社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2009－SLDM－139 (5)
2009－ＥＭＢ－ 12 (5)
2009／3／5

OSから解析可能な無線通信端末の消費電力モデルとその生成手法

石原 亨[1] 奥平 拓見[2] 久住 憲嗣[1] 神山 剛[3] 関根 和寿[3] 片桐 雅二[3]

[1]九州大学システムＬＳＩ研究センター 〒814-0001 福岡県福岡市早良区百道浜 3-8-33
[2]九州大学工学部電気情報工学科 〒819-0395 福岡県福岡市西区元岡 744
[3]株式会社 NTT ドコモ先進技術研究所 〒239-8536 神奈川県横須賀市光の丘 3-5
E-mail: [1]ishihara@slrc.kyushu-u.ac.jp, [2]okuhira@c.csce.kyushu-u.ac.jp, [1]nel@slrc.kyushu-u.ac.jp,
[3]{kamiyamat,sekineka,katagirim}@nttdocomo.co.jp

あらまし 本稿では、携帯電話等の無線通信端末の消費電力モデルとその生成手法を提案する。従来の消費電力モデルの多くは、論理ゲートや演算モジュールなどのハードウェアモジュールの稼働率をモデルパラメータとして使用し、稼働率の計算にはハードウェアシミュレータを用いていた。ハードウェアシミュレータは計算負荷が大きいため、上述のような方法を用いて携帯端末の消費電力を稼働時に解析することは見積もり時間や見積もりに要する消費電力の観点から現実的ではない。また、既存手法の多くは CPU のみを消費電力解析の対象としており、携帯端末全体の消費電力解析には対応していない。提案する消費電力モデルは、無線通信端末全体の消費電力を、演算量、無線送受信量、ストレージのデータ書き込み・消去量などのパラメータを用いて低負荷で精度良く見積もることを可能にする。従って提案する消費電力モデルを使えば、リアルタイムにアプリケーションプログラムの消費電力を解析可能である。Nokia 社の無線通信端末 N810 とその上で動作するいくつかのベンチマークプログラムを用いた消費電力の見積り評価により、実測値に対するモデル誤差が平均 6%であることを確認した。

キーワード 消費電力、携帯電話、モデリング、無線通信

An OS-Analyzable Power Consumption Model and Its Generation Technique for Wireless Communication Devices

Tohru ISHIHARA[1] Takumi OKUHIRA[2] Kenji HISAZUMI[1] Takeshi KAMIYAMA[3] Kazuhisa SEKINE[3] and Masaji KATAGIRI[3]

[1]System LSI Research Center, Kyushu Univ. 3-8-33 Momochihama, Sawara-ku, Fukuoka, 814-0001 Japan
[2]Dept. of EE and CS, School of Eng., Kyushu Univ. 744 Motooka, Nishi-ku, Fukuoka, 819-0395 Japan
[3]Research Laboratories, NTT Docomo, 3-5, Hikarinooka, Yokosuka, Kanagawa, 239-8536 Japan
E-mail: [1]ishihara@slrc.kyushu-u.ac.jp, [2]okuhira@c.csce.kyushu-u.ac.jp, [1]nel@slrc.kyushu-u.ac.jp,
[3]{kamiyamat,sekineka,katagirim}@nttdocomo.co.jp

Abstract This paper proposes a lightweight power consumption model and its generation technique for quickly and accurately estimating the power consumption of wireless communication devices. Many power consumption models for computer power estimation have been proposed before. However, most of them are based on hardware simulation which is very slow and power consuming. Therefore, these approaches cannot be applied for estimating the power consumption of portable battery powered devices where the low power consumption is the most important criterion. Our power estimation method consumes negligible power compared to that of normal application programs run on such portable devices. Experimental results with an N810 terminal developed by Nokia, Inc. demonstrate that the error of our power estimation technique is on an average 6% compared to the measured power results. Once the model has been developed, the power consumption of a target application program can be analyzed in real-time with negligible power consumption.

