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

2008－ARC－176 (13)
2008－EMB－ 7 (13)
2008／1／16

MIPS R3000プロセッサにおける細粒度動的スリープ制御の実装と評価

関 直臣[†1] Lei Zhao[†1] 徐 慧[†1]
池淵大輔[†1] 小島 悠[†1] 長谷川揚平[†1]
天野英晴[†1] 香嶋俊裕[†2] 武田清大[†2]
白井利明[†2] 中田光貴[†2] 宇佐美公良[†2]
砂田徹也[†3] 金井 遵[†3] 並木美太郎[†3]
近藤正章[†4] 中村 宏[†4]

本報告はパワーゲーティング (PG) を使った演算器レベルでの動的スリープ制御による消費電力削減機構の実装及び評価を行う。MIPS R3000 の ALU からシフタ、乗算器、除算器を分離し、それぞれを動的にパワーゲーティングを行う。省電力化を施した R3000 コアと 16KB の L1 キャッシュ、TLB をあわせて、ASPLA 90nm で試作チップ Geyser-0 としてテープアウトした。Geyser-0 の性能、電力と面積をポストレイアウト後のシミュレーションにより評価した。この結果、4 種類のアプケーションについてリーク電力は平均約 47%減らすことができた。一方、スリープ制御の実装によって生じたエリアオーバーヘッドは 41%であった。

A Fine Grain Dynamic Sleep Control Scheme in MIPS R3000

Naomi Seki,[†1] Lei Zhao,[†1] Jo Kei,[†1] Daisuke Ikebuchi,[†1]
Yu Kojima,[†1] Yohei Hasegawa,[†1] Hideharu Amano,[†1]
Toshihiro Kashima,[†2] Seidai Takeda,[†2] Toshiaki Shirai,[†2]
Mitustaka Nakata,[†2] Kimiyoshi Usami,[†2] Tetsuya Sunata,[†3]
Jun Kanai,[†3] Mitaro Namiki,[†3] Masaaki Kondo[†4]
and Hiroshi Nakamura[†4]

A processor with a novel fine grain power gating technique to save leakage power was designed and implemented. An execution unit is divided into four small units: multiplier, divider, shifter and others, the power of each unit is cut-off dynamically based on the operation. We tape-outed the prototype chip Geyser-0, which provides R3000 Core with the power reduction technique, 16KB caches and TLB using 90nm CMOS technology. Evaluation results of four benchmark programs for embedded application show that 47% leakage power are reduced in avarage with 41% area overhead.

1. はじめに

90nm 世代以降のチップでは、リーク電力の増大が顕著である。「革新的電源制御による超低消費電力高性能システム LSI の構想」プロジェクト[1] は、回路技術、アーキテクチャ、システムソフトウェアの各階層が連携、協力し、革新的な電力制御を実現することで高性能システム LSI の消費電力を格段に低下させることを狙っている。このプロジェクトの中でも、プロセッサのリーク電力の削減技術の確立は最も大きなテーマの一つである。

リーク電力の削減する技術には、パワーゲーティング、Selective MTCMOS、基板バイアスなどが提案され、一部は実際に利用されている。この中で、我々は、ソフトウェア、アーキテクチャとの連携によりスリープ期間を制御することで大きな効果が得られることが期待されるパワーゲーティングに着目した。

パワーゲーティングは回路の一部に対して電源供給を断つことにより、リーク電力を削減する手法だが、モード遷移レイテンシによる性能ロスやパワースイッチのエリアオーバーヘッドなどの問題点がある。このため従来、パワーゲーティングは、マルチコアシステムにおけるコア単位等、プロセッサ全体に相当するサイズについて、長いタイムスパンで行われてきた[5][6]。

しかし昨今、パワースイッチのウェイクアップタイムの高速化やレイアウト時におけるパワースイッチの挿入の最適化など回路レベルの技術の進展[4] により、パワーゲーティングを用いるためのハードルは低くなってきている。プロジェクトの一環として、芝浦工大では乗算器を分割して演算データに応じたパワーゲーティングを施すことが可能なチップ Pinnacle[3] を開発している。

