

卒業論文

題目

セル・アロケーションキャッシュの
低電力化手法の提案と評価

指導教員

佐々木 敬泰 助教

2017年

三重大学 工学部 情報工学科
コンピュータアーキテクチャ研究室

白井 翔梧 (411824)

内容梗概

近年，性能維持と低電力を両立したプロセッサが求められている．プロセッサの高性能化手法の一つとして，並列化手法であるマルチコアプロセッサが普及している．しかし並列プロセスを実行することで，シングルコアプロセッサと比較してメモリアクセスが増加する．また，回路の微細化に伴うトランジスタ数の増加や，高性能化のために絶縁膜を薄くした結果，消費電力も増加する．メモリアクセスはキャッシュへのアクセスと比較して低速であるため，性能のボトルネックとなり，これを回避するにはキャッシュの性能向上が必要である．そのため，キャッシュをより効率的に扱うことで性能向上を図る研究が行われている．この研究の一つとしてコアがキャッシュにアクセスする領域を割り当て，制限するキャッシュパーティショニングと呼ばれる手法が存在する．キャッシュパーティショニングはセットアソシアティブキャッシュにおけるメモリアクセス手法の一つであり，各コアにアクセス可能なウェイを割り当て，各コアが扱うデータの場所を制限する．また，各コアの負荷に応じて割り当てるウェイ数を動的に制御することで，タスクに対して適切なキャッシュ容量を割り当てることが可能となる．しかし，割り当て単位がウェイであるため，タスクのメモリ要求的に全てのコアが必要としないウェイが存在する可能性がある．そこで，不要なウェイをシャットダウン状態にするウェイ・アロケーションキャッシュが存在する．ウェイ・アロケーションキャッシュは，不要なウェイを電力を必要としないシャットダウン状態にして性能低下を抑えつつ，消費電力の削減を行う．しかし，共有データを扱えないという問題がある．加えて，ウェイを単位とした割り当てのみであるため，各コアへの割り当てが最適でない場合がある．そこで共有データを扱うことができ，かつウェイをさらに細かく分割した「セル」という単位での割り当てを行い，より細かい領域で管理し，高性能化を図るセル・アロケーションキャッシュが提案されている．しかし，セル・アロケーションキャッシュには未使用セルが電力を無駄に消費し続けているという問題がある．そこで本稿では，セル・アロケーションキャッシュへシャットダウン・スリープ機能の追加手法を提案する．提案手法を評価した結果，従来のセル・アロケーションキャッシュと比較して，消費電力を平均で16.16%削減できた．

Abstract

Multi-core processor is common technique for high computing performance. In many multi-core processor architectures, all processors share L2 and last level cache memory. Thus, a performance of an entire multi-core processor depends strongly on a performance of shared cache memory. In particular, miss rate of shared cache memory is one of the most important factor because every processor needs to wait for 100 to 1000 clock cycles when an access-miss occurs on shared cache memory. In addition, multi-core processor is a core and a program in which required data and allocated locations on the cache memory are different. Thus, in order to reduce the number of access misses, the temporal and spatial locality on shared cache memory, which is the most important concept of memory, is impaired. As one of these researches, there is a method called cache partitioning that allocates and restricts cache spaces where a core can access. Cache partitioning is one of memory access methods in set-associative caches, assigns ways accessible to each core, limits the location of data handled by each core can do. In addition, load of each core allocate according to dynamically controlling the number of way, it is possible to allocate an appropriate cache capacity to the task. However, since the allocation unit is a way, there may be ways that do not require all the cores in terms of memory requirements of tasks. As a previous study, there is a way allocation that makes unused ways unallocated. Way allocation reduces power consumption while suppressing performance degradation by setting unassigned ways in an inactive state that does not require electric power. However, way allocation and cache partitioning have the problem that shared data can not be handled. In addition, since assignment is based only on ways, allocation to each core is not optimal in many cases. Therefore, a cell allocation cache has been proposed in which shared data can be handled, and cells are allocated in units of finely divided ways, managed in finer areas and enhanced. However, there is a problem that unused cells continuously waste power in the cell allocation cache, so further improvements are still needed in terms of lower power consumption. In this paper, we propose a method to

add shutdown sleep function to cell allocation cache. Compared with the conventional cell allocation cache, the proposed method reduces 16.16% of power consumption on average.

目次

1	はじめに	1
2	先行研究と問題点	4
2.1	キャッシュパーティショニング	4
2.2	ウェイ・アロケーションキャッシュ	6
2.3	先行研究の問題点	8
2.3.1	共有データ非対応	8
2.3.2	不適切な割り当て方式	9
3	セル・アロケーションキャッシュ	12
3.1	セル・アロケーションキャッシュの概要	12
3.2	共有データへの対応	14
3.3	割り当て方式の最適化	15
4	シャットダウン・スリープ機能追加手法の提案	17
4.1	提案手法の概要	17
4.2	モード切り換え方法	18
5	性能評価	21
5.1	評価環境	21
5.2	評価方法	21
5.3	評価結果	25
5.4	考察	27
6	おわりに	28
	謝辞	30
	参考文献	31

目 次

2.1	セットアソシアティブキャッシュ	4
2.2	同負荷におけるキャッシュパーティショニング	5
2.3	異なる負荷におけるキャッシュパーティショニング	6
2.4	ウェイ・アロケーション	7
2.5	共有データにおける問題	8
2.6	ウェイ割り当てによる割り当て図	9
2.7	理想的な割り当て図	10
3.8	セル・アロケーションキャッシュの概念図	13
3.9	セル・アロケーションにおけるアクセス制限	14
3.10	セル・アロケーションにおける割り当て	16
4.11	モード切り換え概要	17
4.12	ミス率によるモード切り換え	19
5.13	消費電力比	26
5.14	実行時間比	27

