(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2013-93045
(P2013-93045A)
(43)公開日　平成25年5月16日(2013.5.16)

(51)Int.Cl.		F I		テーマコード（参考）
G06F 9/48	(2006.01)	G06F	9/46	311C

審査請求 有　請求項の数 12　ＯＬ　（全 17 頁）

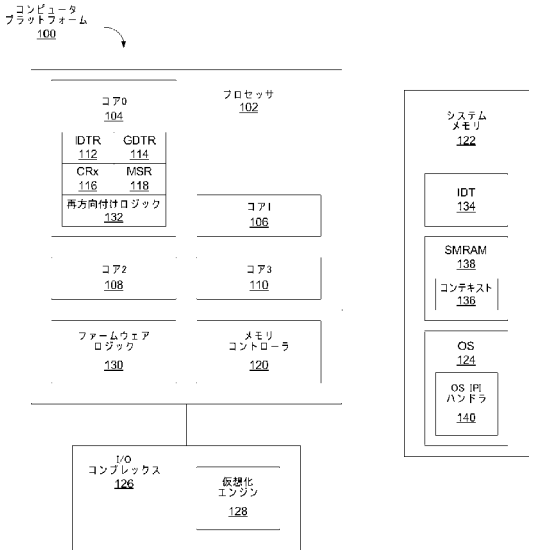
(21)出願番号　特願2013-3917(P2013-3917)
(22)出願日　平成25年1月11日(2013.1.11)
(62)分割の表示　特願2010-85510(P2010-85510)の分割
原出願日　平成22年4月1日(2010.4.1)
(31)優先権主張番号　12/384,725
(32)優先日　平成21年4月8日(2009.4.8)
(33)優先権主張国　米国(US)

(71)出願人　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　ジマー、ビンセント　ジェイ.
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

最終頁に続く

(54)【発明の名称】システム管理モードにおけるプロセッサ間割り込みの再方向付け

(57)【要約】
【課題】マルチコア環境のシステム管理モードにおけるプロセッサ間の割り込み（ＩＰＩ）等のアクションに応答する。
【解決手段】幾つかのプロセッサコアのうち第１のプロセッサコアがシステム管理モードに入り、第１のプロセッサコアとは異なる他のプロセッサコアのうち少なくとも１つは動作を維持してシステム管理モードには入らない。次いで、第１のプロセッサコアがシステム管理モード中にプロセッサ間割り込みに応答する。ＩＰＩを受けると、対象となるコアのシステムのコンテキストをセーブしておき、制御をＩＰＩ用のオペレーティングシステム（ＯＳ）ハンドラ１４０に渡す。ひとたびＯＳ　ＩＰＩが完了すると、システムのコンテキストを復元して、制御を、さらなる処理のためにＳＭＩハンドラに戻す。
【選択図】図１



【特許請求の範囲】
【請求項１】
複数のプロセッサコアのうち第１のプロセッサコアがシステム管理モードに入り、前記複数のプロセッサコアのうちの、前記第１のプロセッサコアとは異なる１以上のプロセッサコアは動作を維持して前記システム管理モードには入らない段階と、
前記第１のプロセッサコアが、前記システム管理モード中にプロセッサ間割り込みに応答する段階と、
前記第１のプロセッサコアが前記システム管理モードに入った後で、前記第１のプロセッサコアによる割り込み処理を許可するべく割り込み記述子テーブルをセーブする段階と、
前記第１のプロセッサコアに対するプロセッサ間割り込みをイネーブルする段階と、
前記第１のプロセッサコアがプロセッサ間割り込みを受信した後で、
前記第１のプロセッサコアの現在のシステムコンテキストをメモリにセーブする段階と、
前記第１のプロセッサコアに対して、オペレーティングシステムのプロセッサ間割り込みハンドラに関する新たなシステムコンテキストを設定する段階と、
前記第１のプロセッサコアが、処理位置を前記プロセッサ間割り込みハンドラの位置にジャンプさせる段階と、
前記第１のプロセッサコアが、前記オペレーティングシステムのプロセッサ間割り込みハンドラを実行する段階と、
前記オペレーティングシステムのプロセッサ間割り込みハンドラを実行した後で、
前記第１のプロセッサコアの処理位置をジャンプ前の位置に戻す段階と、
前記第１のプロセッサコアの前記現在のシステムコンテキストを前記メモリから復元する段階と、
前記第１のプロセッサコアを、前記システム管理モードから抜けさせる段階と、
を備える方法。
【請求項２】
前記第１のプロセッサコアの前記システムコンテキストは、前記第１のプロセッサコアの制御レジスタ、前記割り込み記述子テーブル、命令ポインタ、グローバル記述子テーブルレジスタを含む、
請求項１に記載の方法。
【請求項３】
前記オペレーティングシステムのプロセッサ間割り込みハンドラを制御するオペレーティングシステムが制御を獲得して前記ハンドラを実行する前に、システム管理割り込み転送モニタにより前記オペレーティングシステムをセキュアに計測する段階と、
前記計測時に、前記システム管理割り込み転送モニタが前記オペレーティングシステムに制御を送る段階とをさらに備える請求項１または２に記載の方法。
【請求項４】
前記プロセッサ間割り込みに呼応した前記システムコンテキストの切り替えにおいて、１以上の第１のプロセッサコアの制御レジスタと、一の第１のプロセッサコアのグローバル記述子テーブルレジスタとを含むように、前記第１のプロセッサコアのマシン固有レジスタにビットを設定する段階をさらに備える請求項１から３のいずれか１項に記載の方法。
【請求項５】
コンピュータシステムの第１のプロセッサコアであって、
システム管理モードに入り、
前記システム管理モード中にプロセッサ間割り込みに応答し、
前記コンピュータシステムの中の１以上のさらなるプロセッサコアは動作を維持して前記システム管理モードには入らず、
前記システム管理モードに入った後で、前記第１のプロセッサコアによる割り込み処理