そこで、本研究では、パワーゲーティングを CPU 内部の

†1 慶應義塾大学
Keio University
†2 芝浦工業大学
Shibaura Institute of Technology
†3 東京農工大学
Tokyo University of Agriculture and Technology
†4 東京大学
The University of Tokyo

比較的小さい単位に対して動的に運用する。例えば演算器の中でも乗算回路は面積が大きく、リーク電力も大きいにも関わらず、一般的なプログラムではさほど使用頻度は高くない。このような論理ブロックを細かくスリープさせれば全体として消費電力を大きく削減できると予想される。このように分離した演算器部分をユニット呼び、これらを実行中の命令に対応して、動的にパワーゲーティングを行う。ここでは、演算器以下のレベルでのパワーゲーティングを、従来のプロセッサ全体などのパワーゲーティングと区別して細粒度パワーゲーティングと呼ぶ[10]。我々は、細粒度パワーゲーティングの実装についての検討を進め、その結果を集約して、Geyser-0 を VDEC の ASLPLA 90nm プロセスを利用して試作した。

本稿ではこの Geyser-0 の実装および設計実装段階での評価について報告する。

2. Geyser-0 の設計

2.1 細粒度パワーゲーティング手法

図 1 に今回用いた細粒度動的パワーゲーティングの概要を示す。

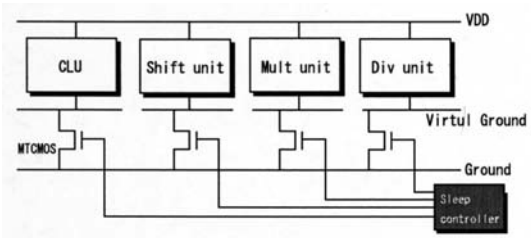

図 1 パワーゲーティングの概要

CPU の中では、命令や CPU の状態によっては、全く動作する必要のない部分がある。これらを分離してユニットと呼ばれる単位を構成し、それぞれにフッタ型のパワーゲーティングを適用する。このゲートを個別にオフすることで、そのユニットのパワーをオフ、すなわちスリープさせることができる。図中ではパワーゲーティング用のトランジスタは 1 個しか書かれていないが、ユニットの大きさに応じて数が調整される。

パワーゲーティングはモード遷移に一定のエネルギーが必要である。また、スリープモードに入っても消費電力を最小にするまでには、一定の時間がかかる。このため、モード遷移の頻度が大きいと、逆にエネルギーロスによって消費電力が大きくなってしまう可能性がある。一定期間経過後、エネルギーロスよりも削減したエネルギーの方が大きくなる地点をブレイクイーブンポイントと呼ぶ。ブレイクイーブンポイントを超えてスリープを継続でれば消費電力を削減できていることになる。

ユニットの選択とこれらに対するパワーゲーティングの制御は、できる限りブレイクイーブンポイントのより長い時間スリープできるようにする必要がある。

2.2 Geyser-0 のスリープ制御

Geyser-0 では、MIPS R3000 ベースの CPU コアの演算ユニットおよびコプロセッサ CP0 にのみ細粒度動的パワーゲーティングを実装した。これは予備評価により演算ユニットが CPU コアの 6 割の面積を占め、またスリープ自体の可能性が大きいことがわかったためである[10]。CPU コアの他の部分、例えばレジスタファイルなどはパワーゲーティングが有効となる可能性はあるものの、データ保持等の問題点から今回は実装を見送った。また、キャッシュおよび TLB の CPU 全体の中で占める面積は大きいが、同様にデータ保持等の問題があり、今回は細粒度動的パワーゲーティングの対象を CPU コアに絞ることとした。