Keyword Power consumption, Cellar phone, Modeling, Wireless communication

1. はじめに

携帯型情報機器の普及に伴って電子機器の低消費電力化が重要になっている。電子機器の消費電力を削減するためには、設計段階で消費電力を評価する仕組みだけでなくシステムの稼働時にも消費電力を可視化する技術が必要である。今日までに提案されている消

費電力見積もり手法の多くは、論理ゲートや演算回路毎の消費電力を積算することにより対象とするシステム全体の消費電力を計算する。論理ゲートや演算回路毎の消費電力は、チップ上の面積や形状などの物理情報のみから計算される場合もあるが、多くの手法は上記の物理情報とハードウェアシミュレーションによる稼働率情報に基づいて計算する。これらの見積もり手法は主に集積回路の設計過程で使用される。回路設計者は上記の消費電力見積もり手法を用いて対象とする回路の評価を行い、その評価結果に基づいて設計改善を行う。しかし、上記の消費電力見積もり技術は、論理ゲートなど信号変化の回数を見積もるためのハードウェアシミュレーションを伴うため、計算負荷が大きく、上記の技術を用いてプロセッサ上で実行されるアプリケーションプログラムの消費電力を見積もることは現実的ではない。例えば、論理ゲート毎の消費電力に基づいてプロセッサの消費電力を計算するゲートレベルの電力見積もり手法は、プロセッサ上で実行される数千ステップ分のソフトウェアの平均消費電力見積もりに約1時間の計算を必要とする[1]。

本稿では、アプリケーションプログラムが消費する電力を小さい計算負荷で高速かつ正確に見積もるための消費電力モデルとその生成手法を提案する。当該消費電力モデルを使用することにより、実時間でアプリケーションプログラムの消費電力を計算することができる。消費電力の見積もりに必要な計算量が非常に小さいため、携帯電話などのバッテリ駆動機器上で消費電力見積もり計算を行っても、バッテリ残量に与える影響は非常に小さい。同様の消費電力見積もり手法として、システム上の電力計測回路を用いた方法が考えられる。電力計測回路を用いた手法も高速かつ低負荷でアプリケーションプログラムの消費電力を見積もることができる。従って、電力計測回路を搭載した携帯端末では、端末利用時にアプリケーションプログラムの電力を実時間で観測可能である。しかし、電力測定回路ではソフトウェアの基本機能毎の消費電力やプロセス毎の消費電力を個別に見積もることはできない。従って、電力測定回路はソフトウェアの消費電力の内訳を解析する目的では利用できない。また、現状の携帯機器では電力測定回路の搭載は一般的ではなく、電力の測定値をソフトウェアから効率よく吸い上げる仕組みも確立されていない。本稿で提案する消費電力見積もり手法は、OSから観測可能なソフトウェアの基本機能やプロセス毎に消費電力を解析可能である。

本稿の構成は次の通りである。2章で関連研究を紹介し、3章で消費電力モデルとその生成手法を説明する。4章では、消費電力モデル生成と電力見積もり手法を適用した事例を紹介し、5章で本稿をまとめる。

2. 関連研究

2.1. 電力見積もり技術

コンピュータシステムに対する消費電力の見積もり技術は大きく分類すると以下の通りである。

(1) オフラインシミュレーションに基づく技術
　(ア) ハードウェアシミュレーション
　(イ) ソフトウェアシミュレーション
(2) オンラインモニタに基づく技術
　(ア) 電力測定回路を用いた技術
　(イ) パフォーマンスカウンタを用いた技術

SimplePower[2]や Wattch[3]は、SimpleScalar と呼ばれる高性能マイクロプロセッサの消費電力をハードウェアシミュレーションに基づいて計算する。マイクロプロセッサのアーキテクチャ改善を目的とした消費電力見積もりツールである。一方、マイクロプロセッサ上で実行されるアプリケーションプログラムの消費電力を命令セットシミュレーションまたはネイティブコードシミュレーションに基づいて算出する手法も数多く提案されている[1,4-9]。これらの手法は何れも、ハードウェアまたはソフトウェアの設計時に利用することを想定しているため、計算負荷が大きく、システム稼働時のオンライン電力解析の目的では利用が難しい。

