

可変レベルキャッシュのモード切り換えアルゴリズムの改良

刀根 舞歌 佐々木 敬泰 深澤 祐樹 近藤 利夫

概要: 携帯端末等のプロセッサでは、低消費電力と高性能の両立が求められている。しかし、トランジスタ数の増加やゲート酸化膜の薄膜化に伴うリーク電力の増加が問題となっている。プロセッサの中でもトランジスタ数の多いキャッシュのリーク電力削減は、プロセッサの省電力化に繋がる。そこで著者らは低消費電力キャッシュの手法の1つとして、可変レベルキャッシュを提案している。可変レベルキャッシュはプログラムの負荷に応じて、通常キャッシュと同じ状態から、キャッシュ容量の50%あるいは75%をスリープ状態に移行させることで、高性能と低消費電力の両立を目指す手法である。具体的には、一定サイクルごとに計測したキャッシュミス率があらかじめ決められた閾値を超えると、容量をどの程度スリープさせるかを表すモードを切り換え、容量を動的に変えて不要な電力を削減する。しかしながら従来の可変レベルキャッシュでは、必要なデータのない領域への無駄なアクセスや、一定サイクルごとのミス率のみを考慮した判定による過度なモード切り換えが見受けられた。また従来の可変レベルキャッシュは閾値を静的に決めていたため、実行状況やベンチマークの特性に合ったモード切り換えを行えない。そこで本論文では、3つの手法、すなわち電源を完全に落とすシャットダウンの導入、実行したサイクルまでの通算のミス率も考慮するモード切り換えアルゴリズム、動的に閾値を変更する動的ミス閾値制御手法を提案する。これら3つの提案手法を実装して評価したところ、従来の可変レベルキャッシュと比較して、平均で消費電力は14.75%、実行時間は0.32%減少した。

1. はじめに

スマートフォンやノートパソコンなどの携帯端末の高性能化に伴い、プロセッサの消費電力が増加している。その結果、バッテリーの駆動時間の減少が懸念されている。そのため、プロセッサの性能維持と消費電力の削減の両立が要求されている。

プロセッサの消費電力は、主に動的消費電力と静的消費電力の2つに分けることができる。動的消費電力は回路の動作によって消費される電力である。一方、静的消費電力はトランジスタから漏れ出てしまう電流、すなわちリーク電流により回路の動作に関係なく消費されるのでリーク電力ともよばれる。今日、回路の微細化に伴うトランジスタ数の増加や、高性能化のための絶縁膜の薄膜化により、リーク電力が増加する傾向にある。キャッシュはプロセッサの構成要素の中でもトランジスタ数が多いため、消費電力増加の原因になりやすい。よって、キャッシュのリーク電力を削減することはプロセッサの低消費電力化に繋がる。そのため、さまざまな低消費電力キャッシュ手法が提案されている。

たとえば一般に用いられているSRAMやDRAMなどの揮発メモリは、電源を遮断しない部分のリーク電力は削減

できない。この問題を解決するため、近年STT-MRAM[1]のような不揮発メモリの開発が進んでいる。不揮発メモリは使用していない領域を遮断できるため、揮発メモリを不揮発メモリで代替すれば電力の削減につながる。しかしこうしたノーマリーオフコンピューティングを実現するには、SRAMのアクセス性に匹敵する不揮発メモリが必要となるが、それを満たす不揮発メモリはまだ存在していない。また全てのメモリを単純に不揮発メモリに置換しただけでは、かえって性能が悪化することがわかっている[2]。よって不揮発メモリの技術が確立していない以上、キャッシュをSRAMで構成することがまだ主流である。

そこで著者らは、低消費電力キャッシュ手法の1つとして、可変レベルキャッシュ[3]を提案している。可変レベルキャッシュはキャッシュミス率に応じて、キャッシュの一部をデータ保持が可能な最低電力状態であるスリープ状態[4]に移行し、キャッシュの容量を擬似的に変更する手法である。著者らは本論文において、可変レベルキャッシュの動作解析を行い、データが無い場所への無駄なアクセスを発見した。また削減効率の悪いベンチマークではプログラム全体のキャッシュミス率が高いことが判明した。これは従来の可変レベルキャッシュが、一定サイクルごとのミス率しか考慮しないためである。さらにベンチマークや実

行状況によって適切なミス閾値は異なるが、実験的に求めた値を静的に定めているため効率が悪くなっている可能性がある。

そこでこれらの問題を解決するために、3つの手法、すなわちシャットダウン状態を導入、実行したサイクルまでの通算のミス率も利用するモード切り換えアルゴリズム、ミス閾値を動的に変更できる動的ミス閾値制御を提案する。これらの手法の有効性を示すためにシミュレーション評価を行った結果、通常のキャッシュと比較して消費電力が平均 52.88 % 減少し、実行時間が平均 1.22 % 増加した。また従来の可変レベルキャッシュと比較して消費電力が平均 14.75 %、実行時間が平均 0.32 % 減少した。