表 目 次

5.1 評估環境	21
--------------------	----

1 はじめに

近年，スマートフォンなどのモバイル端末のプロセッサの高性能化に伴い，消費エネルギーが増大し，バッテリーの駆動時間が短くなるという問題が発生している．そこで，性能の維持と低電力化の両立が要求されている．一般にプロセッサで消費されるエネルギーは，動的エネルギーとリークエネルギーに分けられる．動的エネルギーとはトランジスタのスイッチングによって消費されるエネルギーである．リークエネルギーとは主にトランジスタの漏れ電流（リーク電流）によって引き起こされ，トランジスタのスイッチングに関係なく消費されるエネルギーである．回路の微細化に伴い動的エネルギーが削減される一方，リークエネルギーの割合が増加していたが，近年では割合は同等程度になっている．キャッシュはプロセッサの面積の大半を占めているため，キャッシュのリークエネルギーを削減することはプロセッサの低電力化に繋がる．そこで本研究ではリークエネルギー削減に着目する．リークエネルギーはトランジスタ数に比例するため，プロセッサの高性能化にともない増大したキャッシュシステムのリークエネルギーの削減が重要となっている．そこで，キャッシュをより効率的に扱うことでキャッシュミス削減し，性能向上を図る様々な研究が行われている．

キャッシュミスの削減手法の研究の一つとして、各コアがキャッシュにアクセス可能な領域を割り当て、制限するキャッシュパーティショニングが存在する。キャッシュパーティショニングはマルチコア環境において、各コアで実行されているタスクが異なるため、必要なキャッシュ容量が違うことを利用する。各コアの負荷に応じて、アクセス可能なキャッシュ領域を動的に各コアに割り当てることにより、各コアで必要なキャッシュ容量を割り当てるのが可能となるため、メモリアクセスが削減できる。しかしキャッシュパーティショニングによる割り当てでは、実際のタスクが求めるキャッシュ容量に対し、ウェイと分配粒度が大きいことから、キャッシュを最大限利用できない。そこで、当研究グループでは共有キャッシュの領域をウェイより小さな単位である「セル」に分割し、管理するセル・アロケーションキャッシュを提案している。しかし、セル・アロケーションキャッシュには未使用セルが電力を無駄に消費するという問題があるため、低電力化する必要がある。そこで本研究では、セル・アロケーションキャッシュを改良しシャットダウン・スリープ機能を組み込み、低電力化を図る。シャットダウンが行われている間は電力を消費しない。しかし、シャットダウンが行われるとその部分のデータは失われてしまう。スリープ状態は電力を使い続けているが、データを保持できる最低電力状態で

ある．提案手法を評価した結果，従来のセル・アロケーションキャッシュと比較して，消費電力を平均で 16.16%削減できた．

2 先行研究と問題点

2.1 キャッシュパーティショニング

データ競合を回避する基礎技術として，セットアソシアティブキャッシュがある．図 2.1 に，セットアソシアティブキャッシュを示す．

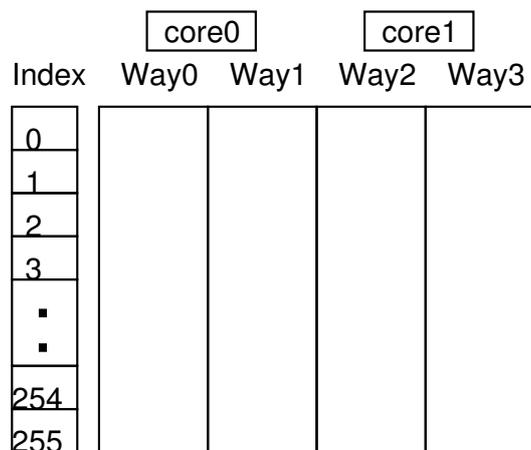


図 2.1: セットアソシアティブキャッシュ

セットアソシアティブキャッシュは，ウェイと呼ばれる領域を複数持つキャッシュ構造である．アドレスを A ，キャッシュメモリが N ブロックとすると，データを配置するブロック，すなわちインデックスは $A \bmod N$ という式で計算する．実際のキャッシュではインデックスは $4k$ から $16k$ エントリ程度あるため，より膨大なエントリ数を持つ．データアクセス時のアドレスが N の倍数に偏っていた場合，同一ブロックに割り当てが集中するため著しく性能が低下する．通常はキャッシュメモリを複数ウェ

イ分用意することにより、同一インデックスのデータを同時に複数格納可能としている。各コアではそれぞれ異なるタスクを実行するため、要求されるメモリ要求が異なる。キャッシュパーティショニングは各コアにアクセス可能なキャッシュ領域を動的に割り当て、各コアが扱えるキャッシュ領域を制限する。これにより、適切なキャッシュ容量を割り当てることができる。割り当ては各コアの負荷に応じて必要なキャッシュ領域の割合を決定し、各コアにウェイを単位として割り当てる。しかしセット・アソシアティブキャッシュはマルチコア環境に適していない。

マルチコア環境におけるメモリアクセス手法の一つであるキャッシュパーティショニングは、セットアソシアティブキャッシュのウェイを利用する。図 2.2、図 2.3 にキャッシュパーティショニングを用いた場合における、各コアへのキャッシュ容量の割り当て図を示す。

