低消費電力マルチコア RP2 上での複数メディアアプリケーション実行時の消費電力評価

見神広紀[†]　北基俊平[†]　佐藤崇文[†]　間瀬正啓[†]
木村啓二[†]　石坂一久[††]　酒井淳嗣[††]　枝廣正人[††]
笠原博徳[†]

組み込み向けマルチコアではユーザーのインタラクティブな操作等で複数の逐次あるいは並列プロセスが動作させる環境においても高い性能を得ることが重要となる．さらに組み込みマルチコアプロセッサを省電力で動作させるためには，動作周波数・電圧を動的に制御することが重要となる．本論文では，OSCAR 自動並列化コンパイラにより自動で電力制御された各アプリケーションを複数同時実行した際の電力性能をルネサスエレクトロニクス/日立/早稲田大学で開発した 8 コアのマルチコア RP2 上で評価した．コンパイラによるデッドライン制御モードにおいて，1 コアでもリアルタイム制約を低周波数動作で満たせる軽負荷の AAC エンコーダの場合には，各アプリケーションに 1 コアを割り当て同時実行したときが 1 アプリケーションあたりの電力が最小になったが，1 プロセッサでは高周波数動作しないとデッドラインを満たせない中負荷の AAC エンコーダでは各 AAC エンコーダを 2 プロセッサで並列処理しつつ複数実行した場合のアプリケーションあたりの電力が低く，1 アプリのみ実行した場合には 1 プロセッサで 1.95W を要したものが 4 プロセッサで 0.84W に下げられる事がわかった．高負荷の MPEG2 デコーダでは 4 プロセッサで並列処理しつつ複数実行した時が 1 アプリケーションの電力が最小となり，負荷のあるアプリケーションでは並列処理を行いつつ電力制御を行うことが有効であると確かめられた．

Evaluation of Power Consumption by Executing Media Applications on Low-power Multicore RP2

Hiroki Mikami[†]　Shumpei Kitaki[†]　Takafumi Sato[†]
Masayoshi Mase[†]　Keiji Kimura[†]　Kazuhisa Ishizaka[††]
Junji Sakai[††] Masato Edahiro[††] and Hironori Kasahara[†]

On embedded multicores, it is important to obtain high performance although multiple sequential or parallel applications run together. In addition, it is important to control frequency and voltage executing application with low power. This paper evaluated OSCAR compiler's power reduction control with media applications on Renesas Electronics / Hitachi / Waseda RP2. Scheduling with "Deadline mode", power consumption is minimum when it allocates 1 application for 1 processor by low-complex long-wait applications. But in middle-complex and high-complex applications, power consumption is minimum when applications are executed by parallel. It is confirmed parallel processing and frequency / voltage control is effective on multicore processors.

1. はじめに

近年，情報家電向け組込み分野におけるマルチコアの採用が進み，SCE・IBM・東芝の Cell Broadband Engine[1)]，ARM の MPCore[2)]，ルネサスエレクトロニクスの NaviEngine[3)]，富士通の FR1000[4)]，パナソニックの UniPhier[5)]，ルネサステクノロジの SH-X3[6)] が発表され，携帯機器，カーナビ，デジタルテレビやゲーム機等に利用され始めている．これらのマルチコア上ではアプリケーションの単独動作のみならず，複数アプリケーションの同時実行が考えられる．

一方，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いる．

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[7)] 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[8)]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[9)]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．

そこで本論文では，OSCAR コンパイラによって実現されている，並列化コンパイ

† 早稲田大学
Waseda University
†† 日本電気株式会社
NEC Corporation

ラによる自動低消費電力制御手法 [10), 11)] と，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 OSCAR API[12]を用いて RP2 上での低消費電力制御を行い，複数のメディアアプリケーションを同時実行し太時の電力評価を行った．

以下，2 章で OSCAR 自動並列化コンパイラとその低消費電力制御について，3 章で情報家電用マルチコア RP2 と評価アプリケーションについて, 4 章で RP2 での消費電力評価の結果について述べる．

2. OSCAR 自動並列化コンパイラによる粗粒度タスク並列処理

本章では，粗粒度タスク並列性の抽出，データローカリティ最適化手法であるデータローカライゼーション手法，粗粒度タスクスケジューリング手法，低消費電力制御手法について述べる．

