

UMLを用いたソフトウェアの消費電力評価シミュレーション

角野友則^{†1} 久住憲嗣^{†2}
中西恒夫^{†3} 福田晃^{†4}

組込みシステム開発における消費電力解析は重要である。今後さらに多様・複雑化するであろう組込みシステムにおけるソフトウェアの低消費電力化の解決策として、ソフトウェアの振る舞いにおける消費電力解析は不可欠である。既存のコンピュータシステムにおける消費電力解析技術には様々な手法が存在する。しかし既存の手法は、ソフトウェアの開発には向かないものが多かった。本論文ではソフトウェア開発に適した消費電力解析手法として、UMLを用いたソフトウェアの消費電力評価シミュレーションを提案する。この手法を用いることで、高速に消費電力を推定することができ、設計レベルにおいて事前検討しやすく、振る舞いの変更にも柔軟に対応できる。評価実験の結果、平均誤差20%で短時間に消費電力を評価出来た。

TOMONORI KADONO,^{†1} KENJI HISAZUMI,^{†2} TSUNEO NAKANISHI^{†3}
and AKIRA FUKUDA^{†4}

It is an important challenge to analyze energy consumptions of embedded systems taking into software behavior in development of large and complex software. Existing energy consumption analysis methods for embedded systems is not suitable to the development of software. The paper proposes a novel method to model software structures, behaviors, and resource consumptions using UML(Unified Modeling Language) which can be used in the design stage. It also proposes a simulation method to estimate energy consumptions from the model. We also demonstrate that our method can estimate it in error by less than 20

1. はじめに

組込みシステムの高性能化・高機能化により、消費電力は増大している。それに伴い、低消費エネルギー化の必要性も高まっている。例えばスマートフォン等の携帯端末は、インターネット閲覧、カメラ、マップ、乗換案内、予定表等、様々な用途で使用されるようになってきている。また、携帯端末の消費電力は増大傾向にある。しかし、フィーチャーフォンの連続待ち受け時間が平均550時間であるのにくらべ、スマートフォンの連続待ち受け時間は平均371時間⁸⁾しかなく、長時間駆動が重要な課題となっている。据え置き組込みシステムにおいても消費電力を少なくすることで、待機電力を減少させたり、電源装置を小型化することができ製品の製造コストを抑えることができる。以上のような背景を持つ組込みシステム開発において、消費エネルギー分析技術は重要な要素である。消費エネ

ルギーを分析することで、消費エネルギーを減少させるように努力・改善することができる。複雑化したソフトウェアの消費電力を既存の手法で解析しようとした場合、特定の新しい言語を修得する必要があったり、解析時間が長すぎたり、設計段階において推定することができないといったデメリットが生じる。

本論文においてこれらの課題を解決するために、UML(Unified Modeling Language)¹⁰⁾を用いたソフトウェアの消費電力評価シミュレーションを提案する。この手法を用いることで、設計時に利用できる抽象度で、設計時に推測可能な情報を元にモデルが構築でき、高速でソフトウェアの消費エネルギーを解析することができ、さらに振る舞いごとに解析を行うことができる。

本論文の構成は、以下のとおりである。第2節では、ソフトウェアの消費電力解析技術について、関連研究を例に挙げて説明する、第3節では、本研究において提案する手法についての説明を行う。第4節では、本研究の評価を行う。第5節では、結論および今後の課題について述べる。

^{†1} 九州大学大学院情報科学府情報知能工学専攻

^{†2} 九州大学システム LSI 研究センター

^{†3} 九州大学大学院システム情報科学研究院

^{†4} 九州大学大学院システム情報科学研究院

2. 関連研究

コンピュータシステムの電力解析手法には様々な手法が存在する。本節では、システムの消費電力技術について、コードベース解析として数種類の解析について説明し、その後モデルベースの解析と既存研究の問題点について説明する。

2.1 コードベースの消費電力解析

ソフトウェアシミュレーションを用いて消費電力解析を行うコードベース解析というものがある。コードベースの消費電力解析技術は大きく分けて、命令レベルでの消費電力解析と、関数レベルでの消費電力解析がある。

