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

特許第5001773号
(P5001773)

(45)発行日　平成24年8月15日(2012.8.15)　(24)登録日　平成24年5月25日(2012.5.25)

(51)Int.Cl.　F I
G06F 9/445 (2006.01)　G06F 9/06 610J
G06F 9/46 (2006.01)　G06F 9/46 350

請求項の数 17　外国語出願　(全 13 頁)

(21)出願番号　特願2007-257926 (P2007-257926)
(22)出願日　平成19年10月1日 (2007.10.1)
(65)公開番号　特開2008-135009 (P2008-135009A)
(43)公開日　平成20年6月12日 (2008.6.12)
審査請求日　平成19年10月1日 (2007.10.1)
(31)優先権主張番号　11/541,242
(32)優先日　平成18年9月29日 (2006.9.29)
(33)優先権主張国　米国 (US)

前置審査

(73)特許権者　591003943
インテル・コーポレーション
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
(74)代理人　110000877
龍華国際特許業務法人
(72)発明者　ロスマン、マイケル　エー.
アメリカ合衆国、９８３７４　ワシントン州、ピュアラップ、１８３アールディー　ストリート　イースト　１１９０５
(72)発明者　ジマー、ビンセント　ジェイ.
アメリカ合衆国、９８００３　ワシントン州、フェデラル　ウェイ、サウス　３６９　ティーエイチ　ストリート　１９３７

最終頁に続く

(54)【発明の名称】プラットフォームのブート効率を増加するシステム、方法および媒体

(57)【特許請求の範囲】
【請求項１】
メモリに結合される少なくとも１つのプロセッサを有する一のプラットフォームと、
一の第１のパーティションにおいて前記少なくとも１つのプロセッサ上で実行される一のブートキャッシュエージェントと、
一の第２のパーティションにおいて実行される一のオペレーティングシステムと、
を含み、
前記第１のパーティションは、一のブートターゲット媒体へのアクセスを制御し、
前記第２のパーティションは、前記第１のパーティションを介して前記ブートターゲット媒体へのアクセスを有し、
ブート時に、前記ブートキャッシュエージェントは、ブートターゲットデータを、前記ブートターゲット媒体より高速のアクセスを有する一の高速メモリに選択的に保存し、
前記ブートキャッシュエージェントは、前記第２のパーティションにおける一のオペレーティングシステムのリセットまたはリブート時に、前記高速メモリから前記保存されたブートターゲットデータを選択的に取り出し、
前記高速メモリは、揮発性システムメモリであり、
前記第１のパーティション以外の一のオペレーティングシステムは、前記高速メモリ内の前記保存されたブートターゲットデータへのアクセスを有さず、
前記プラットフォームは、前記プラットフォーム上のチップセットのリソースをパーティショニングするチップセットパーティショニングが可能にされ、

前記ブートキャッシュエージェントは、前記ブートターゲット媒体へのデバイスアクセスを自動的にインターセプトする一の特権的パーティションである組み込みパーティション内にある、ブート時間を加速するシステム。

【請求項２】

前記チップセットは、一の選択パーティションに対して専用の一のリソースを可能にし、

パーティショニングは、ブート時にプラットフォームファームウェアにより定義される請求項１に記載のシステム。

【請求項３】

前記ブートキャッシュエージェントは、保存されたデータセクタを識別するために前記ブートターゲット媒体から取り出しされた複数のセクタのインデックスを保存する請求項１に記載のシステム。

【請求項４】

前記ブートキャッシュエージェントは、一のリクエストされたセクタは保存されているか否かを判断し、保存されていない場合は、前記リクエストされたセクタを前記ブートターゲット媒体から取り出しし、保存されている場合は、前記ブートキャッシュエージェントは、前記リクエストされたセクタを前記保存されたメモリから取り出しする請求項２に記載のシステム。

【請求項５】

前記プラットフォームは、仮想化が可能にされ、

前記ブートキャッシュエージェントは、一の仮想マシンモニタ（ＶＭＭ）内にある請求項１から４の何れか１項に記載のシステム。

【請求項６】

前記ＶＭＭは、前記ブートターゲット媒体へのデバイスアクセスを仮想化する請求項５に記載のシステム。

【請求項７】

前記ブートキャッシュエージェントは、保存されたデータセクタを識別するために前記ブートターゲット媒体から取り出しされた複数のセクタのインデックスを保存する請求項６に記載のシステム。

【請求項８】

前記ブートキャッシュエージェントは、一のリクエストされたセクタは保存されているか否かを判断し、保存されていない場合は、前記リクエストされたセクタを前記ブートターゲット媒体から取り出しし、保存されている場合は、前記リクエストされたセクタを前記保存されたメモリから取り出しする請求項５に記載のシステム。

【請求項９】

前記ブートキャッシュエージェントは、一のターゲットポリシーに基づいて前記ブートターゲット媒体からの複数のセクタを選択的に保存する請求項１から８の何れか１項に記載のシステム。

【請求項１０】

ブート時に、一のブートキャッシュエージェントによって、一のブートターゲット媒体から取り出しされるブートターゲットデータを選択的に保存する工程と、

一のオペレーティングシステムのリブート時に、前記保存されたブートターゲットデータおよび前記ブートターゲット媒体のうちの少なくとも１つから前記ブートターゲットデータを取り出しする工程と、