R3000 は代表的な RISC 型の 32 ビットプロセッサで、32 個の汎用レジスタと 5 段構成のパイプラインステージで命令を処理する。まず、EX（実行）ステージから、シフタ、乗算器、除算器をユニットとして分離し、これらのユニットに関して、命令に応じた動的なスリープ制御を行う。さらに、コプロセッサ CP0 をこれに加えた。CP0 は OS とプロセッサのインタフェースの役割を持つが、比較的面積が大きいわりに使用頻度が低いため、パワーゲーティングが有効である可能性が大きい。まとめると、動的スリープ制御の対象とスリープする命令は以下のようになる。

- CLU (Common arithmetic and Logic Unit)
  加減算など一般的な演算を行うユニット。分岐命令、NOP 命令、またメモリアクセスでアドレス計算が不要な場合などでもスリープする。
- Shift unit
  シフト演算を行うユニット。
- Mult unit
  乗算を行うユニット。乗算は 4 サイクルかかる。両方のオペランドの上位 16 ビットが 0 の場合は、上位ビットを演算する部分は個別にスリープする。
- Div unit
  除算を行うユニット。除算は 10 サイクルかかる。
- CP0
  TLB の制御や割込み、例外処理など OS とのインターフェースとしての役割を担う。プロセッサがユーザーモードのときは使われないので、長期間スリープさせることができる。

パワーゲートの制御は、スリープコントローラからの信号によって行われる。図 2 にスリープコントローラとパイプラインの関係を示す。

スリープコントローラは、命令以外でもキャッシュミスや演算結果待のストールなども考慮して各ユニットのスリープとウェイクアップを制御する。基本的な動作は次の通りである。

CLU を除いた各ユニットは演算完了後、すぐに自動的にスリープモードに遷移する。CP0 もプロセッサがユーザーモードになると自動的にスリープモードになる。命令フェッチが IF（命令フェッチ）ステージで行われると、スリープコントローラはこの命令を先見して、どのユニットが使われるかを判断する。これは、ID（命令デコード）ステージで命令を判定すると、ウェイクアップが間に合わず、EX ステージで演算の開始が間に合わない可能性があるためである。スリープコントローラではこのための専用の簡易命令デコーダ

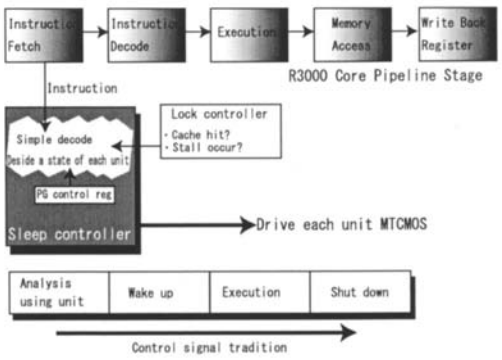

図 2 実行時スリープ制御

を持っており、高速に次の命令で使うユニットを特定して、ウェイクアップ信号を送る。このことにより、ID ステージを命令が通過している間 (Geyser-0 の場合 5ns) で、各ユニットはウェイクアップを行うことができる。

キャッシュミスや他の演算結果待ちの信号を検出すると、場合に応じて CLU を含めて各ユニットに対してスリープを行い、ミスおよびストールの遅延を考慮した時期にウェイクアップ信号を発生する。

2.2.1 PG 制御命令

さて、演算器の中で利用頻度が高いものは、ブレイクイーブンポイントに達せずにモード遷移が頻発して、エネルギーロスを大きくしてしまう可能がある。各ユニットの実際の利用状況は動作時にならないと正確には分からないが、コンパイル時にはある程度の予想は可能である。特にエネルギーロスが大きくなるようなモード遷移が発生する部分では、スリープに移行せず、アクティブな状態をキープする方がエネルギー効率がよい。