システムの稼働時に対象システムの消費電力を計測する方法として、オンチップ温度センサや電力計測回路を用いた手法が考えられる[10]。しかし、上述したように電力計測回路は、システム全体の消費電力を測定するのみで、ハードウェアやソフトウェアの機能ブロックごとの電力内訳を解析することはできない。また、複数のチップに個別のセンサを搭載した場合、測定情報を OS などのソフトウェアが回収するオーバーヘッドが大きくなり現実的ではない。ハードウェアやソフトウェアの機能ブロックごとに消費電力を解析する手法としてパフォーマンスカウンタを用いた手法が数多く提案されている[11-18]。パフォーマンスカウンタとはプロセッサのイベントを計測するカウンタで、典型的には、キャッシュヒット回数・ミス回数、バス転送回数、分岐予測ミス回数、実行命令数などを個別に計測可能である[11]。パフォーマンスカウンタの値はソフトウェアから容易に取得可能であるが、これまでに提案されている手法はマイクロプロセッサと主記憶の消費電力解析を目的としており、無線通信の電力やストレージの電力は解析できない。

ソフトウェアの各プロセスが消費している CPU 時間を UNIX ベースの OS 上で解析するコマンドとして top コマンドが知られている。インテル社は top コマンドの拡張として PowerTop コマンドを公開している

[19]。PowerTop コマンドは稼働中のプロセスの消費する電力を可視化する。割り込みの発生回数や CPU のアイドル時間などの情報から消費電力を計算する。現状ではインテル社のプロセッサ上でのみ動作し、他のプロセッサには対応していない。また、プロセッサ以外の消費電力を解析することはできない。

2.2. 消費電力モデルとその生成技術

2.1.節で紹介した手法はいずれも消費電力を見積もる手法であり、モデリングに関する議論は行われていない。本節では、消費電力モデルの生成手法を議論している既存手法をいくつか紹介する。

ハードウェアモジュールの消費電力を事前にルックアップテーブルに保持しておき、モジュール毎の消費電力を積算することにより対象システムの消費電力を計算する手法が対案されている[20-22]。 ルックアップテーブルの引数は互いに独立なパラメータが使われる。テーブルルックアップモデル生成は、ハードウェアモジュール毎の引数の値に応じた消費電力を計測することにより完了する。従ってモデリング自体は非常に簡単である。しかし、既存手法の多くはハードウェアモジュールの稼働率を引数としているため、消費電力の計算にはハードウェアシミュレーションを必要とする。従って、システム稼働時の消費電力解析には向かない。ソフトウェアの関数の呼び出し回数を引数とする消費電力モデルも提案されているが[8, 9]、関数が消費する電力はその呼び出し回数だけでは精度良く計算できないため、精度改善のためにはキャッシュミス回数などの新たな引数が必要となる。キャッシュミス回数は、関数の呼び出し回数に依存するため、ルックアップテーブル作成の際にそれぞれのパラメータ値を独立に制御することが難しい。文献[8, 9]の手法は、複数の依存関係のある引数が消費電力に与える影響を解析する手法として重回帰分析法を用いている。

