2011 年 4 月 4 日 FS 専攻修士論文中間発表会発表資料

コンパイラによる電源供給制御命令を用いたプロセッサの低消費電力化の研究

高性能コンピューティング学講座 本多・近藤研究室
1053019 中野 雄太
主任指導教員：近藤正章

1 概要

半導体微細化技術の進歩により CPU 1チップあたりのトランジスタの積載数は年々増加し続け、CPU の高性能化が進んでいる。一方、微細化が進むにつれてリーク電流による消費電力（リーク電力）の増加が問題となっている。リーク電流とは、トランジスタが動作しなくとも流れてしまう電流のことである。本研究は、CPU 動作時にリーク電力を削減することを目的としている。

リーク電力を削減する方法の 1 つに、回路への電源供給を遮断するパワーゲーティング（PG）[1] がある。PG では電源供給を制御するのにも一定の電力を必要とするため、そのオーバヘッドを考慮した PG の制御手法が重要となる。そこで従来研究としてコンパイル時に演算器の使用状況を予測することで、電力のオーバヘッドを考慮した上で PG の制御を行う手法が開発されている。

本研究は実際に命令により各演算器の電源供給を制御可能な CPU チップを用い、実機上で消費電力を計測しつつ従来研究の効果を確かめるとともに、より高い精度で演算器の使用状況を解析し、演算器ごとの電源供給制御戦略を最適化していくことが目標である。また、CPU の温度変化を考慮した上での PG 手法も検討していく。

2 背景

2.1 パワーゲーティング手法

現在、リーク電流を削減する手法の 1 つとして PG が期待されている。PG は、アイドル中の回路への電源供給を遮断することでリークを削減する手法である。

PG では図 1 に示すように回路内にスリープトランジスタを組み込み、そのトランジスタを OFF にすることで回路への電源供給を断ち演算器をスリープさせる。演算器のスリープ時のみリーク電力が削減されている [2]。

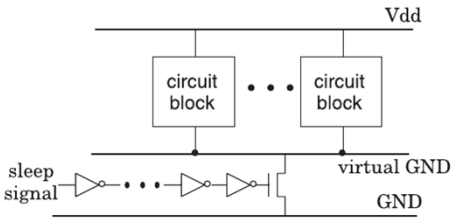

図 1: PG 回路

2.2 PG の考慮すべき点

PG を用いたとしても、必ずリーク電力の削減効果を得られるわけではない。スリープトランジスタの ON/OFF にも電力が消費されてしまうため、演算器をスリープさせることで削減できる電力が、この ON/OFF に必要な電力を上回らなければ逆に CPU 全体の消費電力が増大してしまう。したがって、この ON/OFF の電力と、スリープにより削減できる電力が釣り合う時間 (損益分岐時間) 以上、演算器が使用されない場合にのみ、その当該演算器をスリープさせるという制御が必要となる。

2.3 Geyser

PG を実装したプロセッサとして JST のプロジェクトで開発された Geyser[3] がある。本研究ではこの Geyser の実機を用いて電源供給制御戦略を構築していく。Geyser は MIPS R3000 プロセッサをベースとし ALU、シフタ、乗算器、除算器それぞれを動的に PG 制御することができる。基本的なスリープ制御手法として演算器使用後に常にスリープモードに入る自動スリープ方式が採用されている。また、コンパイル時に演算器の使用状況を予測して PG 制御情報を命令に付加し電源供給制御ができる PG 制御情報付き命令がある。

PG 制御情報付き命令

図 2 は Geyser 用の PG 制御情報付き命令の構造を示している [3]。通常は演算命令の命令コード（上位 6 ビットが 000000) を 100111 に設定すると、演算後に使用した演算器の自動スリープをキャンセルする。例えば、乗算を行う命令で上位 6 ビットが 100111 であれば、演算終了後乗算器を動作モードのままにする。動作モードは次の乗算命令が到着するまで継続される。

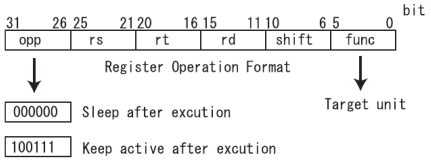

図 2: PG 制御情報付き命令

3 提案手法

従来のコンパイラによる PG 制御では演算器ごとの損益分岐時間を考慮しておらず、設定された 1 つの損益分

岐時間の値のみで PG 制御が行われている。また分岐命令の分岐確立を 50 %と仮定しており、実際の分岐確率を考慮していない。そのため各演算器のアイドル時間の予測が不正確となってしまう。これらが原因となり損益分岐時間に達せずにスリープモードへ遷移してしまい十分なリーク電力削減効果を得られない可能性がある。そこで本研究では以下の点で効率的な PG 制御を狙う。