2.1.1 ハードウェアシミュレーションを用いた消費電力解析¹⁾

まず、SimplePower等のプロセッサベースのシステムにおけるアーキテクチャ改善を目的とした消費電力解析を行うために、ハードウェアシミュレーションを用いて解析する手法がある。手法としては、トレーニングベンチに対してゲートレベルでの電力計算と命令セットシミュレートによるパラメータ抽出を行った後に線形モデルを作成することで、マイクロプロセッサの消費電力を推定している。評価結果としては、平均誤差は3%、最悪でも16%の誤差で消費電力を推定している。

2.1.2 命令レベルでの消費電力解析文献²⁾

文献2)では、解析負荷が大きい等の欠点を持つゲートレベルでの消費電力解析手法に対する新たな手法として、命令レベルでの電力見積もり手法を提案している。Intel社のCISC(Complex Instruction Set Computer)プロセッサ486DX2を用いて、各命令を繰り返し実行することで命令ごとの消費電力を定義し、命令セットシミュレータを用いて命令単位で解析して実行回数を計測し、累積することでソフトウェアの消費電力を見積もることができるというものである。

2.1.3 関数レベルでの消費電力解析³⁾

命令レベルでの解析を関数レベルに拡張したもので、"power data bank"と呼ばれるライブラリに関数実行時の電力消費に関するデータを記載して電力を定義しておく。解析したいプログラムにおいて、プロファイリングツールを用いて関数の実行回数を計測し積算することで消費電力を見積もるといった手法である。評価結果としては、平均誤差3%で消費電力を推定できる。更に、マイクロプロセッサのベンダーは、コアについての詳細を公開することなく、ユーザーに"power data bank"を提供することができる。

2.2 モデルベースの消費電力解析⁶⁾

コードベース解析の、計算負荷が大きい為に解析時間が長く消費電力が大きくなる欠点を改善するために、UMLを用いたモデルベースの解析手法が提案されている。彼らは、解析時間が短く、結果を出力しやすく、専門の知識を修得する必要がない消費電力解析手法を提案した。評価手法のフローは、まず入力であるUML図をGarousi氏の開発したアルゴリズム¹¹⁾を用いてCFG(Control Flow Graph)に変換する。この変換によって、複数のUML図を1つにまとめることでより解析に適した形にしている。変換したCFGに対して、EBU(Energy Behavioral Unit)と呼ばれる電力消費の基本ユニットを探索していく。各EBUに対応する電力消費に関するデータはEnergy libraryと呼ばれるライブラリに格納されているので、Energy libraryを元に消費電力量を累積していくことで、消費電力を推定することができる。この解析手法を用いることで、コードベースの消費電力解析と比較して誤差10%以内を実現した。そして、コードベース解析に比べて解析時間が半分となり、解析結果を表やオブジェクトとして出力でき、結果の反映のさせやすさを実現している。

2.2.1 OSから観測可能な情報に基づくプロセス単位での消費電力解析⁴⁾

同じくモデルベースの解析として、低消費電力であり、短時間でアプリケーションプログラムの消費電力を計算可能で、かつ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な消費電力解析手法がある。この手法において、消費電力モデルにおけるモデルパラメータは以下の4種類を使用している。

- CPUの負荷率
- 無線による単位時間当たりのデータ受信量
- 無線による単位時間当たりのデータ送信量
- 単位時間当たりのディスクへのデータ送信量+データ消費量

モデル式には式1に示す線形式を使用している。

$$P_{estimate} = C_0 + \sum_{i=1}^N C_i * P_i \quad (1)$$

ここで、 $P_{estimate}$ 、 P_i 、 C_i 、 C_0 および N は、それぞれ推定する消費電力量、各パラメータ、各パラメータにおける係数、定数項およびパラメータ数を表す。パラメータ係数は、最小二乗法を用いて求めている。この手法を用いてプロセス毎に消費電力解析を行った結果、平均誤差を4.9%に抑えることができ、解析中の消費電力も、パラメータの値を/procから1秒ごと

に取得する処理で消費する電力は、何もしていない待機状態の時の約 12%である。

2.3 既存研究の問題点

既存の解析手法は 計算負荷が大きいために、ソフトウェアが複雑になればなるほど計算時間も膨大となる。更に、既存のシステムにしか適応できず、設計段階において消費電力を推定することが出来ない。

