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

2009-ARC-181／2009-EMB-11 (17)
2009/1/14

マルチコア上でのOSCAR APIを用いた低消費電力化手法

中川 亮† 間瀬 正啓† 白子 準† 木村 啓二† 笠原 博徳†

† 早稲田大学基幹理工学部 情報理工学科
〒169–8555 東京都新宿区大久保 3–4–1
E-mail: †{ryonaka,mase,shirako,kimura,kasahara}@oscar.elec.waseda.ac.jp

あらまし マルチコアプロセッサ上でアプリケーションプログラム実行時の消費電力を削減するためには，コンパイラやユーザによるプログラム実行中のチップ内リソースの適切な周波数・電圧・電源制御が必要であり，電力制御の指示が可能なAPIの利用が必要となる．本稿ではリアルタイム情報家電マルチコア用のOSCAR (Optimally Scheduled Advanced Multiprocessor) APIを用いた低消費電力化手法を提案する．本手法はOSCARコンパイラに実装されており，最小時間での処理を保証しつつ消費電力を最小化する最速実行モードと，メディアアプリケーションなどのリアルタイム処理において，リアルタイム制約を満たす範囲で消費電力を最小化するリアルタイム制約モードを実現する．NEDOの“リアルタイム情報家電用マルチコア技術”プロジェクトにて，ルネサステクノロジ，日立製作所，早稲田大学が開発した8コアの情報家電用マルチコアRP2上で提案手法を評価したところ，最速実行モードにおいて，SPEC2000のartで消費エネルギーが13.05%削減され，SPEC2000のequakeで消費エネルギーが3.99%削減されることが確かめられた．また，リアルタイム制約モードにおいてはAACエンコーダで平均電力が87.9%削減され，MPEG2デコーダで平均電力が73.2%削減されることが確かめられた．

キーワード マルチコアプロセッサ，低消費電力化制御，並列化コンパイラ，API

A Power Saving Scheme on Multicore Processors Using OSCAR API

Ryo NAKAGAWA†, Masayoshi MASE†, Jun SHIRAKO†,
Keiji KIMURA†, and Hironori KASAHARA†

† Department of Computer Science, Waseda University
Okubo 3–4–1, Shinjuku-ku, Tokyo, 169–8555 Japan
E-mail: †{ryonaka,mase,shirako,kimura,kasahara}@oscar.elec.waseda.ac.jp

Abstract Effective power reduction of an application program on multicore processors requires appropriate power control for each on-chip resource by compilers or users. These low power techniques need an application program interface (API) to realize power control in a user program. This paper proposes a power saving scheme for multicore processors using OSCAR API developed in NEDO “Multicore for Realtime Consumer Electronics” project. The proposed scheme has been implemented in OSCAR compiler to realize the power reduction for fastest execution mode, which minimizes power consumption without performance degradation, and the realtime execution mode to minimize power consumption under realtime constrains. The proposed scheme is evaluated on an 8 cores SH4A multicore processor RP2, newly developed for consumer electronics by Renesas Technology Corp., Hitachi, Ltd. and Waseda University in the above project. For the fastest execution mode, consumed energy was reduced by 13.05% for SPEC2000 art and 3.99% for SPEC2000 equake. Also, for the realtime execution mode, consumed power was reduced by 87.9% for AAC encoder and 73.2% for MPEG2 decoder.

Key words Muticore Processor, Low Power, Parallelizing Compiler, API

1. はじめに

半導体集積度向上に伴ったスケーラブルな処理性能と高い電力効率を実現するために，マルチコアプロセッサは次世代プロセッサアーキテクチャの主流となりつつある．これらマルチコアプロセッサの有効利用には実行するプログラムの適切な並列化が必須であり，従来より多くの自動並列化に関する研究が行われている．OSCAR コンパイラ [1]〜[3] では，従来の市販コンパイラで用いられていたループ並列処理に加え，粗粒度タスク並列処理，近細粒度並列処理を組合せたマルチグレイン並列処理を実現しており，プログラム全域にわたる並列化が可能である．

