

(19) 日本国特許庁(JP)

(12) 特許公報(B2)

(11) 特許番号

特許第4211101号
(P4211101)

(45) 発行日 平成21年1月21日(2009.1.21)

(24) 登録日 平成20年11月7日(2008.11.7)

(51) Int. Cl. F I
G06F 9/445 (2006.01) G06F 9/06 610J
G06F 9/50 (2006.01) G06F 9/06 640H

請求項の数 9 (全 27 頁)

(21) 出願番号	特願平10-322362	(73) 特許権者	000002185 ソニー株式会社
(22) 出願日	平成10年11月12日(1998.11.12)		東京都港区港南1丁目7番1号
(65) 公開番号	特開2000-148504(P2000-148504A)	(74) 代理人	100067736 弁理士 小池 晃
(43) 公開日	平成12年5月30日(2000.5.30)	(74) 代理人	100086335 弁理士 田村 榮一
審査請求日	平成17年8月31日(2005.8.31)	(74) 代理人	100096677 弁理士 伊賀 誠司
		(72) 発明者	鈴木 春良 東京都品川区北品川6丁目7番35号 ソニー株式会社内
		審査官	久保 光宏

最終頁に続く

(54) 【発明の名称】 情報処理装置及び方法並びに記録媒体

(57) 【特許請求の範囲】

【請求項1】

ソフトウェアと当該ソフトウェアを読み込む読み込みプログラムとが記録された着脱可能な記録媒体を用いる情報処理装置において、

上記記録媒体に対する情報の記録/再生を行う記録/再生手段と、

情報を記憶する記憶手段と、

電源投入時に動作開始し、上記記録媒体のブート領域を読み込む基本アクセスプログラムが記憶された不揮発性メモリと、

プログラムを実行する制御手段とを有し、

上記制御手段は、上記記録媒体が挿入された場合上記不揮発性メモリに記憶された基本アクセスプログラムに基づいて、上記記録媒体のブート領域から上記ソフトウェアを読み込む読み込みプログラムと、上記ソフトウェアについて各セグメントに付与されたIDを含む読み込み制御情報とを読み込み、

上記読み込みプログラムに従って、上記読み込み制御情報のIDを参照して上記記録媒体からソフトウェアをセグメント毎に上記記憶手段に読み込んで実行し、

上記記録媒体からソフトウェアを読み込む際に、上記不揮発性メモリにセグメント毎に読み込んで上記記憶手段に移し、上記記録媒体から読み出すセグメントのIDが上記不揮発性メモリに記憶されたソフトウェアのセグメントのIDと一致すると、当該セグメントの読み込みは省略し、

上記ソフトウェアが終了された場合上記基本アクセスプログラムにより当該ソフトウェ

10

20

アで発生した情報を上記記録媒体に書き込む情報処理装置。

【請求項 2】

上記記録媒体は、書き換え可能な記録媒体、又は一部が書き換え可能で他の部分が書き換え不可能な記録媒体である請求項 1 記載の情報処理装置。

【請求項 3】

上記制御手段は、上記ソフトウェアを終了する際に、上記記録媒体の書き換え可能領域に、エラー情報、利用回数、年月日の少なくとも一つを含む情報を記録する請求項 1 記載の情報処理装置。

【請求項 4】

上記制御手段は、上記情報処理装置に記録されたハードウェア ID を読み出して、上記情報処理装置と上記記録媒体との互換性を判断する請求項 1 記載の情報処理装置。

【請求項 5】

上記制御手段は、上記記録媒体から読み出すセグメントの ID と、上記不揮発性メモリに記憶されたソフトウェアのセグメントの ID とを比較し、上記不揮発性メモリに記憶された ID と異なる ID のセグメントを上記記録媒体から読み込む請求項 1 記載の情報処理装置。

【請求項 6】

上記読み込み制御情報は、上記記録媒体の管理情報を含み、

上記制御手段は、上記記録媒体の利用回数を上記管理情報に記録する請求項 1 記載の情報処理装置。

【請求項 7】

ソフトウェアと当該ソフトウェアを読み込む読み込みプログラムとが記録された着脱可能な記録媒体に対する情報の記録/再生を行う記録/再生手段と、情報を記憶する記憶手段と、電源投入時に動作開始し、上記記録媒体のブート領域を読み込む基本アクセスプログラムが記憶された不揮発性メモリと、プログラムを実行する制御手段とを有する情報処理装置における情報処理方法において、

上記制御手段は、上記記録媒体が挿入された場合上記不揮発性メモリに記憶された基本アクセスプログラムに基づいて、上記記録媒体のブート領域から上記ソフトウェアを読み込む読み込みプログラムと、上記ソフトウェアについて各セグメントに付与された ID を含む読み込み制御情報とを読み込み、

上記読み込みプログラムに従って、上記読み込み制御情報の ID を参照して上記記録媒体からソフトウェアをセグメント毎に上記記憶手段に読み込んで実行し、

上記記録媒体からソフトウェアを読み込む際に、上記不揮発性メモリにセグメント毎に読み込んで上記記憶手段に移し、上記記録媒体から読み出すセグメントの ID が上記不揮発性メモリに記憶されたソフトウェアのセグメントの ID と一致すると、当該セグメントの読み込みは省略し、

上記ソフトウェアが終了された場合上記基本アクセスプログラムにより当該ソフトウェアで発生した情報を上記記録媒体に書き込む情報処理方法。

【請求項 8】

着脱可能な記録媒体に対する情報の記録/再生を行う記録/再生手段と、情報を記憶する記憶手段と、電源投入時に動作開始し、上記記録媒体のブート領域を読み込む基本アクセスプログラムが記憶された不揮発性メモリと、プログラムを実行する制御手段とを有する情報処理装置に用いられ、ソフトウェアと当該ソフトウェアを読み込む読み込みプログラムとが記録された記録媒体であって、

上記ソフトウェアを読み込む読み込みプログラムと上記ソフトウェアについて各セグメントに付与された ID を含む読み込み制御情報とがブート領域に記録され、上記情報処理装置に挿入された場合上記不揮発性メモリに記憶された基本アクセスプログラムにより、上記読み込みプログラムと、上記読み込み制御情報とが読み込まれ、

上記読み込みプログラムは、上記読み込み制御情報の ID を参照して上記ソフトウェアをセグメント毎に上記記憶手段に読み込む処理を実行させ、

10

20

30

40

50

上記記録媒体からソフトウェアを読み込む際に、上記不揮発性メモリにセグメント毎に読み込んで上記記憶手段に移し、上記記録媒体から読み出すセグメントのIDが上記不揮発性メモリに記憶されたソフトウェアのセグメントのIDと一致すると、当該セグメントの読み込みは省略させる記録媒体。

【請求項9】

エラー情報、利用回数、年月日の少なくとも一つを含む情報が記録される書き換え可能領域を有する請求項8記載の記録媒体。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は、情報を処理する情報処理装置および方法ならびに情報が記録された記録媒体に関し、詳しくは情報処理装置の本体と記録媒体がソフトウェアおよびデータ上完全に分離されて、記録媒体の交換によっていろいろなアプリケーションが可能となるようなアーキテクチャの情報処理装置および方法並びに記録媒体に関する。

【0002】

【従来の技術】

従来、日本語ワードプロセッサ装置、いわゆるワープロ、電子手帳、小型・携帯型のパーソナルコンピュータ型の情報機器等が提供されている。これらの情報機器では、ハードディスクを内蔵せず、いわゆるROMマシンの形態をとるものが多い。これらROMマシンでは、ソフトウェアは簡単には交換可能ではなく、通常一定のソフトウェアのみしか動作しない専用機となっているものが多い。専用機はその操作、使い勝手において、そのシンプルさゆえに専門技術知識がない人々でも簡単に利用できる利点があるが、逆に、多くの異なったアプリケーションを動作させるのが困難であるという欠点がある。

【0003】

そこで、パーソナルコンピュータ (personal computer; PC)、いわゆるパソコン等の汎用機が発達を始めたが、その初期の段階から、装置内部にはハードディスクのような容量の大きいデータ記録装置を配置し、それを効果的に利用する手段として、ディスクオペレーティングシステム (disk operating system; DOS) のようなファイルシステム (file system) が開発され、それらを核としての汎用オペレーティングシステム (operating system; OS) が発達してきた。現在のパーソナルコンピュータやワークステーション、あるいは、ノートブックPC、PDAと呼ばれるような、いわゆるコンピュータと総称される装置においては、その基本アーキテクチャは、本体の中に内蔵されているハードディスクのようなデータ記録装置の存在を前提に、汎用OSが搭載される形で構成されている。そして、そこには複数のアプリケーションソフトウェアがインストールされて、汎用装置としてのパソコン、コンピュータが成り立っている。

【0004】

続いて、情報処理装置におけるソフトウェアの階層について、図12を参照して説明する。図12は、現在の汎用パソコンのシステム階層を示している。すなわち、システム階層は、上位から下位の順に、アプリケーションプログラム (application program; APP)、アプリケーションプログラムインターフェース (application program interface; API)、オペレーティングシステム (operating system; OS)、デバイスドライバ (device driver; DD)、ハードウェア (hardware; H/W) から構成される。

【0005】

続いて、このようなソフトウェアのシステムを構成するシステム階層の歴史的な発展を進化論的に説明する。

【0006】

パソコン草創期には、デバイスドライバがハードウェアに内蔵されるROMにBIOS (basic I/O interface) という形で書かれており、全てのソフトウェアはこのBIOSを通して、ハードウェアを制御する形になっていた。BIOSは簡単に取りはずしがきくようにはなっておらず、そのため、BIOSがそのマシンの機能、性能の限界を定めていた

10

20

30

40

50

。そのため、BIOSに合致する範囲のOSやアプリケーションしか動作せず、その意味では、全てのソフトウェアはハードウェアに依存していたと言える。

【0007】

次に、ハードディスク等の内蔵データ記録装置が普及するに連れて、デバイスドライバがハードディスク内に置かれるようになると、OSとデバイスドライバは何時でも書き換え可能、交換可能となって、一つのハードウェアでも複数の異なるOSが走るようになった。逆に言えば、OSが同じであれば、下位のハードウェアにはいろんなバリエーションが許されるようになり、ハードウェアに依存しないハードウェアフリーのコンセプトが一般的となった。しかし、この段階でも、アプリケーションがOSに依存する状況は以前と同じであった。

10

【0008】

更に、昨今のいわゆるJAV A等に代表されるように、OSよりも更に上位のレベルで、アプリケーションのインターフェースを共通に定める技術が確立しつつあり、このことによって、アプリケーションがOSからも独立する形が普及し始めている。これは前者よりも一歩進んだ、OSフリーのコンセプトである。この場合は、アプリケーションインターフェースさえ遵守していれば、下位のOSの如何を問わず、同じアプリケーションが動作することを保証している。これによってアプリケーションソフトウェアの汎プラットフォーム化が一段と進展しているのは事実である。

【0009】

【発明が解決しようとする課題】

このように、パソコン等の進化は、一言で言えば、ソフトウェアやアプリケーションの独立化にあると言える。しかし現在、アプリケーションは高度なレベルでその独立性を達成しつつあるが、それはあくまでコンピュータ技術的な進化、作る側での進化であって、一歩、ユーザの視点から、また、人間工学的な側面から見れば、別の不満足な状況が見えてくる。それは、“使いやすさ”、“簡単さ”と言うような視点であるが、この点では、現状のコンピュータは相変わらず“わかりにくく”、“十分複雑”である。