3. UML を用いた消費エネルギー分析手法

まず、提案手法に対する要求としては、以下の条件が挙げられる。

- 設計時に利用できる抽象度でモデルを構築することができる
- 設計時に推測可能な情報を元にモデルが構築できる
- 高速で消費エネルギーを解析することができる

以上の条件を満たす手法として本節では、UML を用いた消費エネルギー解析手法と消費電力モデルの構築手法について述べる。本研究のターゲットである Android のための消費エネルギー解析手法については第 4 節で述べる。

全体像としては、図 1 のような手順で解析を行う。振る舞いの一連の例であるテストケースに対して、振る舞い図を適用させることで消費電力推定の為の変数の総和となる総リソース消費量が分かる。更に消費電力モデルを適用させることで、振る舞いに対する消費エネルギーが算出される。消費電力モデルについては、3.2 で述べる。

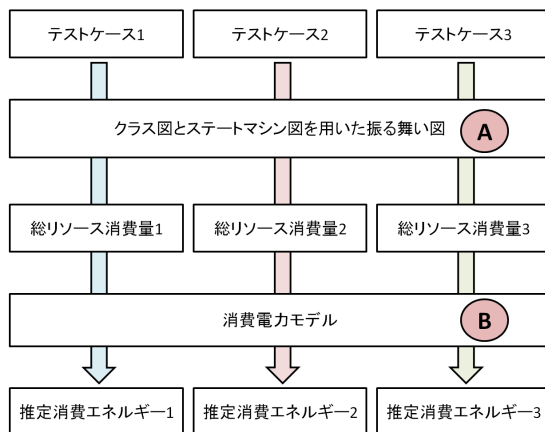


図 1 提案手法の全体像

3.1 リソース消費量を含む振る舞い図の構築

ここでは、図 1 の A に対応する振る舞い図の構築手法について述べる。本論文で提案する手法では、UML

における図であるクラス図とステートマシン図で表現する。クラス図はソフトウェアの構造を、ステートマシン図はクラスの振る舞いを表す。1 つのクラス図に少なくとも 1 つのステートマシン図が所属し、複数クラス間でイベントやデータのやり取りを行うことができる。図 2 に示すように、評価したいソフトウェアの振る舞いをステートマシン図で表現する。

次に、各状態に到達した際に消費するリソース消費量をステートマシン図へ追記する。追記方法は以下のとおりである。

cons(CPU 使用率,Wi-Fi 通信量, データ読み書き量);

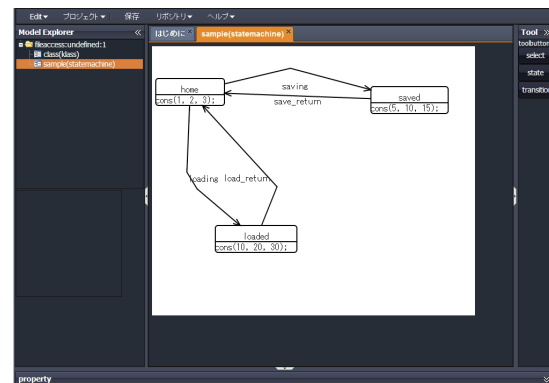


図 2 作成したステートマシン図

3.2 消費電力モデル

ここでは、図 1 における B と対応する消費電力モデルの構築手法について示す。本研究における消費電力モデルとは、対象システムの消費電力を、リソース消費量を引数として算出できるよう数式化したものである。提案手法においては、文献 4) を参考に式 1 に示す線形式を使用する。

次にパラメータの種類について述べる。本提案手法においては、最も消費エネルギーに影響を与えてかつ設計時に推定が容易な以下の三種類をリソース消費量のパラメータとして推定を行う。

- CPU 使用率
- Wi-Fi 通信量
- データ読み書き量

データ読み書き量とは、メモリ間でのデータ読み書き量を表す。パラメータ係数は、CPU 使用率、Wi-Fi 通信量、データ読み書き量それぞれに対してベンチマークプログラムを実行し、最小二乗法で求める。パラメータ係数はソフトウェア設計とは独立に汎用のベンチマークプログラムを用いて求めることができる。

