(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第6089349号
(P6089349)
(45)発行日　平成29年3月8日(2017.3.8)　(24)登録日　平成29年2月17日(2017.2.17)

(51)Int.Cl.　　F I
G06F 9/46 (2006.01)　　G06F 9/46 350
G06F 15/177 (2006.01)　　G06F 15/177 C

請求項の数 13　(全 19 頁)

(21)出願番号　特願2015-559405 (P2015-559405)
(86)(22)出願日　平成25年5月28日 (2013.5.28)
(65)公表番号　特表2016-508647 (P2016-508647A)
(43)公表日　平成28年3月22日 (2016.3.22)
(86)国際出願番号　PCT/CN2013/076329
(87)国際公開番号　WO2014/190486
(87)国際公開日　平成26年12月4日 (2014.12.4)
審査請求日　平成27年8月28日 (2015.8.28)

(73)特許権者　504277388
▲ホア▼▲ウェイ▼技術有限公司
HUAWEI TECHNOLOGIES CO., LTD.
中華人民共和国518129広東省深▲セン▼市龍岡区坂田華為本社ビル
Huawei Administration Building, Bantian, Longgang District Shenzhen, Guangdong 518129 (CN)
(74)代理人　100146835
弁理士　佐伯　義文
(74)代理人　100140534
弁理士　木内　敬二

最終頁に続く

(54)【発明の名称】マルチコアアーキテクチャでのリソース分離を支援するための方法およびシステム

(57)【特許請求の範囲】
【請求項１】
マルチコアアーキテクチャでのリソース分離を支援するためのシステムにおいて、
プロセッサであって、前記プロセッサが、複数の処理コアを備え、前記複数の処理コアが、起動時一次コアおよび少なくとも1つの起動時二次コアを備える、プロセッサと、
メモリ空間であって、前記メモリ空間が、共有メモリセグメントおよび前記処理コアのメモリセグメントを備え、実行情報および前記処理コアの実行時エントリアドレスが、前記共有メモリセグメントに記憶され、前記実行情報が、各処理コアと各オペレーティングシステムとの間のマッピング関係、ならびに前記メモリセグメントのサイズおよび開始アドレスを備え、前記処理コアの前記実行時エントリアドレスが、前記処理コアが実行する必要があるオペレーティングシステムのカーネルイメージの記憶アドレスをそれぞれの対応するメモリセグメントで示すために使用される、メモリ空間とを備え、
前記プロセッサは、前記処理コアが実行する必要がある前記オペレーティングシステムの前記カーネルイメージを前記処理コアの前記メモリセグメントに個別に記憶し、前記処理コアの前記実行時エントリアドレスを前記共有メモリセグメントに個別に書き込み、起動されるべき処理コアは、それぞれの実行時エントリアドレスを個別に取得し、それぞれのメモリセグメントに記憶された前記オペレーティングシステムの前記カーネルイメージを実行する、マルチコアアーキテクチャでのリソース分離を支援するためのシステム。
【請求項２】
前記プロセッサの前記処理コアは、プリセット選出アルゴリズムに従って前記処理コア

から実行時一次コアを選出して決定し、前記実行時一次コアは、入力/出力I/0リソースの調停制御を行うために使用される、請求項1に記載のマルチコアアーキテクチャでのリソース分離を支援するためのシステム。

【請求項３】

実行時二次コアは、I/0操作を必要とするデータを前記共有メモリセグメントに入れ、前記I/0操作に関連するメモリアドレスを前記実行時一次コアに通知し、前記実行時一次コアは、外部デバイスへの前記I/0操作を行うために前記実行時一次コアのために設けられた局所的物理ドライバを呼び出すか、

または、前記実行時二次コアは、外部デバイスへのI/0操作を行うために、前記実行時二次コアのために設けられた局所的物理ドライバを呼び出す、請求項2に記載のマルチコアアーキテクチャでのリソース分離を支援するためのシステム。

【請求項４】

ブートローダーは、前記起動時一次コアの内部に配置され、

前記プロセッサが、前記処理コアが実行する必要がある前記オペレーティングシステムの前記カーネルイメージを前記処理コアの前記メモリセグメントに個別に記憶することは、

前記ブートローダーが、前記処理コアが実行する必要がある前記オペレーティングシステムの前記カーネルイメージを前記処理コアの前記メモリセグメントに前記実行情報に従って個別に記憶することを含むか、

または、前記ブートローダーは、前記起動時一次コアの内部に配置され、

前記プロセッサが、前記処理コアが実行する必要がある前記オペレーティングシステムの前記カーネルイメージを前記処理コアの前記メモリセグメントに個別に記憶することは、

前記ブートローダーが、実行される必要がある前記オペレーティングシステムの前記カーネルイメージを前記プロセッサの少なくとも1つの処理コアの前記メモリセグメントに前記実行情報に従って記憶し、前記オペレーティングシステムの前記カーネルイメージを実行すること、および前記オペレーティングシステムを実行する前記処理コアが、前記オペレーティングシステムを実行する必要がある前記プロセッサの他の処理コアに対応するオペレーティングシステムのカーネルイメージを前記他の処理コアのメモリセグメントに前記実行情報に従って個別に記憶することを含む、請求項1から3のいずれか一項に記載のマルチコアアーキテクチャでのリソース分離を支援するためのシステム。

【請求項５】

前記処理コアは、相互に異なるオペレーティングシステムのカーネルイメージを実行する、または少なくとも2つの処理コアを含む処理コア群は、同じオペレーティングシステムのカーネルイメージを実行し、さらに、前記処理コア群を構成するすべての処理コアは、同じメモリセグメントを共有する、請求項1から4のいずれか一項に記載のマルチコアアーキテクチャでのリソース分離を支援するためのシステム。

【請求項６】

マルチコアアーキテクチャでのリソース分離を支援するためのシステムに適用可能な、マルチコアアーキテクチャでのリソース分離を支援するための方法であって、前記システムは、プロセッサおよびメモリ空間を備え、前記プロセッサは、複数の処理コアを備え、前記複数の処理コアは、起動時一次コアおよび少なくとも1つの起動時二次コアを備え、前記メモリ空間は、共有メモリセグメントおよび前記処理コアのメモリセグメントを備え、実行情報および前記処理コアの実行時エントリアドレスは、前記共有メモリセグメントに記憶され、前記実行情報は、各処理コアと各オペレーティングシステムとの間のマッピング関係、ならびに前記メモリセグメントのサイズおよび開始アドレスを備え、前記処理コアの前記実行時エントリアドレスは、前記処理コアが実行する必要があるオペレーティングシステムのカーネルイメージの記憶アドレスをそれぞれのメモリセグメントで示すために使用され、

マルチコアアーキテクチャでのリソース分離を支援するための前記方法は、

前記処理コアが実行する必要がある前記オペレーティングシステムの前記カーネルイメージを前記処理コアの前記メモリセグメントに前記プロセッサによって個別に記憶し、前記処理コアの前記実行時エントリアドレスを前記共有メモリセグメントに個別に書き込むステップと、

それぞれの実行時エントリアドレスを、起動されるべき処理コアによって個別に取得し、それぞれのメモリセグメントに記憶された前記オペレーティングシステムの前記カーネルイメージを実行するステップとを含む、マルチコアアーキテクチャでのリソース分離を支援するための方法。

【請求項７】

前記プロセッサの前記処理コアは、プリセット選出アルゴリズムに従って前記処理コアから実行時一次コアを選出して決定し、前記実行時一次コアは、入力/出力I/0リソースの調停制御を行うために使用される、請求項6に記載のマルチコアアーキテクチャでのリソース分離を支援するための方法。