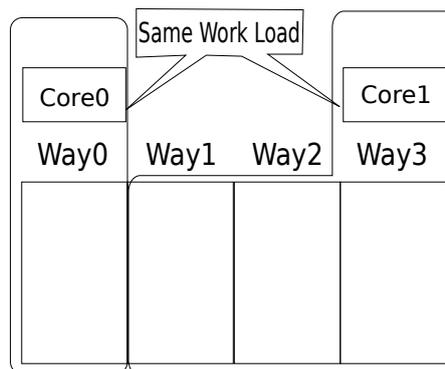


図 2.2: 同負荷におけるキャッシュパーティショニング

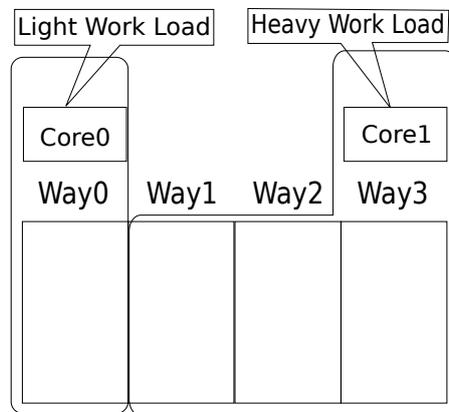


図 2.3: 異なる負荷におけるキャッシュパーティショニング

例として、各コアの負荷が同程度の場合は図 2.2 のように同程度のウェイ数が割り当てられる。各コアの負荷が大きく異なる場合、例えばコア 0 の負荷が小さく、コア 1 の負荷が大きい場合には図 2.3 のように、コア 1 の方へウェイ数を多く割り当てる。各コアに適切なキャッシュ容量を動的に割り当てることで、キャッシュ領域を効率的に扱え、キャッシュの未使用領域の削減に繋がり、全体のキャッシュミス削減できる。

2.2 ウェイ・アロケーションキャッシュ

ウェイ・アロケーションキャッシュはキャッシュパーティショニングを応用した手法である。図 2.4 にウェイ・アロケーションを用いたキャッシュ容量の割り当てを示す。

図 2.4 の灰色部分は電力を消費しないシャットダウン状態である。ウェ

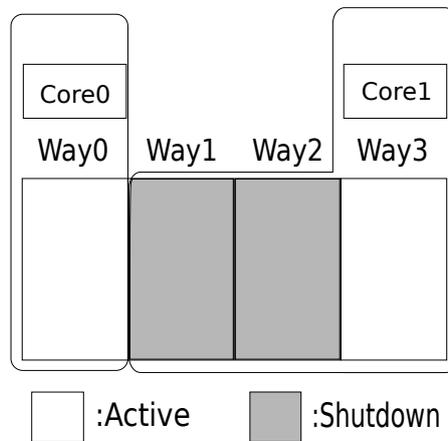


図 2.4: ウェイ・アロケーション

イ・アロケーションのアクセス方法はキャッシュ・パーティショニングと同様であるが、ウェイの割り当てにシャットダウン状態が存在する。しかし、第 2.1 節で示したキャッシュ・パーティショニングの手法の場合、図 2.2 のようにウェイを均等に割り当てる状態になることが考えられる。すなわち、図 2.2 の割り当てでは、キャッシュ容量が過剰である。そこでウェイ・アロケーションキャッシュは割り当てを最低限に抑え、かつ未割り当ての領域を、図 2.4 のようにシャットダウン状態にする。シャットダウンが行われている間は電力を消費しない。これにより、性能低下を抑えつつ消費電力を削減できる。

2.3 先行研究の問題点

2.3.1 共有データ非対応

先行研究は各コアがアクセスできる領域を制限することで、キャッシュをより効率的に使用しているが、共有データを考慮していないという問題がある。そのため共有データを使用するプログラムを実行する場合に問題が発生する。図 2.5 に先行研究におけるコア 0 のデータアクセスの制限を示す。

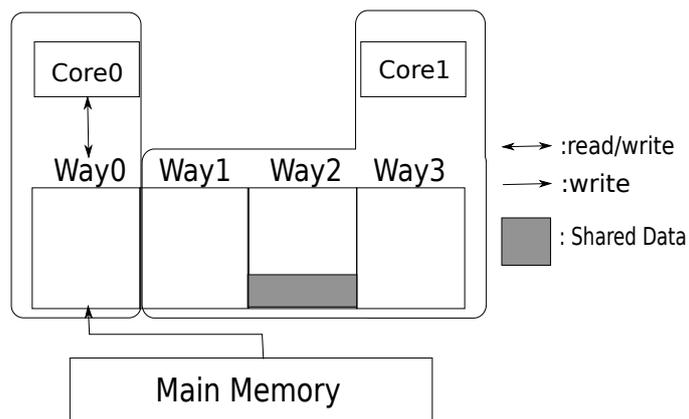


図 2.5: 共有データにおける問題

先行研究は図 2.5 における矢印の指す部分にのみアクセスできる。そのため、図 2.5 から、異なるコアへ割り当てられているキャッシュ領域にアクセスが不可能である。そこで例として、2 コアプロセッサにおいてキャッシュパーティショニングを用いた構成を考える。図 2.3 のようにキャッシュ

容量を割り当てた場合，コア 0 はコア 1 へ割り当てられた領域へアクセスすることができないため，図 2.5 のようにウェイ 1 からウェイ 3 へのアクセスはできない．また，図 2.5 のように共有データがウェイ 2 に存在する場合，コア 0 はウェイ 2 にアクセスできないため，共有データがキャッシュに格納されていても使用できない．共有データの使用不可は，近年のマルチコアやマルチスレッドのようにデータ共有が発生しうる並列処理において大きな問題である．

