(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第5608243号
(P5608243)
(45)発行日　平成26年10月15日(2014.10.15)　(24)登録日　平成26年9月5日(2014.9.5)

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

請求項の数 22 (全 16 頁)

(21)出願番号　特願2012-545042(P2012-545042)
(86)(22)出願日　平成21年12月24日(2009.12.24)
(65)公表番号　特表2013-515983(P2013-515983A)
(43)公表日　平成25年5月9日(2013.5.9)
(86)国際出願番号　PCT/CN2009/001543
(87)国際公開番号　WO2011/075870
(87)国際公開日　平成23年6月30日(2011.6.30)
審査請求日　平成24年6月25日(2012.6.25)

(73)特許権者　591003943
インテル・コーポレーション
アメリカ合衆国　９５０５４　カリフォルニア州・サンタクララ・ミッション　カレッジ　ブーレバード・２２００
(74)代理人　110000877
龍華国際特許業務法人
(72)発明者　ドン、ヤオズ
アメリカ合衆国　９５０５２　カリフォルニア州・サンタクララ・ミッション　カレッジ　ブーレバード・２２００　インテル・コーポレーション内

審査官　木村　貴俊

最終頁に続く

(54)【発明の名称】仮想化環境においてＩ／Ｏ処理を行う方法および装置

(57)【特許請求の範囲】
【請求項１】
サービス仮想マシンが実行する方法であって、
前記サービス仮想マシンのデバイスモデルによって、前記サービス仮想マシンのデバイスドライバを呼び出して、入出力（Ｉ／Ｏ）デバイスの仮想機能インタフェースを制御して、Ｉ／Ｏ処理を、前記Ｉ／Ｏ処理に関連し、１以上のＩ／Ｏ記述子を有するＩ／Ｏ情報であって、ゲスト仮想マシンによって書き込まれたＩ／Ｏ情報を利用して実行する段階を備え、
前記１以上のＩ／Ｏ記述子は、Ｉ／Ｏ処理の種類と前記仮想機能インタフェースによって読出しまたは書込みを行うゲストメモリアドレスとを示すデータを含み、
前記デバイスモデルは、前記サービス仮想マシンのホストＯＳ上で動作し、仮想Ｉ／Ｏデバイスまたは物理Ｉ／Ｏデバイスをエミュレートし、
前記デバイスモデル、前記デバイスドライバ、および、前記Ｉ／Ｏデバイスの前記仮想機能インタフェースは、前記ゲスト仮想マシンに割り当てられ、
前記Ｉ／Ｏデバイスの前記仮想機能インタフェースが前記ゲスト仮想マシンのアーキテクチャに準拠して動作できない場合、
前記デバイスドライバによって、前記ゲスト仮想マシンのアーキテクチャに準拠している前記Ｉ／Ｏ情報を、前記Ｉ／Ｏデバイスの前記仮想機能インタフェースのアーキテクチャに準拠しているシャドーＩ／Ｏ情報に変換する段階と、
前記デバイスドライバによって、前記Ｉ／Ｏデバイスの前記仮想機能インタフェースの

アーキテクチャに準拠している更新済みのシャドーＩ／Ｏ情報を、前記ゲスト仮想マシンのアーキテクチャに準拠している更新済みのＩ／Ｏ情報に変換する段階と
をさらに備え、
前記更新済みのシャドーＩ／Ｏ情報は、前記Ｉ／Ｏ処理の実行に応じて、前記Ｉ／Ｏデバイスの前記仮想機能インタフェースによって更新される方法。
【請求項２】
前記Ｉ／Ｏ処理が実行された後に、前記デバイスドライバによって、前記Ｉ／Ｏデバイスの前記仮想機能インタフェースのステータスを維持する段階をさらに備える請求項１に記載の方法。
【請求項３】
前記Ｉ／Ｏ処理が実行された旨を、前記デバイスモデルが前記ゲスト仮想マシンに通知する段階をさらに備える請求項１または請求項２に記載の方法。
【請求項４】
前記Ｉ／Ｏ情報は、前記Ｉ／Ｏデバイスの前記仮想機能インタフェースによって制御可能なヘッドポインタを先頭とするバッファに書き込まれる請求項１から請求項３のいずれか１つに記載の方法。
【請求項５】
Ｉ／Ｏ情報の終端を示すテイルポインタを、前記ゲスト仮想マシンで更新し、
制御は、前記テイルポインタが更新されることに対応して、前記ゲスト仮想マシンから前記サービス仮想マシンに移行し、
制御が前記ゲスト仮想マシンから前記サービス仮想マシンに移行されることに対応して、前記サービス仮想マシンの前記デバイスモデルによって、前記サービス仮想マシンの前記デバイスドライバを呼び出す、請求項１から請求項４のいずれか１つに記載の方法。
【請求項６】
デバイスモデルと、
デバイスドライバと
を備え、
前記デバイスモデルは、前記デバイスドライバを呼び出して、入出力（Ｉ／Ｏ）デバイスの仮想機能インタフェースを制御させて、Ｉ／Ｏ処理を、前記Ｉ／Ｏ処理に関連し、１以上のＩ／Ｏ記述子を含むＩ／Ｏ情報であってゲスト仮想マシンによって書き込まれるＩ／Ｏ情報を利用して実行させ、
前記１以上のＩ／Ｏ記述子は、Ｉ／Ｏ処理の種類と前記仮想機能インタフェースによって読出しまたは書込みを行うゲストメモリアドレスとを示すデータを含み、
前記デバイスモデルは、サービス仮想マシンのホストＯＳ上で動作し、仮想Ｉ／Ｏデバイスまたは物理Ｉ／Ｏデバイスをエミュレートし、
前記デバイスモデル、前記デバイスドライバ、および、前記Ｉ／Ｏデバイスの前記仮想機能インタフェースは、前記ゲスト仮想マシンに割り当てられ、
前記Ｉ／Ｏデバイスの前記仮想機能インタフェースが前記ゲスト仮想マシンのアーキテクチャに準拠して動作できない場合、
前記デバイスドライバは、
前記ゲスト仮想マシンのアーキテクチャに準拠している前記Ｉ／Ｏ情報を、前記Ｉ／Ｏデバイスの前記仮想機能インタフェースのアーキテクチャに準拠しているシャドーＩ／Ｏ情報に変換し、
前記デバイスドライバによって、前記Ｉ／Ｏデバイスの前記仮想機能インタフェースのアーキテクチャに準拠している更新済みのシャドーＩ／Ｏ情報を、前記ゲスト仮想マシンのアーキテクチャに準拠している更新済みのＩ／Ｏ情報に変換し、
前記更新済みのシャドーＩ／Ｏ情報は、前記Ｉ／Ｏ処理の実行に応じて、前記Ｉ／Ｏデバイスの前記仮想機能インタフェースによって更新される装置。
【請求項７】
前記デバイスドライバはさらに、前記Ｉ／Ｏ処理が実行された後に、前記Ｉ／Ｏデバイ