またマルチコアでの処理性能の向上に加え，増加する消費電力をいかに抑えるかが大きな課題となっており，従来から様々な手法が提案されている．例えばキャッシュミス回数測定用カウンタや命令キューなどのハードウェアサポートにより実行時にプログラム中の各フェーズにおける負荷を判断し，不要なリソースを停止する Adaptive Processing [9] や，計算資源の各部分に対して実行時の負荷に応じた周波数・電圧制御 (FV 制御) を行なう Online Methods for Voltage and Frequency Control [10] やループイタレーションレベルのプロセッサ間の負荷不均衡に対して，実行時のハードウェアサポートにより電力制御を行う Thrifty Barrier [5] などがある．しかし，これらの実行時の情報を利用した手法では，プログラム全域にわたるグローバルな消費電力の最適化は難しく，局所的な最適化にとどまってしまう．また，実行時に電力制御のための追加の処理やハードウェアが必要となるため遅延が大きいという欠点がある．このような手法に対して，コンパイル時の静的な情報に基づく低消費電力化手法では，プログラムの解析による詳細な情報を利用することができるため，よりきめ細やかな電力制御が可能となり，プログラム全域にわたる消費電力の最適化が実現できる．コンパイラ制御によるシングルプロセッサの低消費電力化手法として compiler-directed DVS(dynamic voltage scaling) [11] が挙げられる．しかし，この手法はシングルプロセッサのみを対象としており，マルチコアプロセッサ向けの並列処理と電力制御との両立を実現できていない．マルチコアプロセッサ向けの OSCAR コンパイラではマルチグレイン並列処理による並列性抽出に加え，プログラムの各部分に対するコアごとの適切な動作周波数/電圧および電源遮断のタイミングを決定し，必要な処理性能を維持しつつ電力を低減させている [4], [6].

本稿では，NEDO "リアルタイム情報家電用マルチコア技術" プロジェクトにおいて，早稲田大学，東芝，日本電気，パナソニック，日立製作所，富士通研究所，ルネサステクノロジがマルチコアプロセッサ向けの標準的な API として開発した OSCAR API [7] を用いた，マルチコアプロセッサ向けの低消費電力化手法を提案する．OSCAR API を用いることで，情報家電用マルチコアのためのプログラムを簡単に作成でき，異なるメーカのマルチコア間でのプログラムの移植性を高めることができる．提案手法を 8 コアの情報家電用マルチコア RP2 上で，マルチメディアおよび科学技術計算アプリケーションを用いて評価を行った結果について述べる．

2. OSCAR コンパイラによる低消費電力化手法

本章では OSCAR コンパイラによる低消費電力化手法について述べる．マルチグレイン並列処理により，プログラム全域から並列性を最大限引き出すことが可能となる．しかしプロセッサ通信，同期などの際の待ち時間にリーク電力等の無駄な電力消費が発生してしまう場合がある．また，定められた時刻までに処理を終了すればよいリアルタイム処理においては，ゆっくりと低動作周波数・低電圧で処理を行った方が消費電力を低く抑えられる場合がある．本章では OSCAR コンパイラによる低消費電力化手法が対象とするアーキテクチャと，コンパイラによる電力削減手法について述べる．

提案する低消費電力化手法は以下のアーキテクチャサポートを持つマルチコアプロセッサで最適に利用可能である．

- プロセッサコア毎に周波数が可変 (クロックゲーティングを含む)
- 周波数に応じて電圧も低減可能
- プロセッサコア毎に電源遮断が可能

2.1 OSCAR コンパイラによる低消費電力化手法概要

OSCAR コンパイラによる低消費電力化手法は，最小時間での処理を保証しつつ，消費電力を最小化する最速実行モードと，デッドライン制約を満たす範囲で消費電力を最小化するデッドライン制約モードの 2 つのモードに対応している [6].