を許可するべく割り込み記述子テーブルをセーブし、
前記プロセッサ間割り込みをイネーブルし、
前記プロセッサ間割り込みを受信した後で、
現在のシステムコンテキストをメモリにセーブし、
オペレーティングシステムのプロセッサ間割り込みハンドラに関する新たなシステムコンテキストを設定し、
処理位置を前記プロセッサ間割り込みハンドラの位置にジャンプさせ、
前記オペレーティングシステムのプロセッサ間割り込みハンドラを実行し、
前記オペレーティングシステムのプロセッサ間割り込みハンドラを実行した後で、
処理位置をジャンプ前の位置に戻し、
前記現在のシステムコンテキストを前記メモリから復元し、
前記システム管理モードから抜けさせる、
第１のプロセッサコア。
【請求項６】
前記システムコンテキストは、前記第１のプロセッサコアの制御レジスタ、前記割り込み記述子テーブル、命令ポインタ、グローバル記述子テーブルレジスタを含む、
請求項５に記載の第１のプロセッサコア。
【請求項７】
前記オペレーティングシステムのプロセッサ間割り込みハンドラを制御するオペレーティングシステムが制御を獲得して前記ハンドラを実行する前に、システム管理割り込み転送モニタにより前記オペレーティングシステムをセキュアに計測し、
前記計測時に、前記システム管理割り込み転送モニタが前記オペレーティングシステムに制御を送る請求項５または６に記載の第１のプロセッサコア。
【請求項８】
前記プロセッサ間割り込みに呼応した前記システムコンテキストの切り替えにおいて、１以上の第１のプロセッサコアの制御レジスタと、一の第１のプロセッサコアのグローバル記述子テーブルレジスタとを含むように、前記第１のプロセッサコアのマシン固有レジスタにビットを設定する請求項５から７のいずれか１項に記載の第１のプロセッサコア。
【請求項９】
システム管理モードに入り、前記システム管理モード中にプロセッサ間割り込みに応答する第１のプロセッサコアと、
動作を維持して前記システム管理モードには入らない１以上のさらなるプロセッサコアとを備え、
前記第１のプロセッサコアは、
前記システム管理モードに入った後で、前記第１のプロセッサコアによる割り込み処理を許可するべく割り込み記述子テーブルをセーブし、
前記プロセッサ間割り込みをイネーブルし、
前記プロセッサ間割り込みを受信した後で、
現在のシステムコンテキストをメモリにセーブし、
オペレーティングシステムのプロセッサ間割り込みハンドラに関する新たなシステムコンテキストを設定し、
処理位置を前記プロセッサ間割り込みハンドラの位置にジャンプさせ、
前記オペレーティングシステムのプロセッサ間割り込みハンドラを実行し、
前記オペレーティングシステムのプロセッサ間割り込みハンドラを実行した後で、
処理位置をジャンプ前の位置に戻し、
前記現在のシステムコンテキストを前記メモリから復元し、
前記システム管理モードから抜けさせる
システム。
【請求項１０】
前記システムコンテキストは、前記第１のプロセッサコアの制御レジスタ、前記割り込

み記述子テーブル、命令ポインタ、グローバル記述子テーブルレジスタを含む、
請求項９に記載のシステム。
【請求項１１】
前記第１のプロセッサコアはさらに、前記オペレーティングシステムのプロセッサ間割り込みハンドラを制御するオペレーティングシステムが制御を獲得して前記ハンドラを実行する前に、システム管理割り込み転送モニタにより前記オペレーティングシステムをセキュアに計測し、
前記計測時に、前記システム管理割り込み転送モニタが前記オペレーティングシステムに制御を送る請求項９または１０に記載のシステム。
【請求項１２】
前記第１のプロセッサコアはさらに、前記プロセッサ間割り込みに呼応した前記システムコンテキストの切り替えにおいて、１以上の第１のプロセッサコアの制御レジスタと、一の第１のプロセッサコアのグローバル記述子テーブルレジスタとを含むように、前記第１のプロセッサコアのマシン固有レジスタにビットを設定する請求項９から１１のいずれか１項に記載のシステム。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、システム管理モード中のプロセッサ間の割り込み処理に係る。
【背景技術】
【０００２】
ＳＭＭ（システム管理モード）は３２または６４ビットのインテル（登録商標）マイクロプロセッサ等の中央処理装置の根幹部である。例えばＡＣＰＩ（電力制御インターフェース）のイネーブルおよびディセーブル、デジタル熱センサの利用、メモリのホットプラグ等の、重要なコンピュータのプラットフォームアクティビティの多くが、通常、相手先ブランド供給業者（ＯＥＭ）のファームウェアがコンピュータプラットフォームに提供するＳＭＭコードで実行される。コアは、ＳＭＩ（システム管理割り込み）を受けて、ＳＭＭに入る。
【０００３】
現在のコンピュータプラットフォームに常駐する中央プロセッサは、幾つかのコアを有していることが多い（つまり、プロセッサ１つについて２、４、８、１６等のコア）。一般的にマルチコアの中央プロセッサの性能はシングルコアのものよりも優れているかもしれないが、マルチコア環境における効率的な動作には、幾つかの技術的課題が残されている。現在のところ、プラットフォームのファームウェアは、オペレーティングシステム（ＯＳ）ソフトウェアモデルに適合すべくＳＭＭ用に全てのＣＰＵコアを同期させている、というのもＯＳカーネルでは、１つのコアが見当たらないことも、または、プロセッサ間の割り込み（ＩＰＩ）等のアクションに応答しないことも許されないからである。
【０００４】
ＰＩ（プラットフォーム初期化）１．１（www.uefi.org）仕様では、ＵＥＦＩ（Unified Extensible Firmware Interface）ベースのファームウェアのＳＭＭアーキテクチャが定義されている。しかし、現在のＵＥＦＩファームウェアおよびレガシーバイオス（ＢＩＯＳ）いずれにおいても、ＳＭＭ環境への割り込みはディセーブルされている。
【図面の簡単な説明】
【０００５】
本発明を例示するが、図面は限定を意図しておらず、同様の参照番号の部材は類似したものを示すものとする。
【０００６】
【図１】ＳＭＭにおける応答コアを対象としたＩＰＩの処理機能を有するコンピュータプラットフォームおよびプロセッサの一実施形態を示す。
【０００７】

【図２】処理コアがＳＭＭ中に行うタスクがない場合に、ＳＭＭから処理コアを退出させる処理の一実施形態のフロー図である。
【０００８】
【図３】シングルコアをＳＭＭに入らせる処理の別の実施形態を示す。
【０００９】
【図４】ＳＭＭ内の処理コアに対象を絞ってＩＰＩを再方向付けする処理の一実施形態のフロー図である。
【発明を実施するための形態】
【００１０】
システム管理モード中にプロセッサ間割り込みを再方向付けるデバイス、システム、および方法の実施形態を記載する。
【００１１】
コンピュータプラットフォームの中央プロセッサ内の処理コアは、ＳＭＩ（システム管理割り込み）を受けると、ＳＭＭ（システム管理モード）に入ることができる。コアはＳＭＭに入ることで、そのモード固有の１以上のタスクを完了させる。現在のファームウェアによる解決法を有するコンピュータプラットフォームでは、ＳＭＭにあるコアがＩＰＩ（プロセッサ間割り込み）に応答しない。さらに、現在のコンピュータプラットフォームでは、プラットフォームの処理コアがＳＭＩを受けてＳＭＭに入ると、コンピュータプラットフォーム上のファームウェアは、全てのコアを同期してＳＭＭに入らせる。従って、現在の技術では、ＳＭＭへの入退出は、全てのコアを対象とする同期処理である。これにより、コア間の処理割り込み量が不当に増加していることが懸念される。この割り込みは特に、オーディオ／ビデオストリーミング等のある種のサービス品質（ＱｏＳ）アクティビティで顕著であり、このようなケースでは、ＳＭＭ内のコア数が増える、および／または、ＳＭＭ占有時間が長くなる、といった場合にジッタが生じ始めることもある。
【００１２】
多くの実施形態では、「全てのコア間のＳＭＭ同期」の修正モデルを保証している。例えば、ＳＭＩが発行されるが、必要となるＳＭＭ処理がマルチコアプロセッサ内の単一のコアだけに固有であるような場合には、通常、対象とされている単一のコアだけをＳＭＭに入らせ、残りのコアには標準動作を続行させることでプロセッサの効率が上がると考えられる。現在のＩＰＩにまつわる問題は、この修正された「対象となるコアのＳＭＭ」の入退出モデルを実装するときに起こる。単純に、１つ残され、ＳＭＭに入らなかった１つのコアが、ＳＭＭに入った１以上のコアに対してＩＰＩを発行することがある。しかし、ＩＰＩがＳＭＭに入っていないコアからＳＭＭに入っているコアに送られても、ＳＭＭ内のコアは不在で応答しないように見えるので、該ＩＰＩが受信されないことになる。
【００１３】
従って、多くの実施形態では、対象となるコアの制御フローは、標準ＳＭＩハンドラのものを外れて、ＳＭＭ中であってもＩＰＩに応答するようにする。特に、ＩＰＩを受けると、対象となるコアのシステムのコンテキストをセーブしておき、制御をＩＰＩ用のオペレーティングシステム（ＯＳ）ハンドラに渡す。ひとたびＯＳ　ＩＰＩが完了すると、システムのコンテキストを復元して、制御を、さらなる処理のためにＳＭＩハンドラに戻す。
【００１４】
以下の記載および請求項では、「含む(include)」および「備える（comprise）」、並びにそれらの派生物が利用される場合があるが、これらは互いに同義語として取り扱われることを意図している。さらに、以下の記載および請求項では、「連結された（coupled）」および「接続された（connected）」、並びにそれらの派生物が利用される場合がある。これらの用語は互いに同義語を意図していないことを理解されたい。特定の実施形態では、「接続された」は、２以上の部材が直接物理的または電気的接触関係にあることを示す場合に用いられる。「連結された」も、２以上の部材が直接物理的または電気的接触関係にあることを示していてもよい。しかし、「連結された」は、２以上の部材が直接接

