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

特許第4705051号
(P4705051)

(45)発行日　平成23年6月22日(2011.6.22)　(24)登録日　平成23年3月18日(2011.3.18)

(51)Int.Cl.　　F I
G06F 9/46 (2006.01)　　G06F 9/46 350
G06F 9/50 (2006.01)　　G06F 9/46 465Z

請求項の数 7　(全 11 頁)

(21)出願番号　特願2007-17659(P2007-17659)
(22)出願日　平成19年1月29日(2007.1.29)
(65)公開番号　特開2008-186136(P2008-186136A)
(43)公開日　平成20年8月14日(2008.8.14)
審査請求日　平成21年7月27日(2009.7.27)

(73)特許権者　000005108
株式会社日立製作所
東京都千代田区丸の内一丁目６番６号
(74)代理人　110000350
ポレール特許業務法人
(72)発明者　早川　典充
神奈川県秦野市堀山下１番地　株式会社日立製作所　エンタープライズサーバ事業部内
(72)発明者　松本　周平
神奈川県秦野市堀山下１番地　株式会社日立製作所　エンタープライズサーバ事業部内

審査官　北元　健太

最終頁に続く

(54)【発明の名称】計算機システム

(57)【特許請求の範囲】
【請求項１】
複数の物理ＣＰＵと、該複数の物理ＣＰＵのいずれかが割り当てられ、それぞれゲストＯＳ上でプログラムを実行する複数のＬＰＡＲと、該複数のＬＰＡＲを管理する管理プログラムとを有し、全ての該ＬＰＡＲから必要とされる共有処理を、該管理プログラムによって該複数の物理ＣＰＵに行なわせることができる計算機システムにおいて、
該ＬＰＡＲ上のゲストＯＳの状態を把握することができるゲストＯＳ監視部と、前記ＬＰＡＲに該物理ＣＰＵを割り当てるディスパッチ処理部と、を有し、
前記ディスパッチ処理部は、
各物理ＣＰＵが該共有処理に割り当てられた回数を管理するカウンタを有し、
該共有処理が発生した場合、前記カウンタを参照し、前記カウンタの回数が一番少ない物理ＣＰＵに対して該共有処理をディスパッチし、
該共有処理をディスパッチされた該物理ＣＰＵが処理したＬＰＡＲの状態を前記ゲストＯＳ監視部から取得し、該ＬＰＡＲがアイドル状態であった場合は該カウンタを更新せず、該ＬＰＡＲがアイドル状態でなかった場合は該カウンタを更新する
ことを特徴とする計算機システム。
【請求項２】
各物理ＣＰＵに対応して、該管理プログラムによる共有処理及び共有処理以外の処理の優先度を管理するディスパッチ優先度テーブルを有し、該共有処理を実行した後、関連する該ディスパッチ優先度テーブルを更新することを特徴とする請求項１の計算機システム。

【請求項３】
該ディスパッチ優先度テーブルの内容を更新するテーブル更新部を有し、該テーブル更新部は、共有処理の優先度をアイドル状態のＬＰＡＲよりも高い優先度に設定し、該共有処理の実行が終わった後、該優先度を低い状態に設定することにより優先度の高い共有処理に物理ＣＰＵを占有させず、該物理ＣＰＵが処理しているＬＰＡＲを沈み込ませないことを特徴とする請求項２の計算機システム。
【請求項４】
物理ＣＰＵの数はＬＰＡＲの数以下であることを特徴とする請求項１乃至３のいずれかの計算機システム。
【請求項５】
複数の物理ＣＰＵと、該複数の物理ＣＰＵのいずれかが割り当てられ、それぞれゲストＯＳ上でプログラムを実行する複数のＬＰＡＲと、該複数のＬＰＡＲを管理する管理プログラムと、該ＬＰＡＲ上のゲストＯＳの状態を把握することができるゲストＯＳ監視部及び前記ＬＰＡＲに該物理ＣＰＵを割り当てるディスパッチ処理部を含むスケジューリング処理部とを有し、全ての該ＬＰＡＲから必要とされる共有処理を、該管理プログラムによって該複数の物理ＣＰＵに行なわせることができる計算機システムにおいて、全ての該ＬＰＡＲから必要とされる共有処理のスケジューリングを行うスケジューリング方法であって、
該スケジューリング処理部は；該ディスパッチ処理部が有するカウンタを用いて、各物理ＣＰＵが該共有処理に割り当てられた回数を管理し、
該スケジューリング処理部は；該共有処理が発生した場合、該カウンタを参照して、該カウンタの回数が一番少ない物理ＣＰＵに対して該共有処理をディスパッチし、
該スケジューリング処理部は；該共有処理をディスパッチされた該物理ＣＰＵが処理したＬＰＡＲの状態を前記ゲストＯＳ監視部から取得し、該ＬＰＡＲがアイドル状態であった場合は該カウンタを更新せず、該ＬＰＡＲがアイドル状態でなかった場合は該カウンタを更新する
ことを特徴とするスケジューリング方法。
【請求項６】
該スケジューリング処理部は、
各物理ＣＰＵに対応して備えられたディスパッチ優先度テーブルを用いて、該管理プログラムによる共有処理及び共有処理以外の処理の優先度を管理し、該共有処理を実行した後、関連する該ディスパッチ優先度テーブルを更新することを特徴とする請求項５のスケジューリング方法。
【請求項７】
該スケジューリング処理部は、
各物理ＣＰＵが共有処理に割り当てられた回数を管理するカウンタを参照して、カウンタ値の小さいカウンタに対応する物理ＣＰＵから順に共有処理をディスパッチし、かつ、該カウンタの値を更新する時、ディスパッチされていた該物理ＣＰＵが処理しているＬＰＡＲの状態を前記ゲストＯＳ監視部から取得し、該ＬＰＡＲがアイドル状態である時は該カウンタを更新せず、該ＬＰＡＲがアイドル状態でない時に該カウンタを更新することを特徴とする請求項５又は６のスケジューリング方法。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、計算機システムに係り、特に計算機の物理リソースを分割された複数の論理的なパーティション（以下、ＬＰＡＲという）で利用するシステムにおける共有処理のスケジューリングに関するものである。
【背景技術】
【０００２】
高性能な計算機のハードウェアリソースを分割してＬＰＡＲと称する単位に分け、この