2.1 粗粒度タスク並列性抽出

粗粒度タスク並列処理では，まずソースプログラムを基本ブロックあるいは基本ブロックを融合・分割した形である疑似代入文ブロック（BPA），ループの一般形である繰り返しブロック（RB），サブルーチンブロック（SB）の 3 種類のマクロタスク（MT）に分割する．ループ並列処理不可能な実行時間の大きい RB や，インライン展開を効果的に適用できない SB に対しては，その内部を階層的に粗粒度タスクに分割して並列処理を行う．マクロタスクの生成後，マクロタスク間のコントロールフローとデータ依存を解析し，図 1（a）に示すようなマクロフローグラフ（MFG）を生成する．図 1（a）の各ノードはマクロタスクを表し，実線エッジはデータ依存を，点線エッジはコントロールフローを表す．また，ノード内の小円は条件分岐を表す．

MFG 生成後, コンパイラは並列性を抽出するためにコントロールフローとデータ依存の両方を考慮した最早実行可能条件解析を MFG に対して行う．マクロタスクの最早実行可能条件とは，コントロール依存とデータ依存を考慮したそのマクロタスクが最も早い時点で実行可能になる条件である．マクロタスクの最早実行可能条件は図 1（b）に示すようなマクロタスクグラフ（MTG）で表される．MFG，MTG 共にエッジの矢印は省略されているが，下向きが想定されている．MTG では横に並んでいるマクロタスク間の並列性が表現されている．

2.2 データローカライゼーション手法

データローカライゼーション手法とは，アクセス速度の異なる複数のメモリ機構を持つアーキテクチャにおいて，プロセッサ近傍の高速なメモリを有効に利用して，処理速度の向上を図るものである．データローカライゼーション手法は従来の単一ループネスト内のブロッキングを用いたキャッシュ最適化とは異なる，複数のループネスト集合に対するグローバルなキャッシュあるいはローカルメモリ最適化技術である．データ依存を持つ複数のループをそれらの使用データサイズがキャッシュあるいはローカルメモリサイズにおさまるように，複数ループを整合して分割をする [13)]．分割後のデータローカライザブルグループ（DLG）を同一プロセッサで連続実行することでキャッシュあるいはローカルメモリを介したデータの授受により複数ループネスト間でのデータ転送の最小化が可能となる．

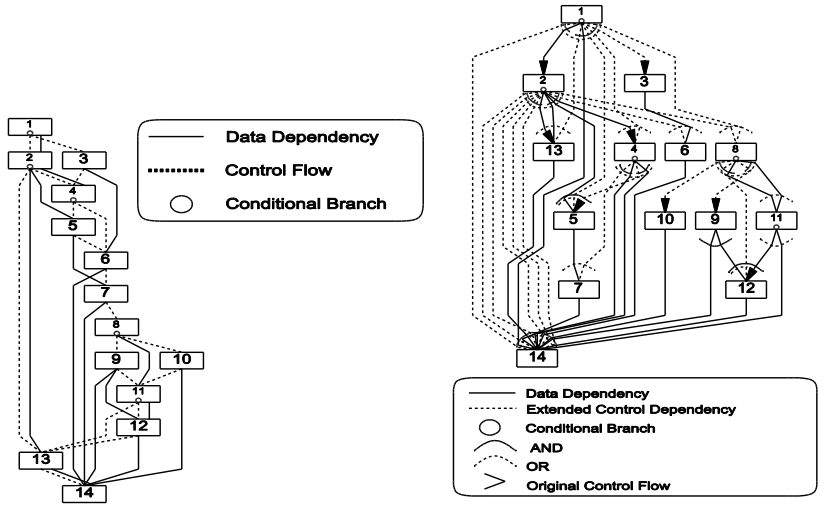

図 1　マクロフローグラフとマクロタスクグラフ

2.3 粗粒度タスクスケジューリング手法

粗粒度タスク並列処理では，生成されたマクロタスクを複数のプロセッサエレメント (PE) から構成されるプロセッサグループ (PG) に割り当てて実行することにより，マクロタスク間の並列性を利用する．本報告書では MTG がデータ依存エッジのみを持つ場合にコンパイラがスタティックにマクロタスクの PG への割り当てを決定することで実行時スケジューリングオーバーヘッドを無くし，データ転送と同期のオーバーヘッドを最小化することが可能であるスタティックスケジューリングを用いた．本

論文では中粒度，近細粒度の並列性を利用しないため以後 PG を PE として表す．

2.4 コンパイラによる低消費電力制御手法 [10), 11)]

ここでは，OSCAR コンパイラによって実現される低消費電力制御 [10), 11)] の概念について述べる．本手法では．OSCAR コンパイラによって対象アプリケーションを並列化する際に，プログラムの特性を考慮して，デッドライン制約に応じた周波数・電圧制御やプロセッサを停止するクロックゲーティング及びパワーゲーティングなどの処理を自動的に挿入する．入力プログラムの持つ特性や並列性によっては，チップ上の資源に対して必ずしも充分な並列性が得られない場合があり，この時，処理の割り当てられていない資源は無駄な電力を消費してしまうことになる．本手法は，そのような場合あるいはプロセッサが同期待ちをしている場合に周波数・電圧制御や電源制御等を適用することで，無駄に消費される電力を削減する．