スの前記仮想機能インタフェースのステータスを維持する請求項６に記載の装置。
【請求項８】
前記デバイスモデルはさらに、前記Ｉ／Ｏ処理が実施された旨を、前記ゲスト仮想マシンに通知する請求項６または請求項７に記載の装置。
【請求項９】
前記Ｉ／Ｏ情報は、前記Ｉ／Ｏデバイスの前記仮想機能インタフェースによって制御可能なヘッドポインタを先頭とするバッファに書き込まれる請求項６から請求項８のいずれか１つに記載の装置。
【請求項１０】
Ｉ／Ｏ情報の終端を示すテイルポインタを、前記ゲスト仮想マシンで更新し、
制御は、前記テイルポインタが更新されることに対応して、前記ゲスト仮想マシンから前記サービス仮想マシンに移行し、
制御が前記ゲスト仮想マシンから前記サービス仮想マシンに移行されることに対応して、前記サービス仮想マシンの前記デバイスモデルによって、前記サービス仮想マシンの前記デバイスドライバを呼び出す、請求項６から請求項９のいずれか１つに記載の装置。
【請求項１１】
システムに、
サービス仮想マシンのデバイスモデルによって、前記サービス仮想マシンのデバイスドライバを呼び出して、入出力（Ｉ／Ｏ）デバイスの仮想機能インタフェースを制御して、Ｉ／Ｏ処理を、前記Ｉ／Ｏ処理に関連し、１以上のＩ／Ｏ記述子を含むＩ／Ｏ情報であって、ゲスト仮想マシンによって書き込まれたＩ／Ｏ情報を利用して実行する段階
を実行させるためのプログラムであり、
前記１以上のＩ／Ｏ記述子は、Ｉ／Ｏ処理の種類と前記仮想機能インタフェースによって読出しまたは書込みを行うゲストメモリアドレスとを示すデータを含み、
前記デバイスモデルは、前記サービス仮想マシンのホストＯＳ上で動作し、仮想Ｉ／Ｏデバイスまたは物理Ｉ／Ｏデバイスをエミュレートし、
前記デバイスモデル、前記デバイスドライバ、および、前記Ｉ／Ｏデバイスの前記仮想機能インタフェースは、前記ゲスト仮想マシンに割り当てられ、
前記Ｉ／Ｏデバイスの前記仮想機能インタフェースが前記ゲスト仮想マシンのアーキテクチャに準拠して動作できない場合、
前記デバイスドライバによって、前記ゲスト仮想マシンのアーキテクチャに準拠している前記Ｉ／Ｏ情報を、前記Ｉ／Ｏデバイスの前記仮想機能インタフェースのアーキテクチャに準拠しているシャドーＩ／Ｏ情報に変換する段階と、
前記デバイスドライバによって、前記Ｉ／Ｏデバイスの前記仮想機能インタフェースのアーキテクチャに準拠している更新済みのシャドーＩ／Ｏ情報を、前記ゲスト仮想マシンのアーキテクチャに準拠している更新済みのＩ／Ｏ情報に変換する段階と
をさらに前記システムに実行させ、
前記更新済みのシャドーＩ／Ｏ情報は、前記Ｉ／Ｏ処理の実行に応じて、前記Ｉ／Ｏデバイスの前記仮想機能インタフェースによって更新される
プログラム。
【請求項１２】
前記システムに、
前記Ｉ／Ｏ処理が実行された後に、前記デバイスドライバによって、前記Ｉ／Ｏデバイスの前記仮想機能インタフェースのステータスを維持する段階をさらに実行させる請求項１１に記載のプログラム。
【請求項１３】
前記システムに、前記Ｉ／Ｏ処理が実施された旨を、前記デバイスモデルから前記ゲスト仮想マシンに通知する段階をさらに実行させる請求項１１または請求項１２に記載のプログラム。
【請求項１４】

前記Ｉ／Ｏ情報は、前記Ｉ／Ｏデバイスの前記仮想機能インタフェースによって制御可能なヘッドポインタを先頭とするバッファに書き込まれる請求項１１から請求項１３のいずれか１つに記載のプログラム。

【請求項１５】

Ｉ／Ｏ情報の終端を示すテイルポインタを、前記ゲスト仮想マシンで更新し、

制御は、前記テイルポインタが更新されることに対応して、前記ゲスト仮想マシンから前記サービス仮想マシンに移行し、

制御が前記ゲスト仮想マシンから前記サービス仮想マシンに移行されることに対応して、前記サービス仮想マシンの前記デバイスモデルによって、前記サービス仮想マシンの前記デバイスドライバを呼び出す、請求項１１から請求項１４のいずれか１つに記載のプログラム。

【請求項１６】

入出力デバイス（Ｉ／Ｏデバイス）を有するハードウェアマシンと、

前記ハードウェアマシンと複数の仮想マシンとの間でやり取りを行うための仮想マシンモニタと

を備え、

前記複数の仮想マシンは、

入出力処理（Ｉ／Ｏ処理）に関連し、１以上のＩ／Ｏ記述子を含むＩ／Ｏ情報を書き込むゲスト仮想マシンと、

デバイスモデルおよびデバイスドライバを含むサービス仮想マシンと、

を有し、

前記デバイスモデルは、前記デバイスドライバを呼び出して、前記Ｉ／Ｏデバイスの仮想機能インタフェースを制御させて、前記Ｉ／Ｏ処理を前記Ｉ／Ｏ情報を利用して実行させ、

前記１以上のＩ／Ｏ記述子は、Ｉ／Ｏ処理の種類と前記仮想機能インタフェースによって読出しまたは書込みを行うゲストメモリアドレスとを示すデータを含み、

前記デバイスモデルは、前記サービス仮想マシンのホストＯＳ上で動作し、仮想Ｉ／Ｏデバイスまたは物理Ｉ／Ｏデバイスをエミュレートし、

前記デバイスモデル、前記デバイスドライバ、および、前記Ｉ／Ｏデバイスの前記仮想機能インタフェースは、前記ゲスト仮想マシンに割り当てられ、

前記Ｉ／Ｏデバイスの前記仮想機能インタフェースが前記ゲスト仮想マシンのアーキテクチャに準拠して動作できない場合、

前記サービス仮想マシンの前記デバイスドライバはさらに、

前記ゲスト仮想マシンのアーキテクチャに準拠している前記Ｉ／Ｏ情報を、前記Ｉ／Ｏデバイスの前記仮想機能インタフェースのアーキテクチャに準拠しているシャドーＩ／Ｏ情報に変換し、

前記Ｉ／Ｏデバイスの少なくとも前記仮想機能インタフェースのアーキテクチャに準拠している更新済みのシャドーＩ／Ｏ情報を、前記ゲスト仮想マシンのアーキテクチャに準拠している更新済みのＩ／Ｏ情報に変換し、

前記更新済みのシャドーＩ／Ｏ情報は、前記Ｉ／Ｏ処理の実行に応じて、前記Ｉ／Ｏデバイスの前記仮想機能インタフェースによって更新されるシステム。

【請求項１７】

前記ゲスト仮想マシンは、前記Ｉ／Ｏ情報を、前記Ｉ／Ｏデバイスの前記仮想機能インタフェースによって更新されるヘッドポインタを先頭とするバッファに書き込む請求項１６に記載のシステム。