2. 低電力キャッシュ手法に関する研究

2.1 DRI キャッシュ

2.1.1 DRI キャッシュの概要

可変レベルキャッシュは低電力キャッシュ手法の1つである DRI キャッシュ [5] を改良した手法である。DRI キャッシュは命令キャッシュ向けに開発された手法である。DRI キャッシュの概念図を図 1 に示す。

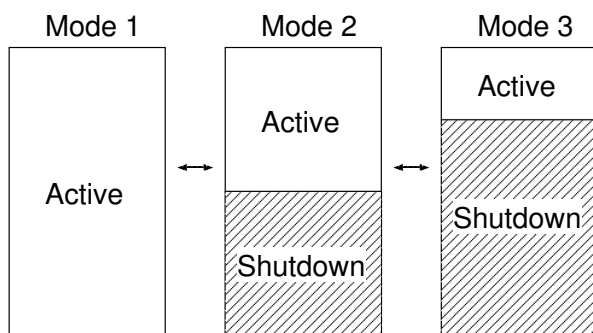


図 1 DRI キャッシュの概念図

初期状態は通常のキャッシュと同じく活性化している状態、すなわち通常状態である。プログラム実行中においてキャッシュのミス率が低い場合、キャッシュ容量をあまり必要としないと判断し、現在のキャッシュ容量の半分をシャットダウン状態に移行する。シャットダウン領域は完全に電源を落とすため、その部分の消費電力が削減できる。反対にキャッシュミス率が高い場合、キャッシュ容量を多く必要としていると判断し、シャットダウン領域を通常状態に戻し、現在のキャッシュ容量を倍にする。

このようにキャッシュミス率でキャッシュ容量を切り換えることにより、不要な領域の電力削減をはかる。しかし、この手法をそのままデータキャッシュに適用すると、2つの問題が発生する。

2.1.2 DRI キャッシュの問題点

1つ目はキャッシュに書き込んだデータが消失する問題である。シャットダウン状態は電源を遮断するため、その

領域ではデータを保持できない。そのため書き込んだデータが消失する可能性がある。データの消失を防ぐために、データキャッシュではメインメモリにデータを書き戻すライトバックが必要となる。そのため、レイテンシが増加してしまう。

2つ目は保持できるデータ量が減少する問題である。キャッシュ容量を減らすため、当然保持できるデータ量が減少する。そのため、すぐに使用するかもしれないデータを保持できず、キャッシュミス率が増加してしまう。

2.2 可変レベルキャッシュ

2.2.1 可変レベルキャッシュの概要

データを保持できない問題を解決するために、著者らは可変レベルキャッシュ (Variable Level Cache : VLC) を提案している。可変レベルキャッシュは DRI キャッシュと同様に、L2 キャッシュのミス率に対応して、L2 キャッシュの容量を動的に変化させる手法である。しかし、可変レベルキャッシュでは完全に電源を落とすシャットダウン状態の代わりに、データを保持できる程度の低電力状態であるスリープ状態を使用する。スリープ状態は、完全に電源を落とすシャットダウン状態よりは消費電力が大きい、通常状態よりはるかに消費電力が小さくてすむ。可変レベルキャッシュの概念図を図 2 に示す。

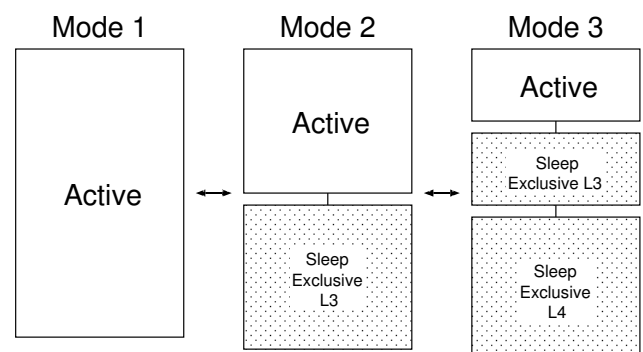


図 2 可変レベルキャッシュの概念図

従来の可変レベルキャッシュには3つのモードがある。モード1は、通常のキャッシュと同じくキャッシュ全体が活性化している通常状態である。モード1でキャッシュミス率が低い場合、すなわちキャッシュ容量をそれほど必要としない場合は、モード2へ移行する。モード2は、L2 キャッシュの下位 50 % の領域をスリープ状態に移行させることで、キャッシュの容量を擬似的に半分になっている状態である。さらにモード2でもキャッシュミス率が低い場合は、モード3へ移行する。モード3は、L2 キャッシュの下位 75 % の領域をスリープ状態に移行させることで、キャッシュの容量を擬似的に 1/4 にしている状態である。一方、モード2やモード3においてキャッシュミス率が高い場合、すなわちキャッシュ容量を多く必要とする場合は