パーティションをそれぞれ１台のサーバとして運用する「サーバ分割」が実用化されている。サーバ分割は、ＬＰＡＲの動作環境によって、ハイパーバイザ型とホストＯＳ型に分類される。前者は、複数のＬＰＡＲをベアマシン上に存在する特殊な管理プログラム（ハイパーバイザ）上で動作させる方法である。後者は一般的なＯＳ上（ホストＯＳ）で管理プログラムを一アプリケーションとして稼動させ、その上でＬＰＡＲを動作させる方法である。何れの方式に関わらずサーバ分割には、ＶＭＭ(Virtual Machine Monitor)と呼ばれる制御機能と、その上で動作するＬＰＡＲが存在する。ＬＰＡＲは「仮想計算機」と呼ばれ、その上で動作するＯＳは「ゲストＯＳ」と呼ばれる。ゲストＯＳの動作はサーバ分割していないサーバ上での動作と機能的に同じものとなる。

【０００３】

ハイパーバイザは、主として「命令エミュレーション」、「メモリ管理」、「Ｉ／Ｏ処理」、「スケジューリング」の処理を司る。ＬＰＡＲではホストのリソースを共有して使用することもあり、ハイパーバイザには全てのＬＰＡＲから必要とされる共有処理が存在する。一般にこのような処理は、各ＬＰＡＲの処理よりも優先して処理しなければならない。この共有処理用にＣＰＵが割り当てられていないと、ＬＰＡＲ処理用に割り当てられているＣＰＵリソースを独占してしまうことがあり、これによってＬＰＡＲ側の処理が続行できなってしまうという問題が生じる。

【０００４】

上記のように、ある特定の処理のため、他の処理に物理ＣＰＵリソースが配分されない問題を解決する方法としては、処理の公平性を保証するスケジューリングアルゴリズムとしてタイムシェアリングが知られている。この技術を用いることにより共有処理を沈み込ませること無く、全ての処理に対して物理ＣＰＵを割り当てることが可能となる。しかし、このタイムシェリングは、リソースを奪取した物理ＣＰＵが、ビジー状態のＬＰＡＲを処理している物理ＣＰＵであるか、内部的にはアイドル状態であるＬＰＡＲを処理している物理ＣＰＵであるかを考慮することができない。

【０００５】

一方、優先度に応じてリソース配分を変動させ、特定のジョブの沈み込みを回避する手段として、「ジョブクラス」を定義する方法が知られている。この方法は、システムには予め定義された、複数の階層的な優先度があり、プロセス実行時に当該ジョブクラスを指定して実行させる方法である。これにより、優先度を考慮したＣＰＵリソースの配分が可能となる。

【０００６】

しかし、この方法は一つの計算機（ＯＳ）内の閉じた環境では有効であるが、ＬＰＡＲとハイパーバイザを跨ぐような環境への適応は困難である。それは、ハイパーバイザからは、全てのＬＰＡＲの優先度は静的には同一のものでありながら、ＬＰＡＲの内部的な状態はビジー状態からアイドル状態へまたはその逆へ動的に遷移するためである。

【０００７】

また、特許文献１（特開２００６－２４４４８３公報）には、ＣＰＵリソースを大量に消費するプロセスを検出し、リソース配分が他のＬＰＡＲよりも少なく定義してある「隔離ＬＰＡＲ」に割り当てることで、他のＬＰＡＲへの影響を防ぐ方法が開示されている。しかし、この方法ではＬＰＡＲの処理よりも優先度を上げて実行しなくてはならないハイパーバイザの共有処理には適用できない。