【請求項１８】

前記ゲスト仮想マシンは、前記Ｉ／Ｏ情報の終端を示すテイルポインタを更新する請求項１６または請求項１７に記載のシステム。

【請求項１９】

前記仮想マシンモニタは、前記テイルポインタが更新されたことを検出すると、前記シ

ステムの制御を、前記ゲスト仮想マシンから前記サービス仮想マシンへと移行し、
前記デバイスモデルは、制御が前記ゲスト仮想マシンから前記サービス仮想マシンに移行されることに対応して、前記サービス仮想マシンの前記デバイスドライバを呼び出す、請求項１８に記載のシステム。
【請求項２０】
前記Ｉ／Ｏデバイスの前記仮想機能インタフェースは、前記Ｉ／Ｏ処理が実行されたことに応じて前記Ｉ／Ｏ情報を更新する請求項１６から請求項１９のいずれか１つに記載のシステム。
【請求項２１】
前記デバイスドライバは、前記Ｉ／Ｏ処理が実行された後、前記Ｉ／Ｏデバイスの前記仮想機能インタフェースのステータスを維持する請求項１６から請求項２０のいずれか１つに記載のシステム。
【請求項２２】
前記デバイスモデルは、前記Ｉ／Ｏ処理が実行された旨を前記ゲスト仮想マシンに通知する請求項１６から請求項２１のいずれか１つに記載のシステム。
【発明の詳細な説明】
【背景技術】
【０００１】
仮想マシンアーキテクチャは、物理マシンを論理的に分割して、マシンの下層ハードウェアを共有し、１以上の独立して動作する仮想マシンに見えるようにするアーキテクチャである。入出力（Ｉ／Ｏ）仮想化（ＩＯＶ）によって、複数の仮想マシンが利用する一のＩ／Ｏデバイスの機能を実現するとしてよい。
【０００２】
ソフトウェア完全デバイスエミュレーションは、Ｉ／Ｏ仮想化の一例であるとしてよい。Ｉ／Ｏデバイスの完全エミュレーションによって、仮想マシンは既存のデバイスドライバを再利用することができるようになるとしてよい。シングル・ルート・Ｉ／Ｏ仮想化（ＳＲ－ＩＯＶ）または任意のその他のリソースパーティション化方法は、Ｉ／Ｏ仮想化の別の一例であるとしてよい。Ｉ／Ｏデバイス機能（例えば、データ移動に関するＩ／Ｏデバイス機能）を複数の仮想インターフェース（ＶＩ）にパーティション化して、各仮想インターフェースを１つの仮想マシンを割り当てる場合、ソフトウェアエミュレーションレイヤのＩ／Ｏオーバーヘッドが少なくなる場合がある。
【図面の簡単な説明】
【０００３】
添付図面では、本明細書で説明する発明を図示しているが、一例に過ぎず限定するものではない。図示を簡略化して分かりやすくするべく、図面に図示する構成要素は必ずしも実寸に即したものではない。例えば、一部の構成要素の寸法は、分かりやすいように、他の構成要素と比較して強調しているとしてもよい。また、適切であると考えられる場合には、対応する構成要素または同様の構成要素を示す際に同じ参照符号を複数の図面にわたって繰り返し用いる。
【図１】ゲスト仮想マシンを発行元とするＩ／Ｏ処理を制御するサービス仮想マシンを含むコンピューティングプラットフォームの実施形態を示す図である。
【図２Ａ】Ｉ／Ｏ処理のためのＩ／Ｏ記述子を格納する記述子リング構造の実施形態を示す図である。
【図２Ｂ】Ｉ／Ｏ処理のためのＩ／Ｏ記述子を格納する記述子リング構造およびシャドー記述子リング構造の実施形態を示す図である。
【図３】Ｉ／Ｏデバイスによるダイレクトメモリアクセス（ＤＭＡ）のための入出力メモリ管理ユニット（ＩＯＭＭＵ）テーブルの実施形態を示す図である。
【図４】ゲスト仮想マシンによるＩ／Ｏ処理に関連するＩ／Ｏ情報を書き込む方法の実施形態を示す図である。
【図５】サービス仮想マシンによってＩ／Ｏ情報に基づきＩ／Ｏ処理を実行する方法の実

施形態を示す図である。
【図６Ａ】サービス仮想マシンによってＩ／Ｏ情報に基づきＩ／Ｏ処理を実行する方法の別の実施形態を示す図である。
【図６Ｂ】サービス仮想マシンによってＩ／Ｏ情報に基づきＩ／Ｏ処理を実行する方法の別の実施形態を示す図である。
【発明を実施するための形態】
【０００４】
以下に記載する説明では、仮想化環境においてＩ／Ｏ処理を実行する方法を説明する。以下に記載する説明では、論理実施例、疑似コード、オペランド指定手段、リソースのパーティション化／共有／複製の実施例、システム構成要素の種類および相関関係、および、論理のパーティション化／統合化に関する選択肢等、具体的且つ詳細な内容を数多く記載する。これによって、本発明を深く理解していただきたい。しかし、以下に記載する具体的且つ詳細な内容を採用することなく本発明を実施し得る。また、制御構造、ゲートレベル回路および完全なソフトウェア命令列は、本発明をあいまいにすることを避けるべく、詳細な説明を省略している。当業者であれば、本明細書に記載した内容に基づき、過度な実験を必要とすることなく適切な機能を実現可能であろう。
【０００５】
本明細書において「一実施形態」、「ある実施形態」、「実施形態例」という場合、当該実施形態は特定の特徴、構造または特性を含むが、どの実施形態でもその特定の特徴、構造または特性を必ずしも含むものではない。さらに、上記の表現は、必ずしも同じ実施形態を意味しているものではない。さらに、特定の特徴、構造または特性がある実施形態に関連付けて説明されている場合、明示的な言及の有無に関わらず、この特徴、構造または特性を他の実施形態に関連付けて実施することは当業者の想到する範囲内であると考えられる。
【０００６】
本発明の実施形態は、ハードウェア、ファームウェア、ソフトウェアまたはこれらの任意の組み合わせで実施され得る。本発明の実施形態はまた、１以上のプロセッサで読み出されて実行される機械可読媒体に格納されている命令として実現されるとしてもよい。機械可読媒体は、機械（例えば、コンピューティングデバイス）が読み出し可能な形式で情報を格納または送信する任意のメカニズムを含むとしてよい。例えば、機械可読媒体は、リードオンリーメモリ（ＲＯＭ）、ランダムアクセスメモリ（ＲＡＭ）、磁気ディスク格納媒体、光格納媒体、フラッシュメモリデバイス、電気的、光学的、音響的またはその他の方法で伝搬する信号（例えば、搬送波、赤外信号、デジタル信号等）等を含むとしてよい。
【０００７】
仮想化環境においてＩ／Ｏ処理を行うコンピューティングプラットフォーム１００の実施形態を図１に示す。コンピューティングシステム１００の例を列挙すると、これらで全てを網羅しているわけではないが、分散型コンピューティングシステム、スーパーコンピュータ、コンピューティングクラスタ、メインフレームコンピュータ、ミニコンピュータ、パーソナルコンピュータ、ワークステーション、サーバ、ポータブルコンピュータ、ラップトップコンピュータ、および、データを送受信して処理するその他のデバイスがある。
【０００８】
当該実施形態によると、コンピューティングプラットフォーム１００は、１以上のプロセッサ１１１、メモリシステム１２１、チップセット１３１、Ｉ／Ｏデバイス１４１およびその他の構成要素を有する下層ハードウェアマシン１０１を備えるとしてよい。１以上のプロセッサ１１１は、プロセッサバス（図１には不図示）等の１以上のバスを介して、さまざまな構成要素（例えば、チップセット１３１）に通信可能に結合されているとしてよい。プロセッサ１１１は、適切なアーキテクチャでコードを実行する１以上のプロセッシングコアを含む集積回路（ＩＣ）として実現されるとしてよい。