1つ上位のモードに戻す。

このようにプログラムの動作に合わせてモードを動的に切り換えることにより、必要なデータを保持しつつ電力消費を抑えることができる。

2.2.2 可変レベルキャッシュの動作

次に、可変レベルキャッシュの動作について説明する。プログラム開始時、最初のモードはモード1である。モード1でアクセスが発生したとき、データが存在する場合は図3のようにヒットとなり、データが存在しない場合は図4のようにミスとなる。

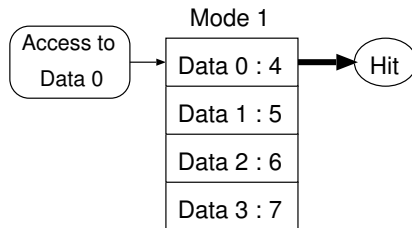


図3 ヒット時の動作

ミスの場合は、メインメモリからデータを取得しなければならない。

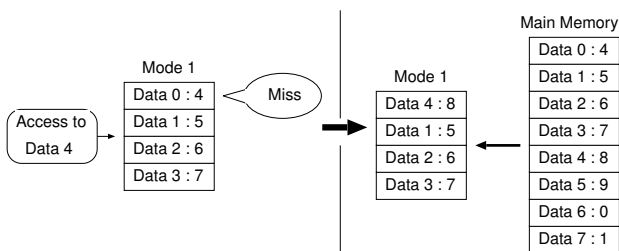


図4 ミス時の動作

モード1において、一定サイクルごとに計測したキャッシュミス率が、下位のモードへ移行するためのミス閾値、すなわち下方のミス閾値を下回った場合、モード2に移行する。可変レベルキャッシュでは、スリープ領域は擬似的に下位の排他的キャッシュ [6] として扱われる。モード2では、キャッシュの上位50%の領域をL2キャッシュ、スリープ状態になる下位50%の領域をL3キャッシュとみなす。モード2でアクセスが発生したときの動作を図5に示す。まずL2キャッシュにアクセスする。データが存在する場合はヒットとなる。データが存在しない場合、スリープ状態に移行したL3キャッシュを一旦通常状態に戻す。それからL3キャッシュに再アクセスする。そこにデータが存在する場合はヒットとなる。このとき、ヒットしたラインを、L2キャッシュの対応するラインと交換する。これにより、良く使用するデータをL2キャッシュに保持しておくことができる。

