(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第5010164号
(P5010164)
(45)発行日　平成24年8月29日(2012.8.29)　(24)登録日　平成24年6月8日(2012.6.8)

(51)Int.Cl.　F I
GO6F 9/46 (2006.01)　GO6F 9/46 350

請求項の数 8 (全 35 頁)

(21)出願番号　特願2006-97594(P2006-97594)
(22)出願日　平成18年3月31日(2006.3.31)
(65)公開番号　特開2007-272576(P2007-272576A)
(43)公開日　平成19年10月18日(2007.10.18)
審査請求日　平成20年10月6日(2008.10.6)

(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　100114236
弁理士　藤井　正弘
(74)代理人　100075513
弁理士　後藤　政喜
(74)代理人　100084537
弁理士　松田　嘉夫
(72)発明者　服部　直也
東京都国分寺市東恋ヶ窪一丁目２８０番地
株式会社日立製作所　中央研究所内
(72)発明者　森本　俊臣
東京都国分寺市東恋ヶ窪一丁目２８０番地
株式会社日立製作所　中央研究所内
最終頁に続く

(54)【発明の名称】サーバ装置及び仮想計算機の制御プログラム

(57)【特許請求の範囲】
【請求項１】
一つ以上のＣＰＵと、データ及びプログラムを格納するメモリと、外部とでデータを送受信するインターフェースと、を備えたサーバ装置において、
前記ＣＰＵが前記メモリに格納されたプログラムを実行することによって、オペレーティングシステムが稼働する一以上の仮想計算機を稼働させる仮想計算機モニタが実行されており、
前記オペレーティングシステム又は前記オペレーティングシステム上で稼働するアプリケーションは、所定の処理の実行を要求する操作コードを出力し、
前記ＣＰＵは、ハードウェア操作許可モードとハードウェア操作禁止モードとの二つの動作モードを備え、
前記仮想計算機モニタは、
前記オペレーティングシステム又は前記オペレーティングシステム上で稼働するアプリケーションから前記サーバ装置のハードウェアに対する操作を要求するハードウェア操作コードを含む前記操作コードが出力されたときに、前記ハードウェア操作コードに基づいて前記サーバ装置のハードウェアに対する操作をエミュレーションするエミュレーションモジュールと、
前記ハードウェア操作コードを検出することによって前記ＣＰＵが発生させる例外イベントを契機として、前記ＣＰＵの前記動作モードを前記ハードウェア操作許可モードに切り替え、前記エミュレーションモジュールを呼び出す例外イベントハンドラモジュールと

、
前記ＣＰＵの前記動作モードを前記ハードウェア操作許可モードに切り替え、前記エミュレーションモジュールを呼び出す変換コードを管理するコード管理モジュールと、
前記操作コードに対応する前記変換コードを生成し、前記操作コード及び前記生成された変換コードを対応づけて前記メモリ上に格納する生成モジュールと、
所定の判断条件との比較の結果、前記操作コードに含まれる前記ハードウェア操作コードの実行頻度が高頻度であるか否かを判断する頻度判断モジュールと、
前記実行頻度の判断の結果に基づいて、前記ハードウェア操作コードを含む前記操作コードに対して、前記ＣＰＵが前記例外イベントを発生させ、前記操作コードの実行の強制中断処理を終了した後に実行される前記例外イベントハンドラモジュールによって前記エミュレーションモジュールを呼び出すＶＴ方式、又は、前記ハードウェア操作コードを含む前記操作コードに対応する前記変換コードを実行することによって前記エミュレーションモジュールを呼び出すコード変換方式のいずれの動作方式を適用するかを決定する切り替えモジュールと、を備え、
前記変換コードは、前記ＣＰＵの前記動作モードを前記ハードウェア操作禁止モードから前記ハードウェア操作許可モードに切り替え、前記ハードウェア操作コードをエミュレーションさせる昇格コードと、前記昇格コードを含むシャドウコードとから構成され、
前記コード管理モジュールは、前記ハードウェア操作コード及び前記変換コードを対応づけた変換コード表を保持し、
前記切り替えモジュールは、前記ハードウェア操作コードを含む前記操作コードと、当該操作コードに対して適用される前記動作方式とを対応づけた動作方式表を保持し、
前記仮想計算機モニタは、
前記動作方式表を参照して、前記ハードウェア操作コードを含む前記操作コードに適用される前記動作方式を判断し、
前記ハードウェア操作コードを含む前記操作コードの前記動作方式がＶＴ方式である場合、前記例外イベントを発生させ、前記ＣＰＵの動作モードをハードウェア操作禁止モードから前記ハードウェア操作許可モードに変更した後に、前記ハードウェア操作コードのエミュレーションを実行し、
前記実行された操作コードに含まれる前記ハードウェア操作コードの実行頻度が高頻度であるか否かを判断し、
前記実行された操作コードに含まれるハードウェア操作コードの実行頻度が高頻度である場合、前記動作方式表のうち、前記実行され他ハードウェア操作コードを含む前記操作コードに対応する前記動作方式を、前記ＶＴ方式から前記コード変換方式に変更し、
前記変換コード表を参照して、前記実行されたハードウェア操作コードを含む前記操作コードに対応する前記変更コードを検索し、
前記変換コード表に、前記実行されたハードウェア操作コードを含む前記操作コードに対応する前記変換コードが存在しないと判断された場合、前記実行されたハードウェア操作コードを含む前記操作コードに対応する前記シャドウコードと、前記実行されたハードウェア操作コードに対応する前記昇格コードとを含む前記変換コードを生成し、
前記生成されたシャドウコード及び前記生成された昇格コードとの対応関係を前記コード変換表に追加することを特徴とするサーバ装置。
【請求項２】
前記メモリは、前記ハードウェア操作コードを含む前記操作コードと、当該操作コードが実行された時を特定可能な時刻情報とを対応づけた操作履歴表を格納し、
前記仮想計算機モニタは、
前記実行された操作コードに含まれる前記ハードウェア操作コードの実行頻度が高頻度であるか否かを判断する場合に、当該操作コードが実行された時を特定可能な第１の時刻情報を取得し、前記操作履歴表を参照して、当該操作コードが前回実行された時を特定可能な第２の時刻情報を取得し、
前記第１の時刻情報及び前記第２の時刻情報に基づいて、前記操作コードに含まれる前

記ハードウェア操作コードによって実行された前記ハードウェアの操作の間隔を算出し、
前記算出した間隔が所定の閾値よりも小さいときに、前記実行された操作コードに含まれる前記ハードウェア操作コードの実行頻度が高いと判断することを特徴とする請求項１に記載のサーバ装置。
【請求項３】
前記切り替えモジュールは、前記オペレーティングシステム又は前記オペレーティングシステム上で稼働するアプリケーションによって、前記コード変換方式が適用される前記操作コードが変更されたことを前記仮想計算機モニタが検出したときに、前記変更された操作コードに対応する前記シャドウコードを無効化し、
前記仮想計算機モニタは、前記変更された操作コードを実行するときに、前記例外イベントを発生させて前記変更された操作コードに含まれる前記ハードウェア操作コードのエミュレーションを実行するように設定することを特徴とする請求項２に記載のサーバ装置。
【請求項４】
前記切り替えモジュールは、
前記シャドウコード及び前記昇格コードを保持するために必要な前記メモリの容量が不足したとき、前記変換コード表に格納される前記変換コードのうち、実行されていない期間が最も長い前記変換コードを選択し、
前記選択された変換コードの情報を前記変換コード表から削除し、
前記仮想計算機モニタは、前記削除された変換コードと対応する前記操作コードを実行する場合に、当該削除された変換コードを実行する代わりに、前記例外イベントを発生させて、前記削除された変換コードと対応する前記操作コードに含まれる前記ハードウェア操作コードの前記エミュレーションを実行するように設定することを特徴とする請求項２に記載のサーバ装置。
【請求項５】
一つ以上のＣＰＵと、データ及びプログラムを格納するメモリと、外部とでデータを送受信するインターフェースと、を備えたサーバ装置に、オペレーティングシステムが稼働する一以上の仮想計算機を稼働させる仮想計算機の制御プログラムであって、
前記オペレーティングシステム又は前記オペレーティングシステム上で稼働するアプリケーションは、所定の処理の実行を要求する操作コードを保持し、
前記ＣＰＵは、ハードウェア操作許可モードとハードウェア操作禁止モードとの二つの動作モードを備え、
前記メモリは、
前記サーバ装置のハードウェアに対する操作を要求するハードウェア操作コードを含む前記操作コードと、前記ＣＰＵの前記動作モードを前記ハードウェア操作許可モードに切り替え、前記ハードウェア操作コードをエミュレーションさせる変換コードとを対応づけた変換コード表と、
前記ハードウェア操作コードを含む前記操作コードと、前記ハードウェア操作コードのエミュレーション方式とを対応づけた動作方式表と、を格納し、
前記変換コードは、前記ＣＰＵの前記動作モードを前記ハードウェア操作禁止モードから前記ハードウェア操作許可モードに切り替え、前記ハードウェア操作コードをエミュレーションさせる昇格コードと、前記昇格コードを含むシャドウコードとから構成され、
前記エミュレーション方式は、前記ハードウェア操作コードを含む前記操作コードに対して、前記ＣＰＵが前記例外イベントを発生させ、前記操作コードの実行の強制中断処理を終了した後に、当該ハードウェア操作コードをエミュレーションするＶＴ方式、又は、前記ハードウェア操作コードを含む前記操作コードに対応する前記変換コードを実行することによって、前記ハードウェア操作コードをエミュレーションするコード変換方式とを含み、
前記仮想計算機の制御プログラムは、
前記ハードウェア操作コードを含む操作コードを検出した場合に、前記動作方式表を参

照し、前記ハードウェア操作コードを含む前記操作コードに対応する前記エミュレーション方式を特定する第１の手順と、
前記特定されたエミュレーション方式が前記ＶＴ方式である場合、例外イベントを発生させて、前記ＣＰＵの前記動作モードを前記ハードウェア操作許可モードに切り替えて、前記出力された操作コードに含まれる前記ハードウェア操作コードをエミュレーションする第２の手順と、
前記特定されたエミュレーション方式が前記コード変換方式である場合、前記検出された操作コードに対応する前記シャドウコードに含まれる前記昇格コードに基づいて、前記出力された操作コードに含まれる前記ハードウェア操作コードをエミュレーションする第３の手順と、
前記操作コードに含まれる前記ハードウェア操作コードのエミュレーションが完了した後、所定の判断条件に基づいて、当該ハードウェア操作コードの実行頻度が高頻度であるか否かを判断する第４の手順と、
前記判断された実行頻度に基づいて、前記第２の手順、又は、前記第３の手順のいずれを実行して前記操作コードに含まれる前記ハードウェア操作コードをエミュレーションさせるかを決定し、前記決定に基づいて前記動作方式表を更新する第５の手順と、
前記変換コード表を参照して、前記実行されたハードウェア操作コードを含む前記操作コードに対応する前記変更コードを検索する第６の手順と、
前記変換コード表に、前記実行されたハードウェア操作コードを含む前記操作コードに対応する前記変換コードが存在しないと判断された場合、前記実行されたハードウェア操作コードを含む前記操作コードに対応する前記シャドウコードと、前記実行されたハードウェア操作コードに対応する前記昇格コードとを含む前記変換コードを生成する第７の手順と、
前記生成されたシャドウコード及び前記生成された昇格コードとの対応関係を前記コード変換表に追加する第８の手順と、を前記サーバ装置に実行させることを特徴とする仮想計算機の制御プログラム。
【請求項６】
前記メモリは、前記ハードウェア操作コードを含む前記操作コードと、当該操作コードが実行された時を特定可能な時刻情報とを対応づけた操作履歴表を格納し、
前記第４の手順は、
前記ハードウェア操作コードを含む前記操作コードが実行された時を特定可能な第１の時刻情報を取得し、前記操作履歴表を参照して、前記ハードウェア操作コードを含む前記操作コードが前回実行された時を特定可能な第２の時刻情報を取得し、
前記第１の時刻情報及び前記第２の時刻情報に基づいて、前記操作コードに含まれる前記ハードウェア操作コードによって実行された前記ハードウェアの操作の間隔を算出する手順と、
前記算出した間隔が所定の閾値よりも小さいときに、前記操作コードに含まれる前記ハードウェア操作コードの実行頻度が高いと判断する手順と、を含むことを特徴とする請求項５に記載の仮想計算機の制御プログラム。
【請求項７】
管理者によって入力された前記所定の判断条件を受け付ける手順と、
前記入力された所定の判断条件を記憶装置に保存する手順と、を含むことを特徴とする請求項６に記載の仮想計算機の制御プログラム。
【請求項８】
前記第２の手順は、ＶＭｅｘｉｔイベントの発生によって前記ＣＰＵをＶＭＸ　ｒｏｏｔモードに設定することによって、前記ハードウェア操作許可モードに遷移する手順を含み、
前記第３の手順は、前記昇格コードに含まれる特権レベル減少分岐命令を実行して前記ＣＰＵの特権レベルを０に設定することによって、前記ハードウェア操作許可モードに遷移する手順を含むことを特徴とする請求項６に記載の仮想計算機の制御プログラム。