【０００８】

【特許文献１】特開２００６－２４４４８３号公報

【発明の開示】

【発明が解決しようとする課題】

【０００９】

図１～３を参照して、発明が解決しようとしている課題について説明する。

図１は、タイムシェアリング方式による共有処理のディスパッチについて示す。

３つの物理ＣＰＵを占有的に３つのＬＰＡＲに割り当てた状態を示している。物理ＣＰＵ

０は占有的にＬＰＡＲ０に割り当てられ、同様に物理ＣＰＵ１はＬＰＡＲ１に割り当てられ、物理ＣＰＵ２はＬＰＡＲ２に割り当てられている。ＬＰＡＲ０、ＬＰＡＲ２上で動作しているゲストＯＳは終始ビジー状態であり、ＬＰＡＲ１のそれはアイドル状態である。

【００１０】

ある時点１００でハイパーバイザの共有処理が発生した時、タイムシェアリング方式を用いて共有処理をディスパッチさせるとすると（１０１～１０３）、各々の物理ＣＰＵを用いて時分割で共有処理を処理するので特定のＬＰＡＲ処理の沈み込みは回避できる。

しかし、全てのＬＰＡＲの優先度が等しいとすると、内部的にはアイドル状態のＬＰＡＲが存在するにも拘わらず、１０４に示すように、ゲストＯＳがビジー状態で共有処理の合間にＬＰＡＲの処理を実行しているような物理ＣＰＵからもリソースを均等に奪取してしまい、１０５のようにアイドル状態のＬＰＡＲを処理している物理ＣＰＵが存在してしまう。

【００１１】

そこで、図２に示すように、タイムシェアリング方式を用いず、アイドル状態のＬＰＡＲを検知できる機能と、時分割ではなくディスパッチされた回数を基に、各物理ＣＰＵに分散させてディスパッチする方式がある。ある時点２００でハイパーバイザの共有処理が発生したとすると、アイドル状態のＬＰＡＲに共有処理を割り当てる。しかしながら、その後、２０１においてＬＰＡＲ側のＯＳやアプリケーションでプロセス実行要求が発生しても、共有処理の方がＬＰＡＲの処理よりも優先度が高いために共有処理が終了する２０２までＣＰＵが明け渡されない。そのため、２０３に示すように、ＬＰＡＲ上のＯＳが沈み込む状態が発生する。

【００１２】

ここで、図３に示すように、ゲストＯＳがアイドル状態のＬＰＡＲ１に割り当てられている物理ＣＰＵ１が、ある時点３００において発生した共有処理を複数回処理した後に、３０１においてアイドル状態からビジー状態へ遷移すれば、上記問題は解決され、他のビジー状態のＬＰＡＲを処理している物理ＣＰＵに共有処理を迅速に引き継げたと考えられる。

しかし、カウント方式にてスケジューリングしている場合、大域的には平準化されるわけだが、局所的には一方のＬＰＡＲを処理している物理ＣＰＵに共有処理が集中してディスパッチされてしまうことが起こる可能性がある。これでは、３０２で平準化されるまで、一方の物理ＣＰＵに集中的に割り当てられてしまい、その物理ＣＰＵが処理しているＬＰＡＲが沈み込んでしまう。これは、アイドル状態のＬＰＡＲに割り当てられている物理ＣＰＵで処理された方では、カウンタ値が積算されているため、ビジー状態に遷移した後では平準化のため処理が割り当てられず発生する問題である。

【００１３】

本発明の目的は、共有処理のためにＬＰＡＲの沈み込みが生じることを防止した計算機システムを提供することにある。より具体的には、本発明は、アイドル状態のＬＰＡＲに割り当てられているＣＰＵを有効利用して、ビジー状態のＬＰＡＲ処理を沈み込ませることを回避して、ＬＰＡＲ処理よりも優先度の高い共有処理のスケジューリングを行うことにある。

【課題を解決するための手段】

【００１４】

本発明に係る計算機システムは、好ましくは、複数の物理ＣＰＵと、複数の物理ＣＰＵのいずれかが割り当てられ、それぞれゲストＯＳ上でプログラムを実行する複数のＬＰＡＲと、複数のＬＰＡＲを管理する管理プログラムとを有する計算機システムにおいて、ＬＰＡＲ上のゲストＯＳの状態を把握することができるゲストＯＳ監視部と、ＬＰＡＲの処理よりも優先度が高い管理プログラムによる共有処理を、いずれかの物理ＣＰＵに割り当てるディスパッチ処理部と、を有し、ゲストＯＳ監視部による監視に基づいて、ディスパッチ処理部は、アイドル状態のＬＰＡＲを処理している物理ＣＰＵに対して共有処理を優先的にディスパッチする計算機システムとして構成される。