3.1 より高い精度での演算器ごとの制御

コンパイル時に命令コードを解析することで将来いつ各演算器が使われるかを予測する。その際に分岐確率をプロファイリングにより求めることでより高い精度で演算器ごとの使用状況を予測する。そして使用状況や演算器ごとの損益分岐時間からスリープ可能時間を求め、それを基に PG 制御情報付き命令の挿入された命令コードを生成できるようにする。以上のようなソフトウェアのサポートによりスリープトランジスタ ON/OFF のオーバーヘッドを抑制し、リーク電力の削減率を向上させることができる。

3.2 チップ温度による損益分岐時間の変化

損益分岐時間は CPU チップの温度が高くなると短くなることがわかっている。これは高温下ではスリープ時に対してアクティブ時のリーク電力の比率が大きくなるためである。ここでチップ温度を検知して適切に PG を行うことができれば電力のロスを少なくすることができる。

4 進捗状況

現在は Geyser に搭載されている各演算器の損益分岐時間を実機で評価中である。評価には測定した電流値を用いている。

・演算器ごとの損益分岐時間を求め方

スリープトランジスタの ON/OFF は数 ns 単位で切り替わるため、瞬時に変化する電流値を読み取ることは難しい。つまり ON/OFF にかかる電力を求めることができないので、PG で削減できた電力との比較による損益分岐時間の導出はできない。そこで 2.3 節で述べた PG 制御情報付き命令を用いて、PG を行う（演算器が命令実行後に毎回スリープモードになる）場合と PG を行わない（演算器が常にアクティブになる）場合の消費電力を比較することで導出する。図 3 のようにコード内の nop 命令を増やしていくことでサイクル数を増やしていき、それぞれの消費電力が等しくなるサイクル数を損益分岐時間とする。

```
A : mult     // 掛け算
    nop      // ノーオペレーション
    nop
    nop
     :
     :
    jump A  // Aへジャンプ
```

図 3: 損益分岐時間を求めるための命令コード

・乗算器における損益分岐時間

例として乗算器における損益分岐時間を求める。電圧値を一定とし、PG を行う場合（PG）と PG を行わない場合（Not PG）の電流値で比較する。電流値を比較した結果を図 4 に示す。それぞれのグラフは 6 サイクルと 7 サイクルの間で交わっているので、乗算器の損益分岐時間はおよそ 7 サイクルだとわかる。

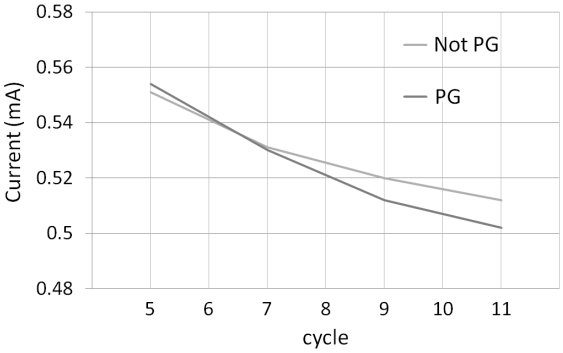

図 4: 損益分岐時間の導出

5 今後の方針

アプリケーションを用いて評価しつつ、温度や命令数、命令の種類などの条件を変えながら消費電力を測定していく。そして消費電力が低くなる条件を見つけていき、命令コード内のどこにスリープ命令を挿入すれば良いかを検討していく。そこからコンパイラによる演算器ごとの PG 手法のアルゴリズム考案に繋げていきたい。

6 まとめ

本研究ではリーク電力を効率的に削減するためにコンパイラによって損益分岐時間以上スリープできるようにすることが目的である。そのためには各演算器の損益分岐時間がわからなければならない。そこで現在は実際に Geyser で評価をして損益分岐時間を求めている。

参考文献

[1] N.Hanchate and N.Ranganathan,“ LECTOR:A technique for leakage reduction in CMOS circuit, ” *IEEE Trans. Very Large Scale Integer.(VLSI) Syst.,* vol.12,no.2,pp.196-205,Feb.2004.

[2] 近藤正章, 高木紀子, 中村宏, “ Pipeline Blocking: 走行時パワーゲーティングのための命令実行制御手法, ”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.1-13,2008.

[3] 関直臣, 他, “ 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, ”電子情報通信学会論文誌,Vol.J93-D,No.6,pp.920-930,2010.