そこで、このような状況がコンパイル時に明らかな場合のために PG 制御情報を付加した命令を用意した。図 3 は PG 制御情報を付加した命令の構造を示している。

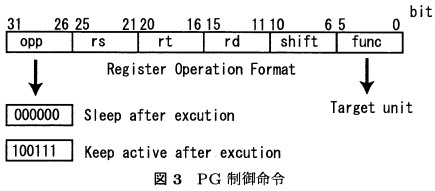

図 3 PG 制御命令

R3000 の ISA でレジスタ演算命令は、仕様上上位 6 ビット (ROP ビット) が 0 になり、下位 6 ビットで演算の種類を表す。この ROP ビットを 100111(2) に設定してやると、演算後に利用した演算ユニットの自動スリープをキャンセルする。例えば、シフト演算を行う命令で ROP ビットが 100111(2) であれば、演算終了後 Shift unit をアクティブのままキープする。アクティブ維持は次のシフト命令が到着するまで継続される。

このようにコンパイル時にエネルギー効率の悪い命令列を発見したら、この ROP ビットの付加情報を利用して、スリープをキャンセルするようにすることでエネルギーロスを小さくすることができる。

2.2.2 PG 制御レジスタ

PG 制御命令と同様に、CP0 内の特殊レジスタを用いてパワーゲーティングを制御することができる。この PG 制御用レジスタはプロセッサがカーネル (特権) モード時に書き込み可能で、OS や割込みによる制御を想定している。図 4 は PG 制御用レジスタの概要である。

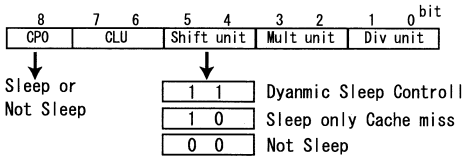

図 4 PG 制御用レジスタ

このレジスタの各ビットは各ユニットがスリープをキャンセルするかどうかのフラグとなっている。また、スリープをキャンセルする場合でもキャッシュミス時にスリープを行うかどうかも設定できる。そのため、1 つのユニットにつき 2 ビット必要である。ただし、CP0 に関してはユーザーモード時にスリープ行うかどうかを決めれば良いため、ここでは 1 ビットとした。よって PG 制御レジスタは合計 9 ビットで構成される。

パワーゲーティングのブレイクイーブンポイントは、回路構成の他に温度に影響受ける。例えば、外気温を検知して OS が適切に PG 制御レジスタを設定すればパワーゲーティングによるエネルギーロスを少なくできる。

2.3 キャッシュと TLB

Geyser-0 に搭載されるキャッシュは L1 キャッシュのみで、命令キャッシュ、データキャッシュともに 8KB の 2Way セットアソシアティブ方式で 64Byte のラインが 64 本あり合計 4KB を 1Way として構成した。データキャッシュはライトバック方式を採用している。先に述べた通り、今回のパワーゲーティングの対象ではない。

TLB は仮想ページアドレスを物理ページアドレスに変換する。ページアドレス変換用のテーブルのエントリ数は 16 個とした。

図 5 はキャッシュアクセスの流れを示す。仮想アドレスの上位 20bit を仮想ページアドレスとし、下位 12bit をページ内オフセットとして一階層のページアドレス変換を行う。

ページサイズと 1Way のサイズを同じにしたので、キャッシュを引くと同時に TLB で変換を行うことができる。

2.4 Geyser-0 の実装

2.4.1 概　　要

Geyser-0 は動作周波数を 200MHz を想定して設計している。ASPLA 90nm のプロセスルールを用いて 2.5mm × 5mm サイズのチップを試作した。キャッシュメモリ及びそのタグメモリは、VDEC が供給する ARM の RAM を用いた。I/O はピン数の関係上、出力するデータとアドレスを多重化した。このための簡易な MMU をチップ上搭載してメモリ