本手法では，図 1 のようなマクロタスク (以下，MT) のスケジューリング結果をもとに各 MT に対して適切な動作周波数と電圧を決定する．この例では MT_1, MT_2, MT_5 が PG0(プロセッサグループ 0) に，MT_3, MT_6, MT_8 が PG1 に，MT_4, MT_7, MT_9 が PG2 に割り当てられている．また，図中の time は時間経過を，MT 間の実線はデータ依存を，Margin は与えられたデッドラインまでの余裕度を表している．MT_i の実行時間を T_i と定義すると，マクロタスクグラフ (以下，MTG) の実行時間 T_{TMG} は一般に次のように与えられる [6].

$$T_{MTG} = T_m + T_n + ... + max_1(...) + max_2(...) + ...$$

図 1 の例においては，

$$T_{MTG} = T_1 + max(T_8, T_9) + max(T_2 + T_5, T_6 + max(T_2, T_3), T_7 + max(T_3, T_4))$$

となる．T_i の値は MT_i を実行する周波数によって変化する．デッドライン制約モードで，当該 MTG に対して定められたデッドラインを $T_{MTG\_deadline}$ とすると，$T_{MTG} \leqq T_{MTG\_deadline}$ を満たしつつ，消費電力を最小化するように各 MT_i の動作周波数を決定するのが本手法の基本方針である [6]．ただし，最速実行モードにおいては，$T_{MTG\_deadline} = 0$ であり，最小時間での処理を保証しつつ消費電力を最小化する．

図 1 のスケジューリング結果に対して，本手法を適用した結果の例を図 2 に示す．この例では，MT_3 と MT_5 が最速実行時の半分の周波数で実行するように制御された．

各 MT を処理する電圧と周波数が決定した後でもプロセッサ

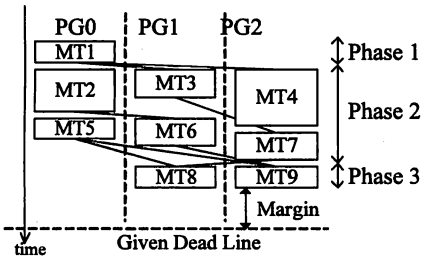

図 1 マクロタスクのスケジューリング結果の例
Fig. 1 Example of Macro Task Scheduling Result

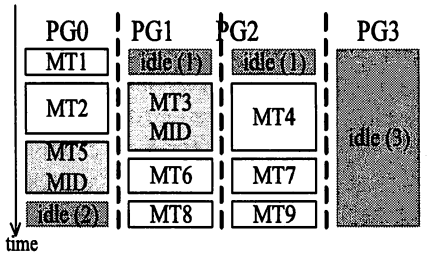

図 2 周波数・電圧制御の結果の例
Fig. 2 Example of F/V Control Result

がアイドル状態の時間（図 2 では idle と示す）が存在する場合がある．そのような場合には，アイドル状態の時間に対してクロックゲーティングや電源遮断を用いることでさらに無駄な電力消費を削減することが可能となる．

2.2 最速実行モードにおける，実行時負荷不均衡並列ループを含む MTG に対する低消費電力化手法

本節では，最速実行モードにおける実行時負荷不均衡並列ループを含む MTG に対する低消費電力化手法について述べる．本手法の目的は，実行時にプロセッサ間で負荷の不均衡が起こりうる並列ループ等を含む MTG において，他のプロセッサより早くバリア同期に到達したプロセッサがビジーウェイト状態で消費している電力を削減することである．

コンパイラによってループが並列処理可能と判定され，各プロセッサに処理が割り当てられたとしても，プログラムの入力などに依存する原因により，実行時にプロセッサ間で負荷の不均衡が起こる可能性がある．例えば，並列ループ内部に条件分岐がある場合や，回転数が実行時まで不定のループがある場合，並列ループの各イタレーションのコストがばらつき，プロセッサ間の負荷不均衡が起こり得る．