OSCAR コンパイラによって実現される消費電力制御手法には，実行時間最少スケジューリングモードとデッドライン制約スケジューリングモードがある．図 2 にこれらの基本概念を示す．図中の実行時間最小スケジューリングモードでは，プログラムのクリティカルパスにあたる処理に対しては消費電力制御を行わず，それ以外の部分に対して消費電力制御を行うものである．これにより，プログラムの実行時間を延ばすこと無く低消費電力化を実現する．また，図中のデッドラインスケジューリングモードでは，プログラム終了のデッドラインを保証する範囲内で電力消費が最小となるように低消費電力制御を行う．本論文では，デッドラインスケジューリングモードを使用している．

なお，本手法では，プログラム内の各 MTG に対してスタティックスケジューリングが適用された結果に対し，粗粒度タスク間の負荷バランスや各 MTG の実行終了時間などを考慮して各 MT を実行する際の周波数を決定する．この際，周波数・電圧切り替えのオーバヘッドやそれぞれの MT の取り得る周波数などを考慮して周波数･電圧，電源制御を適用する．

3. 複数アプリケーション実行の評価環境

本章では，複数アプリケーションの同時実行時の評価環境として，早稲田大学，ルネサスエレクトロニクス，日立製作所開発の RP2，および評価対象アプリケーションとその特徴について述べる．

3.1 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 [14)]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，ルネサスエレクトロニクス，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

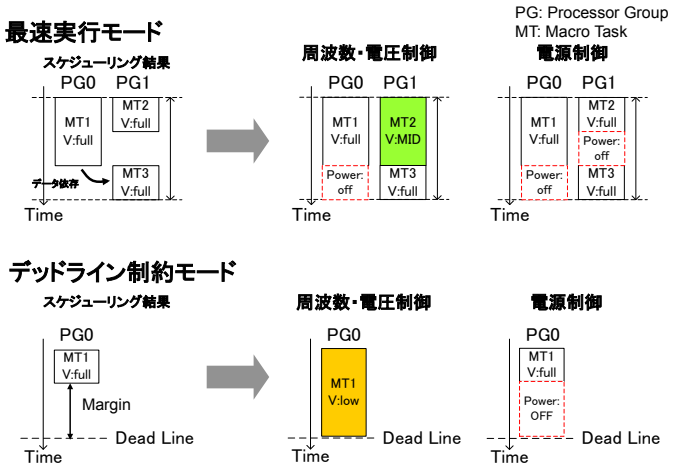

図 2 OSCAR コンパイラにおける低消費電力化手法の概要

図を図 3 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．

3.2 評価対象アプリケーション

本論文では，評価対象アプリケーションとして，マルチメディアアプリケーションである AAC エンコードと MPEG2 デコードを用いた．

状態	クロックゲーティング対象	電源遮断対象	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
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

表 1 RP2 の電力状態

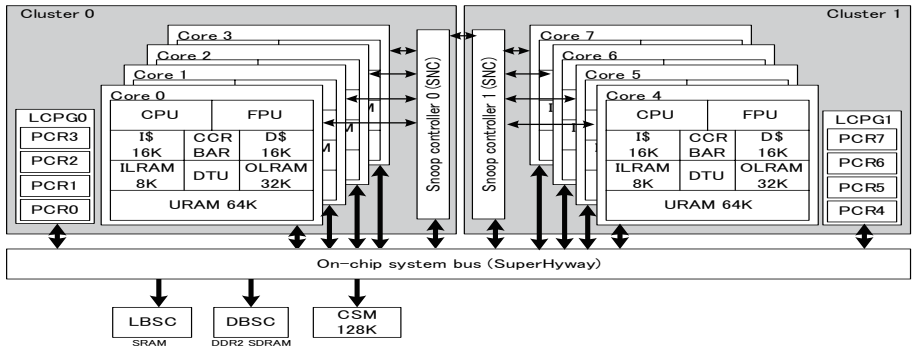

図 3 RP2 の構成図

3.2.1 AAC エンコード

本プログラムは音声データの読み込み後，各フレームに対してフィルタバンク処理，MS ステレオ処理，量子化およびハフマン符号化を行う．本プログラムに実装されているアルゴリズムでは，各フレームに対する処理を並列に行うことが可能であり，粗粒度並列性抽出の為，エンコード処理はプロセッサ数分アンローリングされている．低消費電力制御のデッドラインは 1 フレームの処理について 23ms と指定してスケジューリング及び制御を行っている．