【００１５】
好ましい例では、各物理ＣＰＵに対応して、管理プログラムによる共有処理及び共有処理以外の処理の優先度を管理するディスパッチ優先度テーブルを有し、共有処理を実行した後、関連するディスパッチ優先度テーブルを更新する計算機システムとして構成される。
また、好ましくは、ディスパッチ優先度テーブルの内容を更新するテーブル更新部を有し、テーブル更新部は、共有処理の優先度をアイドル状態のＬＰＡＲよりも高い優先度に設定し、共有処理の実行が終わった後、優先度を低い状態に設定することにより優先度の高い共有処理に物理ＣＰＵを占有させず、物理ＣＰＵが処理しているＬＰＡＲを沈み込ませない計算機システムとして構成される。
また、好ましくは、前記ディスパッチ処理部は、各物理ＣＰＵが共有処理に割り当てられた回数を管理するカウンタを有し、カウンタの値を更新する時、ディスパッチされていた物理ＣＰＵが処理しているＬＰＡＲの状態を前記ゲストＯＳ監視部から取得し、ＬＰＡＲがアイドル状態である時は、カウンタを更新しない計算機システムとして構成される。
また、好ましくは、前記複数のＬＰＡＲの内少なくとも１つの所定のＬＰＡＲは主として共有処理を行うように割り当てられており、共有処理が発生した場合、前記ディスパッチ処理部は、所定のＬＰＡＲ又はアイドル状態のＬＰＡＲを処理している物理ＣＰＵに対して共有処理を優先的にディスパッチする計算機システムとして構成される。
【００１６】
また、本発明に係るスケジューリング方法は、好ましくは、ハイパーバイザによって複数の物理ＣＰＵを複数のＬＰＡＲに割り当てると共に、全てのＬＰＡＲに関係する共有処理のスケジューリングを行うスケジューリング方法であって、ゲストＯＳ監視部によってＬＰＡＲ上のゲストＯＳの状態を監視し、共有処理が発生した場合、ゲストＯＳ監視部による監視に基づいて、アイドル状態のＬＰＡＲを処理している物理ＣＰＵに対して優先的に共有処理をディスパッチすることを特徴とするスケジューリング方法として構成される。
【発明の効果】
【００１７】
本発明によれば、複数の物理ＣＰＵがＬＰＡＲに割り当てられている場合、アイドル状態のＬＰＡＲに割り当てられた物理ＣＰＵを有効に利用して共有処理を行うことができる。また、アイドル状態のＬＰＡＲが存在しない場合は、全てのＬＰＡＲに割り当てられた物理ＣＰＵを均等に使用して特定のＬＰＡＲを沈み込ませることなく、ＬＰＡＲ処理よりも優先度の高い共有処理を優先したスケジューリングを実現することができる。
【発明を実施するための最良の形態】
【００１８】
本発明の実施例を図面に基づいて詳細に説明する。
[実施例１]
図４は一実施例による計算機システムを示す。
この計算機システムは、４つの物理ＣＰＵ４１１～４１４（総称して４１ということがある）をサーバ分割して、同数の４つのＬＰＡＲ４３０～４３３（総称して４３ということがある）が割り当てられる。また、ＬＰＡＲの制御を司るハイパーバイザを一つの物理ＣＰＵが専用で処理するのではなく（隠れＣＰＵは存在しない）、複数の物理ＣＰＵが協調して処理する。また、共有処理をシステムで共有して使用されるデバイスのＩ／Ｏ制御処理とする。
【００１９】
計算機システムの物理的なリソース群（ハードウェアリソース）４１０は、複数の物理ＣＰＵ４１１～４１４、主記憶装置４１５、外部記憶装置４１６、およびネットワークインタフェースカード４１７等の構成要素を有する。
ハイパーバイザ４２０は、ハードウェアリソース４１０を用いて動作し、その上で動作する複数のＬＰＡＲ４３０～４３３の管理及び制御を司る。ハイパーバイザ４２０は、主に

、命令エミュレーション部４２１、メモリ管理部４２２、スケジューリング処理部４２３、Ｉ／Ｏ処理部４２４の各処理機能を有する。
ＬＰＡＲ４３はそれぞれゲストＯＳ４４０～４４３（総称して４４ということがある）と、各ゲストＯＳ上で動作するアプリケーションプログラム４５０～４５３（総称して４５ということがある）を有している。アプリケーションの動作はサーバ分割されていないハードウェア上の動作と同様である。

【００２０】

図５はスケジューリング処理部４２３の構成を示す。
スケジューリング処理部４２３は、ゲストＯＳ監視部５１、ディスパッチ処理部５２、ディスパッチ優先度テーブル更新部５３を有して構成される。ゲストＯＳ監視部５１は共有処理に限らず、各ＬＰＡＲ４３上で動作しているＯＳのリアルタイムな状態や、共有処理を実施したＬＰＡＲの状態を監視して把握する。

【００２１】

ディスパッチ処理部５２はスケジューリングアルゴリズムを用いて、ハイパーバイザ４２０および各ＬＰＡＲ４３にＣＰＵリソースを割り当てる処理を行う。ディスパッチ処理部５２は、ディスパッチカウンタテーブル５２１を備え、物理ＣＰＵ４１を各ＬＰＡＲ４３に偏り無く平均的にディスパッチさせるために、各物理ＣＰＵ４１が共有処理に割り当てられた回数を計数して、ディスパッチカウンタテーブル５２１に保持する。例えば、ある時点における、物理ＣＰＵ４１１～４１４に対応するディスパッチカウンタテーブル５２１の値が、「２」「３」「４」「１」の場合、次回の共有処理は、カウンタ値が最小「１」の物理ＣＰＵ４１４が割り当てられるが如きである。