そこで MTG 中でバリア処理が必要な場合に，ビジーウェイトによる消費電力を削減するため，クロックゲーティングもしくは電源遮断を適用する．

このように電力制御を適用した場合，最後のプロセッサがバリアに到達した後に，スリープ状態もしくは電源遮断状態にあるプロセッサの電源復帰を行なわなければならない．図 3 に本手法を適用した結果の例を示す．図中の time は経過時間を表す．この例では，コンパイル時には並列ループが数イタレーションずつ均等にプロセッサ 0〜プロセッサ 3 に割り当てられ

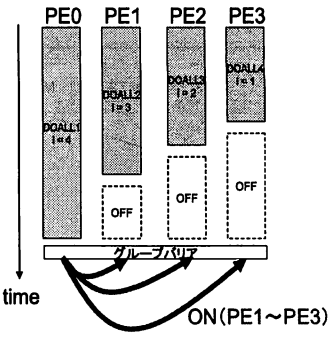

図 3 負荷不均衡並列ループに対する低消費電力化手法の適用例
Fig. 3 Example of Applying Power Saving Scheme for Parallel Loop with Load Imbalance

たが，プログラムの入力などに依存する原因により，実行時にはプロセッサ間で負荷の不均衡が生じた場合を想定している．この例では，3 番のプロセッサ，2 番のプロセッサ，1 番のプロセッサの順に処理が終了し，バリアに到達するため，これらのプロセッサに対してクロックゲーティングもしくは電源遮断が適用される．そして，最後にバリアに到達した 0 番のプロセッサから，1 番〜3 番のプロセッサに対して電源復帰命令が発行される．

2.3 リアルタイム制約モードでの低消費電力化手法

本節では，リアルタイム処理に対して消費電力を最小化するリアルタイム制約モードでの低消費電力化手法について述べる．本手法を適用するためには，2. 章で示したアーキテクチャサポートに加えて，現在時刻を取得することができるタイマーが必要である．

一般にメディアアプリケーションのリアルタイム処理においてはプログラムの一部に周期的なデッドラインが存在する．このようなプログラムに対して，2.1 節で述べたデッドライン制約モードでの低消費電力化手法を適用することができる．しかし，リアルタイム処理の消費電力を最小化するためには，従来のデッドライン制約を保証する範囲での周波数・電圧制御に加え，デッドライン時刻までの待機時間中に発生する電力消費を電源遮断などにより可能な限り削減する必要がある．

例として図 4 のような周期的なリアルタイム制約が存在する MTG を考える．この MTG 全体はループとなっており，周期的に繰り返される．図中の MTG 内部における実線はデータ依存，円弧は and 関係を表している．すなわち MT_1，MT_2，MT_3，MT_4 が終了後に，MT_5，続いて MT_6 が実行可能となる．この MTG に対してコンパイラが 4 つのプロセッサにスケジューリングした結果を図 5 に示す．この段階では，全 MT は最速の周波数で動作するものとしてスケジューリングされる．このスケジューリング結果に対して，リアルタイム制約モードでの低消費電力化手法を適用した結果の例を図 6 に示す．

この例のように，MTG 内の最後の MT(MT_5) の終了時刻からデッドラインまでの余裕度が十分にある場合，最後の MT を処理するプロセッサをマスタプロセッサに指定する．マスタプロセッサ以外のプロセッサは自身に割り当てられた最後の MT

