Vol.2010-ARC-187 No.2
Vol.2010-EMB-15 No.2
2010/1/28

# マイクロプロセッサ内の演算器に対する適応型パワーゲーティング

橋田達徳[†1]　小山慧[†1]　山本辰也[†1]　宇佐美公良[†1]
池淵大輔[†2]　天野英晴[†2]　並木美太郎[†3]　近藤正章[†4]
中村宏[†5]

本稿では break-even time（BET）の温度依存性に基づいた演算器の適応型細粒度パワーゲーティングについて述べる．BET は温度によって大きく変化するため，温度依存性を考慮することでより効果的なエネルギー削減ができる．本手法を演算器単位でスリープ制御をするマイクロプロセッサに適用すると，エネルギー削減効果は非適応型に比べて time-based アプローチでは最大 21%, history-based アプローチでは最大 18%向上することが分かった．また，パワーゲーティングを施さない場合と比べて 100℃で最大 11.8%まで消費エネルギーを低減できた．

# Adaptive Power Gating for Function Units in a Microprocessor

Tatsunori Hashida[†1]　Satoshi Koyama[†1]
Tatsuya Yamamoto[†1]　Kimiyoshi Usami[†1]
Daisuke Ikebuchi[†2]　Hideharu Amano[†2]
Mitaro Namiki[†3]　Masaaki Kondo[†4]
Hiroshi Nakamura[†5]

This paper describes adaptive fine-grain control to power gate function units based on temperature dependent break-even time (BET). Due to strong temperature dependency, dependency consideration provides effective energy saving. Applying this scheme to fine-grain power gated microprocessor which is controlled granularity of functional unit has archived energy savings by up to 21% in the time-based approach and by up to 18% in the history-based approach. Also this scheme has reduced the total energy of four function units up to 11.8% at 100°C as compared to the non-power-gating case.

## 1. はじめに

リーク電力は微細プロセスでのプロセッサ設計における主要な関心事の一つである．リーク電力はアクティブ状態でも全体の消費電力の主要な成分であるため，リーク電流の低減は不可欠である．既存のリーク電力低減技術の中では，パワーゲーティング（PG）は最も確実なアプローチの一つである．PG は回路の非アクティブ期間に回路への電力供給を遮断する技術である．近年では，インテルが Nehalem で PG の採用を発表している．モバイル用途向け SoC では，CPU や DSP の IP コアがアプリケーションにより動作時にも電源遮断される[1]．このようにこれまでの商業向けプロセッサや SoC では，動作時の PG の制御は主に IP コアレベルで実装されている．

一方，内部回路をより細かい粒度で動作時に電源遮断する，より積極的な技術が研究されている．文献[2]では，整数 ALU，浮動小数点 ALU，乗算器のような演算器がプロセッサコアの総消費エネルギーの主要部分を構成していると報告されている．文献[3]では，プロセッサ中の固定小数点ユニットや浮動小数点ユニットなどの演算器を電源遮断する技術が紹介されている．文献[4]では，命令をプリデコードすることによって，CPU コア中の ALU，シフタ，乗算器，除算器の電源遮断を制御する手法が提案されている．

これらの細粒度電源遮断（Fine-Grain Power Gating :FGPG）手法のエネルギー低減効果は，粗粒度な手法よりも回路の電源遮断によって生じるエネルギーオーバーヘッドに対して敏感になる．FGPG には 2 つの重要項目があり，それは損益分岐点(BET)と，BET に基づく PG 制御ポリシーである．BET は電源遮断によるリークエネルギーの低減量がエネルギーオーバーヘッドと等しくなる時間である．Hu らは BET の解析モデルを提案し[3]，そのモデルは様々な研究[5][6]で用いられている．しかし，これらの論文では，温度変化に対する BET の変化については考慮されていない．通常，動作中のチップ温度はチップ自身の放熱により上昇する．文献[7]では，温度の上昇にしたがって演算器の BET は著しく減少するというシミュレーション結果が報告されている．

BET に基づく PG 制御ポリシーはもう一つの重要な課題である．理想的には，BET より長いスリープ期間のスリープイベントでのみ PG を有効化するべきである．逆に，

†1 芝浦工業大学
Shibaura Institute of Technology
†2 慶應義塾大学
Keio University
†3 東京農工大学
Tokyo University of Agriculture and Technology
†4 電気通信大学
Tokyo University of Electro-Communications
†5 東京大学
The University of Tokyo