【発明の詳細な説明】
【技術分野】
【０００１】
本発明はサーバ装置において実現される仮想計算機システムに関して、ハードウェアエミュレーションの効率化に関する。
【背景技術】
【０００２】
ＩＴシステムに含まれるサーバ台数が増加するにつれて、運用に関する複雑さが増加するため、その運用コストが問題化する。運用コストを低減するために、複数サーバを１台のサーバに統合するサーバ統合が知られている。サーバ統合を実現する技術として、一つのコンピュータを任意の割合で論理的に分割する仮想計算機技術が知られている。
【０００３】
仮想計算機技術は、例えば、ハイパバイザなどのファームウェア又はミドルウェアが、物理計算機を複数の論理区画（ＬＰＡＲ：Logical PARtition）に分割し、分割された各ＬＰＡＲに計算機資源（ＣＰＵ、主記憶及びＩ／Ｏ）を割り当てる。この各ＬＰＡＲ上でオペレーティングシステム（ＯＳ）が動作する。また、一つのサーバ上で一つのホストＯＳ（物理計算機を直接使用するＯＳ）を実行し、このホストＯＳ上で稼動するハイパバイザが同様に物理計算機をＬＰＡＲに分割する。この各ＬＰＡＲ上でゲストＯＳ（ＬＰＡＲ上で稼動するＯＳ）が稼動してもよい。
【０００４】
このように、仮想計算機技術は、従来複数のサーバで稼動していたＯＳ及びＯＳ上で稼動するソフトウェアを、１台のサーバで稼動させることを可能にする。これによって、サーバ統合が実現される。
【０００５】
この仮想計算機技術は、従来は汎用機（MainFrame）等の大型計算機で用いられてきた技術であるが、近年のマイクロプロセッサの性能向上によって、ローエンドサーバやパーソナルコンピュータにも適用可能である。
【０００６】
この仮想計算機技術を適用することによって、物理計算機は、ゲスト（ゲストＯＳ及びゲストＯＳ上で稼動するアプリーケーションソフトウェアの総称）を稼動させる複数の仮想計算機（Virtual Machine）と、仮想計算機の制御を行う仮想計算機モニタ（Virtual Machine Monitor、以下ＶＭＭとする）とを有する仮想計算機システムが構成される（特許文献１参照）。
【０００７】
なお、ＯＳは、本来、サーバのハードウェアを全て独占して使用することを前提に作られている。しかし、仮想計算機システムでは、複数のゲストＯＳが稼動する。しかしながら、ゲストＯＳはサーバのハードウェアを独占することはできない。そこで、ＶＭＭが、ゲストＯＳによるハードウェア操作に対応して、ゲストＯＳがサーバのハードウェアを独占している状態と同じ振る舞いをエミュレーションする。
【０００８】
ＶＭＭは、このエミュレーションの処理のために時間とメモリを使用する。そのため、ゲストＯＳによってＶＭＭによるエミュレーションの回数が増加すると、処理時間とメモリが増加することによって、ゲストの性能が低下し、ゲストの利用可能なメモリ量が減少する問題がある。この問題を軽減するために、エミュレーションの高速化（高性能化）技術や省メモリ化技術が考案されている。
【０００９】
エミュレーションの高速化技術の一つにコード変換方式がある。コード変換方式は、ＶＭＭが、ゲストＯＳによる操作コード（ＯＳコードとも呼ぶ）に対応する変換コードをメモリ上に作成する。この変換コードの作成処理をコード変換と呼ぶ。変換コードを作成するときに、対応するＯＳコードに含まれるハードウェア操作は、エミュレーションコード

に置き換えられる。このコード変換方式では、ＯＳコードの代わりに変換コードを実行することで、必要最小限の処理のみでハードウェア操作のエミュレーションを可能とする。コード変換方式は、実行に必要な全てのＯＳコードに対してコード変換を実施する。コード変換は処理時間を要するため速度低下の原因になり得る。そこで本方式では、作成した変換コードをメモリ上に保持し続け、２度目以降はメモリ上の変換コードを再利用し、速度低下を防止する。ただし、変換コードを保持するメモリが不足した場合は、全ての変換コードを保持できないので実行頻度の低い変換コードを破棄する。

【００１０】

エミュレーションの省メモリ化技術であるＶＴ方式は、Ｉｎｔｅｌ社製ＣＰＵに搭載されたＶＴ（Virtualization Technology）機能を利用する。ＶＴ機能を用いるとＣＰＵの動作モードとして、ＨＷ操作コードを一切実行できないＨＷ操作完全禁止モードが選択可能になる。ＣＰＵがＨＷ操作完全禁止モードのときにＨＷ操作コードの実行を試みると、ＣＰＵが例外イベントを発生してコード実行を強制中断する。このとき、ＣＰＵはＨＷ操作が可能なＨＷ操作許可モードに遷移した上で、事前に登録されたコードを実行する。逆に、ＨＷ操作許可モードで例外イベントからの再開処理を実行すると、ＣＰＵはＨＷ操作完全禁止モードに遷移し、指定アドレスから実行を再開する（非特許文献１、非特許文献２参照）。

【００１１】

この方式では、エミュレーション処理を実施するエミュレーションモジュールをＣＰＵに事前登録し、ゲストをＨＷ操作禁止モードで実行させることで、ＨＷ操作をエミュレーションする。本方式は、ＶＭＭのメモリにエミュレーションモジュール及びエミュレーションに使用するデータのみを保持するため、ＶＭＭのメモリ使用量が少ない。

【特許文献１】米国特許６，３９７，２４２号明細書

【非特許文献１】Intel Virtualization Technology Specification for the Intel Itanium Architecture

【非特許文献２】Intel Virtualization Technology Specification for the IA-32 Intel Architecture

【発明の開示】

【発明が解決しようとする課題】

【００１２】

先行技術であるコード変換方式と、ＶＴ方式はそれぞれ問題を有している。

【００１３】

コード変換方式は高性能だが、実行に必要な全てのＯＳコードを変換するため、変換コードの保持に使用するメモリ量が多く、省メモリは望めない。省メモリ化のためにメモリ量を無理に減らした場合は、変換コードの再利用が十分に機能しなくなり、コード変換の回数が増加して性能低下を招くため、高性能が望めない。

【００１４】

ＶＴ方式はエミュレーションの開始時に必ず例外イベントを起こすが、昨今のＣＰＵは例外イベントが発生しないことを仮定して高度な並列実行を行っているため、例外イベントが発生するとＣＰＵは内部状態の有効性判断、破棄、再実行等の複雑な処理を行う必要がある。したがって例外イベントの発生には遅延を要する。そのためＶＴ方式は、省メモリではあるが、高性能は望めない。

【００１５】

以上のように先行技術は、高性能と省メモリの何れか一方のみを解決しており、高性能と省メモリがトレードオフの関係にあって、両立できていない問題がある。

【００１６】

本発明は、前記のような問題点を鑑みてなされたものであり、エミュレーション方式における高性能と省メモリを両立できる仮想計算機システムを提供することを目的とする。

【課題を解決するための手段】

【００１７】

課題を解決するために、本発明ではゲストのプログラムコードを、ＨＷ操作の実行頻度に応じて二つの部分に分類し、実行頻度の高い部分にコード変換方式を、実行頻度の低い部分にＶＴ方式を適用する。

【発明の効果】

【００１８】

本発明ではゲストコードを、高頻度のＨＷ操作コードを含む部分と、その他、すなわち低頻度の部分に分類し、高頻度のＨＷ操作コードのエミュレーションにコード変換方式を適用し、低頻度に含まれるＨＷ操作のエミュレーションにＶＴ方式を適用する。このようにすることによって、高性能と省メモリを両立させる仮想計算機システムが実現できる。

【発明を実施するための最良の形態】

【００１９】

以下に、本発明の実施の形態を添付の図面に基づいて説明する。

【００２０】

＜実施形態１＞

図１は、本発明の第１の実施の形態の物理計算機のブロック図である。

【００２１】

この図は、仮想計算機システムが動作する物理計算機９０の構成を示す。

【００２２】

物理計算機９０は、１つ以上のＣＰＵ４００（４００－１～４００－ｎ）を含む。これらのＣＰＵ４００は、フロントサイドバス４２５を介してノースブリッジ４２０と接続する。

【００２３】

ノースブリッジ４２０は、メモリバス４３５を介して物理メモリ４１５と接続する。また、バス４４５を介してＩ／Ｏインターフェース４４０と接続する。また、コンソール４３０と直接接続する。

【００２４】

Ｉ／Ｏインターフェース４４０は、物理計算機９０の外部にあるＩ／Ｏデバイスと接続し、これらとデータを送受信する。図１の例では、Ｉ／Ｏインターフェース４４０は、ＬＡＮ４５０に接続されているネットワークアダプタ４５１、ディスク装置４６０に接続されているＳＣＳＩアダプタ４６１、ＳＡＮ（Storage Area Network）４７０に接続されるファイバーチャネルアダプタ４７１に接続する。

【００２５】

ＣＰＵ４００は、ノースブリッジ４２０を介して物理メモリ４１５にアクセスし、物理メモリ４１５に格納されているプログラムを実行することによって、プログラムに規定された処理を実行する。また、ＣＰＵ４００は、ノースブリッジ４２０及びＩ／Ｏインターフェース４４０を介して、物理計算機９０の外部にあるＩ／Ｏデバイスにアクセスする。

【００２６】

ノースブリッジ４２０は、ホスト物理メモリ４１５へのデータの入出力を制御する。また、ノースブリッジ４２０は、グラフィックコントローラを含んでおり、コンソール４３０に画像やデータを表示する。

【００２７】

物理メモリ４１５には、仮想計算機モニタ（Virtual Machine Monitor、以下、「ＶＭＭ」と表記する）１０とゲスト２０（２０－１～２０－ｎ）を実現するためのプログラムが格納されている。ＣＰＵがこのプログラムを実行することによって、ＶＭＭ１０が実行され、このＶＭＭ１０によって、ゲスト２０が実行される。

【００２８】

ＣＰＵ４００（４００－１～４００－ｎ）は、ＶＴ－ｉ機能を備えるＩＰＦ（Itanium Processor Family）プロセッサである。ＣＰＵ４００は、ＣＰＵ４００が備えるレジスタのうち、ＰＳＲ（Processor Status Register）のＶＭビットが１であるときはＨＷ操作完全禁止モードで動作する。また、ＰＳＲのＶＭビットが０であり、かつ、ＰＳＲのＣＰ

Ｌ（Current Privilege Level）フィールドが０であるときはＨＷ操作許可モードで動作する。また、何れの条件も満たさない場合は、ＨＷ操作部分禁止モードで動作する。

【００２９】

また、ＣＰＵ４００は、命令の仮想アドレスと物理アドレスとを変換するための命令アドレス変換機構を備える。この命令アドレス変換機構は、命令ＴＬＢ（Translation Look aside Buffer）を用いる。なお、ＣＰＵ４００が、命令ＴＬＢと命令ページテーブルを組み合わせてアドレス変換機構を実現してもよい。

【００３０】

次に、本実施の形態の仮想計算機システムを説明する。

【００３１】

図２は、物理計算機９０において、ゲスト２０（２０－１～２０－ｎ）とゲスト２０を稼動させるＶＭＭ１０とを含むハードウェア及びソフトウェアのブロック図である。

【００３２】

図２において、物理計算機９０上では、複数のゲスト２０（２０－１～２０－ｎ）を管理するＶＭＭ１０が稼動している。ＶＭＭ１０は、これら複数のゲスト２０が実行するハードウェアへの操作（以下、「ＨＷ操作」と表記する）をエミュレーションする。

【００３３】

各ゲスト２０は、稼動に必要なゲストコード２１（２１－１～２１－ｍ）を１つ以上備える。このゲストコード２１は、ＨＷ操作コード２５を含む。

【００３４】

ＶＭＭ１０は、コード管理モジュール３０、コード生成モジュール４０、例外イベントハンドラモジュール５０、エミュレーションモジュール６０、頻度判断モジュール７０、切り替えモジュール８０を含む。

【００３５】

以降、ゲスト２０がＨＷ操作コードを含むゲストコードを実行したときのＶＭＭ１０の処理を説明する。

【００３６】

ＶＭＭ１０は、ゲストコード２１をＨＷ操作完全禁止モードで実行させる。ゲスト２０がＨＷ操作コードを含むゲストコード２１を実行した場合は、ＣＰＵ４００は例外イベントを発生させる。そして、ＣＰＵ４００がＨＷ操作許可モードに遷移した後、ＶＭＭ１０の例外イベントハンドラモジュール５０が実行される。

【００３７】

例外イベントハンドラモジュール５０は、例外イベントの発生要因を解析し、その例外イベント、すなわちＨＷ操作コードに対応するエミュレーションモジュール６０を呼び出す。

【００３８】

エミュレーションモジュール６０は、必要に応じて物理計算機９０を操作する。これによって、ＨＷ操作コードに対応するエミュレーションが実施される。エミュレーションモジュール６０は、エミュレーションを実施する際に、頻度判断モジュール７０を呼び出す。

【００３９】

頻度判断モジュール７０は、アドレスグループ毎のＨＷ操作の頻度を判断し、切り替えモジュール８０に判断結果を渡す。

【００４０】

なお、当該ＨＷ操作コードの命令アドレスを含む所定のアドレスの範囲をアドレスグループと呼ぶ。本実施の形態では、アドレスグループに、ページを用いる。

【００４１】

また、頻度判断モジュール７０は、モジュールの初期化時など必要に応じて、物理計算機９０が有する記憶装置９１を参照して、記憶装置９１に格納された方式切り替え条件を取得する。