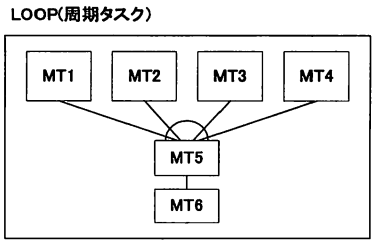

図 4 周期タスクグラフ
Fig. 4 Cyclic Task Graph

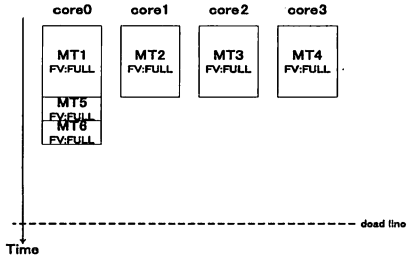

図 5 スケジューリング結果の例
Fig. 5 Example of Scheduling Result

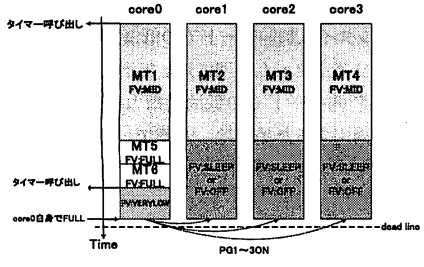

図 6 リアルタイム制約モードの適用結果例
Fig. 6 Result of Applying Realtime Execution Mode

の終了後に電源遮断状態に移行する．マスタプロセッサは自身に割り当てられた先頭 MT の開始時および最後の MT の終了時にタイマーより現在時刻を取得し，デッドラインまでの残り時間を計算する．残り時間が十分に大きい場合はマスタプロセッサはとりうる最低周波数でデッドライン直前までビジーウェイトする．そして，デッドラインの直前に他のプロセッサに対して電源復帰命令を発行し，プログラムは次の周期に移る．

このような制御を行なうことで，リアルタイム制約を守りつつ，電力消費を最小化することが可能となる．

3. 電力制御用 OSCAR API

本章では提案手法を実装した OSCAR コンパイラが出力する電力制御用の OSCAR API について述べる．OSCAR API は OpenMP をベースとした API で，並列実行，メモリ配置，データ転送，電力制御，タイマ，同期の 6 つのカテゴリ，15 の指示文で構成される [7].

本稿では OSCAR API のうち，図 7 に示す 3 つの API を

```
#pragma oscar get_fvstatus(pe_no, module_id, fv_state)
#pragma oscar fvcontrol(pe_no, (module_id, fv_state))
#pragma oscar get_current_time(current_time, timer_no)
```

図 7 本手法で用いた OSCAR API
Fig. 7 OSCAR API for Proposed Scheme

用いて，情報家電用マルチコア RP2 上での電力制御を実現した．指示文 get_fvstatus によって，pe_no 番目のプロセッサの，module_id で示されたモジュールの電力状態を変数 fv_state に取得することができる．本手法ではこの指示文を用いて，状態遷移中の電源復帰命令発行の防止を実現した．また，指示文 fvcontrol によって，pe_no 番目のプロセッサの，module_id で示されたモジュールの電力状態を変数 fv_state で指定した電力状態に変更することができる．また，指示文 get_current_time によって，timer_no 番目のタイマーから現在時刻を取得し，変数 current_time に取得することができる．この指示文を用いて，2.3 節で述べたリアルタイム制約モードでの低消費電力化手法を情報家電用マルチコア RP2 上で実現した．

4. 性能評価

2. 章で述べた低消費電力化手法を実装した OSCAR コンパイラの性能評価を情報家電用マルチコア RP2 上で行なった．

4.1 情報家電用マルチコア RP2

本節では，提案手法の評価に用いた RP2 について述べる．RP2 は NEDO の “リアルタイム情報家電用マルチコア技術” プロジェクトにおいて，早稲田大学，ルネサステクノロジ，日立製作所によって共同開発された情報家電向けマルチコアプロセッサで，一つのチップ上に 8 つの SH-4A のコアを搭載している．RP2 の構成図を図 8 に示す．各プロセッサコアには，CPU，キャッシュ，ローカルメモリ (ILRAM，OLRAM)，分散共有メモリ (URAM)，DTU などが搭載されており，4 コアまではスヌープコントローラが MESI プロトコルでキャッシュコヒーレンシを保証する SMP モードとして動作する．5 コア以上を使用する場合は，ソフトウェアが明示的にキャッシュコヒーレンシを保証する必要がある．

