



The interface between Firmware and Operating System : ACPI implementation on FreeBSD

渡邊, 尊紀

(Degree)

博士 (学術)

(Date of Degree)

2002-03-31

(Resource Type)

doctoral thesis

(Report Number)

甲2611

(URL)

<https://hdl.handle.net/20.500.14094/D1002611>

※ 当コンテンツは神戸大学の学術成果です。無断複製・不正使用等を禁じます。著作権法で認められている範囲内で、適切にご利用ください。



【239】

氏名・(本籍) 渡邊 尊紀 (静岡県)

博士の専攻分野の名称 博士 (学術)

学位記番号 博い第412号

学位授与の要件 学位規則第4条第1項該当

学位授与の日付 平成14年3月31日

【学位論文題目】

**The interface between Firmware and Operating System
:ACPI implementation on FreeBSD**

**(Firmware と Operation System の間のインターフェース
: Free BSD における A C P I の実装)**

審査委員

主査 教授 郡司 幸夫

教授 角田 譲

教授 宮田 隆夫

ACPI は最近のいわゆる PC だけでなく、Intel の 64 ビット CPU を使用したアーキテクチャでも採用された、電源管理とデバイスの構成をファームウェアから通知するための方法であり、ファームウェアによって書き込まれた、AML と呼ばれる一種の中間言語を含んだメモリブロックを読み込み、カーネル内のインタプリタで行う物である。これによって、柔軟なハードウェアの記述やきめの細かい電源管理、そして、将来的に拡張の余地を、ベンダ独自のコードに頼らずとることが出来る。また、ファームウェアにバグがあった場合はファームウェアのアップデートを行う必要があるが、ファームウェアはブートを行うと言う役割があるため、アップデートに失敗するとソフトウェアの不具合にもかかわらずハードウェアの故障と代わらない損傷となる危険を伴う。ところが、それが中間コードのバグであればファームウェアから渡される中間コードを無視し、ブートルードから使うべき中間コードを OS に対して送り込むことによって、修正を行う事が可能である。APM や PnPBIOS は動的に生成されるイベントを扱う事ができ、特にモバイル環境の APM では電池容量の検出等不可欠であるが、割り込みを生成する事が出来ないため、定期的に BIOS 呼び出しを行わなければならない。これは性能に影響する可能性がある。

ACPI はオンボードデバイスを列挙するのに使われる PnPBIOS や、電源管理メカニズムである APM や PCI の特に割り込みの設定を行う PCIBIOS を置き換えより柔軟にする事を意図している。これは最近の PC には殆ど全てに ACPI が組み込まれており、ACPI の実装はこれからの PC の上で Free な OS が実用的に使い続ける事が出来るためには必要不可欠である。特にノートパソコンのような環境では APM の実装の負担が重いためか、ACPI が出来た頃から APM の実装が不十分なノートパソコンが多くなり、APM の時代に比べて明らかに利便性が低下した。ACPI はデバイス認識の点では PnPBIOS の概念を引き継いでいて、ハードウェア ID や、リソース定義のバイトストリームなど似ている点も多い。だが、ACPI の名前空間そのものの柔軟性を利用してより理解しやすい様に拡張されている。また、温度管理は昔から殆どユーザーに見えない形で行われていたが、ACPI に於いては、これを OS 側で管理するようになった。これは負担でもあるが同時にユーザーが介入できる余地を与える事でもある。

ACPI と似て OS 側から高級な概念を扱ってデバイスの認識等を行う物に Power Macintosh や、Sun Spare WS 等で使われている OpenFirm と言われる物がある。これは OS 側が Forth のインタプリタにアクセスするものであるが、ACPI とは決定的な違いがある。それは、インタプリタの実行されるコンテキストが、ファームウェアことでむしろ ACPI とはある意味逆のアプローチと言える。ACPI は OS のカーネルに近い部分で中間コードインタプリタを実行する必要があるため、実装の負担が非常に大きい。そのため、最近まで Free な OS では実装されていなかったが、Intel が ACPI-CA と呼ばれる実装を公開し、Linux や FreeBSD にはその実装がマージされた。これを FreeBSD に独立した実装をし、

また Intel による ACPICA 実装の移植の一部を行った事に付いて特に FreeBSD のドライバ認識メカニズムの解説を行い、現状の実装の問題点とその解決策の実装に付いて考察する。メモリの内容だけが残っていて CPU が完全に止まった状態から、復帰する過程に付いて記述する。この時には CPU は自分がどういう状態だったか忘れていたため、Intel CPU の初期化に付いてのある程度の知識が必要とされるため、それに付いても解説する。そして、次に実装すべき部分を考察する。

氏名	渡辺尊紀		
論文題目	The interface between firmware and operating system: ACPI implementation on Free BSD		
審査委員	区分	職名	氏名
	主査	教授	郡司 幸夫
	副査	教授	角田 謙
	副査	教授	宮田 隆夫
	副査		