【請求項８】

実行時二次コアは、I/0操作を必要とするデータを前記共有メモリセグメントに入れ、前記I/0操作に関連するメモリアドレスを前記実行時一次コアに通知し、その結果前記実行時一次コアは、外部デバイスへの前記I/0操作を行うために前記実行時一次コアのために設けられた局所的物理ドライバを呼び出す、請求項7に記載のマルチコアアーキテクチャでのリソース分離を支援するための方法。

【請求項９】

実行時二次コアは、外部デバイスへのI/0操作を行うために、前記実行時二次コアのために設けられた局所的物理ドライバを呼び出す、請求項7に記載のマルチコアアーキテクチャでのリソース分離を支援するための方法。

【請求項１０】

ブートローダーは、前記起動時一次コアの内部に配置され、

前記処理コアが実行する必要がある前記オペレーティングシステムの前記カーネルイメージを前記処理コアの前記メモリセグメントに前記プロセッサによって個別に記憶するステップは、

前記処理コアが実行する必要がある前記オペレーティングシステムの前記カーネルイメージを前記処理コアの前記メモリセグメントに前記実行情報に従って前記ブートローダーによって個別に記憶するステップを含む、請求項6から9のいずれか一項に記載のマルチコアアーキテクチャでのリソース分離を支援するための方法。

【請求項１１】

ブートローダーは、前記起動時一次コアの内部に配置され、

前記処理コアが実行する必要がある前記オペレーティングシステムの前記カーネルイメージを前記処理コアの前記メモリセグメントに前記プロセッサによって個別に記憶するステップは、

実行される必要がある前記オペレーティングシステムの前記カーネルイメージを前記プロセッサの少なくとも1つの処理コアに割り当てられる前記メモリセグメントに前記実行情報に従って前記ブートローダーによって記憶し、前記オペレーティングシステムの前記カーネルイメージを実行するステップ、および前記オペレーティングシステムを実行する必要がある前記プロセッサの他の処理コアに対応するオペレーティングシステムのカーネルイメージを前記他の処理コアのメモリセグメントに前記実行情報に従って、前記オペレーティングシステムを実行する前記処理コアによって個別に記憶するステップを含む、請求項6から9のいずれか一項に記載のマルチコアアーキテクチャでのリソース分離を支援するための方法。

【請求項１２】

前記処理コアは、相互に異なるオペレーティングシステムのカーネルイメージを実行する、または少なくとも2つの処理コアを含む処理コア群は、同じオペレーティングシステムのカーネルイメージを実行し、さらに、前記処理コア群を構成するすべての処理コアは

、同じメモリセグメントを共有する、請求項6から11のいずれか一項に記載のマルチコアアーキテクチャでのリソース分離を支援するための方法。

【請求項１３】

プログラムの記録されたコンピュータ可読記憶媒体であって、

前記プログラムは、請求項6から12のいずれか一項に記載の方法をマルチコアアーキテクチャのプロセッサを有するコンピュータに実行させる、コンピュータ可読記憶媒体。

【発明の詳細な説明】

【技術分野】

【０００１】

本発明の実施形態は、コンピュータ技術に関し、詳しくは、マルチコアアーキテクチャでのリソース分離を支援するための方法およびシステムに関する。

【背景技術】

【０００２】

大規模統合リソースプールに基づくクラウド記憶システムは、データ記憶および管理に焦点を合わせ、複数のサービスの統合展開を支援し、ディスクを効率的に使用し、オンデマンド供給を用いて投資損失を低減し、合理的なスケジューリングを用いてリソースを効率的に使用し、総合エネルギー消費を低減する。

【０００３】

従来技術では、マルチコアARMプロセッサおよび少数のハードディスクを有するサーバーが、採用される。例えば、縮小命令セットコンピューティング(Reduced Instruction Set Computing、略してRISC)に基づくARMサーバーは、ARMプロセッサの利点、すなわち低消費電力、高集積度、および低総合コストを発揮することもある。

【０００４】

しかしながら、従来技術での複数のハードディスクと一緒のマルチコアARMプロセッサの場合は、オペレーティングシステムは、マルチコアプロセッサで実行され、オペレーティングシステムが、機能しなくなるとき、サーバーの障害粒度は、複数のハードディスクである。さらに、単一ハードディスクの容量は、大きく、入力/出力(Input/Output、略してI/O)スループット帯域幅は、小さいので、データ復元量およびネットワーク送信データ量は両方とも、指数関数的に増大し、障害回復粒度は、粗く、大きな圧力をシステムにもたらす。

【発明の概要】

【課題を解決するための手段】

【０００５】

本発明の実施形態は、マルチコアプロセッサの障害ドメインが単一ハードディスクにとどまることを可能にするために、マルチコアアーキテクチャでのリソース分離を支援するための方法およびシステムを提供する。

【０００６】

第1の態様によると、本発明の実施形態は、マルチコアアーキテクチャでのリソース分離を支援するためのシステムを提供し、ここでシステムは、プロセッサであって、プロセッサが、複数の処理コアを含み、複数の処理コアが、起動時一次コアおよび少なくとも1つの起動時二次コアを含む、プロセッサと、メモリ空間であって、メモリ空間が、共有メモリセグメントおよび処理コアのメモリセグメントを含む、メモリ空間とを含む。実行情報および処理コアの実行時エントリアドレスは、共有メモリセグメントに記憶される。実行情報は、各処理コアと各オペレーティングシステムとの間のマッピング関係、ならびにメモリセグメントのサイズおよび開始アドレスを含む。処理コアの実行時エントリアドレスは、処理コアが実行する必要があるオペレーティングシステムのカーネルイメージの記憶アドレスをそれぞれのメモリセグメントで示すために使用される。プロセッサは、処理コアが実行する必要があるオペレーティングシステムのカーネルイメージを処理コアのメモリセグメントに個別に記憶し、処理コアの実行時エントリアドレスを共有メモリセグメントに個別に書き込む。起動されるべき処理コアは、それぞれの実行時エントリアドレス

を個別に取得し、それぞれのメモリセグメントに記憶されたオペレーティングシステムのカーネルイメージを実行する。

【０００７】

第1の態様の第1の可能な実施方式では、プロセッサの処理コアは、プリセット選出アルゴリズムに従って処理コアから実行時一次コアを選出して決定し、ここで実行時一次コアは、入力/出力I/0リソースの調停制御を行うために使用される。

【０００８】

第1の態様の第1の可能な実施方式を参照すると、第2の可能な実施方式では、実行時二次コアは、I/0操作を必要とするデータを共有メモリセグメントに入れ、I/0操作に関連するメモリアドレスを実行時一次コアに通知し、その結果実行時一次コアは、外部デバイスへのI/0操作を行うためにそれ自身の局所的物理ドライバを呼び出す。

【０００９】

第1の態様の第1の可能な実施方式を参照すると、第3の可能な実施方式では、実行時二次コアは、外部デバイスへのI/0操作を行うために、それ自身の局所的物理ドライバを呼び出す。

【００１０】

第1の態様および第1の態様の第1から第3の可能な実施方式の任意の1つを参照すると、第4の可能な実施方式では、ブートローダーは、起動時一次コアの内部に配置され、プロセッサが、処理コアが実行する必要があるオペレーティングシステムのカーネルイメージを処理コアのメモリセグメントに個別に記憶することは、ブートローダーが、処理コアが実行する必要があるオペレーティングシステムのカーネルイメージを処理コアのメモリセグメントに実行情報に従って個別に記憶することを含む。