RP2 では，各 CPU の周波数を 600MHz，300MHz，150MHz，75MHz に独立して変更することが可能である．また，供給電圧はチップ一括での変更が可能である．それに加えて，コアの中の CPU のみのクロックを遮断する Light Sleep，コアの中の URAM と DTU 以外のクロックを遮断する Normal Sleep，コア中の URAM のクロックを遮断，それ以外を電源遮断する Resume Standby，コアの電源をすべて遮断する CPU off の 4 つの低電力状態もとることが可能である．表 1 に RP2 の持つ電力状態と，8 つのプロセッサを各電力状態にした場合の消費電力を示す．

また，電源復帰は他のコアからの電源復帰命令の発行を受けた割り込み処理によって実現されている．

4.2 最速実行モードでの SPEC2000 art 及び equake を用いた評価

SPEC2000 の art と equake を制約付き C [8] に書き換えた

表 1　RP2 の電力状態

Table 1　Power Modes of RP2

状態名	クロックゲーティング対象モジュール	電源遮断対象モジュール	消費電力 [W]
FULL(600MHz, 1.404V)	なし	なし	5.99
MID(300MHz, 1.196V)	なし	なし	2.61
LOW(150MHz, 1.004V)	なし	なし	1.27
VERYLOW(75MHz, 1.004V)	なし	なし	1.00
Light Sleep	CPU	なし	1.089
Normal Sleep	CPU, cache, ILRAM, OLRAM	なし	0.725
Resume Standby	URAM	CPU, cache, ILRAM, OLRAM, DTU	0.563
CPU off	なし	CPU, cache, ILRAM, OLRAM, DTU, URAM	0.554

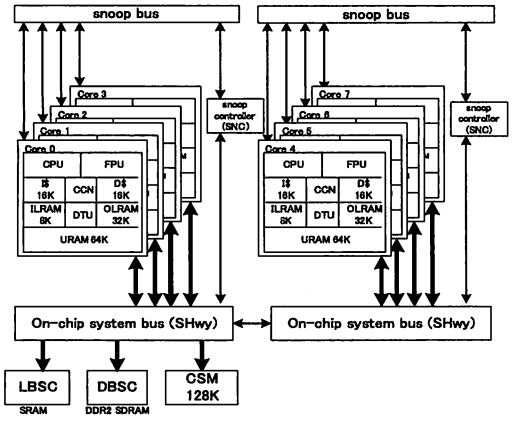

図 8　RP2 の構成図

Fig. 8　Configuration of RP2

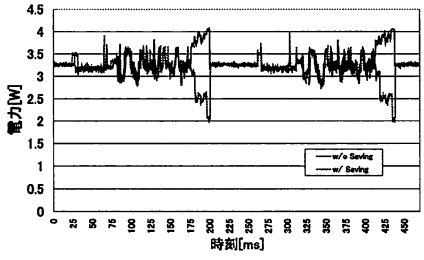

図 9　equake の電力波形

Fig. 9　Power Wave for equake

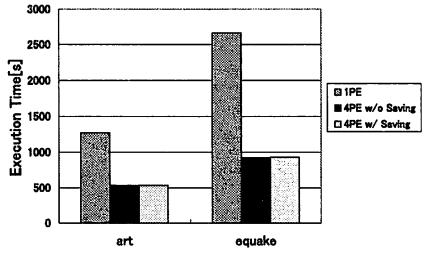

図 10　最速実行モードにおける実行時間

Fig. 10　Execution Time in Fastest Execution Mode

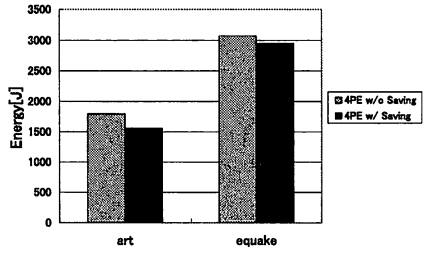

図 11　最速実行モードにおける消費エネルギー

Fig. 11　Energy Consumption in Fastest Execution Mode

プログラムを用いて，RP2 上で最速実行モードでの電力評価を行なった．評価には 4 コアの SMP モードを用い，使用しないプロセッサはプログラム開始時に CPU off を適用した．また，2.2 節で示した条件を満たし電力制御適用ループと判定した場合，ループを割り当てられた各プロセッサに対して，処理が終了しバリアに到達後，Resume Standby を適用した．

