データの重要度を利用したキャッシュメモリの省電力化

千代延, 昭宏
九州工業大学大学院情報工学研究科

藤井, 誠一郎
日立アドバンスドデジタル

佐藤, 寿倫
九州大学システムLSI研究センター

出版情報：情報処理学会論文誌. 48 (SIG8), pp.114-126, 2007-05-15. 情報処理学会
バージョン：

データの重要度を利用したキャッシュメモリの省電力化

千代延 昭宏[†] 藤井 誠一郎[††] 佐藤 寿倫[†††]

データの重要度を利用する省電力キャッシュメモリについて検討する．本稿で検討するキャッシュメモリはレイテンシと消費電力の異なる領域を持つ．クリティカルパス情報や時間的局所性で決定されるデータの重要度を利用して，省電力を指向したデータの配置を行う．このキャッシュメモリに対し，読み出す領域を考慮したアクセスを行うことで，処理性能を維持しつつ省電力化を目指す．シミュレーションによる評価の結果，データの重要度決定には時間的局所性を用いる方が良いこと，全ての領域が高速な L1，L2 キャッシュメモリと比較して，平均で約 10%の処理性能低下で約 14%の ED^2P 削減を達成できることがわかった．

Low Power Cache Memories Prioritizing Data

Akihiro Chiyonobu,[†] Seiichiro Fujii[††] and Toshinori Sato[†††]

Considering the priorities of data could improve power efficiency of cache memories. The cache memories investigated in this paper have different ways regarding access latencies and power consumption. Two priorities based on criticality and locality are utilized to determine data location. Several cache access policies are evaluated to reduce power consumption with maintaining processor performance. The detailed simulations show that the priority based on locality is better in power efficient than that based on criticality and that 14% of the improvement in ED^2P is achieved with only 10% of performance loss.

1. はじめに

集積回路技術の進展により，プロセッサの処理性能は向上してきた．一方で，集積回路技術の進展はプロセッサ設計に新たな問題をもたらしている．それは消費電力である．動的消費電力のみならず，これまで無視できていた静的消費電力も非常に大きくなっている．このため，我々はプロセッサの処理性能だけでなく，その消費する電力を考慮に入れた設計を行う必要に迫られている．

プロセッサの動的消費電力 P_{active} は負荷の充放電によって消費する電力で，以下の式で表される．

$$P_{active} \propto f \times C_{load} \times V_{dd}^2 \quad (1)$$

ここで，f はクロック周波数，C_{load} は負荷容量，V_{dd} は電源電圧である．一方，静的消費電力 P_{leak} は半導体の特性上流れてしまうリーク電流によって発生する電力である．リーク電流にはゲートリーク，ジャンクションリーク，サブスレッショルドリークなどがあるが，本稿では対象をサブスレッショルドリークに限定する．リーク電流のうちサブスレッショルドリークによる静的消費電力は，以下の式で表される．

$$P_{leak} \propto 10^{-\frac{V_{th}}{S}} \times V_{dd} \quad (2)$$

ここで，S はサブスレッショルドファクタ，V_{th} はしきい値電圧を示す．プロセッサの動的消費電力の削減を図るならば，式 (1) の電源電圧を低下させることが効果的である．しかし，電源電圧の低下は次式 (3) に示すゲート遅延の増加を招く．

$$t_{pd} \propto \frac{V_{dd}}{(V_{dd} - V_{th})^{\alpha}} \quad (3)$$

式 (3) において α はキャリア移動度の飽和を表すパラメタである．ゲート遅延の増加はプロセッサの動作周波数低下につながり，処理性能が低下する．ゲート遅延の増大を抑えるには，しきい値電圧を同時に低くすることが考えられる．しかし式 (2) から容易にわかるように，しきい値電圧の低下は静的消費電力を飛躍的に増加させる．また，静的消費電力は回路の待機時にも発生する．したがって，動的消費電力だけでなく静的消費電力も削減する必要がある．

