’は本質的に、機能レベルにおいて、後述の図３Ａのホストオペレーティングシステム１０４の役割を置き換える。
【００１８】
図３Ａを再び参照すると、ホストＯＳ　１０４（またはＶＭＭ　１０４’）は、２つの仮想マシン（ＶＭ）実施、例えば仮想化されたＩｎｔｅｌ　３８６プロセッサとすることができるＶＭ　Ａ　１０８および例えばＭｏｔｏｒｏｌａ　６８０Ｘ０ファミリのプロセッサのうちの１つの仮想化されたプロセッサとすることができるＶＭ　Ｂ　１１０である。ＶＭ　１０８および１１０のそれぞれの上に、ゲストオペレーティングシステム（ゲストＯＳ）Ａ　１１２およびＢ　１１４がある。ゲストＯＳ　Ａ　１１２の上で、２つのアプリケーション、すなわちアプリケーションＡ１　１１６およびアプリケーションＡ２　１１８が実行されており、ゲストＯＳ　Ｂ　１１４の上に、アプリケーションＢ１　１２０がある。
【００１９】
仮想デバイス(virtual devices)
物理ハードウェアデバイスに関して、ＶＭはソフトウェアアプリケーションが前記ハードウェアデバイスを利用するためのいくつかのオプションを用意する。いくつかＶＭシステムでは、ハードウェアデバイス（ハードドライブまたはネットワークアダプタなど）は、単一のＶＭ、およびＶＭがそのハードウェアデバイスを利用できるという点で実行されているソフトウェア（専用デバイスアーキテクチャ）のみに割り当てられることができる。しかし、この設計では、異なるＶＭ間だけでなく、ＶＭとホストシステムとの間でもハードウェアデバイスの共有が妨げられる。このような理由から、多くのＶＭ実施形態では、代わりに「仮想デバイス」を使用する。
【００２０】
仮想デバイスは、ソフトウェアで実現(implement)され、ある種の実際の(actual)または理想化された(idealized)物理デバイスに対応する論理デバイス(logical device)である。仮想デバイスは、一般に、例えばレジスタ設定、バッファリングされたデータ、保留コマンドキューなどを含む固有の一組のデバイス状態を保有する。制限のない例を使用すると、仮想ネットワークアダプタに関して、仮想アダプタに関連するソフトウェアはネットワークアダプタがＶＭ内で実行中のコードによりアクセスされるときに呼び出され、その後、仮想アダプタは実アダプタカードに適合する方法でコマンドに応答することが可能である。このようにして、ＶＭ内で実行中のコードは、それが実アダプタではなく仮想アダプタ(virtual adapter)と「交信している(talking to)」ということを意識せず、仮想アダプタは直接または間接的に実アダプタ(real adapter)と相互作用し、その結果、実アダプタは望み通りの動作をする。
【００２１】
多くの場合、仮想デバイスへの要求は対応するホストデバイス上にマッピングされる。例えば、仮想ネットワークアダプタから送信されるネットワーキングパケットは、ホストにインストールされた実ネットワークアダプタの１つへ経路指定されることができる。この点に関して、仮想デバイスアーキテクチャは、前述の専用デバイスアーキテクチャのと似た機能を備える。しかし、専用デバイスアーキテクチャとは異なり、仮想デバイスアーキテクチャでは、複数の仮想デバイスが単一のホストデバイス(single host device)（実デバイス(real device)）にマッピング(map)されることができる。例えば、３つの独立の

ＶＭがあるとすると、それぞれ仮想ネットワークアダプタを備え、３つの仮想アダプタはすべて単一のホストアダプタを共有することができるが、専用デバイスアーキテクチャでは、１つのＶＭのみがホストアダプタを使用することができる。