本論文で用いる AAC エンコーダはルネサスエレクトロニクスおよび日立製作所提供の AAC-LC エンコードプログラムを Parallelizable C [15] で参照実装したものを使用した．

3.2.2 MPEG2 デコード

MPEG2 デコード処理は，可変長複合化，逆量子化，逆量子化後の各係数値の制限処理，逆離散コサイン変換，動き補償予測，足し合わせ処理の 6 つのステージからなる．MPEG2 デコード処理ではスライスレベルの並列性とスライス処理内部でのマクロブロックレベルの並列性が存在する．本報論文では並列性向上のためにスライスに対する可変長復号化処理中のスライスヘッダの検出処理を分割するプレスキャニング手法を適用した．プレスキャニング手法ではビットストリームを先頭から走査するためにスライス毎に逐次で処理を行う必要がある．低消費電力制御のデッドラインはデコードの 1 フレーム分について 33ms と指定してスケジューリング及び制御を行っている．

本論文では，MediaBench[16]の``mpeg2decode"を Parallelizable C[15]により参照実装した MPEG2 デコードプログラムを使用した．

3.3 アプリケーションの負荷

本論文で使用した各アプリケーションの負荷について表 2 に特徴をまとめた．

軽負荷のアプリケーションとしては 1 プロセッサによる低周波数動作でもデッドラインを満たせる AAC エンコーダを選択した．具体的にはデッドライン 19 秒に設定した AAC エンコーダを用いる．AAC エンコーダは入力 19 秒のデータを使用した際に，1 プロセッサでの実行時間が 2.7 秒でありデッドラインまで十分に時間がある．

1 プロセッサでは最高周波数で動作市内とデッドラインを満たせない中負荷のアプリケーションとしては，デッドラインを 3 秒に設定した AAC エンコーダを用いる．前述の通り入力 19 秒の時に 1 プロセッサでの実行時間は 2.7 秒であり，1 プロセッサ実行時には FULL の電力状態で実行する必要のあるアプリケーションである．

高負荷のアプリケーションとしては 352x128 ピクセル解像度の MPEG2 デコーダを用いる．MPEG2 デコーダは 1 プロセッサ時の実行時間が 8.3 秒である．加えて，プレスキャニングではビット走査のために連続的に I/O が発生する I/O 負荷の重いアプリケーションでもある．

最後に，超高負荷アプリケーションとして 352x240 ピクセル解像度の MPEG2 デコーダを用いる．MPEG2 デコーダは 352x240 ピクセル解像度では実行時間は 1 プロセッサで 17.6 秒，2 プロセッサで 11.1 秒であり 2 プロセッサ以上で並列処理をしなければ，デッドライン 15 秒を満たす事ができないアプリケーションである．

本論文では，これらの負荷の異なるアプリケーションを組み合わせて実行した際の

消費電力について述べる．

負荷	対応アプリケーション	特徴
軽負荷	AAC エンコーダ （デッドライン 19 秒）	・デッドラインまでの待ち時間に余裕がある ・エンコードの計算負荷が低い
中負荷	AAC エンコーダ （デッドライン 3 秒）	・デッドラインまでの待ち時間に余裕がない ・エンコードの計算負荷が低い
中負荷	MPEG2 デコーダ （解像度 352x128）	・デッドラインまでの待ち時間に余裕がない ・デコードの計算負荷が高い ・I/O 負荷が高い
高負荷	MPEG2 デコーダ （解像度 352x240）	・並列処理をしないとデッドラインを満たせない ・デコードの計算負荷がとても高い ・I/O 負荷が高い

表 2 アプリケーションの負荷

4. 複数アプリケーション動作時の消費電力評価

本章では，OSCAR 自動並列化コンパイラにより並列化および電力制御された評価アプリケーションを情報家電用マルチコア RP2 上で複数動作させた際の消費電力を評価した．

4.1 軽負荷アプリケーション同士の複数同時実行

前述の軽負荷 AAC エンコーダを複数同時実行させたときの消費電力を図 4 に示す．横軸は，使用した総プロセッサ数を示し，縦軸は，図 4 はプロセッサ全体の消費電力を示している．図中の各バーは各 AAC エンコーダを同時実行する数を示しており，電力制御なしは OSCAR コンパイラによる電力制御を適用せずに 1 つの AAC を実行した場合の消費電力を示し，1AAC は 1 つの AAC をプロセッサ数で並列実行した場合の消費電力，2AAC は 2 つの AAC でプロセッサ数を半分ずつ使って同時実行した場合の消費電力，4AAC は 4 つの AAC でプロセッサ数を均等に 4 分割して同時実行した場合の消費電力，8AAC は 8 つの AAC でプロセッサ数を均等に 8 分割して同時実行した場合の消費電力を示している．