【００４２】
切り替えモジュール８０は、頻度判断モジュール６０からページ毎のＨＷ操作の頻度を取得する。そして、頻度が高いと判断した場合は、コード管理モジュール３０に、当該ＨＷ操作に対応する変換コードの検索を依頼する。
【００４３】
コード管理モジュール３０は、変換コードを管理する。コード管理モジュール３０は、変換コード管理表３５と、変換コードＬＲＵ（Least Recent Used）リスト３６とを含む。変換コード管理表３５は、変換コードを検索するための情報を保持する。変換コードＬＲＵリスト３６は、未使用時間が長い変換コードの情報を保持する。
【００４４】
切り替えモジュール８０は、コード管理モジュール３０への依頼の結果、変換コード管理表３５が当該ＨＷ操作コードのアドレスが含まれるページに対応する情報を記録していない場合は、コード生成モジュール４０を呼び出す。
【００４５】
コード生成モジュール４０は、当該ＨＷ操作コードのアドレスが含まれるページを参照して、対応する複数のシャドウコード３１（３１－１～３１－ｐ）と複数の昇格コード３２（３２－１～３２－ｑ）とを、コード管理モジュール３０が管理するメモリ上に生成する。そして、その内容を変換コード管理表３５に格納する。
【００４６】
その後、切り替えモジュール８０は、物理計算機９０が有する命令アドレス変換機構９３に、当該ページの仮想アドレスと、シャドウコード３１の物理アドレスとの対応付けを設定する。
【００４７】
以上の操作によって、以降は、ゲスト２０によるＨＷ操作コードのアドレスが含まれるページが実行されたときは、シャドウコード３１が実行される。
【００４８】
ＶＭＭ１０は、一連の処理の最後に、ＣＰＵの動作モードをＨＷ操作完全禁止モードに戻すための「ｒｆｉ命令」を実行する。これによって、シャドウコードはゲストコードと同じく、ＨＷ操作完全禁止モードで実行される。
【００４９】
シャドウコード３１は、ＨＷ操作コードの代わりに昇格コード３２を呼び出す。
【００５０】
昇格コード３２は、ＣＰＵをＨＷ操作許可モードに遷移させ、エミュレーションモジュール６０を呼び出す。又は、エミュレーションモジュール６０の部分複製であるエミュレーションコード３３（３３－１）を実行する。これによって、ＨＷ操作のエミュレーションが実行される。その後、ＣＰＵをＨＷ操作完全禁止モードに戻す。
【００５１】
なお、ＣＰＵをＨＷ操作許可モードに遷移させるために、昇格コード３２は、ＰＳＲのＶＭビットを操作する命令である「ｖｍｓｗ．０」と「ｖｍｓｗ．１」とを含む。
【００５２】
なお、ゲスト２０によって、シャドウコード３１に対応するゲストコード２１が書き換えられる場合がある。エミュレーションモジュール６０は、この書き換えを検出した場合は、頻度判断モジュール７０と切り替えモジュール８０に、この書き換えを通知する。頻度判断モジュール７０は、書き換えられたページの頻度を「低頻度」に設定する。切り替えモジュール８０は、書き換えられたページに対応していた変換コードを無効化し、これを変換コード管理表３５に記録する。さらに、物理計算機９０が有する命令アドレス変換機構９３から、書き換えられたページに関する設定を削除する。
【００５３】
物理計算機９０は、コンソール９２と接続されている。システム管理者がコンソール９２を用いてコード変換の設定を変更したときは、コード変換の設定を格納する記憶装置９

１に、その内容を記録する。また、頻度判断モジュール７０及びコード管理モジュール３０に、その内容を伝える。

【００５４】

図３は、記憶装置９１に格納されているコード変換の設定表９５の一例の説明図である。

【００５５】

コード変換の設定表９５は、ゲスト２０毎に、そのゲストが使用するメモリ量の上限と、そのゲストのゲストコードが高頻度であるか低頻度であるかを判断するための頻度の閾値とを格納する。

【００５６】

コード変換の設定９５は、ゲスト識別番号フィールド５０１、使用するメモリ量の上限フィールド５２０及び頻度の閾値フィールド５２１を含む。

【００５７】

ゲスト識別番号フィールド５０１は、ゲスト２０毎に付された識別子であるゲスト識別番号を格納する。使用するメモリ量の上限フィールド５２０は、メモリ変換コードを保持するために使用するメモリ量の上限を格納する。頻度の閾値フィールド５２１は、ゲストコードが高頻度であるか低頻度であるかを判断するための閾値を格納する。

【００５８】

図４は、頻度判断モジュール７０の構成ブロック図である。

【００５９】

頻度判断モジュール７０は、頻度判断コード７１とHW操作頻度表７２とを含む。

【００６０】

頻度判断コード７１は、HW操作頻度表７２を参照・更新して、HW操作の頻度を判断するプログラムである。HW操作頻度表７２は、ゲスト２０によって実行されたHW操作の回数を、ゲストコードの物理ページ単位で格納する表である。

【００６１】

図５は、HW操作頻度表７２の一例の説明図である。

【００６２】

HW操作頻度表７２は、ゲスト２０毎に、ゲストコードの物理ページアドレスと、そのページでのHW操作の実行回数とを格納する。

【００６３】

HW操作頻度表７２は、ゲスト識別番号フィールド５０１と、ゲストコードの物理ページのアドレスフィールド５０３と、HW操作実行回数フィールド５０６とを含む。

【００６４】

ゲスト識別番号フィールド５０１は、前述のコード変換の設定表９５のゲスト識別番号と同じものを格納する。ゲストコードの物理ページのアドレスフィールド５０３は、ゲストコードの物理ページのアドレスを格納する。HW操作実行回数フィールド５０６は、ゲストコードの物理ページに対応するHW操作が実行された回数を格納する。なお、HW操作実行回数の初期値は０である。

【００６５】

図６は、変換コード管理表３５の一例の説明図である。

【００６６】

変換コード管理表３５は、ゲスト２０毎に、ゲストコードの物理ページアドレスと、そのゲストコードに対応する変換コードの物理ページのアドレスを格納する。

【００６７】

変換コード管理表３５は、ゲスト識別番号フィールド５０１と、ゲストコード物理ページのアドレスフィールド５０３と、ｖａｌｉｄビットフィールド５０４と、変換コードの物理ページのアドレス５０５とを含む。

【００６８】

ゲスト識別番号フィールド５０１は、前述のコード変換表９５及びHW操作頻度表７０

のゲスト識別番号と同じものを格納する。ゲストコード物理ページのアドレスフィールド５０３は、前述のHW操作頻度表７２のゲストコードの物理ページのアドレスと同じものを格納する。ｖａｌｉｄビットフィールド５０４は、そのページに対応する変換コードが有効であるか否かを示す識別子であるｖａｌｉｄビットを格納する。このｖａｌｉｄビットの初期値は０（無効）である。変換コードの物理ページのアドレスフィールド５０５は、ゲストコードの物理ページのアドレスに対応する変換コードの物理ページのアドレスを格納する。

【００６９】

図７は、変換コードＬＲＵリスト３６の一例の説明図である。

【００７０】

変換コードＬＲＵリスト３６は、変換コード物理ページのアドレス毎に、そのアドレスの未使用時間が短い順に保持するための双方向線形リストである。

【００７１】

変換コードＬＲＵリスト３６は、ｐｒｅｖポインタフィールド５３０（５３０－１～５３０－ｔ）、変換コードの物理ページのアドレスフィールド５０５（５０５－１～５０５－ｔ）及びｎｅｘｔポインタフィールド５３１（５３１－１～５３１－ｔ）を一つの要素とし、これらの要素を、それぞれのポインタによって結合したものである。

【００７２】

ｐｒｅｖポインタフィールド５３０は前の要素を示すｐｒｅｖポインタを格納する。ｎｅｘｔポインタフィールド５３１は次の要素を示すｎｅｘｔポインタを格納する。変換コードの物理ページのアドレスフィールド５０５は、前述の変換コード管理表３５の変換コードの物理ページのアドレスと同じものを格納する。

【００７３】

次に、ゲスト２０の動作に伴うＶＭＭ１０の処理を、フローチャートを参照して説明する。

【００７４】

図８は、ＶＭＭ１０がゲスト２０を稼動するときの処理のフローチャートである。

【００７５】

まず、仮想計算機の電源の投入等によって、ゲスト２０が起動すると、ＶＭＭ１０は、自身が有するゲスト情報を参照し、起動されたゲストに関するエントリを初期化する（Ｓ２００）。

【００７６】

その後、ゲスト２０が、HW操作完全禁止モードでゲストコード又は変換コードを実行する（Ｓ２４０）。ＶＭＭ１０は、ゲスト２０によるゲストコードの実行によって、HW操作コードの実行を検出したか否かを判断する（Ｓ２５０）。

【００７７】

ゲスト２０によるHW操作コードの実行を検出したと判断した場合は、ＶＭＭ１０は、例外イベントを発生させて、ＣＰＵ４００の動作モードをHW操作許可モードに変更する。その後、ステップＳ２１０へ進む。また、ゲスト２０による昇格コードの実行を検出し、エミュレーションモジュール６０が呼ばれたと判断した場合も同様に、ＶＭＭ１０は、ＣＰＵ動作モードをHW操作許可モードに変更する。その後、Ｓ２１０へ進む。一方、ＶＭＭ１０が、HW操作コード又は昇格コードの実行を検出しないと判断した場合は、ステップＳ２４０に戻り、次のゲストコードの実行の検出を待機する。

【００７８】

ステップＳ２１０では、ＶＭＭ１０は、HW操作のエミュレーション処理を実施する。また、このエミュレーション処理によって、次に実行するコードの種別を決定する。

【００７９】

次に、ＶＭＭ１０は、仮想計算機の電源遮断等によってゲスト２０が終了したか否かを判断する。ゲスト２０が終了したと判断した場合は、ＶＭＭ１０は、当該ゲスト２０の処理を終了する。一方、ゲスト２０は終了せず、動作が継続すると判断した場合は、ステッ

プＳ２６０に進む。
【００８０】
Ｓ２６０では、ＶＭＭ１０は、システム管理者によるコード変換の設定の変更がコンソール９２に入力されたか否かを判断する。コンソール９２への入力によってコード変換の設定９５に変更があったと判断した場合は、ステップＳ２７０に移行する。一方、コード変換の設定９５に変更がなければ、ステップＳ２３０に進む。
【００８１】
ステップＳ２７０では、物理計算機９０は、コンソール９２によって設定された内容を受け取り、記憶装置９１のコード変換の設定９５を更新するコード変換の設定更新処理を実行する。
【００８２】
次に、ＶＭＭ１０は、ステップＳ２１０によって決定したコードの種別を設定する。すなわち、ゲストコード又はシャドウコードに復帰する場合は、ＶＭＭ１０は、ＣＰＵ動作モードをHW操作完全禁止モードに変更する命令を実行する。その後、ステップＳ２４０に戻る。また、昇格コードに復帰する場合は、なにもせずそのままＳ２４０に戻る。その後、昇格コード内の命令が実行されることによって、ＣＰＵ動作モードがHW操作完全禁止モードに変更される。
【００８３】
図９は、図８のステップＳ２１０のＶＭＭのエミュレーション処理のフローチャートである。
【００８４】
まず、ゲスト２０によって実行されたHW操作コードの内容に応じて、ＶＭＭ１０がエミュレーションを実施する（Ｓ１００）。
【００８５】
このとき、ＶＭＭ１０は、エミュレーションの内容はコード変換の対象となったゲストコードの変更を含むか否か、すなわち、命令アドレス変換機構９３の設定を変更するか否かを判断する（Ｓ１１０）。なお、ＶＭＭ１０は、ゲスト２０によってキャッシュ操作命令であるＦＣ（Flush Cache）命令の実行が試みられたと場合に、ゲストコードが書き換えられたと判断する。変更しないと判断した場合は、ステップＳ１３０に進む。
【００８６】
変更すると判断した場合は、ＶＭＭ１０は、変換コード管理表３５を検索して（Ｓ１１１）、該当するゲストコードに対応する変換コードがあるか否かを判断する（Ｓ１１２）。
【００８７】
対応する変換コードがあると判断した場合は、ＶＭＭ１０は、該当するゲストコードの仮想アドレスと変換コードの物理アドレスとの対応付けを命令アドレス変換機構９３に設定する（Ｓ１１３）。さらに、ＶＭＭ１０は、当該変換コードの情報を、変換コードＬＲＵリスト３６に追加して、変換コードＬＲＵリスト３６を更新する（Ｓ１１４）。
【００８８】
一方、対応する変換コードがないと判断した場合は、ＶＭＭ１０は、該当するゲストコードの仮想アドレスと該当するゲストコードの物理アドレスとの対応付けを命令アドレス変換機構９３に設定する（Ｓ１１５）。
【００８９】
次に、ＶＭＭ１０は、書き換えられたゲストコードに対応する変換コードの無効化処理を実行する。この処理によって、当該ゲストコードのページのエミュレーション方式をＶＴ方式に切り替える（Ｓ１２０）。
【００９０】
ステップＳ１３０では、ＶＭＭ１０は、HW操作の頻度判定処理を実行する。この処理によって、ゲスト２０によってなされたHW操作の頻度が高頻度であるか低頻度であるかが判定される。