【０００９】
メモリシステム１２１は、プロセッサ１１１が実行すべき命令およびデータを格納するとしてよい。メモリ１２１の例としては、シンクロナスダイナミックランダムアクセスメモリ（ＳＤＲＡＭ）デバイス、ＲＡＭＢＵＳダイナミックランダムアクセスメモリ（ＲＤＲＡＭ）デバイス、ダブルデータレート（ＤＤＲ）メモリデバイス、スタティックランダムアクセスメモリ（ＳＲＡＭ）およびフラッシュメモリデバイス等の半導体デバイスのうち、１つまたは任意の組み合わせを含むとしてよい。
【００１０】
チップセット１３１は、１以上のプロセッサ１１１、メモリ１２１およびその他の構成要素、例えば、Ｉ／Ｏデバイス１４１の間をつなぐ１以上の通信経路を提供するとしてよい。Ｉ／Ｏデバイス１４１は、これらに限定されないが、ペリフェラル・コンポーネント・インターコネクト（ＰＣＩ）バスまたはＰＣＩ Ｅｘｐｒｅｓｓ（ＰＣＩｅ）バスを介してホストマザーボードと接続されているＰＣＩデバイスおよび／またはＰＣＩｅデバイスを含むとしてよい。Ｉ／Ｏデバイス１４１の例は、ユニバーサルシリアルバス（ＵＳＢ）コントローラ、グラフィクスアダプタ、オーディオコントローラ、ネットワークインターフェースコントローラ（ＮＩＣ）、ストレージデバイス等を含むとしてよい。
【００１１】
コンピューティングプラットフォーム１００はさらに、仮想マシンモニタ（ＶＭＭ）１０２を備えるとしてよい。ＶＭＭ１０２は、下層のハードウェアおよび上層の下層マシン（例えば、サービス仮想マシン１０３、ゲスト仮想マシン１０３_{1}－１０３_{n}）のインターフェースとして機能して、仮想マシン（例えば、サービス仮想マシン１０３のホストオペレーティングシステム１１３、ゲスト仮想マシン１０３_{1}－１０３_{n}のゲストオペレーティングシステム１１３_{1}－１１３_{n}）の複数のオペレーティングシステム（ＯＳ）の動作を円滑化および管理して、下層物理リソースを共有する。仮想マシンモニタの例としては、Ｘｅｎ、ＥＳＸサーバ、仮想ＰＣ、Ｖｉｒｔｕａｌ Ｓｅｒｖｅｒ、Ｈｙｐｅｒ－Ｖ、Ｐａｒａｌｌｅｌ、ＯｐｅｎＶＺ、Ｑｅｍｕ等が挙げられるとしてよい。
【００１２】
ある実施形態によると、Ｉ／Ｏデバイス１４１（例えば、ネットワークカード）は、複数の機能部にパーティション化されるとしてよい。これらの機能部は、入出力仮想化（ＩＯＶ）アーキテクチャ（例えば、シングル・ルートＩＯＶ）をサポートするコントロールエンティティ（ＣＥ）１４１_{0}と、専用アクセス用のランタイムリソース（例えば、ネットワークデバイスにおけるキュー対）を持つ複数の仮想機能インターフェース（ＶＩ）１４１_{1}－１４１_{n}とを含む。ＣＥおよびＶＩの例は、シングル・ルートＩ／Ｏ仮想化アーキテクチャまたはマルチ・ルートＩ／Ｏ仮想化アーキテクチャにおける物理機能および仮想機能を含むとしてよい。ＣＥはさらに、ＶＩ機能を設定および管理するとしてよい。ある実施形態によると、複数のゲスト仮想マシン１０３_{1}－１０３_{n}が、$ＣＥ１４１_{0}$が制御する複数の物理リソースを共有する一方、ゲスト仮想マシン１０３_{1}－１０３_{n}のそれぞれは、$ＶＩ１４１_{1}$－１４１_{n}のうち１以上に割り当てられるとしてよい。例えば、ゲスト仮想マシン１０３_{1}は$ＶＩ１４１_{1}$に割り当てられるとしてよい。
【００１３】
他の実施形態はＩ／Ｏデバイス１４１の構造について他の技術を採用し得るものと考えられたい。ある実施形態によると、Ｉ／Ｏデバイス１４１は、１以上のＶＩを含むがＣＥを含まないとしてもよい。例えば、パーティション化機能を持たないレガシーＮＩＣは、ＮＵＬＬ ＣＥ条件において機能する１つのＶＩを含むとしてよい。
【００１４】
サービス仮想マシン１０３には、デバイスモデル１１４、ＣＥドライバ１１５およびＶＩドライバ１１６のコードがロードされているとしてよい。デバイスモデル１１４は、実際のＩ／Ｏデバイス１４１のソフトウェアエミュレーションであってもなくてもよい。ＣＥドライバ１１５は、コンピューティングプラットフォーム１００の初期化およびランタイムにおいて、Ｉ／Ｏデバイスの初期化および設定に関連する$ＣＥ１４１_{0}$を管理すると

してよい。ＶＩドライバ１１６は、管理ポリシーに応じて、ＶＩ１４１$_1$－ＶＩ１４１$_n$のうち１以上を管理するデバイスドライバであってよい。ある実施形態によると、管理ポリシーに基づいて、ＶＩドライバは、ＶＩドライバがサポートしているゲストＶＭに割り当てられているリソースを管理するとしてよく、ＣＥドライバは、グローバル動作を管理するとしてよい。

【００１５】

ゲスト仮想マシン１０３$_1$－１０３$_n$はそれぞれ、ＶＭＭ１０２が提示する仮想デバイスを管理するゲストデバイスドライバ、例えば、ゲスト仮想マシン１０３$_1$のゲストデバイスドライバ１１６$_1$またはゲスト仮想マシン１０３$_n$のゲストデバイスドライバ１１６$_n$のコードがロードされているとしてよい。ゲストデバイスドライバは、ＶＩ１４１およびそれらのドライバ１１６に適応しているモードで動作することが可能または不可能であるとしてよい。ある実施形態によると、ゲストデバイスドライバは、レガシードライバであってよい。

【００１６】

ある実施形態によると、ゲスト仮想マシンのゲストオペレーティングシステム（例えば、ゲストＶＭ１０３$_1$のゲストＯＳ１１３$_1$）がゲストデバイスドライバ（例えば、ゲストデバイスドライバ１１６$_1$）をロードすることに応じて、サービスＶＭ１０３がＶＩドライバ１１６およびデバイスモデル１１４のインスタンスを実行するとしてよい。例えば、デバイスモデル１１４のインスタンスは、ゲストデバイスドライバ１１６$_1$にサービスを提供し、一方で、ＶＩドライバ１１６のインスタンスは、ゲストＶＭ１０３$_1$に割り当てられたＶＩ１４１$_1$を制御するとしてよい。例えば、ゲストデバイスドライバ１１６$_1$が、８２５７１ＥＢベースのＮＩＣ（Ｉｎｔｅｌ　Ｃｏｒｐｏｒａｔｉｏｎ（米国カリフォルニア州サンタクラーラ）社製のネットワークコントローラ）のレガシードライバであり、ゲストＶＭ１０３$_1$に割り当てられているＶＩ１４１$_1$が、８２５７１ＥＢベースのＮＩＣであるか、または、８２５７１ＥＢベースのＮＩＣに準拠した、または、していないその他の種類のＮＩＣである場合、サービスＶＭ１０３は、仮想８２５７１ＥＢベースのＮＩＣを表すデバイスモデル１１４のインスタンス、および、ＶＩ１４１$_1$を制御するＶＩドライバ１１６のインスタンス、つまり、８２５７１ＥＢベースのＮＩＣ、または、８２５７１ＥＢベースのＮＩＣに準拠した、または、していないその他の種類のＮＩＣを実行するとしてよい。

【００１７】

図１に図示されている実施形態は説明のためのものであり、他の技術を採用すればコンピューティングシステム１００の他の実施形態が実施され得るものと考えられたい。例えば、デバイスモデル１１４は、ＶＩドライバ１１６またはＣＥドライバに組み込まれているとしてよく、または、全て１つにまとめられているとしてもよい。ＯＳカーネル等の特権モードで動作するとしてもよいし、ＯＳユーザランド等の非特権モードで動作するとしてもよい。サービスＶＭは、さらに複数のＶＭに分割され、１つのＶＭがＣＥを実行する一方で、別のＶＭが、複数のＶＭ間での通信を十分に実行しつつ、デバイスモデルおよびＶＩドライバまたは任意のその他の組み合わせを実行するとしてよい。

【００１８】

ある実施形態によると、ゲストＶＭ１０３$_1$で実行されているアプリケーション（例えば、アプリケーション１１７$_1$）によってＩ／Ｏ処理が指示された場合、ゲストデバイスドライバ１１６$_1$は、当該Ｉ／Ｏ処理に関連するＩ／Ｏ情報を、ゲストＶＭ１０３$_1$に割り当てられているバッファ（図１には不図示）に書き込むとしてよい。例えば、ゲストデバイスドライバ１１６$_1$は、Ｉ／Ｏ記述子を図２Ａに示すリング構造に書き込むとしてよい。この場合、リンク構造の１つのエントリを、１つのＩ／Ｏ記述子に使う。ある実施形態によると、Ｉ／Ｏ記述子は、データパケットに関連するＩ／Ｏ処理を示すとしてよい。例えば、ゲストアプリケーション１１７$_1$がゲストメモリアドレスｘｘｘ－ｙｙｙとの間で１００個のパケットの読出または書込を指示すると、ゲストデバイスドライバ１１６$_1$は、１００個のＩ／Ｏ記述子を図２Ａの記述子リングに書き込むとしてよい。ゲストデバ