【００２２】

図４Ａは、さらにドライバ、仮想デバイス、およびハードウェアを含む図３の仮想化コンピューティングシステムを例示している。ゲストＯＳ　Ａ　１１２は、ドライバＡ　１２２を備え、仮想マシンＡ　１０８は、ドライバＡ　１２２に対応する仮想デバイスＡ　１２４を備える。同様に、ゲストＯＳ　Ｂ　１１４は、ドライバＢ　１２６を備え、仮想マシンＢ　１１０は、ドライバＢ　１２６に対応する仮想デバイスＢ　１２８を備える。さらに、ホストＯＳ　１０４は、ドライバＸ　１３０を備え、コンピュータハードウェア１０２は、ドライバＸ　１３０に対応するハードウェアデバイスＸ　１３２を備える。

【００２３】

図４Ｂは、ゲストオペレーションシステム上で実行中のアプリケーションからコンピュータハードウェア内のハードウェアデバイスへの動作経路を例示している。アプリケーションＡ１　１１６は、ドライバＡ　１２２とインターフェースし（１３４）、さらにこのドライバは、仮想デバイスＡ　１２４とインターフェースする（１３６）。その後、この仮想デバイスＡ　１２４は、（ホストＯＳ　１０４内の）ドライバＸ　１３０とインターフェースし（１３８）、これはさらに、ハードウェアデバイスＸ　１３２と直接通信する（１４０）。図５は、図４Ｂに示されているようにドライバ、仮想デバイス、およびハードウェアの間の同じ相互関係をさらに簡単にして例示している。

【００２４】

バイモーダルデバイス(bimodal device)

仮想デバイスをモデル化する方法としては、一般に、ハードウェアの既存の個々の部分を直接モデル化する「ハードウェア仮想デバイス(hardware virtual device)」手法(approach)と物理ハードウェア(physical hardware)の単なる反映(reflection)ではない、ＶＭ環境向けに最適化された、「理想化された仮想デバイス(idealized virtual device)」手法の２つがある。ハードウェア仮想デバイス手法には互換性に関する利点がある、すなわち、仮想デバイスはあらゆる点で実デバイスとまったく同じように動作するため、そのデバイスと相互作用するように設計されているソフトウェア（例えば、ドライバ）は変更なしでハードウェア仮想デバイスと共に動作する。しかし、ハードウェア仮想デバイスは、パフォーマンスに関しては不利である、すなわち、一般的にハードウェア設計者は仮想化に関する問題を考慮しないため物理ハードウェアは著しいオーバーヘッドコスト（および不効率）を被ることなく仮想デバイスでエミュレートすることが困難なことが多く、したがって、ハードウェア仮想デバイスは、多くの場合、対応するハードウェア実デバイスに比べて目立って低速である。

【００２５】

一方、理想化された仮想デバイスでは、開発者は、実施しやすくかつ使用効率のよい仮想デバイスをかなり自由に設計することができる。理想化された仮想デバイスの設計は物理ハードウェア設計により課される制限に従う必要がないため、理想化された仮想デバイスはＶＭ環境内での使用に合わせて最適化できる。さらに、理想化された仮想デバイスの開発者は、正しく動作するために既存のソフトウェアが依拠する可能性のある微妙な副作用（タイミング、状態変化など）を懸念しなくてもよい。その上、開発者は、実際には存在していないハードウェアに類似する理想化された仮想デバイス、例えば、ゲストシステムとホストシステムとの間の通信を可能にする仮想デバイスを作成することもできる。しかし、欠点は、仮想デバイスが実際にはあらゆる点で実デバイスとまったく同じように動作するわけではないため、理想化された仮想デバイス手法に関する互換性問題が発生する可能性があり、またその物理デバイスと相互作用するように設計されているソフトウェア（例えば、ドライバ）が、変更なしでは理想化された仮想デバイスと共に正常にまたはまったく動作しない可能性があるという点である。

