2013 年 4 月 2 日 情報システム基盤学専攻中間発表資料

GPU における細粒度パワーゲーティング向けスレッド発行制御に関する研究

所属： 高性能コンピューティング学講座
発表者： 松本 洋平
主任指導教員： 近藤 正章

1 はじめに

近年、GPU の演算処理能力が注目されており、その高い演算処理能力を活用して、HPC アプリケーションでの利用が広まっている。

GPU は大量の SIMD 型演算器を搭載することで、高性能を達成するが、消費電力が大きく、アイドル時の消費電力が大きいことが問題点としてあげられる。例えば NVIDIA GeForceGTX 280 のアイドル時電力は 83[W] である。アイドル時の消費電力には、回路動作やクロックゲーティングができないことで消費される動的消費電力、およびリーク電流によるリーク消費電力が含まれる。近年、半導体プロセスの微細化に伴うリーク電流の増大が問題となっており、リーク消費電力を削減することは GPU にも効果的である。

2 研究の背景

LSI においてリーク消費電力を削減する手法としてパワーゲーティング (PG) がある。PG は LSI の回路ブロックに対して電源遮断用のスイッチを設け、動作の必要がないときに電源供給を遮断することでリーク電力を削減する手法である。従来の GPU には Streaming Multiprocessor (SM) 単位での粗粒度の PG が実装されている。しかし、SM 単位の PG では電源 ON/OFF の際の時間的・電力的なオーバーヘッドが大きく、また、SM 全体を使用しない場合にしか PG を適用できない。本研究では、SM に対して処理が割り当てられている場合でも SIMD 演算器内で演算処理が行われていないサイクルがある点に着目し、SIMD 演算器の各演算器単位で細粒度に PG を適用することを考える。GPU の SIMD ユニットの各演算器に対して PG を適用する場合には図 1 のように、それぞれの演算器とグラウンド線との間にスリープトランジスタを挿入する。各スリープトランジスタは sleep signal によって制御され、電源供給の ON/OFF が行われる。細粒度な制御より、従来の SM 単位での PG よりもリーク消費電力が削減できると期待できる。ただし、電源の ON/OFF によるスリープモードとアクティブモードの遷移にはある程度の電力的なオーバーヘッドが生じるため、頻繁なモード遷移を行うと、かえって消費電力が増大する可能性がある (図 2)。そのため、時間的に細粒度に PG を行うに際には、電力オーバーヘッドよりもリーク電力の削減量が大きくなるように、1 回電源遮断した際の PG サイクルを長

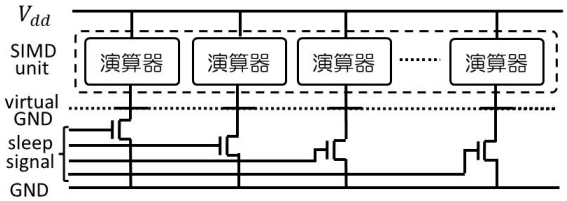

図 1: 細粒度パワーゲーティング

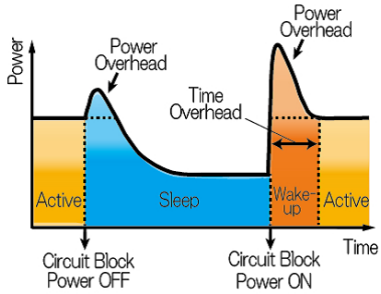

図 2: パワーゲーティングのオーバーヘッド

時間化する必要がある。このオーバーヘッドが吊り合う時間 を Break Even Time (BET) と呼ぶ。

3 スレッド発行制御手法

GPU は分岐命令やキャッシュミスによるストールが発生してしまう為に、各演算器で演算が実行されないサイクルが多く発生する。この場合、使用されない演算器の電源供給を遮断し、リーク電力を削減できる可能性がある。しかし、使用されない演算器の位置やサイクルがばらばらであると、各演算器の ON/OFF の切り替えが頻発し、効率的な PG はできない。本章では SIMD 演算器に細粒度 PG を適用した場合に、リーク電力削減効果を増大させるための 2 つのスレッド発行制御手法を説明する。

3.1 スレッドコンパクション

SIMD 演算器単位で PG を行う場合、その演算器の位置がばらばらであると、演算器毎に ON/OFF の切り替えが頻発し、効率的な PG はできない。本提案手法では、文献 [1] で提案されている SIMD ユニットの演算器使用率を向上させるためのスレッドコンパクションを応用し、それぞれの演算器で分散されたアイドルサイクルを一部の演算器にまとめて、一回のアイドル時間が長くなるようにスレッド発行制御を行うことで効率的なリーク電力削減を狙う。図 3(a) のスレッド実行の例に対して、スレッドコンパクションを適用した場合の実行の様子を図 3(b) に示す。コンパクションにより、リーク電力削減効果がある演算器は 4 つに増加している。