イスドライバ１１６$_1$は、ヘッドポインタ２０１を先頭として、記述子を記述子リングに書き込むとしてよい。ゲストデバイスドライバ１１６$_1$は、Ｉ／Ｏ処理に関連する記述子の書き込みを完了した後、テイルポインタ２０２を更新するとしてよい。ある実施形態によると、ヘッドポインタ２０１およびテイルポインタ２０２は、ヘッドレジスタおよびテイルレジスタ（不図示）に格納しているとしてよい。

【００１９】

ある実施形態によると、記述子は、データ、Ｉ／Ｏ処理の種類（読出か書込か）、ＶＩ１４１$_1$がデータの読出または書込を行うゲストメモリアドレス、Ｉ／Ｏ処理のステータス、Ｉ／Ｏ処理に必要な他の情報を含むとしてよい。

【００２０】

ある実施形態によると、ゲストデバイスドライバ１１６$_1$がゲストＶＭ１０３$_1$に割り当てられているＶＩ１４１$_1$に適応したモードで動作できない場合、例えば、ＶＩ１４１$_1$およびゲストデバイスドライバ１１６$_1$がサポートしているビット形式および／または意味論が異なるために、ゲストデバイスドライバ１１６$_1$が書き込んだ記述子に基づいてＶＩ１４１$_1$がＩ／Ｏ処理を実施できない場合、ＶＩドライバ１１６は、シャドーリング（図２Ｂに示す）を生成して、ゲストＶＭ１０３$_1$のアーキテクチャに準拠している記述子、ヘッドポインタおよびテイルポインタを、ＶＩ１４１$_1$のアーキテクチャに準拠しているシャドー記述子（Ｓ記述子）、シャドーヘッドポインタ（Ｓヘッドポインタ）およびシャドーテイルポインタ（Ｓテイルポインタ）に変換するとしてよい。この結果、ＶＩ１４１$_1$がシャドー記述子に基づいてＩ／Ｏ処理を実施できるようになる。

【００２１】

図２Ａおよび図２Ｂに図示されている実施形態は説明のためのものであり、他の技術を採用すればＩ／Ｏ情報の他の実施形態を実施し得ると考えられたい。例えば、Ｉ／Ｏ情報は、図２Ａおよび図２Ｂのリング構造以外のデータ構造、例えば、ハッシュテーブル、リンクテーブル等で書き込まれるとしてもよい。別の例を挙げると、１つのリングを受信および送信の両方に用いるとしてよく、複数の異なるリングを受信または送信に用いるとしてよい。

【００２２】

ＩＯＭＭＵまたは同様の技術によって、Ｉ／Ｏデバイス１４１は、記述子リングまたはシャドー記述子リングの記述子から取得されたゲストアドレスをホストアドレスに再マッピングすることによって、メモリシステム１２１に直接アクセスすることができるとしてよい。図３は、ＩＯＭＭＵテーブルの実施形態を示す図である。ゲストＶＭ１０３$_1$等のゲスト仮想マシンは、ゲストＶＭのアーキテクチャに準拠したゲストメモリアドレスと、ホストコンピューティングシステムのアーキテクチャに準拠しているホストメモリアドレスとの間の対応関係を示すＩＯＭＭＵテーブルを少なくとも１つ持つとしてよい。ＶＭＭ１０２およびサービスＶＭ１０３は、全てのゲスト仮想マシンのＩＯＭＭＵテーブルを管理するとしてよい。更に、ＩＯＭＭＵページテーブルは、さまざまな方法でインデックスが付与されるとしてよく、例えば、デバイス識別子（例えば、ＰＣＩｅシステムにおけるバス：デバイス：機能番号）でインデックスが付与されたり、ゲストＶＭ番号、または、ＩＯＭＭＵ実施例で特定されるその他の任意の方法でインデックスが付与されるとしてよい。

【００２３】

他の実施形態では他の技術を用いてメモリアクセスを実現し得るものと考えられたい。ある実施形態によると、ＩＯＭＭＵは、例えば、ソフトウェアソリューションによってゲストアドレスがホストアドレスと等しくなる場合には、利用されないとしてよい。別の実施形態によると、ゲストデバイスドライバは、ＶＭＭ１０２と協働して、ＩＯＭＭＵテーブルと同様のマッピングテーブルを利用して、ゲストアドレスをホストアドレスに変換するとしてよい。

【００２４】

図４は、ゲスト仮想マシンによるＩ／Ｏ処理に関連するＩ／Ｏ情報を書き込む方法の実

施形態を示す図である。以下の説明は、ゲストＶＭ１０３$_1$を一例に挙げて行う。他のゲストＶＭについても同一または同様の技術が適用可能であると理解されたい。

【００２５】

ブロック４０１では、ゲストＶＭ１０３$_1$で実行されているアプリケーション１１７$_1$が、Ｉ／Ｏ処理、例えば、ゲストメモリアドレスｘｘｘ－ｙｙｙへの１００個のパケットの書込を指示するとしてよい。ブロック４０２では、ゲストデバイスドライバ１１６$_1$は、Ｉ／Ｏ処理に関するＩ／Ｏ記述子を生成して、ゲストＶＭ１０３$_1$の記述子リング（例えば、図２Ａまたは図２Ｂに示す記述子リング）に書き込むとしてよい。ブロック４０３では、Ｉ／Ｏ処理に関連する全ての記述子が記述子リングに書き込まれる。ある実施形態によると、ゲストデバイスドライバ１１６$_1$は、ヘッドポインタ（例えば、図２Ａに示すヘッドポインタ２０１または図２Ｂに示すヘッドポインタ２２０１）を先頭として、Ｉ／Ｏ記述子を書き込むとしてよい。ブロック４０４において、ゲストデバイスドライバ１１６$_1$は、Ｉ／Ｏ処理に関する記述子が全てバッファに書き込まれた後、テイルポインタ（例えば、図２Ａのテイルポインタ２０２または図２Ｂに示すテイルポインタ２２０２）を更新するとしてよい。

【００２６】

図５は、サービスＶＭ１０３によるＩ／Ｏ処理を行う方法の実施形態を示す。当該実施形態は、ゲスト仮想マシンのゲストデバイスドライバが、ゲスト仮想マシンに割り当てられているＶＩおよび／またはＶＩのドライバに準拠したモードで動作可能であるという条件の場合に適用されるとしてよい。例えば、ゲストデバイスドライバが８２５７１ＥＢベースのＮＩＣのレガシードライバである一方、ＶＩは、８２５７１ＥＢベースのＮＩＣまたは８２５７１ＥＢベースのＮＩＣに準拠した他の種類のＮＩＣ、例えば、８２５７６ＥＢベースのＮＩＣの仮想機能である。以下の説明は、ゲストＶＭ１０３$_1$を一例に挙げて行う。他のゲストＶＭにも同一または同様の技術を適応可能であると理解されたい。

【００２７】

ブロック５０１において、ゲストＶＭ１０３$_1$がテイルポインタ（例えば、図２Ａのテイルポインタ２０２）を更新することによって、仮想マシンイグジット（例えば、ＶＭＥｘｉｔ）がトリガされるとしてよい。仮想マシンイグジットがＶＭＭ１０２によって取得されると、ＶＭＭ１０２は、システムの制御をゲストＶＭ１０３$_1$のゲストＯＳ１１３$_1$からサービスＶＭ１０３のデバイスモデル１１４に移行するとしてよい。