【００２２】

ディスパッチ優先度テーブル更新部５３は、ディスパッチ優先度テーブル５４１～５４４（総称して５４ということがある）を有し、各物理ＣＰＵ０～３に対応して現時点でディスパッチすべき処理の優先度を管理する。
より具体的には、図６を参照するに、ディスパッチ優先度テーブル５４は、優先度として、ハイパーバイザの処理（例えば、共有処理）、ビジー状態のＬＰＡＲ処理、アイドル状態ＬＰＡＲ処理等の優先度を設定して管理する。ディスパッチ優先度テーブル５４が状態６１の時は、初期状態であり、共有処理を含んだハイパーバイザ処理の優先度が最も高いものとなっている。この状態は定常状態６１であり、ＬＰＡＲ処理よりも優先度の高い共有処理を最優先で実行する。

【００２３】

しかし、共有処理によるＬＰＡＲの沈み込みを避けるために、状態６２に示すように、ある物理ＣＰＵで共有処理を実行した後に、当該物理ＣＰＵに対する共有処理の優先度を意図的に低いものに変更する。この状態は排他状態である。排他状態６２へ遷移した物理ＣＰＵは、共有処理よりも優先度が高くなったＬＰＡＲ処理が存在すれば、一度ＣＰＵリソースを開放する。これにより、ＬＰＡＲ処理に物理ＣＰＵを割り当てることができ、ＬＰＡＲの沈み込みを回避することができる。ディスパッチ優先度テーブル５４ｎを更新した後に、ＬＰＡＲ処理を再び実行した物理ＣＰＵは、ディスパッチ優先度テーブル５４ｎを元に戻し、定常状態へ復帰する。

【００２４】

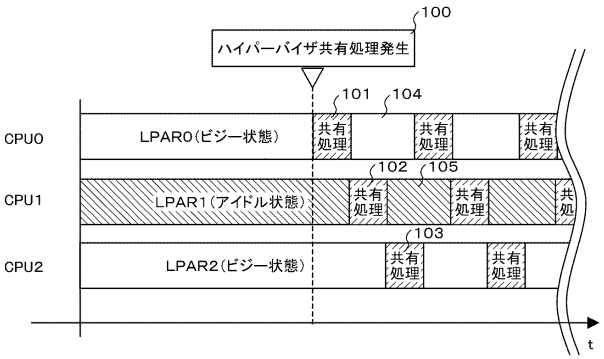
図７は、ディスパッチ優先度テーブル５４の更新処理のシーケンスを示す。
この図では、便宜上３つの物理ＣＰＵ０～２から構成される計算機の動作例を示しているが、４つ以上の物理ＣＰＵを有する計算機の場合の動作例も実質的に同様である。
初期状態では全てのＣＰＵ（ＣＰＵ０～ＣＰＵ２）は定常状態（７０１～７０３）である。ハイパーバイザ側より共有処理が発生した場合（７０４）、何れのＣＰＵにおいても最も優先度高い処理であるため（即ち定常状態）、直ちにＣＰＵが割り当てられる（７０５）。タイムスライス分の実行が終了すると、共有処理を実行していたＣＰＵ０は排他状態に切り替わる（７０６）。排他状態になったＣＰＵ０は共有処理よりもＬＰＡＲ０の処理が優先的に割り当てられる（７０７）。