触していなくてもよく、互いに協働または相互作用することを意味する場合もある。

【００１５】

図１は、ＳＭＭにおける応答コアを対象としたＩＰＩの処理機能を有するコンピュータプラットフォームおよびプロセッサの一実施形態を示す。コンピュータプラットフォーム１００は、プロセッサ（例えばプロセッサ１０２）を含みうる。図示されていない他の実施形態では、コンピュータシステム１００は、２以上のプロセッサを含んでもよい。プロセッサ１０２は、インテル（登録商標）ベースの中央処理装置（ＣＰＵ）であっても、別のブランドのＣＰＵであってもよい。他の実施形態では、プロセッサ１０２は、１以上のコアを有してよい。例えば図１では、４つのコア（コア０（１０４）、コア１（１０６）、コア２（１０８）、コア３（１１０））を有するプロセッサ１０２が示されている。しかし、図示されていない他の実施形態では、プロセッサ１０２は、２、８、１６、またはこれより多くの数のコアを含んでもよい。

【００１６】

各プロセッサコアは、実行ユニットおよび命令撤回ユニット等の、幾つかの内部動作ユニットを含んでよい。さらに、各コアは、レジスタおよびキャッシュ等の、データを保存するのに利用される内部メモリ位置を含んでよい。例えば、インテル（登録商標）ベースのマイクロプロセッサアーキテクチャでは、各コアが割り込み記述子テーブル（ＩＤＴ）レジスタ（ＩＤＴＲ１１２）、グローバル記述子テーブル（ＧＤＴ）レジスタ（ＧＤＴＲ１１４）、１以上の制御レジスタ（ＣＲｘ１１６）、１以上のモデル固有のレジスタ（ＭＳＲ１１８）等を含むことができる。

【００１７】

ＩＤＴＲ１１２は、ＩＤＴのアドレス位置を記憶する。ＩＤＴは、汎用ｘ８６コンピュータアーキテクチャにおいて割り込みベクトルテーブルを実装するのに利用されるデータ構造である。ＩＤＴは、コアが、コアが受ける割り込みおよび例外への正しい応答を決定するのに利用される。

【００１８】

ＧＤＴＲ１１４は、ＧＤＴのアドレス位置を記憶する。ＧＤＴは、プログラムを実行する際の様々なメモリ領域の特性を定義するのに利用されるデータ構造である（例えば個々のメモリセグメントのベースアドレス、サイズ、アクセス特権等）。

【００１９】

各ＣＲｘ１１６（例えばＣＲ０、ＣＲ１等）は、コア０（１０４）については、該コアの汎用行動を変更する、または制御するプロセッサレジスタである。制御レジスタが制御しうるタスクのなかには、割り込み制御、メモリページング制御、およびアドレスモード制御等が含まれる。

【００２０】

各ＭＳＲ１１８（幾つかのＭＳＲがあることもあり、各々が異なる情報を記憶している）は、ＯＳ等のシステムソフトウェアに対して、特定のプロセッサの実装に関しており他のプロセッサの実装に関するものではないフィーチャを提供する。ＭＳＲ内の情報により、ＯＳは、コアまたはプロセッサが一般的に有する機能について知ることができる。

【００２１】

コア０（１０４）に示す記憶レジスタは、特定のコアに位置するレジスタのサンプルに過ぎない。現実には、コア０（１０４）はこれ以外に、多くの記憶レジスタおよび１以上のメモリキャッシュを含みうるが、このようなさらなる記憶位置は図１に示さない。他の実施形態では、プロセッサ１０２はインテル（登録商標）プロセッサではなくてもよく、代わりに、類似した機能を有する他の記憶位置があってもよい。

【００２２】

プロセッサ１０２は、メモリコントローラ１２０を介してメモリサブシステムに連結される。図１はプロセッサ１０２に集積されたメモリコントローラ１２０を示しているが、図示されていない他の実施形態では、メモリコントローラは、ブリッジデバイスあるいはコンピュータシステム内のプロセッサ１０２とは離間している他の集積回路内に集積され

ていてもよい。メモリサブシステムは、プロセッサが実行する命令を記憶するシステムメモリ１２２を含む。メモリサブシステムのメモリデバイスは、ダブルデータレート（ＤＤＲ）同期ＤＲＡＭ等の任意の種類の揮発性ダイナミックランダムアクセスメモリ（ＤＲＡＭ）、および／または、フラッシュメモリ（登録商標）等の任意の種類の不揮発性メモリであってよい。プロセッサは、プロセッサメモリインタフェースによりメモリに連結され、プロセッサメモリインタフェースは、データ、アドレス、制御、およびその他の情報を、プロセッサ（１または複数）およびメモリ間で交換させる個々のラインを含むリンク（つまり、インターコネクト／バス）であってよい。

【００２３】

ホストオペレーティングシステム（ＯＳ）１２４は、コンピュータプラットフォーム１００が、プラットフォームおよびプラットフォームに取り付けられた周辺機器に対して汎用動作制御を提供するよう動作可能な間、該コンピュータプラットフォーム１００のメモリにロードされるオペレーティングシステムを表す。ホストＯＳ１２４は、マイクロソフト（登録商標）、ウィンドウズ（登録商標）、ＵＮＩＸ（登録商標）、ＬＩＮＵＸ（登録商標）、またはその他の機能ＯＳであってよい。ホストＯＳ１２４は、１以上のプログラム、サービス、またはエージェントがその内部で動作できる環境を提供する。多くの実施形態では、１以上のソフトウェアアプリケーションを、ホストＯＳ１２４の上部で動作させうる。アプリケーションは、システムリソースの利用中に１以上のタスクを実行する任意の種類のソフトウェアアプリケーションであってよい。