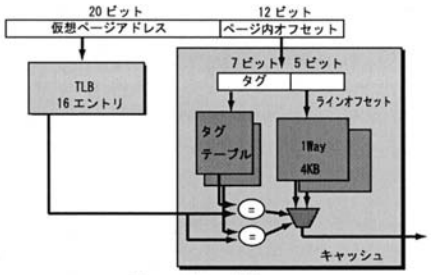

図 5 アドレス変換フロー

アクセスを管理し、評価ボード側には FPGA による MMU を搭載し柔軟に対応する。

全体の構成は図 6 のようにした。

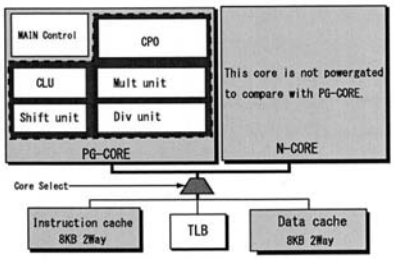

図 6 Geyser-0 の構成

Geyser-0 は、パワーゲーティングを施したコア (PG-CORE) と、施してない比較用のコア (N-CORE) の 2 つが、TLB およびキャッシュに対して切り替え式で接続する構成を持つ。点線で囲った部分が細粒度パワーゲーティングを行う部分である。各部は、独立した電源を持っており、別々に電流を測定できるようになっている。

図中に示すように、今回は CPU 部の効果のみを調べるため、TLB およびキャッシュは、パワーゲーティングの対象としていないが、今後 Geyser-1 以降ではこれらも含めたシステム全体での低消費電力化を目指す予定である。

2.4.2 実装フロー

Verilog RTL で記述した Geyser-0 を Design Compiler☆ で論理合成し、合成結果のネットリストを Astro☆ を用いて配置配線を行った。

STARC が提供するスタンダードセルの一部をパワーゲーティング用のセルとして独自に開発し、これらを用いてスリープを行うモジュールを設計した。パワーゲーティング用のセルは使用頻度の高いものを優先的に作ったが時間の関係上全てをパワーゲーティング用に直すことはできなかった。このため一部の複合セルなどが使えず、フルに使った場合と比べて面積が 17%ほど増加した。

配置配線後のレイアウトデータにおけるパワースイッチを Cool Power☆☆ を用いて最適化した。このときの制約条件として、パワースイッチが（常時）オンしている状態で 1 つの MTCMOS を共有している複数の論理ゲートが動作して放電したときに、仮想 Ground ラインの最大電圧が VDD の 10%未満になるように行った。

最適化済みのレイアウトデータを再び Astro で配線しマスク用の GDS を構築した。

☆ Synopsys, Inc
☆☆ Sequence Design, Inc

3. 評価方法

3.1 ダイナミック電力とアクティブ時のリーク電力

現在、Geyser-0 は製造中で、実チップでの評価は行うことができない。そこでダイナミック電力とアクティブ時のリーク電力を、レイアウト後のネットリストから PowerCompiler☆ を用いて算出した。スイッチング確率情報 (saif) は、Geyser-0 をレイアウト後のネットリストでゲートレベルシミュレーションを行い生成した。レイアウト後のデータであるため、配線時のバッファ挿入や配線そのもののキャパシタンスも考慮してあり、精度の高い評価が期待される。

3.2 スリープ時のリーク電力

パワーゲーティング対象の回路がスリープモードへ遷移する場合の電力は、シャットダウン後の電圧の過渡状態と、その後の定常状態があり少し複雑である。また、実行するアプリケーションがどのユニットをどれくらいの頻度で使うかによってスリープできる期間が変わってくるため頻度情報も考慮する必要がある。このため、まず、あるアプリケーションを実行したとき各ユニットにおいて “N サイクルのスリープが M 回あった” という情報を取得する。具体的な取得方法に関しては次節で述べる。

次に、それぞれのユニットがスリープモードに遷移した場合の電圧の過渡情報を HSPICE☆ を用いて取得し、N サイクルを連続スリープさせた場合の電力評価モデルを作る。この評価モデルを用いて、あるユニットが N サイクルスリープした後にウェイクアップする場合のの平均消費電力 Lws_N を算出した。Lws_N 値とアクティブ時のリーク電力 Lwa から各ユニットの平均リーク電力を算出できる。