AAC エンコーダはデッドラインスケジューリングモードにおいて，デッドラインに対して処理が十分小さい軽負荷のメディアアプリケーションであり，1 プロセッサで実行される際にも常に電力が最小である VERYLOW の電力状態で動作している．そのため，電力制御を伴う並列処理をしても電力はほぼ 0.60W と電力を削減する事はできないが 4 プロセッサで増加しないように制御が行われている．周波数を単一プロセッサ時より下げる事が出来ないので消費電力は変わらない．この時電力制御をしない場合 1.89W に比べ，消費電力は 8 プロセッサで最大 68%の削減されている．複数同時実行の場合，消費電力は緩やかに増加し，8AAC で 8 プロセッサの場合に平均電力は 1.01W となった．この時，1 つの AAC あたりにかかる電力は，プロセッサにかかるスタティック電力が 8 等分されるため平均 0.13W となり，8 プロセッサで 1 つの AAC を実行した場合に比べ 78%低い電力で実行できる．

AAC エンコーダのような軽負荷のメディアアプリケーションの場合，並列化による電力削減より出来るだけ多く複数実行する方が効率よく処理が出来ると言える．

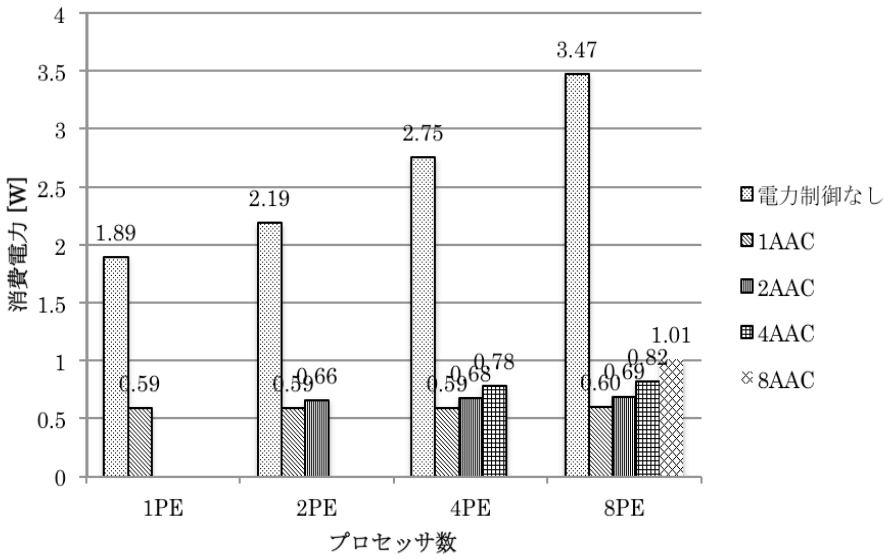

図 4 AAC エンコーダ（軽負荷）複数実行時の平均消費電力

4.2 中負荷アプリケーション同士の複数実行

前述の中負荷 AAC エンコーダを複数同時実行させたときの消費電力を図 5 に示す．横軸は，使用した総プロセッサ数を示し，縦軸はプロセッサ全体の消費電力を示している．図中の各バーは各 AAC エンコーダを同時実行する数を示しており，電力制御なしは OSCAR コンパイラによる電力制御を適用せずに 1 つの AAC を実行した場合の

消費電力を示し, 1AAC は 1 つの AAC をプロセッサ数で並列実行した場合の消費電力, 2AAC は 2 つの AAC でプロセッサ数を半分ずつ使って同時実行した場合の消費電力, 4AAC は 4 つの AAC でプロセッサ数を均等に 4 分割して同時実行した場合の消費電力を示している.

中負荷 AAC エンコーダでは, 1 プロセッサで動作時には FULL の電力状態で実行しなければデッドラインに間に合わなくなるため，中負荷のアプリケーションにおいては，1 プロセッサで 1.95W を要するものが 8 コアで電力制御を行うと 0.83W に電力を削減できる．この時 1 アプリケーションあたりのプロセッサ数を増やすことで，プロセッサ数を倍に増やすごとに平均 30%の電力削減率となっている．同時実行の場合においても 1 アプリケーションあたりのプロセッサ数を増やすことで電力削減の効果が見られる． 4AAC で各 AAC を 2 プロセッサで実行した時，1AAC あたりの消費電力は 0.55W となり，8 プロセッサで 1 つの AAC を実行した場合に比べ 37%，1 プロセッサで 1 つの AAC を実行した場合に比べの 69%電力を削減できる．