【００２５】
一方、定常状態にあるＣＰＵ１にはＬＰＡＲ１の処理よりも共有処理が優先的に割り当てられ、共有処理は引継いで処理される（７０８）。タイムスライス分の実行が終了すると、排他状態にあったＣＰＵ０は定常状態へ復帰し（７０９）、共有処理を実行していたＣＰＵ１は排他状態に切り替わる（７１０）。このように、共有処理はＬＰＡＲ処理を実行している複数の物理ＣＰＵ間を順次渡りながら実行されていく。
【００２６】
図８はスケジューリング処理部４２３の処理シーケンスを示す。
共有処理が発生した場合（８０１）、スケジューラは、ＬＰＡＲに割り当てられていない空き物理ＣＰＵがあるか、をチェックする（８０２）。チェックの結果、空き物理ＣＰＵがあればそのＣＰＵをディスパッチする。一方、空き物理ＣＰＵが無ければ、ゲストＯＳ監視部５２に問い合わせを実施し（８０３）、アイドル状態のＬＰＡＲに割り当てられている物理ＣＰＵがあるか、をチェックする（８０４）。チェックの結果、アイドル状態のＣＰＵがあればそれをディスパッチする。一方、アイドル状態のＣＰＵが存在しない場合は、ディスパッチカウンタテーブル５２１を参照して（８０５）、ディスパッチ回数の一番少ない物理ＣＰＵに共有処理を割り当てる（８０６）。
【００２７】
物理ＣＰＵを最終的に割り当てる前に、ディスパッチ優先度テーブル５４を参照し（８０７）、現時点の優先度から判断して共有処理を割り当てることができるか、を確認する（８０８）。当該物理ＣＰＵが排他状態であり、割り当てが不可な場合は、その物理ＣＰＵを選択の対象から除外して（８０９）、再度ディスパッチカウンタテーブル５２１を参照する。一方、割り当てが可能な場合は、その物理ＣＰＵで共有処理を実行する（８１０）。
【００２８】
共有処理は、決められた期間（タイムスライス分）実施される（８１１）。共有処理の実行がタイムスライス分の期間を経過した後、共有処理を実行した物理ＣＰＵに対して、ディスパッチ優先度テーブル５４ｎを更新する（８１２）。共有処理を実行した物理ＣＰＵが割り当てられていたＬＰＡＲの状態を調査するため、ゲストＯＳ監視部５１に、アイドル状態のＬＰＡＲの物理ＣＰＵを使用していたか、を問い合わせる（８１３）。問合せの結果、アイドル状態のＬＰＡＲの物理ＣＰＵを使用していた場合（８１３Ｙ）には、ディスパッチカウンタテーブル５２１を更新しない。一方、アイドル状態でないＬＰＡＲの物理ＣＰＵを使用していた場合は（８１３Ｎ）、ディスパッチカウンタテーブル５２１を更新する（８１４）。
【００２９】
その後、共有処理が終了したかを判断し（８１５）、終了した場合は本スケジューリング処理を終了する（８１６）。一方、共有処理が終了していない場合、次のタイムスライスにおけるＣＰＵディスパッチとして、当該物理ＣＰＵに割り当てられているＬＰＡＲが継続してアイドル状態であるかをチェックし（８１７）、アイドル状態の場合は同じ物理ＣＰＵを使用する。一方、当該ＬＰＡＲがアイドル状態からビジー状態へ遷移した場合、ディスパッチされていた物理ＣＰＵを開放する（８１８）。そして、ある一定の期間ごとにディスパッチカウンタテーブル５２１をリフレッシュし（８１９，８２０）、リフレッシュのための一定時間が経過していなければ、ステップ８３０に戻って、上記した処理を繰り返す。
【００３０】
図９は、共有処理のスケジューリングの例を示す。
ＬＰＡＲ１上のゲストＯＳ１は初期段階でアイドル状態である。ある時点９０１において、ハイパーバイザの共有処理が発生すると、共有処理は、ＬＰＡＲ１を処理している物理ＣＰＵ１にディスパッチされる。ゲストＯＳ１（物理ＣＰＵ１）がアイドルである期間は物理ＣＰＵ１を有効利用し、共有処理を実行する。一方、ゲストＯＳ０、２、３は当該期間において共有処理に何ら阻害されることなく、ＬＰＡＲの処理に専念できる。

【００３１】
その後、時点９０２においてゲストＯＳ１のプロセスが発生すると、次のスケジューリングのタイミングにおいて、物理ＣＰＵ１を開放し、本来の処理であるＬＰＡＲ１の処理に移行する。以後、何れかの物理ＣＰＵを独占することなく、各物理ＣＰＵを均等に割り当てることで共有処理を実行して終えることができる（９０３）。
[実施例２]
図１０は、他の実施例による計算機システムの構成を示す。
この例は、各ゲストＯＳのＩ／Ｏ処理を各ＬＰＡＲに実行させるのではなく、全てのＬＰＡＲのＩ／Ｏ処理を一括して引き受けるドライバＯＳをＬＰＡＲ１００１上に設けたものである。これにより容易にＩ／Ｏデバイスを複数のＬＰＡＲ間で共有することができ、中間バッファ領域等を設けることで効率よくＩ／Ｏ処理を実行できる。
このようなドライバＯＳでは複数のＩ／Ｏ処理１００２を実行するため、一般的には命令処理よりも長期間ＣＰＵを割り当てる必要がある。しかしこのような場合でも、本実施例によればドライバＯＳ（特定のＬＰＡＲ上で動作）を高い優先度を維持したまま、かつＬＰＡＲ４３０～４３３を処理しているＣＰＵの内、特定のＣＰＵのリソースを食い潰すことなく実現できる。
【図面の簡単な説明】
【００３２】
【図１】タイムシェアリング方式による共有処理のディスパッチについて説明するための図、
【図２】優先度の高い共有処理によるＬＰＡＲの沈み込みについて説明するための図、
【図３】割り当て平準化に伴うＬＰＡＲの沈み込みについて説明するための図、
【図４】本発明の一実施例による計算機システムの構成を示す図、
【図５】計算機システムにおけるスケジューリング処理部４２３の構成を示す図、
【図６】ディスパッチ優先度テーブル５４の構成を示す図、
【図７】ディスパッチ優先度テーブル更新部５３の処理シーケンスを示す図、
【図８】計算機システムにおけるスケジューリング処理を示すフローチャート、
【図９】一実施例によるスケジューリングの例を示す図、
【図１０】他の実施例による計算機システムの構成を示す図。
【符号の説明】
【００３３】
４１０：ハードウェアリソース　４１１～４１４：ＣＰＵ　４１５：主記憶装置　４１６：外部記憶装置　４１７：ネットワークインタフェースカード　４２０：ハイパーバイザ　４２１：命令エミュレーション部　４２２：メモリ管理部　４２３：スケジューリング処理部　４２４：Ｉ／Ｏ処理部　４３０～４３３　ＬＰＡＲ　４４０～４４３：ゲストＯＳ　４５０～４５３：アプリケーションプログラム　５１：ゲストＯＳ監視部　５２：ディスパッチ処理部　５２１：ディスパッチカウンタテーブル
５３：ディスパッチ優先度テーブル更新部　５４１～５４４：ディスパッチ優先度テーブル