【００２４】

コンピュータプラットフォームは、さらに、入出力（Ｉ／Ｏ）ロジックコンプレックス１２６を含んでよい。Ｉ／Ｏロジックコンプレックス１２６は、コンピュータプラットフォーム１００内のＩ／Ｏサブシステムの一部を管理する１以上の集積コントローラを含みうる。例えば、Ｉ／Ｏコンプレックス１２６は、プロセッサと、コンピュータプラットフォーム１００にプラグインされうる１以上のユニバーサルシリアルバス（ＵＳＢ）デバイスとの間の情報の流れを制御する１以上の集積ＵＳＢホストコントローラ（不図示）を含みうる。

【００２５】

多くの実施形態では、Ｉ／Ｏコンプレックスは、仮想化エンジン１２８を含む。仮想化エンジン１２８は、コンピュータプラットフォーム１００を多数の仮想マシン（ＶＭ）に分割させるロジックを含みうる。例えば、２つのＶＭがコンピュータプラットフォーム１００上で実行されているとすると、各々は、別このシステムリソースを割り当てられている（例えばシステムメモリ）。１つのＶＭは、ホストＯＳ１２４にリソースを提供する際の優先度の低いＶＭとして実装されてよい。２つ目のＶＭは、コンピュータプラットフォーム１００に対して遠隔情報技術（ＩＴ）アクセスを実現すべく、システム管理者にある種の優先リソースを提供する優先度の高いＶＭとして実装されてよい。別の例ではさらに別のＶＭを実装することもできる（例えば、コンピュータプラットフォーム１００に連結されたＶｏＩＰ（Voice-over-Internet-Protocol）電話機専用のリソースを提供するＶＭ等）。

【００２６】

仮想化エンジン１２８は、さらに、強力なハードウェアベースのセキュリティチェック処理をホストＯＳ１２４に実装することができる。例えば、優先度の低いホストＯＳベースのＶＭの整合性は、最初の起動処理中または、より優先度の高いセキュリティＶＭによるプラットフォーム動作中の任意の時点において、セキュアに計測することができる。これらセキュリティ計測値により、コンピュータプラットフォーム１００のエンドユーザは、ホストＯＳ１２４およびホストＯＳの上部で実行されているソフトウェアアプリケーションいずれもが危険に曝されていないことを確認することができうる。

【００２７】

多くの実施形態では、プロセッサ１０２は、ファームウェアロジック１３０の記憶装置を含む。ファームウェアロジック１３０は、起動処理の初めにプロセッサを初期化する助

けとなり、且つ、ホストＯＳ１２４のランタイムサービスを提供する起動前および起動後のルーチンを含んでよい。多くの実施形態では、コンピュータプラットフォーム１００は、ホストＯＳ１２４とプロセッサ／プラットフォームファームウェアとの間の通信を促進する目的で存在するＵＥＦＩ（Unified Extensible Firmware Interface）を利用する。ＵＥＦＩは部分的に、ホストＯＳ１２４を、標準的なインタフェースを利用して多くのプロセッサの１つと協働させる。

【００２８】

上述したように、プロセッサ１０２はマルチコアプロセッサであり、個々のコアは他のコアに対してＩＰＩを利用して割り込みを送り、ＩＰＩが対象とする１以上の他のコアに対してある形態の機能動作を要求することができる。加えて、多くの実施形態では、プロセッサ１０２は、ＳＭＭへ入退出する機能を有する。ＳＭＭにある間は通常、各コア内において通常のプロセッサの実行は中断される。言い換えると、現在のところＳＭＩがプロセッサに到達すると、ファームウェアロジック１３０またはプロセッサ１０２内の他のロジックは、全てのコアをＳＭＭに入らせて同期させる。

【００２９】

しかし多くの実施形態では、プロセッサ１０２内の１以上のコアがＳＭＭに入ることなく通常動作を続けつつ、１以上の他のコアがＳＭＭにおける動作をすることができる。例えば、図２は、処理コアがＳＭＭ中に行うタスクがない場合に、ＳＭＭから該処理コアを退出させる処理の一実施形態のフロー図である。処理は、ハードウェア、ソフトウェア、またはこれらの組み合わせにより行うことができる。処理は、コアが、第１のブロードキャストＳＭＩを受信するロジックを処理することにより始まる（処理ブロック２００）。ＳＭＩは、全ての処理コアにブロードキャストされる。ＳＭＩを受信すると、コア内の処理ロジックがＳＭＭに入る（処理ブロック２０２）。次に、これも全てのコアに送信された第２のブロードキャストＳＭＩを受信する（処理ブロック２０４）。多くの実施形態では、第２のブロードキャストＳＭＩは、ＳＭＭでビジー状態にない各コアに対して、ＳＭＭを退出して通常予定していたタスクに戻ってよい旨を伝える。

【００３０】

従って、任意のコア内の処理ロジックは、対象となっているコアが完了すべきＳＭＭタスクを有しているのか、アイドル状態であるのかを判断する（処理ブロック２０６）。コアが完了すべきＳＭＭタスクを有している場合には、処理ロジックはそのコアをＳＭＭ内に留め、ＳＭＭ関連のタスクを行わせる（処理ブロック２０８）。逆にコアがＳＭＭでアイドル状態にあり、完了すべきＳＭＭ関連のタスクを持たない場合、処理ロジックはそのコアをＳＭＭから退出させて、通常の動作に戻す（処理ブロック２１０）。この処理は、各コア内の処理ロジックにより、または、各コア外の、各コアのステータスを遠隔判断する機能を有するプラットフォームレベルロジックにより行うことができる。従って、この処理の終了時には、ＳＭＭに送られタスクを与えられた各コアがＳＭＭに留まることができる。逆にＳＭＭコア同期処理のみの理由でＳＭＭへと送られ、ＳＭＭ関連のタスクを持たない各コアは、ＳＭＭから退出して、通常動作（例えば、ホストＯＳの上部のアプリケーションコードの実行）を続ける。

【００３１】

図３は、シングルコアをＳＭＭに入らせる処理の別の実施形態を示す。図２の処理の欠点は、ＳＭＩのブロードキャストがあまり効率的でないことである。本来ならばＳＭＭから退出せねばならないコアが先ずはＳＭＭに入ってから退出させられる。これは、他の１以上のコアがタスクを行っている間中、動作しないコアをアイドル状態のままＳＭＭに留めておくよりは効率的ではあるが、対象を絞ったＳＭＩを利用するよりは潜在的に非効率である。対象を絞ったＳＭＩは、特定のコアに方向付けられる。故に、単一のコアがＳＭＭでタスクを行う必要がある場合、対象を絞ったＳＭＩを対象となるコアに直接送ることができ、他の各コアはＳＭＩを受け取らない。従って、対象を絞ったＳＭＩをコンピュータプラットフォームに実装する場合、各コアの処理ロジックは、対象を絞ったＳＭＩを受け取った場合には自身をＳＭＭに入らせることができるが、ＳＭＩを受け取っていない場

合には邪魔されずに通常動作を続けることができる。このように、コアの処理ロジックは、対象を絞ったＳＭＩを受信したか否かを判断する（処理ブロック３００）。コアの処理ロジックが対象を絞ったＳＭＩを受け取っていない場合には、処理ロジックはコアに対して、通常動作を続けさせることができる（処理ブロック３０２）。他方、処理ロジックが対象を絞ったＳＭＩを受け取ったと判断する場合には、ロジックはコアをＳＭＭに入らせ、そのコアにＳＭＭ関連のタスクを行わせる（処理ブロック３０４）。