20

【0010】

第1に、アプリケーションはまずもって、ハードウェアにインストールする必要がある、必要ならば各種のシステムパラメーターをセットアップ(set up)して、他のアプリケーションやOSとの整合性を取っていかないとうまく動作させることができない。より詳しく言えば、アプリケーションソフトウェアは通常例えばいわゆるCD-ROM等の媒体で販売されるか、インターネット等からダウンロードされるのが一般的になりつつあり、ユーザはこれらのソフトウェアをまず自分のマシンのハードディスク上にインストールしなければならない。その後、自分のマシンの環境、状態に合わせて、各種のシステムパラメーター等を設定するためにセットアップという作業を行って初めて装置を使用できる状態になる。通常は、ハードディスク上にはそれぞれ異なったベンダーが開発した各種のアプリケーションや共有ファイルが存在しており、これらの相互調整、整合性などを、ユーザ自身が正確に行い得ないと、目的のアプリケーションがうまく動作しなかったり、また、動作しても、不完全であったり、スピードが遅かったり、他のアプリケーションの動作に悪影響を与えてしまうこともしばしばである。また、これらの作業の途中においては誤って必要なファイルを壊してしまったり、思いもかけないトラブルが発生するのが常である。メーカーサイドではこれらのトラブルが生じた場合の緊急的措置として、サポートセンター等を設置して解決に努力をしているものの、何が問題になっているのか、トラブルの現象はどうなっているのか、ハードかソフトウェアかどちらの問題なのか等をユーザ自身が説明できなければならず、結局、ユーザ側に、コンピュータそのものの知識が必要となり、これが、子供、主婦、老人など素人の利用を著しく制限している大きな要因となっている。

30

40

【0011】

第2には、デバイスドライバ、OSやアプリケーションソフトウェアが高機能化する度にバージョンアップを行ったり、また、周辺装置を加えた場合のデバイスドライバのコンフ

50

イグレーションやマシンの再起動等もユーザ自身の責任の下で行うことが必要とされている。この場合も、現在どのようなソフトウェアがディスク上に存在し、自分のこれまで作業してきた貴重なデータを破壊しないで、スムーズにシステムを書き換えたり、アドオン（add-on）していくためには、どのような技術的注意を払わなければならないのか、全ては、ユーザ自身の責任になっている。これらの技術的知識は暗黙のルールに基づくものや、ノウハウ的な事柄を含んでいる場合も多く、そういう意味では、ユーザは常に自分のマシンの管理を行うべき運命にあるが、このような状況下においても多くのトラブルが発生しやすく、結局、知識のないユーザはお手上げとなり、素人ユーザへのコンピュータの普及を阻害する原因になっている。

【 0 0 1 2 】

第3に、アプリケーションがうまく動作してからの操作方法に関して言えば、各メーカー共、簡単操作を目的とした GUI (graphic user interface) や、簡単メニューなどの工夫を凝らして、表面的には“やさしさ”、“簡単さ”をアピールしてはいるが、一旦、操作中にエラーが生じた場合や、マシンやOSの動作に伴うエラーなどによって、本来の動作環境が壊れてしまった場合などでは、マシンの回復手段には相当高度の知識が必要とされるので、技術知識のないユーザは本当に困ってしまう。マシンの回復もさりながら、今まで作成した自分の固有のデータなども失われてしまう恐れにも直面することになる。このようなことも、素人がパソコン等を敬遠してしまっている要因ではないかと考えられる。

【 0 0 1 3 】

以上のように考えてくると、素人への普及を阻んでいる大きな要因は、アプリケーションの準備段階や、エラーやメンテナンスの段階で生じると考えられるが、それをコンピュータアーキテクチャ的に検討すると、問題の所在は、ハードディスクのような内蔵のデータ記録装置上に、ありとあらゆるソフトウェアを混在させんがために、より複雑なOSを搭載し、共通ライブラリのようなソフトウェアを肥大化させつつ汎用性を確保しようとする現在のPCのアーキテクチャの基本的な考え方にあるとの見解に到達する。

【 0 0 1 4 】

今後、社会生活においても、ますます、情報化、デジタル化が重要となるが、企業内はともかく、潜在的に幅広いユーザ層がいると考えられる家庭においては、主たるユーザ層は技術者ではないため、そこで利用されるべきコンピュータは、シンプル且つ汎用性があり、技術知識不用の新しいアーキテクチャのパソコンでなければならない。かといって、初めに述べたように、ROMマシンのような専用機では、マシンそのものはシンプル化されるが、各種のアプリケーションに対してはそれぞれの専用ハードウェアが必要となり、いちいちハードウェアを分けて利用するのは不便であり、家中に物が溢れかえることになってしまう。しかし、汎用性を維持しつつ、現在のパソコンアーキテクチャの延長線上を推し進めていっても、真の問題解決になり得ないのは、今まで見てきた通りである。

【 0 0 1 5 】

更にもうひとつの大きな問題として、ユーザデータの取り扱いがある。昨今では、アプリケーションプログラムとともに、いわゆるコンテンツデータがますます重要となってきている。いわゆるCD-ROMやいわゆるDVD-ROMなどのROMメディアに置かれるデジタルコンテンツは、それを見るだけであれば、そのメディアを携帯することで済むことが多いが、アプリケーションを実行させていく過程の中で生成されたコンテンツ関連データや、自分が作成した各種のユーザデータ、システムデータ、マシン環境データなどは、ROMメディアであるため書き込みができず、ハードディスク内に置かれるか、あるいはフロッピーディスクやICカードのような可搬型RAMメディアに分離格納されざるを得ない。ハードディスクに置かれていく場合は、作業の継続性を維持しようとするれば、そのマシンでしか、作業を進めることができなくなってしまい、作業場所に制約が生じ、家、学校、会社というように場所が違ふところで作業を継続するには不便となっている。このことは、アプリケーションがハードウェアから完全には独立していないことを意味し、依然として、マシンに従属している状況を示している。そのため、企業、学校、家

10

20

30

40

50

などの異なる場所で作業を継続しようとする、多くの場合は、そのマシンを携帯せざるを得ない状況に陥ることになる。また、可搬型RAMメディアに分離格納される場合は、オリジナルコンテンツの格納されたいわゆるCD-ROM等とユーザデータの格納されたRAMメディアを常に一緒に管理する必要が生じ、煩雑で使い勝手も悪い。そして、これら複数のメディアを同一アプリケーション用のデータとして、はっきりした意識下のもとに管理、操作していくには、ユーザ自身にもある程度の技術知識を持っていることが要求されてくる。

【0016】

本発明は上述の実情に鑑みて提案されるものであって、コンピュータの知識のないユーザ層においても簡単に使用できるような情報処理装置及び方法並びに情報を記録した記録媒体を提供することを目的とする。

【0017】

【課題を解決するための手段】

上述の課題を解決するために、本発明は、ソフトウェアと当該ソフトウェアを読み込む読み込みプログラムとが記録された着脱可能な記録媒体を用いる情報処理装置において、上記記録媒体に対する情報の記録/再生を行う記録/再生手段と、情報を記憶する記憶手段と、電源投入時に動作開始し、上記記録媒体のブート領域を読み込む基本アクセスプログラムが記憶された不揮発性メモリと、プログラムを実行する制御手段とを有し、上記制御手段は、上記記録媒体が挿入された場合上記不揮発性メモリに記憶された基本アクセスプログラムに基づいて、上記記録媒体のブート領域から上記ソフトウェアを読み込む読み込みプログラムと、上記ソフトウェアについて各セグメントに付与されたIDを含む読み込み制御情報とを読み込み、上記読み込みプログラムに従って、上記読み込み制御情報のIDを参照して上記記録媒体からソフトウェアをセグメント毎に上記記憶手段に読み込んで実行し、上記記録媒体からソフトウェアを読み込む際に、上記不揮発性メモリにセグメント毎に読み込んで上記記憶手段に移し、上記記録媒体から読み出すセグメントのIDが上記不揮発性メモリに記憶されたソフトウェアのセグメントのIDと一致すると、当該セグメントの読み込みは省略し、上記ソフトウェアが終了された場合上記基本アクセスプログラムにより当該ソフトウェアで発生した情報を上記記録媒体に書き込むことを特徴としている。

【0018】

また、本発明は、ソフトウェアと当該ソフトウェアを読み込む読み込みプログラムとが記録された着脱可能な記録媒体に対する情報の記録/再生を行う記録/再生手段と、情報を記憶する記憶手段と、電源投入時に動作開始し、上記記録媒体のブート領域を読み込む基本アクセスプログラムが記憶された不揮発性メモリと、プログラムを実行する制御手段とを有する情報処理装置における情報処理方法において、上記制御手段は、上記記録媒体が挿入された場合上記不揮発性メモリに記憶された基本アクセスプログラムに基づいて、上記記録媒体のブート領域から上記ソフトウェアを読み込む読み込みプログラムと、上記ソフトウェアについて各セグメントに付与されたIDを含む読み込み制御情報とを読み込み、上記読み込みプログラムに従って、上記読み込み制御情報のIDを参照して上記記録媒体からソフトウェアをセグメント毎に上記記憶手段に読み込んで実行し、上記記録媒体からソフトウェアを読み込む際に、上記不揮発性メモリにセグメント毎に読み込んで上記記憶手段に移し、上記記録媒体から読み出すセグメントのIDが上記不揮発性メモリに記憶されたソフトウェアのセグメントのIDと一致すると、当該セグメントの読み込みは省略し、上記ソフトウェアが終了された場合上記基本アクセスプログラムにより当該ソフトウェアで発生した情報を上記記録媒体に書き込むことを特徴としている。

【0019】

また、本発明は、着脱可能な記録媒体に対する情報の記録/再生を行う記録/再生手段と、情報を記憶する記憶手段と、電源投入時に動作開始し、上記記録媒体のブート領域を読み込む基本アクセスプログラムが記憶された不揮発性メモリと、プログラムを実行する制御手段とを有する情報処理装置に用いられ、ソフトウェアと当該ソフトウェアを読み込

10

20

30

40

50

む読み込みプログラムとが記録された記録媒体であって、上記ソフトウェアを読み込む読み込みプログラムと上記ソフトウェアについて各セグメントに付与されたIDを含む読み込み制御情報とがブート領域に記録され、上記情報処理装置に挿入された場合上記不揮発性メモリに記憶された基本アクセスプログラムにより、上記読み込みプログラムと、上記読み込み制御情報とが読み込まれ、上記読み込みプログラムは、上記読み込み制御情報のIDを参照して上記ソフトウェアをセグメント毎に上記記憶手段に読み込む処理を実行させ、上記記録媒体からソフトウェアを読み込む際に、上記不揮発性メモリにセグメント毎に読み込んで上記記憶手段に移し、上記記録媒体から読み出すセグメントのIDが上記不揮発性メモリに記憶されたソフトウェアのセグメントのIDと一致すると、当該セグメントの読み込みは省略させることを特徴としている。