【００１１】

第1の態様および第1の態様の第1から第3の可能な実施方式の任意の1つを参照すると、第5の可能な実施方式では、ブートローダーは、起動時一次コアの内部に配置され、プロセッサが、処理コアが実行する必要があるオペレーティングシステムのカーネルイメージを処理コアのメモリセグメントに個別に記憶することは、ブートローダーが、実行される必要があるオペレーティングシステムのカーネルイメージをプロセッサの少なくとも1つの処理コアに割り当てられるメモリセグメントに実行情報に従って記憶し、オペレーティングシステムのカーネルイメージを実行すること、およびオペレーティングシステムを実行する処理コアが、オペレーティングシステムを実行する必要があるプロセッサの他の処理コアに対応するオペレーティングシステムのカーネルイメージを他の処理コアのメモリセグメントに実行情報に従って個別に記憶することを含む。

【００１２】

第1の態様および第1の態様の第1から第5の可能な実施方式の任意の1つを参照すると、第6の可能な実施方式では、少なくとも2つの処理コアを含む処理コア群は、同じオペレーティングシステムのカーネルイメージを実行し、さらに、処理コア群を構成するすべての処理コアは、同じメモリセグメントを共有する、または処理コアは、相互に異なるオペレーティングシステムのカーネルイメージを実行する。

【００１３】

第2の態様によると、本発明の実施形態は、マルチコアアーキテクチャでのリソース分離を支援するためのシステムに適用可能な、マルチコアアーキテクチャでのリソース分離を支援するための方法を提供し、ここでシステムは、プロセッサおよびメモリ空間を含む。プロセッサは、複数の処理コアを含み、複数の処理コアは、起動時一次コアおよび少なくとも1つの起動時二次コアを含む。メモリ空間は、共有メモリセグメントおよび処理コアのメモリセグメントを含む。実行情報および処理コアの実行時エントリアドレスは、共有メモリセグメントに記憶される。実行情報は、各処理コアと各オペレーティングシステムとの間のマッピング関係、ならびにメモリセグメントのサイズおよび開始アドレスを含む。処理コアの実行時エントリアドレスは、処理コアが実行する必要があるオペレーティングシステムのカーネルイメージの記憶アドレスをそれぞれのメモリセグメントで示すた

めに使用される。

【００１４】

マルチコアアーキテクチャでのリソース分離を支援するための方法は、

処理コアが実行する必要があるオペレーティングシステムのカーネルイメージを処理コアのメモリセグメントにプロセッサによって個別に記憶し、処理コアの実行時エントリアドレスを共有メモリセグメントに個別に書き込むステップと、

それぞれの実行時エントリアドレスを、起動されるべき処理コアによって個別に取得し、それぞれのメモリセグメントに記憶されたオペレーティングシステムのカーネルイメージを実行するステップとを含む。

【００１５】

第2の態様の第1の可能な実施方式では、プロセッサの処理コアは、プリセット選出アルゴリズムに従って処理コアから実行時一次コアを選出して決定し、ここで実行時一次コアは、入力/出力I/0リソースの調停制御を行うために使用される。

【００１６】

第2の態様の第1の可能な実施方式を参照すると、第2の可能な実施方式では、実行時二次コアは、I/0操作を必要とするデータを共有メモリセグメントに入れ、I/0操作に関連するメモリアドレスを実行時一次コアに通知し、その結果実行時一次コアは、外部デバイスへのI/0操作を行うためにそれ自身の局所的物理ドライバを呼び出す。

【００１７】

第2の態様の第1の可能な実施方式を参照すると、第3の可能な実施方式では、実行時二次コアは、外部デバイスへのI/0操作を行うためにそれ自身の局所的物理ドライバを呼び出す。

【００１８】

第2の態様および第2の態様の第1から第3の可能な実施方式を参照すると、第4の可能な実施方式では、ブートローダーは、起動時一次コアの内部に配置され、処理コアが実行する必要があるオペレーティングシステムのカーネルイメージを処理コアのメモリセグメントにプロセッサによって個別に記憶するステップは、

処理コアが実行する必要があるオペレーティングシステムのカーネルイメージを処理コアのメモリセグメントに実行情報に従ってブートローダーによって個別に記憶するステップを含む。

【００１９】

第2の態様および第2の態様の第1から第3の可能な実施方式を参照すると、第5の可能な実施方式では、ブートローダーは、起動時一次コアの内部に配置され、

処理コアが実行する必要があるオペレーティングシステムのカーネルイメージを処理コアのメモリセグメントにプロセッサによって個別に記憶するステップは、

実行される必要があるオペレーティングシステムのカーネルイメージをプロセッサの少なくとも1つの処理コアに割り当てられるメモリセグメントに実行情報に従ってブートローダーによって記憶し、オペレーティングシステムのカーネルイメージを実行するステップ、およびオペレーティングシステムを実行する必要があるプロセッサの他の処理コアに対応するオペレーティングシステムのカーネルイメージを他の処理コアのメモリセグメントに実行情報に従って、オペレーティングシステムを実行する処理コアによって個別に記憶するステップとを含む。

【００２０】

第2の態様および第2の態様の第1から第5の可能な実施方式を参照すると、第6の可能な実施方式では、少なくとも2つの処理コアを含む処理コア群は、同じオペレーティングシステムのカーネルイメージを実行し、さらに、処理コア群を構成するすべての処理コアは、同じメモリセグメントを共有する、または処理コアは、相互に異なるオペレーティングシステムのカーネルイメージを実行する。

【００２１】

本発明の実施形態によって提供されるマルチコアアーキテクチャでのリソース分離を支

援するための方法およびシステムでは、コア間オペレーティングシステム分離、メモリセグメント分離、およびI/0リソース分離の方式が、採用され、その結果マルチコアプロセッサの異なる処理コアで実行されるオペレーティングシステムは、互いに影響を及ぼすことなく独立して実行されてもよい。従って、マルチコアプロセッサの高集積度および低総合コストの利点は、完全に使用され、マルチコアプロセッサの障害ドメインが単一ハードディスクにとどまることが、達成され、マルチコアプロセッサは、高信頼性を有する。

【図面の簡単な説明】

【００２２】

【図１ａ】本発明の第1の実施形態によるマルチコアアーキテクチャでのリソース分離を支援するためのシステムの概略構造図である。

【図１ｂ】本発明の第1の実施形態におけるマルチコアアーキテクチャでのリソース分離を支援するためのシステムを採用するクラウド記憶システムのトポロジーの概略構造図である。

【図２】図1aで示されるマルチコアアーキテクチャでのリソース分離を支援するためのシステムでの共有メモリセグメントの部分的概略構造図である。

【図３】本発明の第2の実施形態によるマルチコアアーキテクチャでのリソース分離を支援するためのシステムの概略構造図である。

【図４ａ】本発明の第2の実施形態によるマルチコアアーキテクチャでのリソース分離を支援するためのシステムの実行時二次コアが外部デバイスにアクセスする実施の概略図である。

【図４ｂ】本発明の第2の実施形態によるマルチコアアーキテクチャでのリソース分離を支援するためのシステムの実行時二次コアが外部デバイスにアクセスする別の実施の概略図である。

【図５】本発明の第3の実施形態によるマルチコアアーキテクチャでのリソース分離を支援するための方法のフローチャートである。

【発明を実施するための形態】

【００２３】

第1の実施形態

図1aは、本発明の第1の実施形態によるマルチコアアーキテクチャでのリソース分離を支援するためのシステムの概略構造図である。図1aで示されるマルチコアアーキテクチャでのリソース分離を支援するためのシステム10は、大規模統合リソースプールに基づくクラウド記憶システムに適用可能である。図1bは、本発明の第1の実施形態におけるマルチコアアーキテクチャでのリソース分離を支援するためのシステムを採用するクラウド記憶システムのトポロジーの概略構造図である。図1bで示されるように、クラウド記憶システム100は、スイッチ102、マルチコアアーキテクチャでのリソース分離を支援するための少なくとも1つのシステム10、および少なくとも1つのアプリケーションサーバー101を含む。アプリケーションサーバー101は、スイッチ102に接続され、マルチコアアーキテクチャでのリソース分離を支援するためのシステム10は、スイッチ102に接続される。