【００３２】

このように、図２または図３の処理を実装するにあたっては、プラットフォームは、任意の時点で、幾らかのコアをＳＭＭに有し、幾らかのコアをＳＭＭ外に有するプロセッサを有することができる。ＩＰＩは、幾らかのコアがＳＭＭにあり、これと同時に、幾らかのコアがＳＭＭ外にあるようなプロセッサを許容するプラットフォーム上で、重要な問題を生じる可能性がある。上述したように、ＳＭＭにないあるコアが、ＳＭＭにある別のコア宛にＩＰＩを開始した場合、このＩＰＩは、各コアのＩＰＩ処理メカニズムに修正を加えない限り、宛先のコアには達しないことになる。現在のところ、ＳＭＭ内のコアは、不在に見えるためにＩＰＩを受信することはできない。従って、ＳＭＭ内のコアにＩＰＩを受信させるような修正を行うことが望ましい。

【００３３】

図１に戻ると、多くの実施形態では、各コアは、ＳＭＭモードにあるＩＰＩの受信および再方向付けを処理するロジックを含む。例えばコア０（１０４）は、再方向付けロジック１３２を含みうる。再方向付けロジック１３２は、ハードウェア回路またはファームウェア記憶位置に記憶されるファームウェアコードを備えうる。

【００３４】

図４は、ＳＭＭ内の処理コアに対象を絞ってＩＰＩを再方向付けする処理の一実施形態のフロー図である。本処理は、ハードウェア回路またはソフトウェア（つまりファームウェア）、またはこれらの組み合わせを含みうる処理ロジックにより行うことができる。本処理は、例えばコア０（図１の１０４）等のコアがＳＭＩを受信することにより開始される（処理ブロック４００）。別の実施形態では、ＳＭＩはブロードキャストされるＳＭＩであっても、そのコアに対象が絞られたＳＭＩであってもよい。その後コアは、ＳＭＩに応じてＳＭＭに入る（処理ブロック４０２）。

【００３５】

コアがＳＭＭに入ると、処理ロジック（例えば図１の再方向付けロジック１３２）は、割り込み記述子テーブル（ＩＤＴ）を設定する（図１の１３４）（処理ブロック４０４）。ＩＤＴは、割り込みベクトルテーブルを実装するのに用いられるデータ構造であり、コアが、割り込みおよび例外への正しい応答を決定するのに利用される。ＩＤＴはシステムメモリ内に構築されてよい（図１の１２２）。ＩＤＴの開始のメモリアドレス位置をＩＤＴＲ内に記憶させてよい（図１の１１２）。次に、処理ロジックは、コアへの割り込みをイネーブルする（処理ブロック４０６）。これにより、コアはＳＭＭ中に割り込みを受信することができ、従来のように無視はしない。デバイスドライバの機能に影響を及ぼさないように、ＳＭＭ中のコアは、ＣＰＵ割り込みをイネーブルしつつ、ＰＩＣ（プログラム可能な割り込み制御回路）またはＡＰＩＣ（拡張された割込制御回路）により幾らかのデバイス割り込みを選択的にイネーブルおよび／またはディセーブルすることができる。

【００３６】

そして処理ロジックは、ＳＭＩハンドラの実行を開始する（処理ブロック４０８）。ＳＭＩハンドラは、ＳＭＩが要求したタスクの種類に応じた特定の処理セットを含みうる。コアへの割り込みがイネーブルされるので、ＳＭＩ実行中に処理ロジックは、別のコアからのＩＰＩの受信について、継続してポーリングを行う（処理ブロック４１０）。ＩＰＩが受信されていない間は、処理ロジックは標準ＳＭＩハンドラを実行し続ける（処理ブロック４０８）。他方、ＩＰＩが受信されると、処理ロジックは、先ずＩＰＩのベクトル数（vector number）を入手することによりＩＰＩ処理ルーチンに入る（処理ブロック４１２）。ＩＰＩのベクトル数はＩＤＴに見つかる場合が多い。

【００３７】
ベクトル数により、処理ロジックは、ＩＤＴ内の適切なエントリを見つけることができる。ひとたび利用するＩＤＴエントリが見つかると、処理ロジックはスーパジャンプ前のシステムコンテキストをセーブすることができる（処理ブロック４１６）。コアは、ＩＤＴ、１以上の制御レジスタ（図１のＣＲｓ－ＣＲｘ１１６）、および、おそらくは他のコア固有の情報をも変更させるスーパジャンプを行う。スーパジャンプ後のシステムコンテキストは、コア動作制御を、通常は標準ＯＳ制御の範囲外で動作するＳＭＩハンドラロジックから、ＯＳ実行ＩＰＩハンドラに渡すことができる。スーパジャンプは、一時的なＯＳ実行ＩＰＩハンドラに対して、ＩＰＩに関するタスクを完了させるよう制御することができる。処理ロジックは、ＩＰＩハンドラの完了時にコアをＳＭＩハンドラに戻すべく、スーパジャンプ前にコアのシステムコンテキスト（状態）をセーブする。システムコンテキストのセーブには、現在のＩＤＴ、１以上のＣＲ、命令ポインタ（ＥＩＰ）、および追加的な情報のセーブが含まれる。多くの実施形態では、システムコンテキスト（図１の１３６）は、システムメモリのシステム管理ランダムアクセスメモリ（ＳＭＲＡＭ）（図１の１３８）の部分内にセーブされる。ＳＭＲＡＭは、プロセッサが、ＳＭＭ関連のコードを保存するのに用いるシステムメモリの部分である。

【００３８】
システムコンテキストをセーブすると、処理ロジックはＯＳエントリポイントへのスーパジャンプを行う（処理ブロック４１８）。スーパジャンプにより、１以上のセーブ情報が、スーパジャンプに関する新たな情報（例えば新たなＥＩＰ）に置き換わる。次いで、処理ロジックは、ＯＳ　ＩＰＩハンドラの実行を開始する（処理ブロック４２０）。ＯＳ　ＩＰＩハンドラは、ＩＰＩが要求するタスクを行わせることでコアにＩＰＩに応答させる命令セットを実行する。ＯＳ　ＩＰＩハンドラ（図１の１４０）は、システムメモリ内に記憶されているオペレーティングシステム内で実行されうる。ＯＳ　ＩＰＩハンドラが自身のタスクを完了すると、処理ロジックは、割り込み終了（ＥＯＩ）コマンドをコンピュータシステムのＡＰＩＣに送り、ＡＰＩＣに割り込みタスクが完了した旨を通知する（処理ブロック４２２）。

【００３９】
多くの実施形態では、スーパジャンプを行うのに利用される命令を、特殊な遠い呼（specialized Far Call）としてよい。通常の遠い呼（normal Far Call）は、現在のコードセグメントとは異なるセグメントに位置するプロシージャへの呼である。ＯＳ　ＩＰＩハンドラが実行できるようにシステムコンテキストを切り替えるには、多くのレジスタ（ＣＲ０、ＣＲ３、ＣＲ４、ＧＤＴＲ、およびＩＤＴＲ等）を復元する必要がある。通常の遠い呼は、ＣＳ（コードセグメント）レジスタよびＥＩＰのみを切り替えればよい。従って、多くの実施形態では、プロセッサコアにフルコンテキスト切り替えをサポートさせるには、１ビットをＭＳＲに追加させればよい。ＭＳＲビットがイネーブルされると、遠い呼の命令が、超遠い呼（Super Far Call）としてパースされる。これにより、ＣＳ／ＥＩＰが切り替わるのみならず、CRx/GDTR/IDTR/等も切り替わる。従って、１つの命令を利用してＯＳ環境に入ることができる。ＯＳ　ＩＰＩハンドラの完了時に、ＯＳ　Ｉｒｅｔ（割り込みリターン）命令をスーパＩｒｅｔとしてパースすることで、CRx/GDTR/IDTR/等を切り替え、コンテキストをＳＭＭ動作に戻すことができる。