重回帰分析を用いたモデリング手法の多くは、測定した消費電力値を基準値として、モデルフィッティングによりパラメータ係数を求める。一般的には、モデルフィッティングには最小二乗法が用いられる。モデル式には式(1)に示す線形式を用いた手法が数多く提案されている[1-9, 11-18]。

$$P_{esimated} = c_0 + \sum_{i=1}^{N} c_i \cdot P_i \qquad (1)$$

ここで、$P_{estimated}$、P_i、c_i、c_0、および N は、それぞれ消費電力の見積もり値、線形モデルのパラメータ、各パラメータの係数、定数項およびパラメータの数を表す。最小二乗法を用いた重回帰分析では、式(2)に示す ε_j の二乗和を最小化する c_i の集合を決定する。ここで、PM_j、P_{ij}、ε_j は、それぞれサンプル j の消費電力の測定値（サンプル j の基準値）、サンプル j におけるパラメータ i の値、サンプル j におけるモデル誤差を表す。

$$PM_j = c_0 + \sum_{i=1}^{N} (c_i \cdot P_{ij}) + \varepsilon_j \qquad (2)$$

重回帰分析ではパラメータ間の相関が強いと有効な重回帰式が得られないことが知られている。相関の強いパラメータが 2 つ以上存在すると、どちらのパラメータがどの程度消費電力に影響を与えるかが解析できないためである。この性質は多重共線性と呼ばれる[25]。多重共線性が観測された場合には、片方のパラメータを除去することによりこの問題を解消できる[25]。しかし、除去したパラメータが、対象とするシステムの消費電力に対して大きな影響力を持つ場合は、このパラメータを除去することによりモデル精度の低下を招くことになる。効率よくモデルパラメータを取捨選択する方法としてステップワイズ法が知られているが、重回帰分析に使用しているサンプルに統計的偏りが存在する場合には多重共線性の問題を解消できない。

本稿では多重共線性の問題が発生しないようなテストベンチを意図的に生成することにより、影響力の大きい少数のパラメータを用いて精度良く無線通信端末の消費電力をモデル化する手法を提案する。

3. 消費電力モデルの生成手法

3.1. 消費電力モデル

本稿で提案する消費電力モデルは重回帰分析によりパラメータ係数を決定することを前提としている。一般に、次の 4 つの要素がモデルの精度を決定する。

① モデルパラメータ
② モデル式
③ パラメータ係数の決定方法
④ 重回帰分析に使用するサンプル

本研究では、多くの既存研究同様にモデル式には線形式を用い、パラメータ係数の算出には最小二乗法を用いる。モデルパラメータには次の 4 種類を使用する。

A) CPU の負荷率
B) 無線による単位時間当たりのデータ受信量
C) 無線による単位時間当たりのデータ送信量
D) 単位時間当たりのディスクへのデータ書き込み量＋データ消去量

上記のパラメータは、消費電力に対して線形で、かつ大きな影響を与えることが予想できることを理由に選択した。上記以外のパラメータも無線通信端末の消費電力に影響を与えるが、本研究では上記以外のパラメータは定数項として扱う。

3.2. テストベンチとサンプル取得

3.1.節で述べた、モデル精度を決定する要素の④に関しては、CPU の演算負荷を調整可能なテストベンチを作成し、他のパラメータ値と独立に設定することで

精度の良いモデルを作成する。4 つのパラメータ値の相関係数に応じて演算負荷を変更する。具体的には次の 4 種類のテストベンチを作成した。

(1) 端末の演算負荷を調整可能なテストベンチ
(2) サーバから端末へ無線通信でデータを送信するテストベンチ
(3) サーバが端末から無線通信でデータを受信するテストベンチ
(4) 端末のディスクへのデータ書き込みとデータ消去を行うテストベンチ

