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

特許第6646114号
(P6646114)

(45)発行日　令和2年2月14日(2020.2.14)　(24)登録日　令和2年1月14日(2020.1.14)

(51)Int.Cl.　F I
G06F 9/50 (2006.01)　G06F 9/50 120A
G06F 9/455 (2006.01)　G06F 9/455 150

請求項の数 15　(全 17 頁)

(21)出願番号　特願2018-144824(P2018-144824)
(22)出願日　平成30年8月1日(2018.8.1)
(62)分割の表示　特願2014-93823(P2014-93823)の分割
原出願日　平成26年4月30日(2014.4.30)
(65)公開番号　特開2018-190454(P2018-190454A)
(43)公開日　平成30年11月29日(2018.11.29)
審査請求日　平成30年8月1日(2018.8.1)
(31)優先権主張番号　13/886,360
(32)優先日　平成25年5月3日(2013.5.3)
(33)優先権主張国・地域又は機関
米国(US)

(73)特許権者　510149482
ヴイエムウェア　インコーポレイテッド
ＶＭｗａｒｅ，Ｉｎｃ．
アメリカ合衆国　９４３０４　カリフォルニア州　パロ　アルト　ヒルビュー　アベニュー　３４０１
(74)代理人　100105957
弁理士　恩田　誠
(74)代理人　100068755
弁理士　恩田　博宣
(74)代理人　100142907
弁理士　本田　淳

最終頁に続く

(54)【発明の名称】動的仮想マシンサイジング

(57)【特許請求の範囲】
【請求項１】
ホストコンピュータシステムの複数の物理ＣＰＵ上で実行する仮想マシンの複数の仮想ＣＰＵのスケジュールと、前記ホストコンピュータシステムの複数の物理ＣＰＵ上での実行からの前記仮想マシンの前記複数の仮想マシンのスケジュール解除とを管理する仮想化層によりホストコンピュータシステム上で実行される方法であって、
現在のシステム負荷に基づいて仮想マシンによって要求される仮想ＣＰＵの数に等しい仮想ＣＰＵの要求数を決定すること、
前記仮想マシンが付与された仮想ＣＰＵの数である仮想ＣＰＵの付与数を決定すること、
前記仮想マシンに対する仮想ＣＰＵの目標数は、前記仮想ＣＰＵの要求数と前記仮想ＣＰＵの付与数とのうち少ない方として決定すること、
前記仮想ＣＰＵの目標数を前記仮想マシンに通知すること、
前記仮想ＣＰＵの目標数と前記仮想マシン用に活性化されている仮想ＣＰＵの現在の数とを比較すること、
比較結果に基づいて、１つ以上の仮想ＣＰＵをスケジューリングするかまたはスケジュール解除して、前記仮想マシン用に活性化される仮想ＣＰＵの目標数を達成すること、を備える方法。
【請求項２】
前記仮想ＣＰＵの要求数は、前記仮想マシンの前記複数の仮想ＣＰＵに対する全要求と

、前記仮想マシンの前記複数の仮想ＣＰＵの予想利用率とに基づいて決定される、請求項１に記載の方法。
【請求項３】
前記仮想ＣＰＵの付与数は、前記仮想マシンに設定された仮想ＣＰＵの数と、前記仮想ＣＰＵの準備時間の量とに基づいて決定される、請求項２に記載の方法。
【請求項４】
各仮想ＣＰＵの準備時間は、前記複数の物理ＣＰＵが他の仮想ＣＰＵを実行するのにビジーであるので、仮想ＣＰＵが実行可能であるが物理ＣＰＵ上で実行するようにスケジュールされていない時間の量である、請求項３に記載の方法。
【請求項５】
前記仮想ＣＰＵの目標数の通知を受信すると、前記仮想マシンが、
前記仮想ＣＰＵの目標数が、前記仮想マシン用に活性化された仮想ＣＰＵの現在数よりも少ないことを判定すること、
前記判定することに応答して、
前記仮想マシンのゲストオペレーティングシステムにおいて、仮想ＣＰＵ上の動作に対して、停止命令を含む優先プロセススレッドを立ち上げること、
前記優先プロセススレッドを前記仮想ＣＰＵにピン留めして前記優先プロセススレッドを前記仮想ＣＰＵに維持すること、
前記ゲストオペレーティングシステム内のゲストスケジューラに、前記仮想ＣＰＵを使用して前記優先プロセススレッドを実行させること、をさらに備える請求項１に記載の方法。
【請求項６】
前記仮想ＣＰＵが停止命令を実行していることを検出すること、
前記検出することに応答して、前記ホストコンピュータシステムの１つまたは複数の物理ＣＰＵ上の前記仮想ＣＰＵの実行をスケジュール解除すること、をさらに備える請求項１に記載の方法。
【請求項７】
前記仮想マシンは、バックドアコールを介して前記仮想ＣＰＵの目標数を通知される、請求項１に記載の方法。
【請求項８】
コンピュータシステムであって、
複数の物理プロセッサと、
複数の仮想プロセッサを有するようにそれぞれ構成された複数の仮想マシンと、
前記複数の仮想マシン間の前記複数の物理プロセッサの共有を管理するように構成された仮想化層と、を備え、
前記仮想化層は、
現在のシステム負荷に基づいて仮想マシンによって要求される仮想プロセッサの数に等しい仮想プロセッサの要求数を決定すること、
前記仮想マシンが付与された仮想プロセッサの数である仮想プロセッサの付与数を決定すること、
前記仮想マシンに対する仮想プロセッサの目標数は、前記仮想プロセッサの要求数と前記仮想プロセッサの付与数とのうち少ない方として決定すること、
前記仮想マシンの仮想プロセッサの目標数を前記仮想マシンに通知すること、を実行するように構成され、
前記複数の仮想マシンの各々は、
前記仮想プロセッサの目標数と前記仮想マシン用に活性化されている仮想プロセッサの現在の数とを比較すること、
比較結果に基づいて、１つ以上の仮想プロセッサをスケジューリングするかまたはスケジュール解除して、前記仮想マシン用に活性化される仮想プロセッサの目標数を達成すること、を実行するように構成される、コンピュータシステム。

【請求項９】
前記仮想プロセッサの要求数は、前記仮想マシンの前記複数の仮想プロセッサに対する全要求と、前記仮想マシンの前記複数の仮想プロセッサの予想利用率とに基づいて決定される、請求項８に記載のコンピュータシステム。
【請求項１０】
前記仮想プロセッサの付与数は、前記仮想マシンに設定された仮想プロセッサの数と、前記仮想プロセッサの準備時間の量とに基づいて決定される、請求項９に記載のコンピュータシステム。
【請求項１１】
前記準備時間は、前記複数の物理プロセッサが他の仮想プロセッサを実行するのにビジーであるので、仮想プロセッサが実行可能であるが物理プロセッサ上で実行するようにスケジュールされていない時間の量である、請求項１０に記載のコンピュータシステム。
【請求項１２】
前記仮想マシンは、
前記仮想プロセッサの目標数の通知を受信しかつ前記仮想プロセッサの目標数が、前記仮想マシン用に活性化された仮想プロセッサの現在数よりも少ないことを判定すると、
前記仮想マシンのゲストオペレーティングシステムにおいて、仮想プロセッサ上の動作に対して、停止命令を含む優先プロセススレッドを立ち上げ、
前記優先プロセススレッドを前記仮想プロセッサにピン留めして前記優先プロセススレッドを前記仮想プロセッサに維持し、
前記ゲストオペレーティングシステム内のゲストスケジューラに、前記仮想プロセッサを使用して前記優先プロセススレッドを実行させる、請求項８に記載のコンピュータシステム。
【請求項１３】
前記仮想化層は、
前記仮想プロセッサが停止命令を実行していることを検出することに応答して、前記コンピュータシステムの１つまたは複数の物理プロセッサ上の前記仮想プロセッサの実行をスケジュール解除するように構成される、請求項８に記載のコンピュータシステム。
【請求項１４】
前記仮想マシンは、バックドアコールを介して前記仮想プロセッサの目標数を通知される、請求項８に記載のコンピュータシステム。
【請求項１５】
命令が記録されたコンピュータ可読媒体であって、
実行されると、前記命令は、請求項１～７のいずれか１項に記載の方法を実行させるか、または請求項８～１４のいずれか一項に記載のコンピュータシステムを構成させる、コンピュータ可読媒体。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は概して、コンピュータとコンピュータシステムの分野に関する。いくつかの例では、本発明は、仮想コンピュータ環境における仮想マシンをホストする物理的コンピュータデバイスに関する。
【背景技術】
【０００２】
物理的コンピュータデバイス上に仮想コンピュータ環境を設けることが知られている。仮想コンピュータ環境は、複数の仮想マシン（VM：virtual machine）ゲストが単一物理的プラットホーム上で実行され物理資源を共有できるようにし得る。いくつかの仮想コンピュータ環境は、使用のためにＶＭにより指定されるプロセッサの総数がホスト上で利用可能な物理的プロセッサの実際の数より多くなる方法でＶＭを構成できるようにする。これはＣＰＵオーバーコミットメントと呼ばれ、より多くのＶＭを単一ホスト上に詰め込め