データが存在しない場合はミスとなる。モード2におい

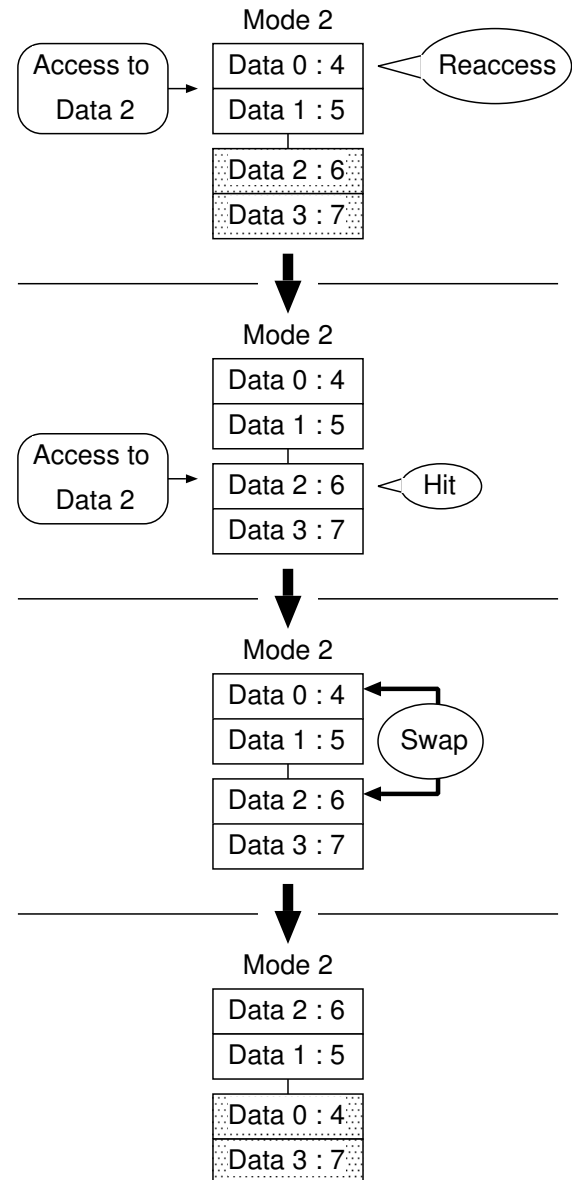


図5 再アクセス・交換時の動作

て、一定サイクルごとのミス率が、上位のモードへ移行するためのミス閾値、すなわち上方のミス閾値を上回った場合、モード1に移行する。反対に、一定サイクルごとのミス率が下方のミス閾値を下回った場合、モード3に移行する。モード3では、キャッシュの上位25%の領域をL2キャッシュ、スリープ状態の領域のうち、上位25%の領域をL3キャッシュ、残りの下位50%の領域をL4キャッシュとみなす。モード3でアクセスが発生したとき、まずL2キャッシュにアクセスする。データが存在する場合はヒットとなる。データが存在しない場合、スリープ状態に移行したL3キャッシュを一旦通常状態に戻す。それからL3キャッシュに再アクセスする。そこにデータが存在する場合はヒットとなる。このとき、ヒットしたラインを、L2キャッシュの対応するラインと交換する。データが存在しない場合、今度はL4キャッシュを、一旦通常のキャッシュ状態に戻し再アクセスする。そこにデータが存在する場合はヒットとなり、存在しない場合はミスとなる。モード3において、一定サイクルごとのミス率が、上方の閾値を上回った場合、モード2に移行する。

2.3 従来の可変レベルキャッシュの問題

従来の可変レベルキャッシュでは、実際のキャッシュの挙動を考慮せず、一定サイクルごとに計測したキャッシュミス率のみでモード切り換えを行っていた。しかし通常状態やスリープ状態のキャッシュメモリの中でも、アクセスが無くデータを保持する必要のない領域や、必要なデータが存在しない領域へのアクセスが発生する可能性がある。こうした領域を減らすことができればさらなる電力削減が見込める。また従来の可変レベルキャッシュでは、ミス閾値としてあらかじめ実験的に求めた値を用いているため、ベンチマークや実行状況ごとに最適なミス閾値を利用できず、最適な結果を得られていない可能性もある。そこでまずはこれらの予想を確認するために、可変レベルキャッシュの動作を確認するため可視化ツールを開発した。この可視化ツールはトレースドリブン型のシミュレータであり、メモリアクセスのみを読み出すためキャッシュのみの評価を高速に行うことができる。

可視化ツールによる可変レベルキャッシュの動作の解析の結果、再アクセスが一部のスリープ領域に集中している、目的のデータが存在しないにもかかわらず再アクセスを繰り返している、上位のモードと下位のモードを連続で行き来しているといったことを確認した。また電力削減効果の低いベンチマークでは、シミュレーション開始から終了までのミス総回数をアクセス総回数で割り求めたキャッシュミス率、すなわちプログラム全体のミス率が高いことが判明した。

プログラム全体のミス率が高いということは、必要なデータが存在しないということである。つまりキャッシュ

容量を多く、上位のモードでいる時間を長く必要としている。またミス率が高くなる原因の1つに、モード切り換えが多発していることがあげられる。従来の可変レベルキャッシュでは、プログラム全体のミス率が高くても一定サイクルごとのミス率が低ければ、下位のモードへ移行してしまう。そのため必要以上のモード切り換えやデータが存在しないために起こるキャッシュミス、さらにスリープ状態による再アクセスとキャッシュラインの交換が増加している。またプログラム全体のミス率が高いということは、適切なミス閾値を設定できていないということでもある。しかし静的にミス閾値を決める方法では、あらゆるベンチマークや状況に適切な値を設定することは不可能である。

これらの問題を解決するためには、必要以上のモード切り換えの削減と、動的なミス閾値の決定を行わなければならない。

3. シャットダウン状態の再導入

3.1 シャットダウン状態の再導入の理由と効果

DRIキャッシュを改良して可変レベルキャッシュを作成する際に、データの損失を防ぐためにシャットダウン状態からスリープ状態に変更した。しかし、使用しないデータを保持するスリープ領域をシャットダウンすれば、その領域への再アクセスやデータの交換が減少し動的消費電力の削減につながる。そこで、通常状態とスリープ状態に加えシャットダウン状態も用いる手法を提案する。DRIキャッシュではモードにより決められた領域全体をシャットダウンするためデータの保持ができないが、この手法ではデータが必要な領域はスリープ状態に、不要な領域はシャットダウン状態に移行できる。よってデータを保持しつつ、電力削減を効率的に行うことができる。

3.2 シャットダウン状態の概要

シャットダウン状態とは完全に電源を遮断する状態である。シャットダウンのイメージを図6に示す。

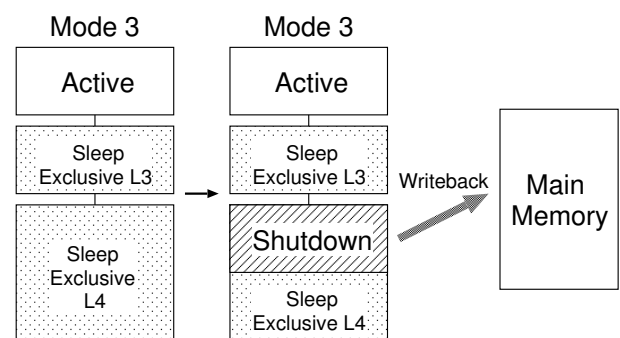


図6 シャットダウン時の動作

シャットダウン判定はキャッシュを4分割した領域ごとで行う。スリープ領域がある場合、一定サイクルごとに領

域単位でヒット数を計測し、基準値を超えたらシャットダウン状態へ移行する。このとき、シャットダウンする領域内に書き換えられたデータがあればメモリへ書き戻す。

3.3 シャットダウン状態の利点と欠点

シャットダウン領域はデータを保持していないことが確定しているため、データを探すときにその領域への再アクセスは不要となる。またシャットダウン状態は完全に電源を遮断するため、シャットダウン中はリーク電力が発生しない。しかしデータを保持できなくなるためミス率が上昇し、復帰電力とレイテンシの増加を招く。ミス率の上昇にともなうメモリアクセスにかかる電力よりも、シャットダウンによる削減が大きければ消費電力の減少が期待できる。