【００２６】

本発明の様々な実施形態は、それぞれの弱みを補いながらハードウェアおよび理想化されたデバイス仮想化の手法の相対的な強みを組み合わせたバイモーダル仮想デバイス手法（すなわち、「バイモーダルデバイス」）を対象とする。いくつかの実施形態では、バイモーダルデバイスは、ハードウェアの個々の実部分に主に基づく仮想デバイスであり、ゲスト環境で実行中のソフトウェアと広範な互換性を実現する（ハードウェアデバイス仮想化手法と類似）。しかし、ハードウェア仮想デバイスにつきまとう不十分なパフォーマンスという問題を克服するために、これらの実施形態では、オリジナルのハードウェアベースのデバイスには見られない理想化された「高性能モード(high-performance mode)」をも提供する。オリジナルのハードウェアデバイスと相互作用するように開発され、高性能モードを意識しない（および使用できない）ソフトウェアドライバ（およびその他のソフトウェア）では、そのまま「レガシーモード(legacy mode)」（ハードウェア仮想化(hard ware viutualization)）を使用し続けるが、ゲストソフトウェアの機能強化版は高性能モード（理想化された仮想化）を認識して利用することができる。

【００２７】

いくつかの実施形態では、高性能モードは、仮想デバイスのレガシーモードのオリジナル機能とは完全に独立したものとすることができる。他の実施形態として、高性能モードはオリジナル機能の拡張または代替を含むこともできる。いずれにせよ、高性能モードは、レガシーモード機能と無関係であろうと代替であろうと、理想化されたプロセッサ仮想化手法によく似ている。さらに、使用する実施形態に応じて、高性能モードへの切り替えで、レガシーモードを完全にディスエーブル(disable)にする、レガシーモードの一部をディスエーブルにする、かつ／またはレガシーモードの機能を拡張することができることにも留意されたい。

【００２８】

本発明のいくつかの実施形態では、高性能モードは、仮想デバイスのレジスタ内の予約ビット(reserved bit)を使用することによりイネーブル(enable)にすることができるが、他の実施形態では、代わりに、新しいレジスタ（例えば、１つまたは複数の仮想デバイスにより使用するため特に作成されたレジスタ）を追加するか、または何らかの形式のハンドシェーキングを実施し、所定のコマンドまたはデータのシーケンスが従来レジスタに書き込まれるようにすることができる。

【００２９】

次に、Ｅｔｈｅｒｎｅｔ（登録商標）ＮＩＣバイモーダルデバイスについて述べる。レガシーモードでは、Ｅｔｈｅｒｎｅｔ（登録商標）カードに情報を１パケット分送信することは、従来のＩ／Ｏレジスタへの最大８回の個別書き込みを伴うことがある。これらのＩ／Ｏアクセスは、ＤＭＡアドレス（すなわち、パケットデータが見つかるＲＡＭ内の位置）、転送の長さ、および他のパラメータが指定されてから転送を開始するコマンドバイトを含むパケット転送に対しパラメータを設定する必要がある。しかし、Ｉ／Ｏレジスタへのこれらの書き込みはそれぞれ、ＶＭ内では比較的コストが高く、これらのＩ／Ｏアクセスを仮想化するために必要な合計時間は著しく長く、その結果パフォーマンスの損失が大きくなる。しかし、「理想化された」仮想Ｅｔｈｅｒｎｅｔ（登録商標）カードーすなわち、高性能モードで実行中の仮想カードーでは、パケット送信を開始するために１回Ｉ／Ｏアクセスする必要があり、したがって高性能モードで強化されたＥｔｈｅｒｎｅｔ（登録商標）カードバイモーダルデバイス（例えば、ＤＥＣ２１１４０チップセットに基づくデバイス）では、高性能モードを使用することができる場合にはパフォーマンスが著しく高まる。

【００３０】

いくつかの実施形態では、高性能モードは、ゲストオペレーティングシステム環境内でインストールされた特別な修正済みドライバを使用することによりイネーブルにされることができ、高性能モードがイネーブルにされると、ドライバでは、パケット送信を開始するのにより最適なメカニズムを使用することができる。しかし、修正済みドライバがゲスト内に存在しない場合でも、従来ドライバは、高性能モードにより可能なパフォーマンス