【００２８】

ブロック５０２において、デバイスモデル１１４は、テイル更新に応じて、ＶＩドライバ１１６を呼び出すとしてよい。ブロック５０３－５０６では、ＶＩドライバ１１６が、ゲストＶＭ１０３$_1$に割り当てられたＶＩ１１４１を制御して、ゲストＶＭ１０３$_1$が書き込んだＩ／Ｏ記述子（例えば、図２ＡのＩ／Ｏ記述子）に基づいてＩ／Ｏ処理を実施するとしてよい。具体的には、ブロック５０３において、Ｉ／Ｉ記述子を準備するべく、ＶＩドライバ１１６がＶＩ１１４１を呼び出すとしてよい。ある実施形態によると、ＶＩドライバ１１６は、テイルレジスタ（不図示）を更新することによって、ＶＩ１１４１を呼び出すとしてよい。ブロック５０４において、ＶＩ１１４１は、ゲストＶＭ１０３$_1$（例えば、図２Ａに図示した記述子リング）の記述子リングから記述子を読み出して、Ｉ／Ｏ記述子の記述内容にしたがってＩ／Ｏ処理を実行するとしてよい。例えば、パケットを受信して、当該パケットをゲストメモリアドレスｘｘｘに書き込むとしてよい。ある実施形態によると、ＶＩ１１４１は、記述子リングのヘッドポインタ（例えば、図２Ａのヘッドポインタ２０１）が指し示しているＩ／Ｏ記述子を読み出すとしてよい。

【００２９】

ある実施形態によると、ＶＩ１１４１は、ＩＯＭＭＵまたは同様の技術を利用して、Ｉ／Ｏ処理のためのダイレクトメモリアクセス（ＤＭＡ）を実行するとしてよい。例えば、ＶＩ１１４１は、ゲストＶＭ１０３$_1$のために生成されたＩＯＭＭＵテーブルから、ゲストメモリアドレスに対応するホストメモリアドレスを取得して、メモリシステム１２１に対してパケットの読出または書込を直接実行するとしてよい。別の実施形態によると、Ｖ

Ｉ１１４１は、ゲストアドレスとホストアドレスとの間の固定マッピングにおいてゲストアドレスがホストアドレスに等しい場合には、ＩＯＭＭＵテーブルを利用することなくダイレクトメモリアクセスを実行するとしてよい。ブロック５０５では、ＶＩ１１４１がさらに、Ｉ／Ｏ記述子を更新するとしてよい。例えば、Ｉ／Ｏ記述子に含まれているＩ／Ｏ処理のステータスを更新して、Ｉ／Ｏ記述子が実施された旨を示すとしてよい。ある実施形態によると、ＶＩ１１４１は、Ｉ／Ｏ記述子の更新において、ＩＯＭＭＵテーブルを利用してもしなくてもよい。ＶＩ１１４１はさらに、ヘッドポインタを更新して、ヘッドポインタを進めて、記述子リングの次のＩ／Ｏ記述子を指し示すようにするとしてよい。

【００３０】

ブロック５０６において、ＶＩ１１４１は、テイルが指し示すＩ／Ｏ記述子に到達したか否かを判断するとしてよい。到達していない場合、ブロック５０４および５０５において、ＶＩ１１４１は、記述子リングからのＩ／Ｏ記述子の読出を継続して行い、Ｉ／Ｏ記述子によって指示されたＩ／Ｏ処理を実行するとしてよい。到達している場合、ブロック５０７において、ＶＩ１１４１は、Ｉ／Ｏ処理が完了した旨を、例えば、ＶＭＭ１０２への割り込みを通知することによって、ＶＭＭ１０２に通知するとしてよい。ブロック５０８において、ＶＭＭ１０２は、Ｉ／Ｏ処理が完了した旨を、例えば、サービスＶＭ１０３への割り込みを挿入することによって、ＶＩドライバ１０６に通知するとしてよい。

【００３１】

ブロック５０９において、ＶＩドライバ１１６は、$ＶＩ１１４_1$のステータスを維持して、Ｉ／Ｏ処理が完了した旨をデバイスモデル１１４に通知するとしてよい。ブロック５１０において、デバイスモデル１４がゲスト$ＶＭ１１３_1$に仮想割り込みを通知して、ゲストデバイスドライバ１１６_1が、イベントを処理してＩ／Ｏ処理が実行された旨をアプリケーション１１７_1に通知するとしてよい。例えば、ゲストデバイスドライバ１１６_1は、データが受信され利用する準備が整っている旨をアプリケーション１１７_1に通知するとしてよい。ある実施形態によると、デバイスモデル１１４はさらに、ヘッドレジスタ（不図示）を更新して、記述子リングの制御がゲストデバイスドライバ１１６_1に戻された旨を示すとしてよい。ゲストデバイスドライバ１１６_1への通知は他の方法で実行され、その方法は、デバイス／ドライバポリシー、例えば、ゲストデバイスドライバがデバイス割り込みをディセーブルした場合に作成したデバイス／ドライバポリシーによって決まり得ると考えられたい。

【００３２】

上述した実施形態は説明のためのものであり、他の技術を採用すれば他の実施形態が実現され得るものと考えられたい。例えば、ＶＭＭメカニズムによっては、ＶＩ１１４１は、Ｉ／Ｏ処理が完了した旨を上層のマシンに通知する方法を異ならせるとしてよい。ある実施形態によると、$ＶＩ１１４１_1$は、ＶＭＭ１０２を介するのではなく、直接サービスＶＭ１０３に通知するとしてよい。別の実施形態によると、ＶＩ１１４１は、記述子リングに列挙されているＩ／Ｏ処理の全てではなく１以上が完了した時点で上層のマシンに通知するとしてよい。この構成では、ゲストアプリケーションに、Ｉ／Ｏ処理の一部分が完了した旨が遅滞なく通知されるとしてよい。

【００３３】

図６Ａおよび図６Ｂは、サービスＶＭ１０３によってＩ／Ｏ処理を行う方法の別の実施形態を示す図である。当該実施形態は、ゲスト仮想マシンのゲストデバイスドライバが、ＶＩおよび／またはゲスト仮想マシンに割り当てられているＶＩのドライバに準拠したモードで動作できないという条件において適用されるとしてよい。以下の説明は、ゲスト$ＶＭ１０３_1$を一例に挙げて行う。同一または同様の技術を他のゲストＶＭに適用可能であると理解されたい。

【００３４】

ブロック６０１では、ＶＭＭは、例えば、ゲストデバイスドライバ１１６が仮想デバイス（例えば、デバイスモデル１１４）にアクセスするとゲスト$ＶＭ１０３_1$が発生させる仮想マシンイグジット（例えば、ＶＭＥｘｉｔ）を取得するとしてよい。ブロック６０２

において、ＶＭＭ１０２は、システムの制御を、ゲストＶＭ１０３$_1$のゲストＯＳ１１３$_1$からサービスＶＭ１０３のデバイスモデル１１４へと移行させるとしてよい。ブロック６０３において、デバイスモデル１１４は、仮想マシンイグジットが、ゲストデバイスドライバ１１６$_1$がＩ／Ｏ処理に関連するＩ／Ｏ記述子の記述子リング（例えば、図２Ｂの記述子リング）への書き込みを完了したことに応じてトリガされているか否かを判断するとしてよい。ある実施形態によると、ゲストＶＭ１１３$_1$は、テイルポインタ（例えば、図２Ｂのテイルポインタ２２０２）を更新して、Ｉ／Ｏ記述子の終端を示すとしてよい。この場合、デバイスモデル１１４は、テイルポインタの更新によって仮想マシンイグジットがトリガされたか否かを判断するとしてよい。