を含み、

前記ブートターゲットデータを選択的に保存する工程において、前記ブートターゲットデータは、前記ブートターゲット媒体より高速の一の高速メモリ内に保存され、

前記取り出し工程は、前記ブートキャッシュエージェントにより制御され、

前記高速メモリは、揮発性システムメモリであり、

前記ブートキャッシュエージェントは、一の第１のプラットフォームパーティションに

あり、
前記オペレーティングシステムは、一の第２のプラットフォームパーティションにあり、
前記第２のプラットフォームパーティションは、前記保存されたブートターゲットデータへのアクセスを有さず、
前記第１のプラットフォームパーティションは、プラットフォーム上のチップセットのリソースをパーティショニングするチップセットパーティショニングが可能にされた前記プラットフォームにおける、前記ブートターゲット媒体へのデバイスアクセスを自動的にインターセプトする一の特権的パーティションである組み込みパーティションである、ブート時間を加速し、
前記ブートターゲットデータを取り出しする工程は、
一のリクエストされたセクタは保存されているか否かを判断する工程と、
保存されていない場合は、前記リクエストされたセクタを前記ブートターゲット媒体から取り出しする工程と、
保存されている場合は、前記リクエストされたセクタを前記保存されたメモリから取り出しする工程と、
を含む
する方法。
【請求項１１】
前記第１のプラットフォームパーティションは、一の仮想マシンモニタであり、
前記第２のプラットフォームパーティションは、一の仮想マシンである請求項１０に記載の方法。
【請求項１２】
前記ブートキャッシュエージェントによって、保存されたデータセクタを識別するために前記ブートターゲット媒体から読み出しされた複数のセクタのインデックスを保存する工程をさらに含む請求項１０または１１に記載の方法。
【請求項１３】
ブートターゲットデータを選択的に保存する工程は、
一のターゲットポリシーに基づいて前記ブートターゲット媒体からの複数のセクタを選択的に保存する工程をさらに含む請求項１０から１２の何れか１項に記載の方法。
【請求項１４】
一のマシンにより実行されると該マシンに、
ブート時に、一のブートキャッシュエージェントによって、一のブートターゲット媒体から取り出しされるブートターゲットデータを選択的に保存させ、
一のオペレーティングシステムのリブート時に、前記保存されたブートターゲットデータおよび前記ブートターゲット媒体のうちの少なくとも１つから前記ブートターゲットデータを取り出しさせる複数の命令が格納されたマシン可読媒体であって、
前記ブートターゲットデータを選択的に保存させるときに、前記ブートターゲットデータは、前記ブートターゲット媒体より高速の一の高速メモリ内に保存され、
前記取り出しは、前記ブートキャッシュエージェントにより制御され、
前記高速メモリは、揮発性システムメモリであり、
前記ブートキャッシュエージェントは、一の第１のプラットフォームパーティションにあり、
前記オペレーティングシステムは、一の第２のプラットフォームパーティションにあり
前記第２のプラットフォームパーティションは、前記保存されたブートターゲットデータへのアクセスを有さず、
前記第１のプラットフォームパーティションは、プラットフォーム上のチップセットのリソースをパーティショニングするチップセットパーティショニングが可能にされた前記プラットフォームにおける、前記ブートターゲット媒体へのデバイスアクセスを自動的にインターセプトする一の特権的パーティションである組み込みパーティションであり、

前記ブートターゲットデータを取り出させる命令は、前記マシンにより実行されると、前記マシンに、
一のリクエストされたセクタは保存されているか否かを判断させ、
保存されていない場合は、前記リクエストされたセクタを前記ブートターゲット媒体から取り出しさせ、
保存されている場合は、前記リクエストされたセクタを前記保存されたメモリから取り出しさせる命令を含む、媒体。
【請求項１５】
前記第１のプラットフォームパーティションは、一の仮想マシンモニタであり、
前記第２のプラットフォームパーティションは、一の仮想マシンである請求項１４に記載の媒体。
【請求項１６】
前記マシンにより実行されると、前記マシンに、
前記ブートキャッシュエージェントによって、保存されたデータセクタを識別するために前記ブートターゲット媒体から読み出しされた複数のセクタのインデックスを保存させる命令をさらに含む請求項１４または１５に記載の媒体。
【請求項１７】
ブートターゲットデータの選択的な保存は、前記マシンにより実行されると、前記マシンに、
一のターゲットポリシーに基づいて前記ブートターゲット媒体からの複数のセクタを選択的に保存させる命令をさらに含む請求項１４から１６のいずれか１項に記載の媒体。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明の一実施形態は、一般的にコンピューティングプラットフォームに係り、より具体的には、仮想化またはパーティショニング技法の使用によるブート時間の減少に関する。一実施形態では、ハイパーバイザ／プラットフォームパーティションが、プラットフォームを制御し、残りのメインパーティションの初期化を許可する。
【背景技術】
【０００２】
プラットフォームのブート効率を増加する、すなわち、ブート時間を減少するために存在する様々なメカニズムが、既存のシステムにおいて実施されてきている。
【０００３】
多くのプラットフォームのベンダおよびユーザは、プラットフォームをブートするのにかかる時間の長さに対する関心を有する。ここでは、ブート時間には、プラットフォームの電源を投入して、ユーザログインプロンプトを得る、またはユーザアプリケーションを実行することができる時点まで進むことが含まれる。プラットフォームが可能な限り迅速にファームウェアパスを通り抜けることを確実にすることに関連する多くの標準がある。オペレーティングシステム（ＯＳ）を始動するのに必要な時間は、ＯＳの複雑さに基づいて異なりうる。一般的なデスクトッププラットフォームは、ＯＳを始動するためにブートするのに７．５秒かかりうる。実際の標準は、業界全体により左右される。このことは、非常に可用性の高いシステムに対応するためのサーバ環境においても明らかである。ブート時間が短いほど、メンテナンスまたはクラッシュ後にサーバをより早く使用することができるようになる。より迅速なブート時間は、高い可用性／信頼性を必要とするシステム、または、各ユーザセッション後にシステムがリブートされるインターネットカフェにおいて非常に望ましい。
【０００４】
本発明の特徴および利点は、本発明の以下の詳細な説明から明らかとなろう。
【発明を実施するための最良の形態】
【０００５】