【００９１】
次に、ＶＭＭ１０は、ＨＷ操作の頻度低処理の結果、頻度が高頻度であるか否かを判断する（Ｓ１４０）。高頻度であると判断した場合はステップＳ１５０に進む。低頻度であると判断した場合はステップＳ１６０に進む。
【００９２】
ステップＳ１５０では、ＶＭＭ１０は、当該ページに含まれるＨＷ操作のエミュレーションにコード変換方式を適用する処理を実行する。
【００９３】
ステップＳ１６０では、ＶＭＭ１０は、当該ＨＷ操作の次の処理の選択処理を実行する。
【００９４】
図１０は、前述の図９のステップＳ１２０の変換コードの無効化処理のフローチャートである。
【００９５】
ＶＭＭ１０は、変換コード管理表３５を参照して、書き換えられたゲストコードに対応するエントリのｖａｌｉｄビットをクリアする。すなわち、ｖａｌｉｄビットを０に変更する（Ｓ６００）。
【００９６】
次に、ＶＭＭ１０は、ＨＷ操作頻度表７２を参照して、書き換えられたゲストコードに対応するエントリのＨＷ操作の実行回数を０に設定する（Ｓ６１０）。
【００９７】
次に、ＶＭＭ１０は、書き換えられたゲストコードの仮想アドレスと当該ゲストコードの物理アドレスとの対応付けを命令アドレス変換機構９３に設定する（Ｓ６３０）。
【００９８】
図１１は、前述の図９のＳ１３０の頻度の判定処理のフローチャートである。
【００９９】
まず、ＶＭＭ１０は、当該ＨＷ操作コードを実行したのは、ゲスト２０のアプリケーションであるか否かを判断する（Ｓ７９０）。アプリケーションが実行したと判断した場合は、ステップＳ７３０に進む。ゲスト２０のＯＳが実行したと判断した場合は、ステップＳ７００に進む。
【０１００】
ステップＳ７００では、ＶＭＭ１０は、当該ＨＷ操作コードが属するページの物理アドレスを算出する。
【０１０１】
次に、ＶＭＭ１０は、ＨＷ操作頻度表７２を参照して、算出したページの物理アドレスに対応するエントリの、ＨＷ操作の実行回数を取得する（Ｓ７１０）。
【０１０２】
次に、ＶＭＭ１０は、コード変換の設定表９５を参照して、ＨＷ操作コードの要求元のゲスト２０の識別番号のエントリから、頻度の閾値を取得する。そして、取得したＨＷ操作の実行回数と取得した頻度の閾値とを比較して、実行回数が閾値以上であるか否かを判断する（Ｓ７２０）。実行回数が閾値以上であると判断した場合は、ステップＳ７４０に進む。実行回数が閾値未満であると判断した場合は、ステップＳ７３０に進む。
【０１０３】
Ｓ７４０では、ＶＭＭ１０は、当該ページでのＨＷ操作の頻度は高頻度であると判定する。
【０１０４】
Ｓ７３０では、ＶＭＭ１０は、当該ページでのＨＷ操作の頻度が低頻度であると判定する。
【０１０５】
次に、ＶＭＭ１０は、ＨＷ操作頻度表７２の当該ページのＨＷ操作の実行回数に１を加

算して、HW操作頻度表７２を更新する（Ｓ７５０）。

【０１０６】

図１２は、前述の図９のステップＳ１６０の次処理の選択処理のフローチャートである。

【０１０７】

ＶＭＭ１０は、ゲスト２０が終了したか否か、すなわち、ゲスト２０が稼働する仮想計算機２１の電源がＯＦＦにされたか否かを判断する（Ｓ１０００）。仮想計算機２１の電源がＯＦＦにされたと判断した場合はステップＳ１０１０に進む。そうでなければステップＳ１０２０に進む。

【０１０８】

ステップＳ１０１０では、ＶＭＭ１０は、次に実行する処理として、ゲスト終了を選択する。

【０１０９】

ステップＳ１０２０では、ＶＭＭ１０は、次に実行する処理にゲストコード又は変換コードを継続して適用することを決定する。

【０１１０】

図１３は、前述の図９のステップＳ１５０のコード変換方式の適用処理のフローチャートである。

【０１１１】

ステップＳ１２００では、ＶＭＭ１０は、変換コード管理表３５を参照して、当該HW操作コードを含むページの物理アドレスに対応する変換コードの有無を判断する。変換コードが存在しないと判断した場合は、ステップＳ１２１０に進む。変換コードが設定されていると判断した場合は、本処理を終了して、図９のフローチャートに戻る。

【０１１２】

Ｓ１２１０では、ＶＭＭ１０は、変換コードを格納するためのメモリ領域のメモリ容量に必要な空き容量が存在するか否かを判断する。メモリ領域に必要な空き容量が存在しないと判断した場合は、ステップＳ１２２０に移行する。必要な空き容量が存在すると判断した場合は、ステップＳ１２３０に進む。

【０１１３】

ステップＳ１２２０では、ＶＭＭ１０は、変換コードＬＲＵリスト３６を参照して、未使用時間の最も長い変換コードを含む要素を一つ選択する。そして、選択された要素に含まれる変換コードに属するページの物理アドレスを無効化することによって、変換コードをメモリ領域から破棄する。さらに、選択された要素に含まれる変換コードの情報を、変換コードＬＲＵ３６から削除する。そして、ステップＳ１２１０に戻る。すなわち、必要な空き容量が確保できるまで、未使用時間の最も長い変換コードをメモリ領域から破棄する。

【０１１４】

ステップＳ１２３０では、ＶＭＭ１０は、当該ページのゲストコードに対応する変換コードを生成し、生成された変換コードを保存する。変換コードのうち、シャドウコードはゲストコードと１対１に対応し、HW操作コードの代わりに昇格コードを呼び出す。また、変換コードのうち、昇格コードは、ｖｍｓｗ命令を含み、エミュレーションモジュール６０を呼び出すコード、又は、エミュレーションモジュール６０の部分複製であるエミュレーションコード３３の少なくとも一方を含む。

【０１１５】

次に、ＶＭＭ１０は、生成した変換コードの情報を、変換コード管理表３５に格納する。また、生成した変換コードの情報を変換コードＬＲＵ３６リストに追加する（Ｓ１２４０）。

【０１１６】

図１４は、前述の図８のステップＳ２７０のコード変換の設定更新処理のフローチャートである。

【０１１７】
まず、物理計算機９０が、システム管理者によって設定されたコード変換の設定の変更を、コンソールから受信する（Ｓ１４００）。
【０１１８】
物理計算機９０は、受け取った変更内容を、記憶装置９１のコード変換の設定表９５に反映させて、コード変換の設定表９５を更新する（Ｓ１４１０）。
【０１１９】
次に、物理計算機９０は、更新されたコード変換の設定表９５を、頻度判断モジュール７０及びコード管理モジュール３０に通知する。これによって、頻度判断モジュール７０及びコード管理モジュール３０は、自身が格納しているコード変換の設定表９５を更新する。
【０１２０】
以上のように、本発明の第１の実施の形態の仮想計算機システムでは、ゲスト２０によってＨＷ操作を含むゲストコードの実行が要求されたときに、ＶＭＭ１０は、当該ＨＷ操作の実行頻度に応じて処理方法を変える。すなわち、当該ＨＷ操作を二つの部分に分け、実行頻度の高い部分に実行効率の高いコード変換方式を適用し、実行頻度の低い部分にメモリ消費量の少ないＶＴ方式を適用する。このようにすることによって、高性能と省メモリとを両立することが可能なエミュレーション処理が可能となる。
【０１２１】
＜実施形態２＞
次に、本発明の第２の実施の形態の仮想計算機システムを説明する。第２の実施の形態では、操作の種別である命令タイプによって、ＨＷ操作の頻度を判断する。
【０１２２】
なお、第２の実施の形態のハードウェア構成は図１と同一である。また、第１の実施の形態と同一の作用をする構成には同一の符号を付し、その説明は省略する。
【０１２３】
第２の実施形態では、頻度判断に命令タイプを用いるため、コード変換の設定内容、ＨＷ操作頻度表のフォーマットの２点のみが実施例１と異なる。
【０１２４】
図１５は、第２の実施の形態の、コード変換の設定表９５の説明図である。
【０１２５】
第１の実施の形態において前述した図４とは異なり、コード変換の設定表９５は、高頻度を判断するために、閾値ではなく命令タイプを格納する。
【０１２６】
コード変換の設定９５は、ゲスト識別番号フィールド５０１、使用するメモリ量の上限フィールド５０２及び命令タイプのリストフィールド５２２を含む。
【０１２７】
ゲスト識別番号フィールド５０１は、ゲスト２０毎に付された識別子であるゲスト識別番号を格納する。使用するメモリ量の上限フィールド５２０は、メモリ変換コードを保持するために使用するメモリ量の上限を格納する。命令タイプのリストフィールド５２２は、ゲストコードの命令タイプのうち、高頻度であると判断する命令タイプをリスト形式で格納する。
【０１２８】
図１６は、第２の実施の形態のＨＷ操作頻度表７２の説明図である。
【０１２９】
第１の実施の形態において前述した図５とは異なり、ＨＷ操作頻度表７２は、ゲスト２０毎に、命令タイプ５０７と、その命令タイプが高頻度であるか否かを判断するための頻度フラグ５０８とを格納する。
【０１３０】
このＨＷ操作頻度表７２は、管理者によって設定されたコード変換の設定表９５を元に

、頻度判断コード７１によって生成される。すなわち、コード変換の設定表９５のゲスト識別番号及び命令タイプのリストを、ゲスト識別番号と命令タイプとによって並び替えたものである。

【０１３１】

ＨＷ操作頻度表７２は、ゲスト識別番号フィールド５０１と、命令タイプフィールド５０７と、頻度フラグフィールド５０８とを含む。

【０１３２】

ゲスト識別番号フィールド５０１は、前述のコード変換の設定表９０１のゲスト識別番号と同じものを格納する。命令タイプフィールド５０７は、ゲストコードの命令タイプを示す識別子を格納する。頻度フラグフィールド５０８は、命令タイプに対応する頻度フラグ格納する。頻度フラグは１であれば高頻度であることを示し、０であれば低頻度であることを示す。なお、頻度フラグの初期値は、頻度判断モジュール７０がコード変換の設定表９５から取得する。

【０１３３】

次に、第２の実施の形態の、ゲスト２０の動作に伴うＶＭＭ１０の処理を、フローチャートを参照して説明する。

【０１３４】

なお、第２の実施の形態の処理の概要は、前述の第１の実施の形態において前述した図８と同一である。

【０１３５】

また、前述の図８のステップＳ２１０のＶＭＭのエミュレーション処理は、前述の図９の処理と同一である。

【０１３６】

図１７は、前述の図９のステップＳ１２０の、第２の実施の形態の変換コードの無効化処理のフローチャートである。

【０１３７】

ＶＭＭ１０は、変換コード管理表３５を参照して、書き換えられたゲストコードに対応するエントリの、ｖａｌｉｄビットフィールド５０４をクリアする。すなわち、ｖａｌｉｄビットを０に変更する（Ｓ６００）。

【０１３８】

次に、ＶＭＭ１０は、書き換えられたゲストコードの仮想アドレスと当該ゲストコードの物理アドレスとの対応付けを命令アドレス変換機構９３に設定する（Ｓ６３０）。

【０１３９】

図１８は、前述の図９のステップＳ１３０の、第２の実施の形態の頻度判断処理のフローチャートである。

【０１４０】

まず、ＶＭＭ１０は、当該ＨＷ操作コードを実行したのは、ゲスト２０のアプリケーションであるか否かを判断する（Ｓ７９０）。アプリケーションが実行したと判断した場合は、ステップＳ８３０に進む。ゲスト２０のＯＳが実行したと判断した場合は、ステップＳ８００に進む。

【０１４１】

ステップＳ８００では、ＶＭＭ１０は、当該ＨＷ操作コードの命令タイプを取得する。

【０１４２】

次に、ＶＭＭ１０は、ＨＷ操作頻度表７２を参照して、取得した命令タイプに対応するエントリの頻度フラグを取得する（Ｓ８１０）。

【０１４３】

次に、ＶＭＭ１０は、取得した頻度フラグが１であるか否かを判断する。頻度フラグが１、すなわち高頻度を示すものであればステップＳ８４０に進む。頻度フラグが０、すなわち、低頻度を示すものであればステップＳ８３０に進む（Ｓ８２０）。

【０１４４】

ステップＳ８４０では、ＶＭＭ１０は、当該ページでのＨＷ操作の頻度が高頻度であると判定する。

【０１４５】

ステップＳ８３０では、ＶＭＭ１０は、当該ページでのＨＷ操作の頻度が低頻度であると判定する。

【０１４６】

なお、図９のステップＳ１６０の次処理の選択処理は、前述の第１の実施の形態の図１２の処理と同一である。

【０１４７】

また、図９のステップＳ１５０のコード変換方式の適用処理は、前述の第１の実施の形態の図１３の処理とほぼ同一であるが、以下の１点のみが異なる。

【０１４８】

まず、ステップＳ１２３０では、ＶＭＭ１０は、当該ページに含まれる命令タイプが高頻度でない命令は、コード変換の対象から除外し、シャドウコードにゲストコードの命令をそのままコピーする。