10

【0020】

【発明の実施の形態】

以下、本発明に係る情報処理装置及び方法並びに記録媒体の好適な実施の形態について図面を参照して説明する。

【0021】

まず、本実施の形態のアーキテクチャの概念（コンセプト）を説明する。

【0022】

従来の技術の項で説明したように、真に使いやすい、家電化、日用品（コモディティ）化された新しい形のパソコンを目指すため、シンプルで且つ複数の異なるアプリケーションが利用できるような基本ハードウェアアーキテクチャシステムを考える。本発明の基本的な概念は、ハードウェア本体には、簡単なメディアロード・アンロード機構である基本アクセスプログラム（以下、本明細書中ではBMAP（Basic Media Access Program）という）というプログラムしか存在せず、全てのソフトウェアおよびコンテンツは着脱可能なリムーバブルメディア（removable media）上に自己完結な形態で搭載されており、そのメディアをドライブに挿入すると、それらのソフトウェアは自動的にロードされ、アプリケーションが使える状態になる。アプリケーションを終了して、メディアをドライブから排出（イジェクト）すると、全ての操作結果やデータ等はそのメディアに記録されており、その意味では、メディアそのものがコンピュータ成立のための全ての情報を担っていると考えることができるものである。このコンセプトに基づく本アーキテクチャを単一メディア活用プラットフォーム（以下、本明細書中ではSMAP（Single Media Activated Platform）という）アーキテクチャと呼び、このアーキテクチャに基づいて構成されるシステムをSMAPプラットフォームと呼ぶ。

20

30

【0023】

本アーキテクチャに基づいて構成された情報処理装置の具体例について、図1を参照して説明する。

【0024】

情報処理装置1は、文字、画像等を表示する表示画面3を主面に有し、側部にディスクを格納して記録/再生を行うドライブ10を備えている。ドライブ10にはリムーバブルメディア2を挿入することができ、リムーバブルメディア2としては、電子本2₁、教育ソフトウェア2₂、電子メール2₃、ワープロ2₄等の各種のソフトウェアを格納したものが提供されている。情報処理装置1には、入力装置としてキーボード4およびマウス5が接続され、出力装置としてヘッドフォン6が接続されている。また、プリンタ7に出力したり、スキャナ8によって読みとったイメージを入力することもできる。更に、表示画面3にはタッチパネルを装着することもでき、ペン型タブレットやその他の入出力装置を接続することもできる。更に、情報処理装置1は、インターネット/ネットワーク9を経由してデータの入出力をすることもできる。

40

【0025】

上記SMAPアーキテクチャに基づく情報処理装置1、すなわち上記SMAPプラットフォームは、挿入するリムーバブルディスク2によっていろいろな装置として使用することができる。すなわち、情報処理装置は、リムーバブルディスク上のアプリケーションソフ

50

トウェアとして、例えば、ワープロを搭載しておけばワープロ専用機となり、電子メールソフトウェアの搭載されたメディアを挿入すれば、電子メール端末になる。教育ソフトウェアを搭載したメディアならば、情報処理端末は教育機器となる。また、これらのアプリケーションソフトウェアのためのOSも統一する必要はなく、それぞれのアプリケーションに適したOSを採用することもできる。本アーキテクチャは1ディスクに1アプリケーション基本概念から出発しているため、大きなアプリケーションには不向きだが、逆に、シンプルオペレーション、すなわち簡単な操作という視点を徹底追及するができたため、家庭向き、素人向きの日用品(コモディティ)化したコンピュータを実現する上で、基本的な役割を果たすことができるものである。

【0026】

また、本発明の実施の形態の具体例としては、図2に示すような小型の情報処理装置1を挙げる事ができる。この情報処理装置1は、小型軽量のパームトップサイズのコンピュータ、いわゆるパームトップ型のコンピュータである。この小型の情報処理装置2も、図1に示した情報処理装置1と同じく、挿入するリムーバブルディスク2によっていろいろな装置として使用することができる。

【0027】

続いて、情報処理装置のハードウェア構成について、図3を参照して説明する。情報処理装置1は、リムーバブルメディア2を格納するリムーバブルメディアドライブ10と、情報処理装置1における処理を集中して実行するCPU11と、主メモリ12と、不揮発性のメモリであるフラッシュメモリ13と、接続される外部装置とのインターフェースを行うインターフェース(I/O)14とを有している。

【0028】

本アーキテクチャでは情報処理装置1の内部には、ハードディスク等の固定型内蔵データ記録装置を一切使用せず、リムーバブルディスク2のみを唯一のデータ記録装置とし、このリムーバブルメディア2が格納されるリムーバブルメディアドライブ10は所定のインターフェースを通して、CPU11と接続されている。本アーキテクチャによれば、リムーバブルメディア2としては、フロッピーディスクのようなものから、100MBから200MB程度の大容量フロッピーディスク、いわゆるMOやいわゆるMDデータ、いわゆるCD-ROM、いわゆるDVD-ROM、ICメモリカード、いわゆるPCカード、いわゆるメモリスティックなどであってもよい。尚、MDデータとは、音楽用で広く利用されているミニディスクのデータバージョンである。メディアの種別で言えば、ROM型(read only)、RAM型(re-writable)、ROM/RAM一体型のハイブリッド(hybrid)型があるが、本アーキテクチャはいずれにも適用可能であるが、ハイブリッド型の場合に、特にその特徴を発揮する。メディアの物理的形態で言えば、ディスク型であっても、非ディスク型、例えばICメモリカードのような形態であってもよい。RAM型メディアの場合は、生産者側がそこに搭載すべきソフトウェアやコンテンツを業務用等のコピーマシンで一枚一枚生産しなければならず、少量生産では有効だが、大量生産には不向きである。ただ、ディスクであって、一部がROM化されているならば、ROM部分はCD-ROM等と同じに生産は簡単であり、有利である。ここでは、ROM/RAM一体型のメディアをハイブリッドメディアと呼ぶことにするが、以後の説明では、小型で取り扱いが便利、ハイブリッド化も容易で、且つ安価にできるディスク型のハイブリッドメディアとしていわゆるハイブリッドMDデータを想定して、実施例を述べていく。尚、本文内でROM部、ROMID等は、RAM型のMDデータの場合は、ソフトウェアやコンテンツが予め記録されている部分と同義である。

【0029】

情報処理装置1は、本体内蔵のメモリとしては、DRAM等の通常の半導体メモリを主記憶領域である主メモリ12とし、別途、不揮発性の、例えば、フラッシュ(flash)メモリ13を内蔵している。不揮発性メモリとは、電源を切っても情報が保持されるようなメモリを指す。本アーキテクチャでは、電源投入時に自動的に動作開始する上記BMAPという初期プログラムが必要であるが、それは、このフラッシュメモリ13の一部に格納さ

10

20

30

40

50

れていても、別途のEEPROMなどのROMメモリにあってもよい。いずれの場合でも、上記BMAPやフラッシュメモリ13はCPUのバスに直接接続されており、プログラム実行可能なメモリ空間に割当てられている。上記BMAPはリムーバブルディスク上に搭載されているシステムソフトウェアやアプリケーションソフトウェアに全く依存しない独立固定プログラムから成っている。

【0030】

インターフェース14としては通常のコンピュータ周辺装置、例えばディスプレイモニタ、キーボード、マウス、ネットワーク、プリンタなどが接続されているものとするが、本アーキテクチャの構成要件として必須のものではない。

【0031】

続いて、本実施の形態で利用するリムーバブルメディアの例としての上記ハイブリッドMDデータの構造と、UWエリアの定義について説明する。

【0032】

図3におけるリムーバブルディスク2は、ROM部2aとRAM部2bを有するハイブリッドメディアである。上記ハイブリッドMDデータの場合には、その基本構造は図4のようになる。現在普及している音楽用のMDにはMDデータというデータ版の規格があり、これはソニー株式会社により提唱されている“Rainbow Book Part 2:MD DATA System”に規定されている。図4はこの規格に則った上で、本アーキテクチャが加えた仕組みを説明するためのものである。

【0033】

図4の中で、情報領域(information area)とあるのは、デジタルデータが記録される領域である。内周側には予めデータが記録されており、読み出しのみが可能なROM領域となっており、その外側は、記録、読み出し可能なRAM領域である。リードイン領域(lead-in area)にはROM部の目次部(table of contents; TOC)が固定の位置に置かれているが、例えば、このディスクが音楽用か、データ用かを判断できる情報や、各領域の開始アドレス、長さなどの管理情報が置かれている。リードイン領域(lead-in area)の直後から1クラスタ(64KB)分はブート領域(boot area)として定義された領域で、ここには実際のシステムロードプログラム等を置くことができる。残りのROM部はMDデータ規格で定められた論理フォーマットに基づいて動作するファイルシステム(file system)等で制御されなければならない。このファイルシステムのディレクトリー等はVMA(volume management area)に書かれている。次にRAM部であるが、RAM部の開始からはUTOC(50クラスタ)というRAM部をカバーする目次部が置かれ、記録可能ユーザ領域(recordable user area)が実際の使用領域となる。この領域もROM部と同様に、ファイルシステム等で制御されなければならないし、そのためのディレクトリー等はRAM部内のVMAに置かれる。ここで、上記ハイブリッドMDデータの場合は、ROM部のTOCやVMA領域はRAM部のUTOC内やVMA領域にコピーされ、RAM部を参照するだけでROM部を含めたメディア管理情報がわかるようになっている。最後のリードアウト領域(lead-out area)は未使用の領域で、通常のファイルシステムからの関数ではアクセス不可と定義されるが、本アーキテクチャにおいては、記録可能領域の直後から始まるこのリードアウト領域の最初の1クラスタ(64KB)分のシステム使用領域を定義し、これをUW領域と呼ぶ。ドライブのフォーマットコマンド等では、UTOCを含めた記録可能領域は再フォーマットされてしまうが、このUW領域はフォーマットの対象からははずれているため、情報処理装置本体側との連携によって、半永久的なデータ保存が可能となる。尚、ROM型のMDデータディスクの場合は、記録可能領域が存在せず、全て前マスタ領域(pre-mastered area)となり、リードイン領域(lead-out area)への書き込みはできない。RAM型のMDデータディスクの場合は、前マスタ領域が存在せず、全て記録可能領域になるが、この場合は、TOC情報からUTOCの位置がわかり、UTOC後の1クラスタ分がブート領域として確保される。尚、通常はリードインの直後からUTOCが始まるが、交代トラック(alternate track)をリードインとUTOCの間に置くこともできる。VMAはその後に置かれる。従って、本体側

10

20

30

40

50

では、まず、TOCを読み出し、その情報によって、ROM/RAM/ハイブリッドの区別を得る。RAM型の場合は、更にUTO Cの情報から、ブート領域を見つけることができるし、ROM/ハイブリッドの場合は、リードイン領域の直後にブート領域のあることがわかる。従って、本アーキテクチャで用いるMDデータディスクはROM/RAM/ハイブリッドのどれでも同一に扱うことができるが、ROM型の場合は、UW領域が利用できないという制限があるがそれ以外は同じことである。尚、前マスタ (pre-mastered) と記録可能 (recordable) を合せた記憶容量は凡そ140MBとなっている。

【0034】

本アーキテクチャは、このブート領域とUW領域を本体側の上記BMAPやフラッシュメモリとの関連において、うまく利用する仕組みを考案したものである。

10

【0035】

続いて、本実施の形態に基づく、上記ハイブリッドMDデータ上へのソフトウェア格納方法について説明する。図5は、本アーキテクチャにおける、ソフトウェアやデータの格納方法を示したものである。リムーバブルメディア2のデータ記録領域は、ソフトウェアやコンテンツを格納するROM部2aと、ユーザデータを格納するRAM部2bとに大別される。