4. 通算のミス率によるモード切り換えアルゴリズムの提案

4.1 通算のミス率によるモード切り換えアルゴリズムの概要

本論文では、実行したサイクルまでの総ミス数を総アクセス数で除算し求めたミス率を通算のミス率と定義する。従来の可変レベルキャッシュは、一定サイクルごとに計測したミス率のみでモード切り換え判定を行うため、実際の挙動にあったモード切り換えを行うことができない。そこで過度なモード切り換えを避けるために、通算のミス率を利用するモード切り換えアルゴリズムを提案する。このアルゴリズムでは一定サイクルごとに計測するミス率、すなわち一定サイクルごとのミス率でモード切り換え判定を行う前に、通算のミス率でモード切り換え判定を行う。通算のミス率を利用することで、短期間に集中したアクセスの影響を減らす、アクセスが無かった場合にも判定を行うといった、より実際の挙動に合ったモード切り換えを行うことができる。

4.2 通算のミス率によるモード切り換えアルゴリズムの詳細

通算のミス率を利用するモード切り換えアルゴリズムのフローチャートを図7に示す。

図7より、最初に通算のミス率と通算のミス率用の閾値とを比較する。通算のミス率が通算のミス率用の閾値以上である場合、キャッシュ容量が多く必要であると判断し、現在のモードを強制的にモード1にする。

通算のミス率が通算のミス率用の閾値を下回っている場合、一定サイクル内にアクセスがあったかどうか確認する。一定サイクル内にアクセスが無かった場合、通算のミス率と下方のミス閾値とを比較する。通算のミス率が、下方のミス閾値以上である場合は何もしない。通算のミス率が、下方のミス閾値を下回っている場合、1つ下位のモー

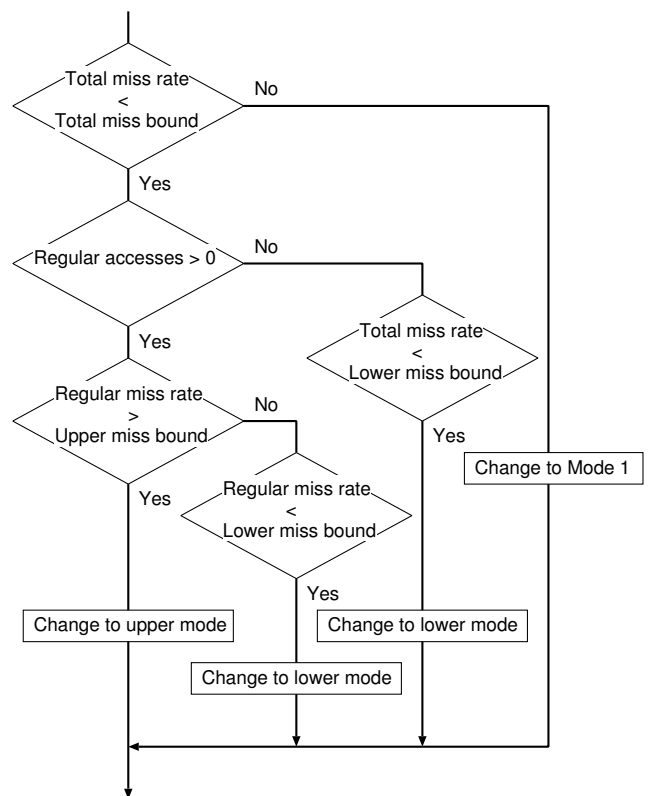


図7 通算のミス率によるモード切り換えアルゴリズムのフローチャート

ドへ移行する。すでに最下位のモードである場合は何もしない。

一定サイクル内にアクセスがあった場合、従来の可変レベルキャッシュと同じ処理をする。つまり一定サイクルごとに計測したミス率と、上方のミス閾値とを比較する。一定サイクルごとのミス率が上方のミス閾値を上回っている場合、1つ上位のモードへ移行する。すでに最上位のモードである場合は何もしない。一定サイクルごとのミス率が、上方のミス閾値を下回っている場合、一定サイクルごとのミス率と、下方のミス閾値とを比較する。一定サイクルごとのミス率が、下方のミス閾値を下回っている場合、1つ下位のモードへ移行する。すでに最下位のモードである場合は何もしない。

