アプリケーションプログラム主導の省電力制御

1S－3

古市 実裕　相原 達
日本アイ・ビー・エム (株) 東京基礎研究所

1 はじめに

ノートパソコンの省電力制御は，OS や BIOS などシステム主導であり，実行中のタスクの性質に応じて動的に制御方法を切替えることは困難である．本研究では，個々のアプリケーションが，タスクの性質に応じて動的に省電力パラメータを決定し，直接電力制御に参加する手法とその効果について考察する．

2 省電力制御の現状と問題点

ノートパソコンの多くは，一定時間キー入力がないと LCD を消したり，OS がアイドル状態を検知すると CPU クロックを停止する機能 [1] を持つ．また，CPU クロック周波数 (CPU 速度) を定常的に下げて消費電力を抑制する方法もある．さらに，OS による統一的な電力管理を行なう ACPI[2] も提案されている．

OS や BIOS は，NVRAM やファイルに記憶された設定に基づき，これらの省電力機能を制御する．設定の変更のために特別なツールが提供されている場合が多く，ユーザはタイマ設定値や CPU 速度などを変更できる．

しかし，実際の使用場面では，ユーザの作業内容は一様ではなく，アプリケーションの種類が変われば，適切な省電力制御方法も変わる．また，同じアプリケーション内でも，作業内容によって最適な制御方法が異なる．固定的な省電力制御では，タスクの性質やユーザの作業内容に応じて柔軟に制御方法を切替えられない．また，アプリケーションを切替えるたびにユーザが専用のツールを用いて設定を変更するのも現実的ではない．多くのユーザは，最も差障りのないデフォルトの設定のままで使い続ける．

3 新たな省電力制御方法の提案

3.1 理想的な制御方法

理想的には，実際に使用するデバイスにのみ電力供給を行ない，使用しないデバイスは直ちに停止する．作業内容の変化に応じて電力供給を動的に制御し，パフォーマンスやユーザビリティをできるだけ犠牲にしない方法が望ましい．

このような制御には，その時点のタスクが要求するデバイスの種類と必要なパフォーマンスを把握する必要がある．キー入力状況などのイベントの監視だけでは情報不足で，理想的な制御は困難である．

3.2 アプリケーション主導型制御の利点

アプリケーションは，ユーザの作業内容や使用するデバイスの情報を持ち，今後のタスクの動向をある程度予測できるので，省電力機構の最適な制御方法を決めるのに適している．

動作に不必要なデバイスは積極的に停止し，CPU のように数段階のパフォーマンスを提供するデバイスに対しては，過剰なパフォーマンスを供給しないように必要最低限の設定を選択する．

3.3 実現方法

まず，アプリケーションごとに性質を反映した省電力設定を定め，アプリケーションが切り替わるたびに設定を変更することで，アプリケーション主導の制御が実現できる．アプリケーション自身や OS，あるいは専用の管理プロセスが，その都度，NVRAM や設定ファイルの制御パラメータを変更する．

さらに同一アプリケーション内部にも複数の異なる設定を用意し，スレッドが切替わるたびに各々適切な設定に変更すると，より緻密な制御が可能になる．この場合，スレッドの切替え時期を把握しているアプリケーション自身が自発的に設定を変更する．

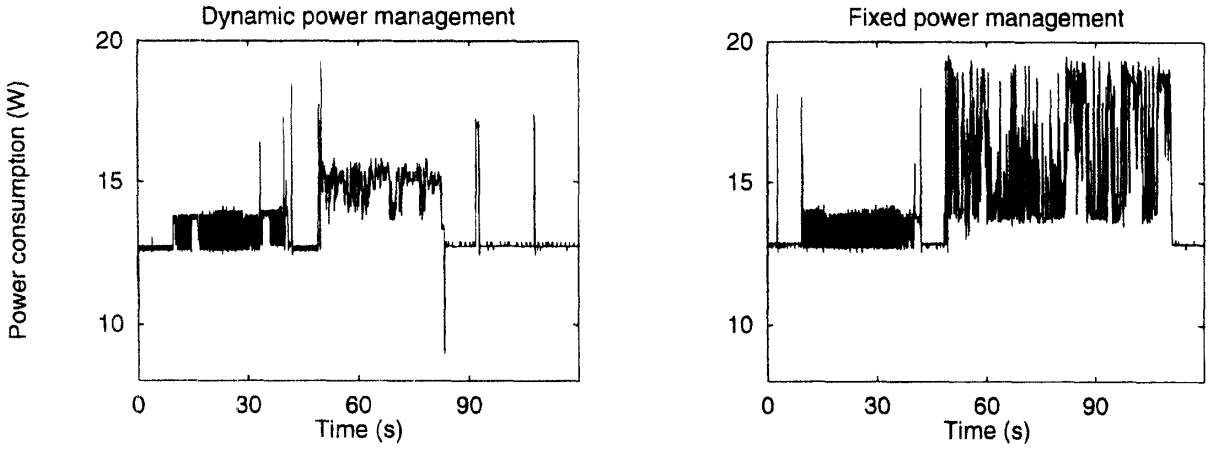

図 1: 本手法と固定的な設定との比較．ワープロではCPU 速度を下げ，コンパイラでは上げることで，消費電力量を減らす．

4 具体例

例として，ワープロやエディタでプログラムを編集し，その後コンパイルする一連の作業を考える．各アプリケーションごとに適切な省電力設定を考察し，簡単な実験で本手法の省電力効果を確認する．

4.1 省電力制御パラメータ

表1に，ワープロとコンパイラの各々の省電力制御パラメータを記す．テキスト入力のCPU 負荷は軽いため，ワープロではCPU 速度を最低にする．コンパイラでは，ユーザインタラクションの必要がないのでLCD を消し，計算を短時間に処理するため，CPU 速度を最高に設定する．

表 1: アプリケーションごとの省電力制御設定値の例

	ワープロ	コンパイラ
CPU 速度	12.5%	100%
LCD 輝度	50%	0
サスペンドタイマ	30s	-

4.2 省電力効果の比較と考察

本手法の省電力効果を評価するために，アプリケーションによらず低消費電力状態に固定 (CPU 速度 12.5%，LCD 輝度 50%) した場合と比較する実験を行なった．実測にはIBM ThinkPad 760，アプリケーションには，エディタ機能とコンパイラ機能を兼ね備えたMicrosoft Visual C++ 4.0 Developer Studio を用いた．実験ではマニュアル操作で省電力設定を変更した．

図1に，それぞれの場合の消費電力の推移を示す．

プログラム編集時には，どちらもCPU 速度を最低レベルまで下げるため消費電力に差はない．コンパイル時は，CPU 速度を最大としたが，LCD を消灯したため平均消費電力には大きな違いがない．しかし，計算時間が短縮されるので，作業全体の消費電力量は少なくなる．

5 まとめ

本研究では，アプリケーションが省電力制御に参加することで，柔軟で効果的な制御が可能であることを示した．ただし，実用性を高めるためにはいくつかの課題が残されている．

例で示したように実質上シングルタスクの場合は実現が容易であるが，マルチタスク環境では，複数タスクによる異なるデバイスの要求を調停する必要がある．また，きめ細かい制御をするためには，アプリケーションとOS・BIOS との間に統一的なインタフェースを定義し，普及させる必要がある．

参考文献

[1] Intel and Microsoft. *Advanced Power Management BIOS Interface Specification 1.2*, 1996.

[2] Intel,Microsoft, and Toshiba. *Advanced Configuration and Power Interface Specification 1.0*, 1996.