るようにする。
【０００３】
さらに、仮想マシンには２つ以上の仮想ＣＰＵを割り振ることができ、ユーザに、複数のプロセスまたはマルチスレッドアプリケーションを生み出すアプリケーションを実行できるようにする。しかし、その作業負荷よりも多い仮想ＣＰＵ（ｖＣＰＵ）により仮想マシンを構成することは、オーバーヘッドのためより多くの資源を使用する可能性があり、これにより高負荷システムの性能に影響を与えることになる。このシナリオの例としては、複数のｖＣＰＵ仮想マシンにおいて実行するシングルスレッド作業負荷、または作業負荷が有効使用できるより多くのｖＣＰＵを有する仮想マシンにおいて実行するマルチスレッド作業負荷が挙げられる。
【０００４】
さらに、仮想マシンには通常、仮想マシンの配備時にＣＰＵ資源（およびメモリ資源）が割り振られ、これらの割り振りを変更することは通常、仮想マシンをオフラインにし、設定を再構成し、仮想マシンをオンラインに戻すことを含む。このプロセスはシステム管理者にとって時間がかかる可能性があり、仮想マシン上のサービスへのアクセスを妨げる。
【先行技術文献】
【非特許文献】
【０００５】
【非特許文献１】Ｃ．Ａ．ワルドスパージャー（C. A. Waldspurger）著、「ＶＭｗａｒｅ・ＥＳＸ・サーバーのメモリリソース管理（Memory Resource Management in Vmware ESX Server）」、ＯＳＤＩの第５回討論会の議事録（"ln Proceedings of the 5th Symposium on OSDI）、ボストン、マサチューセッツ州、２００２年１２月９～１１日、インターネット<http://www.usenix.org/events/osdi02/tech/waldspurger.html>、１５ページ
【非特許文献２】Ｚ．ムワイカンボ、他著（Z. Mwaikambo, et al.,）、「リナックスカーネルホットプラグＣＰＵサポート（Linux Kernel Hotplug CPU Support）」、オタワでのリナックス討論会（In Ottawa Linux Symposium）、第２巻、オタワ、オンタリオ州、カナダ、２００４年7月２１～２４日、１６ページ
【非特許文献３】Ｋ．アダムスおよびＯ．アジェセン（K. Adams and O.Agesen）著、「ｘ８６仮想化のソフトウェア技術およびハードウェア技術の比較（A Comparison of Software and Hardware Techniques for x86 Virtualization）」、オペレーションシステムのレビュー（Operating Systems Review）、４０（５）：２－１３、２００６年１２月、ISSN 0163-5980、１２ページ
【非特許文献４】VMware lnc.,、「ゲストオペレーティングシステム要求のタイマー割り込みのレート決定および変更（Determining and Changing the Rate of Timer lnterrupts a Guest Operating System Requests）」、インターネット<http://kb.vmware.com/kb/1005802>、更新：２０１１年1月２５日、４ページ
【非特許文献５】VMware lnc.,、「ＶＭｗａｒｅ・ｖＳｐｈｅｒｅ　５．０　のベスト動作の実行（Performance Best Practices for VMware vSphere 5.0）」、インターネット<http://www.vmware.com/pdf/Pert_Best_Practices_vSphere5.0.pdf>、改訂：２０１１年８月２２日、７６ページ
【発明の概要】
【発明が解決しようとする課題】
【０００６】
ここで、仮想コンピュータ環境を実現する際に物理的コンピュータプラットホームの資源をより効率的に使用することなど現在利用可能なコンピュータシステムを改善する要望がある。例示的実施形態の他の利点は以下に記載されるか、または本明細書の教示を実行することにより当業者らにより理解される。
【課題を解決するための手段】

【０００７】
本発明によると、添付の特許請求の範囲に記載されるコンピュータシステムと方法が提供される。本発明の他の特徴は従属請求項と以下の説明から明確になる。
本開示の実施形態は、ホストであってホスト上で実行する仮想マシンを有するホスト内のＣＰＵを管理する方法を提供する。仮想マシンには複数の仮想ＣＰＵが割り振られる。本方法は、仮想マシンによるプロセッサ要求に基づき、仮想マシン用に活性化された仮想ＣＰＵの現在数よりも多い仮想ＣＰＵの目標数を決定することを含む。本方法はさらに、仮想マシンのゲストオペレーティングシステム内の優先プロセススレッドを立ち上げることを含む。優先プロセススレッドは、複数の仮想ＣＰＵのうちの第１の仮想ＣＰＵに関連付けられ、且つ停止命令を含む。本方法は、ゲストオペレーティングシステム内のゲストスケジューラの動作により、複数の仮想ＣＰＵのうちの第１の仮想ＣＰＵを使用して優先プロセススレッドを実行することを含む。本方法はさらに、ホスト内のハイパーバイザの動作により、第１の仮想ＣＰＵが停止命令を実行していることを検出することに応答して、ホストの１つまたは複数の物理的ＣＰＵ上の第１の仮想ＣＰＵの実行をスケジュール解除することを含む。
【図面の簡単な説明】
【０００８】
【図１】実施形態が実施され得る仮想化コンピュータアーキテクチャを示すブロック図である。
【図２】仮想マシンの目標仮想ＣＰＵサイズを決定するために仮想化層により実行される工程を示す流れ図である。
【図３】カーネルスケジューラから目標仮想ＣＰＵサイズを受信することに応答して仮想マシンの目標仮想ＣＰＵサイズを実装するためにバルーンドライバにより実行される工程を示す流れ図である。
【図４Ａ】仮想ＣＰＵサイズがカーネルスケジューラから受信された目標仮想ＣＰＵサイズと同じである場合の、仮想マシンのゲストスケジューラによる仮想ＣＰＵへのスレッドのディスパッチングを示すブロック図である。
【図４Ｂ】仮想ＣＰＵサイズがカーネルスケジューラから受信された目標仮想ＣＰＵサイズを超えた場合の、スレッドを複数の仮想ＣＰＵへディスパッチし、バルーンスレッドを１つの仮想ＣＰＵへディスパッチする仮想マシンのゲストスケジューラを示すブロック図である。
【発明を実施するための形態】
【０００９】
本開示の態様はプロセス、装置、システム、デバイス、コンピュータ読み取り可能媒体上の方法など多くの方法で実施されることができるということを理解すべきである。本開示のいくつかの実施形態について以下に説明する。
【００１０】
図１は、実施形態が実施され得る仮想化コンピュータアーキテクチャを表すコンピュータシステム１００のブロック図を描写する。図示のように、コンピュータシステム１００は、共通のハードウェアプラットホーム１０２上で実行し共通のハードウェアプラットホーム１０２を共有する複数の仮想マシン（ＶＭ）118_1～118_Nをホストする。ハードウェアプラットホーム１０２は、１つまたは複数の中央処理装置（CPU : central processing unit）１０４、ランダムアクセスメモリ（RAM : random access memory）１０６、１つまたは複数のネットワークインタフェース１０８、および固定記憶装置１１０など従来のコンピュータハードウェアの構成要素を含む。
【００１１】
以下では、ハイパーバイザ１１１と呼ばれる仮想化ソフトウェア層が、ハードウェアプラットホーム１０２上に設けられる。ハイパーバイザ１１１は、１つまたは複数のＶＭ118_1～118_Nの並列インスタンス化と実行を可能にする。ＶＭ１１８とハイパーバイザ１１１との相互作用は、仮想マシンモニタ（VMM : virtual machine monitor）により