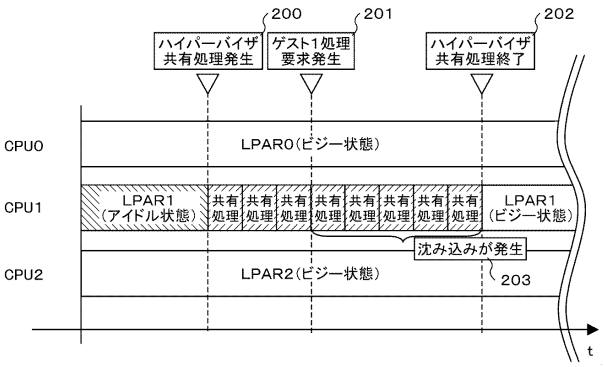
【図１】

図１



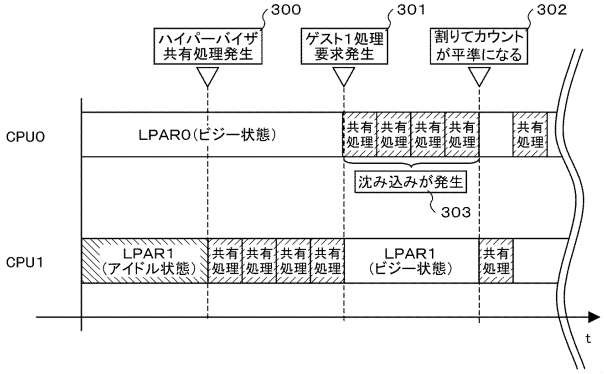
【図２】

図２



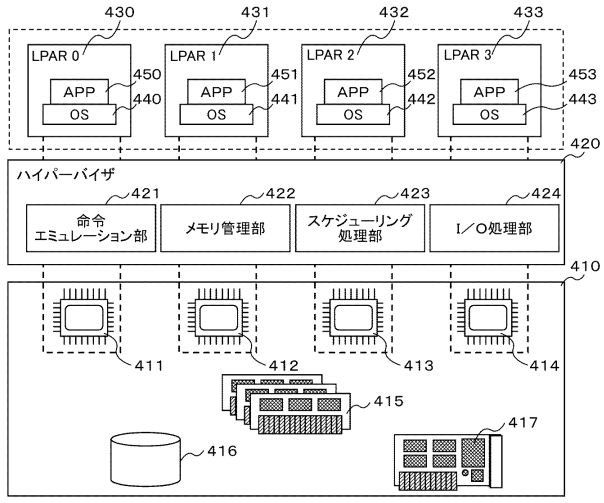
【図３】

図３



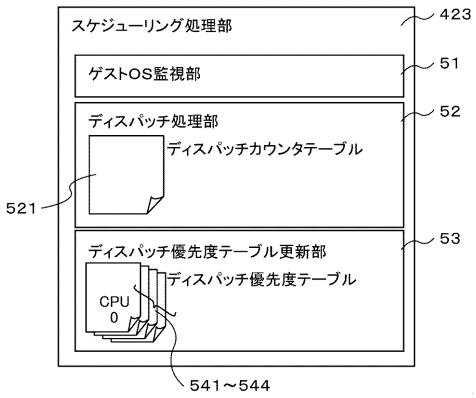
【図４】

図４



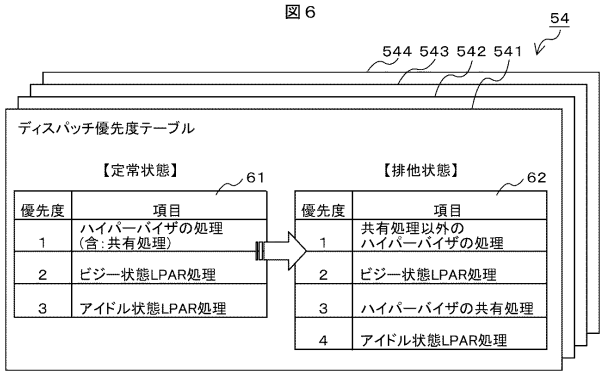
【図５】

図５



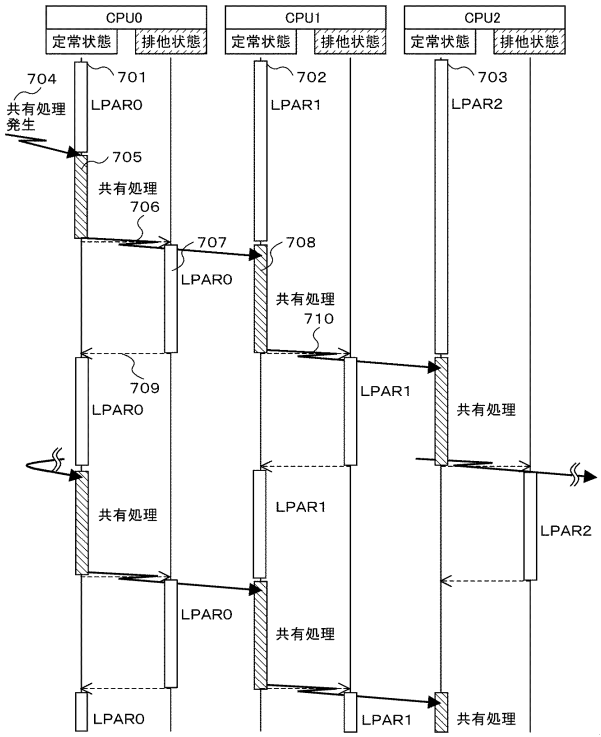
【図６】

図６



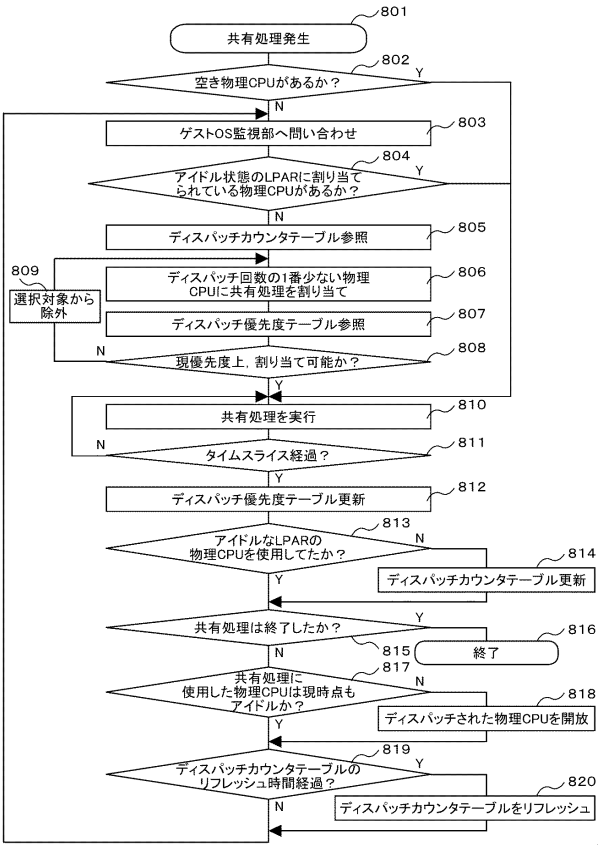
【図７】

図７



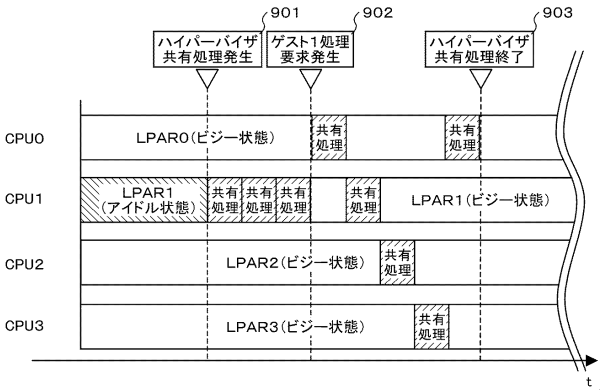
【図８】

図８