【0036】

ROM部2aは内周側から、通常5つのセグメントに分割されており、各セグメントには図に示されたようなソフトウェアの種類が前マスタ (pre-mastered) される。すなわち、最内周から外周側へROM部#1(2a₁)ROM部#2(2a₂)ROM部#3(2a₃)ROM部#4(2a₄)ROM部#5(2a₅)がそれぞれ同心円状の領域に配置されている。各セグメントの長さや開始アドレスの情報は前マスタするとき図6に示すブート領域 (boot area) 内のSPデータ部に記述される。尚、カウントの単位はクラスター (64KB) である。実際にこれらのソフトウェアを本体にロードするプログラムはブート領域内のシステムローダ (system loader; SYSLD) 部としてディスクから提供される。ROM部2aの内、#1から#4はソフトウェアプログラムを、その役割によって、分類したものである。すなわち、ROM部#1は共通データ、共通プログラム、フォント類、辞書などであり、ROM部#2はOS、デバイスドライバであり、ROM部#3はAPI、ミドルウェアプログラムであり、ROM部#4はアプリケーションプログラムであり、ROM部#5はデジタルコンテンツである。実際の使用にあたっては、ROM部#1に全てのプログラムが入ってしまうものや、セグメント数を更に増加させることも可能である。デジタルコンテンツデータはユーザ側で変更する必要のない、または、ユーザに勝手に変更されては困るデータの場合が多く、通常は、電子化された本のデータであったり、大量の静止画、音声、ビデオデータなどがこれに相当する。図5ではデジタルコンテンツデータはROM部#5に置かれているが、通常、これはシステムローダで初期にロードされない場合が多く、アプリケーションが立ち上がった後にアプリケーションによってアクセスされる。実際の使用にあたっては、必要に応じてコンテンツ部分も複数のセグメントに分割しておくこともでき、また、その位置もROM部2a内のどこにあってもよい。アプリケーションソフトウェアが電子メールのような場合は、コンテンツは搭載されていないこともあるが、本アーキテクチャでは、コンテンツ部分が存在しても、しなくても矛盾なく動作する。SPデータ内に予め記述されるセグメント情報は、一般的には、セグメント数の情報と、システムローダがロードすべきセグメント、ロードしなくてよいセグメントの情報がビットマップ形式で、前マスタ時に記録されている。本実施の形態においてはビットマップとして32バイトを割当てたので、セグメント数の最大は256となる。RAM部2bは書いたり消したりできる領域であり、OSやアプリケーションに解放されていて、いわゆるユーザデータやシステムデータが格納されていく領域である。

20

30

40

【0037】

続いて、上述の情報処理装置1およびリムーバブルディスク2における上記BMAP13a、フラッシュメモリ13内のデータ、システムローダ、SPデータ、UWデータの構成例を示す。

50

【 0 0 3 8 】

図 7 に示すように、上記 B M A P のプログラムは、A 点から B 点のハード初期化プログラム、B 点から C 点のブート領域と U W 領域の読み込みプログラム、C 点以降の U W 領域への U W データの書き込みプログラムから構成されている。そして、上記 B M A P の先頭から A 点までの領域には、A 点へジャンプ、B 点開始番地、C 点開始番地、R B T 開始番地、R U W 開始番地、R S 開始番地、保留、ハード I D などの項目データが置かれている。これら A 点、B 点、C 点については後述する。

【 0 0 3 9 】

図 8 に示すように、フラッシュメモリ 1 3 内のデータは、任意サイズの上記 B M A P 領域、6 4 K B の R B T 領域、6 4 K B の R U W 領域、任意サイズの R S 領域から構成されている。

10

【 0 0 4 0 】

図 6 に示すように、ブート領域は、6 0 K B のシステムローダ部と 4 K B の S P データ部からの 6 4 K B である。システムローダ部には、この上記ハイブリッド M D データに搭載されているソフトウェアをセグメント毎に読み込むためのシステムローダ用のプログラムが格納されている。S P データ部は、R O M 部セグメント数 (2 バイト)、ビットマップ (3 2 バイト)、R O M 部 # 1 開始アドレスと長さ (4 バイト)、・・・、R O M 部 # 5 開始アドレスと長さ (4 バイト)、R O M 5 I D (4 バイト)、保留、U S E L I M I T (4 バイト) の順で構成されている。

【 0 0 4 1 】

図 9 に示すように、U W 領域は 6 4 K B であって、U S E D F L A G (2 バイト)、I D S T A M P (3 2 バイト)、U S E C O U N T (4 バイト)、U S E T I M E (2 0 バイト)、E R R S T A T U S (1 K バイト)、保留の順に構成されている。これら各要素については後述する。

20

【 0 0 4 2 】

本アーキテクチャの最も基本的な特徴は、上記ハイブリッド M D データをドライブに挿入すると、そのディスクに格納されているソフトウェアが自動的に情報処理装置の本体内にロードされ、アプリケーションが立ち上がっていくものである。従って、ハードウェア側から見れば、O S やアプリケーションを特定する必要がなく、O S として携帯端末用の各種 O S が入っている上記ハイブリッド M D データを挿入すると、その O S に応じたマシンとなる。本実施の形態に基づけば、携帯端末用の O S に限られず、各種の組み込み型 O S を採用する場合にも本アーキテクチャは適用できるので、ハードウェア側で利用する O S を一つに定める必要がない。ある意味で、ハードウェアの O S からの独立であると言える。

30

【 0 0 4 3 】

従来のパソコン等では、ハードウェアは何らかの特定 O S を前提として製品化されてきたが、本アーキテクチャにおいてはこの制約をなくすことができ、ハードウェアの汎用性が更に高められ、逆に言えば、上記ハイブリッド M D データがコンピュータそのものとなり、ハードウェアは単なるがらんどろの入れ物装置ということになる。この考えを推し進めれば、例えば、企業や学校の机にハードウェアを埋め込み、個人、個人は自分の上記ハイブリッド M D データ持ち歩くだけで、何時でもどこでも自分のコンピューティング環境を 1 0 0 % 再現できる。すなわち、自分の専用ハードウェアという考えを持つ必要がなくなる。

40

【 0 0 4 4 】

続いて、本アーキテクチャに基づく情報処理装置がどのように動作するかについて、図 1 0 に示すフローチャートを参照して説明する。

【 0 0 4 5 】

情報処理装置のハードウェアには、上記 B M A P と呼ばれる、不揮発性 R O M 常駐プログラムが存在し、ステップ S 1 1 における電源投入 (パワーオン) 時には、いつでも上記 B M A P がハード初期化プログラムの開始点 (A 点) から起動される。上記 B M A P は図 7

50

に示したように3つの部分に別れており、それぞれのプログラム開始点をA点、B点、C点とする。ハードウェア構成によっては、これらのプログラムサイズは一定しない可能性があるが、外部からA点、B点、C点を参照できるために上記BMAPの頭の256バイトにこれらの開始アドレス情報を参照できる固定領域を設けておく。通常のハードウェア構成では、メモリ空間の0番地からCPUが走り出すので、ここにA点へのジャンプ命令を置く。ステップS12における、A点からの動作はハードウェアの初期化動作である。初期化が完了するとB点に制御が移る。これに続いて、ステップS13でドライブにメディアが挿入されたかを判断しする。メディアが挿入されていると次のステップS14に進み、メディアが挿入されていないと挿入されるまで待ち状態となる。S14においては、電源遮断（パワーオフ）の禁止およびメディア排出の禁止を行う。ステップS15においては、B点に入ると、その中に予め組み込まれている上記ハイブリッドMDデータ上のデータを読み込むための最低限のドライブ制御プログラムが動作して、上記ハイブリッドMDデータのブート領域から1クラスターである64KB分のデータすなわちSYSLD部とSPデータ部を主メモリにロードし、その後、やはり、固定長の領域で、本実施の形態で新たに定義したUW領域から1クラスターすなわち64KBのデータ（UWデータ）をロードする。システムローダは、このディスク上に搭載されているOSやシステムを実際にロードできるようなシステムローディングプログラムであって、ブート領域の先頭番地から書かれている。SPデータ部（4KB）は、このディスク上のソフトウェアやコンテンツがどのような種類、構成からできていて、どこに格納されているかの情報を提供するものである。UWデータ部には、本体におけるディスクの使用回数や、エラー情報などの発生データが記録される。上述のように、ステップS14において、上記BMAPの動作中は、電源遮断（パワーオフ）やディスクの排出（イジェクト）がユーザの操作からできないようにソフトウェアロックがかけられ、誤動作を防止する。具体的には所定のロックコマンドを発行することで実現する。また、ディスプレイの接続されている場合は、上記BMAP内のハードウェア初期化時に、ハードウェア固有の初期画面を映し出すこともできる。上記BMAPはディスク上に搭載されているOSやアプリケーションの如何にかかわらず、一定の動作であるので、予めハードウェア本体内に常駐しておいても、先ほど述べたハードウェアやソフトウェアの構成上のフレキシビリティに何ら制約を与えるものではない。ブート領域およびUW領域をロードした後は、上記BMAPは一旦制御をSYSLDの先頭番地に移す。

【0046】

システムローダの動作は、必要となるOSやアプリケーションソフトウェアをディスクから本体にロードすることであるが、これらの制御は読み込まれたSPデータ部の情報に基づく。SPデータ部は、セグメントの管理情報とそれ以外のディスク管理情報とから構成される。前者は、以後のシステムローダの動作で詳しく説明するが、後者の一例として、本実施例ではUSELIMITを定義している。USELIMITには、ディスク販売元が前マスタ時に最大利用回数を設定しておくことができ、ディスク利用を有限回に制限するために用いることができる。尚、後述のUWデータ部のUSECOUNTデータと比較することで、この機能は実現できる。さて、セグメントの管理情報であるが、各セグメントの前マスタされた開始アドレスや長さはクラスター単位で与えられており、ビットマップを参照することで、ロードすべきセグメントがどれとどれであるかもわかる。従って、通常はこれらの情報を基に、システムローダ内に含まれているディスク制御プログラムを使って、フラッシュメモリ内のRS領域にソフトウェアをロードする。フラッシュメモリには、直前のディスクからロードされたソフトウェアが残存しているが、もし、今回インサートしたディスクが直前のディスクと同じものであったり、OSやアプリケーションのバージョンが同じである場合には、フラッシュメモリ内のソフトウェアをそのまま利用するほうがロードにかかる時間を節約できる。そこで、システムローダでロードを開始する前に、現在のフラッシュメモリ内に存在するソフトウェアをチェックする。前回ロードされたブート領域の情報、すなわちSYSLD、SPデータはRBT領域に、UW領域の情報はRUW領域に、ソフトウェアは全てRS領域に保存されているので、ROM部#1か

10

20

30

40

50

ら順にROM 1 ID、ROM 2 ID、・・・をチェックし、IDが異なるセグメント以降のものをロードする方式をとる。このことによって、再ロード時間を実質的に節約することができる。また、この制御判断はユーザの操作等は一切必要なく、自動的に行われるものである。

【0047】

すなわち、ステップS 16においては、システムローダのプログラムを実行する。具体的には、読み込まれたSPデータとRBT内のSPデータを比較し、必要なソフトウェアセグメントをディスクからRS領域にロードする。その後、RBT、RUWを更新する。RSに存在しているソフトウェアを実行領域である主メモリに移し、実行の先頭番地に制御を移す。ステップS 17では、OSやアプリケーション等のソフトウェアが順次立ち上がりユーザが使用できる状態になる。これに続いて、ユーザがアプリケーションで提供されている機能の操作を行っている状態がある。この状態では、ユーザはコンテンツを見たり、ユーザデータをRAM部に作成されたり、更新する。特殊なケースとして、別のメディアを入れ替えるために排出の禁止は解除される場合もある。そして、ステップS 18においては、OSやアプリケーションの正常終了に伴い、制御は上記BMAPのC点に移り、メディアの排出が禁止される。ステップS 19では、OSやアプリケーションによって更新されたUWデータをUW領域に書く。ステップS 20においては、メディアを自動排出し、電源遮断可能状態にする。上記BMAPのA点に制御が移る。そして、この一連の動作を終了する。