デッドラインを短縮した AAC エンコーダのような中負荷のメディアアプリケーションの場合，アプリケーションを並列化し複数実行することで電力・エネルギーを押さえることが出来ると言える．

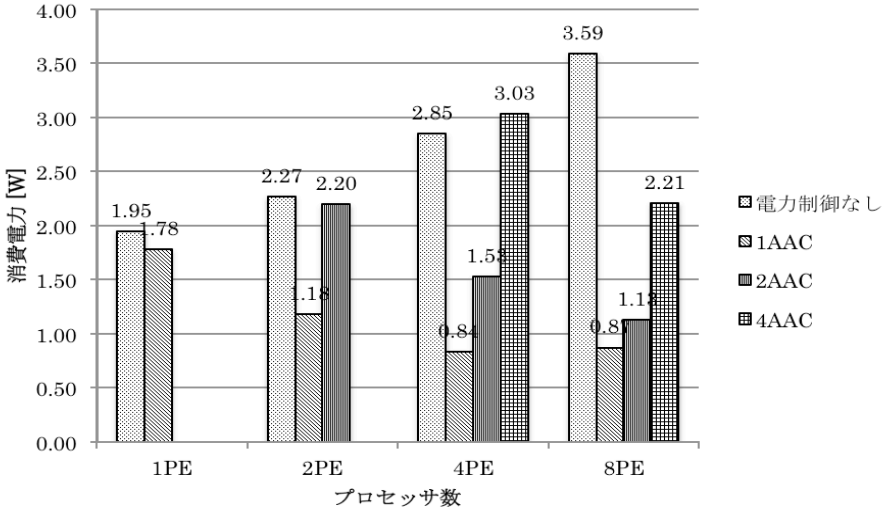

図 5 中負荷 AAC エンコーダ複数実行時の消費電力

4.3 中負荷 MPEG2 デコーダアプリケーション同士の複数実行

中負荷 MPEG2 デコーダを複数同時実行させたときの消費電力を図 6 に示す．横軸は，使用した総プロセッサ数を示し，縦軸は，プロセッサ全体の消費電力を示している．図中の各バーは各 MPEG2 デコーダを同時実行する数を示しており，電力制御なしは OSCAR コンパイラによる電力制御を適用せずに 1 つの MPEG2 を実行した場合の消費電力を示し，1MPEG2dec は 1 つの MPEG2 をプロセッサ数で並列実行した場合の消費電力，2MPEG2dec は 2 つの MPEG2 でプロセッサ数を半分ずつ使って同時実行した場合の消費電力を示している．

この場合は実行で 1.99W 要する処理が 4 コア並列実行で 1.0W と削減されている. 2 つの MPEG2 デコーダを各 4 プロセッサ割り当てて実行した場合の消費電力は 1.46W であり, これは 1 つの MPEG2 デコーダを 1 プロセッサで実行した時の消費電力 1.49W より低い電力で実行できる事がわかる．また，この時に 1 つの MPEG2 デコーダあたりの消費電力は 0.73W となり，1 つの MPEG2 デコーダを 1 プロセッサで実行した場合の消費電力 1.49W に比べ消費電力を 51%削減できる．MPEG2 デコーダは計算負荷に加えてプレスキャニング時に入力データを走査するために I/O への負荷が大きい．このため複数実行する際には I/O のタイミングをずらして実行する必要がある．その結果，同じアプリケーションであっても周波数制御のタイミングが異なるので，プロセッサ一括でしか制御できない動作電圧が下がりにくくなっているため AAC エンコーダと比較すると消費電力が下がりにくくなっている．

このような高負荷のアプリケーションで効率よく電力を削減するには，ソフトウェアによる並列処理と電力制御に加え，ハードウェアにプロセッサの電圧制御ドメインを区切ることが出来るような仕組みが必要である．

4.4 中負荷アプリケーションと高負荷アプリケーションの複数実行

352x128 ピクセルの解像度の 450 フレーム（15 秒）の入力データに対してデッドラインを 15 秒に指定した MPEG2 デコーダと，352x240 ピクセルの解像度の 450 フレーム（15 秒）の入力データに対してデッドラインを 15 秒に指定した高負荷 MPEG2 デコーダを複数同時実行させたときの消費電力を図 7 に示す．横軸は，前者は 352x240 ピクセル解像度の MPEG2 で使用したプロセッサ数を示し後者は 352x128 ピクセル解像度の MPEG2 で使用したプロセッサ数を示している．縦軸は，プロセッサ全体の消費電力を示している．