Vol.2010-ARC-187 No.2
Vol.2010-EMB-15 No.2
2010/1/28

BET より短いスリープイベントでは PG を無効化するべきである．しかし回路がスリープ状態になるとき，そのスリープが BET を超えて長期間続くかどうかを正確に予測することは不可能である．文献[3]では，スリープ期間に入ると，スリープ期間のサイクル数をカウントし，カウント数が BET に到達したら PG を有効化する Time-Based（TB）方式を提案している．この手法にはエネルギーの低減に失敗する極めて短いスリープを抑制するという利点がある．一方では，この手法にはスリープサイクルをカウントしている期間，潜在的なリーク電力削減効果を失うという欠点がある．文献[3]の中では，BET の温度依存性を積極的に利用するという考え方は紹介されていない．

本稿では，BET の温度依存性を消費エネルギーの低減化に利用するような PG 制御ポリシーを提案する．なお、本稿における実験では回路レベルシミュレーションによって取得した温度毎に異なる BET を用いる．

## 2. 提案する PG 制御ポリシー

提案する 4 つの PG 制御ポリシーについて述べる．PG はスリープサイクル数を検知することで制御されるため，この章では BET の代わりに BEC (break-even cycles) という表現を用いる．

### 2.1 Adaptive Time-Based（ATB）方式

最初に Adaptive Time-Based (ATB)方式を提案する．この手法ではオンチップ温度センサによってチップ温度をモニタし，検出された温度における BEC 値に基づいて適応的に PG 制御を変化させる．一方，オリジナルの Time-Based（TB）方式[3]では一つに固定された BEC（通常は室温での BEC）に基づいて PG は制御されていた．文献[7]で報告されている通り，BET（BEC も同様）は温度変化に対して著しく変化する．これは，室温での BEC までスリープ期間を数えることによって，高温時には潜在的なリーク電力削減効果を大きく失うことを意味する．ATB 方式は温度センサと温度依存の BEC に基づく PG 制御ポリシーを導入することでこの欠点を克服する．

温度変化は緩やかに起こることから，温度変化の検出は 10000 サイクル位の粒度で十分である[8]．この時間的粒度でオンチップ温度センサを読み，温度から BEC への変換テーブルを参照し，得られた BEC 値を BEC レジスタに格納する．なお，変換テーブルは，設計段階での回路シミュレーションによって予め用意しておく必要がある．

### 2.2 History-Based（HB）方式

次に，History-Based（HB）方式を提案する．この方式は，前回のスリープイベントのスリープ期間に基づいて次のスリープイベントが BEC を超えるかどうかを予測する方式である．仮に前回のスリープイベントが BEC を超えた場合，次のスリープイベントも BEC を超えると予測する．この予測に従って，スリープ期間に入ると即座に PS をオフする．反対に，前回のスリープイベントが BEC を超えなかった場合には次のスリープイベントも BEC より短くなると予測し，PG を行わない（PS はオンのままにする）．

全てのスリープイベントにおいてスリープ期間はカウンタによって数えられ，レジスタに記録される．レジスタのデータを BEC と比較し，BEC を超えているならば予測フラグレジスタに‘1’をセットする．そうでなければ‘0’にリセットする．過去の命令の履歴を記録して予測に利用することはコンピュータアーキテクチャの分野ではかなり一般的であり，分岐予測はもっとも良く知られる手法の一つである．分岐予測と比較して HB 方式が特徴的である点は，命令のアドレスを保持しないことである．代わりに，前回のスリープイベントが BEC を超えたか否かの情報だけを記録する．

TB 方式と比較して，HB 方式はスリープ期間に入ると直ちに PS をオフすることができるという利点がある．これにより，スリープ期間をカウントする間の潜在的リーク電力削減を逃すという TB 方式における欠点は克服される．代わりに，HB 方式の有効性は予測が当たるかどうかに依存する．これについては 4 章で議論する．

HB 方式で必要なハードウェアは予測フラグレジスタ（1bit のフリップフロップ）を除いて TB 方式と同様である．どちらの方式でも，スリープ期間のカウンタ，スリープイベントのスリープ期間を保持するレジスタ，BEC を記録するレジスタ，及び比較器を必要とする．なお，HB 方式では室温での BEC 値が常に使用される．