この問題に対して本研究では，プロセッサの消費電

† 九州工業大学大学院 情報工学研究科
Graduate School of Computer Science and System Engineering, Kyushu Institute of Technology
†† 日立アドバンスドデジタル
Hitachi Advanced Digital, Inc.
††† 九州大学 システム LSI 研究センター
System LSI Research Center, Kyushu University

力の大部分を占めるキャッシュメモリに着目している．動的消費電力については，従来通り電源電圧の削減によって省電力化を図る．一方静的消費電力については，リーク電流の削減によって省電力化を図る．リーク電流削減の方法の一つは，リーク電流の小さなトランジスタでキャッシュを構成する事である．しかしこの手法ではアクセスレイテンシが増大してしまう．そこで以下の手法が考えられる．低レイテンシなウェイの領域と高レイテンシなウェイの領域を用意する．低レイテンシなウェイの領域は，高い電源電圧が供給され，しきい値電圧の低いトランジスタで構成される．このためアクセス時の電力とリーク電流は大きい．一方，高レイテンシなウェイの領域は，低い電源電圧が供給され，しきい値電圧の高いトランジスタで構成される．このためアクセス時の電力とリーク電流が抑えられる．これらの領域のうち，低レイテンシな領域には重要度の高いデータを配置する．一方，重要度の低いデータは高レイテンシな領域に配置する．このようにデータ配置の最適化を行うことで処理性能を低下させずに動的消費電力と静的消費電力を削減可能になる[16]．

プロセッサのキャッシュメモリ以外の各ユニットの動的，静的消費電力を削減することも重要である．しかし既に様々な検討が行われていることを考慮して，本稿ではキャッシュメモリのみを省電力化の対象とする．

本稿の構成は以下の通りである．次章でデータの重要度を定義し，3 章でデータの重要度を利用する省電力キャッシュメモリについて説明する．4 章でキャッシュメモリ内のデータの移動について述べた後，5 章でキャッシュアクセスの方法について検討する．6 章で省電力キャッシュメモリを評価する．7 章で関連研究を紹介し，8 章でまとめる．

2. データの重要度

本稿で検討する省電力キャッシュアーキテクチャでは，データの重要度に応じてデータの配置場所を決める．よって，キャッシュメモリ内のデータの重要度を決める必要がある．データの重要度を決める方法として，時間的局所性に基づく方法とクリティカルパス情報に基づく方法とを検討する．

2.1 時間的局所性に基づく方法

時間的局所性に基づく方法では，プログラムの時間的局所性に着目する．プログラムには，最近アクセスしたデータに近い将来再びアクセスする可能性が高いという時間的局所性が存在する．最近アクセスされたデータほど再びアクセスされる可能性が高く，過去にアクセスされたデータになるほど再びアクセスされる可能性が低くなる．この特性に着目して，最近アクセスされたデータの重要度を過去にアクセスされたデータよりも高いとする．つまり，MRU(Most Recently Used) で重要度を定める．

2.2 クリティカルパス情報に基づく方法

クリティカルパス情報を用いてデータの重要度を決定する場合には，クリティカルパス上のメモリアクセス命令によってアクセスされたデータの重要度をそうでない命令によってアクセスされたデータよりも高いとする．

クリティカルパスとは命令間の依存関係を結んだ鎖のうち最長のものを結んだ実行パスであり，プログラムの実行時間を決定する命令列である[15]．図 1 に命令列中に現れるクリティカルパスを表すデータフローグラフの例を示す．図 1 において矢印は命令間の依存関係を示す．つまり，依存している命令は矢印の始点にある依存先の命令実行が終了しない限り実行できない．全ての命令のレイテンシが 1 サイクルであるとすると，最も長いパスである命令 I:0 → I:3 → I:4 → I:6 → I:7 を結んだパスがクリティカルパスとなる．実行中のプログラムのクリティカルパス情報を獲得する方法としてクリティカルパス予測器[5),13),17)] とパス情報テーブル (PIT:Path Information Table)[15)] が提案されている．