図 9 に equake を実行した際の電力波形の抜粋を示す．図の横軸が時刻，縦軸が消費電力 [W] を表している．図 9 の電力制御が適用された箇所の波形を見ると，まず 2.5[W] 付近まで電力が下がり，次に 2.05[W] 付近まで電力が下がっていることが分かる．これは，各プロセッサがバリアに到達した順に電力制御が適用されたためである．

提案手法を適用した場合と適用しない場合の，実行時間と消費エネルギーを比較した．ただし，データの入出力部分は評価から除いている．まず実行時間を図 10 に示す．art における電力制御適用時の遅延率は 0.0042%，equake における遅延率は 0.62%であった．次に平均電力を図 11 に示す．art において，低消費電力化制御を適用しない場合の消費エネルギーは 1795.32[J]，適用した場合の消費エネルギーは 1560.92[J] で 13.05%の消費エネルギー削減効果を得た．equake において，低消費電力化制御を適用しない場合の消費エネルギーは 3071.64[J]，適用した場合の消費エネルギーは 2949.07[J] で 3.99%の消費エネルギー削減効果を得た．

4.3　リアルタイム制約モードでの AAC エンコーダ及び MPEG2 デコーダを用いた評価

ルネサステクノロジ提供の AAC エンコーダと，MPEG2 デコーダを制約付き C に書き換えたプログラムを用いて，RP2 上

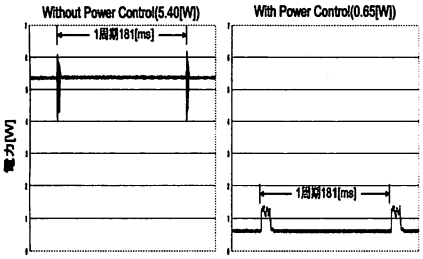

図 12 AAC エンコーダにおける電力波形
Fig. 12 Wave form for AAC encoder

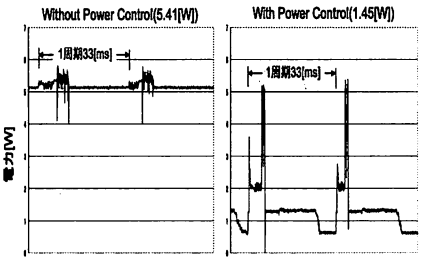

図 13 MPEG2 デコーダにおける電力波形
Fig. 13 Wave form for MPEG2 decoder

でリアルタイム制約モードでの電力評価を行なった．AAC エンコーダにおけるリアルタイム制約は 44.1[フレーム/s]，MPEG2 エンコーダにおけるリアルタイム制約は 30[フレーム/s] とした．なお，評価に先立ち，RP2 上でこれらのプログラムを実行したプロファイル情報 (プログラム中のループや関数の実行時間や実行回数の情報) をコンパイラに与え，コストの見積もり精度を高めている．

AAC エンコーダおよび MPEG2 デコーダを実行した際の電力波形の抜粋をそれぞれ図 12 と図 13 に示す．図の横軸が時刻，縦軸が電力 [W] を表す．AAC エンコーダにおいては，デッドラインまでの余裕度が十分に大きいため，エンコード処理は LOW の状態で行なわれ，その後，デッドラインまで VERY LOW 及び Resume Standby が適用された．