10

【0048】

続いて、ステップS 16における上記判断動作の動作の例を、図11のフローチャートに示す。

20

【0049】

ROM部の各セグメント属性を表わすIDは、本実施例ではROM 1 ID、ROM 2 ID、・・・で示されているが、このIDによって、各セグメントのバージョン管理、履歴管理情報等を表わすことができる。本実施例ではIDは4バイトで構成した。IDの決め方であるが、本アーキテクチャでは、異なるOSやいろんなミドルウェア(middleware)をも利用可能とするために、それぞれを区別していく必要がある。従って、その割り振りに関しては、インターネットのIPアドレスのように、発行元が一意であることが望ましく、体系的且つ一元的に管理する必要があるが、これは運用上の問題で解決される。また、ロード時のシステムローダの判断からわかるように、ROM部の小さい数字のIDのほうには、より共通なソフトウェアやデータを置き、後になるほど、変り得るソフトウェアを置くのが効率的となる。例えば、フォントデータ、辞書データ等のような、いろんなソフトウェアで共通に利用するようなデータは、最初のROM部に置くのが、効率的と言える。ロードが完了すると、主メモリ上のブート領域情報、すなわちSYSLD、SPデータをRBTに書き込み更新し、UWデータも後の使用に備えて、RUWに書き込み更新しておく。この時点で、フラッシュメモリには、挿入されたディスクと同じソフトウェアおよびそれらに関する情報が格納されたと言える。

30

【0050】

この後、システムローダはRS領域のソフトウェアを実行領域である主記憶メモリにマップし直して、実行の先頭番地に制御を移す。ここで、OSやアプリケーション等が順次立ち上がり、ユーザが使用できる状態となる。

40

【0051】

すなわち、最初のステップS 21においては、ROM 1 IDが一致するか否かが判断され、一致しすると“YES”としてステップS 23に進み、一致しないと“NO”としてステップS 22に進む。ステップS 22においてはROM部# 1セグメントを所定のRS領域にロードし、ステップS 24に進む。ステップS 23においてはROM 2 IDが一致するか否かを判断し、一致すると“YES”として次の判断のステップへ進み、一致しないと“NO”としてステップS 24に進む。ステップS 24ではROM部# 2セグメントを所定のRS領域へロードし、次のセグメントをロードするステップに進む。ステップS 2

50

5においてはROM4IDが一致するか否かを判断し、一致すると“YES”としてステップS27に進み、一致しないと“NO”としてステップS26に進む。ステップS26では、ROM部#4セグメントを所定のRS領域へロードする。ステップS27は、主メモリ内のブート領域データであるSYSLD、SPデータでRBTを更新し、UWデータでRUWを更新する。ステップS28においては、RS領域の全てのプログラムを主メモリに移し、実行の先頭番地に制御を移し、この一連のステップS16の工程を終了する。

【0052】

アプリケーションが立ち上がると、本体はそのアプリケーションの世界一色になり、ユーザはそのアプリケーションで提供される機能を使って、コンテンツを見たり、ユーザデータを作成して、必要に応じてディスクのRAM部に記録、更新していく。アプリケーションによっては、別のディスクを入れ替えたりする場合もあるが、この場合には、アプリケーション側からのディスク排出禁止解除が適宜なされてもよい。但し、アプリケーションを終了するにあたっては、いつでも、最初のディスクが再び、本体にインサートされ、ディスク排出禁止がなされていなければならないが、これはアプリケーション側の操作・制御によって実現される。

【0053】

アプリケーションが稼動している状態においては、本アーキテクチャがサポートする種々のメタデータ、すなわちUWデータとして定義されているものをフラッシュメモリ内のRUW領域に適宜記録、更新しておくことができる。例えば、USECOUNTとは、本ディスクが一回利用される度に1つつカウントアップされるもので、本ディスクの利用回数を示している。USECOUNTの更新は、例えば、アプリケーション開始の直後とか、正常終了の直前になされてもよい。UWデータの利用方法は、アプリケーション開発側に秩序ある形で開放され、その内容の定義、解釈はアプリケーション開発元、ディスク供給元等が技術面とビジネス面の両方において運用していくことができる。UWデータの内容はディスク毎に定義可能であるので、各ソフトウェア供給ベンダーが自由に定義しても矛盾はおきないが、共通項目に関しては、ある程度共通定義しておく。図9の例はその一端をしめしている。

【0054】

すなわち、図9に示したUWデータにおいて、USED FLAGは初めてディスクが利用された時にオンとなる。このUSED FLAGによって、ディスクの新旧の区別が可能である。IDSTAMPは、情報処理装置の本体側にハードIDがある時は、そのIDを書き込むことで、ディスクがどの本体で利用されたかがわかる。本実施の形態では、図7のBMAPの先頭256バイト領域に確保している。情報処理装置のハードウェアの種類を特定するためである。情報処理装置の本体側のハードウェア上のラインアップ、メーカー別、機種別などの区別をしたい場合このデータを用いることで、ディスクとの互換性等が制御可能となる。USECOUNTは、ディスクが一回利用される毎にカウントアップされる。情報処理装置の本体側で利用回数を制限することやビジネス上の課金の基礎データ等に使用できる。USETIMEは、年月日時分秒形式で記録され利用時間等を記録しておくことができる。ERRSTATUSは、エラーが発生した場合のログデータを記録できる。OSやアプリケーションが認識できるエラーが生じた場合はそのエラーのログを取ることが可能であるので、ログデータをRUW部の所定の場所に記録しておく。上記BMAPのC点動作の中で、このログデータはディスクに記録されるので、エラーの原因、修理等の情報として役立つ。

【0055】

本実施の形態のアーキテクチャでは、アプリケーションの終了に伴い、UWデータを更新するため、再び上記BMAPに制御が移ってこなければならないが、この為には、アプリケーションやOSにおいて、ソフトウェアの終了が明示的に意識され、正常終了、異常終了のどちらの場合も全て、制御が上記BMAPのC点に戻るようプログラミングされている必要がある。これは各アプリケーションやOSが終了にあたってC点にジャンプするような命令をプログラミングしておくことで解決する。ジャンプ先である上記BMAPの

10

20

30

40

50

C点の開始アドレスは図7のように、上記BMAP内の固定位置に定められるから、上記BMAP内のプログラム量の増減に係わらず、確実にC点へジャンプできる命令をプログラムすることができる。尚、プログラムの暴走という事体が生じる場合があるが、そのような場合に備えて、強制的にC点動作に入れるようなスイッチを本体側に設けておくこともできる。

【0056】

C点動作に入ると、念のため、ディスク排出禁止がなされ、RUWデータをディスクのUW領域に書き込む。これによって、ディスクがどのような状態で利用されたかのメタ情報がディスク自身に記録されることになる。ここで、ディスク排出は解除され、自動排出コマンドの発行、または、本体の排出ボタンを操作するかして、ディスクが取り外される。そして電源遮断（パワーオフ）可能状態となり、制御は次のアプリケーション待ちとして、上記BMAPのA点に戻る。

10

【0057】

以上が本実施の形態のアーキテクチャの動作説明である。本アーキテクチャに基づけば、ディスクをインサートするだけで簡単に起動し、専門家以外あるいはパソコンやコンピュータ知識のない素人ユーザが使いやすいコモディティ化された家庭用のコンピュータ（ホームコンピュータ）が実現できるものである。

【0058】

尚、上述の本実施の形態の中で、システムローディングの部分であるが、初期のパソコンにおいて、フロッピーからシステムロードする形態があったが、ハードウェア側にBIOSというデバイス駆動プログラムの一部が搭載されており、本実施の形態のアーキテクチャのようにソフトウェアやコンテンツの全てをディスク側に置くという発想ではなかった。また、アプリケーションをCD-ROMからロードするという形態はCD-I、ゲーム機で採用されているが、OS等はハード側に置かれていた。いずれにしても、本アーキテクチャにおける上記BMAP動作のように、アプリケーション終了後に再び上記BMAPに制御が移り、メディアをトータルにコントロールしていく形態のものはなかった。

20

【0059】

次に、本発明を記録媒体に適用した実施の形態について説明する。

【0060】

本発明に係る記録媒体の実施の形態としては、情報を記録された記録媒体であって、情報処理装置における処理に必要な全てのソフトウェアとコンテンツを記録された記録媒体を用いる情報処理装置に利用され、上記記録媒体に対する情報信号の記録/再生を行う記録/再生工程と、情報を記憶する記憶工程と、上記各工程を制御し、上記記録媒体が上記情報処理装置に挿入されると、上記記録/再生工程にて上記記録媒体から必要なソフトウェアやコンテンツを上記記憶工程に読み込んで実行し、必要な情報を上記記録媒体に記録し、上記ソフトウェアの実行を終えると上記記録媒体を排出するように制御する制御工程とによって実行される少なくともデバイスドライバが記録されてなるものを挙げる事ができる。

30

【0061】

また、本発明に係る記録媒体の実施の形態としては、情報を記録された記録媒体であって、情報処理装置における処理に必要な全てのソフトウェアとコンテンツを記録された記録媒体を用いる情報処理装置に利用され、上記記録媒体に対する情報信号の記録/再生を行う記録/再生工程と、情報を記憶する記憶工程と、上記各工程を制御し、上記記録媒体が上記情報処理装置に挿入されると、上記記録/再生工程にて上記記録媒体から必要なソフトウェアやコンテンツを上記記憶工程に読み込んで実行し、必要な情報を上記記録媒体に記録し、上記ソフトウェアの実行を終えると上記記録媒体を排出するように制御する制御工程とによって実行される少なくともオペレーティングシステムが記録されてなるものを挙げる事ができる。

40

【0062】

更に、本発明に係る記録媒体の実施の形態としては、情報を記録された記録媒体であって

50

、情報処理装置における処理に必要な全てのソフトウェアとコンテンツを記録された記録媒体を用いる情報処理装置に利用され、上記記録媒体に対する情報信号の記録/再生を行う記録/再生工程と、情報を記憶する記憶工程と、上記各工程を制御し、上記記録媒体が上記情報処理装置に挿入されると、上記記録/再生工程にて上記記録媒体から必要なソフトウェアやコンテンツを上記記憶工程に読み込んで実行し、必要な情報を上記記録媒体に記録し、上記ソフトウェアの実行を終えると上記記録媒体を排出するように制御する制御工程とによって実行される少なくともアプリケーションが記録されてなるものを挙げる事ができる。

【0063】

そして、本発明に係る記録媒体の実施の形態としては、情報を記録された記録媒体であって、情報処理装置における処理に必要な全てのソフトウェアとコンテンツを記録された記録媒体を用いる情報処理装置に利用され、上記記録媒体に対する情報信号の記録/再生を行う記録/再生工程と、情報を記憶する記憶工程と、上記各工程を制御し、上記記録媒体が上記情報処理装置に挿入されると、上記記録/再生工程にて上記記録媒体から必要なソフトウェアやコンテンツを上記記憶工程に読み込んで実行し、必要な情報を上記記録媒体に記録し、上記ソフトウェアの実行を終えると上記記録媒体を排出するように制御する制御工程とによって実行される少なくともコンテンツが記録されてなるものを挙げる事ができる。