2.3.2 不適切な割り当て方式

先行研究は図 2.1 のように，ウェイ単位での割り当て変更を行う．

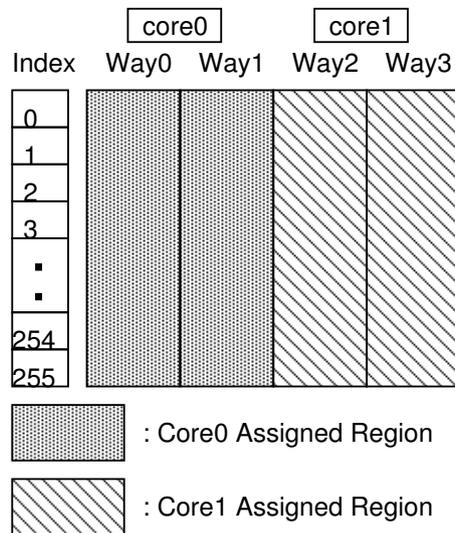


図 2.6: ウェイ割り当てによる割り当て図

しかし，プログラムごとにインデックスに偏りが生じる可能性が十分

にある．その場合，必要外のキャッシュ領域が割り当てられたり，本当に必要とするキャッシュ領域が割り当てられない．図 2.7 に理想的な割り当てを示す．

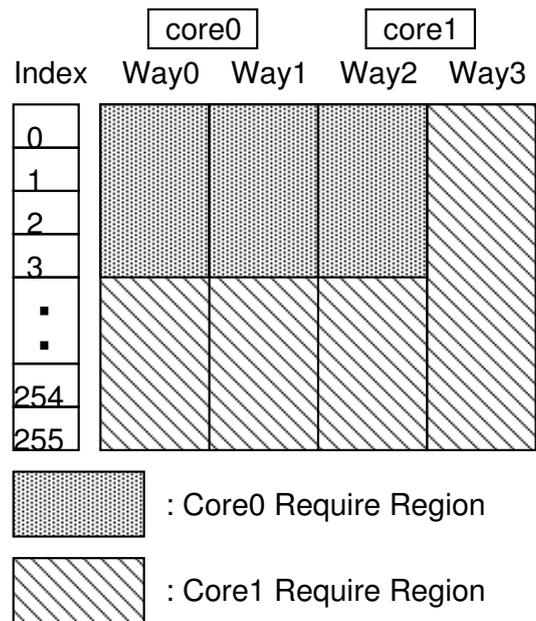


図 2.7: 理想的な割り当て図

例として図 2.6 のようなキャッシュ構成を考える．この場合コア 0 がキャッシュメモリの前半に集中しており，コア 1 はそれ以外の領域でアクセスが集中していると，各コアの割り当ては図 2.7 が望ましいが，キャッシュパーティショニングはウェイ単位での割り当てを行っているため，全体の負荷が同程度の場合，実際に可能な割り当ては図 2.6 である．各コアがアクセスの集中するインデックスを最大限使えないため，最適な割り

当てが出来ていない．そのため，ウェイ単位では最適な割り当てが不可能である．

3 セル・アロケーションキャッシュ

3.1 セル・アロケーションキャッシュの概要

先行研究は各コアがアクセスできる領域を制限し、キャッシュをより効率的に使用しているが、共有データを考慮していない問題がある。そのため、共有データを使用するプログラムを実行する場合に問題が発生する。例えば、異なるコアへ割り当てられているキャッシュ領域にアクセスできないため、プログラムを正常に実行できないことが考えられる。共有データの使用不可は、近年のマルチコアやマルチスレッドのようにデータ共有が発生しうる並列処理において大きな問題である。また、先行研究はキャッシュ領域の割り当てのためにウェイ単位での割り当て変更を行う。しかし、プログラムごとに扱うデータは異なることから、インデックスに偏りが生じる可能性がある。そのため、ウェイ単位では最適な割り当てが不可能である。

この問題に対して、共有データを扱うことが可能であり、かつより細かな領域を割り当てられるセル・アロケーションキャッシュ[1]が提案されている。セル・アロケーションキャッシュはウェイを細分化した領域をセルとし、各コアへの割り当てをセル単位で行い、インデックスを考慮した割り当てにより性能向上を図る。セル・アロケーションキャッシュの

概念図を図 3.8 に示す .

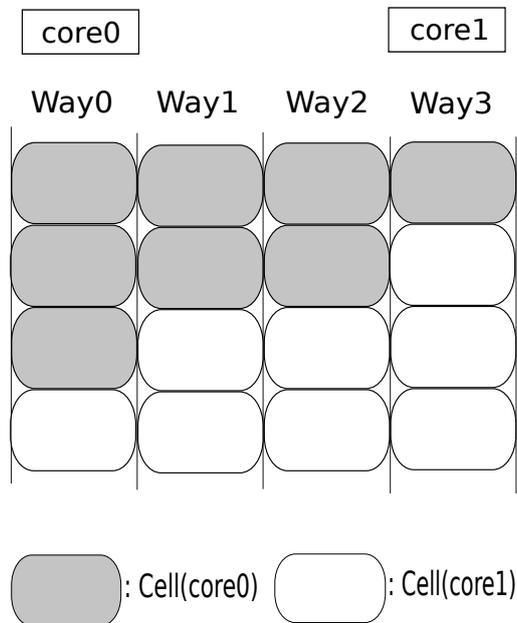


図 3.8: セル・アロケーションキャッシュの概念図

セル・アロケーションキャッシュはメモリからの書き込みの場合に限り、各コアに割り当てられたセルに対してのみ書き込みを行うように制限する。しかし、コアからの読み込みや書き込みのアクセスは制限することなく、通常のキャッシュと同様にアクセスを行える。これにより、共有データを扱うことを可能にしている。また、ウェイ単位による割り当てを行うため、求めるキャッシュ容量を適切に割り当てることができない問題に対して、割り当て領域をセル単位で行い、適切に割り当てる。共有デー

タへの対応については第 3.2 節で、割り当てについては第 3.3 節で詳しく説明する。

3.2 共有データへの対応

全ての読み込み・書き込み命令に制限を掛けるキャッシュパーティショニングとは異なり、セル・アロケーションはキャッシュパーティショニングの割り当てを無視するため、各コアからの読み込み・書き込み命令をキャッシュ全体にアクセスできる。図 3.9 にセル・アロケーションにおけるアクセス制限を示す。