CPU の演算負荷を調整可能にするために、テストベンチ実行中に可変周期で sleep コマンドを実行する（図 1 参照）。

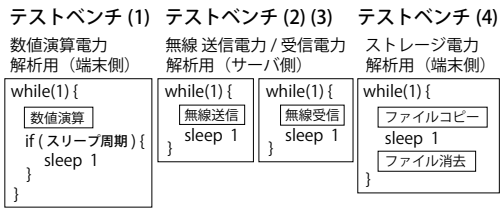

図 1　重回帰分析用テストベンチ

CPU の負荷が消費電力に与える影響を解析する際には、携帯端末上でテストベンチを何も実行しない場合と上記(1)のテストベンチを携帯端末上で単独実行した場合の消費電力値とパラメータ値を取得する。無線通信によるデータ送受信量が消費電力に与える影響をモデル化する際には、図 2 に示す通り、上記の(2)または(3)のテストベンチをサーバ上で実行し、同時に(1)のテストベンチを携帯端末上で実行する。ディスクへのデータ書き込みおよびデータ消去の影響を解析する際には、上記の(4)と(1)のテストベンチを同時に携帯端末上で実行する。データの送信量、受信量、およびディスクの書き込み・消去量、は独立に制御することが容易である。しかし、これらの値は CPU の負荷率と強い相関をもつ場合がある。提案手法はテストベンチ(1)のスリープ周期を調整することにより、CPU 負荷率と他のパラメータの相関を低減することを可能にする。

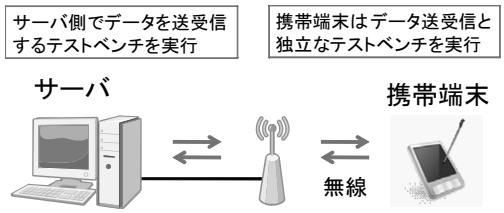

図 2　消費電力モデルの生成環境

重回帰分析用のサンプルを取得するために、具体的には下記の 5 種類のテストシナリオに基づいて消費電力測定とパラメータ値の取得を行う。

(a) テストベンチを何も実行しない。
(b) テストベンチ(1)のスリープ周期を最大（CPU 負荷最大）にし、携帯端末上で実行する。
(c) テストベンチ(2)をサーバ上で実行し、同時にテストベンチ(1)のスリープ周期を最大（CPU 負荷最大）にし、携帯端末上で実行する。
(d) テストベンチ(3)をサーバ上で実行し、同時にテストベンチ(1)のスリープ周期を最大（CPU 負荷最大）にし、携帯端末上で実行する。
(e) テストベンチ (4)を携帯端末上で実行し、同時にテストベンチ(1)のスリープ周期を最大（CPU 負荷最大）にし、携帯端末上で実行する。
(f) テストベンチ(1)のスリープ周期を“**最小**”（CPU 負荷最小）にし、携帯端末上で実行する。
(g) テストベンチ(2)をサーバ上で実行し、同時にテストベンチ(1)のスリープ周期を“**最小**”（CPU 負荷最小）にし、携帯端末上で実行する。
(h) テストベンチ(3)をサーバ上で実行し、同時にテストベンチ(1)のスリープ周期を“**最小**”（CPU 負荷最小）にし、携帯端末上で実行する。
(i) テストベンチ (4)を携帯端末上で実行し、同時にテストベンチ(1)のスリープ周期を“**最小**”（CPU 負荷最小）にし、携帯端末上で実行する。

3.1.節で述べた 4 種類のパラメータの値をそれぞれのシナリオに対して計測し、パラメータの相関が強い場合は、相関を減らすようにテストベンチ(1)のスリープ周期を変更する。

3.3. モデル生成フロー

重回帰分析用テストベンチの生成フローを図 3 に示す。

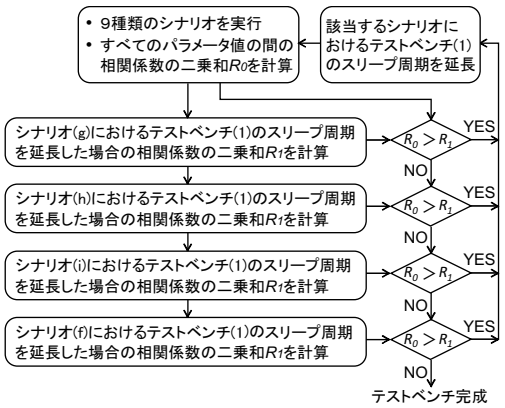

図 3　テストベンチ生成の詳細フロー

パラメータ A)～D)は 3.1.節に示した通りである。テストベンチ(2)、(3)、(4)は個別に実行するため、無線受信量、無線送信量、ディスク書き込み・消去量はそれぞれ独立に制御し易い。しかし、無線通信やディスクアクセスの際には CPU も稼働するため、CPU 負荷率と他のパラメータとの相関は強い。図 3 のフローでは、すべてのパラメータ間の相関係数を計算し、その二乗和が最小になるように CPU 負荷率を変更することによりパラメータの相関を低減する。テストベンチが生成されると、重回帰分析によりモデルを自動生成する。