容易にされる。各ＶＭＭ１３４$_{1}$～１３４$_{N}$は、対応するＶＭ１１８$_{1}$～１１８$_{N}$に割り当てられ、対応するＶＭ１１８$_{1}$～１１８$_{N}$を監視する。一実施形態では、ハイパーバイザ１１１は、米国カリフォルニア州パロアルトのＶＭｗａｒｅ（商標）社から入手可能なＶＭｗａｒｅのｖＳｐｈｅｒｅ（登録商標）仮想化製品の商品として実装されるＶＭｋｅｒｎｅｌ（商標）であり得る。別の実施形態では、ホストオペレーティングシステムは、ハイパーバイザ１１１とハードウェアプラットホーム１０２との間に設けられる。このような実施形態では、ハイパーバイザ１１１は、ホストオペレーティングシステムにより提供される抽象化レベル上で動作する。

【００１２】

インスタンス化（instantiation）後、各ＶＭ１１８$_{1}$～１１８$_{N}$は、ハイパーバイザ１１１の制御下で実行される物理的コンピュータプラットホームを隠蔽する（encapsulate）。ＶＭ１１８の仮想デバイス（virtual device）は、これらに限定しないが、１つまたは複数の仮想中央処理装置（virtual CPU）（ｖＣＰＵ）１２２$_{1}$～１２２$_{N}$、仮想ランダムアクセスメモリ（ｖＲＡＭ）１２４、仮想ネットワークインタフェースアダプタ（ｖＮＩＣ）１２６、および仮想記憶装置（ｖＳｔｏｒａｇｅ）１２８からなる仮想ハードウェアプラットホーム１２０内に具現される。仮想ハードウェアプラットホーム１２０は、アプリケーション１３２を実行することができるゲストオペレーティングシステム（ゲストＯＳ）１３０の実装（installation）を支援する。ゲストＯＳ１３０の例としては、マイクロソフトウインドウズ（登録商標）、リナックス（登録商標）などの周知の汎用オペレーティングシステムの任意のものが挙げられる。

【００１３】

図１に示す実施形態では、ゲストＯＳ１３０は、複数のコンピュータタスク（プロセスと呼ばれる）が同じ期間中に行われ共通の処理資源を共有する方法であるマルチタスキング（multitasking）を支援するスケジューラ部品（ゲストスケジューラ１３３として描写される）を含む。ゲストスケジューラ１３３は、様々なアルゴリズム（例えばラウンドロビン（round-robin）、ファーストインファーストアウト、プリエンプティブスケジューリング（pre-emptive scheduling）など）を同時に使用することにより実行しコンピュータ資源（例えばｖＣＰＵ１２２）にアクセスする複数のプロセスをスケジューリングしディスパッチするように構成される。例えば、ゲストスケジューラ１３３は、高い実行優先順位を有するプロセスが低い優先順位を有するプロセスよりも多くの時間を許可されるように、プロセスの優先順位を考慮することによりｖＣＰＵ１２２へのアクセスを管理し得る。別の例では、マルチプロセッサ環境（例えば複数のｖＣＰＵ１２２を有するＶＭ）では、ゲストスケジューラ１３３は、他のｖＣＰＵよりビジー状態でないｖＣＰＵ１２２へプロセスをディスパッチし得る。

【００１４】

上述のように、コンピュータシステム１００内で実行するＶＭ１１８は、１乃至多数のｖＣＰＵ１２２$_{1}$～１２２$_{N}$を有するように構成することができる（Ｎ個のｖＣＰＵを有するＶＭはＮ－ウェイ・仮想マシンと呼ばれることもある）。論述のために、本明細書で使用される「大きな」仮想マシンは多くのｖＣＰＵを有する仮想マシンを指し、本明細書で使用される「小さな」仮想マシンはわずかなｖＣＰＵを有する仮想マシンを指す。一実施形態では、ＶＭ１１８は、最大６４のｖＣＰＵを有するように構成され得る。しかし、いくつかの場合には、多くのｖＣＰＵ１２２を有するＶＭ１１８は、内在する物理的ＣＰＵ１０４の利用率、スループット、他の性能メトリックという点で、少ないｖＣＰＵ１２２を有するＶＭ１１８よりも低効率で動作し得る。多くの要因が、大きなＮ－ウェイ・ＶＭの非効率に寄与し得る。未使用ｖＣＰＵは、いくつかのゲストオペレーティングシステムにおいてタイマー割り込み（timer interrupt）を依然として実行し続ける。ゲストスケジューラ１３３は、シングルスレッド作業負荷を複数のｖＣＰＵ間で不必要に移動し得、これによりキャッシュ局所性（cache locality）を失う。ゲストＯＳ１３０は、非活性の期間中に、そうでなければ他の使用のために利用可能な資源を消費することになるアイドルループを実行し得る。ＶＭ内で実行するすべてのｖＣＰＵの仮想メモリについての一

貫した見方を維持することは、ゲストＯＳ１３０と内在するハイパーバイザ１１１の両方において追加資源を消費する可能性がある。このような問題のために、システム管理者は、最近のコンピュータアプリケーションの要求がより厳しくなり、ますます大きな仮想マシンを必要としたとしても、２－ウェイＶＭを超えるものを提供することに乗り気でないこともある。このＶＭサイジング問題は、ＶＭ効率（例えば、ユーザに８－ウェイ・ＶＭを与えることで効率問題を引き起こす）とＶＭ機能との間の矛盾を引き起こす（例えば、ユーザに２－ウェイ・ＶＭを与えることで、大きなＶＭを必要とするハイエンドアプリケーションを要求する用途を排除する）。したがって、本開示の実施形態は、ＶＭにより必要とされないｖＣＰＵを動的に「非活性化する」技術（本明細書ではＣＰＵバルーニング（CPU ballooning）と呼ぶ）を提供する。これは、多くの仮想ＣＰＵを有する仮想マシンを実行する効率費用を発生することなく大きなＮ－ウェイ・仮想マシンを提供する。

【００１５】

本開示の実施形態は、ホストであってホスト上で実行する仮想マシンを有するホスト内のＣＰＵ資源を管理するためのＣＰＵバルーニングとして知られた方法またはシステムを提供する。図１は、ゲストＯＳ１３０下において設けられたバルーンドライバ１３１を描写する。一実施形態では、バルーンドライバ１３１は、ゲストＯＳ１３０下で実行するデバイスドライバである。いくつかの実施形態では、バルーンドライバ１３１は定期的に実行する、すなわち、バルーンドライバ１３１はタイマーイベントによりトリガされるまでアイドル状態のままである。実行後、バルーンドライバ１３１は別のタイマーイベントにより再びトリガされるまでアイドル状態に戻る。タイマーイベントの頻度は管理者により設定され得る。バルーンドライバ１３１は、ＶＭ１１８による使用のために利用可能なｖＣＰＵ１２２の数を調整するためにカーネルスケジューラ１１３から情報を得る。バルーンドライバ１３１は、ＶＭ１１８用に活性化されたｖＣＰＵの現在数のカウントを維持するように構成される。例えば、ＶＭ（例えば$VM118_1$）が最初に開始されると、バルーンドライバ１３１は、$VM118_1$に割り振られたｖＣＰＵの数と等しいｖＣＰＵの現在数のカウントを設定し、この設定にしたがって、以下に述べられる動作中にカウントをインクリメントおよびディクリメントする。