本発明の一実施形態は、ブートキャッシュエージェントを使用してブート時間を加速することに関するシステムおよび方法である。少なくとも１つの実施形態では、本発明は、選択されたブートターゲットデータを、一般的には揮発性システムメモリである高速メモリにキャッシュし、リブートまたはリセット時には、低速ブートターゲットメモリからデータを読み出しするのではなくキャッシュされたブートターゲットデータにアクセスするようリクエストをインターセプトすることを意図する。

【０００６】

明細書中、本発明の「一実施形態」との参照は、実施形態に関連して説明する特定の機能、構造、または特徴が、本発明の少なくとも１つの実施形態に含まれることを意味する。したがって、明細書全体において様々な箇所に出現する「一実施形態では」との表現の表現は、必ずしもすべてが同一の実施形態を参照しているわけではない。

【０００７】

説明目的のために、特定の構成および詳細を、本発明を完全に理解することができるよう示している。しかし、当業者には、本発明の実施形態は、本願に示す特定の詳細なしで実施しうることは明らかであろう。さらに、周知の機能は、本発明を曖昧にしないよう省略または簡略化されうる。様々な例をこの説明全体において与える。これらは、本発明の特定の実施形態の説明に過ぎない。本発明の範囲は、ここに与える例に限定されない。

【０００８】

一実施形態では、仮想マシンモニタ（ＶＭＭ）またはハイパーバイザを使用してより効率のよいブートを実現する。ＶＭＭは、ブート時にシステムを追跡しうる。ＶＭＭは、ハードウェアへのアクセスを仮想化し、特定の装置に対してアブストラクションを供給する。一部の装置は、ブート時にオペレーティングシステム（ＯＳ）に対して可視であるようにされうる一方で、その他の装置は、ＶＭＭによってブート時にはＯＳから隠されうる。将来のシステムでは、多くのプラットフォームは、ＶＭＭまたはプラットフォームリソースレイヤ（ＰＲＬ）により制御またはモニタリングされうる。ＶＭＭは、今日使用されるソフトウェア実施形態である。

【０００９】

別の実施形態では、プラットフォームリソースレイヤ（ＰＲＬ）は、たとえば、インテル社からの将来のシリコン製品リリースにおいて可能にされうるチップセット支援されたトポロジとして実施されうる。これらのチップセット機能は、様々な実行環境の、ハードウェア支援されるがソフトウェアプログラム可能である分離を与える能力を支援しうる。このチップセットは、従来のソフトウェアＶＭＭの多くの制御ポイントをシミュレートする適度なソリューションを提供するが、プラットフォームハードウェア／チップセットの基礎をなす様々なコンポーネントにおけるハードウェア支援を介して実現される。このチップセット支援されたトポロジは、プラットフォーム上に複数のパーティションが定義されることを可能にする。組み込みパーティションまたはシステムパーティションは、以下にさらに説明するようにブート処理を制御しうる。

【００１０】

一実施形態では、ＶＭＭの一部でありうるブートキャッシュエージェントは、ブート時にプラットフォームが行っている行為をモニタリングする。エージェントは相対的に静的である。従来の環境では、電源プラグを抜くとシステム全体は黙従し、ブートが最初から開始される。しかし、システムの一部はほとんど変化せず、最初から開始することはしばしば不必要である。システムの一部は、プラットフォームがブートしている間に知覚力を有し続ける、すなわち、常駐して運転可能であり続ける。このブート処理の間に、たとえば、ハードウェアは初期化され、ブートターゲットは決定され、ブートコードはターゲット媒体からロードされて実行される。すべてにおいて、数百メガバイトのデータが、ブートターゲット媒体から読み出しされることが必要となりうる。必要なドライバに加えてＯＳもディスクからロードされうる。

【００１１】

一実施形態では、キャッシュエージェントは、ブート処理をモニタリングし、ブートタ

ーゲットから取り出しした情報を含むブート処理の一部に関する情報をシステムメモリ（ＲＡＭ）に保存しうる。物理的またはシステム揮発性メモリは、歳月が経るにしたがって安価になってきている。したがって、数百メガバイトのデータの保存はかつてほど費用がかかるわけではない。このメモリは一般的に高速で、プラットフォーム上のプロセッサに高度にアクセス可能である。ブート処理時に、ＯＳローダは、一般的に、ハードドライブであるブートターゲットから特定量のデータおよび命令を取り出しする。ブートキャッシュエージェントは、このデータの取り出しを所定時点までモニタリングし、その情報を、システムメモリの一部にコピーまたはミラーリングしうる。この方法は、システムが再びブートされるとき、この同じ情報がほぼまたはまったく変更なしで必要となるという可能性があるという事実を利用している。レガシファームウェア実施形態では、ファームウェアは、ハードドライブ（ブートターゲット媒体）から第１のデータセクタ（すなわち、ブートレコード）をロードし、次に、オペレーティングシステムによりハードドライブ上に置かれたブートレコードが、ＯＳローダとして機能する。拡張可能ファームウェアインタフェース（ＥＦＩ）と互換性のあるアーキテクチャの実施形態では、第１のセクタがロードされるのではなく、しばしばＯＳローダとして知られるＯＳアプリケーションが始動され同様の効果を有する。ＯＳローダは続けてブートターゲット媒体からブートに必要な残りのデータをロードする。この残りのデータはしばしば前回のブートから変更されていない。