【００２４】

図1aで示されるように、この実施形態におけるマルチコアアーキテクチャでのリソース分離を支援するためのシステム10は、プロセッサ11およびメモリ空間12を含む。

【００２５】

プロセッサ11は、複数の処理コア、すなわち起動時一次コア111および少なくとも1つの起動時二次コア112を含む。

【００２６】

メモリ空間12は、共有メモリセグメント123、起動時一次コア111のメモリセグメント121および起動時二次コア112のメモリセグメント122を含む。実行情報および処理コアの実行時エントリアドレスは、共有メモリセグメント123に記憶される。実行情報は、各処理コアと各オペレーティングシステムとの間のマッピング関係、ならびにメモリセグメントのサイズおよび開始アドレスを含む。処理コアの実行時エントリアドレスは、処理コアが

実行する必要があるオペレーティングシステムのカーネルイメージの記憶アドレスをそれぞれのメモリセグメントで示すために使用される。

【００２７】

プロセッサ11は、処理コアが実行する必要があるオペレーティングシステムのカーネルイメージを処理コアのメモリセグメントに個別に記憶し、処理コアの実行時エントリアドレスを共有メモリセグメント123に書き込む。起動されるべき処理コアは、それぞれの実行時エントリアドレスを個別に取得し、それぞれのメモリセグメントに記憶されたオペレーティングシステムのカーネルイメージを実行する。

【００２８】

具体的には、図1aで示されるように、プロセッサ11は、複数の処理コアを含むマルチコアプロセッサである。例えば、ARMまたはIntel ATOMプロセッサが、採用されてもよいが、しかし本発明は、それらに限定されない。プロセッサ11は、複数の処理コアを含み、図1aは、プロセッサ11が起動時一次コア111および起動時二次コア112を含む状況の例を示す。しかしながら、プロセッサ11は、1つまたは複数の起動時二次コアを含んでもよく、詳細は、再び本明細書では述べられない。起動時一次コア111は、ハードウェアによってあらかじめ決められた処理コアを指し、ブートローダー1111で構成される。さらに、起動時一次コア111は、他の処理コアが起動し、実行するようにトリガーする能力がある。起動時二次コア112は、プロセッサ11が起動されるときに起動時一次コア111によってトリガーされ、起動される別の処理コアを指す。具体的適用では、起動時一次コアの内部に配置されるブートローダー1111、例えばUBootは、デバイス初期化を実施してもよく、またオペレーティングシステムのカーネルイメージを、オペレーティングシステムを実行する必要がある処理コアのメモリセグメントに記憶してもよい。図1aは、起動時一次コア111とオペレーティングシステム1との間、および起動時二次コア112とオペレーティングシステム2との間のマッピング関係を示し、ここでオペレーティングシステム1およびオペレーティングシステム2は、LINUX（登録商標）、Windows（登録商標）、Solaris、DOS、OS/2、UNIX（登録商標）、XENIX、Netware、または同じオペレーティングシステムの異なるバージョンであってもよい。加えて、オペレーティングシステム1およびオペレーティングシステム2は、同一のオペレーティングシステムであってもよく、また同じタイプのオペレーティングシステムの異なるバージョン、または異なるタイプのオペレーティングシステムであってもよいが、しかし本発明は、それらに限定されない。

【００２９】

図2は、図1aで示されるマルチコアアーキテクチャでのリソース分離を支援するためのシステムでの共有メモリセグメントの部分的概略構造図である。図1aおよび図2で示されるように、実行情報123a、起動時二次コア112の実行時エントリアドレス123b、および起動時一次コア111の実行時エントリアドレス123cは、共有メモリセグメント123に記憶される。実行情報123aは、メモリセグメントの計画情報を記録し、各処理コアと各オペレーティングシステムとの間のマッピング関係、ならびにメモリセグメントのサイズおよび開始アドレスを少なくとも含む、すなわち、次の情報、(1)起動時一次コア111がオペレーティングシステム1を実行すること、ならびに起動時一次コア111に割り当てられるメモリセグメント121のサイズおよび開始アドレス、(2)起動時二次コア112がオペレーティングシステム2を実行すること、ならびに起動時二次コア112に割り当てられるメモリセグメント122のサイズおよび開始アドレス、ならびに(3)共有メモリセグメント123のサイズおよび開始アドレスを少なくとも含む。メモリセグメントは、互いに独立しており、互いに分離され、それによって互いに独立しているメモリセグメントを異なるオペレーティングシステムに割り当てる。もし各メモリセグメントの開始アドレスが、指定されないならば、初期設定により開始アドレスは、アドレス0から始まる、または前のメモリセグメントの終端アドレスから始まる。実行情報アドレス1231は、共有メモリセグメント123での実行情報123aの記憶アドレスである。処理コアは、実行情報アドレス1231の一致に従い、実行情報アドレス1231を使用することによって実行情報を問い合わせる必要がある。オプションとして、処理コアの実行情報アドレスは、各処理コアによって記憶されたオペレーティング

システムのカーネルイメージで個別に示され、実行情報は、システム実行プロセスで動的に変更され、調整されてもよい。起動時二次コア112の実行時エントリアドレスポインタ1232は、起動時二次コア112の実行時エントリアドレス123bを指し、ここで実行時エントリアドレス123bは、起動時二次コア112が実行する必要があるオペレーティングシステム2のカーネルイメージの記憶アドレスをメモリセグメント122で示すために使用される。起動時一次コア111の実行時エントリアドレスポインタ1233は、起動時一次コア111の実行時エントリアドレス123cを指し、ここで実行時エントリアドレス123cは、起動時一次コア111が実行する必要があるオペレーティングシステム1のカーネルイメージの記憶アドレスをメモリセグメント121で示すために使用される。各処理コアの実行時エントリアドレスポインタは、システムが起動される前にUBootで個別にプリセットされる。

【００３０】

本発明の実施形態によって提供されるマルチコアアーキテクチャでのリソース分離を支援するためのシステムでは、コア間オペレーティングシステム分離およびメモリセグメント分離の方式が、採用され、その結果マルチコアプロセッサの異なる処理コアで実行されるオペレーティングシステムは、互いに影響を及ぼすことなく独立して実行されてもよい。単一処理コアまたはその上で実行されるオペレーティングシステムが、機能しなくなるとき、他の処理コアまたは他の処理コアで実行されるオペレーティングシステムは、影響を受けないことになる。従って、マルチコアプロセッサの高集積度および低総合コストの利点は、完全に使用され、マルチコアプロセッサの障害ドメインが単一ハードディスクにとどまることが、達成され、マルチコアプロセッサは、高信頼性を有する。

【００３１】

前述の実施形態に基づいて、プロセッサ11は、処理コアが実行する必要があるオペレーティングシステムのカーネルイメージを処理コアのメモリセグメントに少なくとも次の2つの方式で記憶する。

【００３２】

方式1、ブートローダー1111は、処理コアが実行する必要があるオペレーティングシステムのカーネルイメージを処理コアのメモリセグメントに実行情報123aに従って個別に記憶する。