このようにして、通算のミス率を利用してモード切り換えを行う。

4.3 通算のミス率によるモード切り換えアルゴリズムの利点

図8に、通算のミス率によるモード切り換えアルゴリズムを実装した可変レベルキャッシュの具体例を示す。従来の可変レベルキャッシュでは、一定サイクルごとのミス率とミス閾値との比較のみを行う。図8の表Aより、一定サイクルごとのミス率が下方のミス閾値を下回っているため、1つ下位のモードへ切り換える。一方、通算のミス率によるモード切り換えでは、先に通算のミス率と通算のミ

ス閾値とを比較する。図8の表Bより、通算のミス率が通算のミス閾値を上回っているため、モードを切り換えない。

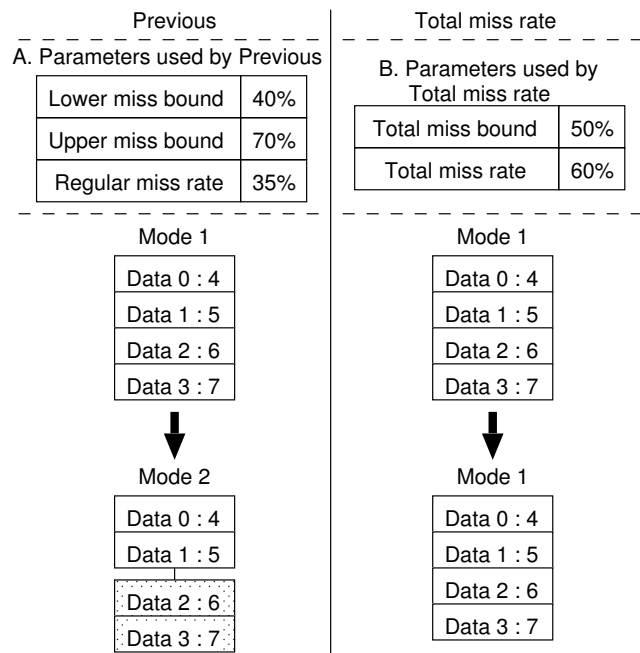


図8 従来の可変レベルキャッシュと通算のミス率によるモード切り換えアルゴリズムの動作 (1/2)

直後に図9のようなアクセスがあった場合、従来の可変レベルキャッシュでは再アクセス・交換・ミスが発生するのに対して、通算のミス率によるモード切り換えアルゴリズムを実装した可変レベルキャッシュでは発生しない。

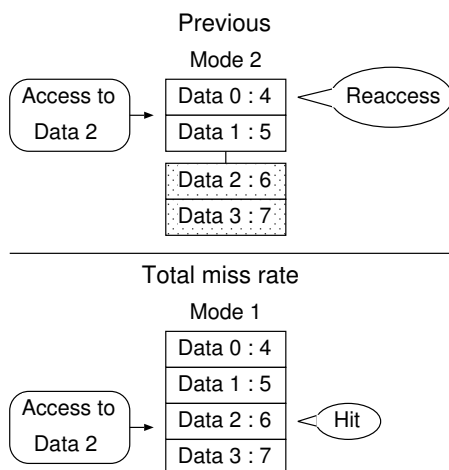


図9 従来の可変レベルキャッシュと通算のミス率によるモード切り換えアルゴリズムの動作 (2/2)

このように、通算のミス率によるモード切り換えアルゴリズムを導入すると、無駄な下位モードへの移行を抑制できる。プログラム通算のミス率が高い場合、ミス率を下げる事ができる。しかしプログラム通算のミス率が低い場合には、従来と同じ処理を行うことが多いため、あまり効果がない。

5. 動的ミス閾値制御の提案

5.1 動的なミス閾値の概要

従来の可変レベルキャッシュは、ミス閾値としてあらかじめ実験的に求めた値を用いていたため、ベンチマークや実行状況に最適なミス閾値を利用できず、最適な結果を得られなかった。そこで、動的にミス閾値を変更する、動的ミス閾値制御 (Dynamic Miss Bound Control : DMBC) を提案する。動的ミス閾値制御はヒット数に応じてミス閾値を上昇させることで、下位のモードへの移行を促す手法である。