【００１２】

本発明の実施形態では、システムが次にブートされる時に、このロードされた情報は、高速でアクセス可能なメモリ内にある。したがって、ブートキャッシュエージェントは、このデータをＯＳまたはファームウェアに供給して、ハードドライブといった低速の媒体装置にアクセスしなければならないことを回避しうる。揮発性システムメモリを使用する利点は、ＲＡＭは揮発性であるものの、電源が完全に除去されない限り、一般的に初期化されない、消去されない、または信頼できないような状態にされないことである。したがって、その電源が完全に除去されていないシステムがリブートされる場合、データは、リブート時にシステムメモリから取り出しされうる。

【００１３】

ブートキャッシュエージェントは、たとえば、別個のプロセッサ、メモリなどの専用リソースを有しうる。ブート時にデータがロードおよび実行されると、ブートキャッシュエージェントは、メインＯＳは使用することのできない専用リソースにブートデータを保存しかつ専用リソースからブートデータ取り出ししうる。リブート時には、ＯＳ専用メモリは通常通り初期化されるが、これは、ブートキャッシュエージェント専用のメモリとは干渉しない。

【００１４】

図１を参照するに、本発明の実施形態による、ブートキャッシュエージェントのための例示的な方法を示す。このシステムは、工程１０１において電源が投入されるか、または、リセットリクエストが行われる。工程１０３において、初めてブートする場合など、必要である場合に、プラットフォームの基礎となる構造が初期化される。工程１０５において、プラットフォームはブートパスキャッシュをサポートするか否か判断される。サポートしない場合は、工程１０７において、プラットフォームは、ブートターゲット媒体からのブートを続行する。

【００１５】

プラットフォームが、ブートパスキャッシュをサポートする場合、工程１０９において、データの必要なルーティングが可能にされる。ＶＭＭ、ハイパーバイザ、または他の特権レイヤを有するプラットフォームの場合、ＶＭＭは、Ｉ／Ｏアクセスを仮想化し、ブートキャッシュエージェントが、ブート媒体からのデータ取り出しを制御することを可能にする。ブートキャッシュエージェントは、揮発性ストア内にセクタデータをミラーリングし、後の使用のために取り出ししたセクタにインデックスを付けることが可能にされる。チップセットパーティショニングサポートを有するプラットフォームの場合、組み込みプ

ラットフォームにあるブートキャッシュエージェントが、装置アクセスを自動的に制御し、次に、後の使用のためにパーティションメモリ内にブートデータをミラーリングすることが可能にされる。

【００１６】

工程１１１で判断されるようにメインパーティションがブートする場合は、工程１１３においてブートパスが以前にキャッシュされたか否か判断される。以前にキャッシュされた場合、ブートキャッシュエージェントは、ブートターゲットからブートデータを読み出しするリクエストをインターセプトする。リクエストされたセクタが、ブートキャッシュエージェントインデックスにある場合、そのセクタは、ブートターゲット媒体ではなく揮発性ストアから取り出しされる。インターセプションに使用される方法は、プラットフォームアーキテクチャ、すなわち、ＶＭＭまたはチップセットパーティショニング（ＰＲＬ）に依存する。

【００１７】

メインパーティション（または他のゲストＯＳ）のブートではないと判断される場合、通常の動作、すなわち、通常のブートが、工程１０７において再開されうる。

【００１８】

ブートパスは以前にキャッシュされていないと工程１１３で判断される場合、ターゲットポリシーを参照して、ブートデータがキャッシュされるべきか否か判断されうる。たとえば、一部では、プラットフォームは、マイクロソフト（登録商標）のＷｉｎｄｏｗｓ（登録商標）およびＬｉｎｕｘ（登録商標）の両方をブートすることが可能にされうる。ブートターゲットデータをキャッシュするのに使用できるメモリは制限されうるので、ユーザは、１つのオペレーティングシステムかもう１つのオペレーティングシステム用のデータだけをキャッシュするよう選択しうる。その場合、ブートデータは、リセット／ブートのために選択されるブートターゲットに基づいてキャッシュされうる。別の場合では、ブートターゲットの使用頻度が保存され、最も頻繁に使用されるブートターゲットだけがキャッシュされてもよい。その他の場合では、ブートターゲット、またはその一部は常にキャッシュされる。当業者は、異なる機能を有するプラットフォームに対してさまざまなポリシーを作成しうることは理解できよう。

【００１９】

工程１１５で決定されるように、ターゲットポリシーは、ターゲットはキャッシュできると決定する場合、ブートターゲットから取り出しされるセクタは、工程１１９において揮発性または高速ストア内にミラーリングされる。本発明の実施形態は、揮発性メモリへのデータのキャッシュを説明するが、ブートターゲットより高速である任意のメモリをキャッシュに使用して、依然として開示する方法の利点を得られることが考えられうる。リセットまたはリブートのアラートが受信されると、処理は、再び工程１１１に進み、メインパーティションはリブートされるべきか、また、データはブートターゲット媒体ではなく高速ストアから取り出しされるべきか否か判断する。