### 2.3 Adaptive History-Based（AHB）方式

三つ目に，Adaptive History-Based（AHB）方式を提案する．この方式では PG の有効化/無効化は HB 方式と同様に前回のスリープイベントに基づいて決定されるが，温度依存の BEC と比較することで予測が行われる点が HB 方式と異なる．温度は ATB 方式と同様にオンチップ温度センサを用いて取得する．温度が上昇するにつれて BEC は指数関数的に減少するため，高い温度では BEC を超えるようなスリープイベントは増加する．これは高温での PG 有効化の機会を増加させる結果をもたらす．BEC レジスタは温度をモニタした時だけ更新される．HB 方式と比較すると温度センサが必要になる．

### 2.4 リミッタ付き Adaptive History-Based（AHB-LM）方式

四つ目に，リミッタ付き Adaptive History-Based（AHB-LM）方式を提案する．先に述べた AHB 方式では，short-sleep だと予測された場合には次のスリープが起こると PG は無効化されるが，予測に反して実際のスリープが long-sleep であった場合，リーク電流は流れ続けて消費エネルギーを抑えることは出来ない．この無駄を最小化するために，AHB 方式にリミッタを導入する．スリープ期間に入るとスリープサイクルをカウントし始め，もし short-sleep という予測が外れた場合には，カウントサイクルが BEC に達した時点で，それ以上リーク電力削減効果を無駄にしないために PS をオフする．

AHB 方式にリミッタを導入するのに，追加ハードウェアは必要ない．これは AHB

Vol.2010-ARC-187 No.2
Vol.2010-EMB-15 No.2
2010/1/28

方式がすでにスリープ期間を数えるカウンタを備えているためである．AHB-LM 方式でも，カウンタと比較器は共通のものを使用することが出来る．

## 3. 実験手順

評価を行うためのプラットフォームとして，ALU，シフタ，乗算器，除算器に動的 PG を適用した MIPS R3000 ベースの CPU チップ（Geyser-1）を設計した．チップは富士通 65nm プロセスを用いてチップを実装し，レイアウトから抽出したデータに基づいて実験と評価を行った．

### 3.1 PG の制御方式

Geyser-1 では，命令毎に PG の制御が行われ，アクティブになる演算器はパイプライン中で検出される．パイプラインは Instruction Fetch（IF），Instruction Decode（ID），Execution（EX），Memory Access（MEM），Write Back（WB）の 5 ステージから成る．Geyser-1 では演算器が使用されない間はデフォルトでスリープするというポリシーが採用されており，ここでは WIPS（Whenever Idle Put to Sleep）と呼ぶことにする．ALU を除く演算器は実行が終わると直ちにパワーオフ状態に入る．演算器のスリープ信号はチップ上のスリープコントローラで生成される．使用される演算器は ID ステージで決定されるが，パワーオフ状態にある演算器のウェイクアップには時間がかかるため，EX ステージで実行が始まるタイミングには間に合わない．3.2 節で述べる実装方式を用いて，動作周波数 200MHz を目標にして 1 サイクル以内の復帰時間を達成した．しかし，まだ ID ステージと EX ステージの間で 1 サイクルのストールが必要であり性能劣化につながってしまう．これを避けるため，アクティブになる演算器を IF ステージで検出して ID ステージで復帰させるプリウェイクアップ手法を導入した[4]．アクティブになる演算器の検出用の非常にシンプルなデコーダを IF ステージ設ける．

### 3.2 PG の実装方式

我々は文献[5]で提案されている Locally-Shared Virtual ground (LSV)方式を用いて演算器の PG を実装した．この方式では，全ての演算器は小さなパワードメインに分割される．パワードメイン内の論理ゲートはローカルな仮想グランド (VGND)線を介して nMOS フッター型パワースイッチ（PS）に接続されている．各 PS はローカルなパワードメイン内でのみ共有されている．演算器内の各 PS は一つのスリープ制御信号によって制御されるが，PS サイズはそれぞれのパワードメイン内で個別に調整される．さらに，この方式では真のグランド線が PG される論理セル内に存在しているため，フリップフロップ，クロックバッファや PS ドライバ（PSD）などの PG されないセルをロウ内のどこにでも置くことができる．