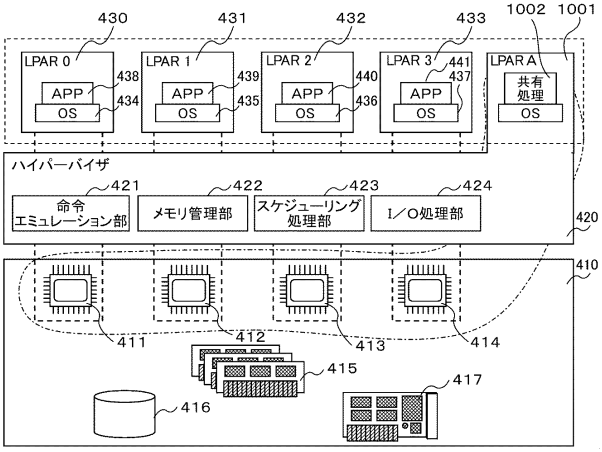
【図９】

図９



【図１０】

図１０



フロントページの続き

(56)参考文献 国際公開第２００５／０３６３６７（WO，Ａ２）
特表２００７－５０８６２３（ＪＰ，Ａ）
特開２００４－２５２５２６（ＪＰ，Ａ）
特開平８－７７０２５（ＪＰ，Ａ）
特開平１１－２５９３１６（ＪＰ，Ａ）
特開２００２－３１８６９９（ＪＰ，Ａ）
特開昭６２－２２１０４１（ＪＰ，Ａ）
特開２００６－１２７５２４（ＪＰ，Ａ）
特開２００５－７１１６１（ＪＰ，Ａ）
特開平４－１４８４６０（ＪＰ，Ａ）
Tivoli パフォーマンス・レポーター OS/390 CICSパフォーマンス・フィーチャーガイドおよびリファレンス リリース5，日本アイ・ビー・エム株式会社，２０００年１１月，第1刷，pp.153－158

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