【００４０】
ＥＯＩをＡＰＩＣに送った後で、処理ロジックは、ＯＳエントリポイントにジャンプする前にコアが存在していた位置に戻ることができるが、これは、ＯＳ　ＩＰＩハンドラが完了したから可能となる（処理ブロック４２４）。スーパジャンプの前の位置に戻る際に、処理ロジックは、ジャンプ前にセーブされていたシステムコンテキストを復元する（ＩＤＴ，ＥＩＰ、１以上のＣＲ等を含み）（処理ブロック４２６）。ひとたびもとのシステムコンテキストが復元されると、処理ロジックは、コアをＳＭＭのＳＭＩハンドラの処理に戻す（処理ブロック４２８）。ＳＭＩハンドラの処理は、別のＩＰＩが受信されるまで（ブロック４１０から初めて全スーパジャンプを繰り返すことになる）、あるいは、ＳＭ

Ｉ処理が完了するまで続けられ、こうしてコアはＳＭＭを抜け（処理ブロック４３０）、処理が完了する。

【００４１】

多くの実施形態では、オペレーティングシステムは、ＯＳが制御を獲得しＯＳ　ＩＰＩハンドラを実行する前にＶＭＭ（仮想マシンマネージャ）または別のセキュアな仮想マシンによりセキュアに計測される。この計測処理により、スーパジャンプ中にセキュアな環境をハンドオフ制御（hand off control）へと確実に移行させる。多くの実施形態では、ＳＭＩ転送モニタ（ＳＴＭ）がサポートされる。ＳＴＭのサポートにより、ＳＴＭは、ＳＭＭのＩＰＩ処理用のＯＳゲストを再開する。ＳＴＭは、ＶＭＣＳ（仮想マシン制御構造）環境を設定して、割り込みを受信した際にＶＭＥｘｉｔをイネーブルすることができる。このイネーブルは初期化中に行われてよい。そして、ＩＰＩが起こった際には、ＳＴＭは、割り込みに起因したＶＭＥｘｉｔコマンドを受信する。次いでＳＴＭはＯＳ　ＩＰＩハンドラを開始するセキュアな環境を設定して、ＳＭＭのＩＰＩハンドラに対してＶＭＥｎｔｒｙを行うことができる。加えて、ＳＴＭは、コンテキストを記憶させているＳＭＲＡＭをセキュアにすることにより、記憶されているコンテキストを保護し、ＳＭＭ実行環境を、ＳＭ　ＩＰＩハンドラによる傍受から守ることができる。これにより、ＳＭＭ実行環境を、不良または悪意のあるＯＳ実体による危険から守ることもできるであろう。

【００４２】

システム管理モード中にプロセッサ間割り込みを再方向付けるデバイス、システム、および方法の実施形態が記載された。これら実施形態は特定の例示的な実施形態に基づいて記載されてきた。しかし当業者であれば、本開示に基づき開示される実施形態の広義の精神および範囲から逸脱しないで、これら実施形態に対して様々な変形例および変更例を相当することができよう。従って明細書および図面は例示的であり、制限的な意味合いで捉えられるべきものではない。実施形態の例を項目として示す。

［項目１］

複数のプロセッサコアのうち第１のプロセッサコアがシステム管理モードに入り、複数のプロセッサコアのうちの、第１のプロセッサコアとは異なる１以上のプロセッサコアは動作を維持してシステム管理モードには入らない段階と、