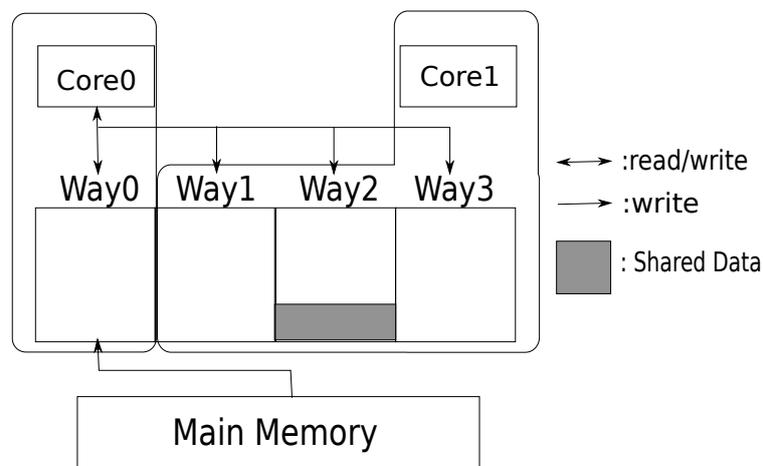


図 3.9: セル・アロケーションにおけるアクセス制限

図 3.9 のようにウェイ 2 に共有データが存在する場合、キャッシュパーティショニングではコア 0 からアクセスができない。それに対して、セ

ル・アロケーションではウェイ 2 に存在する共有データを扱える。ただし、メモリ側からキャッシュへ書き込む場合のみ、コアごとのアクセス制限を設ける。この変更により、先行研究では非対応であった共有データを扱える。

3.3 割り当て方式の最適化

セル・アロケーションはウェイ単位からさらに細分化した、セル単位で割り当てを行う。また、同一のインデックスを持つセル、すなわち行方向に存在する 4 つのセルで 1 つのセル群と呼ぶ。割り当て変更を行う際、最初に各コアのミス率を比較し、割り当て対象を決定する。

図 3.10 にセル・アロケーションにおける割り当ての一例を示す、

例えば、コア 1 のミス率が最も高く、コア 0 が最も低い場合、コア 0 からコア 1 に割り当てを変更する。割り当て対象を決定後、最もミス率の高いコアの中でミス数が最も多いセル群を探す。ミス数最大のセル群を発見後、同じ領域に存在するセル群で、最もアクセスの古いセルの割り当て先をコア 1 に変更する。この割り当て方式により、セル群ごとに異なる負荷に応じた最適な割り当てを実現する。このような手法による割り当て方式によって、図 3.10 のように、セル群ごとに異なる負荷に応じ

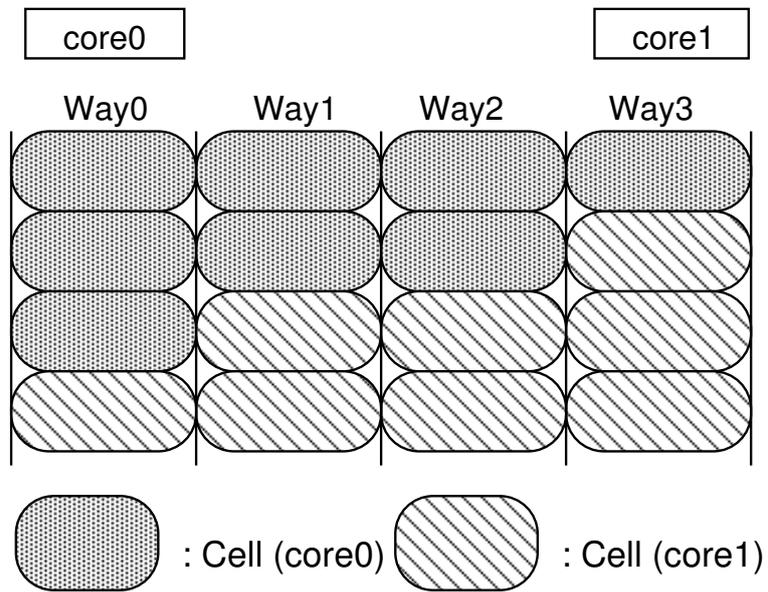


図 3.10: セル・アロケーションにおける割り当て

た最適な割り当てを実現できる。

4 シャットダウン・スリープ機能追加手法の提案

4.1 提案手法の概要

セル・アロケーションキャッシュは、通常のキャッシュと比較してミス率を低減することができる。しかし、未使用セルでは電力を無駄に消費し続けている問題がある。そこで本研究では従来のセル・アロケーションキャッシュの低電力化を目指す。具体的には、セル・アロケーションキャッシュを改良し、文献 [2] で提案されているようなモード切り換え手法を適用することで、未使用なセルで無駄に消費している電力を抑えることを目指す。以下、シャットダウン・スリープ機能追加手法の提案である。提案手法ではモードをミス率によって切り換える。図 4.11 に Mode1, Mode2, Mode3 の概要を示す。