【００３３】

方式2、ブートローダー1111は、プロセッサの少なくとも1つの処理コアが実行する必要があるオペレーティングシステムのカーネルイメージを処理コアのメモリセグメントに実行情報123aに従って記憶し、処理コアは、オペレーティングシステムのカーネルイメージを実行し、オペレーティングシステムを実行する処理コアは、オペレーティングシステムを実行する必要がある他の処理コアに対応するオペレーティングシステムのカーネルイメージを他の処理コアのメモリセグメントに実行情報に従って個別に記憶する。

【００３４】

具体的には、実施方式2では、ブートローダー1111は、プロセッサの少なくとも1つの処理コアが実行する必要があるオペレーティングシステムのカーネルイメージをプロセッサのメモリセグメントに実行情報123aに従って記憶する。例えば、ブートローダー1111は、起動時一次コア111が実行する必要があるオペレーティングシステム1のカーネルイメージを起動時一次コア111に割り当てられるメモリセグメント121に実行情報123aに従って記憶し、次いで、起動時一次コア111は、メモリセグメント121でのオペレーティングシステム1のカーネルイメージを実行し、その後、起動時一次コア111は、処理コアが実行する必要があるオペレーティングシステムのカーネルイメージをそれぞれのメモリセグメントで広げるために、起動時二次コア112が実行する必要があるオペレーティングシステム2のカーネルイメージを起動時二次コア112のメモリセグメント122に実行情報123aに従って、実行オペレーティングシステム1を使用することによって記憶する。この実施形態は、起動時一次コア111が、オペレーティングシステムを初めに実行する処理コアとしての役割を果たすという例を説明することに留意すべきであるが、しかし当業者は、オペレーティングシステムを初めに起動して実行する処理コアが、起動時一次コア111に限定されず、プロ

セッサ11の任意の処理コアが、例えば前もって指定された方式でオペレーティングシステムを初めに起動して実行する処理コアとして決定されてもよいことを理解するはずである。

【００３５】

本発明のこの実施形態によって提供されるマルチコアアーキテクチャでのリソース分離を支援するためのシステムでは、処理コアが実行する必要があるオペレーティングシステムのカーネルイメージは、それぞれのメモリセグメントで広げられ、その結果異なる処理コアのメモリセグメントは、それぞれの処理コアが実行する必要があるオペレーティングシステムのカーネルイメージを記憶し、メモリセグメント上のリソースは互いに分離される。

【００３６】

前述の実施形態に基づいて、実行情報に記録された各処理コアと各オペレーティングシステムとの間のマッピング関係は、次の2つの関係を少なくとも含む。

【００３７】

マッピング関係1、少なくとも2つの処理コアを含む処理コア群は、同じオペレーティングシステムのカーネルイメージを実行する。オプションとして、処理コア群を構成するすべての処理コアは、同じメモリセグメントを共有する。

【００３８】

具体的には、少なくとも2つの処理コアを含む処理コア群は、同じオペレーティングシステムのカーネルイメージを実行する。処理コア群を構成するすべての処理コアは、同じメモリセグメントを共有してもよい。処理コア群の数は、1つより多くてもよく、各処理コア群によって個別に実行されるオペレーティングシステムは、互いに独立している。互いに独立しているメモリセグメントは、異なるオペレーティングシステムに割り当てられ、その結果単一オペレーティングシステムでの障害は、他の処理コア群または他の処理コア群で実行されるオペレーティングシステムに影響を及ぼさず、それによってマルチコアプロセッサの障害ドメインが単一ハードディスクにとどまることが、達成される。

【００３９】

マッピング関係2、処理コアは、相互に異なるオペレーティングシステムのカーネルイメージを実行する。

【００４０】

オプションとして、実行情報に従って、処理コア群および独立した単一処理コアがプロセッサ11に共存するという状況があってもよい。

【００４１】

前述の実施形態に基づいて、互いに独立しているオペレーティングシステムは、処理コア群または独立した単一処理コアで実行され、それによってコア間オペレーティングシステム分離を達成する。マルチコアプロセッサのI/0リソースおよびコプロセッサが、全体的に共有されることを考慮すると、処理コアが起動された後、I/0リソースの統合調停制御を行うために処理コアから1つの処理コアを選択することが、必要とされる。さらに、プロセッサの処理コアは、プリセット選出アルゴリズムに従って処理コアから実行時一次コアを選出して決定する。実行時一次コアは、I/0リソースの調停制御を行うために使用される。

【００４２】

オプションとして、実行時二次コアは、I/0操作を必要とするデータを共有メモリセグメントに入れ、I/0操作に関連するメモリアドレスを実行時一次コアに通知し、その結果実行時一次コアは、外部デバイスへのI/0操作を行うためにそれ自身の局所的物理ドライバを呼び出す。

【００４３】

オプションとして、実行時二次コアは、外部デバイスへのI/0操作を行うために、それ自身の局所的物理ドライバを呼び出す。

【００４４】

マルチコアアーキテクチャでのリソース分離を支援するための前述のシステムの具体的実施は、具体的実施形態を使用することによって詳細に述べられる。

【００４５】

第2の実施形態

図3は、本発明の第2の実施形態によるマルチコアアーキテクチャでのリソース分離を支援するためのシステムの概略構造図である。この実施形態は、前述の実施形態に基づいて実施される。図3で示されるように、この実施形態におけるマルチコアアーキテクチャでのリソース分離を支援するためのシステム20は、プロセッサ21およびメモリ空間22を含む。プロセッサ21は、起動時一次コア111、起動時二次コア112、起動時二次コア211、および起動時二次コア212を含む。ブートローダー1111は、起動時一次コア111の内部に配置される。加えて、起動時一次コア111および起動時二次コア112は、オペレーティングシステム1およびオペレーティングシステム2をそれぞれ実行する。起動時二次コア211および起動時二次コア212は、処理コア群214を構成する。処理コア群214は、同じオペレーティングシステム3を実行する。起動時一次コア111は、メモリセグメント121に対応し、起動時二次コア112は、メモリセグメント122に対応し、処理コア群214は、メモリセグメント221に対応し、すべてのこれらの情報は、実行情報に記録される。

【００４６】

具体的には、実行情報は、例えば次の情報、(1)起動時一次コア111がオペレーティングシステム1を実行すること、ならびに起動時一次コア111に割り当てられるメモリセグメント121のサイズおよび開始アドレス、(2)起動時二次コア112がオペレーティングシステム2を実行すること、ならびに起動時二次コア112に割り当てられるメモリセグメント122のサイズおよび開始アドレス、ならびに(3)処理コア群214がオペレーティングシステム3を実行すること、ならびに処理コア群214に割り当てられるメモリセグメント221のサイズおよび開始アドレスを記録する。メモリ22では、メモリセグメントは、メモリセグメント121、メモリセグメント122、メモリセグメント221、および共有メモリセグメント123の間で分離される。

【００４７】

システムが起動された後、各処理コアは、スピン状態にあり、ここでスピン状態は、処理コアがアイドリングを保ち、外部事象によって起こされるのを待つということを指す。ブートローダー1111がデバイス初期化を完了した後、ブートローダー1111を使用することによって、プロセッサ21は、各処理コアと各オペレーティングシステムとの間のマッピング関係、ならびに実行情報に記録されるメモリセグメントのサイズおよび開始アドレスに従って、処理コアが実行する必要があるオペレーティングシステムのカーネルイメージをプロセッサのメモリセグメントにブートローダーモードまたはダウンロードモードを使用することによって個別に記憶し、処理コアの実行時エントリアドレスを共有メモリセグメント123に個別に書き込む。

【００４８】

この実施形態では、オペレーティングシステムを初めに起動して実行する処理コアは、起動時一次コア111であるようにプリセットされるが、しかし本発明は、それに限定されず、オペレーティングシステムを初めに起動して実行する処理コアはまた、プロセッサ11の別の処理コアであるようにプリセットされてもよい。