【０１４９】

また、図８のステップＳ２７０のコード変換の設定更新処理は、前述の第１の実施の形態の図１４と同一である。

【０１５０】

以上のように、本発明の第２の実施の形態では、ゲストコードをＨＷ操作の命令タイプに応じて二つの部分に分類する。そして、実行頻度の高い命令タイプを含む部分に、コード変換方式を適用し、実行頻度の高い命令タイプを含まない部分にＶＴ方式を適用する。このようにすることによって、高性能と省メモリを両立させるエミュレーション処理が可能となる。

【０１５１】

＜実施形態３＞

次に、本発明の第３の実施の形態の仮想計算機システムを説明する。第３の実施の形態では、ＨＷ操作の間隔によって頻度を判断する。

【０１５２】

なお、第１の実施の形態と同一の作用をする構成には同一の符号を付し、その説明は省略する。

【０１５３】

第３の実施の形態のハードウェア構成は図１とほぼ同一である。しかしながら、第３の実施の形態では、用いられるＣＰＵ（４００－１～４００－ｎ）の機能が異なる。

【０１５４】

ＣＰＵ４００（４００－１～４００－ｎ）は、ＶＴ－ｘ機能に対応したｘ８６プロセッサである。このＣＰＵ４００は、ＨＷ操作完全禁止モードと、ＨＷ操作許可モードと、ＨＷ操作部分禁止モードとを備える。ＨＷ操作完全禁止モードはＣＰＵ４００をＶＭＸ ｎｏｎ－ｒｏｏｔモードに設定することで有効化される。ＨＷ操作許可モードはＣＰＵ４００をＶＭＸ ｒｏｏｔモードかつ特権レベル（ＣＰＬ：Current Privilege Level）を０に設定することで有効化される。ＨＷ操作部分禁止モードはＣＰＵ４００をＶＭＸ ｒｏｏｔモードかつＣＰＬを非０に設定することで有効化される。

【０１５５】

また、ＣＰＵ４００が、ＶＭＸ ｒｏｏｔモードからＶＭＸ ｎｏｎ－ｒｏｏｔモードに遷移するためには、ＶＭｅｎｔｒｙ操作を実行する必要がある。ＶＭｅｎｔｒｙ操作とは、ゲストＯＳの起動を指示するためのＶＭＬＡＵＮＣＨ命令の実行と、ゲストＯＳを一時停止するためのＶＭＲＥＳＵＭＥ命令の実行との総称である。また、ＶＭＸ ｎｏｎ－ｒｏｏｔモードからＶＭＸ ｒｏｏｔモードに遷移するためには、ＶＭｅｘｉｔイベントを実行する必要がある。ＶＭｅｘｉｔイベントとは、ＨＷ操作の実行に起因する例外イベントである。

【０１５６】
次に、第３の実施の形態の仮想計算機システムを説明する。
【０１５７】
図１９は、物理計算機９０において、ゲスト２０（２０－１～２０－ｎ）とゲスト２０を稼動させるＶＭＭ１０とを含むハードウェア及びソフトウェアのブロック図である。
【０１５８】
図１９は、前述の第１の実施の形態の図２のブロック図とほぼ同一である。しかしながら、コード管理モジュール３０の動作が異なる。
【０１５９】
以降、ゲスト２０がHW操作コードを含むゲストコードを実行したときのＶＭＭ１０の処理を説明する。
【０１６０】
ＶＭＭ１０は、ゲスト２０がHW操作コード２５を含むゲストコード２５を実行した場合は、ＶＭＭ１０は例外イベントを発生させる。この例外イベントによって、ＣＰＵ４００がHW操作許可モードに遷移する。その後、ＶＭＭ１０の例外イベントハンドラモジュール５０が実行される。
【０１６１】
例外イベントハンドラモジュール５０は、例外イベントの発生要因を解析し、その例外イベント、すなわち、HW操作コードに対応するエミュレーションモジュール６０を呼び出す。
【０１６２】
エミュレーションモジュール６０は、必要に応じて物理計算機９０を操作する。これによって、HW操作コードに対応するエミュレーションが実施される。エミュレーションモジュール６０は、エミュレーションを実施する際に、頻度判断モジュール７０を呼び出す。
【０１６３】
頻度判断モジュール７０は、ゲスト２０によって、前回のHW操作がされたときの時刻情報と今回のHW操作がされたときの時刻情報との差から、HW操作間隔の短さ、すなわちHW操作の頻度を算出する。そして、算出された頻度の履歴を用いて、今回のHW操作がされたときの時刻情報と次回にHW操作がされたときの時刻情報の間隔の短さ、すなわち頻度を判断する。この結果を、切り替えモジュール８０に渡す。
【０１６４】
なお、時刻情報とは時刻を特定可能な情報であり、本実施形態にではゲストによって実行されたページ間の分岐命令の回数である。時刻情報には、物理計算機９０が備える時計の時刻の値や、実行されたゲスト命令数、実行されたゲストのＣＡＬＬ命令及びＲＥＴ命令数等、時刻を近似できる任意の情報が利用できる。
【０１６５】
頻度判断モジュール７０は、モジュール初期化時など、必要に応じて、物理計算機９０の記憶装置９１に格納された方式切り替え条件を参照し、その内容を頻度判断モジュール７０内に格納しておく。
【０１６６】
頻度判断モジュール７０によって、今回のHW操作と次回のHW操作の間隔が長い、すなわち、低頻度であると判断された場合は、切り替えモジュール８０は、ＣＰＵ４００の動作モードをHW操作完全禁止モードに戻す。すなわち、切り替えモジュール８０は、ＣＰＵ４００に対してＶＭｅｎｔｒｙ操作を実行する。この操作の後に、処理をゲストコードに分岐する。
【０１６７】
一方、頻度判断モジュール７０によって、今回のHW操作と次回のHW操作の間隔が短い、すなわち、高頻度であると判断された場合は、切り替えモジュール８０は、まず、次に実行するゲストコードに対応する変換コードのアドレスを、コード管理モジュール３０

に問い合わせる。そして、ＣＰＵ４００の動作モードをHW操作部分禁止モードに戻す。すなわち、切り替えモジュール８０は、ＣＰＬ増加分岐命令を実行する。この命令の後に、処理を変換コードに分岐する。この処理によって、変換コードはHW操作部分禁止モードで実行される。なお、ＣＰＬ増加分岐命令には、ＩＲＥＴ命令、ＳＹＳＲＥＴ命令、ＳＹＳＥＸＩＴ命令等を用いる。

【０１６８】

切り替えモジュール８０からの問い合わせに対して、コード管理モジュール３０は、変換コード管理表３５を検索して、問い合わせコードに対応する変換コードが存在するか否かを判断する。変換コードが存在する場合は、当該変換コードのアドレスを返す。問い合わせコードに対応する変換コードが存在しない場合は、コード生成モジュール４０を呼び出して、変換コードを生成し、生成された変換コードのアドレスを返す。

【０１６９】

コード生成モジュール４０は、次のゲストコードページを参照して対応するシャドウコード３１（３１－１～３１－ｐ）と昇格コード３２（３２－１）とを、コード管理モジュール３０が管理するメモリ上に生成する。そして、その内容を変換コード管理表３５に登録する。

【０１７０】

なお、シャドウコード３１（３１－１～３１－ｐ）は、HW操作コードの代わりに昇格コード３２（３２－１）を含む。昇格コード３２（３２－１）は、ＣＰＵをHW操作許可モードに遷移させ、エミュレーションモジュール６０を呼び出してHW操作をエミュレーションし、その後、ＣＰＵをHW操作部分禁止モードに戻す。昇格コード３２は、ＣＰＬ減少分岐命令（特権レベル減少分岐命令）を含む。ＣＰＬ減少分岐命令には、ｆａｒＣＡＬＬ命令、ＳＹＳＣＡＬＬ命令、ＳＹＳＥＮＴＥＲ命令等を用いる。

【０１７１】

なお、ゲスト２０によって、シャドウコード３１に対応するゲストコード２１が書き換えられる場合がある。エミュレーションモジュール６０は、この書き換えを検出した場合は、切り替えモジュール８０に、この書き換えを通知する。切り替えモジュール８０は、書き換えられたページに対応していた変換コードを無効化して、これを変換コード管理表３５に記録する。

【０１７２】

なお、記憶装置９１に格納されているコード変換の設定表９５は、前述の第１の実施の形態の図３と同一である。

【０１７３】

図２０は、頻度判断モジュール７０の構成ブロック図である。

【０１７４】

頻度判断モジュール７０は、第１の実施の形態の頻度判断モジュール７１とほぼ同一であるが、HW操作履歴表７３を含んでいる。

【０１７５】

HW操作頻度表７２は、あるHW操作Ｘと次に実行されたHW操作との間隔が短い、すなわち高頻度である場合に、そのHW操作Ｘを実施するゲストコードの物理アドレスを格納する。HW操作履歴表７３は、最後に処理されたHW操作を実行した動作主体（ＯＳまたはＡＰ）と、ゲストコードの物理アドレスと、時刻情報を格納する。

【０１７６】

図２１は、HW操作頻度表７２の一例の説明図である。

【０１７７】

HW操作頻度表７２は、ゲスト２０毎に、当該エントリの有効性を示すｖａｌｉｄビットと、ゲストコードの物理アドレスとを格納する。

【０１７８】

HW操作頻度表７２は、ゲスト識別番号フィールド５０１と、ｖａｌｉｄビットフィールド５０４と、ゲストコードの物理アドレスフィールド５０９とを含む。

【０１７９】
ゲスト識別番号フィールド５０１は、ゲスト２０毎に付された識別子であるゲスト識別番号を格納する。ｖａｌｉｄビットフィールド５０４は、ゲストコードのうち、その物理アドレスが、HW操作の間隔が短い、すなわち、高頻度である部分の始点又は中間点であることを示すｖａｌｉｄビットを格納する。ゲストコードの物理のアドレスフィールド５０９は、ゲストコードの物理アドレスを格納する。
【０１８０】
なお、本表に含まれないゲストコードの物理アドレスは、HW操作の間隔が長い、すなわち、低頻度である部分の始点又は中間点である。また、ｖａｌｉｄビットの初期値は０、すなわち無効である。
【０１８１】
図２２は、HW操作履歴表７３の一例の説明図である。
【０１８２】
HW操作履歴表７３は、ゲスト識別番号フィールド５０１と、ＣＰＵ識別番号フィールド５０２と、動作主体フィールド５２３と、ゲストコードの物理アドレスフィールド５０９と、時刻情報フィールド５１０とを含む。
【０１８３】
ゲスト識別番号フィールド５０１は、前述の図２１のゲスト識別番号と同じものを格納する。ＣＰＵ識別番号フィールド５０２は、ＣＰＵ４００を識別するための番号を格納する。動作主体フィールド５２３は、最後に処理されたHW操作を実行した主体はＯＳであるかＡＰであるかを区別するための識別子を格納する。ゲストコードの物理アドレス５０９は、前述の図２１のゲストコード物理アドレスと同じものを格納する。時刻情報フィールド５１０は、前述のように、最後に処理されたHW操作の時点での、ゲストによって実行されたページ間の分岐命令の回数を格納する。なお、物理計算機９０が備える時計の時刻の値や、実行されたゲスト命令数、実行されたゲストのＣＡＬＬ命令及びＲＥＴ命令数等、時刻を近似できる任意の情報が利用できる。
【０１８４】
図２３は、切り替えモジュール８０の構成ブロック図である。
【０１８５】
切り替えモジュール８０は、方式切り替えコード８１と動作方式表８２とを含む。
【０１８６】
なお、動作方式とは、エミュレーションモジュールの呼び出し方式である。本実施の形態では、ＶＴ方式又はコード変換方式を用いる。方式切り替えコード８１は、頻度に応じて動作方式であるコード変換方式とＶＴ方式を切り替え、その結果を操作方式表８２に格納する。動作方式表８２は、現在適用中の動作方式を格納する。
【０１８７】
図２４は、動作方式表８２の一例の説明図である。
【０１８８】
動作方式表８２は、ゲスト識別番号フィールド５０１と、ＣＰＵ識別番号フィールド５０２と動作方式フィールド５１３とを含む。
【０１８９】
ゲスト識別番号フィールド５０１は、前述の図２１のHW操作頻度表７２のゲスト識別番号と同じものを格納する。ＣＰＵ識別番号フィールド５０２は、前述の図２２のHW操作履歴表７３のＣＰＵ識別番号と同じものを格納する。動作方式フィールド５１３は、現在適用中の動作方式を格納する。なお、動作方式の初期値は「ＶＴ方式」である。
【０１９０】
図２５は、変換コード管理表３５の一例の説明図である。
【０１９１】
変換コード管理表３５は、ゲスト２０毎に、ゲストコードの物理アドレス５０９と、そのゲストコードに対応する変換コードの物理アドレス５１１を格納する。