N サイクルのスリープが M_N 回あり、アプリケーションの総サイクル数が T サイクルだとすると、次の式でこのユニットの平均リーク電力 Lw を算出することできる。

$$Lw = \sum_i (Lws_i * \frac{M_i}{T}) + Lwa * (1 - \frac{\sum_i (M_i * i)}{T})$$

モデル作成の際には、N を 1 サイクル刻みで変えてデータを取得するのが理想的だが、計算量の制約から 2〜40 までは 2 サイクル刻みで、後は 50,60,70,80,90,100,200,300,500,750,1000 サイクルで取得し、間は線形補間した。ここでは 200MHz の動作を想定しているため、1000 サイクルは 5ms となる。これ以上は定常状態として扱った。

3.3 利用アプリケーション

アプリケーションには主に MiBench を用いた。MiBench は組み込み系プロセッサでも動くように設計された小規模なベンチマークアプリケーション群で、いくつかのパッケージから構成されている。パッケージには、数学演算、Office アプリケーション、ネットワーク処理など、分野ごとにいくつかの代表的なアルゴリズムや演算処理が集められている。

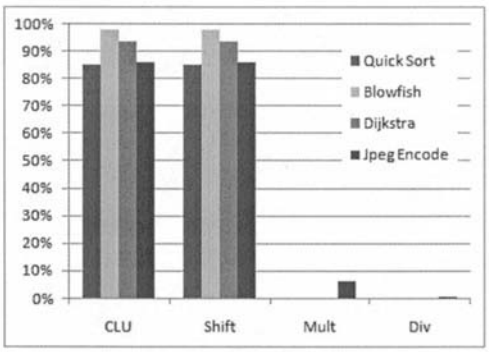

図 7 各ユニットの利用率

本報告ではこれらのアプリケーションのうち、数学演算パッケージから Quick Sort、ネットワークパッケージから Dijkstra、セキュリティパッケージから Blowfish、また、MiBench 以外からメディア系の処理として JPEG エンコードを用いた。

評価用のアプリケーションプログラムは、Linux 環境で GCC(2.95.29) を使って MIPS 用にクロスコンパイルした。最適化には O3 オプションを用いた。これらのアプリケーションを論理シミュレーション環境で動作されるため、コンパイル後のオブジェクトデータを GCC のサブセットである Binary utility の objdump(2.16.1) を用いて逆アセンブルし、機械語をリスト化して整形した。この整形後のデータを Verilog のテストベンチから読み込ませた。

4. 評価結果

4.1 ユニットの使用頻度

4 つの演算ユニットについて、それぞれの使用率を各アプリケーションごとに解析した。この結果を図 9 に示す。

この使用頻度はキャッシュミスによるスリープも含んでいる。どのアプケーションでも、処理量による Power Switch(PS) の ON になる頻度はそれほど大きく変わらないことから、評価プログラムが十分な演算量を行っていることがわかる。

CLU と Shift は同等程度使われており、ブロック暗号の Blowfish では特に多用されている。一方で、Mult unit と Div unit の使用率は全体的に低い。

JPEG エンコードは内部で DCT を行っているため、この演算で Mult unit を利用し、QSORT では Mult, Div 共に 0.2%程使われているが、Dijkstra と Blowfish においては全く使用されていない。このようにアプリケーションプログラムによって、スリープ可能なユニットは全く異なることがわかる。

4.2 ブレイクイーブンポイント

図 8 は温度 65 ℃での CLU のシャットダウン時における消費電力の推移である。

左縦軸に電力横軸 (mW) に時間 (ns) を取ってある。右縦軸は薄い線で表した消費エネルギー (mJ) を表す。モード遷移直後はパワースイッチのダイナミック電力により、一時的に消費電力が増加する。その後一気に減少をはじめて、スリープ定常電位に近づくにつれて減少速度が遅くなる。どのユニットもこのような特徴を持つ曲線となる。