【００４９】

起動時一次コア111は、それ自身の実行時エントリアドレスポインタを読み出し、有効な物理アドレスがあるかどうかを確認し、もしそうなら、実行時エントリアドレスポインタが指す起動時一次コア111の実行時エントリアドレスにジャンプし、オペレーティングシステム1を実行するために、オペレーティングシステム1のカーネルイメージを実行する。起動時一次コア111は、コア間通信機構を使用することによって起動されるべき処理コアを起こす。起動されるべき処理コアは、それぞれの実行時エントリアドレスポインタを個別に読み出し、有効な物理アドレスがあるかどうかを確認し、もしそうなら、それぞれの実行時エントリアドレスポインタが指す実行時エントリアドレスにジャンプし、対応す

るオペレーティングシステムを実行するために、対応するオペレーティングシステムのカーネルイメージを実行する。実行プロセスでは、処理コアは、実行情報を読み出すために実行情報アドレスにジャンプし、それぞれの処理コアが実行する必要があるオペレーティングシステム、ならびに割り当てられるメモリセグメントのサイズおよび開始アドレスを学習する。従って、処理コアは、共有メモリセグメント123およびそれぞれのメモリセグメントへの操作を行うことができるだけである。従って、この実施形態におけるマルチコアアーキテクチャでのリソース分離を支援するためのシステム20では、互いに独立しているオペレーティングシステムは、プロセッサの異なる処理コアまたは処理コア群で実行される。さらに、システムが、起動されるときだけ、起動時一次コア111で動くブートローダー1111は、各処理コアまたは処理コア群が実行する必要があるオペレーティングシステムのカーネルイメージを処理コアまたは処理コア群のメモリセグメントに記憶する。次いで、各処理コアは、他の処理コアの正常な実行に影響を及ぼすことなく、独立して再起動され、休止され、または終了されてもよい。特に、単一処理コアまたはそのオペレーティングシステムが、機能しなくなるとき、他の処理コアまたは他の処理コアで実行されるオペレーティングシステムは、影響を受けず、それによってコア間オペレーティングシステム分離を達成する。

【００５０】

オプションの実施シナリオは、独立して再起動されるとき、処理コアは、リセットされる必要があるだけであり、対応する実行時エントリアドレスを取得するために対応する実行時エントリアドレスポインタを読み出し、他の処理コアのオペレーティングシステムに依存することなく処理コアのメモリセグメントに記憶されたオペレーティングシステムのカーネルイメージを実行するということである。

【００５１】

別のオプションの実施シナリオは、システム実行プロセスにおいて、ある処理コアのために、他の処理コアが、システム要件に従って実行情報を変更してもよく、処理コアは、実行情報が変更されたことを学習した後リセットされ、再起動されるとき、処理コアは、実行情報アドレスでの新しい実行情報を読み出すことによって新しい実行情報を学習し、実行時エントリアドレスポインタでの実行時エントリアドレスを読み出すことによって再指定されるオペレーティングシステムを実行するということである。例えば、システムが起動された直後に、マルチコアプロセッサの処理コアaは、メモリセグメントをすでに占有しており、処理コアaでオペレーティングシステムAを実行し、処理コアbは、スピン状態にあり、処理コアcは、システム実行状態に従って実行情報を変更してもよく、変更された実行情報は、処理コアaおよび処理コアbが、処理コア群Tを構成し、処理コア群Tで同じオペレーティングシステムBを実行すること、処理コア群Tに割り当てられるメモリセグメントのサイズ、およびメモリセグメントの開始アドレスを再指定する。処理コアaが、実行情報が変えられたことを学習した後、オプションとして、処理コアaは、処理コアaで実行されるオペレーティングシステムAを初めにアンロードし、占有されたメモリセグメントを開放し、処理コアaは、再起動される。処理コア群T(処理コアaおよび処理コアb)は、変更された実行情報に従ってオペレーティングシステムBを実行し、新しいメモリセグメントを占有する。

【００５２】

プロセッサ21の処理コアが起動された後、互いに独立しているオペレーティングシステムは、処理コアまたは処理コア群で実行され、それによってメモリセグメント分離およびコア間オペレーティングシステム分離を達成する。マルチコアプロセッサのI/Oリソースおよびコプロセッサが、全体的に共有されることを考慮すると、処理コアが起動された後、プロセッサ21の処理コアから1つの実行時一次コアを選択することが、必要とされる。実行時一次コアは、プロセッサ21の処理コアが起動されるとき、処理コアで集中制御および調停を行うことができる処理コアを指す。実行時一次コアは、I/Oリソースの統合調停制御に関与する。具体的には、実行時一次コアは、プロセッサ21の処理コアによって処理コアからプリセット選出アルゴリズムに従って選出されて決定される。図3で示されるよ

うに、プロセッサ21の起動時一次コア111、起動時二次コア112、起動時二次コア211、および起動時二次コア212はすべて、実行状態にあり、実行状態にある4つの処理コアは、クラスタを構成すると仮定される。このクラスタでは、4つの処理コアは、実行時一次コアを選出して決定するために、コア間通信を使用することによって選出アルゴリズム、例えばPAXOSアルゴリズムを実行する。例えば、選出結果は、起動時一次コア111が実行時一次コアとして選出され、他の処理コアが実行時二次コアであるということである。マルチコアネゴシエーションを使用することによってクラスタによって選出される実行時一次コアは、起動時一次コア111に限定されず、任意の処理コアであってもよいことに留意すべきである。この実施形態は単に、起動時一次コアが、選出された実行時一次コアの役割を果たすという例を示すだけであり、本発明は、それに限定されない。選出された実行時一次コアが、機能しなくなるとき、処理コアのクラスタは、新しい実行時一次コアを再び選出する。実行時一次コアの一意性は、選出アルゴリズム、例えばPAXOSアルゴリズムによって確保される。

【００５３】

実行時一次コアは、I/0リソースの統合調停制御に関与し、すべてのI/0リソースおよびコプロセッサリソースの使用は、実行時一次コアによって承認される必要がある。

【００５４】

実行時二次コアが、外部デバイス23にアクセスする必要があるとき、実行時二次コアは、リソース使用要求を実行時一次コアに送る。リソース使用要求に従って、実行時一次コアは、システムリソース使用条件およびリソース割り当てポリシーに基づく排他性使用リソース調停の後、リソース要求応答を実行時二次コアに送る。リソース要求応答を受け取った後、実行時二次コアは、I/0リソースの使用承認を取得し、外部デバイス23へのI/0操作を行うことを始める。

【００５５】

システムリソース使用条件およびリソース割り当てポリシーが、許可するとき、実行時一次コアは、外部デバイスへのI/0操作を行うためにそれ自身の局所的物理ドライバを呼び出してもよい。各実行時二次コアは、少なくとも次の2つの方式で外部デバイス23にアクセスするためにI/0リソースを使用してもよい。

【００５６】

方式1、実行時二次コアは、I/0操作を必要とするデータを共有メモリセグメントに入れ、I/0操作に関連するメモリアドレスを実行時一次コアに通知し、その結果実行時一次コアは、外部デバイスへのI/0操作を行うためにそれ自身の局所的物理ドライバを呼び出す。

【００５７】

方式2、実行時二次コアは、外部デバイスへのI/0操作を行うためにそれ自身の局所的物理ドライバを呼び出す。

【００５８】

図4aは、本発明の第2の実施形態によるマルチコアアーキテクチャでのリソース分離を支援するためのシステムの実行時二次コアが外部デバイスにアクセスする実施の概略図であり、図4bは、本発明の第2の実施形態によるマルチコアアーキテクチャでのリソース分離を支援するためのシステムの実行時二次コアが外部デバイスにアクセスする別の実施の概略図である。