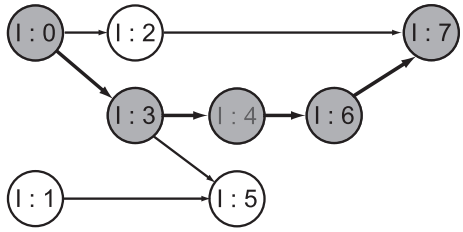


図 1 クリティカルパス
Fig. 1 Critical Path

本稿ではクリティカルパスを用いてデータの重要度を決める場合，クリティカルパス上に存在するメモリアクセス命令によって起こるキャッシュメモリへのアクセスをクリティカルアクセス，クリティカルパス上に存在しないメモリアクセス命令によって起こるキャッシュメモリへのアクセスをノンクリティカルアクセスと呼ぶ．また，それぞれのアクセスで供給されたデータをクリティカルデータ，ノンクリティカルデータと呼ぶ．

3. 省電力キャッシュアーキテクチャ

図 2 に本稿で検討している省電力キャッシュアーキ

テクチャを示す．ライトバック方式のキャッシュメモリを想定している．本キャッシュメモリでは，キャッシュメモリの持つウェイを低レイテンシと高レイテンシなウェイに分割する．低レイテンシなウェイはしきい値電圧の低いトランジスタで構成され，高い電源電圧を供給される．このため，アクセスに必要な電力と回路のリーク電流は大きいが，データ供給は高速に行われる．一方，高レイテンシなウェイはしきい値電圧の高いトランジスタで構成され，低い電源電圧を供給される．このため，アクセスに必要な電力と回路のリーク電流は小さくなる．しかし，データ供給は低速に行われる．以後本稿では，低レイテンシなウェイの集合を高速な領域，高レイテンシなウェイの集合を低速な領域と呼ぶ．

このキャッシュメモリでは，セット内のデータの重要度をライン毎に識別できなければならない．これはデータの読み出しがライン毎に行われるためである．時間的局所性を利用してデータの重要度を決める場合，LRU(Least Recently Used) 情報を用いて重要度を識別する．このとき，LRU 情報は高速・低速な領域ごとに管理されるとする．クリティカルパス情報を利用してデータの重要度を決める場合，クリティカルデータであるかノンクリティカルデータであるかを識別するための 1 ビットのフラグがライン毎に必要である．

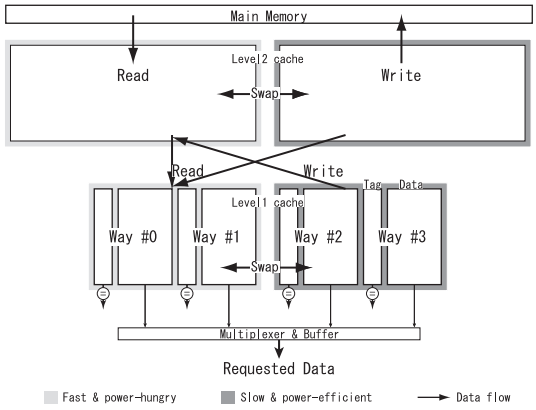


図 2 データの重要度を利用したキャッシュメモリ
Fig. 2 The Cache Memory Prioritizing Data

図 2 は，プロセッサと主記憶の間にレベル 1(L1) とレベル 2(L2) の二階層のキャッシュメモリを配置し，かつ，それぞれを高速・低速な領域に分割している場合を表している．各矢印はキャッシュメモリ内でのデータ移動の方向を示す．