【００１６】

図１にさらに示すように、カーネルスケジューラ１１３は、ハイパーバイザ１１１の構成要素である。カーネルスケジューラ１１３は、所定時間にコンピュータシステム１００上で実行する様々なプロセス間に物理的ＣＰＵ１０４を割り振ることに関与する。本明細書で使用されるプロセスは、コンピュータプログラムを実行することである。例えば、カーネルスケジューラ１１３は、どのプロセスがＣＰＵ１０４（またはマルチプロセッサコンプレックスにおける多くのＣＰＵのうちの任意のものの）上で実行されるべきかと、このようなプロセスが実行される順序と、どの実行プロセスが高い優先順位を有するプロセスによりプリエンプトされる（preempted）べきかとを判定し得る。そのスケジューリングの判定をなすために、カーネルスケジューラ１１３は、実行の準備ができている一組のプロセスと、現在実行中のプロセスのそれぞれの優先順位と、１つまたは複数のシステム資源を現在待っている一組のプロセスとを含む特定データを使用し得る。

【００１７】

物理的ＣＰＵ１０４へのアクセスの管理に加えて、カーネルスケジューラ１１３は、本明細書に記載のいくつかの実施形態では、特定のＶＭ１１８が所与時点で使用すべきｖＣＰＵ１２２の目標数である目標ｖＣＰＵサイズを決定するように構成される。この目標ｖＣＰＵサイズは、例えばバックドア（backdoor）インタフェースに対する呼び出しを使用することによりゲストＯＳ１３０のバルーンドライバ１３１へ伝達される（方向線１１４により描写される）。次に、バルーンドライバ１３１は、以下にさらに詳細に説明されるように、ゲストスケジューラ１３３がプロセスをディスパッチするｖＣＰＵ１２２の数を調整するためにこの推奨（recommendation）を利用する。例えば、ＶＭ１１８が自由に有するｖＣＰＵ１２２が十分に利用されない場合、バルーンドライバ１３１はＶＭ１１８による使用のために利用可能なｖＣＰＵ１２２の数を低減する。対照的に、カーネルスケジ

ューラ１１３がＶＭ１１８により使用されるｖＣＰＵの数を超えるＶＭ１１８の目標ｖＣＰＵサイズを与えると、バルーンドライバ１３１は、ＶＭ１１８による使用のために利用可能なｖＣＰＵ１２２の数を増加しようと試みる。

【００１８】

図１の構成要素を説明するために使用される様々な用語、層、カテゴリ化は、それらの機能または本開示の精神または範囲から逸脱することなく異なる方法で参照され得るということを認識すべきである。例えば、ＶＭＭ１３４$_1$～１３４$_N$は、インスタンス化されたＶＭ毎に別個のＶＭＭが存在するのでＶＭ１１８$_1$～１１８$_N$とハイパーバイザ１１１間の別個の仮想化部品と考えられ得る。または、このようなＶＭＭが仮想マシンのハードウェアエミュレーションの要素を含むので、各ＶＭＭは、その対応する仮想マシンの要素であると考えられ得る。このような代替概念では、例えば、仮想ハードウェアプラットホーム１２０として説明された概念的層はＶＭＭ１３４にマージされ得る。

【００１９】

図２は、特定のＶＭの目標ｖＣＰＵサイズを決定する際にカーネルスケジューラ１１３により実行される工程を描写する流れ図である。図２に示すように、工程２００では、ＶＭにより要求されたｖＣＰＵの数がコンピュータシステム１００の現在のシステム負荷に基づき決定される。一実施形態では、カーネルスケジューラ１１３は、式１に示すように、ｖＣＰＵの要求数を決定する。

【００２０】

【数１】

$$\text{demanded}_{\text{vcpus}} = \lceil \text{demand}_{\text{VM}} / \text{expectedUtilRatio}_{\text{VM}} \rceil \qquad (1)$$

ｖＣＰＵの要求数（すなわちｄｅｍａｎｄｅｄ$_{ｖｃｐｕｓ}$）は、（１）ＶＭ１１８に関連付けられたすべてのｖＣＰＵ１２２の全要求（すなわちｄｅｍａｎｄ$_{ＶＭ}$）と、（２）ＶＭ１１８に関連付けられたすべてのｖＣＰＵ１２２の予想利用率（すなわちｅｘｐｅｃｔｅｄＵｔｉｌＲａｔｉｏ$_{ＶＭ}$）とに基づく。

【００２１】

一実施形態では、ｖＣＰＵの要求は、「奪われた」時間（stolen time）が無い場合にｖＣＰＵが消費することができる時間である。ｖＣＰＵの奪われた時間は、準備時間、オーバーラップサイクル、パワーマネージメントに対する時間損失、ハイパースレッディング（Hyper-threading）により奪われた時間、および他の変数を含む。準備時間は、システムが他のｖＣＰＵを実行するのにビジーであるので、ｖＣＰＵは実行可能であるが物理的ｖＣＰＵ上で実行するようにスケジュールされていない時間の量である。オーバーラップサイクルは、割り込みと、このｖＣＰＵの実行をプリエンプトしたボトムハーフ（BHs: bottom halves）とにより奪われた時間の量である。パワーマネージメントによる損失時間は、周波数スケーリングのための効率損失を表す。例えば、周波数が公称周波数の２０パーセントまで落ちると、ＣＰＵの８０パーセントが奪われたと考えられる。ハイパースレッディング（hyper-threading）に対する時間損失は、パートナーである物理的ＣＰＵ上で実行する作業負荷により奪われた時間を表す。

【００２２】

ｖＣＰＵの要求は、実際に使用されるサイクルの量と、「奪われた」サイクルが無い場合にｖＣＰＵが使用していたであろうサイクルの量とに基づき推定され得る。一実施形態によると、ＶＭ１１８に関連付けられたｖＣＰＵの全要求（すなわちｄｅｍａｎｄ$_{ＶＭ}$）は、式２のように計算される。

【００２３】

【数２】

$$\text{demand}_{\text{vcpu}} = \text{CyclesUsed}_{\text{vcpu}} + \text{CyclesStolen}_{\text{vcpu}} * \text{CyclesCapacity}_{\text{vcpu}} \quad (2)$$

式２が示すように、ｖＣＰＵ１２２の要求は、（１）所定期間内にＶＭ１１８内で実行

するｖＣＰＵ１２２により使用されるサイクルの百分率（すなわちＣｙｃｌｅｓＵｓｅｄ$_{ｖｃｐｕ}$）、（２）所定期間にＶＭ１１８内で実行するｖＣＰＵ１２２から「奪われた」サイクルの百分率（すなわちＣｙｃｌｅｓＳｔｏｌｅｎ$_{ｖｃｐｕ}$）、および（３）同じ期間に実行するための容量をｖＣＰＵ１２２が有するサイクルの百分率（すなわちＣｙｃｌｅｓＣａｐａｃｉｔｙ$_{ｖｃｐｕ}$）に基づく。ｖＣＰＵ１２２により使用されるサイクルは、当該ｖＣＰＵ１２２が命令を実行するサイクルである。対照的に、ｖＣＰＵ１２２から奪われたサイクルは、ｖＣＰＵ１２２が実行すべき命令を有するサイクルであるが、例えばシステム負荷のためにそれらの命令を実行することからプリエンプトされる。奪われたサイクルの例としては、ｖＣＰＵ１２２は実行準備されたが、他のＶＭのプロセスを実行するコンピュータシステム１００のためにディスパッチされなかったサイクルと、ｖＣＰＵ１２２が外部割り込みを処理するコンピュータシステム１００によりプリエンプトされたサイクルとが挙げられる。最後に、ｖＣＰＵ１２２の容量（すなわちＣｙｃｌｅｓＣａｐａｃｉｔｙ$_{ｖｃｐｕ}$）は、ｖＣＰＵ１２２が「奪われた」サイクルが無い場合に所定期間にわたって消費する能力を有するサイクルの百分率である。さらに、ＶＭ１１８の要求（すなわちｄｅｍａｎｄ$_{ＶＭ}$）は、ＶＭ１１８内で実行するｖＣＰＵ１２２の要求の合計である。