最適化が行われていないとしてもネットワーキング機能を提供することができる。

【００３１】

図６Ａは、図５のサブシステム内の仮想デバイスの代わりにバイモーダルデバイスを使用することを例示しており、前記バイモーダルデバイスはレガシーモードで動作する。この図では、従来ドライバＡ’１２２’は、ハードウェア仮想化のためのレガシーモード１５２および理想化された仮想化のための高性能モード１５４の両方を備えるバイモーダルデバイス１５０とインターフェースする（１３６’）。従来ドライバＡ’１２２’は、バイモーダルデバイス１５０の高性能モード１５４により実現される理想化された仮想化と連携するように設計されていないため、その代わり、（図のように）バイモーダルデバイス１５０のレガシーモード１５２により実現されるハードウェア仮想化と連携し、さらにこのバイモーダルデバイスは、ホストＯＳドライバＸ　１３０とインターフェースする（１３８’）。

【００３２】

これと対照的に、図６Ｂは、高性能モードで動作するバイモーダルデバイスを例示している。この図において、高性能ドライバＡ”１２２”はバイモーダルデバイス１５０とインターフェースする（１３６”）。高性能ドライバＡ”１２２”はバイモーダルデバイス１５０の高性能モード１５４により実現される理想化された仮想化と連携することができるため、（図に示されているように）そのように実行し、さらに、バイモーダルデバイス１５０は、ホストＯＳドライバＸ　１３０とインターフェースする（１３８”）。

【００３３】

ハードウェアベースの「バイモーダル」手法は、新しい機能を古い機能と他の方法では非互換であるハードウェアデバイスに追加するために異なる技術分野において使用されている。例えば、従来のＰＳ／２マウスは、３バイトの情報（Ｘ座標、Ｙ座標、およびボタン状態）をコンピュータに送信するが、マウスメーカーが「スクロールホイール」をマウスに追加したときに、スクロールホイール状態を含む４バイトパケットの情報に切り替える必要が生じた。スクロールホイールを認識しないマウスドライバとの後方互換性を維持するため、新しいマウスは「レガシーモード」で起動し、３バイトの情報を送るだけであるが、新しいスタイルのドライバがロードされたときには、このドライバは、マウスに４バイトパケットの送信を開始するよう指示することにより「スクロールホイールモード(scroll-wheel mode)」をイネーブルする。

【００３４】

バイモーダルデバイスの技法(bimodal device techniques)

ゲストモードドライバ(guest-mode driver)がバイモーダルデバイスを識別(identify)できる（すなわち、実(real)ハードウェアデバイスと、追加(additional)オペレーションモードをサポートするためのエミュレートされてたデバイス(emulated device)とを区別する）ための技法がいくつかある。以下の技法があるが、これらに限られるわけではない。

【００３５】

・固有のデバイスバージョン番号：多くのデバイスがソフトウェアからアクセス可能なデバイスバージョン識別子を持つ。これらの値は、ときには、Ｉ／Ｏポートまたはメモリマップトレジスタを通じて直接アクセス可能な場合もある。別のときは、デバイスに特定のコマンドが送信されることで、バージョン情報の返却を要求することができる。これらの場合、既存の実ハードウェアに応答しないようにバージョン番号を修正することが可能なこともある。大半のドライバは、バージョン番号を無視するか、または最小バージョン番号のテストを行うので、既存の従来ドライバとの互換性に影響を及ぼすことなく新しい固有のバージョン番号を選択することが可能でなければならない。

【００３６】

・未使用レジスタ：いくつかのデバイスでは、Ｉ／Ｏポートまたはメモリマップトレジスタの範囲を定義するが、その範囲の部分範囲しか使用していない。これは、範囲は、通常、２の累乗個のブロック単位で割り当てられ、さらにエンジニアは、通常、将来の拡張

に必要とする以上に割り当てるため、よくあることである。その範囲の中で「未定義」または「予約済み」レジスタを識別子として使用することが可能である。実ハードウェアは、通常、予約済みレジスタが読み出されたときに０ｘ００または０ｘＦＦを返す。エミュレートされたバイモーダルデバイスは、異なる区別可能な値を返すことができる。