プロセッサは L1 キャッシュメモリの高速・低速な領域の両方に読み出し・書き込みの処理を行える．一方 L1 キャッシュメモリは，読み出しは L2 キャッシュメモリのどちらの領域からでも可能であるが，書き込みは L1 キャッシュメモリの低速な領域から L2 キャッシュメモリの高速な領域へしか行えない．データを素早くプロセッサに供給するため，L2 キャッシュメモリから読み出されるデータは必ず L1 キャッシュメモリの高速な領域へ置かれる．同様に主記憶から読み出されるデータは L2 キャッシュメモリの高速な領域へ置かれる．主記憶への書き込みは L2 キャッシュメモリの低速な領域からのみ可能である．L2 キャッシュメモリへ書き戻されたデータは近い将来再び利用される可能性が高い．その点を考慮して L1 キャッシュメモリから追い出されたデータはまず L2 キャッシュメモリの高速な領域へ書き戻され，次に L2 キャッシュメモリの低速な領域へ追い出されるとする．

以上の点を考慮して，検討するキャッシュメモリは図 2 に示したデータ移動を行うとする．図 2 に示したデータの配置，移動方法が最適かどうかは不明であるが，本稿ではこの最適性についてではなく，図 2 に示した構成のキャッシュメモリ，データの配置方法を採用した際の効果について述べる．

以上の様なデータ移動が生じるため，L2 キャッシュメモリの高速な領域へは最近参照されたデータが集まり，低速な領域へは長期間参照されていないデータが集まる．L1 キャッシュメモリへのアクセス方法や追い出されるデータの決定方法が変化しても，自然に最近参照されたデータが L2 キャッシュメモリの高速な領域へ集まる．このように，L2 キャッシュメモリではもともと時間的局所性に基づくデータ配置が行われる構成になっていることを考慮して，時間的局所性でデータの重要度を決めるとする．一方，L1 キャッシュメモリは最近使われたデータが自然に高速な領域へ集まる構成をしていない．L2 キャッシュメモリのようにアーキテクチャ上の優位が存在しないため，クリティカルパス情報と時間的局所性を利用して重要度を決定することが可能である．

本キャッシュアーキテクチャのように各レベルを高速・低速な領域に分割することを考える場合，L1 キャッシュメモリの低速な領域を L2 キャッシュメモリ，L2 キャッシュメモリの高速な領域を L3 キャッシュメモリといったように分割した領域を利用して新たな記憶階層を構築することも考えられる．本研究では上位階層に存在するデータは下位階層にも存在するキャッシュメモリを想定している．また，分割前と分割後のキャッシュメモリの総容量を変化させないとすると，分割した領域で階層化を行った場合キャッシュメモリ内でア

クセス可能なデータ量が減少する．これはキャッシュミスの増加に繋がり，処理性能への影響が考えられる．このため，本研究では分割後の領域で新たな記憶階層を構築する構成はとらない．

4. キャッシュメモリ内のデータ移動

本節では重要度に基づいてキャッシュメモリ内のデータを移動する方法について説明する．

4.1 時間的局所性に基づいた重要度の場合

時間的局所性を用いてデータの重要度を決める場合のデータの移動方法について説明する．時間的局所性を用いてデータの重要度を決める場合，各セット毎にLRU を適用して移動させるデータを決める．低速な領域のデータがアクセスされた場合，高速な領域の最古にアクセスされたデータが低速な領域へ追い出される．低速な領域のデータは高速な領域へ移される．高速な領域にあるデータがアクセスされた場合は，高速・低速な領域間でデータの移動は起こらない．

4.2 クリティカルパス情報に基づく重要度の場合

クリティカルパス情報に基づいたキャッシュメモリ内のデータ移動について説明する．初めにクリティカルアクセスの場合について説明し，次にノンクリティカルアクセスの場合について説明する．

クリティカルアクセスの場合は高速な領域からデータが供給されることが理想である．従って，供給されたデータがもともと高速な領域にあった場合はデータの移動は必要ない．しかしクリティカルアクセスされたデータが低速な領域にあった場合は，次回以降のアクセスは高速な領域からデータが供給された方がパフォーマンスに悪影響を与えない．よって，データを移動させる．まず，当該アクセスは低速な領域から必要なデータを供給する．データ供給後，今後は高速な領域からデータを供給できるようにデータを移動させる．高速な領域の置き換え対象となるデータの決定方法を図 3 を用いて説明する．図中の CD はクリティカルデータを NCD はノンクリティカルデータを表す．