第１のプロセッサコアが、システム管理モード中にプロセッサ間割り込みに応答する段階とを備える方法。

［項目２］

第１のプロセッサコアがシステム管理モードに入った後で、

第１のプロセッサコアによる割り込み処理を許可するべく割り込み記述子テーブルをセーブする段階と、

第１のプロセッサコアに対するプロセッサ間割り込みをイネーブルする段階とをさらに備える項目１に記載の方法。

［項目３］

第１のプロセッサコアがプロセッサ間割り込みを受信した後で、

第１のプロセッサコアの現在のシステムコンテキストをメモリにセーブする段階と、

第１のプロセッサコアに対して、スーパジャンプに利用される新たなスーパジャンプシステムコンテキストを設定する段階と、

スーパジャンプを、オペレーティングシステムのプロセッサ間割り込みハンドラの位置に行う段階と、

オペレーティングシステムのプロセッサ間割り込みハンドラを実行する段階とをさらに備える項目２に記載の方法。

［項目４］

オペレーティングシステムのプロセッサ間割り込みハンドラを実行した後で、

スーパジャンプの前の位置に戻る段階と、

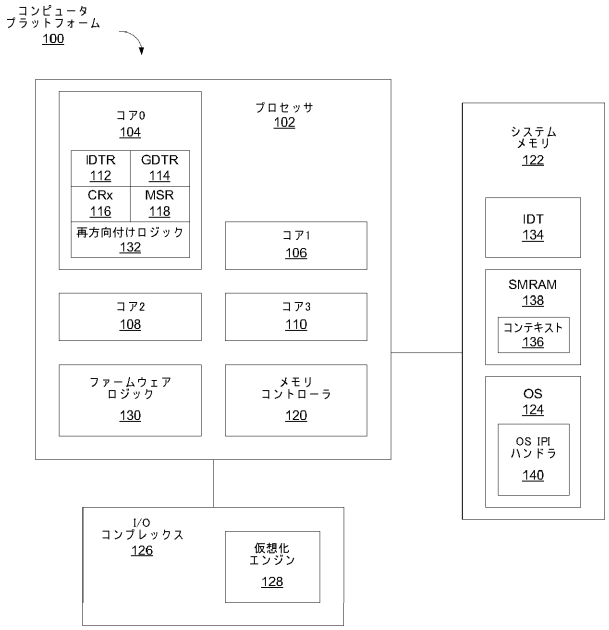
第１のプロセッサコアの現在のシステムコンテキストをメモリから復元する段階と、

第１のプロセッサコアを、システム管理モードから抜けさせる段階とをさらに備える項

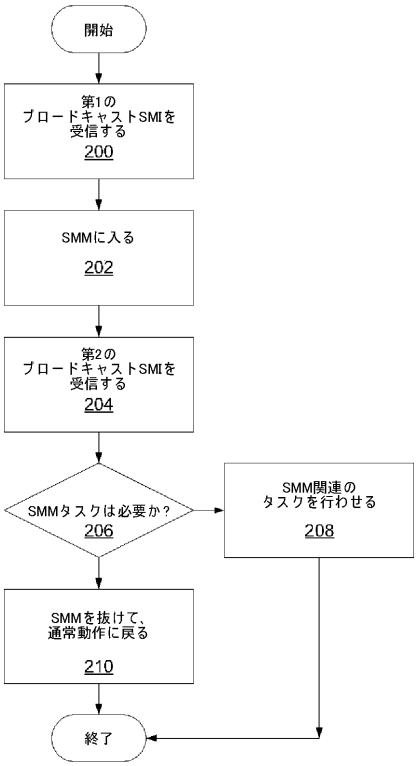
目３に記載の方法。
［項目５］
オペレーティングシステムのプロセッサ間割り込みハンドラを制御するオペレーティングシステムが制御を獲得してハンドラを実行する前に、システム管理割り込み転送モニタによりオペレーティングシステムをセキュアに計測する段階と、
計測時に、システム管理割り込み転送モニタがオペレーティングシステムに制御を送る段階とをさらに備える項目３に記載の方法。
［項目６］
プロセッサ間割り込みに呼応して、スーパジャンプに関連するコンテキスト切り替えにおいて、１以上の第１のプロセッサコアの制御レジスタと、一の第１のプロセッサコアのグローバル記述子テーブルレジスタとを含むように、第１のプロセッサコアのマシン固有レジスタにビットを設定する段階をさらに備える項目３に記載の方法。
［項目７］
コンピュータシステムの第１のプロセッサコアであって、
システム管理モードに入り、
システム管理モード中にプロセッサ間割り込みに応答し、
コンピュータシステムの中の１以上のさらなるプロセッサコアは動作を維持してシステム管理モードには入らない第１のプロセッサコア。
［項目８］
システム管理モードに入った後で、さらに、
割り込み処理を許可するべく割り込み記述子テーブルをセーブし、
プロセッサ間割り込みをイネーブルする項目７に記載の第１のプロセッサコア。
［項目９］
プロセッサ間割り込みを受信した後で、さらに、
現在のシステムコンテキストをメモリにセーブし、
スーパジャンプに利用される新たなスーパジャンプシステムコンテキストを設定し、
スーパジャンプを、オペレーティングシステムのプロセッサ間割り込みハンドラの位置に行い、
オペレーティングシステムのプロセッサ間割り込みハンドラを実行する項目８に記載の第１のプロセッサコア。
［項目１０］
オペレーティングシステムのプロセッサ間割り込みハンドラを実行した後で、さらに、
スーパジャンプの前の位置に戻り、
現在のシステムコンテキストをメモリから復元し、
システム管理モードから抜ける項目９に記載の第１のプロセッサコア。
［項目１１］
オペレーティングシステムのプロセッサ間割り込みハンドラを制御するオペレーティングシステムが制御を獲得してハンドラを実行する前に、システム管理割り込み転送モニタによりオペレーティングシステムをセキュアに計測し、
計測時に、システム管理割り込み転送モニタがオペレーティングシステムに制御を送る項目９に記載の第１のプロセッサコア。
［項目１２］
プロセッサ間割り込みに呼応して、スーパジャンプに関連するコンテキスト切り替えにおいて、１以上の第１のプロセッサコアの制御レジスタと、一の第１のプロセッサコアのグローバル記述子テーブルレジスタとを含むように、第１のプロセッサコアのマシン固有レジスタにビットを設定する項目９に記載の第１のプロセッサコア。
［項目１３］
システム管理モードに入り、システム管理モード中にプロセッサ間割り込みに応答する第１のプロセッサコアと、
動作を維持してシステム管理モードには入らない１以上のさらなるプロセッサコアとを

備えるシステム。
［項目１４］
第１のプロセッサコアはさらに、システム管理モードに入った後で、
割り込み処理を許可するべく割り込み記述子テーブルをセーブし、
プロセッサ間割り込みをイネーブルする項目１３に記載のシステム。
［項目１５］
第１のプロセッサコアはさらに、プロセッサ間割り込みを受信した後で、
現在のシステムコンテキストをメモリにセーブし、
スーパジャンプに利用される新たなスーパジャンプシステムコンテキストを設定し、
スーパジャンプを、オペレーティングシステムのプロセッサ間割り込みハンドラの位置に行い、
オペレーティングシステムのプロセッサ間割り込みハンドラを実行する項目１４に記載のシステム。
［項目１６］
第１のプロセッサコアはさらに、オペレーティングシステムのプロセッサ間割り込みハンドラを実行した後で、
スーパジャンプの前の位置に戻り、
現在のシステムコンテキストをメモリから復元し、
システム管理モードから抜ける項目１５に記載のシステム。
［項目１７］
第１のプロセッサコアはさらに、オペレーティングシステムのプロセッサ間割り込みハンドラを制御するオペレーティングシステムが制御を獲得してハンドラを実行する前に、システム管理割り込み転送モニタによりオペレーティングシステムをセキュアに計測し、
計測時に、システム管理割り込み転送モニタがオペレーティングシステムに制御を送る項目１５に記載のシステム。
［項目１８］
第１のプロセッサコアはさらに、プロセッサ間割り込みに呼応して、スーパジャンプに関連するコンテキスト切り替えにおいて、１以上の第１のプロセッサコアの制御レジスタと、一の第１のプロセッサコアのグローバル記述子テーブルレジスタとを含むように、第１のプロセッサコアのマシン固有レジスタにビットを設定する項目１５に記載のシステム。

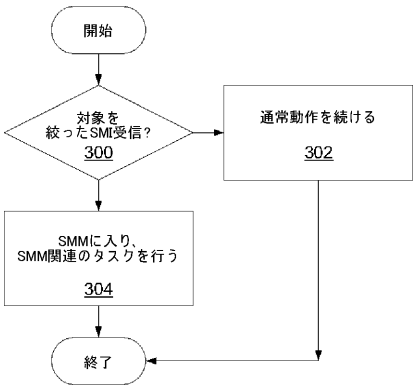
【図１】


コンピュータ
プラットフォーム
100
プロセッサ
102
コア0
104
IDTR
112
GDTR
114
CRx
116
MSR
118
再方向付けロジック
132
コア1
106
コア2
108
コア3
110
ファームウェア
ロジック
130
メモリ
コントローラ
120
I/O
コンプレックス
126
仮想化
エンジン
128
システム
メモリ
122
IDT
134
SMRAM
138
コンテキスト
136
OS
124
OS IPI
ハンドラ
140

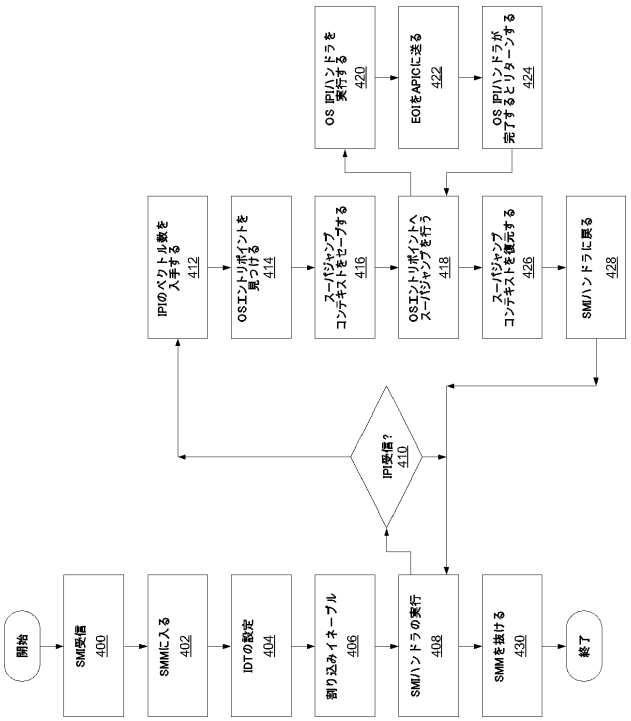
【図２】


開始
第1の
ブロードキャストSMIを
受信する
200
SMMに入る
202
第2の
ブロードキャストSMIを
受信する
204
SMMタスクは必要か？
206
SMM関連の
タスクを行わせる
208
SMMを抜けて、
通常動作に戻る
210
終了