5.2 動的ミス閾値制御の詳細

動的ミス閾値制御のフローチャートを図10に示す。

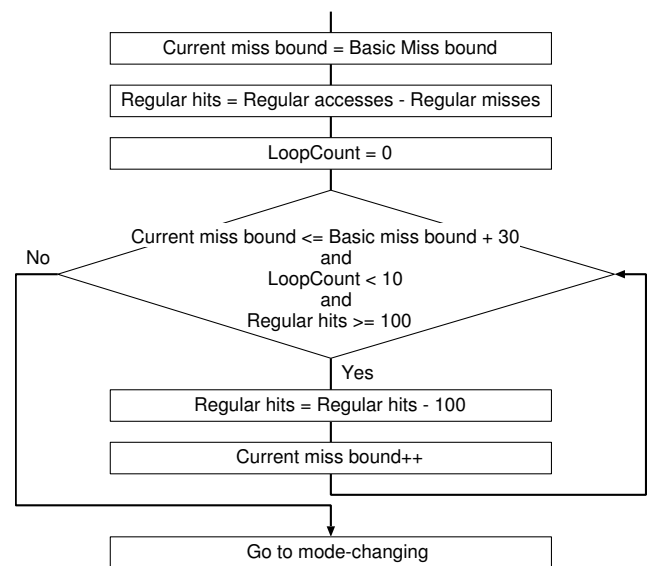


図10 動的ミス閾値制御のフローチャート

動的ミス閾値制御では、最初に一定サイクルごとに計測したアクセス回数とミス回数よりヒット数を求める。次に、一定サイクルごとのヒット数が100回に到達するごとに、閾値を1%増加する。ミス閾値は一度に最大10%まで、全体で基本の閾値から最大30%まで増加できる。ヒット数分増加し終わるか、最大値に達するまで、この処理を繰り返す。図10の動的ミス閾値制御を終えたら、モード切り換え判定に移る。動的ミス閾値制御は上方・下方・通算の3つのミス閾値全てに適用させた。

5.3 動的ミス閾値制御の利点

動的ミス閾値制御では一定サイクルごとのミス率が高い場合、ミス閾値はあまり増加しない。反対にミス率が低い場合、ミス閾値は急激に上昇する。よって従来の可変レベルキャッシュではミス閾値を超えないミス率でも、増加したミス閾値を超え、モード切り換えを行う。ミス閾値を増加させると、下位のモードへの切り換えが、上位のモード

への切り換えより優位になる。その結果省電力なモードでいる時間が長くなり、低電力化でき、さらに余計なモード切り換えを防げる。

6. 提案手法の性能評価

6.1 評価結果

評価は通常キャッシュ、通常の可変レベルキャッシュ、3つの提案手法を導入した可変レベルキャッシュで行う。評価環境は文献 [3] と同じとする。評価項目は低消費電力と高性能の両立を目標とするため、消費電力と実行時間とする。消費電力は文献 [3] と同じ方法で計算する。実行時間はシミュレーションにかかった総サイクル数とする。ベンチマークは SPEC2000[7] より 9 種類を使用した。

今回の評価に使用したキャッシュの仕様を表 1 にまとめる。

表 1 評価に用いた可変レベルキャッシュの仕様

キャッシュ容量	512 KB
セット数	2048
ウェイ数	4
ライン長	64 bit
キャッシュレイテンシ	10 cycle
メモリレイテンシ	250 cycle
ライトバックレイテンシ	1 cycle
再アクセスレイテンシ	10 cycle
交換レイテンシ	30 cycle
スリープからの復帰レイテンシ	10 cycle
シャットダウンからの復帰レイテンシ	10 cycle
モード切り換え判定を行う間隔	8192 cycle
シャットダウン判定を行う間隔	8192 cycle
モード切り換え後シャットダウンしない間隔	16384 cycle
通算ミス率用のミス閾値	50 %
下方のミス閾値	40 %
上方のミス閾値	70 %
ミス閾値の最大増加幅	30 %
ミス閾値を増加する基準のヒット数	100 回

通常のキャッシュ、従来の可変レベルキャッシュと、提案手法を導入した可変レベルキャッシュの消費電力比を図 11 に、実行時間比を図 12 に示す。なお、消費電力、実行時間ともに通常キャッシュの結果で正規化した。

通常のキャッシュと比較して、消費電力が平均で 52.88 % と大幅に減少し、実行時間が平均でわずか 1.22 % 増加した。また通常の可変レベルキャッシュと比較して、消費電力が平均で 14.75 %、実行時間が平均で 0.32 % 減少した。

6.2 考察

消費電力の大幅削減と Mcf2 の実行時間の原因を探るため、可視化ツールで挙動を確認したところ、全てのベンチマークにおいて、モード 3 でシャットダウンできる領域は