(1) 同一セットに有効なデータが存在しなければ，高速な領域の空きセットへデータを移動する (図 3(a))．
(2) 同一セットにノンクリティカルデータが存在した場合は，そのデータと置き換える (図 3(b))．
(3) 同一セットにクリティカルデータしかなかった場合，置き換え対象のデータは LRU で決定される (図 3(c))．

ノンクリティカルアクセスの場合は，アクセス時間に余裕がある．アクセスの際の動的消費電力を削減するためには，低速な領域からデータが供給されることが望ましい．従って，供給されたデータがもともと低速な領域にあった場合はデータの移動を行う必要がない．しかしノンクリティカルアクセスされたデータが高速な領域にあり，低速な領域にクリティカルデータが存在する場合は低速な領域へ移動させた方がよい場合がある．低速な領域の置き換え対象のデータは以下のように決定される．

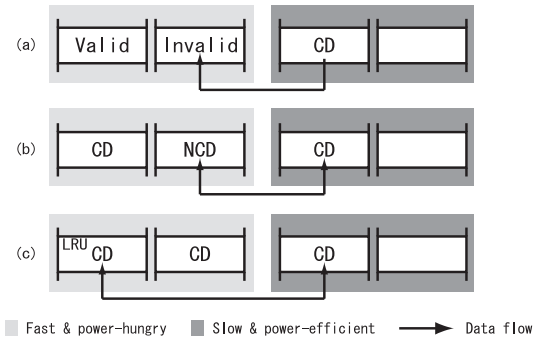


図 3 データ再配置の一例
Fig. 3 An Example of Data Allocation

(1) 同一セットにクリティカルデータが一つしか存在しなかった場合，そのデータと置き換える．
(2) 複数クリティカルデータが存在した場合，置き換え対象のデータは MRU で決定される．

一方，同一セット内のデータが全てノンクリティカルデータであった場合，置き換える必要はない．この場合，置き換えを抑制した方がデータの移動にかかるレイテンシや電力を削減できるからである．

4.3 キャッシュミスとデータの移動

L1 キャッシュメモリでミスが起こり，L2 キャッシュメモリへデータの書き戻しが必要な場合のデータの移動について説明する．まず L2 キャッシュメモリへ当該データが要求される．同時に，L1 キャッシュメモリの低速な領域の該当するセットから最も重要度の低いデータが L2 キャッシュメモリへ追い出される．L1 キャッシュメモリから追い出されたデータは L2 キャッシュメモリの高速な領域へ書き込まれる．L2 キャッシュメモリの高速な領域に当該データが存在する場合は，書き込みのみが行われる．同様に低速な領域にデータが存在する場合は，まず領域間でデータの移動が行われる．その後，高速な領域に書き込みが行われる．L2 キャッシュメモリから供給されるデータは，L1 キャッシュメモリの高速な領域に置かれる．このデータの置き場所を空けるため，L1 キャッシュメモリの高速な領域のデータを低速な領域へ移動させる．このようにして L2 キャッシュメモリから供給されたデータ

は，L1 キャッシュメモリの高速な領域からプロセッサへ供給される．

続いて，L2 キャッシュメモリでもミスが起こった場合を説明する．まず主記憶に当該データの要求がされる．同時に L1 キャッシュメモリの場合と同様に低速な領域からデータを主記憶へ追い出す．このデータは LRU で決まる．次に主記憶から供給されるデータの配置場所を空けるため，L2 キャッシュメモリの高速な領域の該当するセットから最も重要度の低いデータが低速な領域へ移動される．このようにして主記憶から供給されたデータは，L2 キャッシュメモリの高速な領域から L1 キャッシュメモリへ供給される．