【図３】


開始
対象を
絞ったSMI受信？
300
通常動作を続ける
302
SMMに入り、
SMM関連のタスクを行う
304
終了

【図４】


開始
SMI受信
400
SMMに入る
402
IDTの設定
404
割り込みイネーブル
406
SMIハンドラの実行
408
SMMを抜ける
430
終了
IPI受信？
410
IPIのベクトル番号を
入手する
412
OSエントリポイントを
見つける
414
スーパバイザ
コンテキストをセーブする
416
OSエントリポイントへ
スーパバイザを行う
418
スーパバイザ
コンテキストを復元する
426
SMIハンドラに戻る
428
OS IPIハンドラを
実行する
420
EOIをAPICに送る
422
OS IPIハンドラが
完了するとリターンする
424

【手続補正書】
【提出日】平成25年1月28日(2013.1.28)
【手続補正１】
【補正対象書類名】特許請求の範囲
【補正対象項目名】全文
【補正方法】変更
【補正の内容】
【特許請求の範囲】
【請求項１】
単一のプロセッサに存在する複数のプロセッサコアが、全てのコアにシステム管理モードに入るように命令する第１のブロードキャストシステム管理割り込みを受信する段階と、
前記複数のプロセッサコアが、前記第１のブロードキャストシステム管理割り込みを受信したことに応じて前記システム管理モードに入る段階と、
前記複数のプロセッサコアが、前記システム管理モード内で１以上のタスクを実行するのにビジーでない全てのコアに前記システム管理モードから退出するように命令する第２のブロードキャストシステム管理割り込みを、受信する段階と、
前記複数のプロセッサコアの１つを含む少なくとも第１のプロセッサコアが、前記システム管理モード内で１以上のタスクを実行するのにビジーでないと決定したことに応じて前記システム管理モードを退出する段階と、
を備える方法。
【請求項２】
前記第１のプロセッサコアが前記システム管理モードに入った後で、
前記第１のプロセッサコアによる割り込み処理を許可するべく割り込み記述子テーブルをセーブする段階と、
前記第１のプロセッサコアに対するプロセッサ間割り込みをイネーブルする段階とをさらに備える請求項１に記載の方法。
【請求項３】
前記第１のプロセッサコアがプロセッサ間割り込みを受信した後で、
前記第１のプロセッサコアの現在のシステムコンテキストをメモリにセーブする段階と、
前記第１のプロセッサコアに対して、オペレーティングシステムのプロセッサ間ハンドラに関する新たなシステムコンテキストを設定する段階と、
前記第１のプロセッサコアが、処理位置を前記プロセッサ間ハンドラの位置にジャンプさせる段階と、
前記オペレーティングシステムのプロセッサ間ハンドラを実行する段階とをさらに備える請求項２に記載の方法。
【請求項４】
前記オペレーティングシステムのプロセッサ間ハンドラを実行した後で、
前記第１のプロセッサコアの処理位置をジャンプ前の位置に戻す段階と、
前記第１のプロセッサコアの前記現在のシステムコンテキストを前記メモリから復元する段階と、
前記第１のプロセッサコアを、前記システム管理モードから抜けさせる段階とをさらに備える請求項３に記載の方法。
【請求項５】
前記オペレーティングシステムのプロセッサ間ハンドラを制御するオペレーティングシステムが前記オペレーティングシステムのプロセッサ間ハンドラを実行する前に、
前記オペレーティングシステムの前記プロセッサ間ハンドラを開始するセキュアな環境を設定する段階とをさらに備える請求項３に記載の方法。
【請求項６】

前記プロセッサ間割り込みに呼応して、前記システムコンテキストの切り替えにおいて、１以上の第１のプロセッサコアの制御レジスタと、一の第１のプロセッサコアのグローバル記述子テーブルレジスタとを含むように、前記第１のプロセッサコアのマシン固有レジスタにビットを設定する段階をさらに備える請求項３に記載の方法。

【請求項７】

プロセッサ中の第１のプロセッサコアであって、

前記プロセッサ内の全てのコアにシステム管理モードに入るように命令する第１のブロードキャストシステム管理割り込みを受信し、

前記第１のブロードキャストシステム管理割り込みを受信したことに応じて前記システム管理モードに入り、

前記システム管理モード内で１以上のタスクを実行するのにビジーでない前記プロセッサ内の全てのコアに、前記システム管理モードから退出するように命令する、第２のブロードキャストシステム管理割り込みを受信し、

前記第１のプロセッサコアは１以上のタスクを実行するのにビジーでないと決定し、

前記第１のプロセッサコアが前記１以上のタスクを実行するのにビジーでないと決定したことに応じて前記システム管理モードを退出し、

前記第２のブロードキャストシステム管理割り込みを受信した後に、前記プロセッサ内の少なくとも他の１つのプロセッサコアは前記１以上のタスクを実行するのにビジーであり、前記システム管理モードに留まる、

第１のプロセッサコア。

【請求項８】

前記システム管理モードに入った後で、さらに、

割り込み処理を許可するべく割り込み記述子テーブルをセーブし、

プロセッサ間割り込みをイネーブルする請求項７に記載の第１のプロセッサコア。

【請求項９】

さらに、プロセッサ間割り込みを受信した後で、

現在のシステムコンテキストをメモリにセーブし、

オペレーティングシステムのプロセッサ間ハンドラに関する新たなシステムコンテキストを設定し、

処理位置を前記プロセッサ間ハンドラの位置にジャンプさせ、

前記オペレーティングシステムのプロセッサ間ハンドラを実行する請求項８に記載の第１のプロセッサコア。

【請求項１０】

前記オペレーティングシステムのプロセッサ間ハンドラを実行した後で、さらに、

処理位置をジャンプ前の位置に戻し、

前記現在のシステムコンテキストを前記メモリから復元し、

前記システム管理モードから抜ける請求項９に記載の第１のプロセッサコア。

【請求項１１】

前記オペレーティングシステムのプロセッサ間ハンドラを制御するオペレーティングシステムが前記オペレーティングシステムのプロセッサ間ハンドラを実行する前に、

前記オペレーティングシステムの前記プロセッサ間ハンドラを開始するセキュアな環境を設定する請求項９に記載の第１のプロセッサコア。

【請求項１２】

前記プロセッサ間割り込みに呼応して、前記システムコンテキストの切り替えにおいて、１以上の第１のプロセッサコアの制御レジスタと、一の第１のプロセッサコアのグローバル記述子テーブルレジスタとを含むように、前記第１のプロセッサコアのマシン固有レジスタにビットを設定する請求項９に記載の第１のプロセッサコア。

フロントページの続き

(72)発明者　ジャオ、ジェウェン
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