【００２０】

ターゲットポリシーが、ブートデータはキャッシュされるべきではないと決定する場合、動作は、工程１０７において通常のブートを行う。

【００２１】

ＶＭＭを実行する一実施形態では、ＶＭＭは、リクエストをリセットするインタメディアリとして機能する。ＶＭＭは、リセットリクエストをインターセプトし、リクエストをしているＯＳを実行する仮想マシンにより使用されるメモリおよびリソースの一部だけをリセットする。プラットフォームの他のコンポーネントはリセットされなくてもよい。ＰＲＬが可能にされたチップセットとともに実行する一実施形態では、チップセットがインタメディアリとして機能し、リセットリクエストをインターセプトしうる。

【００２２】

プラットフォームリソースレイヤ（ＰＲＬ）アーキテクチャ、または、組み込みパーティションアーキテクチャでは、プラットフォームの様々なコンポーネントは、プロセッサ

、メモリ、および他のリソースのパーティショニングを可能にするよう高められる。次に、図２を参照するに、本発明の実施形態によるＰＲＬアーキテクチャの例示的ブロック図を示す。パーティショニングをより詳しく説明するために、メインパーティション２１０が使用することのできるコンポーネントは実線のブロックで示す。組み込み、すなわちシステムパーティション２２０が使用することのできるコンポーネントは太い実線のブロックで示す。両方のパーティションが使用することのできるコンポーネントは、一点鎖線のブロックで示す。

【００２３】

この例示的な実施形態では、プラットフォームは、ソケット０－３（２３１－２３４）内に４つのマルチコアプロセッサを有する。この例は４つのプロセッサソケットだけを示すが、当業者には、プロセッサおよびコアの様々な構成を使用して本発明の実施形態を実施しうることは明らかであろう。たとえば、ソケット０（２３１）は、４つの処理コア２３５ａ－ｄを有しうる。基本的に、この例では、図示する実施形態は、プラットフォーム上に１６の実効プロセッサ（たとえば、４つのソケットにそれぞれ４つのコアがある）を有する。この例では、ソケット０－２（２３１－２３３）は、メインパーティション２１０だけが使用することができる。ソケット３（２３４）は、メインパーティション２１０と組み込みパーティション２２０の両方が使用することができる。ソケット３（２３４）において、コア０は、メインパーティション２１０だけが使用することができ、コア１－３は、組み込みパーティション２２０だけが使用することができる。組み込みパーティション２２０は、先に及び後でより詳細に説明するように、ブートキャッシュエージェント２２１を有する。

【００２４】

この実施形態では、プラットフォームは、メモリ２０２に結合されるメモリコントローラハブ（ＭＣＨ）２０１（「ノースブリッジ」とも知られる）を有する。メモリ２０２は、２つのパーティションＭＥＭ１（２０３）およびＭＥＭ２（２０５）を有しうる。メモリパーティションＭＥＭ１（２０３）は、組み込みパーティションだけが使用することができ、メモリパーティションＭＥＭ２（２０５）は、メインパーティションだけが使用することができる。ＭＣＨを含むチップセットは、ソフトウェア構成体を使用するＶＭＭソリューションとは対照的にハードウェア構成体を使用してメモリをパーティションするよう構成される。メモリ２０２は、ハードディスク、フロッピー（登録商標）ディスク、ランダムアクセスメモリ（ＲＡＭ）、読み出し専用メモリ（ＲＯＭ）、フラッシュメモリ、またはプロセッサにより読み出し可能な任意の他のタイプの媒体でありうることは理解されよう。メモリ２０２は、本発明の実施形態の実行を行うための命令を保存しうる。この例では、２つのパーティションだけを示すが、それぞれ独自のパーティションにおいて実行される１つ以上のゲストＯＳがありうることは理解されよう。

【００２５】

ＭＣＨ２０１は、周辺コンポーネント相互接続（ＰＣＩ）バスを介して、「サウスブリッジ」とも知られるＩ／Ｏコントローラハブ（ＩＣＨ）２０７と通信しうる。ＩＣＨ２０７は、ＰＣＩハードドライブ、ＩＤＥ、ＵＳＢ、ＬＡＮ、およびオーディオといったレガシコンポーネント、およびスーパーＩ／Ｏ（ＳＩＯ）コントローラといった１つ以上のコンポーネントに、ローピンカウント（ＬＰＣ）バス（図示せず）を介して結合されうる。この例では、ＩＣＨ２０７は、ハードディスクドライブ２０９およびネットワークインタフェースコントローラ（ＮＩＣ）２１１に結合するよう示す。

【００２６】

ＭＣＨ２０１は、メモリへのアクセスを制御するよう構成され、ＩＣＨ２０７は、Ｉ／Ｏアクセスを制御するよう構成される。組み込みパーティションアーキテクチャでは、チップセットはファームウェアによって、ブート後、プラットフォーム上の様々なリソースをパーティションするよう構成される。一部では、１つのパーティションしかなく、プラットフォームは、多くの点においてレガシプラットフォームのように機能しうる。図示する例では、２つのパーティション、すなわち、メインパーティション２１０および組み込

みパーティション２２０がある。各指定パーティションには、一意のパーティション識別子（ＩＤ）が与えられる。

【００２７】

組み込みパーティションアーキテクチャでは、デバイスがアラートを送信すると、チップセットは、アラートを適切なパーティションに適切にルーティングしうる。これは、この情報がブート時間に符号化されるからである。ＶＭＭが可能にされたシステムでは、ハードウェアが、デバイスアラートをＶＭＭ（仮想デバイス）に送り、ソフトウェアが、情報を、様々な仮想マシンに適切にルーティングする。組み込みパーティションは、ハードウェア支援された仮想化として機能しうる。