4. 消費電力評価事例

4.1. 対象システムと評価環境

実験対象の無線通信端末には Nokia 社の N810 を使用した。N810 は、無線 LAN 機能内蔵の携帯型無線端末である。Linux ベースの Internet Tablet OS 2008 を搭載している。CPU には TI 社の OMAP2420 を、ディスクには 256MB と 2GB の NAND フラッシュメモリを使用している。モデルパラメータには、CPU 負荷率、無線受信量、無線送信量、フラッシュディスクへの書き込み・消去量を使用し、これらの値は/proc ファイルから 1 秒間隔で取得する。電力計で計測した消費電力値に、3 章で述べた方法でモデルをフィッティングさせることでパラメータ係数を求める。モデルが生成されると、/proc から取得した上述の 4 種類のパラメータ値を用いて端末の消費電力を見積もることができる。

消費電力モデルの評価には wget と MiBench の 7 種類のベンチマークプログラムを用いた。wget は http または ftp プロトコルでファイルを受信するプログラムである。MiBench からは FFT、ADPCM、SHA、basicmath、JPEG、quicksort、stringsearch を選択した。

4.2. モデルフィッティングの結果

3 章で述べたテストシナリオとフローに従ってモデルを生成した結果、測定値に対する平均誤差が 0.6%のモデルを作成できた。図 4 の折れ線グラフは上記の 9 つのテストシナリオに対する消費電力の測定結果を示している。棒グラフは測定値にフィッティングさせたモデルによる消費電力見積もり値である。線形モデルが測定値に良くフィットしていることが確認できる。

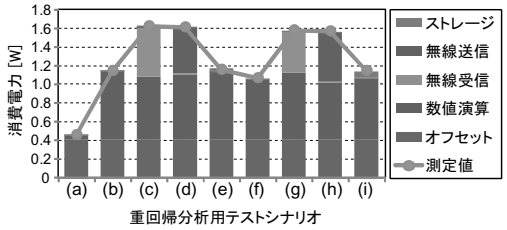

図 4　モデルフィッティングの結果

9 種類のテストシナリオを実行した時のパラメータの相関係数を表 1 に示す。表 1 の結果から、任意の 2 パラメータ間の相関が十分に小さいことが確認できる。提案するテストシナリオとモデル生成フローに従って生成した消費電力モデルでは、多重共線性がモデル精度に与える負の影響は小さいと考えられる。

表 1　パラメータ間の相関係数

パラメータ 1	パラメータ 2	1 と 2 の相関係数
CPU 負荷率	無線受信量	0.213
CPU 負荷率	無線送信量	0.104
CPU 負荷率	ディスク書込量	0.136
無線受信量	無線送信量	-0.236
無線受信量	ディスク書込量	-0.260
無線送信量	ディスク書込量	-0.264

4.3. 消費電力モデルの評価実験結果

MiBench を実行した際の消費電力の見積もり結果を図 5 に示す。ベンチマーク名の後に付けた L と S はそれぞれ入力に用いたデータのサイズ（Large と Small）を表す。提案する消費電力モデルによる見積もり値と測定値との誤差は平均 4.9%で最大 24%であった。ADPCM を除くベンチマークでは誤差が 5%未満と非常に精度が良い。