LSV 構造を実装するために，我々は以下のような設計フローを開発した．Low-Vth スタンダードセルライブラリを用いて RTL からネットリストを合成する．アイソレーションセルは信号のフローティングを避けるために PG 部分とそうでない部分との境界に挿入される．配置を実行した後に，論理セルを同サイズの PG セルに置き換える．PG セルはオリジナルのスタンダードセルと同じ論理の low-Vth セルであるが VGND ピンを備えており，PG セル内の nMOS トランジスタのソースは真のグランドの代わりに VGND ピンに接続されている．適切な大きさに分割された PS セルと PSD セルを挿入した後，最終的には配線工程でローカルな VGND 線と信号線で接続される．復帰時のラッシュカレントを抑制するために，PSD はそれぞれのパワードメインがパワーオンし始める時間が重ならないように調整されている．

チップのレイアウトを図 1 に示す．演算器に PG を適用した結果を表 1 にまとめた．PS とアイソレーションセルの面積オーバーヘッドは約 5%～8%であった．

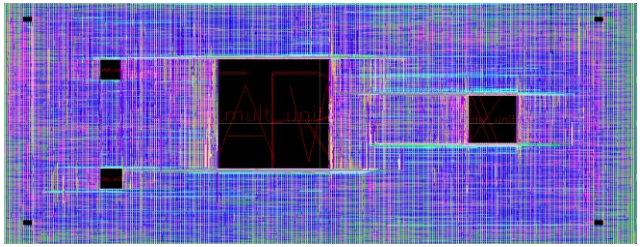


図 1 Geyser-1 チップのレイアウト
中央周辺にあるブラックボックスは PG された演算器を示す．

表 1 演算器の PG 実装結果．

| | ALU | Shifter | Multiplier | Divider |
|---|---|---|---|---|
| # Logic Cells | 1690 | 1628 | 9252 | 14837 |
| # PS Cells | 106 | 100 | 648 | 704 |
| # Isolation Cells | 33 | 32 | 64 | 64 |
| # Total cells | 1829 | 1760 | 9964 | 15605 |

### 3.3 評価方法

スリープ動作の間のそれぞれの演算器の消費エネルギーを解析するために Hsim を用いて回路レベルシミュレーションを行った．また，PG 制御ポリシーの評価の際に CPU 全体の RTL モデルの Verilog シミュレーションを行った．Verilog シミュレーションは 3 つのアプリケーションプログラムについて実行した：(i)マルチメディア処理で使用される DCT（離散コサイン変換），(ii)MiBench に入っているクイックソートプログラム（QSORT），(iii)JPEG エンコードを使用した．実行の際，現実的な時間内に実行するために入力データサイズを制限した．DCT は 16x16DCT, QSORT は要素数 100,

Vol.2010-ARC-187 No.2
Vol.2010-EMB-15 No.2
2010/1/28

JPEGSMALL を想定した．Verilog シミュレーションの総クロックサイクルは，DCT で 43,232 サイクル, QSORT で 188,569 サイクル, JPEG では 2,940,467 サイクルであった．

## 4. 実験結果

### 4.1 BET の評価

各演算器における温度ごとの BET を評価するために, 各演算器がスリープ開始から再びアクティブ状態に変化するまでに必要となるエネルギーを分類し，それらについて Hsim を用いた回路レベルシミュレーションにより解析した．なお，PS をオフすることを‘sleep-in’として，PS をオンすることを‘wakeup’と定義する．スリープ期間は sleep-in の開始から wakeup の終了までの連続した期間として定義される．スリープ期間にはエネルギーは次の 3 つの要素で消費される．

(1) リークエネルギー（$E_{Leak\_sleep}$）は，PG 回路の状態‘0’を出力するノードと VGND 線に流れ込むリーク電流によってスリープ期間中に消費される．
(2) スイッチングエネルギー（$E_{PS}$）は，sleep-in と wakeup 時に起こる PSD ネットワークのスイッチングで消費される．
(3) グリッチエネルギー（$E_{Glitch}$）は，$E_{Leak\_sleep}$ によって充電された電荷が原因で wakeup 時に発生するスイッチングによって消費される．