【００３７】

・ハンドシェーキング：上の２つの技法が実施可能でなければ、通常、何らかの形式の裏口から「ハンドシェーキング」を設計することが可能である。これは、現実世界の利用の仕方にはまず見られないレジスタアクセスまたはデバイスコマンドの特定のシーケンスを伴う。例えば、デバイスがディスクコントローラであった場合、ハンドシェーキング技法は、特定の順序（例えば、０、３、および７）でセクタの特定のあらかじめ定められているリストを開始する複数の０バイトを読み込むコマンドシーケンスを送信することを伴う。実コントローラでは、これは、何の効果ももたらさない（０バイトの読み取りはｎｏ－ｏｐとして定義されているからである）。しかし、バイモーダルデバイスでは、このようなシーケンスは、他の何らかの副作用を引き起こす可能性がある（例えば、ステータスレジスタ内の特定のステータスビットがセットされる）。

【００３８】

当然のことながら、バイモーダルオペレーションが可能なエミュレートされたデバイスを識別するために使用される同じ技法も、前記デバイスの「機能強化(enhanced)」モードを自動的にイネーブルにするために使用することが可能である。

【００３９】

結論：

本明細書で説明されている様々なシステム、方法、および技法は、ハードウェアまたはソフトウェアで、あるいは適切であれば、それらの組み合わせにより実施することができる。したがって、本発明の方法および装置、またはその態様または一部は、磁気ディスク、ＣＤ－ＲＯＭ、ハードドライブ、または他のマシン可読記憶媒体などの有形媒体内に実現されるプログラムコード（すなわち、命令）の形をとることができ、プログラムコードがコンピュータなどのマシンにロードされ実行されるときに、そのマシンは本発明を実施する装置となる。プログラム可能コンピュータ上でプログラムコードを実行する場合、コンピュータは、一般に、プロセッサ、プロセッサにより読み取り可能な記憶媒体（揮発性および不揮発性メモリおよび／または記憶素子）、少なくとも１つの入力デバイス、および少なくとも１つの出力デバイスを備える。コンピュータシステムと通信する１つまたは複数のプログラムは、高水準手続き型またはオブジェクト指向型プログラミング言語で実施するのが好ましい。しかし、プログラムは、必要ならば、アセンブリ言語またはマシン語で実施することができる。いずれの場合も、言語はコンパイル型言語またはインタプリタ型言語であり、ハードウェアの実施と組み合わせられる。

【００４０】

本発明の方法および装置は、電気配線またはケーブル配線、光ファイバ、または他の形態の伝送などの、何らかの伝送媒体を介して伝送されるプログラムコードの形で実現されることも可能であり、プログラムコードがＥＰＲＯＭ、ゲートアレイ、プログラム可能論理回路（ＰＬＤ）、クライアントコンピュータ、ビデオレコーダなどのマシンによりロードされ実行されるときに、マシンは本発明を実施する装置となる。汎用プロセッサで実施する場合、プログラムコードはプロセッサと連携して、本発明のインデックス作成機能を実行する動作をする独自装置を実現する。

【００４１】

本発明については、様々な図の好ましい実施形態に基づいて説明してきたが、本発明から逸脱することなく本発明の同じ機能を実行するために、他の類似の実施形態を使用することができる。また、修正および追加を、これまで説明してきた実施形態に加えることができることが理解されるであろう。例えば、本発明の実施例はパーソナルコンピュータの機能をエミュレートするデジタルデバイスに関して説明されているが、当業者であれば、本発明は、本出願で説明されているように、そのようなデジタルデバイスに限定されず、