MPEG2 デコーダで解像度が違うものを同時実行した場合，デコードの各ステージでの実行時間が異なるため，周波数制御のタイミングがずれる．このため，プロセッサ一括でしか制御できない動作電圧が下がりにくくなる．さらに解像度が異なる場合，そもそもコンパイラの指定した電力状態が異なるという事が上げられる．2 プロセッサ+2 プロセッサの場合，電力状態は FULL+MID となるため本来 MID ならば電圧は

1.196V に下げられるはずであるが FULL で実行しているプロセッサがあるため 1.404V から下げられないため消費電力は 2.28W となり，2 プロセッサ+1 プロセッサで実行した場合 2.35W に比べ殆ど電力は下がっていない．それに対し，4 プロセッサ+2 プロセッサで実行した場合電力状態は MID+LOW となり動作電圧が 1.196V に下がるため消費電力は 1.85W で最小となり，これは 2 プロセッサ+1 プロセッサで実行した場合 2.35W に比べ．21%の削減となった．

このような高負荷と超高負荷のアプリケーションで効率よく電力を削減する場合はよりいっそう，ソフトウェアによる並列処理と電力制御に加え，ハードウェアにプロセッサの電圧制御ドメインを区切ることが出来るような仕組みが必要である．

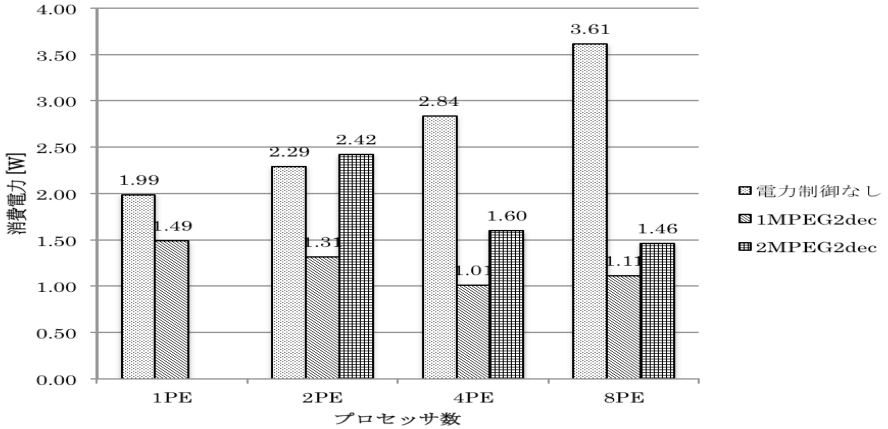

図 6 中負荷 MPEG2 デコーダ（352x128 ピクセル解像度）複数実行時の消費電力

5. まとめ

本論文では，OSCAR 自動並列化コンパイラの低消費電力制御を早稲田大学，ルネサスエレクトロニクス，日立製作所開発 RP2 上で複数アプリケーションの同時実行時の消費電力評価を行った．

複数アプリケーション同時実行において，軽負荷のアプリケーションを実行した際は，おなじ 8 コアで電力制御をしない場合に比べ，消費電力は 8 プロセッサで最大 68%の削減となった．複数同時実行の場合， 1 アプリ 1 コアで 1.89W かかる処理が 8AAC で 8 プロセッサの場合に平均電力は 1.01W となり，1 つの AAC あたりにかかる電力