【０１９２】
変換コード管理表３５は、ゲスト識別番号フィールド５０１と、ゲストコード物理アドレスフィールド５０９と、ｖａｌｉｄビットフィールド５０４と、変換コードの物理アドレス５１１とを含む。
【０１９３】
ゲスト識別番号フィールド５０１は、前述の図２１のＨＷ操作頻度表７２のゲスト識別番号と同じものを格納する。ゲストコードの物理アドレス５０９は、前述の図２１のゲストコード物理アドレスと同じものを格納する。ｖａｌｉｄビットフィールド５０４は、その物理アドレスに対応する変換コードが有効であるか否かを示す識別子であるｖａｌｉｄビットを格納する。このｖａｌｉｄビットの初期値は０（無効）である。変換コードの物理アドレスフィールド５０９は、ゲストコードの物理アドレスに対応する変換コードの物理アドレスを格納する。
【０１９４】
次に、第３の実施の形態の、ゲスト２０の動作に伴うＶＭＭ１０の処理を、フローチャートを参照して説明する。
【０１９５】
図２６は、ＶＭＭ１０がゲスト２０を稼動するときの処理のフローチャートである。
【０１９６】
図２６は、前述の第１の実施の形態の図８のフローチャートに類似しているが、最初のＨＷ操作は必ずゲストコードによってエミュレーションされる点が異なる。
【０１９７】
まず、仮想計算機の電源の投入等によって、ゲスト２０が起動すると、ＶＭＭ１０は、自身が有するゲスト情報を参照し、起動されたゲストに関するエントリを初期化する（Ｓ２００）。
【０１９８】
その後、ゲスト２０が、ＨＷ操作完全禁止モードでゲストコードを実行する（Ｓ３４０）。ＶＭＭ１０は、ゲスト２０によるゲストコードの実行によって、ＨＷ操作コードの実行を検出したか否かを判断する（Ｓ３５０）。
【０１９９】
ゲスト２０によるＨＷ操作コードの実行を検出した場合は、ＶＭＭ１０は、例外イベントを発生させて、ＣＰＵ４００の動作モードをＨＷ操作許可モードに変更する。その後、ステップＳ２１０へ進む。一方、ＶＭＭ１０が、ＨＷ操作コードの実行を検出しない場合は、ステップＳ３４０に戻り、次のゲストコードを実行する。
【０２００】
ステップＳ２１０では、ＶＭＭ１０は、ＨＷ操作のエミュレーション処理を実施する。また、このエミュレーション処理によって、次に実行するコードの種別を決定する。
【０２０１】
次に、ＶＭＭ１０は、仮想計算機の電源遮断等によってゲスト２０が終了したか否かを判断する。ゲスト２０が終了したと判断した場合は、ＶＭＭ１０は、当該ゲスト２０の処理を終了する。一方、ゲスト２０は終了せず、動作が継続すると判断した場合は、ステップＳ２６０に進む。
【０２０２】
Ｓ２６０では、ＶＭＭ１０は、システム管理者によるコード変換の設定の変更がコンソール９２に入力されたか否かを判断する。コンソール９２への入力によってコード変換の設定９５に変更があったと判断した場合は、ステップＳ２７０に移行する。一方、コード変換の設定９５に変更がなければ、ステップＳ３３０に進む。
【０２０３】
ステップＳ２７０では、物理計算機９０は、コンソール９２によって設定された内容を受け取り、記憶装置９１のコード変換の設定９５を更新するコード変換の設定更新処理を実行する。

【０２０４】
次に、ＶＭＭ１０は、次にゲスト２０によって実行される処理が変換コードの実行であるか否かを判断する（Ｓ３３０）。次の処理が変換コードの実行であると判断した場合は、ＶＭＭ１０は、ＣＰＬ増加分岐命令を実行してＣＰＵ動作モードをＨＷ操作部分禁止モードに変更する。その後、ステップＳ３６０に進む。一方、次の処理がゲストコードの実行であると判断した場合は、ＶＭＭ１０は、ＶＭｅｎｔｒｙ操作を実行してＣＰＵ動作モードをＨＷ操作完全禁止モードに変更する。その後、ステップＳ３４０に戻る。
【０２０５】
ステップＳ３６０では、ＶＭＭ１０は、ＨＷ操作部分禁止モードで変換コードを実行する。
【０２０６】
次に、ＶＭＭ１０は、ゲスト２０による昇格コードの実行を検出し、エミュレーションモジュール６０が呼ばれたか否かを判断する（Ｓ３７０）。昇格コードが実行されてエミュレーションモジュールが呼ばれた場合は、ＶＭＭ１０は、ＣＰＵ動作モードをＨＷ操作許可モードに変更して、Ｓ２１０へ進む。一方、ＶＭＭ１０が、昇格コードの実行を検出しない場合は、ステップＳ３６０に戻り、次の変換コードを実行する。
【０２０７】
図２７は、図２６のステップＳ２１０のＶＭＭのエミュレーション処理のフローチャートである。
【０２０８】
まず、ゲスト２０によって実行されたＨＷ操作コードの内容に応じて、ＶＭＭ１０がエミュレーションを実施する（Ｓ１００）。
【０２０９】
このとき、ＶＭＭ１０は、エミュレーションの内容はコード変換の対象となったゲストコードの変更を含むか否かを判断する（Ｓ１１０）。なお、ＶＭＭ１０は、ライトプロテクトを設定したページに対してゲスト２０が書き込みを行った場合に、当該ページにゲストコードが存在するか否かを判定して、ゲストコードの変更の有無を判断する。変更しないと判断した場合は、ステップＳ１３０に進む。一方、変更すると判断した場合は、ステップＳ１２０に進む。
【０２１０】
ステップＳ１２０では、ＶＭＭ１０は、書き換えられたゲストコードに対応する変換コードの無効化処理を実行する（Ｓ１２０）。
【０２１１】
ステップＳ１３０では、ＶＭＭ１０は、ＨＷ操作の頻度判定処理を実行する。この処理によって、ゲスト２０によってなされたＨＷ操作の頻度が高頻度であるか低頻度であるかが判定される。
【０２１２】
次に、ＶＭＭ１０は、ＨＷ操作の頻度判定処理の結果、頻度が高頻度であるか否かを判断する（Ｓ１４０）。高頻度であると判断した場合はステップＳ１７０に進む。低頻度であると判断した場合はステップＳ１８０に進む。
【０２１３】
ステップＳ１７０では、ＶＭＭ１０は、以降のＨＷ操作の検出手段をコード変換方式に決定する。そして、動作方式表８２の当該ゲスト及びＣＰＵに対応する動作方式５１３をコード変換方式に変更する。
【０２１４】
一方、ステップＳ１８０では、ＶＭＭ１０は、以降のＨＷ操作の検出手段をＶＴ方式に決定する。そして、動作方式表８２の当該ゲスト及びＣＰＵに対応する動作方式５１３をＶＴ方式に変更する。
【０２１５】
次に、ＶＭＭ１０は、Ｓ１６０では、当該ＨＷ操作の次の処理の選択処理を実行する。

【０２１６】
図２８は、図２７のステップＳ１２０の変換コードの無効化処理のフローチャートである。
【０２１７】
ＶＭＭ１０は、変換コード管理表３５を参照して、書き換えられたゲストコードに対応するエントリのｖａｌｉｄビットをクリアする。すなわち、ｖａｌｉｄビットを０に変更する（Ｓ６００）。
【０２１８】
次に、ＶＭＭ１０は、書き換えられたゲストコードに対応するライトプロテクトを解除する（Ｓ６２０）。
【０２１９】
図２９は、図２７のステップＳ１３０の頻度判断処理のフローチャートである。
【０２２０】
まず、ＶＭＭ１０は、ＨＷ操作履歴表７３を参照して、１つ前に実行されたＨＷ操作に関するエントリを取得する。そして、当該エントリの、動作主体とゲストコードの物理アドレスと時刻情報とを取得する（Ｓ９００）。
【０２２１】
次にＶＭＭ１０は、１つ前に実行されたＨＷ操作の動作主体がアプリケーション（ＡＰ）であるか否かを判断する（Ｓ９９０）。動作主体がＡＰであると判断した場合はステップＳ９４０に進む。動作主体がＡＰではない、すなわち動作主体がＯＳであると判断した場合はステップＳ９１０に進む。
【０２２２】
ステップＳ９１０では、ＶＭＭ１０は、今回のＨＷ操作の時刻情報と１つ前のＨＷ操作の時刻情報との差から、ＨＷ操作の間隔を算出する。
【０２２３】
次に、ＶＭＭ１０は、算出したＨＷ操作の間隔と、コード変換の設定表９５から取得した閾値とを比較する。比較の結果、算出したＨＷ操作の間隔が取得した閾値の範囲以内であるか否かを判断する（Ｓ９２０）。算出したＨＷ操作の間隔が閾値以内であると判断した場合はステップＳ９３０に進む。一方、算出したＨＷ操作の間隔が、閾値を越えると判断した場合はステップＳ９４０に進む。
【０２２４】
ステップＳ９３０では、ＶＭＭ１０は、１つ前のＨＷ操作に関するエントリのゲストコードの物理アドレスを、ＨＷ操作頻度表７２に追加する。そして、追加したエントリのｖａｌｉｄビットを１に設定する。これによって、このゲストコードの物理アドレスが、高頻度部分の始点又は中間点として設定される。
【０２２５】
ステップＳ９４０では、ＶＭＭ１０は、今回実行されたＨＷ操作に関するエントリ、すなわち、動作主体とゲストコードの命令アドレスと時刻情報とをＨＷ操作履歴表７３に格納する。
【０２２６】
次に、ＶＭＭ１０は、今回実行されたＨＷ操作に関して、当該ＨＷ操作のゲストコードの命令アドレスがＨＷ操作頻度表７２に含まれているか否かを判断する（Ｓ９５０）。ＨＷ操作頻度表７２に含まれていると判断した場合はステップＳ７４０に進む。ＨＷ操作頻度表７２に含まれていないと判断した場合はステップＳ７３０に進む。
【０２２７】
ステップＳ７４０では、ＶＭＭ１０は、今回実行されたＨＷ操作は高頻度部分の始点又は中間点であると判定する。
【０２２８】
ステップＳ７３０では、ＶＭＭ１０は、今回実行されたＨＷ操作は低頻度部分の始点又は中間点であると判定する。

【０２２９】
図３０は、図２７のステップＳ１６０の次処理の選択処理のフローチャートである。
【０２３０】
ＶＭＭ１０は、ゲスト２０が終了したか否か、すなわち、ゲスト２０が稼働する仮想計算機２１の電源がＯＦＦにされたか否かを判断する（Ｓ１０００）。仮想計算機２１の電源がＯＦＦにされたと判断した場合はステップＳ１０１０に進む。そうでなければステップＳ１１２０に進む。
【０２３１】
ステップＳ１０１０では、ＶＭＭ１０は、次に実行する処理として、ゲスト終了を選択する。
【０２３２】
ステップＳ１１２０では、ＶＭＭ１０は、動作方式表８２を参照して、現在の動作方式がコード変換方式であるか否かを判断する。現在の動作方式がコード変換方式であると判断した場合はステップＳ１０４０に進む。現在の動作方式がＶＴ方式であると判断した場合はステップＳ１０６０に進む。
【０２３３】
ステップＳ１０４０では、ＶＭＭ１０は、次に実行する処理に変換コードを適用することを決定する。そして、次に実行する変換コードを検索するための変換コードの検索処理を実行する（Ｓ１１３０）。
【０２３４】
一方、ステップＳ１０６０では、ＶＭＭ１０は、次に実行する処理にゲストコードを適用することを決定する。
【０２３５】
図３１は、図３０のステップＳ１１３０の変換コードの検索処理のフローチャートである。
【０２３６】
まず、ＶＭＭ１０は、変換コード管理表３５を参照して、今回処理したＨＷ操作コードの次のゲストコードに対応するエントリを検索する（Ｓ１３００）。
【０２３７】
次に、ＶＭＭ１０は、該当するエントリが存在するか否かを判断する（Ｓ１３１０）。エントリが存在すると判断した場合はステップＳ１３２０に進む。エントリが存在しないと判断した場合はステップＳ１２１０に進む。
【０２３８】
ステップＳ１２１０では、ＶＭＭ１０は、変換コードを格納するためのメモリ領域のメモリ容量に必要な空き容量が存在するか否かを判断する。メモリ領域に必要な空き容量が存在しないと判断した場合は、ステップＳ１３４０に移行する。必要な空き容量が存在すると判断した場合は、ステップＳ１３５０に進む。
【０２３９】
ステップＳ１３４０では、ＶＭＭ１０は、変換コードＬＲＵリスト３６を参照して、未使用時間の最も長い変換コードを含む要素を一つ選択する。そして、選択された要素に含まれる変換コードに属するページの物理アドレスを無効化することによって、変換コードをメモリ領域から破棄する。さらに、選択された要素に含まれる変換コードの情報を、変換コードＬＲＵリスト３６から削除する。そして、ステップＳ１２１０に戻る。すなわち、必要な空き容量が確保できるまで、未使用時間の最も長い変換コードをメモリ領域から破棄する。
【０２４０】
一方、ステップＳ１３５０では、ＶＭＭ１０は、次に実行されるゲストコード以降のコードのうち、デコード可能なコードに対応する変換コードを生成し、生成されたコードを保存する。なお、変換コードはシャドウコードと昇格コードとで構成される。シャドウコードはゲストコードに対応して生成される。また、シャドウコードはゲストコードに含ま