【００２４】

式２に示すように、使用されるサイクルの百分率（ＣｙｃｌｅｓＵｓｅｄ$_{ｖｃｐｕ}$）は、ｖＣＰＵ１２２の奪われたサイクルの百分率（ＣｙｃｌｅｓＳｔｏｌｅｎ$_{ｖｃｐｕ}$）とｖＣＰＵの所定期間にわたる容量（ＣｙｃｌｅＣａｐａｃｉｔｙ$_{ｖｃｐｕ}$）の積に加算され、その結果はｖＣＰＵの現在の要求（ｄｅｍａｎｄ$_{ｖｃｐｕ}$）として使用される。例えば、所定期間にわたるＶＭ１１８内で実行するｖＣＰＵ１２２により使用されるサイクルの百分率が３０であり、所定期間にわたるｖＣＰＵ１２２から奪われたサイクルの百分率が５０であり、単一ｖＣＰＵ１２２の容量がその同じ期間にわたって４０パーセントである場合、ｖＣＰＵ１２２の現在の要求は３０＋５０×４０％となり、５０パーセントに等しい。

【００２５】

予想利用率（すなわちｅｘｐｅｃｔｅｄＵｔｉｌＲａｔｉｏ$_{ＶＭ}$）は、ハイパーバイザ１１１によりインスタンス化されたＶＭ毎に構成可能な値であり、ＶＭ１１８のｖＣＰＵ１２２が有すべきかつ、許容可能な性能を依然として提供すべき利用率（百分率表現）を表す。予想利用率は、管理者によりＶＭ１１８が構成されるときに設定され、ＶＭ１１８の実行中に変更され得る。例えば、予想利用率は、システム利用が７０％以下のときにＶＭ内で実行するアプリケーションが十分に動作し続け得るという判定に基づき、７０％として構成され得る。

【００２６】

ＶＭ１１８上の現在の要求（ｄｅｍａｎｄ$_{ＶＭ}$）とＶＭ１１８の予想利用率（ｅｘｐｅｃｔｅｄＵｔｉｌＲａｔｉｏ$_{ＶＭ}$）が決定されると、カーネルスケジューラ１１３は、ｖＣＰＵ１２２の要求数（すなわちｄｅｍａｎｄｅｄ$_{ｖｃｐｕｓ}$）を式１のように計算する。例えば、ＶＭ１１８が１１０％の現在要求と７０％の予想利用率を有すると、ｖＣＰＵ１２２の要求数は２となる（［１１０／７０］＝２であるので）。

【００２７】

工程２１０では、カーネルスケジューラ１１３は、ＶＭ用に構成されたｖＣＰＵの数とｖＣＰＵの準備時間の量とに基づき、所与のＶＭに付与されるｖＣＰＵの数を決定する。所与のＶＭの有効なＣＰＵ資源権利は、例えばＶＭを実行するシステムがオーバーコミットされた場合、またはＶＭの資源割振りが小さい場合、またはその両方の場合、その要求よりも小さくてもよい。したがって、このような場合には少ないｖＣＰＵを有するＶＭを実行する方が効率的であり得ると判定された。一実施形態では、カーネルスケジューラ１１３は、残りのｖＣＰＵが少ない準備時間を有するようにＶＭ用の低減された数のｖＣＰＵを決定し、これにより、より効率的に実行する。

【００２８】

一実施形態では、カーネルスケジューラ１１３は、ＶＭ１１８に付与されるｖＣＰＵの有効数（すなわち$entitled_{vcpus}$）を下記の式３により決定する。

【００２９】

【数３】

$$entitled_{vcpus} = num_{vcpus} - \lfloor ready \rfloor. \qquad (3)$$

いくつかの実施形態では、カーネルスケジューラ１１３は、ＶＭ１１８用に定義されるｖＣＰＵ１２２の数（すなわちnum_{vcpus}）（ＶＭ１１８が構成された時点で設定される）を最初に記録することにより、ＶＭ１１８に付与されるｖＣＰＵ１２２の数（すなわち$entitled_{vcpus}$）を決定する。次に、カーネルスケジューラ１１３は、ＶＭのすべてのｖＣＰＵの準備時間の量を決定する。上述のように、準備時間は、ＶＭが実行したいが実行するための物理的ＣＰＵ資源が提供されていない時間である。一実施形態では、準備時間は百分率形式で表され得る、例えば、５％（すなわち０．０５）の準備時間を有するＶＭは、利用可能なＣＰＵ資源を待つその最後のサンプル期間の５％をＶＭが費やしたことを意味する。したがって、一例では、８－ウェイ・ＶＭがレディ（ＲＥＡＤＹ）状態に時間の２００パーセントを費やせば、権利を付与されるＣＰＵの数は６である（８－｜２．００｜＝６であるので）。

【００３０】

工程２２０では、カーネルスケジューラ１１３は、ｖＣＰＵの要求数（式１において計算される$demanded_{vcpus}$）と特定のＶＭのｖＣＰＵの付与数（式３において計算される$entitled_{vcpus}$）とのうちの少ない方に基づいて、式４に記載されるように、特定のＶＭ１１８の目標ｖＣＰＵサイズ（すなわち$target_{vcpus}$）を決定する。

【００３１】

【数４】

$$target_{vcpus} = \min(demanded_{vcpus}, entitled_{vcpus}) \qquad (4)$$

図３は、その周期的実行サイクルのうちの１サイクル中にバルーンドライバ１３１により実行される工程を示す流れ図である。最初に、バルーンドライバ１３１はタイマーイベントによりトリガされた後「ウェイクアップする（wake up）」。工程３００では、バルーンドライバ１３１は、ハイパーバイザ１１１（特にカーネルスケジューラ１１３を有する）と通信し、カーネルスケジューラ１１３がＶＭ１１８用に計算した目標ｖＣＰＵサイズを受信する。この目標ｖＣＰＵサイズは、カーネルスケジューラ１１３の見積もりの際にＶＭ１１８がその仮想ハードウェアプラットホーム１２０において使用すべきｖＣＰＵ１２２の数を表す。次に、工程３０５では、バルーンドライバ１３１は、この目標ｖＣＰＵサイズとＶＭ１１８が現在使用しているｖＣＰＵ１２２の数とを比較する。ＶＭ１１８が現在使用しているｖＣＰＵ１２２の数が目標ｖＣＰＵサイズを超える場合、工程３２０においてバルーンドライバ１３１は、受信された目標ｖＣＰＵサイズに基づき１つまたは複数のバルーンスレッド（balloon thread）を立ち上げる。工程３２５では、バルーンドライバ１３１は、バルーンスレッドが実行することになる１つまたは複数のｖＣＰＵ１２２を規定する（すなわち、バルーンドライバ１３１がスレッドをｖＣＰＵ１２２に「ピン留めする（pin）」）。このとき、各バルーンスレッドは１つのｖＣＰＵ１２２にピン留めされる。