3.3 ステートマシン図の C 言語への変換

clooca の C 言語自動変換機能を用いて、作成したステートマシン図を C 言語に変換する。

3.4 C プロジェクトの作成

シミュレーション用の C プロジェクトを作成する。clooca で作成した C ファイル群と、ランタイムライブラリを 1 つのファイルに統合する。

3.5 テストケースの追加

生成されたプログラムに状態遷移の命令を追加する。追加した命令を用いて、消費電力を推定したい振る舞いをシミュレーションする。測定したい振る舞いの通りに遷移するように、main.cpp ファイル内の main 関数に、以下のように命令文を加える。

```
class.sendEvent(class_EVENT_event) (2)
```

例えば、図 2 において、初期状態である home から、saved に遷移した後に home に戻り、次に loaded に遷移した後に home に戻るといふ振る舞いについて消費電力を計測したい場合、図 3 のように命令文を加えればよい。

```
test a;
a.sendEvent(test_EVENT_loading);
a.sendEvent(test_EVENT_load_return);
a.sendEvent(test_EVENT_saving);
a.sendEvent(test_EVENT_save_return);
```

図 3 追加する命令

3.6 消費電力のシミュレーション

最後に、完成した C プロジェクトを実行することで、振る舞いに対する消費電力が推定される。図 2 の振る舞いを実行した際は、図 4 のような結果となる。

```
consumption = 0
loaded
consumption = 60
home
consumption = 66
saved
consumption = 666
home
consumption = 672
```

図 4 図 4 の推定結果

4. 評価

本節では、提案する手法を用いた消費電力解析を評

価する。本研究においては、Android をターゲットとして解析を行っている。まず、Android における本手法の適応方法について説明し、評価の手順・結果について説明する。

4.1 Android のための消費電力見積もり構築

本研究においては、携帯端末の Android をターゲットとして消費電力解析を行う。解析環境を構築するにあたって、入力となる UML のステートマシン図は、clooca⁹⁾ というツールを用いて作成する。

本節では、まず clooca の説明を述べた後に、Android における提案手法の構築手順について述べる。

4.1.1 clooca について

clooca とは、Web ベースの SaaS 型ドメイン特化モデリングツールで、九州大学や信州大学において教育用として運用されている。ユーザは clooca を用いることで、主に以下の機能を利用することができる。

- ・プログラミング言語と比べて抽象度の高い UML 等のモデリング言語を用いてプログラミングを行うことができる。

- ・モデリング言語をシミュレートできる。

- ・モデリング言語からソースコードを自動生成できる。

4.1.2 消費電力モデルについて

本提案手法において消費電力解析を行うに当たり、入力としてソフトウェアの振る舞いを clooca を用いて記述したステートマシン図を用いる。その際に、リソース消費量となる 3 種類のパラメータの値を記述しなければならない。Android におけるリソース消費量は、以下のように調べることができる。

4.1.3 CPU 使用率

調べたい端末の /proc/pid(固有のプロセス ID)/stat にアクセスすることで、実行中のプロセスについての情報を取得することができる。

このデータにおいて、14 番目のデータ (utime) と 15 番目のデータ (stime) が、それぞれユーザモードとカーネルモードにおける端末起動時からの累積利用時間である。調べたい動作の開始地点と終了地点においてデータを取得し、それぞれの utime と stime の差を δ_u , δ_s とすると、以下の式で CPU 使用率を求めることができる。

$$\text{CPU 使用率} = \frac{\delta_u}{\delta_u + \delta_s} (\%) \quad (3)$$

4.1.4 Wi-Fi 通信量

CPU 使用率と同様にして、調べたい端末の /proc/pid/net/dev にアクセスすることで、実行中のプロセスの通信量についての情報を取得できる。調べたい動作の