有線無線を問わずゲーム機、ハンドヘルドコンピュータ、携帯型コンピュータなど任意の数の既存のまたはこれから現れるコンピューティングデバイスまたは環境に適用することができ、通信ネットワークを介して接続され、ネットワーク上で相互作用するそのような任意の数のコンピューティングデバイスに適用されることができる。さらに、ハンドヘルドデバイスのオペレーティングシステムおよびその他のアプリケーション特有のハードウェア／ソフトウェアインターフェースシステムを含む、様々なコンピュータプラットフォームは、特に無線ネットワーク接続デバイスの数が増え続けるため、本明細書において考察されることが強調されなければならない。したがって、本発明は、単一の実施形態に限られるべきではなく、むしろ付属の請求項による広がりと範囲において解釈すべきである。

【００４２】

最後に、本明細書で説明されている実施形態は、他のプロセッサアーキテクチャ、コンピュータベースのシステム、またはシステム仮想化での使用に適合させることができ、そのような実施形態は、本明細書で行われる開示により明確に予想され、したがって、本発明は本明細書で説明されている規定の実施形態に限られず、その代わりに最も広く解釈されるべきである。同様に、プロセッサ仮想化以外の目的に合成命令を使用することも、本明細書で行われる開示により予想され、プロセッサ仮想化以外の状況における合成命令のそのような利用は、本明細書で行われる開示に最も広く読み込まれるべきである。

【図面の簡単な説明】

【００４３】

【図１】本発明を実施するためのコンピュータシステムを例示したブロック図である。

【図２】コンピュータシステム内でエミュレートされる動作環境のハードウェアおよびソフトウェアアーキテクチャを示す論理的層図である。

【図３Ａ】仮想化コンピューティングシステムの一例を示す図である。

【図３Ｂ】ホストオペレーティングシステムと共に動作する仮想マシンモニタを備えた仮想化コンピューティングシステムの他の実施形態を示す図である。

【図４Ａ】ドライバ、仮想デバイス、およびハードウェアを更に含んだ仮想化コンピューティングシステムを示す図である。

【図４Ｂ】ゲストオペレーティングシステム上で実行中のアプリケーションから図４Ａのコンピュータハードウェア内のハードウェアデバイスへの動作パスを示す図である。

【図５】図４Ｂに示されているドライバ、仮想デバイス、およびハードウェアの間の相互関係を示す図である。

【図６Ａ】図５に示したサブシステム内の仮想デバイスの代わりにバイモーダルデバイスを使用し、このバイモーダルデバイスがレガシーモードで動作していることを示す図である。