【0064】

また、本発明に係る記録媒体の実施の形態においては、書き換え可能または一部が書き換え可能で他の部分が書き換え不可能であるときには、ソフトウェアを終了する際に、上記記録媒体の所定の領域にシステム情報、エラー情報、利用関係、年月日の少なくとも一つを含む情報を上記記録媒体の記録するようにすることができる。

【0065】

尚、これらの記録媒体すなわちメディアとしては、フロッピーディスクのようなものから、100MBから200MB程度の大容量フロッピーディスク、いわゆるMOやいわゆるMDデータ、いわゆるCD-ROM、いわゆるDVD-ROM、ICメモ리카ード、いわゆるPCカード、いわゆるメモリスティックなどであってもよい。また、これらの記録媒体の内容は、通信回線を介して提供することもできる。

【0066】

以上の説明は、本実施の形態のアーキテクチャを実現するためのリムーバブルメディアの具体例として、いわゆるハイブリッドMDデータを中心に述べてきたが、その中でのデータフォーマット上の変形例も幾つか考えられる。また、本実施の形態のアーキテクチャの有効性は上記ハイブリッドMDデータのみにとどまらず、次世代MDや他のリムーバブルメディアでも実施可能である。また、情報処理装置の本体を実現する変形例も、幾つか考えられる。更に本実施の形態のアーキテクチャに基づく新しい機能、形態も考えられる。これらについて以下、順次説明する。尚、今までの説明では上記ハイブリッドMDデータを中心に説明したので「ディスク」という用語を用いたが、以下の説明ではより一般的な用語として「メディア」を用いて記述する。

【0067】

まず、本実施の形態の第1の変形例について説明する。本実施の形態のアーキテクチャではリムーバブル側に全てのソフトウェアやコンテンツを搭載するが、これが可能なメディアとしては、RAM型、ROM型、ROM/RAM一体のハイブリッド型のどれでもよい。但しROM型の場合は、UWデータ部を書き込む領域が確保できないため、UWデータがメディアと一体化されて初めて意味が出てくる機能は実現されないが、UWデータをフラッシュメモリ内のRUWには保存できるので、使用するメディアが一枚に限られるアプリケーションやメディアの区別の必要のない場合にはRUWデータは同じように活用できる。ROM型の場合は、通常データのビューイング(viewing)だけに限定されるアプリケーションが多いが、ROM型MDデータ、いわゆるCD-ROM、いわゆるDVD-ROM等のメディアを用いた場合がこれに該当する。尚、RAM型とハイブリッド型はメデ

10

20

30

40

50

ィア製作過程が異なるのみで、本アーキテクチャから見た場合の機能上は全く同等に考えてよい。

【 0 0 6 8 】

続いて、本実施の形態の第2の変形例について説明する。リムーバブルメディアの物理形態には大きくディスク型と非ディスク型があり、前者は、100MB～200MB程度の大容量フロッピーディスク、いわゆるFD、いわゆるMO、いわゆるCD、いわゆるMD、いわゆるDVD、いわゆるPD、リムーバブルのHDDなど、次世代光ディスク/磁気ディスクがある。後者には、半導体をベースとしたいわゆるPCカード、各種メモリカード、ICカード、メモリスティックなどがある。カード系であっても、ROM ICとRAM ICを混在させることができるので、ハイブリッド型が実現できる。テープ系はRAM型であって、理論上は本実施の形態のアーキテクチャで実現可能だが、ランダムアクセスの点であまり実用的ではない。いずれにしても、今後開発されてくる全てのリムーバブルメディアに対し、本アーキテクチャにおいて、一つの物理メディアを定めれば、それに応じたドライブが定まるから、本体側の上記BMAPは容易に作成でき、また、メディア側でのソフトウェアセグメントの配列も本説明に沿った形で容易に定義できるので、本アーキテクチャがカバーできる範囲と考えることができる。

10

【 0 0 6 9 】

続いて、本実施の形態の第3の変形例について説明する。情報処理装置の本体に唯一存在する上記BMAPは本説明では、内蔵フラッシュメモリの一部を利用したが、上記BMAP ROMとして内蔵フラッシュメモリと分離することもできる。情報処理装置の本体を生産する場合に、上記BMAP ROMだけ別の部品のほうが取り扱いが便利なこともある。また、上記BMAPにおいて、初期起動時間を節約するため、情報処理装置のハードウェアの初期化の後にシステムロード等の読み込みを開始するという直列制御ではなく、適宜ドライブのインタラプト処理を組み合わせることでドライブのウェイト(wait)状態のときに他の部分のハードウェアの初期化を平行して実行するように上記BMAPをプログラミングしておくこともできる。

20

【 0 0 7 0 】

続いて、本実施の形態の第4の変形例について説明する。上記BMAPおよびシステムロードのプログラム内には、メディアの基本単位をREAD/WRITEするプログラムが入っているが、これらは共通化して一つにして、BMAP内に置くこともできる。更に、そのルーティンの開始アドレス、引数のポインタなどを上記BMAPの先頭256バイト内の保留領域を使用して、固定位置に定義しておけば、システムロードやアプリケーションからも利用することが可能となる。

30

【 0 0 7 1 】

続いて、本実施の形態の第5の変形例について説明する。内蔵のフラッシュメモリの部分は、スタティックRAM等でもよく、必ずしもフラッシュメモリでなければならないわけではない。基本的には不揮発性のメモリならなんでもよい。

【 0 0 7 2 】

続いて、本実施の形態の第6の変形例について説明する。本説明においては、いわゆるハイブリッドMDデータにUWデータ部を書き込むUW領域として、リードアウト領域内にそれを新たに定義したが、別の場所に定義することも可能である。例えば、UTOC内には保留領域も残されており、そこをUW領域の候補とすることもできる。その場合は、フォーマットコマンドでUTOCを書き換える前には、UWデータ情報を主記憶にコピーしておき、後で更新するようにしていけばよい。記録可能ユーザ領域の一部をUW領域として定義し、使用する場合も同様である。

40

【 0 0 7 3 】

続いて、本実施の形態の第7の変形例について説明する。ブート領域のサイズ、UW領域のサイズは本説明では64KBとしたが、これは上記ハイブリッドMDデータの読み出し書き込み単位が1クラスターが64KBのほうが都合がよいためである。異なるメディアではそのメディアのアクセス単位に沿った値で構わない。

50

【 0 0 7 4 】

続いて、本実施の形態の第 8 の変形例について説明する。ソフトウェアセグメントの個数は、本説明では、最大 2 5 6 個としたが、ビットマップを増やすことや、場合によっては S P データ部のサイズを 4 K B 以上にするこゝで、大きく取ることゝできる。

【 0 0 7 5 】

続いて、本実施の形態の第 9 の変形例について説明する。上記 B M A P がブート領域からシステムローダや S P データ部をロードしてくる場所は、本説明では主メモリとしたが、内蔵のフラッシュメモリであっても構わぬ。この場合は内蔵のフラッシュメモリ内に 6 4 K B の領域をブート領域ロード用に別途確保すればよい。

【 0 0 7 6 】

続いて、本実施の形態の第 1 0 の変形例について説明する。上記 B M A P 以外には内蔵のフラッシュメモリが搭載されていない情報処理装置のハードウェア構成も考えられる。本実施の形態のアーキテクチャに沿った小型の情報処理装置を商品化する場合などは、コストの面からそのような考えも成り立つ。この場合は、内蔵のフラッシュメモリに直前のソフトウェアが残存していることがないため、メディアを情報処理装置の本体の挿入するたび毎に、必ず必要な全ソフトウェアセグメントが主メモリに直接ロードされる。この場合、B M A P 内の R B T、R U W、R S の開始アドレス等は主メモリのアドレスに設定されていなければならないが、図 1 0 のシステムローダのプログラムの実行時に R B T、R U W の新旧比較をする必要はなくなる。但し、R B T や R U W は固定長なので、B M A P がフラッシュメモリで構成される場合には、B M A P の空いている場所に領域を確保することもできる。

【 0 0 7 7 】

続いて、本実施の形態の第 1 1 の変形例について説明する。内蔵のフラッシュメモリのサイズは、基本的には必要なソフトウェアセグメントを全て格納できるサイズが必要である。本実施の形態のアーキテクチャで対象としているソフトウェアはそれほど重いものではなく、一つ一つは軽快なものであるべき、と考えており、基本的には数 M B 程度以下と考えられるが、それはコストや本体の物理的大きさ等から、生産するメーカー判断にまかされている。もし、ハードウェアが搭載している内蔵のフラッシュメモリよりも大きなソフトウェアセグメントを搭載したメディアが挿入された場合は、オーバーした部分は、連続的に主記憶領域にロードしていけばよい。この場合は、全てのソフトウェアセグメントが不揮発性とならないので、主記憶にロードされた部分は、同じメディアが挿入されても毎回再ロードされていかななければならない。これは例えば、ロード後の R B T 内の主メモリにロードされた該当セグメントの R O M I D を一定の予約ビットパターンで書き換えておくことで、次に挿入された時、該当 R O M I D がマッチしないようにすることで再ロードを実行させていくことができる。尚、予約ビットパターンは実際のソフトウェアセグメントには絶対に割当てられることのないビットパターンとして予め定めておくことで可能である。また、全ての R O M I D を予約パターンに書き換えておいて、全セグメントを再ロードさせてもよい。更に、各 R O M 部の長さを予めシステムローダのプログラム内において計算し、オーバーする場合には内蔵のフラッシュメモリを経由せず、直接主メモリにロードするという方法もある。この場合は内蔵のフラッシュメモリの実装されていない本体の動作と同等になる。

【 0 0 7 8 】

続いて、本実施の形態の第 1 2 の変形例について説明する。上記とは逆に内蔵のフラッシュメモリのサイズが通常ロードするソフトウェアセグメントのトータルサイズの数倍になって実装されている場合は、複数個のシステムをフラッシュメモリ内に残存させておくことができる。そうすればその分だけヒット率が上がり、再ロードせずにアプリケーションを立ち上げる確率を高くすることができる。これは内蔵のフラッシュメモリ内に複数のシステムが残存できるように R S 領域分割管理情報を設け、システムローダが新旧比較する時点で、これら複数の R S 領域をそれぞれ比較して行うことゝ実現できる。

【 0 0 7 9 】

続いて、本実施の形態の第13の変形例について説明する。アプリケーションが立ち上がった後は、マシン全体のハードウェアリソースは、全てそのアプリケーションの制御下に置かれていくことになる。本アーキテクチャでは、原則として、ソフトウェアをロードするために最初に挿入されたメディアがそのまま書き込み可能なデータ記録装置の役目を果たすが、アプリケーション次第では、複数枚のメディアを使用したい状況も出てくる。例えば、コンテンツが一枚のメディアに入りきれない場合はアプリケーションの進行に従い、複数枚のROMメディアが順次差し替えられていくことがある。また文書ファイリングアプリケーションや文書作成の場合、あるいは、バックアップコピーやデータコピー機能を実現するためには、ブランクメディア(RAM型)の追加が必要ともなってくる。このような状況下においては、メディアの差し替え制御は全てアプリケーションの制御にまかされるが、アプリケーションを終了する時点では、上記BMAPのC点動作以降でUWデータを書き込み更新するために、元のメディアがドライブ差し込まれている必要がある。元のメディアが差し込まれているかどうかは、それを排出する時に、アプリケーション側で制御可能な書き込み領域に、のちに照合できるような情報を書き込んでおくかして、そこを参照することによって、違うメディアの場合は、ユーザに元のメディアを差し込むようにメッセージを出すなりして、促すことができる。もう少し、統一的に取り扱う方法としては、アプリケーション側はどのメディアが差し込まれている状態でも終了してよいことにする代わりに、元のメディアを排出する際、そのインタラプト処理において上記BMAPへ制御を一時移し、上記BMAP側でUWデータにメディア特定情報を記録し、アプリケーションが終了して再び上記BMAPのC点動作が開始した時点で、まず、UWデータを再び読み込み、違うメディアが差し込まれている場合は、それを排出し、元のメディアをユーザに差し込むように促すこともできる。