【００３５】

Ｉ／Ｏ記述子の書込をゲストデバイスドライバ１１６$_1$が完了したことによって仮想マシンイグジットがトリガされていない場合、図６Ａおよび図６Ｂの方法は、ブロック６０１に戻るとしてよい。つまり、ＶＭＭは次のＶＭイグジットを取得するとしてよい。Ｉ／Ｏ記述子の書込をゲストデバイスドライバ１１６$_1$が完了したことによって仮想マシンイグジットがトリガされている場合、ブロック６０４において、デバイスモデル１１４は、ＶＩドライバ１１６を呼び出して、ゲストＶＭ１０３$_1$のアーキテクチャに準拠しているＩ／Ｏ記述子を、ゲストＶＭ１０３$_1$に割り当てられているＶＩ１４１$_1$のアーキテクチャに準拠しているシャドーＩ／Ｏ記述子へと変換して、シャドーＩ／Ｏ記述子をシャドー記述子リング（例えば、図２Ｂに示すシャドー記述子リング）に格納する。

【００３６】

ブロック６０５において、ＶＩドライバ１１６は、ゲストＶＭ１０３$_1$のアーキテクチャに準拠したテイルポインタを、ＶＩ１４１$_1$のアーキテクチャに準拠したシャドーテイルポインタに変換するとしてよい。

【００３７】

ブロック６０６－６１０では、ＶＩドライバ１１６は、ＶＩ１１４１を制御して、ゲストＶＭ１０３$_1$が書き込んだＩ／Ｏ記述子に基づいてＩ／Ｏ処理を実施するとしてよい。具体的には、ブロック６０６において、ＶＩドライバ１１６は、シャドー記述子の準備を整えるべくＶＩ１１４１を呼び出すとしてよい。ある実施形態によると、ＶＩドライバ１１６は、シャドーテイルポインタ（不図示）を更新することによってＶＩ１１４１を呼び出すとしてよい。ブロック６０７において、ＶＩ１１４１は、シャドー記述子リングからシャドーＩ／Ｏ記述子を読み出して、シャドーＩ／Ｏ記述子の記述内容に従ってＩ／Ｏ処理を実施するとしてよい。例えば、パケットを受信して、当該パケットをゲストメモリアドレスｘｘｘに書き込むか、または、ゲストメモリアドレスｘｘｘからパケットを読み出して当該パケットを送信するとしてよい。ある実施形態によると、ＶＩ１１４１は、シャドー記述子リングのシャドーヘッドポインタ（例えば、図２Ｂのシャドーヘッドポインタ２２０１）が指し示すＩ／Ｏ記述子を読み出すとしてよい。

【００３８】

ある実施形態によると、ＶＩ１１４１は、ＩＯＭＭＵまたは同様の技術を用いて、Ｉ／Ｏ処理のためのダイレクトメモリアクセスを実行するとしてよい。例えば、ＶＩ１１４１は、ゲストＶＭ１０３$_1$のために生成されたＩＯＭＭＵテーブルから、ゲストメモリアドレスに対応するホストメモリアドレスを取得して、受信したパケットをメモリシステム１２１に直接書き込むとしてよい。別の実施形態によると、ＶＩ１１４１は、ゲストアドレスとホストアドレスとの間の固定マッピングにおいてゲストアドレスがホストアドレスに等しい場合には、ＩＯＭＭＵテーブルを利用することなくダイレクトメモリアクセスを実行するとしてよい。ブロック６０８では、ＶＩ１１４１はさらに、シャドーＩ／Ｏ記述子を更新するとしてよい。例えば、シャドーＩ／Ｏ記述子に含まれているＩ／Ｏ処理のステータスを更新して、Ｉ／Ｏ記述子が実施された旨を示すとしてよい。ある実施形態によると、ＶＩ１１４１は、Ｉ／Ｏ記述子の更新のためにＩＯＭＭＵテーブルを利用するとしてよい。ＶＩ１１４１はさらに、シャドーヘッドポインタを更新して、シャドーヘッドポインタを進めて、シャドー記述子リングの次のシャドーＩ／Ｏ記述子を指し示すようにする

としてよい。

【００３９】

ブロック６０９において、ＶＩドライバ１１６は、更新済みのシャドーＩ／Ｏ記述子およびシャドーヘッドポインタをＩ／Ｏ記述子およびヘッドポインタに変換し直して、この新しいＩ／Ｏ記述子およびヘッドポインタで記述子リングを更新するとしてよい。ブロック６１０において、ＶＩ１１４１は、シャドーテイルポインタが指し示すシャドーＩ／Ｏ記述子に到達するか否かを判断するとしてよい。到達していない場合、ブロック６０７－６０９において、ＶＩ１１４１は、シャドー記述子リングからのシャドーＩ／Ｏ記述子の読出を継続して行い、シャドーＩ／Ｏ記述子に記述されているＩ／Ｏ処理を実行するとしてよい。到達している場合、ブロック６１１において、ＶＩ１１４１は、Ｉ／Ｏ処理が完了した旨を、例えば、ＶＭＭ１０２への割り込みを通知することによって、ＶＭＭ１０２に通知するとしてよい。ＶＭＭ１０２はこの後、Ｉ／Ｏ処理が完了した旨を、例えば、サービスＶＭ１０３に割り込みを挿入することによって、ＶＩドライバ１０６に通知するとしてよい。

【００４０】

ブロック６１２において、ＶＩドライバ１１６は、ＶＩ１１４１のステータスを維持して、Ｉ／Ｏ処理が完了した旨をデバイスモデル１１４に通知するとしてよい。ブロック６１３において、デバイスモデル１１４は、ゲストデバイスドライバ116_1に仮想割り込みを通知して、ゲストデバイスドライバ116_1がイベントを処理してＩ／Ｏ処理が実施された旨をアプリケーション117_1に通知するとしてよい。例えば、ゲストデバイスドライバ116_1は、データを受信して利用する準備が整っている旨をアプリケーション117_1に通知するとしてよい。ある実施形態によると、デバイスモデル１４はさらに、ヘッドレジスタ（不図示）を更新して、記述子リングの制御がゲストデバイスドライバ116_1に再び移行される旨を示すとしてよい。ゲストデバイスドライバ116_1への通知は他の方法で実行され、その方法は、デバイス／ドライバポリシー、例えば、ゲストデバイスドライバがデバイス割り込みをディセーブルした場合に作成したデバイス／ドライバポリシーによって決まり得ると考えられたい。

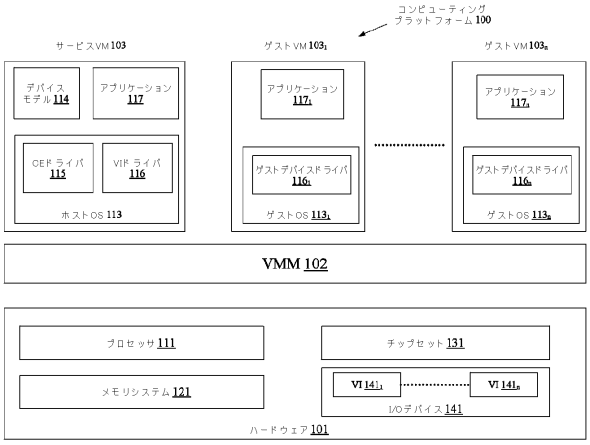
【００４１】

上述した実施形態は説明のためのものであり、他の技術を採用した場合は他の実施形態が実施され得るものと考えられたい。例えば、ＶＭＭメカニズムによっては、ＶＩ１１４１は、Ｉ／Ｏ処理が完了した旨を上層のマシンに通知する方法を異ならせるとしてよい。ある実施形態によると、$ＶＩ１１４１_1$は、ＶＭＭ１０２を介するのではなく、直接サービスＶＭ１０３に通知するとしてよい。別の実施形態によると、ＶＩ１１４１は、記述子リングに列挙されているＩ／Ｏ処理の全てではなく１以上が完了した時点で上層のマシンに通知するとしてよい。この構成では、ゲストアプリケーションに、Ｉ／Ｏ処理の一部分が完了した旨が遅滞なく通知されるとしてよい。