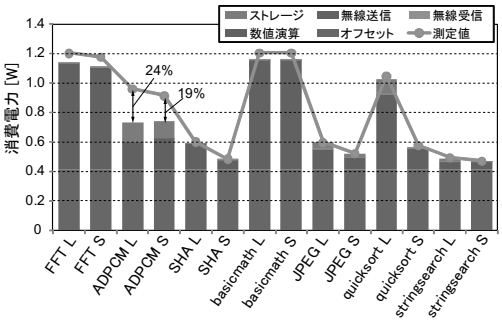

図 5　MiBench の消費電力見積もり結果

wget で国内外のサイトからファイルを取得する際の消費電力の測定値と見積もり結果を図 6 に示した。米国、ベトナム、日本、ベルギーの http サイトからファイルを取得する実験を行った。ディレクトリごとファイルを取得する場合とファイル単体を取得する場合の両方の消費電力を評価した。消費電力モデルによる見積もり値と測定値の誤差は平均 7.4%、最大 16.9%であった。サイズの大きい一つのファイルを取得するケースで誤差が特に大きい。原因は解析中である。

消費電力解析で消費する電力を評価した結果、パラメータの値を/proc から 1 秒おきに取得する処理で消費する電力は 48mW であった。端末が何もしていない時の消費電力の約 12%と非常に小さいことを確認した。

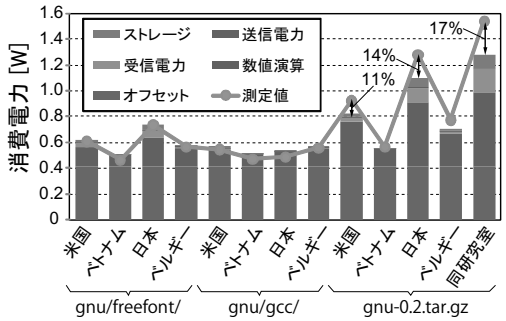

図 6　wget の消費電力見積もり結果

5. おわりに

本稿では Nokia 社の無線通信端末 N810 の消費電力を演算量、無線通信量、ストレージのデータ書き込み・消去量を用いて見積もる手法と消費電力見積もりのためのモデルを生成する手法を提案した。今後は、誤差の原因究明、他の端末での評価を行う。ディスプレイの電力や有線の通信機能、LAN 以外の無線機能および DSP や専用チップなど、今回は対象としなかった機能の消費電力モデルを生成することも今後の課題である。

謝　辞

本研究を進めるに当たりご協力頂いた名古屋大学の横山哲郎氏、曾剛氏、冨山宏之准教授、高田広章教授に感謝します。本研究の一部は科学研究費補助金・若手研究 B(20700049)による。

文　献

[1] D. Lee, T. Ishihara, M. Muroyama, H. Yasuura, and F. Fallah, “An Energy Characterization Framework for Software-Based Embedded Systems,” in Proc. of the 2006 IEEE/ACM/IFIP Workshop on ESTIMedia, pp.59-64, Oct. 2006.

[2] W. Ye, N. Vijaykrishnan, M. Kandemir and M.J. Irwin, “The Design and Use of SimplePower: A Cycle-Accurate Energy Estimation Tool”, Proc. of 37th DAC, pp.340-345, June 2000.

[3] D. Brooks, V. Tiwari, and M. Matonosi, “Wattch: A Framework for Architectural-Level Power Analysis and Optimization”, in Proc. of ISCA, pp.83-94, June, 2000.

[4] V. Tiwari, S. Malik, and A. Wolfe, “Power Analysis of Embedded Software: A First Step towards Software Power Minimization”, in Proc. of ICCAD, pp.384-390, November 1994.

[5] C. Brandolese, W. Fornaciari, F. Salice, and D. Sciuto, “An Instruction-level Functionality-based Energy Eestimation Model for 32-bit Microprocessors,” in Proc. of DAC, pp.346–351, June 2000.