【図６Ｂ】高性能モードで動作する図６Ａのバイモーダルデバイスを示す図である。

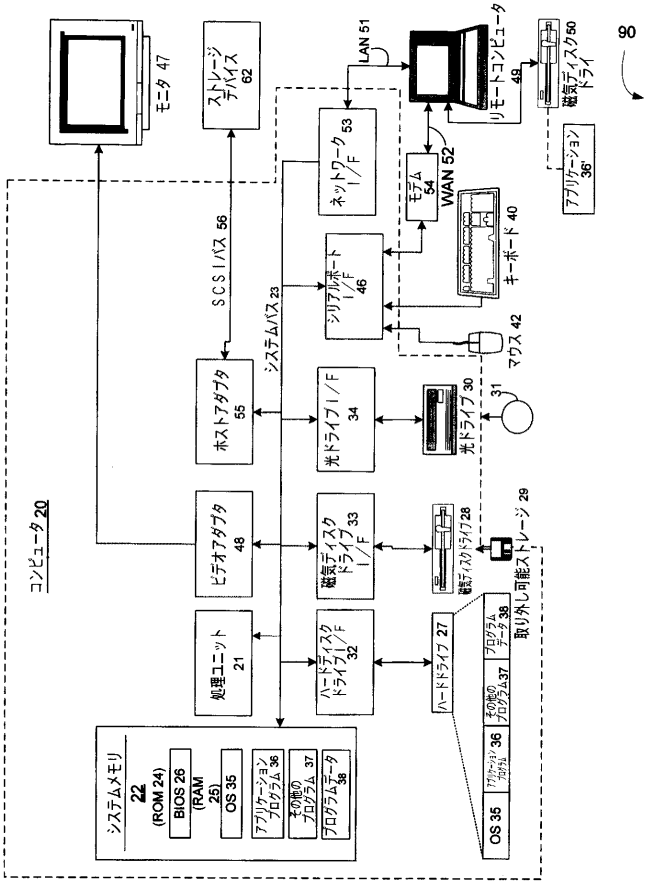
【符号の説明】

【００４４】

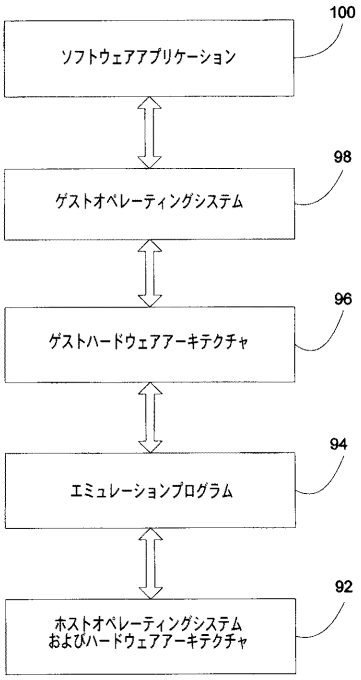
２０　コンピュータ
２１　処理ユニット
２２　システムメモリ
２３　システムバス
２７　ハードドライブ
２８　磁気ディスクドライブ
２９　取り外し可能ストレージ
３０　光ドライブ
３２　ハードディスクドライブＩ／Ｆ
３３　磁気ディスクドライブＩ／Ｆ
３４　光ドライブＩ／Ｆ
３６　アプリケーションプログラム

３７　その他のプログラム
３８　プログラムデータ
４０　キーボード
４２　マウス
４６　シリアルポートＩ／Ｆ
４７　モニタ
４８　ビデオアダプタ
５３　ネットワークＩ／Ｆ
５４　モデム
５５　ホストアダプタ
５６　ＳＣＳＩバス
６２　ストレージデバイス