開始地点と終了地点において、端末起動時からの累積通信量を取得し差分を出すことで、調べたい動作における Wi-Fi 通信量を求めることができる。

4.1.5 データ読み書き量

本論文においては、入力および出力する文字列に対して String クラスの getbytes メソッドを用いることでデータ読み書き量を取得している。

4.2 評価概要

評価の概要としては、CPU の負荷が大きな振る舞いを行うアプリケーションと、データ読み書きを行うアプリケーションに対して、それぞれ異なる 3 種類のテストシナリオを clooca を用いて推定した値と、電流計を用いて実測した値とを比較した。本評価において使用した機材は携帯端末、電流計と電源装置である。それぞれの詳細を以下に示す。

表 1 機材詳細

携帯端末	Galaxy Nexus
電流計	Agilent 34411A 61/2 Digit Multimeter
電源装置	HEWLETT PACKARD E3616A DC POWER

実験端末の周波数は 800000Hz で固定として、電流計に接続して評価を行った。尚、電力消費モデルにおけるリソース消費量のパラメータ係数は、参考文献 13) にある Galaxy Nexus 1 用のモデルを使用した。

4.3 評価手順

本論文の評価において、2 種類のテスト用アプリケーションを作成した。一つ目は、3 種類の異なった、CPU に負荷のかかる計算を一定回数行うという振る舞いを持つアプリケーション、二つ目は、データ読み書きを振る舞いとして持つアプリケーションである。

更に評価のために、それぞれのアプリケーションで図に示す内容の区別しやすい 3 種類のテストケースを準備した。6 種類のテストケースの平均実行時間を表 2 に示す。

表 2 テストケースの平均実行時間

テストケース	実行時間
CPU テストケース 1	15075ms
CPU テストケース 2	13997ms
CPU テストケース 3	17652ms
読み書きテストケース 1	176.6ms
読み書きテストケース 2	162.6ms
読み書きテストケース 3	121.1ms

電流計に接続した実験用携帯端末を用いて振る舞いを実行中に端末を流れる電流を測定、計算することで、消費電力を実測し、比較した。

実測するにあたって、開始と終了のタイミングを区別する方法としては、for ループによる高負荷な処理と wait 状態を交互に挟むことでマーカーとした。実測値の例としては図 5 のようになった。

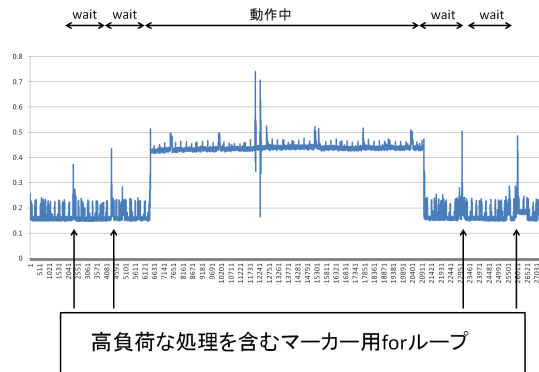


図 5 実測値の例

4.4 評価結果

2 種類のアプリケーションを用いた評価結果を図 6 と図 7 に示す。読み書き動作中心のアプリケーションの評価結果は最低誤差率 8.9%となった。CPU の高負荷動作中心のアプリケーションの評価結果は誤差平均 13.6%となり、テストケースによる誤差率のばらつきも少なかった。全体としての平均誤差率は 20%となった。

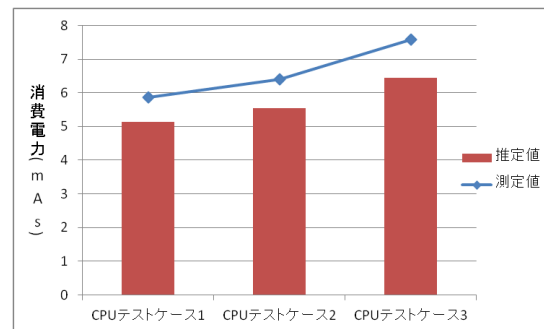


図 6 CPU テストケースのシミュレーションの結果

5. おわりに

本論文では、既存の消費電力解析技術について紹介したあと、UML を用いたソフトウェアの消費電力解析シミュレーションの概要と、その評価方法・評価結果について述べた。既存のコンピュータシステムにおける消費電力解析技術は様々な手法が存在するが、ソフトウェア開発時の消費電力解析には向かない欠点が存在