スリープ期間中に電源から供給された電流を観測することで $E_{Leak\_sleep}$ を取得した．$E_{PS}$ に関しては, シミュレーションの中で PSD ネットワークに別電源を定義し, sleep-in と wakeup の 1 サイクルの間の電流を観測することで $E_{PS}$ を取得した．これは我々の設計において PSD ネットワークの切り替えが 1 サイクル以内に完了するためである．$E_{Glitch}$ は wakeup 時に PG 回路に供給される電流を観測することで取得した．3.1 節で述べたように wakeup は 1 サイクルで完了するため，wakeup の 1 サイクルの間を観測する．スリープ期間を変化させながら $E_{Leak\_sleep}$，$E_{PS}$，$E_{Glitch}$ を解析した．

乗算器の 65℃における結果を図 2 に示す．PG されていない場合の消費エネルギー $E_{non\text{-}PG}$ も図に示す. $E_{non\text{-}PG}$ と $E_{Leak\_sleep}$, $E_{PS}$, $E_{Glitch}$ の合計とが交わる点が break-even point（BEP）である．BEP までのスリープ時間の長さが BET である．もし総消費エネルギーに $E_{Glitch}$ を含めない場合には BET は乗算器において 180ns（36 サイクル）から 80ns（16 サイクル）まで減少することを示している．他の演算器でも同様の実験を行い，温度毎の BEC を表 2 にまとめた．

表 2 各演算器における温度ごとの BEC.

| | ALU | | | Shifter | | | Multiplier | | | Divider | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Temperature[°C] | 25 | 65 | 100 | 25 | 65 | 100 | 25 | 65 | 100 | 25 | 65 | 100 |
| BEC | 144 | 42 | 18 | 146 | 42 | 18 | 122 | 36 | 16 | 76 | 24 | 12 |

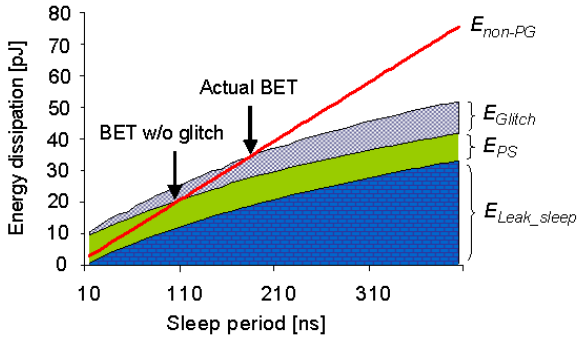



図 2 乗算器の 65℃での BET.

### 4.2 PG 制御ポリシーの評価

我々は本稿で提案する PG 制御ポリシーの有効性を消費エネルギーの観点から以下の方法によって評価した．比較対象としての従来方式は，Time-Based（TB）方式と Whenever-Idle-Put-to-Sleep（WIPS）方式とする．提案する 4 つの方式（ATB, HB, AHB, AHB-LM）を従来方式と比較した．Geyser-1 の 4 つの演算器の消費エネルギーについて，各 PG 制御ポリシーが適用された場合を想定して以下のようにして解析した．なお，各消費エネルギーは PG が適用されていない（non-PG）時のリークエネルギーを基準として正規化している．初めに各演算器のスリープ期間毎の消費エネルギーをスリープ期間と温度を変えてキャラクタライズする．sleep-in と wakeup で発生するエネルギーオーバーヘッドもこの消費エネルギーに含まれている．次に，アプリケーションプログラムを実行している間の各演算器のスリープの振る舞いを Verilog シミュレーションから取得した．最終的にこれらの結果を結合して消費エネルギーを求めた，結果を図 3 の(a) - (i)にまとめた．

まず従来方式である TB 方式と WIPS 方式の消費エネルギー低減効果をみていく．図 3(a)と(b)に示されているように，演算器の総消費エネルギーは DCT，QSORT，JPEG のすべてにおいて，TB 方式では低減する．一方，WIPS 方式では DCT において総エネルギーは増加する．この理由を説明するために次のように用語を定義する．BEC よりもスリープ期間の短いスリープイベントを‘short-sleep’と定義し、逆に BEC 以上のスリープ期間のスリープイベントを‘long-sleep’と定義する．プログラム中のスリープイベントを分析した結果，DCT において ALU，シフタ，乗算器では short-sleep が 99%を占めることが分かった．WIPS 方式では回路は全てのスリープイベントに対してパワーオフ動作をするため，sleep-in と wakeup のエネルギーオーバーヘッドによ