【００２８】

一実施形態では、ブートキャッシュエージェントは、すべてのゲスト仮想マシーン（ＶＭ）とプラットフォーム上で実行されるゲストオペレーティングシステム（ＯＳ）を制御するＶＭＭ内に具現化される。別の実施形態では、ブートキャッシュエージェントは、個々のＯＳに対するＩ／Ｏリクエストを制御する特権パーティション、プロセス、またはハイパーバイザ内に具現化される。いずれの場合においても、ブートキャッシュエージェントは、最初のブート時にブートまたは他のターゲット媒体から取り出しされるデータを選択的にミラーリングし、このデータを次のブート時に低速媒体からの読み出しを回避するために戻す。ＶＭＭアーキテクチャの場合、デバイスアクセスは仮想化され、ブートキャッシュエージェントは、デバイスからデータを取り出しするソフトウェアインタメディアリとして機能する。

【００２９】

図３を参照するに、ブートキャッシュエージェント３２１がＶＭＭ内に存在する、例示的な仮想化プラットフォームを示す。この例示的な実施形態では、仮想マシン（ＶＭ）３１０はゲストＯＳ３１１を有する。様々なユーザアプリケーション３１３がゲストＯＳ３１１の下で実行しうる。ＯＳは、ＶＭＭ３２０内に仮想化されうるデバイスドライバ３１５を有する。ブートターゲット（図示せず）を含むプラットフォームハードウェア３３０へのアクセスは、ＶＭＭの使用を必要とする。ブートの場合、ＶＭＭ３２０内のブートキャッシュエージェント３２１が、ブートターゲットへのデバイスアクセスをインターセプトし、セクタが、低速のブートターゲットかまたは高速の揮発性メモリから読み出しされるかどうかを制御する。

【００３０】

同様に、プラットフォームパーティション、すなわち、独自のＯＳ３４１、ユーザアプリケーション３４３、デバイスドライバ３４５を有するより特権的なパーティション３４０を示す。このプラットフォームパーティションもＶＭＭ３２０を介する仮想化デバイスを有しうる。一部では、ブートキャッシュエージェントは、このパーティションのためのボートターゲットデータもキャッシュする。

【００３１】

一実施形態では、ブートキャッシュエージェントは、ブート時にリクエストされたセクタまたは論理ブロックアドレスに基づいて情報をキャッシュする。ブート処理時には、特定のセクタがブート媒体から読み出しされ、これらのセクタは、キャッシュブートエージェントにより、パーティションまたは専用揮発性メモリにミラーリングされる。一実施形態では、ブートキャッシュエージェントは、実際のセクタデータとともに、保存したセクタのインデックスをバッファに保存する。後続のリブート時には、ブートキャッシュエージェントは、ブート媒体から読み出しするリクエストをインターセプトする。揮発性メモリに以前にミラーリングされたセクタがリクエストされた場合、ブートキャッシュエージェントは、このデータを、不揮発性メモリストア（ブート媒体）からデータを読み出しするのに比較して、有意に時間節約をして戻す。

【００３２】

本願に記載した技法は、任意の特定のハードウェアまたはソフトウェア構成に限定されない。この技法は、任意のコンピューティング、コンシューマエレクトロニクス、または

処理環境において可用性を見出しうるであろう。この技法は、ハードウェア、ソフトウェア、またはこれら２つの組み合わせで実施されうる。

【００３３】

シミュレーションのために、プログラムコードが、ハードウェア記述言または設定されたハードウェアが実行すると期待される方法のモデルを本質的に与える別の機能的記述言語を使用してハードウェアを表しうる。プログラムコードは、アセンブリまたはマシン言語、或いはコンパイルおよび／または解釈されうるデータでありうる。

【００３４】

さらに動作を行うまたは結果をもたらすといったように何らかの形式でソフトウェアについて言及するのも当該技術において一般的である。このような表現は、プロセッサに動作を行わせまたは結果を生成させるプログラムコードの処理システムによる実行を説明する省略形に過ぎない。

【００３５】

各プログラムは、処理システムと通信するために高レベルの手続きまたはオブジェクト指向言語で実施されうる。しかし、プログラムは、必要に応じて、アセンブリまたはマシン言語で実施されてもよい。いずれの場合でも言語は、コンパイルまたは解釈されうる。

【００３６】

プログラム命令は、その命令でプログラムされる汎用または特殊用途向け処理システムに本願に記載する動作を行わせるよう使用されうる。或いは、動作は、その動作を行うための配線論理を含む特定のハードウェアコンポーネント、または、プログラムされたコンピュータコンポーネントおよびカスタムハードウェアコンポーネントの任意の組み合わせにより実行されうる。本願に記載する方法は、処理システムまたは他の電子デバイスが本発明の方法を実行するようプログラムするのに使用されうる命令がその上に格納されたマシンアクセス可能媒体を含みうるコンピュータプログラムプロダクトとして提供しうる。

【００３７】

プログラムコード、または命令は、たとえば、ストレージデバイスおよび／または、半導体メモリ、ハードドライブ、フロッピー（登録商標）ディスク、光学ストレージ、テープ、フラッシュメモリ、メモリスティック、デジタルビデオディスク、デジタルバーサタイルディスク（ＤＶＤ）などを含む関連付けられるマシン可読またはマシンアクセス可能媒体といった揮発性および／または不揮発性メモリ、さらに、マシンアクセス可能な生物学的状態保存ストレージといったより非標準の媒体に格納されうる。マシン可読媒体は、マシンにより可読である形式で情報を格納、送信、または受信する任意のメカニズムを含みえ、また媒体は、アンテナ、光ファイバ、通信インタフェースなどといった、その中をプログラムコードが符号化された電気、光、音響、または他の形式の伝播信号または搬送波が通る有形型媒体を含みうる。プログラムコードは、パケット、シリアルデータ、パラレルデータ、伝播信号などの形で伝送されえ、また、圧縮または暗号化形式で使用されうる。