【００３２】

一実施形態では、バルーンスレッドは、ゲストＯＳ１３０のゲストスケジューラ１３３が$vCPU122_1$をスケジューリング目的のために利用不能と認識するように、特定のｖＣＰＵ（例えばｖＣＰＵ１）を占有するように構成されるスレッドである。いくつかの実施形態では、バルーンスレッドは、ゲストスケジューラ１３３が特定のｖＣＰＵ上のバルーンスレッドの実行をプリエンプトも中断もし得ないように、ゲストＯＳ１３０内で実行する他の処理に比べて高いプロセス優先順位を有する優先プロセススレッドである。い

くつかの実施形態では、バルーンスレッドは、カーネルスレッド、軽量プロセス（LWP：lightweight process）、またはゲストＯＳ１３０内で実行する他のプロセスであり得る。バルーンドライバ１３１はさらに、非活性化対象のｖＣＰＵへ「ピン留めされる」バルーンスレッドを構成し得る。例えば、バルーンドライバ１３１は、バルーンスレッドが特定のｖＣＰＵに拘束されるべきであるということをゲストスケジューラ１３３にシグナリングするプロセッサ親和性設定（processor affinity setting）をバルーンスレッドに対し設定し得る。バルーンスレッドは終了するまで実行を継続するように構成される。

【００３３】

一実施形態では、バルーンスレッドがピン留めされた特定のｖＣＰＵ１２２はスケジュール解除されるべきであるということをハイパーバイザ１１１へ伝達するために、バルーンスレッドはさらに、アイドル命令を実行するように構成され得る。したがって、ハイパーバイザ１１１は、特定のｖＣＰＵ１２２を維持するオーバーヘッドを発生しない。一例では、優先プロセススレッド（バルーンスレッド）は停止命令を含む。１つの特定の実施形態では、バルーンスレッドは、より多くの仕事を行う必要があるまで処理装置（例えばｖＣＰＵ１２２）を停止させるコンピュータ命令を有し得る。これにより、処理装置は停止状態（すなわちレディ状態）に移行する。ｘ８６コンピュータアーキテクチャのバルーンスレッドの実施形態の例を以下の表１に擬似コードとして示す。

【００３４】

【表１】

```
while (1) {

   HLT;

}
```

表１　バルーンスレッドのサンプル擬似コード

示されるように、バルーンスレッドは、ＨＬＴ命令（より多くの仕事を行う必要があるまで処理装置を停止させるアセンブリ言語命令）を繰り返し発行するループとして（例えば、割り込み駆動型プロセッサ内で次の外部割り込みが発射されるまで）実施され得る。他の実施形態では、バルーンスレッドは、スリープモード、ＭＯＮＩＴＯＲ、ＭＷＡＩＴまたは他の機能的に等価な命令を含み得る。いくつかの実施形態では、ハイパーバイザ１１１は、アイドル命令を実行しているＶＭ１１８上で実行する任意のゲストプロセスを検知し、アイドル命令が実行されている任意のｖＣＰＵを、物理的ＣＰＵ上での実行からスケジュール解除するように構成される。これらの実施形態では、バルーンスレッド内のＨＬＴ命令は、バルーンスレッドを実行する特定のｖＣＰＵ１２２がスケジュール解除され得るということをハイパーバイザ１１１のカーネルスケジューラ１１３へ伝達する役目を果たす。したがって、ゲストスケジューラ１３３の観点からは、立ち上げられたバルーンスレッドはｖＣＰＵを生じない高優先順位スレッドである。その一方で、カーネルスケジューラ１１３の観点からは、ｖＣＰＵはバルーンスレッドのアイドルループにより停止され、物理的ＣＰＵからスケジュール解除され得る。

【００３５】

一実施形態では、バルーンドライバ１３１は、ｖＣＰＵの目標数とｖＣＰＵの現在数との差を満たすためにある数のバルーンスレッドを立ち上げる。各バルーンスレッドを立ち上げ、ピン留めした後、バルーンドライバは、仮想マシン用に活性化されたｖＣＰＵの現在数のカウントを更新する。例えば、バルーンドライバ１３１は、立ち上げられたバルーンスレッド毎のＶＭ用に活性化された仮想ＣＰＵの現在数のカウントをディクリメントし得る。

【００３６】

工程３３０では、バルーンドライバ１３１が、カーネルスケジューラ１１３から、ＶＭ１１８により現在使用中のｖＣＰＵ１２２の数を超えるｖＣＰＵ１２２の目標数を受信すると、工程３４０において、バルーンドライバ１３１は、以前に立ち上げられたバルーンスレッドがｖＣＰＵ１２２の１つへピン留めされたかどうかを判定する。このようなバルーンスレッドが立ち上げられ、ｖＣＰＵ１２２上で現在実行中であれば、工程３４５において、バルーンドライバ１３１は、ゲストＯＳ１３０内のバルーンスレッドの実行を停止（kill）する。この停止は、ゲストスケジューラ１３３の観点からは他のプロセスをスケジューリングするために当該ｖＣＰＵ１２２を解放することになる。さらに、バルーンドライバ１３１は、プロセススケジューリングのための追加のｖＣＰＵ１２２を解放するために可能な限り多くのバルーンスレッドの実行を停止する。一実施形態では、バルーンスレッドを停止することに応答して、バルーンドライバは、仮想マシン用に活性化されたｖＣＰＵの現在数のカウントを更新する。例えば、バルーンドライバ１３１は、停止されたバルーンスレッド毎にＶＭ用に活性化された仮想ＣＰＵの現在数のカウントをインクリメントし得る。

【００３７】

前述の工程を実行した後、バルーンドライバ１３１は、タイマー割り込みにより再びトリガされるまで、アイドル状態（すなわち「スリープ」）に戻る。

図４Ａは、プロセスがゲストスケジューラ１３３によりスケジュールされ且つディスパッチされる４つの割り振られたｖＣＰＵ１２２$_1$～１２２$_4$を有するＶＭ１１８$_1$のブロック図である。ゲストＯＳ１３０のゲストスケジューラ１３３は、１つまたは複数のｖＣＰＵ１２２$_1$～１２２$_N$上でコンピュータ命令を実行するプロセス（すなわちスレッド）に時間を割り振る。図４Ａに描写された領域４０２は、命令を実行する各ｖＣＰＵにより費やされた時間の一部としての、各ｖＣＰＵに関連付けられた要求の量を表す。示された例では、ｖＣＰＵはプロセスとスレッドを実行するために使用される要求とサイクルの量が変化し得るということを認識すべきであるが、複数のｖＣＰＵ間の負荷バランスはゲストスケジューラ１３３により行われ、各ｖＣＰＵに対する同様な要求量を生じる。

【００３８】

図４Ｂは、４つのｖＣＰＵ１２２$_1$～１２２$_4$を有するＶＭ１１８$_1$のブロック図である。４つのｖＣＰＵ１２２$_1$～１２２$_4$上では、ＶＭ１１８$_1$がハイパーバイザ１１１により推奨されたｖＣＰＵ１２２の目標数よりも多くのｖＣＰＵ１２２を利用しているとバルーンドライバ１３１が判定した場合、プロセスはゲストスケジューラ１３３によりスケジュールされてｖＣＰＵ１２２$_1$～１２２$_4$のうちの１つまたは複数へディスパッチされる。この例示では、バルーンドライバ１３１は、線４０８により描写するようにハイパーバイザ１１１と通信し（例えばバックドアコールを介し）、ＶＭ１１８$_1$が利用すべきｖＣＰＵ１２２の目標数を受信する。バルーンドライバ１３１は、ｖＣＰＵ１２２のこの目標数と、ＶＭ１１８$_1$により活性化され且つＶＭ１１８$_1$による使用のために利用可能であるｖＣＰＵ１２２の現在数とを比較する。バルーンドライバ１３１は、使用中のｖＣＰＵ１２２の現在数がｖＣＰＵ１２２の目標数を超える場合、１つまたは複数のバルーンスレッド４０４を立ち上げる。示された例では、バルーンドライバ１３１は、ＶＭのｖＣＰＵの目標数が３つのｖＣＰＵである（ｖＣＰＵの現在数（例えば、初めに割り振られた「４」）よりも少ない）と判定し、１つのバルーンスレッド４０４を立ち上げる。