10

20

【0080】

続いて、本実施の形態の第14の変形例について説明する。アプリケーション等の異常終了、暴走等のため、本体側に強制的に図7に示したC点に移行させるスイッチを設けておくことができる。一種のリセットスイッチの役割を果たす。

【0081】

続いて、本実施の形態の第15の変形例について説明する。一般的にメディアにはその種別を示す情報がTOC等書かれているので、それを参照することでROM型メディアと確認できた場合は、上記BMAPのC点動作においても、UWデータをメディアに書き込まないようにプログラミングしておくことができる。

30

【0082】

本実施の形態のアーキテクチャの新しいメリットとして、例えば次のような機能が実現できる。

【0083】

第1の機能としては、メディアのコピーがある。ユーザやシステムの作成するRAM部データが、そのメディアに搭載されたコンテンツに関連している場合は、同じコンテンツの搭載されているメディアへのRAM部分のコピーが有用となる。異なるコンテンツのメディアにRAM部分をコピーしてもあまり意味はない。例えば、コンテンツが教科書データであって、RAM部が教科書データを参照する形で作成される電子ノートなどの場合が該当する。もちろん、著作権保護の立場から、ROM部のデータをコピーするようなことは避けなければならないから、コピーできる部分はRAM部のみに限られる。このような場合は、コピー先のメディアが、コピー元のメディアと同じコンテンツを有していることを判別しなければならないが、これは、アプリケーションソフトウェアが、コピー先メディアのSPデータ部のROMIDとコピー元メディアのSPデータ部のROMIDとを比較することで実現できる。コピーするためには、そのコンテンツを搭載したメディアを用意する必要があるから、コンテンツ供給者から見れば、メディアが一枚売れることになり、よ言われるように、コンテンツがコピーされてしまうことにはならないし、ユーザ側からすれば、自分の作成した電子ノートを自由に誰かにコピーさせてあげることができる。

40

この意味においては、本アーキテクチャに基づくメディアコピー機能は、コンテンツ供

50

給者、ユーザ側の両方を満足させることができるものと言える。

【0084】

2台または、それ以上のドライブが実装されている本体の場合にも、それぞれのコピー元、コピー先のドライブに入っているメディアのSPデータ部のROMIDを比較することで行える。尚、メディアコピーのように全く同じメディアを複製するのではなく、単にRAM部分を別のメディアにバックアップしておくような場合には、SPデータ部の比較を行う必要がないのは言うまでもない。

【0085】

第2の機能としては、ソフトウェア利用回数の制御がある。SPデータ部の利用方法の一つとして、USELIMITがある。ここには、メディア供給元が前マスタする時に、このメディアの使用回数の最大値を書いておく。一方、メディアはそこに搭載されたアプリケーションが本体で稼動される（使用される）度に、UWデータ部のUSECOUNTを一つずつカウントアップしていく。ある時点でUSECOUNTの値がUSELIMITの値を超えれば、そこでこのメディアはそれ以後使用できなくするようにアプリケーション側で制御可能となる。現行のソフトウェアにおいて「お試し版」というものがあるが、これは機能を制限したものや、本体内の時計との比較により期限限定のようなものが見受けられるが、本アーキテクチャでは、使用回数を制限することで、代わりにフル機能ソフトウェアのままの形をユーザに提供できる。ビジネス上の見地からは、同じフル機能版であっても、USELIMITの値のみを変更するだけで、販売価格を複数通りに設定することも可能となり、販売戦略に自由度ができ、供給元、ユーザの両方にとって理想的な価格設定を実現していくことも可能となる。USECOUNTのデータをインターネット等を通してサーバー側で計算するような課金方式も実現できる。

【0086】

以上説明したように、本実施の形態は、一枚のリムーバブルディスクにデバイスドライバ、OS、アプリケーション、コンテンツが全て搭載され、操作結果のデータ等もそのメディアに書き込むことによって、本体とメディアが完全に分離され、メディアの交換のみでいろいろなアプリケーションが可能となるようなコンピュータシステムアーキテクチャを提供するものであって、コンピュータの知識のない人でも容易に操作でき、具体例としてMDデータを用いたものを示した。

【0087】

すなわち、本実施の形態は、ハードウェア本体にはハードディスクのような内蔵の記録装置を一切持たず、リムーバブルメディアを唯一の記録装置として、そこに全てのソフトウェア、アプリケーション、コンテンツが搭載され、本体にメディアをインサートすると自動的に必要なソフトウェアが本体内にロードされ、コンピュータとして稼動し、メディアが排出された時点でコンピュータとしての機能が消滅し、別のメディアをインサートすると、再び、そのメディアに搭載されたソフトウェアによって、別のコンピュータとして稼動できるものである。従って、一つのハードウェアで、さまざまなOSやアプリケーションをメディアの差し替えだけで簡単に実現できる。本コンセプトはRAM型、ROM型、ROM/RAM一体のハイブリッド型のどのメディア種別にも適用でき、また、物理形態としてはディスクタイプや半導体によるICメモリカード等にも適用できる。

【0088】

また、本実施の形態は、RAM型メディアおよびハイブリッド型メディアを使った場合には、コンピュータとして機能している間に生成されるシステムデータ、ユーザデータ類は、アプリケーションの終了と同時に、そのメディアに全てが記録され、この一枚のメディアだけを携帯することで、いつでも、どこでも、自分のコンピュータ環境が100%再現できるものである。言い換えれば、本実施の形態によって、自分用の作業ハードウェアという概念が必要なくなる。

【0089】

更に、本実施の形態は、本体にはメディアに搭載されるさまざまなソフトウェアとは無関係の3つの部分に別れたプログラム群が上記BMAPという形で搭載されており、メディ

10

20

30

40

50

アがインサートされるとまず、このメディアのソフトウェアを本体にロードするシステムローダというプログラム、及び、ローディングを制御するSPデータがメディア上の固定的に定められた場所からロードされる。メディア上のソフトウェアやコンテンツは複数個のセグメントに分割されており、各セグメントにバージョン等を区別するIDが設けられており、システムローダは、これらのSPデータ情報を参照しながらに所定の場所に実際のソフトウェアローディングを行う。アプリケーションの終了にあたっては、本体内の上記BMAPに再び制御が戻ってきて、発生したシステムデータ、エラー情報、利用回数、年月日等のUWデータをメディア上の固定的に定められた場所へ書き込むことによって、全体の動作が完了するローディング、アンローディングメカニズムに基づくものである。メディアの挿入から排出までを本体のBMAPが制御する。

10

【0090】

そして、本実施の形態は、情報処理装置の本体のハードウェアにはハードIDが上記BMAPの一部に予め記録されており、必要に応じて、ハードIDをUWデータ部に記録しておくことにより、本体とメディアとの互換性を認識できるものである。

【0091】

また、本実施の形態は、情報処理装置の本体側に不揮発性メモリ、例えばフラッシュメモリが搭載されているハードウェアの場合には、ローディングの際にこのセグメントIDをフラッシュメモリ内のすでにロードされていたソフトウェアのIDと比較することで、同じソフトウェアローディングを2重に行うことを避け、起動時間を節約できる構造を持ったものである。

20

【0092】

更に、本実施の形態は、メディアとして、いわゆるMDデータまたはいわゆるハイブリッドMDデータを採用し、新たにリードアウト領域、またはUTC領域等にUW領域を設け、すでに定義されているブート領域とこのUW領域を用いて、上記の機構を具体的に実現するものである。

【0093】

そして、本実施の形態は、SPデータ部のROMIDの情報を比較することにより、自分と同じコンテンツをすでに搭載しているメディアにしか、自分のRAM部のデータをコピーさせないことが実現できる。このことにより、コンテンツ供給者の著作権を保護しつつ、ユーザ側で自由にメディアの複製を行うことができる。

30

【0094】

更に、本実施の形態は、起動時にローディングされるSPデータ部には、ローディングの制御データとメディアの管理情報が定義され、メディア管理情報にメディアの利用に関するデータを予告記述しておき、実際のメディアの利用に関するデータをUWデータ部に記録することで、本体がそれらと比較することで、メディアの利用を制御できるコンピュータシステム。例えば、実施例では、USELIMITとUSECOUNTによって、メディアの利用回数を制御できる。

【0095】

尚、本実施の形態においては、メディアとしてMDメディアを例示したが、本実施に用いるリムーバブルメディアがMDデータに限定されないことは言うまでもない。

40

【0096】

また、本実施の形態においては、ハードディスクを内蔵しないアーキテクチャについて説明したが、本発明はこれに限定されない。本発明の実施の形態は、ハードディスクを内蔵するアーキテクチャに適用することができることはもちろんである。

【0097】**【発明の効果】**

上述のように、本発明のアーキテクチャのシステムコンセプトに基づけば、メディアには全てのソフトウェア、データ、コンテンツが搭載されるので、通常のパソコン等で必要なインストール作業が全く要らない。また、インストール後におけるアプリケーションの初回起動準備作業としてシステムパラメーター等を設定するなどの、いわゆる、セットアップ

50

プ作業も必要なく、購入したメディアをインサートするだけでマシンが稼動する。つまり、コンピュータの技術知識がない人でも簡単に管理、操作でき、家電商品のような簡単な取り扱いが実現される。

【0098】

また、本発明によると、ソフトウェアのバージョンアップ、アップグレードが行われた場合であっても、新旧のバージョンの「差分」をユーザみずからマシンにインストールするなどの作業は必要なく、単に、新バージョンの搭載されたメディアを利用するだけでよいので、開発供給元、ユーザ側共に、簡単でわかりやすく、取り扱いやすい。

【0099】

更に、本発明によると、一つのメディアに一つのアプリケーションに関するソフトウェア、コンテンツ、ユーザデータの全てを自己完結した（self-contained）形で搭載できるので、他のアプリケーションとの相互調整、共通ライブラリとの関連など、ややこしく且つ複雑な事項を全く気にせず、シンプル且つ安定したアプリケーション環境をユーザに提供できる。また、一つのメディアに全ての情報が記録され、本体ハードウェア側には関連した一切のデータが残らないので、メディアと本体が完全に分離され、メディアだけを携帯して、いつでも、自分のコンピューティング環境が再現できる。自分の作業用ハードウェアを特定しておく必要がないハードウェアフリーな環境を提供する。

10

【0100】

そして、本発明によると、情報処理装置の本体のハードウェアとは独立に、OSやアプリケーションを自由に選択して、供給メディアを開発、販売できるので、ソフトウェア開発、コンテンツ開発の自由度が大きく、ハードウェアの汎用性も高まる。

20

【0101】

また、本発明によると、メディアを変えれば、異なるアプリケーションが可能で、その意味では、専用機のような簡単さを保持したまま、多目的（multiple purpose）な適用が可能なマシンを実現できる。例えば、ワープロソフトウェアを搭載したメディアを入れれば、本体はワープロ専用機となり、電子メールソフトウェアのメディアを入れれば、電子メール端末となる。教育ソフトウェアのメディアを入れれば教育機器となり、電子本ソフトウェアを入れれば電子書籍ビューアとなる。