[6] A. Sama, M. Balakrishnan, and J. F. M. Theeuwen, “Speeding Up Power Estimation of Embedded Software”, in Proc. of ISLPED, pp.191-196, August 2000.

[7] T. Sinha, and A. P. Chandrakasan, “JouleTrack – A Web Based Tool for Software Energy Profiling”, in Proc. of DAC, pp.220-205, June 2001.

[8] G. Qu, N. Kawabe, K. Usami, and M. Potkonjak, “Function-level power estimation methodology for microprocessors,” in Proc. of DAC, pp.810–813, June 2000.

[9] A. Muttreja, A. Raghunathan, S. Ravi, and N. K. Jha, “Hybrid simulation for embedded software energy estimation,” proc. of DAC, pp.23-26, june 2005.

[10] S. Kaxiras and P. Xekalakis, “4T-Decay Sensors: A New Class of Small, Fast, Robust, and Low-Power, Temperature/Leakage Sensors,” in Proc. of ISLPED, pp.108-113, August, 2004.

[11] R. Joseph and M. Martonosi, “Run-Time Power Estimation in High Performance Microprocessors,” in Proc. of ISLPED, pp.135-140, Aug. 2001.

[12] G. Contreras and M. Martonosi, “Power Reduction for Intel XScale® Processors Using Performance Monitoring Unit Events”, in Proc. of ISLPED, pp.221-226, Augst 2005.

[13] W. L. Bircher, M. Valluri, J. Law, and L. K. John, “Runtime Identification of Microprocessor Energy Saving Opportunities”, in Proc. of ISLPED, pp.275-280, Augst 2005.

[14] T. Li and L. K. John, “Run-time Modeling and Estimation of Operating System Power Consumption”, in Proc. of Int’l Conference on Measurements and Modeling of Computer Systems, pp.160-171, June 2003.

[15] W. Wu, L. Jin, J. Yang, P. Liu, and S. X.-D. Tan, “A Systematic Method for Functional Unit Power Estimation in Microprocessors,” in Proc. of DAC, pp.554-557, July 2006.

[16] W. L. Bircher and L. K. John, “Complete System Power Estimation: A Trickle-Down Approach Based on Performance Events,” in Proc. of ISPASS, pp.158-168, April, 2007.

[17] K. Singh, M. Bhadhauria, and S.A. McKee, “Real Time Power Estimation and Thread Scheduling via Performance Counters,” in Proc. of workshop on Design, Architecture and Simulation of Chip Multi-Processors, Nov. 2008.

[18] C. Isci and M. Martonosi, “Runtime Power Monitoring in High-end Processors: Methodology and Empirical Data,” in Proc. of Internatinal Symposium on Microarchitecture, pp.93-104, 2003.

[19] PowerTop by Lesswatts.org, *http://www.lesswatts.org /projects/powertop*

[20] S. Gupta and F. Najm, “Power Macromodeling for High Level Power Estimation,” in Proc. of DAC, pp.365-370, June 1996.

[21] K. T. Do, Y. H. Kim, Y. H. Kim, and J. Y. Choi, “Power Modeling of Synthesizable Soft Macros,” IEICE Trans. on Fundamentals, vol.E87-A, no.12, Dec., 2004.

[22] M. Onouchi, T. Yamada, K. Morioka, I. Mochiduki, and H. Sekine, “A system-level power-estimation methodology based on IP-level modeling, power-level adjustment, and power accumulation,” in Proc. of ASP-DAC, pp.547-550, Jan. 2006.

[23] T. K. Tan, A. Raghunathan, G. Lakshminarayana, and N. K. Jha, “High-level Software Energy Macro-modeling”, in Proc. of DAC, pp.605-610, June 2001.

[24] B. C. Lee and D.M. Brooks, “Accurate and Efficient Regression Modeling for Microarchitectural Performance and Power Prediction,” in Proc. of ASPLOS, pp.185-194, October, 2006.

[25] F. E. Harrell, “Regression Modeling Strategies,” Springer-Verlag, 2001.