提案手法を適用した場合と適用しない場合の平均電力を図 14 に示す．AAC エンコーダにおいて，低消費電力化制御を適用しない場合の平均電力は 5.40[W]，適用した場合の平均電力は 0.65[W] で 87.9%の消費電力削減効果を得た．MPEG2 デコーダにおいて，低消費電力化制御を適用しない場合の平均電力は 5.41[W]，適用した場合の平均電力は 1.45[W] で 73.2%の電力削減効果を得た．

5. ま と め

本稿では OSCAR API を用いたマルチコアプロセッサ向けの低消費電力化手法を提案し，情報家電用マルチコア RP2 上で評価を行った．最速実行モードでは，負荷不均衡が起こりうる並列ループにおいて電源遮断を適用することで，ビジーウェイトによる無駄な電力消費の削減を実現し，SPEC2000 の art と equake において消費エネルギーをそれぞれ 13.05%，

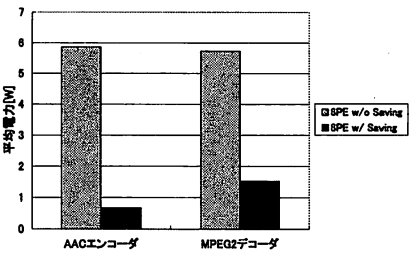

図 14 リアルタイム制約モードでの平均電力
Fig. 14 Average Power in Realtime Execution Mode

3.99%削減することができた．また，メディアアプリケーションのリアルタイム処理向けのリアルタイム制約モードでは AAC エンコーダと MPEG2 デコーダにおいて平均電力をそれぞれ 87.9%，73.2%削減することができた．

謝辞 本研究の一部は NEDO“リアルタイム情報家電用マルチコア技術” “情報家電用ヘテロジニアスマルチコア技術” プロジェクト，早稲田大学グローバル COE“アンビエント SoC” の支援により行なわれた．

文 献

[1] H.Kasahara, et al.: “A multi-grain parallelizing compilation scheme on oscar”, Proc. 4th Workshop on Language and Compilers for Parallel Computing(1991).

[2] 本多，岩田，笠原: “Fortran プログラム粗粒度タスク間の並列性検出手法”，電子情報通信学会論文誌，Vol.J73-D-1，No.12，pp.951-960(1990).

[3] 笠原: “最先端の自動並列化コンパイラ技術”，情報処理，Vol.44 No.4(通巻 458 号)，pp.384-392(2003).

[4] J.Shirako, K.Kimura and H.Kasahara: “Power Reduction Controll for Multicores in OSCAR Multigrain Parallelizing Compiler”, Proc. of International SoC Design Conference(2008).

[5] J.Li, J.F.Martinez and M.C.Huang: “The Thrifty Barrier: Energy-Aware Synchronization in Shared-Memory Multiprocessors”, Proc. of High Performance Computer Architecture(2004).

[6] 白子，吉田，押山，和田，中野，鹿野，木村，笠原: “マルチコアプロセッサにおけるコンパイラ制御低消費電力化手法”，情報処理学会論文誌，47，ACS15(2006).

[7] OSCAR API 仕様書，
http://www.kasahara.cs.waseda.ac.jp/api/regist.html.

[8] 間瀬，馬場，長山，田野，益浦，宮本，白子，中野，木村，笠原: “情報家電用マルチコア smp 実行モードにおける制約付き c プログラムのマルチグレイン並列化”，組込みシステムシンポジウム 2007(2007).

[9] D.H. Albonesi et al: “Dynamically tuning processor resources with adaptive processing”, IEEE Computer(2003).

[10] Q.Wu, P.Juang, M.Martonosi and D.W.Clark: “Formal Online Methods for Voltage/Frequency Control in Multiple Clock Domain Microprocessors”, Eleventh International Conference on Architectural Support for Programming Languages and Operating Systems(2004).

[11] C.Hsu and U.Kremer: “The Design, Implementation and Evaluation of a Compiler Algorithm for CPU Energy Reduction”, The ACM SIGPLAN Conference on Programming Language Design and Implementation(2003).