Vol.2010-ARC-187 No.2
Vol.2010-EMB-15 No.2
2010/1/28

って総エネルギーが増加してしまう．

次に、TB 方式と ATB 方式を比較することで，TB 方式に適応型制御を導入することの有効性について議論する．DCT において ATB 方式は TB 方式に対して 65℃で 5%，100℃で 17%の消費エネルギーを削減できた．QSORT では ATB 方式は TB 方式に対して 65℃で 10%，100℃で 21%の消費エネルギーを削減できた．JPEG では ATB 方式は TB 方式に対して 65℃で 2%，100℃で 10%の消費エネルギーを削減できた．演算器でいえば乗算器とシフタにおいて適応型制御の有効性が確認できる．一方，ALU では TB 方式に適応型制御を導入した効果は得られなかった．この理由はプログラム中のスリープイベントの様子から説明することができる．表 3 に分析の結果をまとめた．

表 3 スリープイベントの解析結果．

| | ALU | | | Shifter | | | Multiplier | | | Divider | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Temperature [°C] | 25 | 65 | 100 | 25 | 65 | 100 | 25 | 65 | 100 | 25 | 65 | 100 |
| BEC | 144 | 42 | 18 | 146 | 42 | 18 | 122 | 36 | 16 | 76 | 24 | 12 |
| DCT | | | | | | | | | | | | |
| Total sleep events | 5532 | | | 1443 | | | 2544 | | | 1 | | |
| Long-sleep | 0 | 0 | 0 | 6 | 207 | 435 | 6 | 192 | 240 | 1 | 1 | 1 |
| HIT Rate (long) | 0% | 0% | 0% | 0% | 7% | 23% | 0% | 49% | 60% | 0% | 0% | 0% |
| Short-sleep | 5532 | 5532 | 5532 | 1437 | 1236 | 1008 | 2538 | 2352 | 2304 | 0 | 0 | 0 |
| HIT Rate (short) | 100% | 100% | 100% | 100% | 84% | 67% | 100% | 96% | 96% | 0% | 0% | 0% |
| QSORT | | | | | | | | | | | | |
| Total sleep events | 18937 | | | 234 | | | 108 | | | 40 | | |
| Long-sleep | 0 | 0 | 0 | 210 | 210 | 213 | 85 | 85 | 85 | 40 | 40 | 40 |
| HIT Rate (long) | 0% | 0% | 0% | 90% | 90% | 90% | 75% | 75% | 75% | 98% | 98% | 98% |
| Short-sleep | 18937 | 18937 | 18937 | 24 | 24 | 21 | 23 | 23 | 23 | 0 | 0 | 0 |
| HIT Rate (short) | 100% | 100% | 100% | 13% | 13% | 0% | 13% | 13% | 13% | 0% | 0% | 0% |
| JPEG | | | | | | | | | | | | |
| Total sleep events | 305647 | | | 124466 | | | 59953 | | | 1853 | | |
| Long-sleep | 0 | 0 | 0 | 156 | 5457 | 44261 | 193 | 3505 | 4657 | 660 | 1853 | 1853 |
| HIT Rate (long) | 0% | 0% | 0% | 3% | 6% | 33% | 25% | 34% | 51% | 30% | 100% | 100% |
| Short-sleep | 305647 | 305647 | 305647 | 124310 | 119009 | 80205 | 59760 | 56448 | 55296 | 1193 | 0 | 0 |
| HIT Rate (short) | 100% | 100% | 100% | 100% | 96% | 63% | 100% | 96% | 96% | 62% | 0% | 0% |

表 3 に示された通り，DCT，QSORT，JPEG における主な long-sleep イベントは乗算器，シフタ，除算器に存在する．これらの long-sleep イベントでは，スリープ検出後から出来るだけ早く PS をオフすることで無駄になるリーク電力を抑えることができる．一つの BEC（25℃での BEC）をすべての温度で用いる TB 方式と比べると，ATB 方式は高温時により短い BEC を制御に使用する．これは TB 方式よりも早く PS をオフすることを可能にし，結果として無駄なリーク電力を削減することができる．除算器における QSORT では同じ理由で ATB 方式のほうが TB 方式よりも低減効果が高い．一方, ALU では ATB 方式, TB 方式ともにエネルギーを削減することが出来なかった．これは高温時の BEC よりも全てのスリープイベントが短いためである．