4.4 データ移動方法の実装

本キャッシュメモリで必要なセット間でのデータ移動には，各セット同士が 1 対 1 で繋がっているのが望ましい．しかし，その実現のためには膨大な付加回路が必要となる上，付加回路の消費する電力も莫大なことが予想される．よって，低コストなデータの入れ替えを実現する必要がある．

この問題に対処するため，キャッシュメモリへの読み出し・書き込み動作に着目する．入れ替わるデータは今回のアクセスで読み出されたデータと入れ替え先のデータである．新たなハードウェアとして，プロセッサ–L1 キャッシュメモリ間，L1 キャッシュメモリ–L2 キャッシュメモリ間にそれぞれ 2 ラインを保持できるバッファを付け加える．今回のアクセスで読み出されたラインは，そのバッファに一時保管される．同様に置き換え先のデータをバッファに読み出した後，キャッシュメモリに書き込む要領で読み出したラインがあったウェイとは異なるウェイにデータを書き込む．以上でデータの入れ替えが完了する．この方式の場合，新たに必要となるハードウェアは 2 ラインを保存できるバッファだけである．このことからセット間を 1 対 1 で結ぶよりも格段に小さなコストでデータの入れ替えを実現可能になる．入れ替えが完了するまでは次のメモリアクセス命令はキャッシュメモリへアクセスできないとする．

この L1 キャッシュでの領域間データ移動がクリティカルとなり，パイプライン処理に大きな悪影響を及ぼすことは無いと考えられる．このデータ移動がクリティカルとなるのは，ロード命令が連続し，しかもそれらが全てデータ移動を必要とする時に限られる．参照の局所性を考慮すると，連続してデータ移動が必要な場合は稀であると思われる．

多くのウェイを持つキャッシュでメモリでは，LRU 制御を実装するコストを抑えるために疑似 LRU[11] を用いることがある．本稿で LRU 情報は，高速な領域，もしくは低速な領域から追い出されるデータを決定するために用いられる．高速な領域へ移動されるデータは，時間的局所性で無条件に，もしくはアクセス命令のクリティカリティといった LRU ではない情報で決定される．疑似 LRU 情報は 8 ウェイのセットアソシアティブキャッシュの場合でも，正しい LRU 情報と同等の情報を利用することができる[11]．このため，各領域から追い出されるデータを決定する際に用いる LRU 情報として疑似 LRU を用いても問題ない．

5. キャッシュメモリへのアクセス方法

高速な領域のアクセスに要するレイテンシを 1 サイクル，低速な領域のアクセスに要するレイテンシを 2 サイクルと仮定してキャッシュメモリのアクセス手順を説明する．図 4，図 5，図 6 において，$Way\#0$, $Way\#1$ が高速な領域，$Way\#2$，$Way\#3$ が低速な領域であるとする．

5.1 クリティカルパス情報に基づくアクセス

クリティカルパス情報を利用してデータの重要度を決定する場合，L1 キャッシュメモリの高速な領域にはクリティカルデータが，低速な領域にはノンクリティカルデータが集まる．本キャッシュメモリのこの性質を利用して，L1 キャッシュメモリのアクセスする領域を決定する．

図 4 左側にメモリアクセス命令がクリティカルアクセスである場合を，右側にノンクリティカルアクセスである場合をそれぞれ示す．図中の $Cycle\#$ はアクセスが起こって何サイクル目かを表している．クリティカルアクセスである場合，まず L1 キャッシュメモリの高速な領域にアクセスする．高速な領域でヒットした場合，1 サイクルでデータを読み出せる．しかし高速な領域に該当するデータが存在しなかった場合，低速な領域に 2 サイクルかけてアクセスする．低速な領域にもデータが存在しなかった場合，L2 キャッシュメモリにアクセスする．一方，ノンクリティカルアクセスの場合は，低速な領域，高速な領域，L2 キャッシュメモリの順にアクセスする．