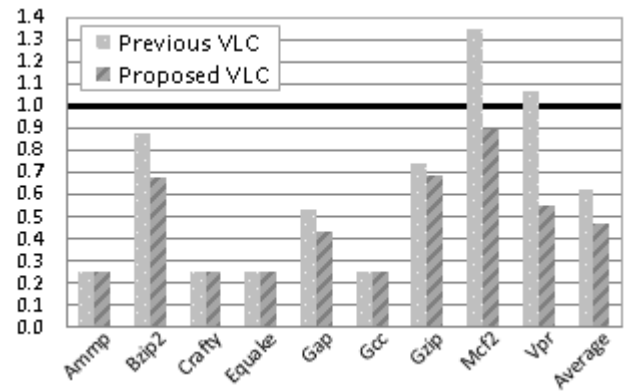


図 11 消費電力比

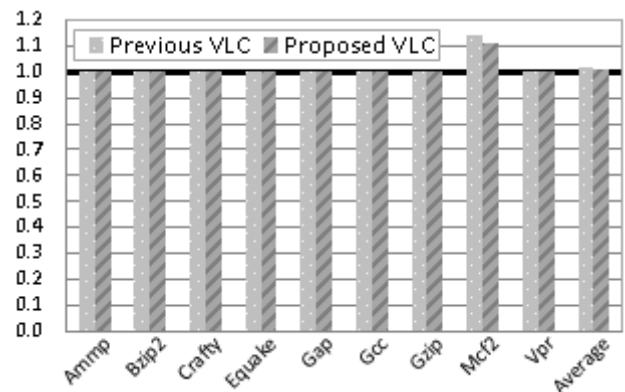


図 12 実行時間比

全てシャットダウンしている状態が実行時間の大部分を占めていることが判明した。

ミス閾値を増加させると、下位のモードへの切り換えが、上位のモードへの切り換えより優位になる。かわりにキャッシュ容量が小さくなるため、全体のミス回数が増え、その分のレイテンシが増大する。しかし最下位のモードでシャットダウンしている場合、キャッシュに必要なデータがないことがわかっているため、余分なアクセスが発生せず、再アクセス回数と交換回数が減少する。

Mcf2 を除くベンチマークでは、アクセス間隔がミスレイテンシより長い場合、ミス回数が増えても実行時間はあまり増加しない。しかし Mcf2 はアクセス間隔がミスレイテンシより短い場合、ミス回数が増えるほど、実行時間が大幅に増加する。その結果、Mcf2 を除くベンチマークでは大幅な電力削減をしつつ実行時間を保てた一方で、Mcf2 では大幅な電力削減ができたが、実行時間が大きく増加してしまった。

7. まとめ

低消費電力キャッシュの手法として、新たにシャットダウン状態の再導入、通算のミス率によるモード切り換えアルゴリズムと、動的ミス閾値制御の 3 つの手法を提案した。提案手法を導入した可変レベルキャッシュは消費電力を通常キャッシュと比較して平均 52.88 %、従来の可変レベル

キャッシュと比較して平均 14.75 %削減できるとわかった。一方で実行時間は通常キャッシュと比べて平均 1.22 %増加し、Mcf2 ベンチマークにおいては 11.00 %も増加する。これは提案した動的ミス閾値制御ではミス閾値を増やすことしかできないため下位のモードに移行しやすくなり、かえってミスが増加するためである。ミスが増加するとレイテンシが増大し、実行時間が急激に増加する。

今後の展望としてモード切り換えアルゴリズムや動的ミス閾値制御の改良、ハードウェア記述言語での動的ミス閾値制御の実装があげられる。また、従来の可変レベルキャッシュはマルチスレッド・マルチコア環境における可変レベルキャッシュの実装と評価 [3][8] が不十分であるため、完全な実装と評価も望まれる。

謝辞 本研究は JSPS 科研費 2470047, 15K00074 の助成を受けたものである。

参考文献

- [1] K.Ando, S.Fujita, S.Yuasa, Y.Nakatani, T.Miyazaki, and H.Yoda, 'Spin-transfer torque magnetoresistive random-access memory technologies for normally off computing (invited)', *Journal of Applied Physics* 115 172607, April, 2014.
- [2] 中村宏, 中田尚, 三輪忍, “ノーマリーオフコンピューティング:1. ノーマリーオフコンピューティング-期待と課題-”, *情報処理* 54(7), pp.654-660, June, 2013.
- [3] 渡部功, 佐々木敬泰, 松原伸幸, 大野和彦, 近藤利夫, “モード切替オーバヘッドを低減した可変レベルキャッシュの提案と評価”, *情報処理学会論文誌*, vol.50, No.2, pp.1234-1247, February, 2006.
- [4] Huifang Qin, Yu Cao, Dejan Markovic, Andrei Vladimirescu, and Jan Rabaey, 'SRAM Leakage Suppression by Minimizing Standby Supply Voltage', Department of EECS, University of California at Berkeley, Berkeley CA 94720, 2004.
- [5] S.H. Yang M.D Powell B. Falsafi K. Roy and T.N. Vijaykumar 'An Integrated Circuit / Architecture Approach to Reducing Leakage in Deep-Submicron High-Performance I-Caches' *International Symposium on High-Performance Computer Architecture*, pp.147-157, January, 2001.
- [6] Ying Zheng, Brian T. Davis, Matthew Jordan, 'Performance Evaluation of Exclusive Cache Hierarchies', *IEEE International Symposium of Performance Analysis of Systems and Software, ISPASS*, pp.89-96, September, 2004.
- [7] 'SPEC -Standard Performance Evaluation Corporation-', <http://www.spec.org/>, 参照 July, 2015.
- [8] 城田幸利, 佐々木敬泰, 大野和彦, 近藤利夫, “可変レベルキャッシュ用モード切替手法のマルチコア環境への適用と評価”, *情報学報*, 2010-ARC-190 No.12, pp.1-8, August, 2010.