History-Based アプローチにおいてもまた, 適用型制御を導入する有効性がみられた．図 3 の(b)と(c)に示すように, DCT において AHB 方式は HB 方式に対して 65℃で 7%，100℃で 18%の消費エネルギーを削減できた．JPEG において AHB 方式は HB 方式に対して 65℃で 40%，100℃で 53%の消費エネルギーを削減できた．Time-Based アプローチ（TB，ATB）とは対照的に，History-Based アプローチではエネルギー削減率は予測のヒット率に影響される．long-sleep に対するヒット率とは、次のスリープイベントを long-sleep と予測して当たった割合である．short-sleep に対するヒット率も同様に定義する．これらのヒット率を表 3 にまとめた．DCT，JPEG では，高温になると乗算器のヒット率が上昇していく．これは HB 方式より AHB 方式がよりエネルギーを削減できるという結果をもたらす．DCT，JPEG とは対照的に，QSORT での AHB 方式は 65℃，100℃のどちらでもエネルギー削減率は向上しない．これは高温時でもシフタ，乗算器，除算器の long-sleep のヒット率が向上しないためである．

我々は最終的にエネルギー削減効果の比較を ATB 方式と AHB 方式で行った．DCT では，AHB 方式は 65℃と 100℃の両方で ATB 方式よりも多くのエネルギーを削減しているのに対して, QSORT では AHB 方式のエネルギー削減効果が ATB 方式よりも小さい．我々は AHB 方式のエネルギー削減効果を悪化させる要素を詳細に調査した．表 3 に示されているように，QSORT では DCT と比べて short-sleep のヒット率がシフタ，乗算器および除算器で著しく低い．short-sleep の予測が外れたとき，実際のスリープが long-sleep であるにもかかわらず PG は機能せず（つまり PS はオンのまま），long-sleep でのエネルギー削減機会が失われている．2.4 節で説明した AHB-LM 方式ではこの問題を解決している．その結果 QSORT において，AHB-LM 方式は ATB 方式よりも 65℃で 4%，100℃で 3%の消費エネルギーを削減することができた．

AHB-LM 方式はこの研究の中で最も良いエネルギー削減効果を達成した．Non-PG と比べて AHB-LM 方式は，4 つの演算器の総消費エネルギーを 100℃のとき，DCT において 41.4%, QSORT では 11.8%, JPEG では 23.5%まで削減することができた．また，毎回のスリープイベントに対して予測が常に当たるオラクルケースでも実験した．結果は，オラクルにおいて 100℃のとき，DCT では 39.8%，QSORT では 11.8%，JPEG では 22.7%までエネルギーは削減されると示している．したがって，AHB-LM 方式によって達成されたエネルギー削減効果はオラクルのそれと非常に近いことが分かる．この事実は，オーバーヘッドの少ない小さな改良によって，非常に理想的なエネルギー削減効果を達成できることを意味している．