本アクセス方法はアクセスする領域を区切っていることから，最初にアクセスした領域にデータが存在した場合，キャッシュアクセス時の動的消費電力を抑えることが可能である．しかし，データが最初にアクセスした領域とは逆の領域にあった場合，ヒットレイテンシが大きくなってしまう．またキャッシュミスの判明も遅れるので，ミスペナルティも大きくなる．

本アクセス方法はクリティカルパス情報に基づくア

クセスを行う．しかし 3 節で述べたように，我々が提案する省電力キャッシュの L2 キャッシュメモリは時間的局所性を用いてデータが配置されている．このため，L2 キャッシュのアクセス時にクリティカルパス情報に基づく本アクセス方法を用いてもその効果が期待できない．よって，L1 キャッシュで本アクセス方法を用いている場合でも L2 キャッシュメモリへのアクセスは，5.2 節もしくは 5.3 節で説明する方法を用いるとする．

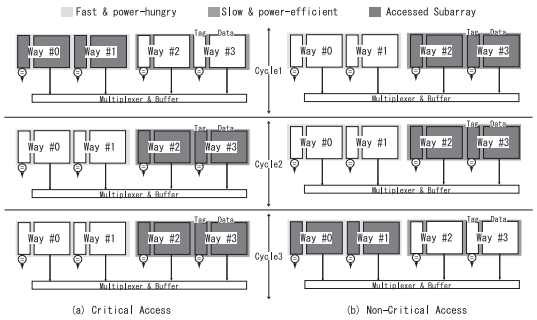


図 4 命令の重要度を利用したアクセス
Fig. 4 The Access Using Instruction Criticality

5.2 段階的アクセス

ライン読み出しをタグ比較の後に遅らせて，参照ラインだけをサブアレイから読み出すことでキャッシュメモリの省電力化を図る方式が提案されている[4),7),14)]．このキャッシュメモリを段階的キャッシュメモリという．段階的キャッシュメモリのタグは，1 サイクルで読み出しを行えるように高速な領域と同じしきい値のトランジスタと電源電圧で作られている．これまでのキャッシュメモリでは，データ読み出しがタグ読み出しと同時に行われていた．これに対し，段階的キャッシュメモリはタグ比較を行った後に必要なウェイだけを読み出す．ミスした場合はラインは読み出されない．この方法はタグ比較，データ読み出しを逐次的に行っている．このため，ヒットした場合に従来よりも余分なアクセス時間が必要となる．しかし，ミスした場合は無駄なライン読み出しが発生しないので消費電力を節約できる．

本アクセス方法を採用したヒットする場合の様子を図 5 に示す．タグ読み出しに 1 サイクルかかると仮定している．図において，左側は高速な領域でヒットする場合，右側は低速な領域でヒットする場合をそれぞれ表す．本アクセス方式では，1 サイクル目にタグのみを読み出し比較を行う．その後ヒットする場合は必要なデータのみ読み出す．ミスする場合も 1 サイクル目でミスが判明する．よって 5.1 節で説明したアクセス方法に比べ，ミスペナルティを少なくすることが可能となる．本アクセス方法は L1，L2 キャッシュメモリのどちらにも適用可能である．

本アクセス方法は，クリティカルパス情報に基づくアクセスとは異なりヒットする場合のレイテンシを実行前に見積もることができない．これは後続命令のスケジューリングに影響を与える．これに対しては，ウェイ予測[6)] を用いることで対応できる．ウェイ予測をデータキャッシュに適用した際の予測精度は平均で 86%と非常に高い[6)] ので，アクセスが予測された領域のレイテンシに従って後続命令をスケジュールすることで，後続命令への影響をなくすことができる．ウェイ予測を行うために用いるウェイ予測器の消費する電力は非常に小さい[9)]．よって，このための電力的なオーバーヘッドは，検討するキャッシュアーキテクチャによる省電力効果でうち消すことが期待できる．