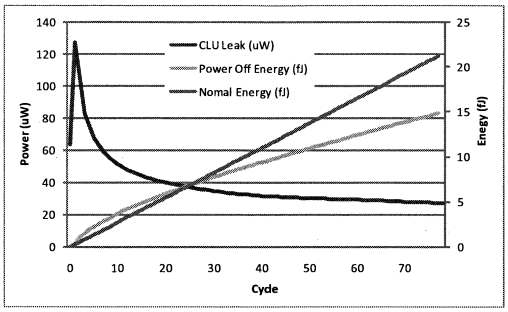

図 8 CLU のシャットダウン電力

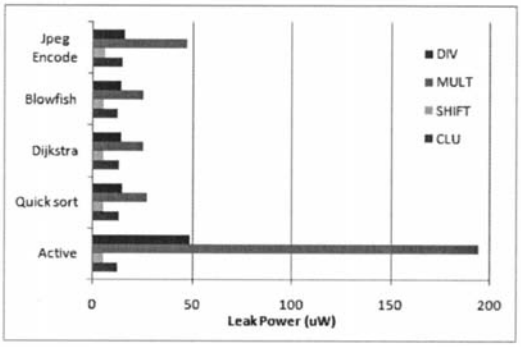

図 9 リーク電力

このとき、シャットダウンをした場合の消費エネルギーとアクティブのままだった場合の消費エネルギーの交点がブレイクイーブンポイントとなる。各ユニットのブレイクイーブンポイントを表 1 に示す。表中の数値は、1 サイクルは 5ns としたサイクル数である。

表 1 各ユニットのブレイクイーブンポイント (サイクル数)

温度	CLU	Shift	Mult	Div	CP0
25 ℃	74	114	74	44	92
65 ℃	26	38	22	14	28
100 ℃	12	16	10	6	12
125 ℃	8	10	8	2	8

温度が上がるとアクティブ時のリークが大きくなるため、ブレイクイーブンポイントは短くなる。

4.3 電力削減効果

各ユニットの使用頻度と ON 時及び OFF 時のリーク電力から、全体のリーク電力の評価を行った。

図 9 に各アプリケーションごとのリーク電力と、パワーゲーティングを行わない場合 (Active) の 25 ℃でのリーク電力をまとめる。

演算器だけで見れば、平均で 76%のリーク電力を削減できている。CP0 はアクティブ時のリーク電力は 16.7uW で、スリープ時は 6.4uW である。コア全体はリーク電力はスリープ制御を行わない場合 439uW であったが、スリープ制御を

行った場合は232uWと47%のリーク電力の削減を達成した。

一方ダイナミック電力はアプリケーションごとの平均において、コア全体で約2.8mWであった。90nmプロセスで200MHzで動作させた場合は、まだリーク電力に対してよりダイナミック電力が大きくなっている。

4.4 エリアとそのオーバヘッド

図10にMIPS-PGコアの各ユニットとメインコントロールが占める面積の割合フロアプランと共に示す。

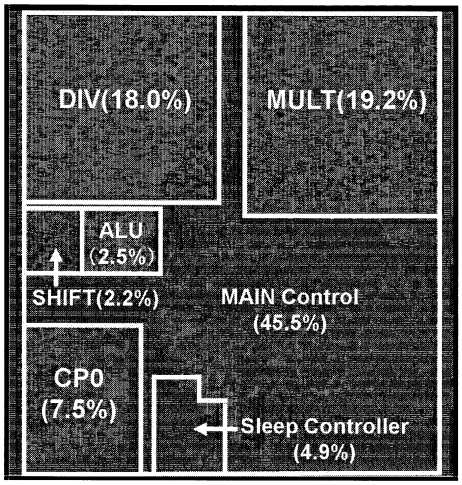

図10 各ユニットの割合

乗算ユニットと除算ユニットを合わせると全体の約4割の面積を占めている。

表2にMIPS-PGコアとMIPS-NMコアのそれぞれの面積を示す。

表2 エリアの比較 (um^2)