れるＨＷ操作コードの代わりに、昇格コードを含む。昇格コードはエミュレーションモジュールを呼び出すためのＣＰＬ減少分岐命令を含む。なお、本ステップにおいて生成した変換コードを、検索の結果とする。

【０２４１】

次に、ＶＭＭ１０は、生成した変換コードの情報を変換コード管理表３５に格納する。さらに、変換コードＬＲＵリスト３６の末尾に生成した変換コードの情報を追加する（Ｓ１３６０）。

【０２４２】

次に、ＶＭＭ１０は、ゲストコードの書き換えを検出するために、当該ゲストコードを含むページにライトプロテクトを設定する（Ｓ１３７０）。

【０２４３】

一方、ステップＳ１３２０では、ＶＭＭ１０は、次に実行すべき変換コードのアドレスを、検索されたエントリから取得する。取得された変換コードを検索の結果とする。

【０２４４】

次に、ＶＭＭ１０は、当該エントリに対応する変換コードの組を最新に変更することによって、変換コードＬＲＵリスト３６を更新する（Ｓ１３３０）。

【０２４５】

なお、図２６のステップＳ２７０のコード変換の設定更新処理は、前述の第１の実施の形態の図１４のフローチャートと同一である。

【０２４６】

以上のように、本発明の第３の実施の形態の仮想計算機システムではゲスト２０によってＨＷ操作を含むゲストコードの実行が要求されたときに、ＶＭＭ１０は、当該ＨＷ操作の実行間隔に応じて処理方法を変える。すなわち、当該ＨＷ操作を二つの部分に分け、実行間隔が小さい部分（すなわち、実行頻度が高い）に実行効率の高いコード変換方式を適用し、実行間隔が大きい（すなわち、実行頻度が高い）部分にメモリ消費量の少ないＶＴ方式を適用する。このようにすることによって、高性能と省メモリとを両立することが可能なエミュレーション処理が可能となる。

【産業上の利用可能性】

【０２４７】

本発明は、ＣＰＵとして、ＶＴ－ｉ機能を搭載したＩＰＦやＶＴ－ｘ機能を搭載したｘ８６を備えた仮想計算機システムに適用することができる。

【図面の簡単な説明】

【０２４８】

【図１】本発明の第１の実施の形態の、仮想計算機システムを動作させる物理計算機のブロック図である。

【図２】本発明の第１の実施の形態の、仮想計算機システムのソフトウェアとハードウェアの要部を示すブロック図である。

【図３】本発明の第１及び第３の実施の形態の、コード変換の設定表の説明図である。

【図４】本発明の第１の実施の形態の、頻度判断モジュールのブロック図である。

【図５】本発明の第１の実施の形態の、ＨＷ操作頻度表の説明図である。

【図６】本発明の第１の実施の形態の、変換コード管理表の説明図である。

【図７】本発明の第１の実施の形態の、変換コードＬＲＵリストの説明図である。

【図８】本発明の第１の実施の形態の、ＶＭＭの動作のフローチャートである。

【図９】本発明の第１の実施の形態の、エミュレーション処理を示すフローチャートである。

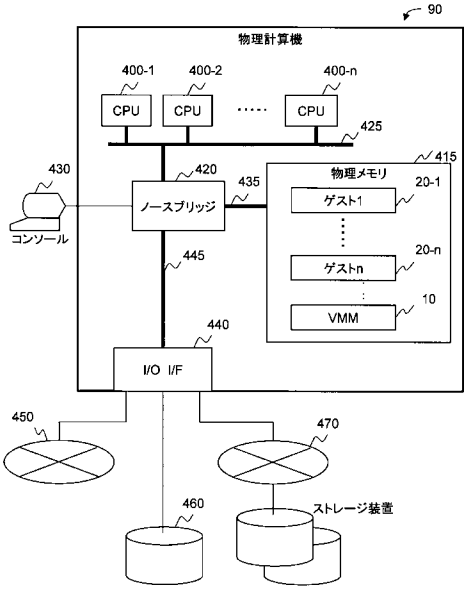
【図１０】本発明の第１の実施の形態の、変換コードの無効化処理のフローチャートである。

【図１１】本発明の第１の実施の形態の、ハードウェア操作の頻度判断処理のフローチャートである。

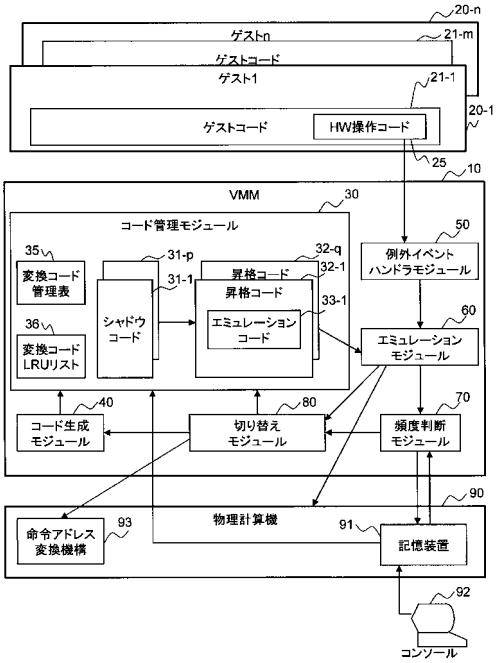
【図１２】本発明の第１の実施の形態の、エミュレーション後の次処理選択処理のフロー

チャートである。
【図１３】本発明の第１の実施の形態の、コード変換方式の適用処理のフローチャートである。
【図１４】本発明の第１の実施の形態の、方式切り替え条件の更新処理のフローチャートである。
【図１５】本発明の第２の実施の形態の、コード変換の設定表の説明図である。
【図１６】本発明の第２の実施の形態の、ＨＷ操作頻度表の説明図である。
【図１７】本発明の第２の実施の形態の、変換コードの無効化処理のフローチャートである。
【図１８】本発明の第２の実施の形態の、ＨＷ操作の頻度判断処理のフローチャートである。
【図１９】本発明の第３の実施の形態の、仮想計算機システムのソフトウェアとハードウェアの要部のブロック図である。
【図２０】本発明の第３の実施の形態の、頻度判断モジュールのブロック図である。
【図２１】本発明の第３の実施の形態の、ＨＷ操作頻度表の説明図である。
【図２２】本発明の第３の実施の形態の、ＨＷ操作履歴表の説明図である。
【図２３】本発明の第３の実施の形態の、切り替えモジュールのブロック図である。
【図２４】本発明の第３の実施の形態の、動作方式表の説明図である。
【図２５】本発明の第３の実施の形態の、変換コード管理表の説明図である。
【図２６】本発明の第３の実施の形態の、ＶＭＭの動作のフローチャートである。
【図２７】本発明の第３の実施の形態の、エミュレーション処理のフローチャートである。
【図２８】本発明の第３の実施の形態の、変換コードの無効化処理のフローチャートである。
【図２９】本発明の第３の実施の形態の、ハードウェア操作の頻度判断処理のフローチャートである。
【図３０】本発明の第３の実施の形態の、エミュレーション後の次処理選択処理のフローチャートである。
【図３１】本発明の第３の実施の形態の、変換コードの検索処理のフローチャートである。
【符号の説明】
【０２４９】
１０　ＶＭＭ
２０　ゲスト
３０　コード変換モジュール
３１　シャドウコード
３２　昇格コード
３５　変換コード管理表
４０　コード生成モジュール
５０　例外イベントハンドラモジュール
６０　エミュレーションモジュール
７０　頻度判断モジュール
８０　切り替えモジュール
９０　物理計算機
９１　記憶装置
９２　コンソール

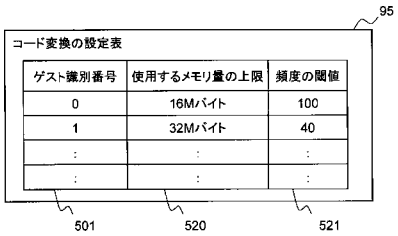
【図１】


90
物理計算機
400-1
400-2
400-n
CPU
CPU
CPU
425
415
物理メモリ
20-1
ゲスト1
20-n
ゲストn
10
VMM
430
コンソール
420
ノースブリッジ
435
445
440
I/O I/F
450
470
460
ストレージ装置

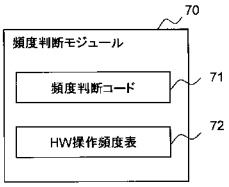
【図２】


20-n
ゲストn
21-m
ゲストコード
ゲスト1
21-1
ゲストコード
HW操作コード
20-1
25
10
VMM
30
コード管理モジュール
35
変換コード管理表
36
変換コードLRUリスト
31-p
31-1
シャドウコード
32-q
昇格コード
32-1
昇格コード
33-1
エミュレーションコード
50
例外イベントハンドラモジュール
60
エミュレーションモジュール
40
コード生成モジュール
80
切り替えモジュール
70
頻度判断モジュール
90
物理計算機
93
命令アドレス変換機構
91
記憶装置
92
コンソール

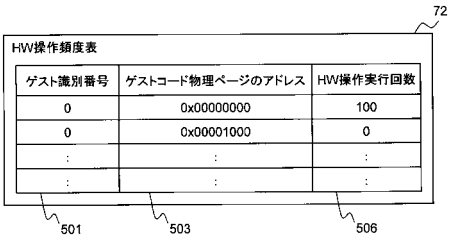
【図３】


95
コード変換の設定表
ゲスト識別番号
使用するメモリ量の上限
頻度の閾値
0
16Mバイト
100
1
32Mバイト
40
501
520
521

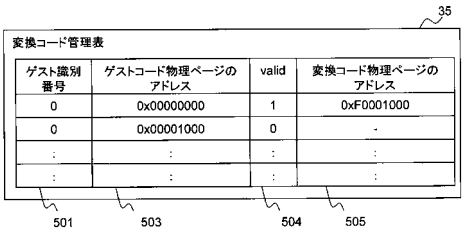
【図４】


70
頻度判断モジュール
頻度判断コード
71
HW操作頻度表
72

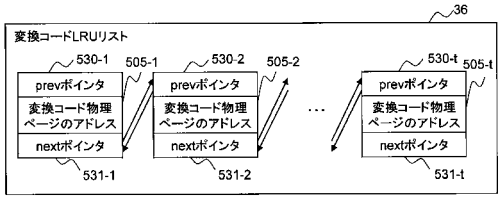
【図５】


72
HW操作頻度表
ゲスト識別番号
ゲストコード物理ページのアドレス
HW操作実行回数
0
0x00000000
100
0
0x00001000
0
501
503
506

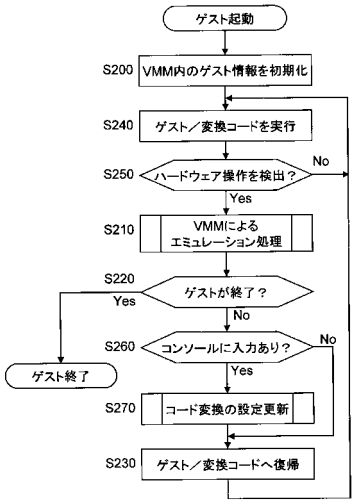
【図６】


35
変換コード管理表
ゲスト識別番号
ゲストコード物理ページのアドレス
valid
変換コード物理ページのアドレス
0
0x00000000
1
0xF0001000
0
0x00001000
0
-
501
503
504
505

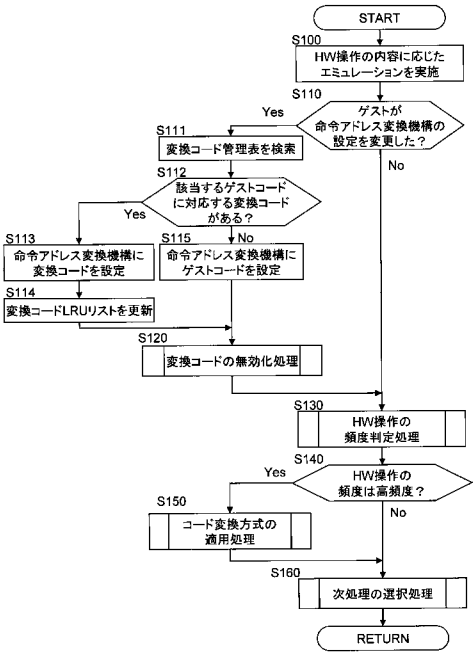
【図７】


36
変換コードLRUリスト
530-1
505-1
530-2
505-2
530-t
505-t
prevポインタ
変換コード物理ページのアドレス
nextポインタ
531-1
531-2
531-t

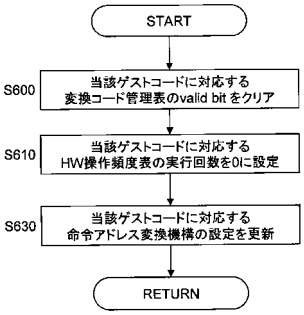
【図８】


ゲスト起動
S200
VMM内のゲスト情報を初期化
S240
ゲスト／変換コードを実行
S250
ハードウェア操作を検出？
No
Yes
S210
VMMによるエミュレーション処理
S220
ゲストが終了？
Yes
No
S260
コンソールに入力あり？
No
Yes
ゲスト終了
S270
コード変換の設定更新
S230
ゲスト／変換コードへ復帰