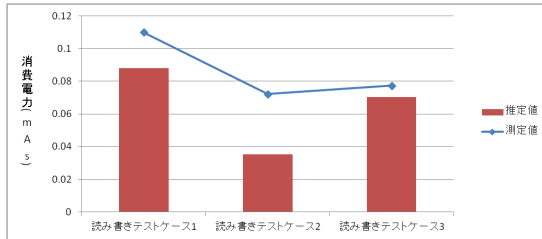


図7 読み書きテストケースのシミュレーションの結果

する。そこで本論文では、ソフトウェア開発時の消費電力解析手法に適した、UMLを用いた消費電力解析シミュレーションを提案した。本手法において入力となるものは、cloocaを用いて振る舞い・リソース消費量等の情報を記したステートマシン図である。ステートマシン図を用いることで、特別な知識を習得する必要なく入力を作成することができる。更にステートマシン図で表現できる範囲であれば、どのような振る舞いも入力として扱うことができる。作成した入力をC言語に変換し、専用のランタイムライブラリを用いてCプロジェクトを作成し消費電力を推定する。推定方法自体は非常に低負荷であり、ほぼ瞬時に推定値を出力することができる。2種類のテスト用アプリケーションを用いて評価を行った結果、平均誤差20%で振る舞いの消費電力を推定することができた。

今後検討すべき課題としては以下のようなものがある。

- 推定精度の向上
- Wi-Fi通信メインのテスト用アプリケーションの制作・適用。
- 様々なテストケースに対しての適用。
- 本解析技術を応用した、ソフトウェアの振る舞いに対応したシステムの最大消費電力の自動見積もりシステム。

参考文献

- 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) V. Tiwari, S. Malik, and A. Wolfe, "Power analysis of embedded software : a first step towards software power minimization," IEEE Trans. Very Large Scale Integr. (VLSI) Syst, vol. 2, no/ 4, pp.437–445, Dec.1994
- 3) G. Qu, N. Usami, and M. Potkonjak, "Function-level power estimation methodology for micro-processors," Proc. Design Automation Confer-

ence, pp.810–813, June, 2000.

- 4) 石原亨, 奥平拓見, 久住憲嗣, 神山剛, 関根和寿, 片桐雅二: 「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」 電子情報通信学会技術研究報告, CPSY2008-92, Mar2009
- 5) 神山剛, 片桐雅二: 「OS レベルの解析に基づく携帯端末の消費電力可視化技術」 NTT DoCoMo テクニカル・ジャーナル, 2009, 17.3: 66–69.
- 6) Doo-Hwan Kim, Jong-Phil Kim, and Jang-Eui Hong : "A Power Consumption Analysis Technique Using UML-Based Design Models in Embedded Software Development" SOFSEM 2011: Theory and Practice of Computer Science, Lecture Notes in Computer Science, Volume 6543. ISBN 978-3-642-18380-5. Springer Berlin Heidelberg, p. 320, 2011
- 7) 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, 2004.
- 8) 日経 BP 社, "日経パソコン", <http://pc.nikkeibp.co.jp/pc/npcs/pdf/110706/tokushu3.pdf>(最終アクセス日 2013/2/7)
- 9) clooca ホームページ, <http://www.clooca.com/>(最終アクセス日 2013/2/7)
- 10) Object Management Group, "OMG Unified Modeling Language", <http://www.uml.org/>.
- 11) V. Garousi, L.C. Briand, Y. Labiche, : "Control Flow Analysis of UML 2.0 Sequence Diagrams", In: A. Hartman, D. Kreische (eds.) ECMDA-FA 2005. LNCS, vol. 3748, pp. 160–174. Springer, Heidelberg, 2001
- 12) Elad Alon, Vladimir Stojanovic, and Mark A Horowitz : "Circuits and Techniques for Resolution Measurement of On-Chip Power Supply Noise" IEEE JOURNAL OF SOLID STATE CIRCUITS, VOL. 40, NO. 4, AP2005
- 13) 古庄裕貴, 久住憲嗣, 神山剛, 稲村浩, 中西恒夫, 福田晃: 「Android アプリケーションの運用時消費電力分析」, 電子情報通信学会技術研究報告, vol.112, no.373, SS2012–58, pp.73–78