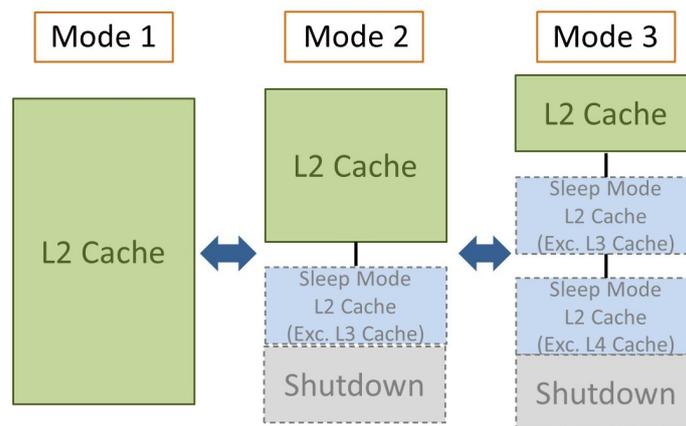


図 4.11: モード切り換え概要

すべてのラインがアクティブの時は Mode1 , キャッシュの下位 1/4 をシャットダウン , 上位 2/4 から 3/4 をスリープ状態にしている時は Mode2 , キャッシュの上位 1/4 から 2/4 をスリープ状態 , 上位 2/4 から 3/4 をスリープ状態 , 下位 1/4 をシャットダウンにしている時は Mode3 とする . また , Mode2 状態の下位半分を仮 L3 キャッシュ , Mode3 の上位 1/4 から 2/4 を仮 L3 キャッシュ , 下位半分を仮 L4 キャッシュとする . シャットダウンの部分では固定の閾値以上アクセスがない場合にシャットダウンし , またアクセスがあった場合に起動する . シャットダウンが行われている間は電力を消費しない . しかし , シャットダウンが行われるとその部分のデータは失われてしまう . スリープ状態は電力を使い続けているが , データを保持できる低電力状態である . そのため , シャットダウン・スリープを使った結果 , 消費電力を削減できる .

次節で具体的なモード遷移を示す .

4.2 モード切り換え方法

図 4.12 に , 全体のミス率によるモード切り換えを示す .

提案手法では , 最初に全体のミス率と全体の固定の閾値を比較する . 全体のミス率が全体の固定の閾値より小さく , 一定サイクルごとのアクセ

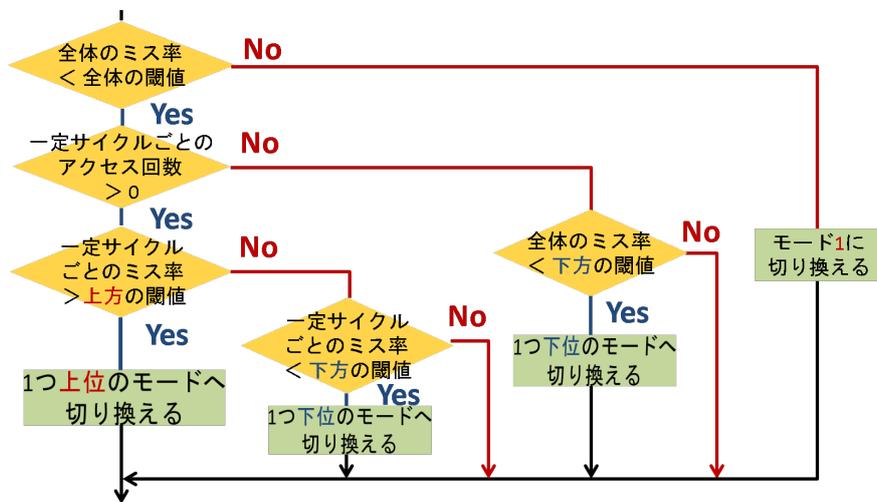


図 4.12: ミス率によるモード切り換え

ス回数が 0 より小さく、全体のミス率が動的な下方の閾値より小さかった場合、1 つ下位のモードへ切り換わる。またそこで下方の閾値より大きかった場合、モードは変わらない。全体のミス率が全体の固定の閾値より小さく、一定サイクルごとのアクセス回数が 0 より大きく、一定サイクルごとのミス率が動的な上方の閾値より小さく、一定サイクルごとのミス率が動的な下方の閾値より小さかった場合、1 つ下位のモードへ切り換わる。またそこで動的な下方の閾値より大きかった場合、モードは変わらない。

全体のミス率が全体の固定の閾値より小さく、一定サイクルごとのアクセス回数が 0 より大きく、一定サイクルごとのミス率が動的な上方の

閾値より大きい場合，1つ上位のモードへ切り換わる．

全体のミス率が全体の固定の閾値より大きかった場合，どのモードであっても Mode1 へ切り換わる．

具体例として，キャッシュの初期状態としてデータは何も格納されておらず，また2コアの場合を考える．Core0 で実行されているプログラムを，メモリアクセス数が多く，参照の局所性が高く，ミス率は低いものとする．また，Core1 で実行されているプログラムはメモリアクセス数が少なく，ヒット率は高いとする．この状態では，提案手法は図 4.12 のアルゴリズムに従い，YES に進む．ここで，Core1 のアクセスパターンが変わり，メモリアクセス数が増加したとする．この時，図 4.12 のアルゴリズムに従い，YES に進み1つ下位のモードへ切り換える．上記の結果，動的に閾値を変更することによって，ベンチマークや実行状況によって適切となるミス閾値を定めることができる．よって適切な結果が得られる．

5 性能評価

5.1 評価環境

当研究グループで開発しているトレースドリブン型キャッシュシミュレータに，提案手法を組み込んで評価を行った．評価項目は高性能と低電力の両立を目標とするため，消費電力と実行時間とする．また，比較対象は通常キャッシュと従来のセル・アロケーションキャッシュとする．ベンチマークプログラムには，共有メモリ型マシンのベンチマークプログラムとして広く用いられている Splash2[3] と姫野ベンチマーク [4] を使用した．その他の評価環境を表 5.1 に示す．

表 5.1: 評価環境

コア数	2 コア
L1 キャッシュ	32kB
L2 キャッシュ	256kB × 8 ウェイ

5.2 評価方法

当研究グループで開発し，以前のキャッシュシミュレータの電力評価等にも使われたパラメータを使用し消費電力等の計算を行った．以下に，詳しい評価方法を示す．

実行時間は提案手法を組み込んだキャッシュシミュレータを実装して評価した。

消費電力については、文献 [5], [6] を参考に以下の計算式を用いて計算した。提案手法における動的エネルギーは、L2 キャッシュへの通常アクセスとスリープモードからの復帰の 2 種類に分類される。通常アクセス E_{access} は、活性化状態のキャッシュへの 1 回のアクセスに必要なエネルギーを E_{one_access} 、キャッシュアクセスの総数を N_{access} とすると、下式で計算できる。

$$E_{access} = E_{one_access} \times N_{access} \quad (1)$$

スリープモード中のセル・アロケーションキャッシュにおいて 1 つのメモリアクセス命令を処理する場合、下記のとおり複数回のアクセスが必要になる。

1. アクティブ状態のセットにアクセス。
2. 活性化させたセット (疑似 L3) にアクセス。
3. 疑似 L3 でヒットした場合、ラインをスワップするために 2 回のアクセス。