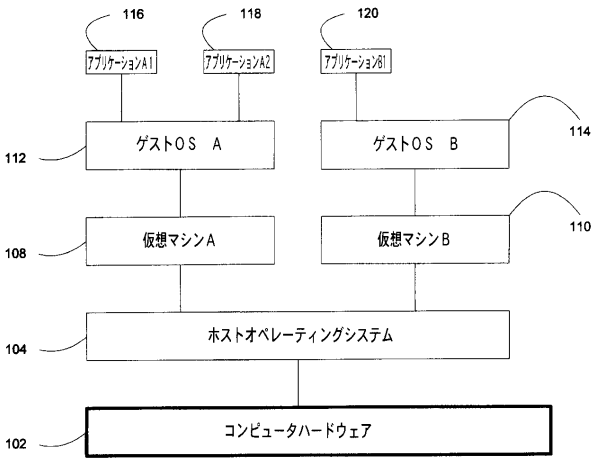
【図１】



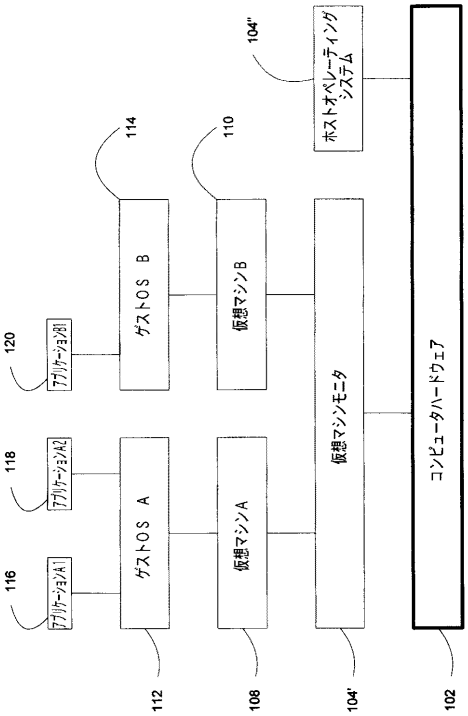
【図２】



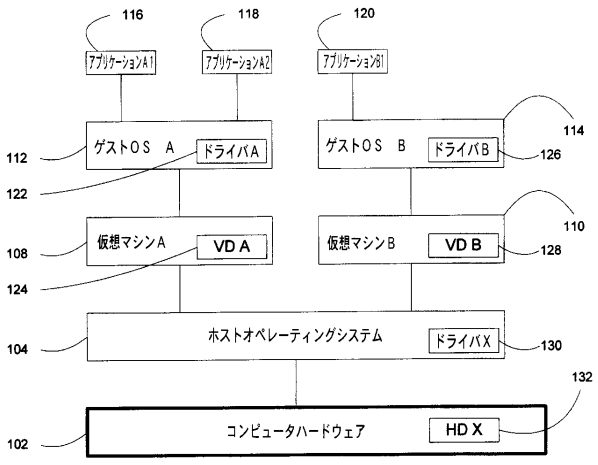
【図３Ａ】



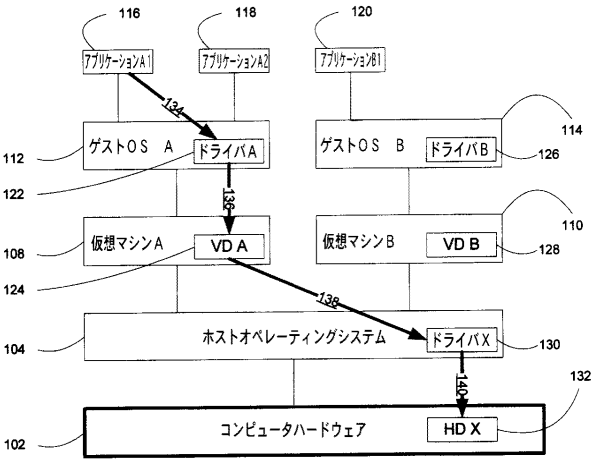
【図３Ｂ】



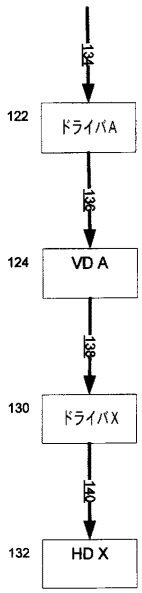
【図４Ａ】



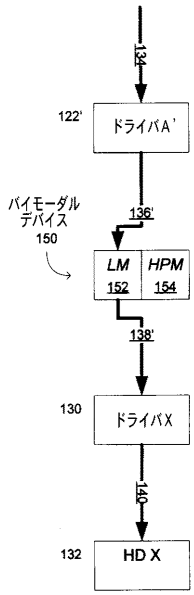
【図４Ｂ】



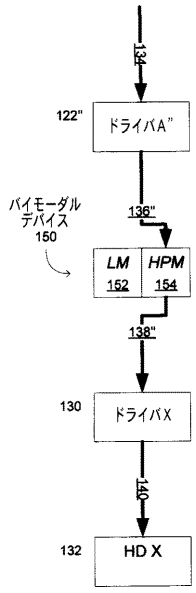
【図５】


134
122
ドライバA
136
124
VD A
138
130
ドライバX
140
132
HD X

【図６Ａ】


134
122'
ドライバA'
バイモーダル
デバイス
150
136'
LM
152
HPM
154
138'
130
ドライバX
140
132
HD X

【図６Ｂ】


134
122"
ドライバA"
バイモーダル
デバイス
150
136"
LM
152
HPM
154
138"
130
ドライバX
140
132
HD X

フロントページの続き

審査官　坂東　博司

(56)参考文献　特開平０７－２４４６２８（ＪＰ，Ａ）
特開２００１－２５６１５３（ＪＰ，Ａ）
国際公開第０３／０２７８３５（WO，Ａ１）
米国特許第０５５４８７８３（ＵＳ，Ａ）
特開平２－１２３４０（ＪＰ，Ａ）
特開平５－１５１０８４（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　１３／１０