【００３８】

プログラムコードは、モバイルまたはステーショナリコンピュータ、携帯情報端末、セットトップボックス、セルラ式電話機およびページャ、消費者電子デバイス（ＤＶＤプレイヤ、パーソナルビデオレコーダ、パーソナルビデオプレイヤ、衛星受信機、ステレオ受信機、ケーブルＴＶ受信機を含む）、および他の電子デバイスといったプログラム可能なマシン上で実行されるプログラム内に使用されうる。各マシンは、プロセッサ、プロセッサにより可読である揮発性および／または不揮発性メモリ、少なくとも１つの入力装置および／または１つ以上の出力装置を含む。プログラムコードは、入力装置を使用して入力されたデータに適用され、それにより、説明した実施形態を実行し、出力情報を生成しうる。出力情報は、１つ以上の出力装置に供給されうる。当業者は、開示した主題の実施形態は、マルチプロセッサまたはマルチプルコアプロセッサシステム、ミニコンピュータ、メインフレームコンピュータ、および実質的に任意のデバイスに組み込みうる一般または小型コンピュータを含む様々なコンピュータシステム構成とともに実施することができる

ことを理解されよう。開示した主題の実施形態はさらに、分散コンピューティング環境において実施することができ、ここでは、タスクまたはその一部は、通信ネットワークを介してリンクされる遠隔処理デバイスにより実行されうる。

【００３９】

動作は、順次処理として説明されうるが、動作の一部は実際には、プログラムコードが単一またはマルチプロセッサマシンによりアクセスされるようローカルにおよび／またはリモートに格納されて、並列、同時、および／または分散環境にて実行されうる。さらに、一部の実施形態では、動作の順序は、開示する主題の精神から逸脱することなく並べ替えられうる。プログラムコードは、組み込みコントローラによりまたは組み込みコントローラとともに使用されうる。

【００４０】

本発明を例示的な実施形態を参照しながら説明したが、この説明は、限定的な意味合いで解釈することを意図しない。例示的な実施形態の様々な変更、および当業者には明らかである本発明の他の実施形態は、本発明の精神および範囲内であると考える。

【図面の簡単な説明】

【００４１】

【図１】本発明の一実施形態による、例示的なブートキャッシュ方法を示すフロー図である。

【００４２】

【図２】本発明の一実施形態による、複数のパーティションを有するプラットフォーム上の１つの組み込みパーティション内にあるブートキャッシュエージェントを示すブロック図である。

【００４３】

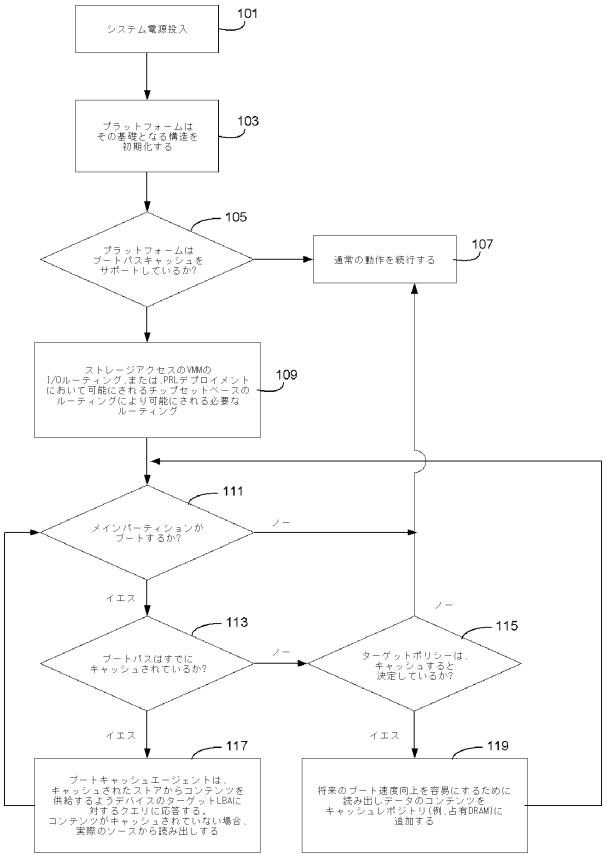
【図３】本発明の一実施形態による、ブートキャッシュエージェントは仮想マシンモニタ（ＶＭＭ）内にある、例示的なプラットフォームを示すブロック図である。