	MIPS-PG	MIPS-NM	オーバーヘッド
MAIN	161866	118585	36%
CLU	7764	4970	56%
SHIFT	6899	2540	172%
MULT	58516	40643	44%
DIV	54863	32498	69%
CP0	22861	22861	0%
TOTAL	312770	222097	41%

MIPS-PGコアはMIPS-NMコアと比べて、パワースイッチやその配線による面積のオーバヘッドがある。ただし、MAIN部分のオーバーヘッドはスリープコントローラやその配線などによるものである。CP0のオーバヘッドがないのは元々配線密度が低く面積制約が少なかったため、パワースイッチを入れる隙間が多かったからである。同様の理由で各ユニット間でオーバーヘッドはかなりのばらつきが見られる。

全体では41%のオーバーヘッドが発生している。

5. おわりに

本研究ではMIPS R3000プロセッサのコアをベースに演算器レベルで動的スリープ制御を行う試作チップGeyser-0について、実装及び評価を行った。

動的スリープ制御でリーク電力は47%削減できた。パワーゲーティングによる面積オーバーヘッドは、41%であった。

今後は回路技術の改良によりパワーゲーティングのオーバヘッドを削減すると共にキャッシュ、TLBを含めたプロセッサ全体に対するスリープについて発展させる予定である。コア部分は実行ステージ以外の部分のスリープやコアのスーパスカラ化による電力性能比の向上を模索していきたいと考えている。

謝　辞

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(VDEC)を通し，株式会社半導体理工学研究センター、富士通株式会社、松下電器産業株式会社、NECエレクトロニクス株式会社、株式会社ルネサステクノロジ、株式会社東芝の協力で行われたものである。

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。

参考文献

1) 中村宏、天野英晴、宇佐美公良、並木美太郎、今井雅、近藤正章“革新的電源制御による超低電力高性能システムLSIの構想”. 情処研報 ARC/信学報 ICD, pp.79-84 (2007).
2) Gerry Kane 著, 前川 守 監訳. “mips RISC アーキテクチャ -R2000/R3000”. 共立出版株式会社 (1992).
3) 香嶋俊裕, 武田清大, 大久保直昭, 白井利明, 宇佐美公良. “走行時パワーゲーティングを適用した低消費電力乗算器のアーキテクチャ設計”. VLD No.73, pp.7-12 (2006).
4) 大久保 直昭, 宇佐美 公良. “細粒度動的スリープ制御による動作時リーク電力低減手法”. DA シンポジウム 2006, pp.199-204 (2006 Nov)
5) M.Ishikawa, et.al, “A 4500 MIPS/W, 86μA Resume-Standby, 11μA Ultra-Standby Application Processor for 3G Cellular Phones,” IEICE Trans. on Electronics Vol.E88-C, No.4, pp.528-535 (2005).
6) Y.Kanno, “Hierarchical Power Distribution with 20 Power Domains in 90-nm Low-Power Multi-CPU Processor, “ ISSCC2006, pp.540-541 (2006 Feb).
7) Zhigang Hu, Alper Buyuktosunoglu, Viji Srinivasan, Victor Zyuban, Hans Jacobson, Pradip Bose, IBM T.J. Watson Reserch Center “Microarchitectural Techniques for Power Gating of Execution Units”. Proceedings of the 2004 International Symposium on Low Power Electronics and Design (ISLPED 04). pp32-37 (2004 Aug).
8) 近藤 正章, 中村 宏“リーク電力削減のための細粒度命令スケジューリング手法の検討” デザインガイア No.127 研究報告, page49-54 (2006).
9) Erin Farquhar, Philip Bunce “THE MIPS PROGRAMMER'S HANDBOOK” Morgan Kaufmann Publishers (1994).
10) 関 直臣、他“MIPS R3000における細粒度動的スリープ方式の提案” 情処研報 ARC/信学報 ICD, pp.49-54 (2007).