【００５９】

図4aで示されるように、実行時二次コア402は、I/0操作を必要とするデータを共有メモリセグメントに入れ、I/0操作に関連するメモリアドレスを実行時一次コア401に通知し、その結果実行時一次コア401は、外部デバイス23へのI/0操作を行うために、それ自身の局所的物理ドライバ4011を呼び出す。実行時二次コア402が外部デバイス23へのI/0操作を行うことは、エージェントとしての実行時一次コア401によって実施される。実行時二次コア402は、このI/0操作を必要とするデータを共有メモリセグメントにメモリを共有する仕方で入れ、I/0操作に関連するメモリアドレスを実行時一次コア401に通知してもよい。実

行時一次コア401は、実行時二次コア402によって送られる通知情報を受け取り、局所的物理ドライバ4011を呼び出し、外部デバイス23にアクセスするために、データを外部デバイス23にI/0リソースを用いて送る。例えば、実行時一次コア401は、外部デバイス23へのI/0操作を行い、アクセスが完了した後、I/0リソースおよび外部デバイス23の関連するリソースを解放する。

【００６０】

図4bで示されるように、実行時二次コア402は、それ自身の局所的物理ドライバ4021を呼び出し、外部デバイス23へのI/0操作を行う。実行時二次コア402が、外部デバイス23にアクセスする必要があるとき、実行時一次コア401は、実行時二次コア402がI/0リソースを制御および調停の方式で使用することを承認する。実行時二次コア402は、実行時一次コア401によって送られるリソース要求応答に従って直線的方式で割り当てられる外部デバイスにアクセスし、それは、外部デバイス23へのI/0直線的操作を行うこと、および局所的物理ドライバ4021を呼び出すことによってI/0リソース操作を直接行うことを含む。すなわち、各処理コアは、直線的方式で外部デバイス23へのI/0操作を行い、操作が完了した後、I/0リソースの制御を実行時一次コア401に開放する、すなわち、外部デバイス23へのI/0操作が完了したことを実行時一次コア401に通知する。実行時一次コア401は、リソースを他の処理コアに開放するために、通知に従ってI/0リソースおよび外部デバイス23の関連するリソースを解放する。

【００６１】

オプションとして、状態同期情報は、実行時一次コアと実行時二次コアとの間で維持される。実行時一次コアが、ある処理コアと連絡を取れないということを見いだすとき、もしこの時に、実行時一次コアが、連絡を取れない処理コアに外部デバイスリソースをすでに割り当てているならば、実行時一次コアは、外部デバイスリソースを取り戻す。本発明の前述の実施形態によって提供されるマルチコアアーキテクチャでのリソース分離を支援するためのシステムでは、実行時一次コアは、各処理コアが外部デバイスにアクセスするためにI/0リソースを使用するように、I/0の統合調停制御に関与する。本発明の前述の実施形態によって提供されるマルチコアアーキテクチャでのリソース分離を支援するためのシステムはまた、各処理コアが共通機能部にアクセスするためにも使用されることに留意すべきである。共通機能部は、コプロセッサリソースを少なくとも含んでもよい。共有コプロセッサリソースの調停適用の詳細な実施方式については前述の実施形態の説明を参照されたく、詳細は、再び本明細書では述べられない。

【００６２】

本発明の実施形態によって提供されるマルチコアアーキテクチャでのリソース分離を支援するためのシステムでは、実行時一次コアは、I/0リソースの統合調停制御に関与するように処理コアから選出され、それによってI/0リソース分離を達成する。各処理コアまたは処理コア群で実行されるオペレーティングシステムが、互いに独立して実行される状況では、マルチコアプロセッサでの統合I/0能力およびコプロセッサスケジュール制御が、達成され、それによってマルチコアプロセッサの障害ドメインが単一ハードディスクにとどまることを確実にする。

【００６３】

第3の実施形態

図5は、本発明の第3の実施形態によるマルチコアアーキテクチャでのリソース分離を支援するための方法のフローチャートである。この実施形態の方法は、マルチコアアーキテクチャでのリソース分離を支援するためのシステムに適用可能であり、ここでシステムは、プロセッサおよびメモリ空間を含む。プロセッサは、複数の処理コアを含み、複数の処理コアは、起動時一次コアおよび少なくとも1つの起動時二次コアを含む。メモリ空間は、共有メモリセグメントおよび処理コアのメモリセグメントを含む。実行情報および処理コアの実行時エントリアドレスは、共有メモリセグメントに記憶される。実行情報は、各処理コアと各オペレーティングシステムとの間のマッピング関係、ならびにメモリセグメントのサイズおよび開始アドレスを含む。処理コアの実行時エントリアドレスは、処理コ

アが実行する必要があるオペレーティングシステムのカーネルイメージの記憶アドレスをそれぞれのメモリセグメントで示すために使用される。図5で示されるように、この実施形態での方法は、次のステップを含む。

【００６４】

501、プロセッサは、処理コアが実行する必要があるオペレーティングシステムのカーネルイメージを処理コアのメモリセグメントに個別に記憶し、処理コアの実行時エントリアドレスを共有メモリセグメントに個別に書き込む。

【００６５】

502、起動されるべき処理コアは、それぞれの実行時エントリアドレスを取得し、それぞれのメモリセグメントに記憶されたオペレーティングシステムのカーネルイメージを実行する。

【００６６】

オプションとして、プロセッサの処理コアは、プリセット選出アルゴリズムに従って処理コアから実行時一次コアを選出して決定し、ここで実行時一次コアは、入力/出力I/0リソースの調停制御を行うために使用される。

【００６７】

オプションとして、実行時二次コアは、I/0操作を必要とするデータを共有メモリセグメントに入れ、I/0操作に関連するメモリアドレスを実行時一次コアに通知し、その結果実行時一次コアは、外部デバイスへのI/0操作を行うためにそれ自身の局所的物理ドライバを呼び出す。

【００６８】

オプションとして、実行時二次コアは、外部デバイスへのI/0操作を行うために、それ自身の局所的物理ドライバを呼び出す。

【００６９】

オプションとして、ブートローダーは、起動時一次コアの内部に配置され、プロセッサが、処理コアが実行する必要があるオペレーティングシステムのカーネルイメージを処理コアのメモリセグメントに個別に記憶することは、ブートローダーが、処理コアが実行する必要があるオペレーティングシステムのカーネルイメージを処理コアのメモリセグメントに実行情報に従って個別に記憶することを含む。

【００７０】

オプションとして、ブートローダーは、起動時一次コアの内部に配置され、プロセッサが、処理コアが実行する必要があるオペレーティングシステムのカーネルイメージを処理コアのメモリセグメントに個別に記憶することは、ブートローダーが、実行される必要があるオペレーティングシステムのカーネルイメージをプロセッサの少なくとも1つの処理コアに割り当てられるメモリセグメントに実行情報に従って記憶し、オペレーティングシステムのカーネルイメージを実行すること、およびオペレーティングシステムを実行する処理コアが、オペレーティングシステムを実行する必要があるプロセッサの他の処理コアに対応するオペレーティングシステムのカーネルイメージを他の処理コアのメモリセグメントに実行情報に従って個別に記憶することを含む。

【００７１】

オプションとして、少なくとも2つの処理コアを含む処理コア群は、同じオペレーティングシステムのカーネルイメージを実行し、さらに、処理コア群を構成するすべての処理コアは、同じメモリセグメントを共有する、または処理コアは、相互に異なるオペレーティングシステムのカーネルイメージを実行する。