【００３９】

バルーンドライバ１３１は、非活性化対象の特定のｖＣＰＵ（例えばｖＣＰＵ１２２$_4$）に対するバルーンスレッド４０４のプロセッサ親和性を規定し、これによりバルーンスレッドをこのｖＣＰＵにピン留めする。さらに、バルーンスレッド４０４は、ゲストＯＳ１３０上で実行する他のプロセスとスレッドに比較して高優先順位を有するカーネルスレッドであり得、したがって特定のｖＣＰＵ１２２$_4$上の他のすべてのプロセスをプリエンプトする。したがって、ゲストスケジューラ１３３は、規定優先順位を有するｖＣＰＵ１２２$_4$上にバルーンスレッド４０４をディスパッチする。いくつかの実施形態では、ゲストスケジューラ１３３は、ｖＣＰＵ４上に他のプロセスをディスパッチし得ないというこ

とを認識すべきである。バルーンスレッド４０４によるｖＣＰＵ１２２$_4$の占有（これによってバルーンスレッド４０４がｖＣＰＵ１２２$_4$のＣＰＵサイクルのすべてを利用する）は図４Ｂでは領域４０６により描写される。ゲストスケジューラ１３３は、ｖＣＰＵ１２２$_4$上で前に実行していたそれらのプロセスとスレッドを含むプロセスとスレッドを残りの利用可能なｖＣＰＵ１２２$_1$～１２２$_3$上にディスパッチし再スケジュールする。図４Ｂに描写された領域４１０は、もはや利用可能でないｖＣＰＵ１２２$_4$の結果としての各ｖＣＰＵ１２２$_1$、１２２$_2$、１２２$_3$に対する要求の増加を表す。したがって、本開示の実施形態は、ＶＭをシャットダウンするかまたはオフラインにする必要無く、要求に基づきｖＣＰＵを効果的に非活性化した。

【００４０】

１つまたは複数の実施形態が理解の明確のために少し詳しく本明細書では説明されたが、いくつかの変更と修正が本開示の精神から逸脱することなくなされ得るということを認識すべきである。例えば、いくつかの実施形態では、ゲストオペレーティングシステム１３０は、オンラインおよびオフラインの動的ＣＰＵを支援するように構成され得る。このような実施形態では、バルーンスレッドを立ち上げるのではなく、バルーンドライバ１３１は、ゲストＯＳ１３０内のＶＭのｖＣＰＵの数を調整するように構成され得る。一実施形態では、バルーンドライバ１３１は、目標ｖＣＰＵサイズをデバイスドライバファイルシステム（例えばｓｙｓｆｓ）へ書き込むこと（／ｓｙｓ／ｄｅｖｉｃｅｓ／ｓｙｓｔｅｍ／ｃｐｕ／ｃｕｐｉｄ／ｏｎｌｉｎｅ　ｓｙｓノードに対する変更など）により、リナックスゲストオペレーティングシステムを実行するＶＭのｖＣＰＵの数を調整し得る。

【００４１】

本明細書に記載の様々な実施形態は、コンピュータシステム内に格納されたデータに関わる様々なコンピュータ実施型動作を採用し得る。例えば、これらの動作は物理量の物理的操作を必要とし得る。通常、必ずしもではないが、これらの量は電気的または磁気的信号の形式を取り得、これらまたはその表現は格納され、転送され、合成され、比較され、またはそうでなければ操作されることができる。さらに、このような操作は、生成、付与、識別、判定、または比較などの用語でしばしば参照される。本開示の１つまたは複数の実施形態の一部をなす本明細書に記載の任意の動作は、有用な機械操作であり得る。加えて、本開示の１つまたは複数の実施形態はまた、これらの動作を実行するためのデバイスまたは装置に関する。上記装置は、特別に必要な目的のために特に構築され得る、またはコンピュータ内に格納されたコンピュータプログラムにより選択的に活性化または構成される汎用コンピュータデバイスであり得る。特に、様々な汎用マシンは本明細書の教示に従って書かれたコンピュータプログラムと共に使用され得るか、または必要な動作を実行するためにより専用化された装置を構築することがより便利な場合もある。

【００４２】

本明細書に記載の様々な実施形態は、携帯型デバイス、マイクロプロセッサシステム、マイクロプロセサベースまたはプログラム可能民生電子機器、ミニコンピュータ、メインフレームコンピュータなどを含む他のコンピュータシステム構成と共に実施され得る。

【００４３】

本開示の１つまたは複数の実施形態は、１つまたは複数のコンピュータプログラム、または１つまたは複数のコンピュータ読み取り可能媒体内に具現化された１つまたは複数のコンピュータプログラムモジュールとして実施され得る。用語「コンピュータ読み取り可能媒体」は、その後コンピュータシステムに入力されることができるデータを格納することができる任意のデータ記憶装置を指す。コンピュータ読み取り可能媒体は、コンピュータプログラムをコンピュータにより読まれることを可能にする方法で具現する任意の既存技術または今後開発される技術に基づき得る。コンピュータ読み取り可能媒体の例としては、ハードディスク駆動装置、ネットワーク付属記憶装置（NAS : network attached storage）、読み取り専用メモリ、ランダムアクセスメモリ（例えばフラッシュメモリ装置）、ＣＤ（コンパクトディスク）、ＣＤ－ＲＯＭ、ＣＤ－ＲまたはＣＤ－ＲＷ、ＤＶＤ（デジタルバーサタイルディスク）、磁気テープ、他の光学的および非光学的データ記憶装置

が挙げられる。コンピュータ読み取り可能媒体はまた、コンピュータ可読コードが分散された方法で格納され実行されるように、ネットワーク結合されたコンピュータシステム上に分散されることができる。

【００４４】

本開示の１つまたは複数の実施形態は理解の明確のために少し詳しく説明されたが、いくつかの変更と修正が本開示の範囲内でなされ得るということは明らかである。したがって、記載の実施形態は例示的であって限定的ではないと考えるべきであり、特許請求の範囲は本明細書に記載された詳細に限定されず、請求項の範囲と均等物のなかで修正され得る。特許請求の範囲において、要素および／または工程は特許請求の範囲に明示的に示されない限り動作の任意の特定の順序を意味しない。

【００４５】

様々な実施形態による仮想化システムは、ホストされた実施形態、ホストされない実施形態、またはこれら２つの差異をあいまいにする傾向がある実施形態として実施されてもよく、すべて想定される。さらに、様々な仮想化動作は、ハードウェアで完全にまたは部分的に実施され得る。例えば、ハードウェア実施形態は、非ディスクデータを安全にするために記憶装置アクセス要求の修正のための参照テーブルを採用し得る。

【００４６】

仮想化の程度にかかわらず、多くの変形、修正、追加、改良が可能である。したがって、仮想化ソフトウェアは、仮想化機能を実行するホスト、コンソール、またはゲストオペレーティングシステムの部品を含むことができる。単一インスタンスとして本明細書に記載の部品、動作、または構造に対し複数のインスタンスが設けられ得る。最後に、様々な部品、動作、データ記憶装置間の境界は多少任意的であり、特定の動作は特定の例示的構成の文脈の中で示されている。機能の他の割り振りが想定され、本開示の範囲に入り得る。一般的に、例示的構成において別個の部品として提示された構造と機能は組み合わせられた構造または部品として実現され得る。同様に、単一構成部品として提示された構造と機能は別々の部品として実現され得る。これらおよび他の変形形態、修正形態、追加形態、改良形態は添付特許請求の範囲に入り得る。