N_{access} はこれらのアクセス回数の総和である .

$$E_{wakeup} = E_{one_wakeup} \times N_{way} \times N_{wakeup} \quad (2)$$

スリープモードからの復帰 E_{wakeup} は上記の様に計算され , この時 E_{one_wakeup} は , スリープ状態のセットを活性化するために必要なエネルギーと蓄積された電荷の持つエネルギーの総和である . N_{way} はキャッシュの連想度 , N_{wakeup} はスリープ状態のセットを活性化させた回数である . 今回の電力評価においては主な周辺回路以外の回路の電力は評価に入れていない . また , 活性化状態とスリープ状態との遷移時の電荷をチャージ・ディスチャージする際にかかるエネルギーは算出が困難なため , SRAM の最大周波数での動作時の動的エネルギーを用いた .

よって提案手法の動的エネルギー $E_{dynamic}$ は

$$E_{dynamic} = E_{access} + E_{wakeup} \quad (3)$$

となる .

次に静的エネルギーは , 提案手法のセット数を N_{set} , プログラム実行にかかったクロックサイクル数を $Cycles$, CPU のクロック周波数を $Freq$ とすると , 下記の計算式で計算される .

$$E_{static} = E_{set_leak} \times N_{set} \times N_{way} \times Cycles \div Freq \quad (4)$$

また，1セットにおけるリークエネルギー E_{set_leak} は下記の計算式で計算される．

$$E_{set_leak} = R_{sleep} \times E_{sleep_leak} + (1 - R_{sleep}) \times E_{active_leak} \quad (5)$$

ここで， R_{sleep} は提案手法のスリープモードに落ちているラインの平均割合， E_{sleep_leak} ， E_{active_leak} はそれぞれスリープ状態と活性状態での1ラインにおけるリークエネルギーである．

この評価式において，通常キャッシュにおいてはスリープモードが存在しないため $E_{dynamic} = E_{access}$ とし， $E_{static} = E_{active_leak}$ となる．

最後に，提案手法の電力については提案手法実装に必要な追加回路の電力も考慮に入れるため，消費電力の総和に以下のエネルギーを加算する．

$$E_{P_circuit} = E_{controller} \times N_{access} + E_{buffer} \times N_{wakeup} \quad (6)$$

ここで， $E_{P_circuit}$ はL2 キャッシュに追加した周辺回路のエネルギーの総和， $E_{controller}$ はモード切替コントローラが1回の動作に必要なエネルギー， E_{buffer} はビクティムバッファへのデータ待避及び読出しを1回ずつ実行する際に必要なエネルギーである．周辺回路のリークエネルギーは非常に小さく，個別の評価が困難なため，考慮しないものとする．その他の追加回路，具体的にはミスカウンタ，再アクセスユニットについて

は、Verilog HDL を用いたハードウェア設計を行い、トランジスタレベルでの電力シミュレーション評価の結果、コントローラ部分とピクティムバッファの消費電力と比較して 0.1% 未満であることがわかったため、電力評価モデル式では考慮しないものとする。

なお、L2 キャッシュからメインメモリへの書戻しの際の DRAM やバスの電力は測定が困難であるため評価に入れないものとする。また、評価式中のスリープ状態と活性状態との遷移に必要なエネルギーは算出が困難なため、仮定値として CACTI で求められる最大周波数での動作時の動的エネルギーの値を用いた。ただし、この仮定値を 10 倍、100 倍、1000 倍に増やしたことによる結果全体への影響は 0.01% 以下とごく微少であり、電力見積りに与える影響は十分に小さいといえる。

以上の式で求めた、実行サイクル数と消費電力の積を求め比較を行う。

5.3 評価結果

従来のセル・アロケーションキャッシュと提案手法の消費電力を、通常キャッシュで正規化した評価結果を図 5.13 に示す。

従来のセル・アロケーションキャッシュと比較すると、消費電力を平均で 16.16% 削減した。また、従来のセル・アロケーションキャッシュと提

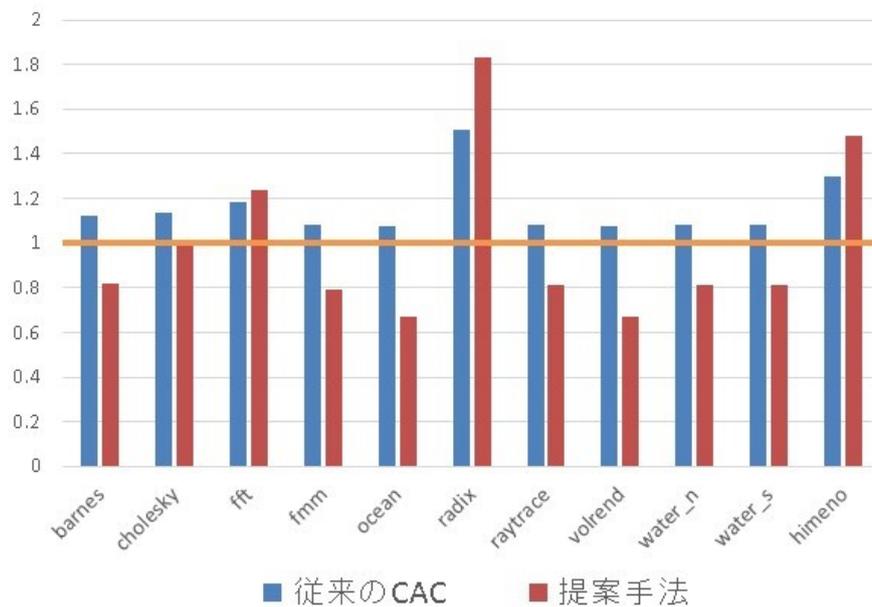


図 5.13: 消費電力比

案手法の実行時間を、通常キャッシュで正規化した評価結果を図 5.14 に示す。

実行時間は一部のベンチマークを除きほぼ変化していないが、平均では 15.12%増加した。考えられる原因として、シャットダウンと起動が頻繁に行われるとその分時間がかかることや、また閾値の設定の仕方が適切でない部分があるなどが挙げられる。