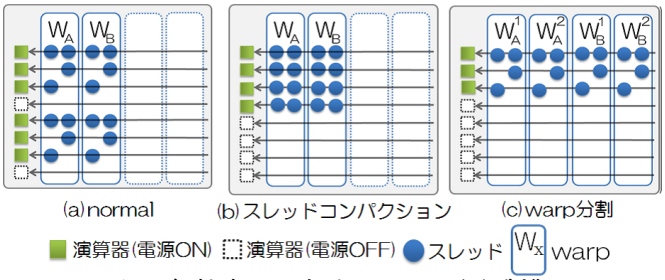

図 3: 細粒度 PG 向けスレッド発行制御

3.2 warp 分割

図 3(a) のスレッドが発行されていない空きサイクル (点線で囲まれているスロット) のようにプログラム上のスレッド数が十分でない場合、キャッシュミスなどによるストールが発生すると実行可能するべき warp が存在せず、全演算器が使用されないサイクルが発生する。そのような場合には、1warp を 2 つの warp に分割することで、使用する演算器を半分にし、長期間スリープにできる演算器数を増やすことを狙う。図 3(c) に図 3(a) を warp 分割した例を示す。warp 分割により、リーク電力削減効果がある演算器は 5 つに増加している。

4 評価

提案手法をサイクルレベルシュミレータの GPGPU-Sim を用い評価を行った。評価に用いた GPU のハードウェアパラメータは NVIDIA GeForce GTX 480 に従った。また、ベンチマークについては CUDA で記述された 6 つのベンチマークプログラムを用いて評価を行った。

提案手法のスレッドコンパクションについては、ログを解析することでコンパクションした際の評価を行った。また、warp 分割は GPGPU-Sim のパラメーター設定で 1warp 内のスレッド数を 32 スレッドから 16 スレッドに変更して評価を行った。なお、BET については 100 サイクルの場合について評価する。

5 評価結果

図 4 に、ベンチマークごとのリーク消費エネルギー可能削減サイクルの割合を示す。全体としてスレッドコンパクション、warp 分割を行うことで手法でリークエネルギー削減の効果が得られることがわかる。また、スレッドコンパクションと warp 分割の両者を適用するとその効果が増加することがわかる。BFS と MUM は分岐命令を含んでいる。また、演算器の使用率が低いベンチマークであるため、normal に比べてスレッドコンパクションの効果は BFS で 1.19 倍、MUM で 1.08 倍となった。warp 分割の効果は BFS で 1.23 倍、MUM で 1.18 倍となった。また、スレッドコンパクションと warp 分割を組み合わせた手法ではの相乗効果により normal に比べて BFS で 1.26 倍、MUM で 1.20 倍となった。LPS と LIB は warp 中のスレッド数が多く演算器の使用率が高いベンチマークのため、スレッドコンパクションのみによるリークエネルギー削減の効果はほとんどなかった。一方で、warp 分割では使用する演算器数を半分にするためリーク電力削減の効果は LPS で 10.2 倍、LIB で 7.60 倍となった。NN は warp

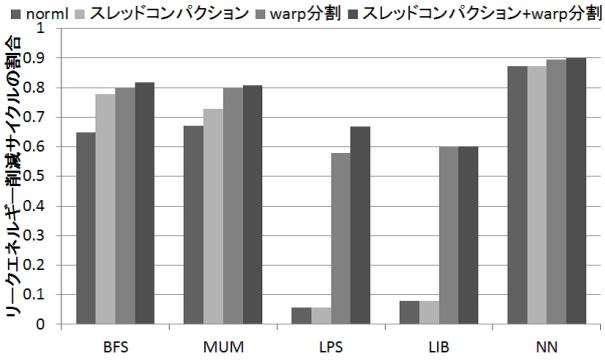

図 4: BFS を実行した場合のリーク電力削減率

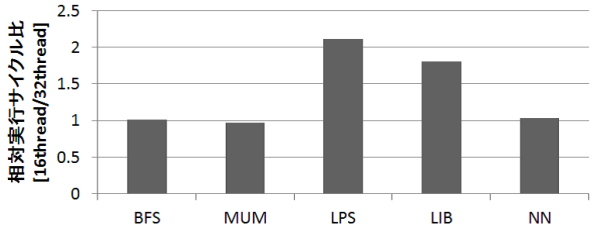

図 5: 実行サイクル数の比率

中のスレッド数が少なく、normal の場合においても特定の演算器のみでスレッドの実行が行われるためリークエネルギー削減量はスレッドコンパクションと warp 分割を組み合わせた手法でも 1.03 倍であり、他のベンチマークに比べ提案手法の効果が現れない結果となった。

図 5 に、normal に対する warp 分割を用いた場合の各ベンチマークの相対実行サイクルを示す。warp 分割を適用し、1 ワープのスレッド数を 32 スレッドから 16 スレッドに削減した場合でも、演算器の使用率が低い BFS、MUM、NN ではほとんど実行サイクルが変わっていない。そのため、このようなベンチマークでは warp 分割手法によりほぼ性能低下なく大幅なリーク消費エネルギー削減効果を得ることができる。一方で、演算器の使用率が高い LPS 、LIB では実行サイクル数が倍近くなる結果となった。このことからも、演算器使用率が高い場合には warp 分割を用いるべきではない。

6 まとめと今後の方針

GPU の SIMD 演算器を対象に、細粒度 PG によるリーク電力削減率の向上を目的としたスレッド発行制御手法について研究を行なっており、これまでの取り組みとして、シミュレーションによる初期評価を行った。

今後、実際にシミュレータ上にスレッド発行制御機構を実装することや、アプリケーションの特性に応じて制御を使い分けることで性能低下を抑えつつ効率的に PG が行えるような手法の開発に取り組んでいく予定である。研究成果として、2013 年 3 月 26 日に行われた第 196 回 計算機アーキテクチャ研究会（情報処理学会）で発表を行った。

参考文献

[1] Wilson W. L , et al. Dynamic Warp Formation and Scheduling for Efficient GPU Control Flow. MICRO 40 , pp.407-420.