【符号の説明】

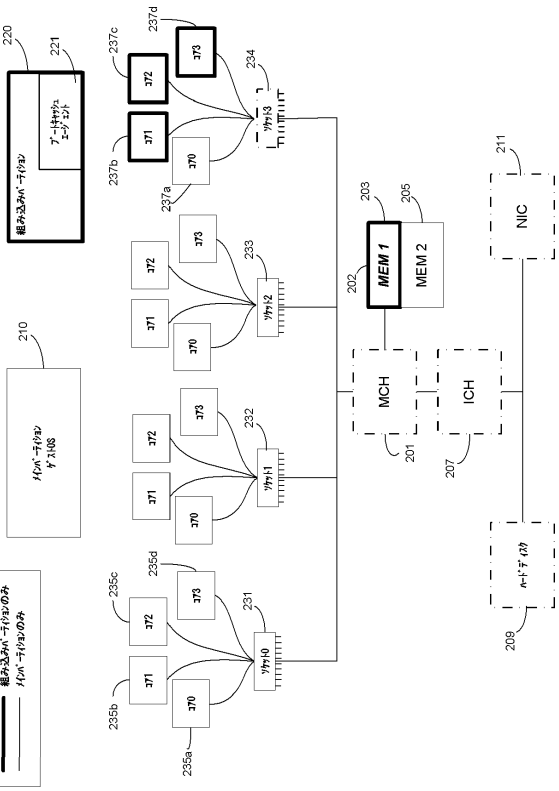
【００４４】

２０１　メモリコントローラハブ
２０２　メモリ
２０３　パーティション
２０５　パーティション
２０７　Ｉ／Ｏコントローラハブ
２０９　ハードディスク
２１０　メインパーティション
２１１　ネットワークインタフェースコントローラ
２２０　組み込みパーティション
２２１　ブートキャッシュエージェント
２３１、２３２、２３３、２３４　ソケット
２３５ａ－ｄ　コア
３１０　仮想マシン
３１１　オペレーティングシステム
３１３　ユーザアプリケーション
３１５　デバイスドライバ
３２０　ＶＭＭ
３２１　ブートキャッシュエージェント
３３０　プラットフォームハードウェア
３４０　プラットフォームパーティション
３４１　オペレーティングシステム
３４３　ユーザアプリケーション
３４５　デバイスドライバ

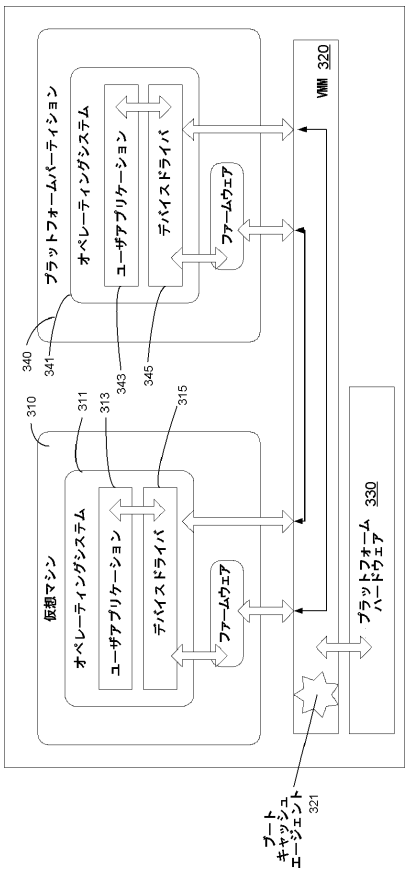
【図１】


システム電源投入
101
プラットフォームは
その基礎となる構造を
初期化する
103
プラットフォームは
ブートパスキャッシュを
サポートしているか?
105
通常の動作を続行する
107
ストレージアクセスのVMMの
I/Oルーティング、または、PRLデプロイメント
において可能にされるチップセットベースの
ルーティングにより可能にされる必要な
ルーティング
109
メインパーティションが
ブートするか?
111
ノー
イエス
ブートパスはすでに
キャッシュされているか?
113
ノー
ターゲットポリシーは、
キャッシュすると
決定しているか?
115
イエス
イエス
ブートキャッシュエージェントは、
キャッシュされたストアからコンテンツを
供給するようデバイスのターゲットLBAに
対するクエリに応答する、
コンテンツがキャッシュされていない場合、
実際のソースから読み出しする
117
将来のブート速度向上を容易にするために
読み出しデータのコンテンツを
キャッシュレポジトリ(例、占有DRAM)に
追加する
119
凡例:
メインパーティションのみ
組み込みパーティションのみ
メインと組み込みの両方の
パーティションにより使用

【図２】


220
221
組み込みパーティション
ブートキャッシュエージェント
210
メインパーティション
237d
237c
237b
237a
373
372
371
370
234
リマップ3
233
リマップ2
232
リマップ1
231
リマップ0
235d
235c
235b
235a
202
MEM 1
203
MEM 2
205
MCH
201
ICH
207
NIC
211
ハードディスク
209

【図３】


プラットフォームパーティション
オペレーティングシステム
ユーザアプリケーション
デバイスドライバ
ファームウェア
VMM 320
340
341
343
345
315
310
311
313
仮想マシン
オペレーティングシステム
ユーザアプリケーション
デバイスドライバ
ファームウェア
プラットフォーム
ハードウェア 330
ブート
キャッシュ
エージェント
321

フロントページの続き

審査官　長谷川　篤男

(56)参考文献　特開２００６－１２６９８７（ＪＰ，Ａ）
特表２００６－５２２９７１（ＪＰ，Ａ）
特開２００１－１０１０３４（ＪＰ，Ａ）
米国特許出願公開第２００３／０１４２５６１（ＵＳ，Ａ１）
新井　利明、関口　知己、佐藤　雅英、木村　信二、大島　訓、吉澤　康文，汎用ＯＳと専用ＯＳを高効率に相互補完するナノカーネルの提案と実現，情報処理学会論文誌，日本，社団法人情報処理学会，２００５年１０月１５日，第４６巻　第１０号，第２４９２－２５０４頁

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　　９／４４５
Ｇ０６Ｆ　　９／４６
ＪＳＴＰｌｕｓ（ＪＤｒｅａｍＩＩ）