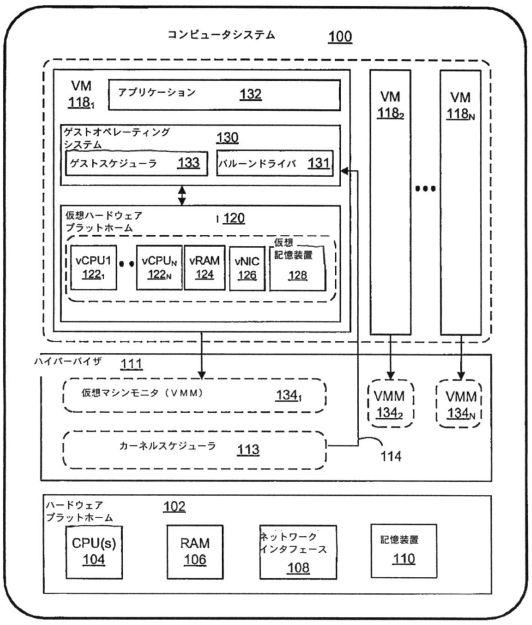
【符号の説明】

【００４７】

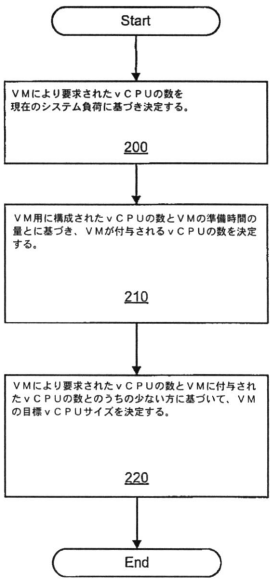
１００　コンピュータシステム
１０２　ハードウェアプラットホーム
１０４　ＣＰＵ
１０８　ネットワークインタフェース
１１０　記憶装置
１１１　ハイパーバイザ
１１３　カーネルスケジューラ
１１４　方向線
１１８　仮想マシン
１２０　仮想ハードウェアプラットホーム
１２２　仮想ＣＰＵ
１２４　仮想ランダムアクセスメモリ
１２６　仮想ネットワークインタフェースアダプタ
１２８　仮想記憶装置
１３０　ゲストオペレーティングシステム
１３１　バルーンドライバ
１３２　アプリケーション
１３３　ゲストスケジューラ
１３４　仮想マシンモニタ
２００、２１０、２２０、３００、３０５、３２０、３２５、３３０、３４０、３４５　工程

４０２　要求の量
４０４　バルーンスレッド
４０６　占有領域
４１０　要求の増加
４０８　線

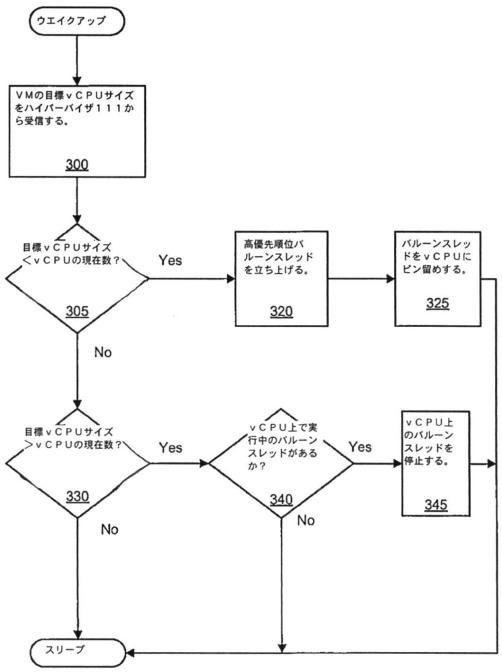
【図１】



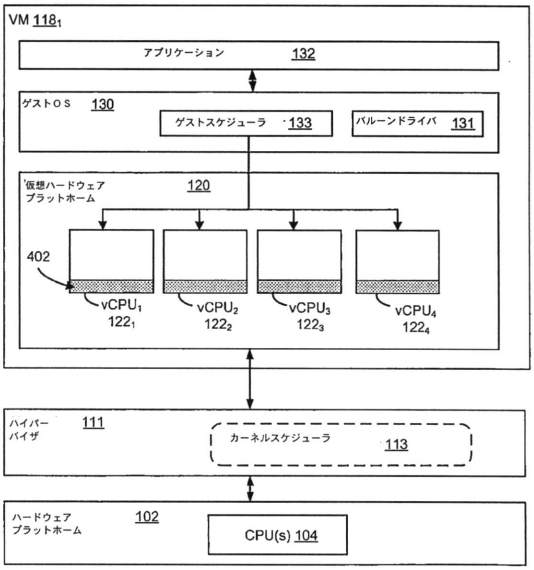
【図２】



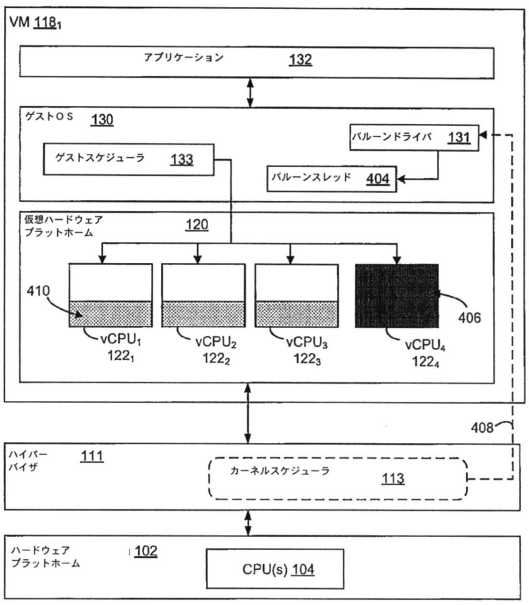
【図３】


ウエイクアップ
VMの目標ｖＣＰＵサイズをハイパーバイザ１１１から受信する。
300
目標ｖＣＰＵサイズ＜ｖＣＰＵの現在数？
305
Yes
No
高優先順位バルーンスレッドを立ち上げる。
320
バルーンスレッドをｖＣＰＵにピン留めする。
325
目標ｖＣＰＵサイズ＞ｖＣＰＵの現在数？
330
Yes
No
ｖＣＰＵ上で実行中のバルーンスレッドがあるか？
340
Yes
No
ｖＣＰＵ上のバルーンスレッドを停止する。
345
スリープ

【図４Ａ】


VM 118₁
アプリケーション 132
ゲストOS 130
ゲストスケジューラ 133
バルーンドライバ 131
仮想ハードウェアプラットホーム 120
402
vCPU₁ 122₁
vCPU₂ 122₂
vCPU₃ 122₃
vCPU₄ 122₄
ハイパーバイザ 111
カーネルスケジューラ 113
ハードウェアプラットホーム 102
CPU(s) 104

【図４Ｂ】


VM 118₁
アプリケーション 132
ゲストOS 130
バルーンドライバ 131
ゲストスケジューラ 133
バルーンスレッド 404
仮想ハードウェアプラットホーム 120
410
406
vCPU₁ 122₁
vCPU₂ 122₂
vCPU₃ 122₃
vCPU₄ 122₄
408
ハイパーバイザ 111
カーネルスケジューラ 113
ハードウェアプラットホーム 102
CPU(s) 104

フロントページの続き

(72)発明者　ジェン、ハオチャン
アメリカ合衆国　９４３０４　カリフォルニア州　パロ　アルト　ヒルビュー　アベニュー　３４０１

審査官　田中　幸雄

(56)参考文献　特表２０１１－５２１３８４（ＪＰ，Ａ）
特開２００７－３１０８８４（ＪＰ，Ａ）
特開２０１１－２２６２７（ＪＰ，Ａ）
特表２０１４－５３１６２５（ＪＰ，Ａ）
米国特許出願公開第２００９／０３００３１７（ＵＳ，Ａ１）
米国特許出願公開第２０１４／００８２６１２（ＵＳ，Ａ１）
米国特許第８１２７３０１（ＵＳ，Ｂ１）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　　９／５０
Ｇ０６Ｆ　　９／４５５