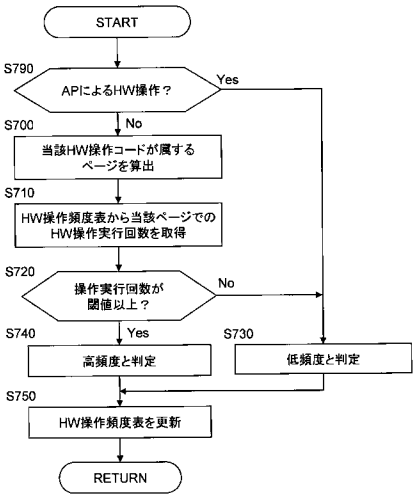
【図９】



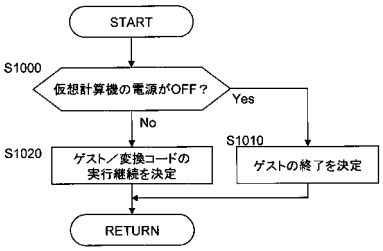
【図１０】



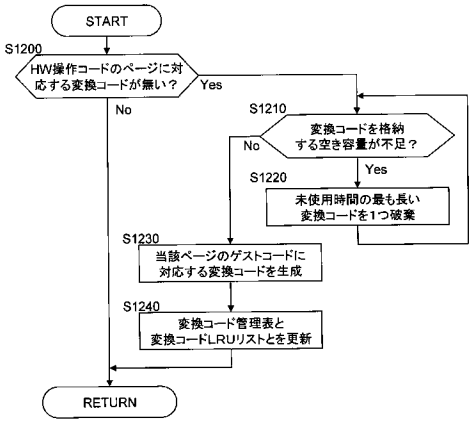
【図１１】



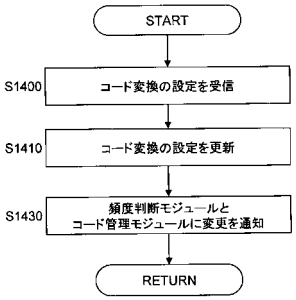
【図１２】



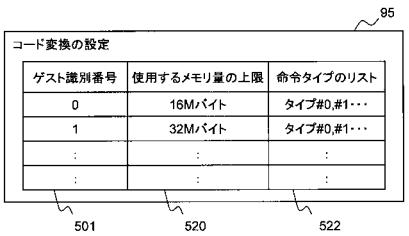
【図１３】



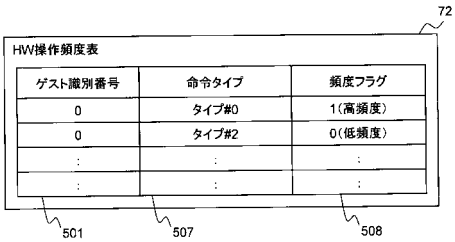
【図１４】



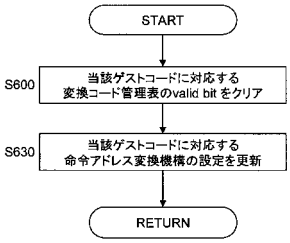
【図１５】



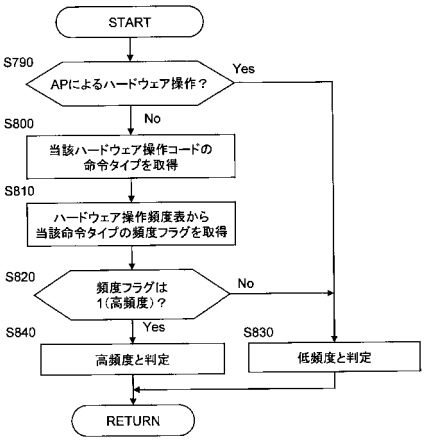
【図１６】



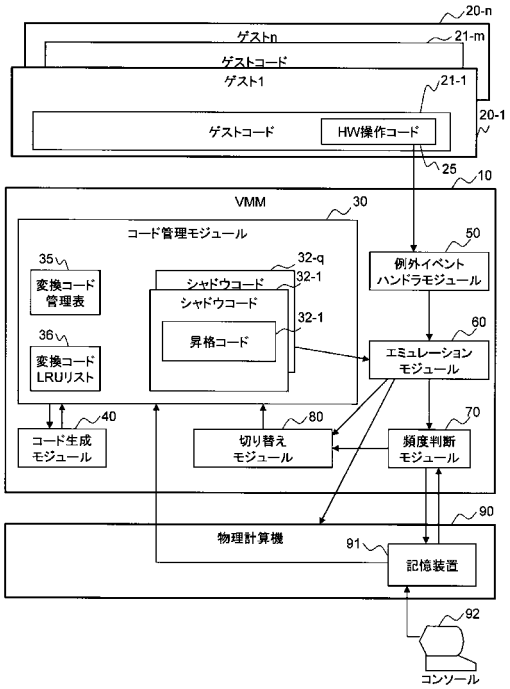
【図１７】



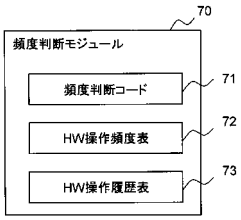
【図１８】



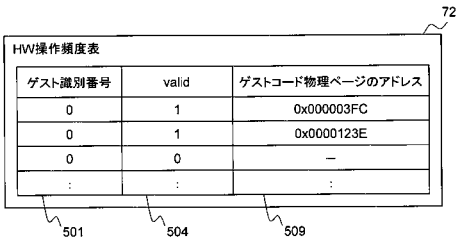
【図１９】



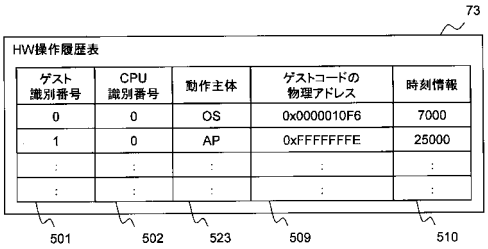
【図２０】



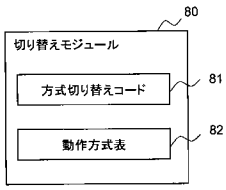
【図２１】



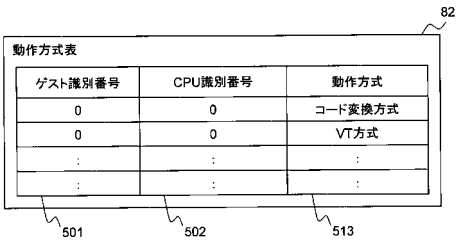
【図２２】



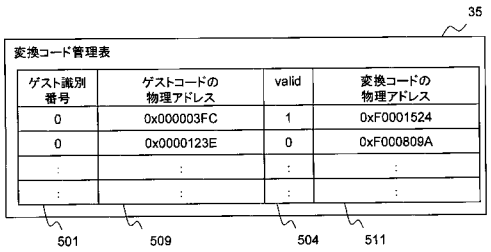
【図２３】



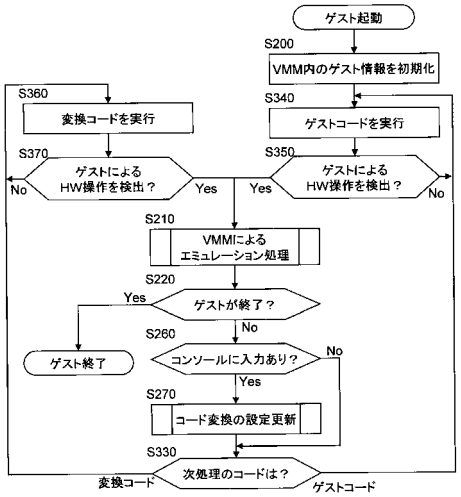
【図２４】



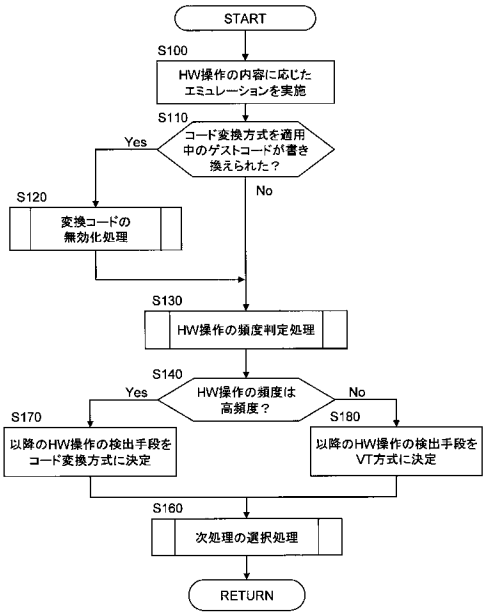
【図２５】



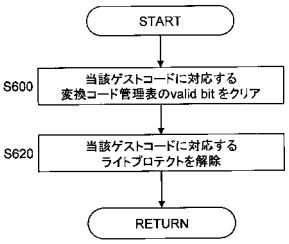
【図２６】



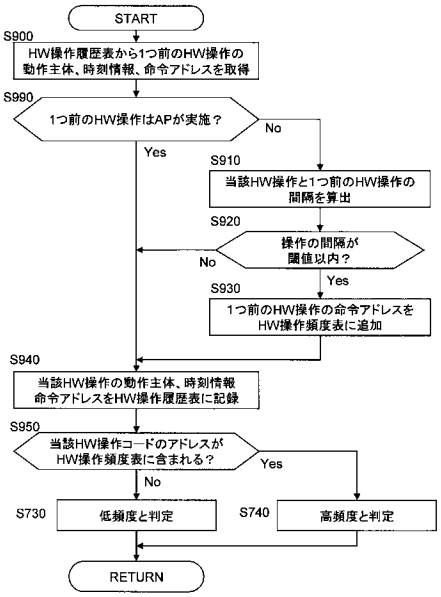
【図２７】



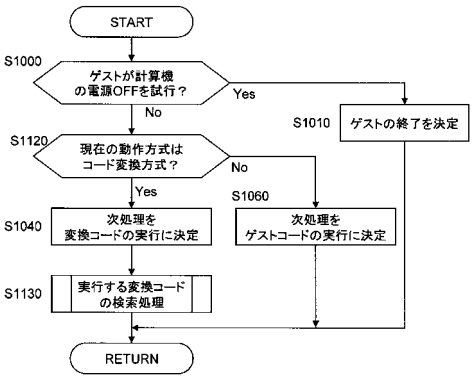
【図２８】



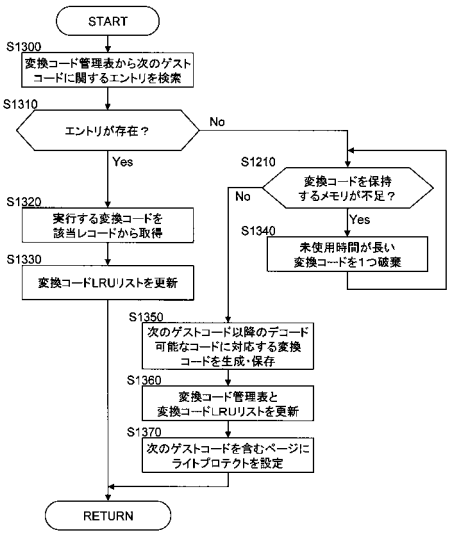
【図２９】


START
S900
HW操作履歴表から1つ前のHW操作の動作主体、時刻情報、命令アドレスを取得
S990
1つ前のHW操作はAPが実施？
No
Yes
S910
当該HW操作と1つ前のHW操作の間隔を算出
S920
操作の間隔が閾値以内？
No
Yes
S930
1つ前のHW操作の命令アドレスをHW操作頻度表に追加
S940
当該HW操作の動作主体、時刻情報命令アドレスをHW操作履歴表に記録
S950
当該HW操作コードのアドレスがHW操作頻度表に含まれる？
Yes
No
S730
低頻度と判定
S740
高頻度と判定
RETURN

【図３０】


START
S1000
ゲストが計算機の電源OFFを試行？
Yes
No
S1010
ゲストの終了を決定
S1120
現在の動作方式はコード変換方式？
No
Yes
S1060
次処理をゲストコードの実行に決定
S1040
次処理を変換コードの実行に決定
S1130
実行する変換コードの検索処理
RETURN

【図３１】


START
S1300
変換コード管理表から次のゲストコードに関するエントリを検索
S1310
エントリが存在？
No
Yes
S1210
変換コードを保持するメモリが不足？
No
Yes
S1320
実行する変換コードを該当レコードから取得
S1340
未使用時間が長い変換コードを1つ破棄
S1330
変換コードLRUリストを更新
S1350
次のゲストコード以降のデコード可能なコードに対応する変換コードを生成・保存
S1360
変換コード管理表と変換コードLRUリストを更新
S1370
次のゲストコードを含むページにライトプロテクトを設定
RETURN

フロントページの続き

(72)発明者　對馬　雄次
東京都国分寺市東恋ヶ窪一丁目２８０番地　株式会社日立製作所　中央研究所内

審査官　吉田　美彦

(56)参考文献　特開平０１－２４３１５５（ＪＰ，Ａ）
特開昭６２－０３１４３７（ＪＰ，Ａ）
特開２０００－２２２２２１（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　　９／４６