【0102】

更に、本発明によると、メディアをインサートしないと、ハードウェアは稼動しないので、通常のパソコン等で問題になるインストールコピー（install copy）ができない仕組みになっており、ソフトウェア供給側にも不正コピーに悩まされずに済むというメリットがある。

30

そして、本発明によると、メディアのROM部、または、予め供給側によって記録されている部分はコピーできないが、同じROM部、または、予め記録されている部分を同じように搭載している別のメディアに対しては、RAM部をコピーすることができ、結果として、著作権を侵害せずにメディアのコピーが実現できる。

【0103】

更に、本発明によると、ソフトウェアの使用回数をカウントできるので、これによって、ソフトウェア開発側が使用回数に応じた販売価格や課金方法を科していくことができる。

40

【図面の簡単な説明】

【図1】本発明の実施の形態を適用した情報処理装置の概略を説明する図である。

【図2】本発明の実施の形態を適用した小型の情報処理装置およびこの情報処理装置に用いられるリムーバブルディスクの外観を示す図である。

【図3】本発明の実施の形態を適用した情報処理装置およびリムーバブルディスクの概略的な構成を示すブロック図である。

【図4】いわゆるハイブリッドMDデータの基本フォーマット構造を示す図である。

【図5】各ソフトウェアのセグメント配置例を示す図である。

【図6】ブート領域の構成例を示す図である。

【図7】基本メディアアクセスプログラムBMAPの構成例を示す図である。

50

【図8】内蔵のフラッシュメモリ内のデータ配置構成例を示す図である。

【図9】UW領域の構成例を示す図である。

【図10】本実施の形態の基本アーキテクチャの動作例を示すフローチャートである。

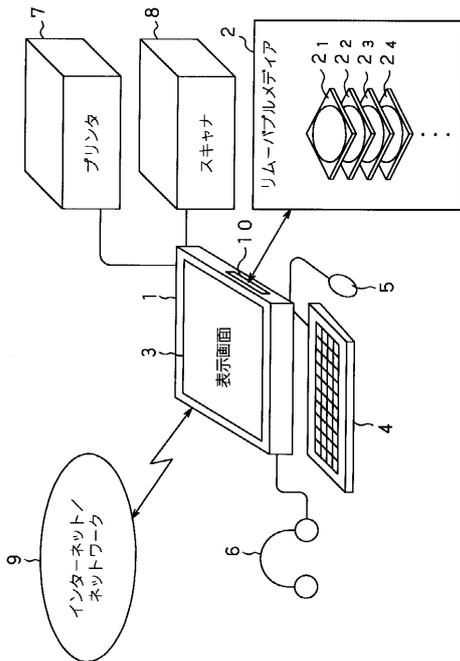
【図11】システムローダプログラムの実行例を示すフローチャートである。

【図12】ソフトウェアの階層を説明する図である。

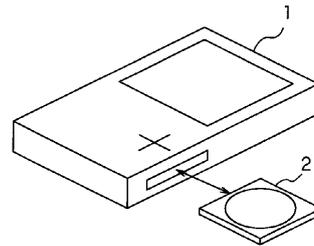
【符号の説明】

- 1 情報処理装置、2 リムーバブルディスク、3 表示画面、4 キーボード、5 マウス、9 インターネット/ネットワーク

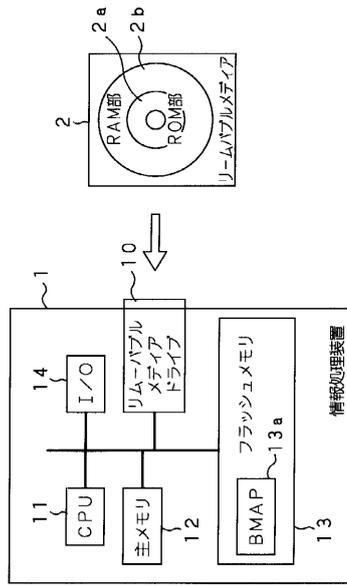
【図1】



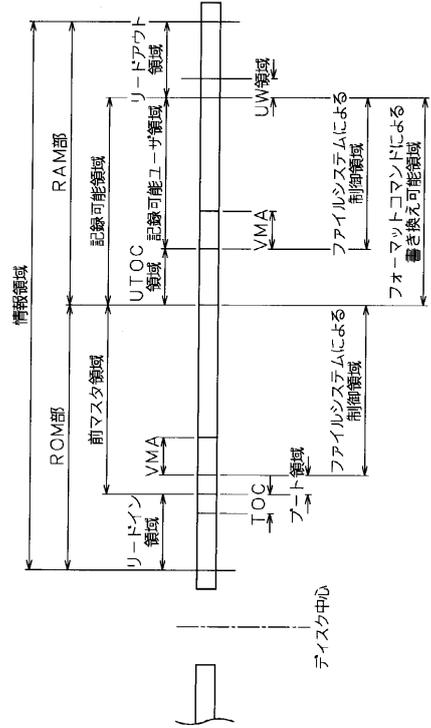
【図2】



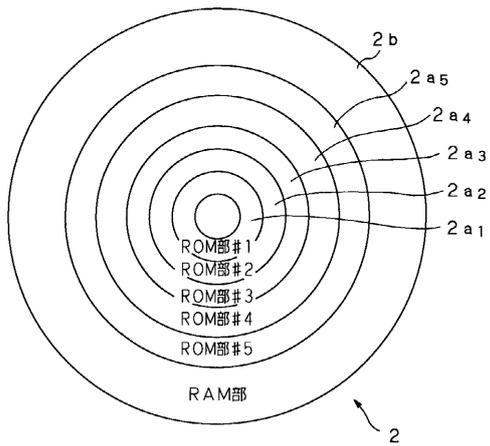
【図3】



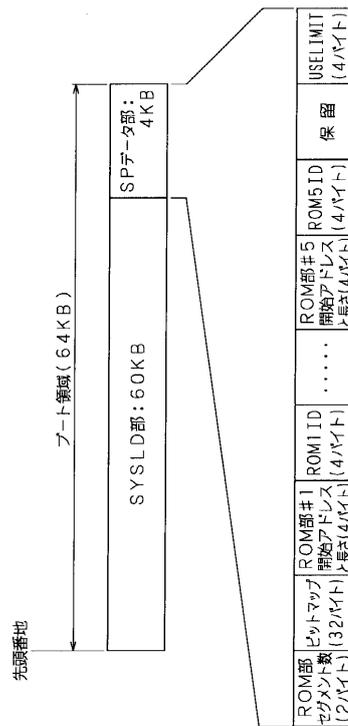
【図4】



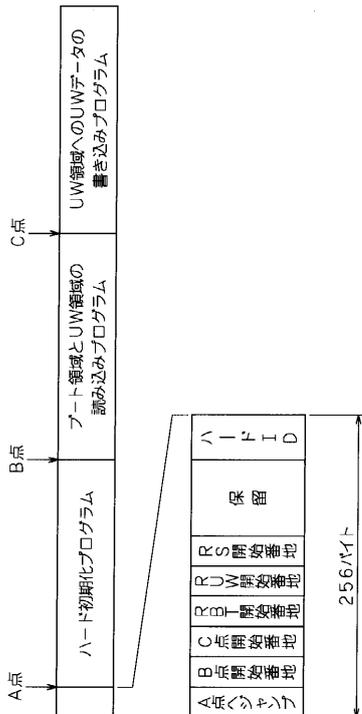
【図5】



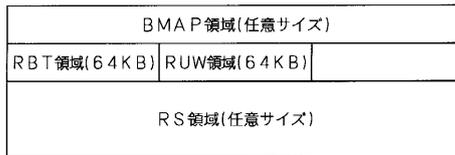
【図6】



【 図 7 】



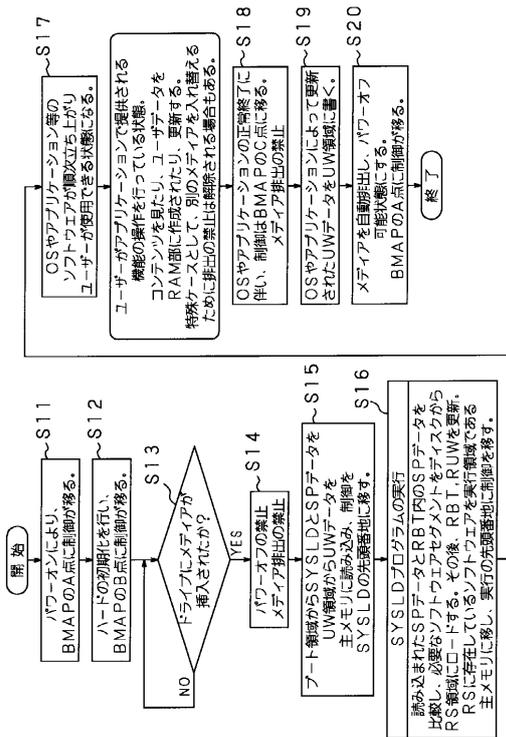
【 図 8 】



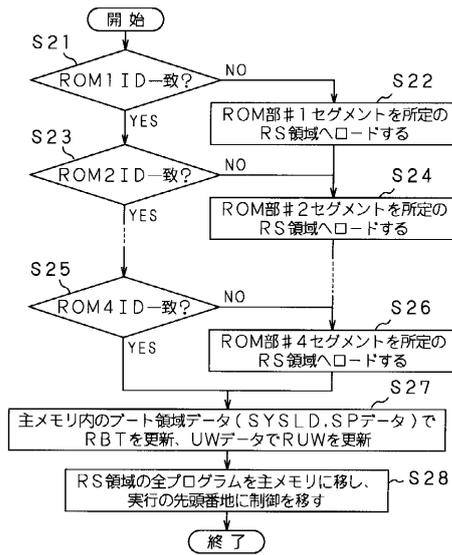
【 図 9 】



【 図 10 】



【図11】



【図12】

アプリケーションプログラム (APP)
アプリケーションプログラムインターフェース (API)
オペレーティングシステム (OS)
デバイスドライバ (DD)
ハードウェア (H/W)

フロントページの続き

- (56)参考文献 特開平10-21084(JP,A)
特開平1-240930(JP,A)
特開平10-40110(JP,A)
特開平7-319709(JP,A)
特開平5-289854(JP,A)
村瀬康治,「入門MS-DOS」,日本,株式会社アスキー,1989年1月1日,第2版,
pp.131-135,151-157,ISBN:4-87148-745-8
小川紘一,「3.5インチ光磁気ディスクとその応用」,FUJITSU,日本,富士通株式会
社,1993年7月19日,Vol.44, No.4, pp.347-359, ISSN:0016-2515
纈纈一起・他,「MS-DOS」,日本,共立出版株式会社,1988年3月10日,初版,
pp.31-35,50-55,ISBN:4-320-02398-6
酒井雄二郎・他,「MS-DOS Ver.3.3ハンドブック」,日本,株式会社ナツメ社,1988年,
p.224,ISBN:4-8163-0886-5
Dino Esposito(翻訳・日下部圭子),「フロッピードライブでAutoPlayをテストする」,Insid
e Windows,日本,ソフトバンク株式会社,1997年11月1日,Vol.3, No.11, pp.113-11
6

(58)調査した分野(Int.Cl.,DB名)

G06F9/445,
G06F9/50,
G11B19/02,
G11B20/10,
CSDB(日本国特許庁)