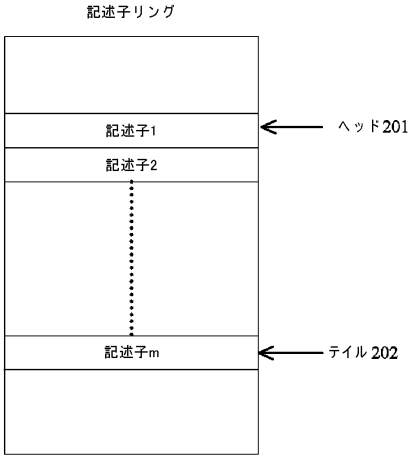
【００４２】

実施形態例に基づき本発明の具体的な特徴を説明してきたが、上記の説明は本発明を限定するものと解釈されるべきものではない。実施形態例のさまざまな変形例、および、本発明の他の実施形態は、本発明の関連技術分野の当業者に明らかであり、本発明の意図および範囲に含まれるものとする。

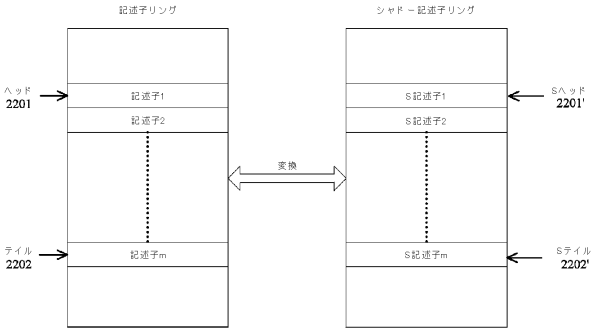
【図１】



【図２Ａ】



【図２Ｂ】

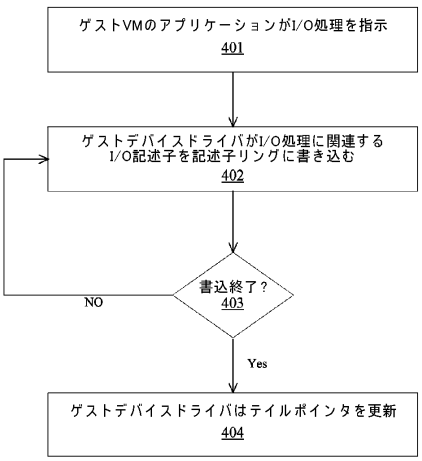


【図３】

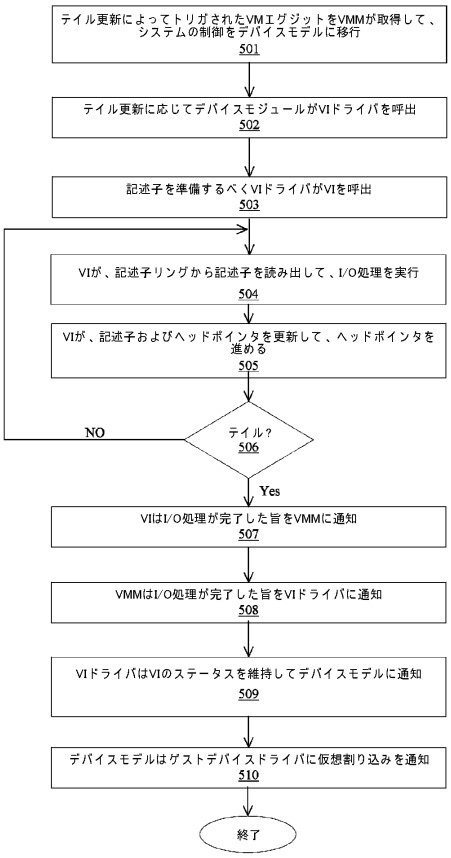
IOMMUテーブル

ゲストアドレス	ホストアドレス

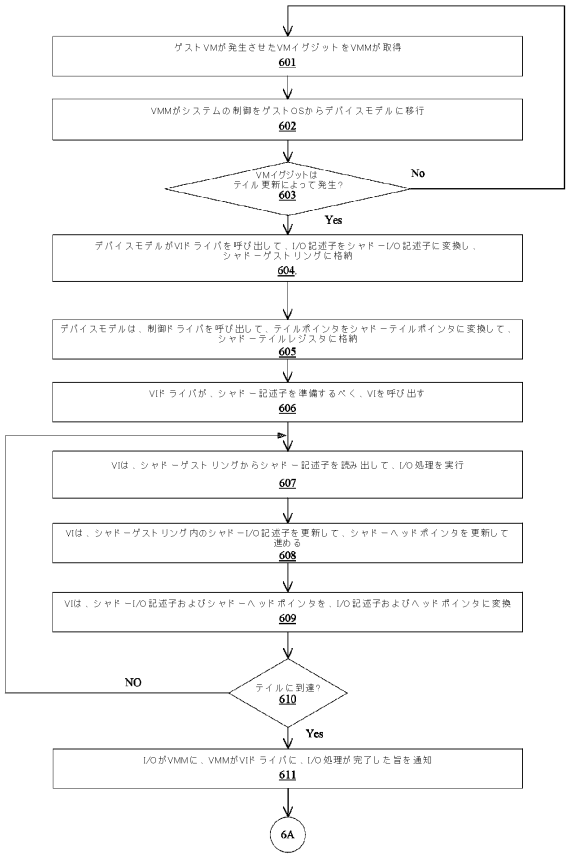
【図４】



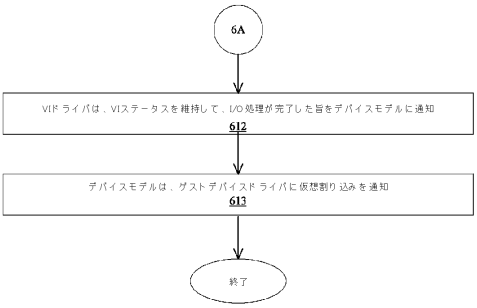
【図５】



【図６Ａ】


ゲストVMが発生させたVMイグジットをVMMが取得
601
VMMがシステムの制御をゲストOSからデバイスモデルに移行
602
VMイグジットは
テイル更新によって発生?
603
No
Yes
デバイスモデルがVIドライバを呼び出して、I/O記述子をシャドーI/O記述子に変換し、
シャドーゲストリングに格納
604
デバイスモデルは、制御ドライバを呼び出して、テイルポインタをシャドーテイルポインタに変換して、
シャドーテイルレジスタに格納
605
VIドライバが、シャドー記述子を準備するべく、VIを呼び出す
606
VIは、シャドーゲストリングからシャドー記述子を読み出して、I/O処理を実行
607
VIは、シャドーゲストリング内のシャドーI/O記述子を更新して、シャドーヘッドポインタを更新して
進める
608
VIは、シャドーI/O記述子およびシャドーヘッドポインタを、I/O記述子およびヘッドポインタに変換
609
NO
テイルに到達?
610
Yes
I/OがVMMに、VMMがVIドライバに、I/O処理が完了した旨を通知
611
6A

【図６Ｂ】


6A
VIドライバは、VIステータスを維持して、I/O処理が完了した旨をデバイスモデルに通知
612
デバイスモデルは、ゲストデバイスドライバに仮想割り込みを通知
613
終了

フロントページの続き

(56)参考文献 国際公開第２００７／１１５４２５（WO，Ａ１）
米国特許出願公開第２００７／０１６８６４１（ＵＳ，Ａ１）
Paul Barhamら，Xen and the Art of Virtualization，SOSP'03 Proceedings of the nineteenth ACM symposium on Operating systems principles，米国，ACM，２００３年１０月１９日，Volume 37 Issue 5，pp.164-177，[online]インターネット<検索日：2013/09/05>，ＵＲＬ，http://dl.acm.org/ft_gateway.cfm?id=945462&ftid=231540&dwn=1&CFID=358346258&CFTOKEN=59321358

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　３／０６－　３／０８
Ｇ０６Ｆ　１２／００－１２／１６
Ｇ０６Ｆ　１３／００－１３／４２