Vol.2010-ARC-187 No.2
Vol.2010-EMB-15 No.2
2010/1/28

## 5. 結論

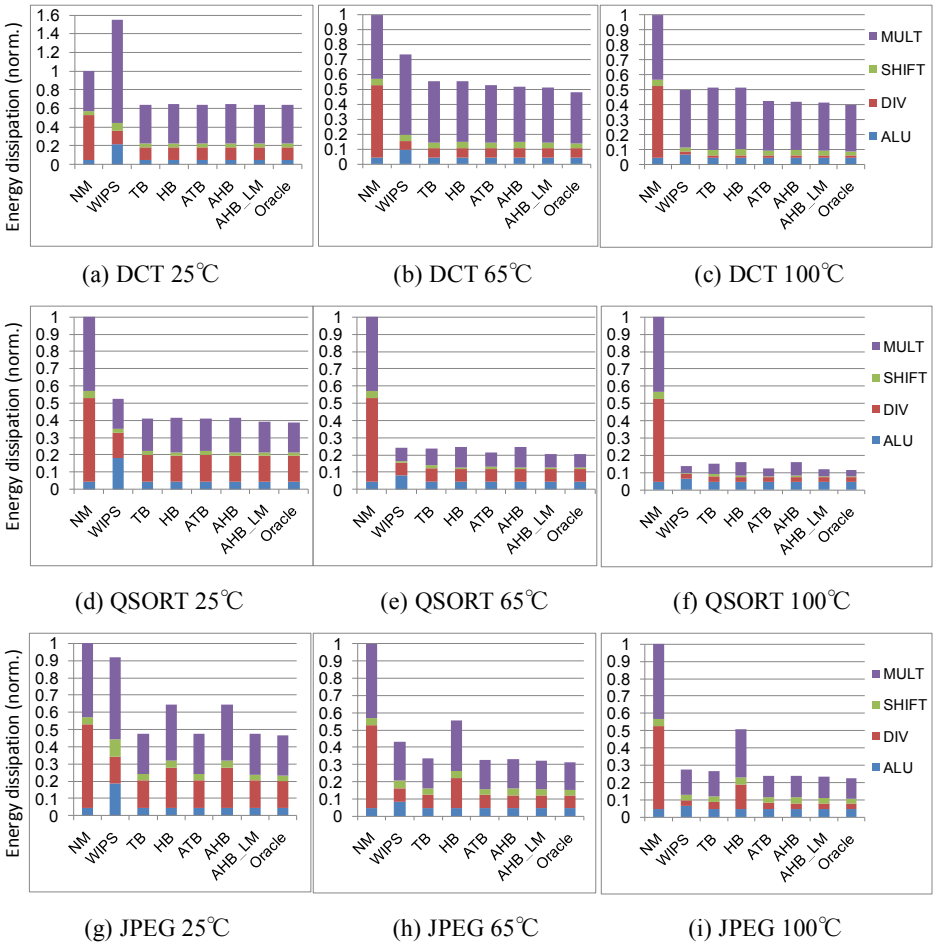
我々は演算器の温度依存の break-even time（BET）に基づいた適応型パワーゲーティング手法として 4 つの PG 制御ポリシーを提案し，それらの消費エネルギーについて比較評価した．温度依存の BET は回路レベルシミュレーションによって値を取得した．PG 制御ポリシーについては，それらの中で，リミッタ付き Adaptive History-Based（AHB-LM）方式が最も大きなエネルギー削減効果を示した．

今後の課題として，我々はさらにエネルギーを削減するために，グリッチによる消費エネルギーを削減する設計技術について研究する．

**謝辞** 本研究は科学技術振興機構（JST）の戦略的創造研究推進事業（CREST）における研究領域「情報システムの超低消費電力化を目指した技術革新と統合化技術」の研究課題「革新的電源制御による次世代超低電力高性能システム LSI の研究」による．本研究は東京大学大規模集積システム設計教育研究センターを通し，シノプシス株式会社・日本ケイデンス株式会社の協力で行われた．

## 参考文献

1) T. Hattori, et. al., “Hierarchical power distribution and power management scheme for a single chip mobile processor,” Proc. of ACM/IEEE Design Automation Conference, pp.292-295, 2006
2) J. Deeney, “Reducing power in high-performance microprocessors,” In International Symposium on Microelectronics, 2002.
3) Z. Hu, et. al., “Microarchitectural techniques for power gating of execution units,” Proc. ISLPED’04, pp.32-37, 2004.
4) N. Seki, et. al., “A Fine-grain Dynamic Sleep Control Scheme in MIPS R3000”, Proc. ICCD’08, Oct. 2008.
5) K. Usami and N. Ohkubo, “A design approach for fine-grained run-time power gating using locally extracted sleep signals,” Proc. ICCD'06, pp.155-161, Oct. 2006.
6) H. Matsutani, et. al., “Run-time power gating of on-chip routers using look-ahead routing,” Design Automation Conference, Proc. of ASP-DAC'08, pp.55-60, March, 2008.
7) K. Usami et. al., “Design and implementation of fine-grain power gating with ground bounce suppression,” Proc. VLSI Design 2009, pp. 381-386, Jan. 2009.
8) D. Kannan, et. al., “Power Reduction of Functional Units considering Temperature and Process Variations, ” Int. Comf. on VLSI Design, pp.533-539, Jan. 2008.




(a) DCT 25℃ (b) DCT 65℃ (c) DCT 100℃

(d) QSORT 25℃ (e) QSORT 65℃ (f) QSORT 100℃

(g) JPEG 25℃ (h) JPEG 65℃ (i) JPEG 100℃

図 3 各 PG 制御ポリシーを適用した場合の消費エネルギー