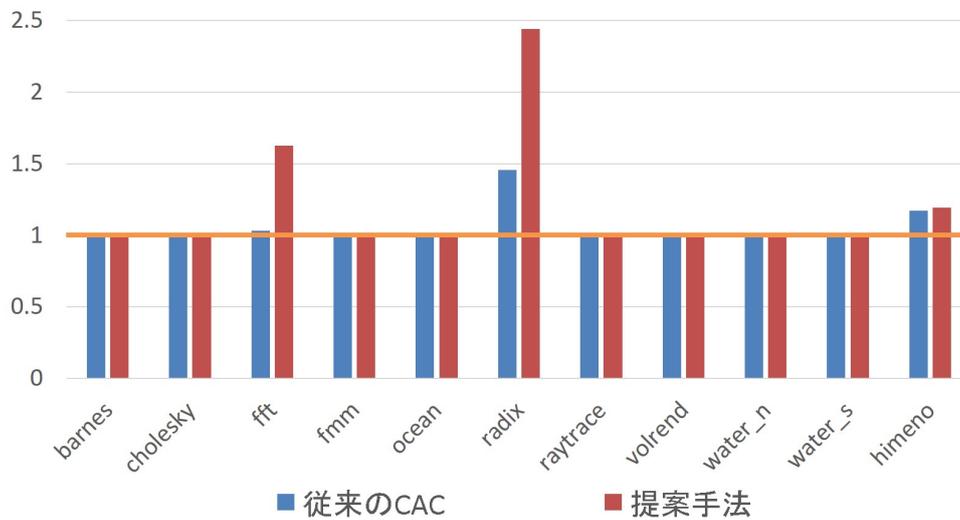


図 5.14: 実行時間比

5.4 考察

従来のセル・アロケーションキャッシュと比較すると、提案手法は消費電力を平均で 16.16%削減できた。一方、実行時間はほとんどのベンチマークで変化していないが、fft、radix など大幅に増加するベンチマークも存在した。考えられる原因として、シャットダウンと起動が頻繁に行われるとその分時間がかかることや、また閾値の設定の仕方が適切でない部分があるなどが挙げられる。これらの改善を図ることで、さらに性能が向上すると考えられる。

6 おわりに

近年，スマートフォンなどのモバイル端末のプロセッサの高性能化に伴い，消費エネルギーが増大し，バッテリーの駆動時間が短くなるという問題が発生している．そこで，性能の維持と低電力化の両立が要求されている．一般にプロセッサで消費されるエネルギーは，動的エネルギーとリークエネルギーに分けられる．リークエネルギーはトランジスタ数に比例するため，プロセッサの高性能化にともない増大したキャッシュシステムのリークエネルギーの削減が重要となっている．そこで，キャッシュをより効率的に扱うことでキャッシュミス削減し，性能向上を図る様々な研究が行われている．共有キャッシュの領域をウェイより小さな単位である「セル」に分割し，管理するセル・アロケーションキャッシュが当研究室で提案されている．しかし，セル・アロケーションキャッシュには未使用セルが電力を無駄に消費し続けているという問題があるため，低電力化という点ではまだ改良が必要である．そこでセル・アロケーションキャッシュへのシャットダウン・スリープ機能追加手法を提案・評価した．その結果，従来のセル・アロケーションキャッシュと比較して，消費電力を均で 16.16%削減した．また，実行時間は一部のベンチマークを除きほぼ変化していないが，平均では 15.12%増加した．今後の課題として，モー

ド切り換え手法のさらなる改良，スリープ・シャットダウンの改善などが
あげられる．また，ソフトウェアシミュレーションでの評価であるため，
ハードウェアでの実装を行う必要がある．

謝辞

本研究の機会を与えて頂いた近藤利夫教授，並びにご指導，頂いた佐々木敬泰助教，深澤祐樹研究員，修士2年の刀根舞歌先輩にこの場を借りてお礼申し上げます．

参考文献

- [1] 刀根舞歌, 佐々木敬泰, 深澤祐樹, 近藤利夫 'キャッシュの分割領域の動的管理による高速化', 信学技報, Vol.CPSY2016-20, pp.119-124 August 2016.
- [2] 近藤舞佳, “キャッシュの可視化ツールを用いた可変レベルキャッシュの改良,” 三重大学卒業論文, 2014年3月.
- [3] S. C. Woo, et. al.: 'The SPLASH-2 programs: Characterization and methodological considerations', ACM SIGARCH Computer Architecture News. Vol. 23. No. 2. ACM, 1995.
- [4] 姫野龍太郎, 姫野ベンチマーク, <<http://accr.riken.jp/supercom/himenobmt/>> (2017年2月20日アクセス)
- [5] 小宮礼子, 井上弘士, モシニヤガ・ワシリー, 村上和彰, “キャッシュ・リーク電力削減アルゴリズムに関する定量的評価,” 第17回回路とシステム軽井沢ワークショップ論文集, pp.235-240, April 2004.
- [6] 関子純平, 富山宏之, 高田広章, 井上弘士, “Drowsy キャッシュにおけるモード切替アルゴリズムの評価,” 情処学研報, 2006-ARC-170,

pp.37-41, December 2006.