【００７２】

本発明の実施形態によって提供されるマルチコアアーキテクチャでのリソース分離を支援するための方法では、コア間オペレーティングシステム分離、メモリセグメント分離、およびI/0リソース分離の方式が、採用され、その結果マルチコアプロセッサの異なる処理コアで実行されるオペレーティングシステムは、互いに影響を及ぼすことなく独立して実行されてもよい。従って、マルチコアプロセッサの高集積度および低総合コストの利点

は、完全に使用され、マルチコアプロセッサの障害ドメインが単一ハードディスクにとどまることが、達成され、マルチコアプロセッサは、高信頼性を有する。

【００７３】

当業者は、本方法実施形態のステップのすべてまたは一部が、プログラムが関連ハードウェアに指示することによって実施されてもよいことを理解することができる。プログラムは、コンピュータ可読記憶媒体に記憶されてもよい。プログラムが、実行されるとき、本方法実施形態のステップは、行われる。前述の記憶媒体は、ROM、RAM、磁気ディスク、または光ディスクなどの、プログラムコードを記憶することができる任意の媒体を含む。

【００７４】

最後に、前述の実施形態は、本発明を限定すること以外に本発明の技術的解決策を説明することを単に目的としていることに留意すべきである。本発明は、前述の実施形態を参照して詳細に述べられるが、当業者は、本発明の実施形態の技術的解決策の範囲から逸脱することなく、前述の実施形態で述べられた技術的解決策に対して変更をなお行うことができるまたはそれのいくつかのもしくはすべての技術的特徴に対して等価な置換を行うことができることを理解するはずである。

【符号の説明】

【００７５】

1　オペレーティングシステム
2　オペレーティングシステム
3　オペレーティングシステム
10　システム
11　プロセッサ
12　メモリ空間
20　システム
21　プロセッサ
22　メモリ空間
23　外部デバイス
100　クラウド記憶システム
101　アプリケーションサーバー
102　スイッチ
111　起動時一次コア
112　起動時二次コア
121　メモリセグメント
122　メモリセグメント
123　共有メモリセグメント
123a　実行情報
123b　実行時エントリアドレス
123c　実行時エントリアドレス
211　起動時二次コア
212　起動時二次コア
214　処理コア群
221　メモリセグメント
401　実行時一次コア
402　実行時二次コア
1111　ブートローダー
1231　実行情報アドレス
1232　実行時エントリアドレスポインタ
1233　実行時エントリアドレスポインタ
4011　局所的物理ドライバ
4021　局所的物理ドライバ

【図１ａ】

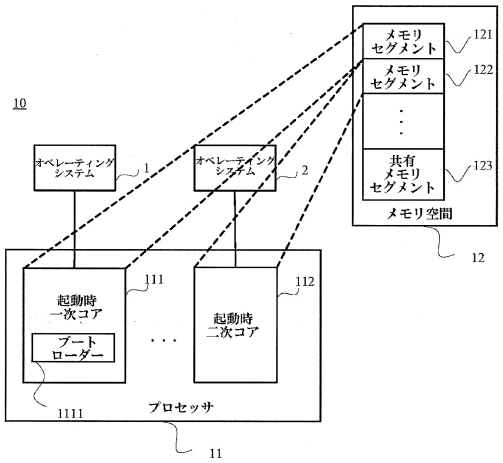

FIG. 1a

【図１ｂ】

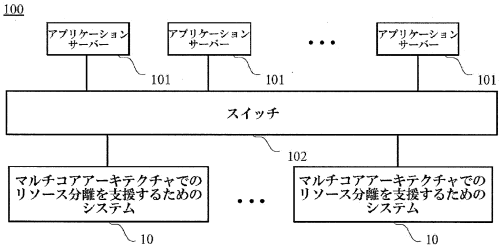

FIG. 1b

【図２】

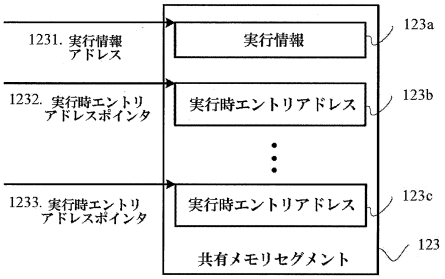

FIG. 2

【図３】

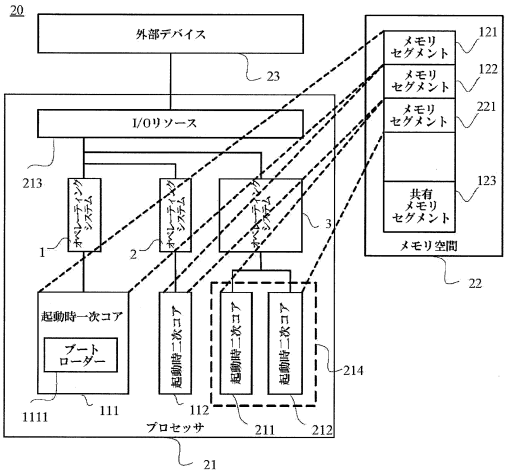

FIG. 3

【図４ａ】

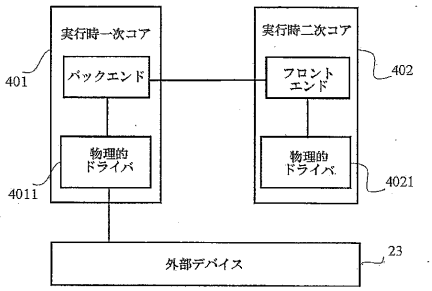

FIG. 4a

【図４ｂ】

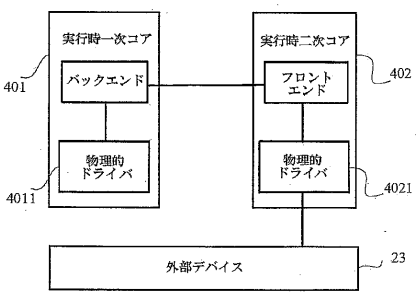

FIG. 4b

【図５】

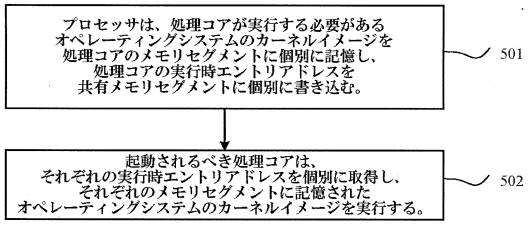
プロセッサは、処理コアが実行する必要がある
オペレーティングシステムのカーネルイメージを
処理コアのメモリセグメントに個別に記憶し、
処理コアの実行時エントリアドレスを
共有メモリセグメントに個別に書き込む。
501
起動されるべき処理コアは、
それぞれの実行時エントリアドレスを個別に取得し、
それぞれのメモリセグメントに記憶された
オペレーティングシステムのカーネルイメージを実行する。
502

FIG. 5

フロントページの続き

(72)発明者　雷　▲暁▼松
中華人民共和国５１８１２９広東省深▲セン▼市龍岡区坂田華為本社ビル

審査官　田中　幸雄

(56)参考文献　国際公開第２０１０／０９７９２５（WO，Ａ１）
特開２０１０－１９１８８６（ＪＰ，Ａ）
国際公開第２０１１／０３９８８７（WO，Ａ１）
特開２０１３－１２２５０（ＪＰ，Ａ）
特開平６－２９５２８９（ＪＰ，Ａ）
米国特許出願公開第２００６／０２０６９０４（ＵＳ，Ａ１）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　　９／４６
Ｇ０６Ｆ　１５／１７７