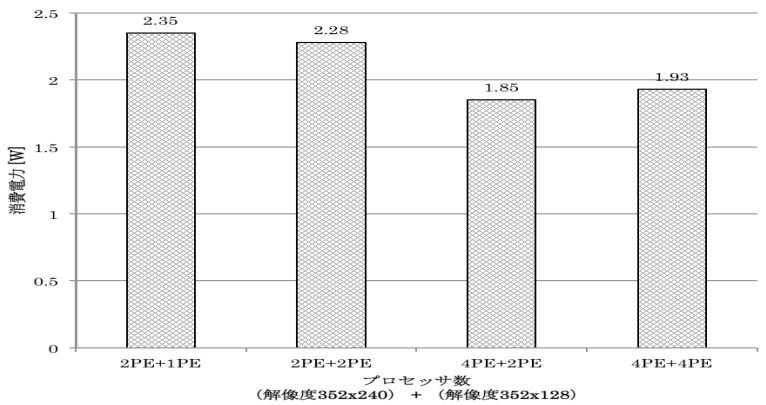

図 7 高負荷 MPEG2 デコーダと中負荷 MPEG2 デコーダの複数実行時の消費電力

は，平均 0.13W となり省電力効果が高い事が確かめられた．軽負荷のメディアアプリケーションの場合，並列化による電力削減より出来るだけ多く複数実行する方が効率よく処理が出来ると言える．中負荷のアプリケーションでは並列処理をすることで電力を削減する事ができ，プロセッサ数を倍に増やすごとに平均 30%の電力削減率となった．さらに 4AAC で各 AAC を 2 プロセッサで実行した時，1AAC あたりの消費電力は 0.55W となり 1 アプリあたりの省電力効果が得られる事が確かめられた．中負荷 MPEG2 デコードのアプリケーションにおいても，並列処理により電力を削減でき，2 つの MPEG2 デコーダを各 4 プロセッサ割り当てて実行した場合の消費電力（1.01W）は 1 つの MPEG2 デコーダを 1 プロセッサで実行した時の消費電力（1.49W）より低い電力で実行できる事がわかった．この時に 1 つの MPEG2 デコーダあたりの消費電力は 0.73W となり，1 つの MPEG2 デコーダを 1 プロセッサで実行した場合の消費電力 1.49W に比べ消費電力を 51%削減できることがわかった．ただし，このようなアプリケーションではハードウェアに電圧制御ドメインを区切る事が出来れば，さらに消費電力削減の余地があるという事もわかった．これは高負荷アプリケーションと中負荷アプリケーションを同時実行した場合，さらに顕著になるということもわかった．

以上により，軽負荷アプリケーションでは OSCAR コンパイラによって自動で電力制御を行う事，負荷のあるアプリケーションでは OSCAR コンパイラによって自動で並列処理を行いつつ電力制御を行う事が有効であるという事が確かめられた．さらに OSCAR コンパイラによる効率的な電力制御を行うにはハードウェアでプロセッサごとに電圧制御を行う事が必要であるという事がわかった．

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” で開発された OSCAR 自動並列化コンパイラ，情報家電用マルチコア RP2，および OSCAR API を利用して行われた．RP2 開発チーム及び OSCAR API 委員会の皆様に深く感謝致します．

参考文献

1) Dac Pham et al. The design and implementation of a first-generation cell processor. In Proceeding of the IEEE International Solid-State Circuits Conference, 2005.
2) J Cornish. Balanced energy optimization. In International Symposium on Low Power Electronics and Design, 2004.
3) ルネサスエレクトロニクス株式会社. NaviEngine. http://www2.renesas.com/automotive/ja/assp/naviengine.html.
4) A. Suga and S. Imai. Fr-v single-chip multicore processor:fr1000. Fujitsu Sci Tech J, 42(2):190–199, 2006.
5) 木村 浩三 et al. デジタル家電統合プラットフォーム uniphier におけるメディアプロセッサ. DA シンポジウム, 2005.
6) D.H. Albonesi et al: “Dynamically tuning processor resources with adaptive processing”, IEEE Computer(2003).
7) 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).
8) 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).
9) T. Kamei. Sh-x3 : An enhanced superh core for low-power mp systems. Fall Microprocessor Forum 2006, 2006.
10) 白子 準，吉田宗弘，押山直人，和田康孝，中野啓史，鹿野裕明，木村啓二，笠原博徳：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コンピューティングシステム， Vol.47, No.SIG12(ACS15), pp.147-158 (2006).
11) 間瀬正啓，中川 亮，大國直人，白子 準，木村啓二，笠原博徳：マルチコア上での OSCAR API を用いた並列化コンパイラによる低消費電力化手法，情報処理学会論文誌コンピューティングシステム， Vol.2, No.3, pp.96-106 (2009).
12) OSCAR API 仕様書, http://www.kasahara.cs.waseda.ac.jp/api/regist.html.
13) 石坂一久 et al. 共有メモリマルチプロセッサ上でのキャッシュ最適化を考慮した粗粒度タスク並列処理. 情報処理学会論文誌, 43(4), Apr. 2002.
14) Masayuki Ito, Toshihiro Hattori, Yutaka Yoshida, Kiyoshi Hayase, Tomoichi Hayashi, Osamu Nishii, Yoshihiko Yasu, Atsushi Hasegawa, Masashi Takada, Masaki Ito, Hiroyuki Mizuno, Kunio Uchiyama, Toshihiko Odaka, Jun Shirako, Masayoshi Mase, Keiji Kimura, Hironori Kasahara, "An 8640 MIPS SoC with Independent Power-off Control of 8 CPU and 8 RAMS by an Automatic Parallelizing Compiler", Proc. of IEEE International Solid State Circuits Conference (ISSCC2008), Feb. 2008.
15) 間瀬正啓 et al. マルチコアにおける Parallelizable C プログラムの自動並列化. 情処処理学会研究報告, 2009-ARC-184(15):1–10, 2009.
16) C. Lee et al. Mediabench: A tool for evaluating and synthesizing multimedia and communications systems. In 30th International Symposium on Microarchi- tecture (MICRO-30), Nov. 1997.