印

要 旨

脳科学において脳は巨大な計算機と想定される。しかし脳は、一般の計算機が有していないだろう意識を有している。ここで注目すべきは、脳という計算機の特異性である。一般の計算機と異なり、脳という計算機は、計算の実行環境と計算機それ自体を兼務している。計算実行環境は、決して閉じていない。それは計算機使用者である人間さえ含むため、どこでその有効な範囲を切断すればよいか決定できない。すなわち、意識を脳なる計算機から解読する際の焦点は、脳は真に計算機なのか、という問題に答えることではなく、計算実行環境を含んだ計算過程をどう理論化するかという点に求められる。

計算実行環境を含む計算過程という問題は、一般の計算機においても計算概念を変えることで問題とすることができる。一般の計算機でさえ、電流を流さなければ計算は実行できない。電流を込みにして計算過程を定義しようとする、送電線や発電所、果ては水力発電を実現する谷川まで計算機に含めるのか、といった問題さえ生じる。このような問題は、計算実行部とそれを実現する環境とを区別し、両者のインターフェースを定義して無限の計算-実行環境階層を構成する形で議論可能である。計算機科学で、このような定式化はリフレクションと呼ばれている。申請者は、オペレーティングシステムの実行とその実行環境であるファームウェアのインターフェースを設計・実装し、計算の実行を含む計算概念の意味を具体的に展開した。

計算機が計算実行環境を自ら創る、というとき、ここには現実の世界から、絶えず計算を実行するための意味論(物理的実行環境)を局所的に構築し、その限りで計算を遂行するシステムがみとれる。実行環境は予め設定されているのではなく、その都度構築される。実行環境の生成過程が実行環境に潜在する。ここに意識への科学的アプローチに対する一つの方法が潜在している。意識と脳=計算機とのインターフェースを射程に捉えようと、計算を現象として丸ごと捉えようという試みが、計算機科学の中でも進行している。その一人が、プログラムの中にプログラムを構築しそのインターフェースであるリフレクション概念を提案し、言語として実装したスミスである。スミスはその後志向性を基礎に置いた野生の計算という概念へと計算概念を拡張しようとしている。申請者の方向性はスミスに親和性があり、計算実行と実行環境の関係を機械的枠組みとして展開する具

氏名	渡辺尊紀
<p>体例をACPIに求めた。</p> <p>ACPIは、電源管理とデバイスの構成を、計算機起動部(実行環境)であるファームウェアから通知する方法である。それは、最近のいわゆるPCだけではなく、Intelの64ビットCPUを搭載したアーキテクチャーでも採用可能で、かなりの一般性・融通性を持っている。ACPIは、AMLと呼ばれる一種の中間言語を含んだメモリブロックを読み込み、カーネル内のインタプリタで実行する。これによって、柔軟なハードウェアの記述や、きめ細かい電源管理を、ベンダ独自のコードに頼らず可能となる。申請者は、ACPIをFreeBSDに独立に実装した。とくにIntelによるオペレーティングシステム独立な実装の移植を行った。そしてFreeBSDのドライバ認識メカニズムおよび、メモリの内容だけ残っていてCPUが完全に止まった状態から、復帰する過程について詳述している。申請者の仕事はACPIの実装という点で既に重要な仕事であるが、この枠組みは単にACPIに留まらず、計算と計算実行環境が非分離な生命系を理解するにあたって極めて有力な概念になると考えられる。</p> <p>またACPIと同様、OS側からデバイス認識を行うものに、OpenFirmと呼ばれるものがある。ACPIが電源管理とデバイスの構成をカーネル内のインタプリタで行うのに対し、OpenFirmはファームウェアで実行する。したがって、カーネル・ファームウェア図式に関して、両者は逆のアプローチとなっている。申請者はOpenFirmもFreeBSDに独立に実装し、この際のドライバ認識メカニズムについて考察している。とくにOpenFirmの場合、メモリの内容だけ残ってCPUが完全に止まった場合、CPUは完全に自分の状態を忘れてしまう。この場合ある程度CPU初期化についての知識が要請され、申請者はどの程度の知識が必要かも詳述している。</p> <p>電源を管理しながら計算資源の配分を管理するという問題は、代謝を実現しその上で様々な機能(計算)を実行する生命系に、普遍的に見出される問題である。抽象的な計算過程と異なり、計算過程の物理的意味を考えると、計算によって生じる熱などある種の剰余が絶えず問題となる。電源管理をしながら計算資源を配分する過程とは、第一に、計算実行以前には予想しがたい物理的剰余を実行以後に考慮し続ける過程であり、第二にそれらが並列的に進行することから以前・以後を単純に同期させて分離できない過程である。したがってこの形式化には底がない。申請者はこの底のなさを、ユーザーによる介入によってある種の解決を試みようとしている。それは、計算実行環境の開放性に関する一つの表現であり、解決でもあり、計算過程とその実行環境が分離しがたいシステム一般に対する重要な知見を示唆している。</p> <p>本研究は計算とその実行環境の相互作用について、電源および計算資源の管理に注目し、ACPIの実装とその意味において研究したものであり、実行環境を含む計算過程について重要な知見を得たものとして価値ある集積であると認める。</p> <p>よって学位申請者の渡辺尊紀は、博士(学術)の学位を得る資格があると認める。</p>	