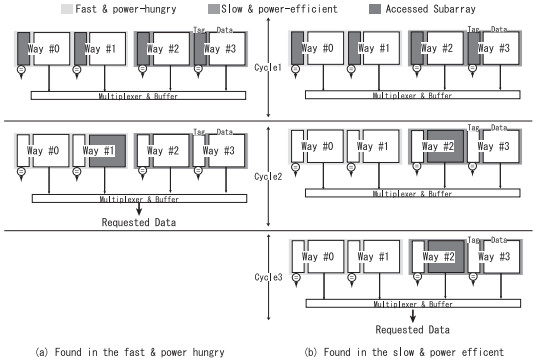


図 5 段階的アクセス
Fig. 5 Phased Access

5.3 高速型段階的アクセス

前述した段階的キャッシュメモリを用いたアクセス方法を改良する．前節で述べた方法をそのまま採用すると，キャッシュミスペナルティを削減できる反面，高速な領域のレイテンシが増加してしまう．これはタグ比較，ライン読み出しを段階的に行うためである．そこで，高速な領域は従来のようにタグ比較とライン読み出しを同時に行う．一方，低速な領域はタグ比較のみ高速な領域と同時に行えるよう，段階的アクセスを導入する．低速な領域は，タグ領域が低しきい値電圧で高速な領域と同時にタグ比較が可能だが，ライン部分は高しきい値で読み出し時間がかかる．以上のような実装をすることで，前節で説明した方式よりも高速な領域のヒットレイテンシを削減できる．本アクセス方式を図 6 に示す．高速な領域にデータが存在した場

合，1 サイクルでデータが供給される．キャッシュミスする場合も 1 サイクルでミスすることが判明する．このことから，5.2 節で説明したアクセス方式のミスペナルティを軽減可能という利点を保ちつつ，高速にデータを供給可能になっていることがわかる．段階的アクセスと同様に本アクセス方法は L1，L2 キャッシュメモリのどちらにも適用可能である．また，後続命令のスケジューリングにはウェイ予測情報を用いる．

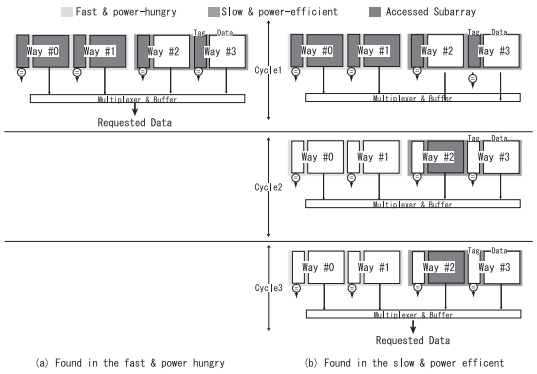


図 6 高速型段階的アクセス
Fig. 6 Fast Phased Access

6. 評　　価

6.1 本章の評価に用いた環境

SimpleScalar Tool Set[3] に省電力キャッシュメモリを組み込んだ環境で評価を行う．評価に用いるプロセッサの構成を表 1 に示す．キャッシュメモリ内でのデータ移動には，各レベルの高速・低速な領域へのアクセス時間を足した 8 サイクル，40 サイクルの入れ替えペナルティをそれぞれ要するとした．このため入れ替えが起こる場合，ヒット，もしくはミスレイテンシに各ペナルティが加算される．

命令セットは MIPS R10000 の ISA を拡張した SimpleScalar/PISA である．使用するベンチマークプログラムは SPEC 2000 CINT の中の 7 本で，それぞれの入力は表 2 に示す通りである．いずれのプログラムもはじめの 10 億命令をスキップし，続く 5 億命令をシミュレーションする．評価にはキャッシュメモリへのヒット率，プロセッサの処理性能，キャッシュメモリのエネルギー，エネルギー遅延二乗積 (Energy-Delay Square Product: ED^2P)[8] を用いる．

ED^2P はパワー削減による遅延の増加と省電力効果のトレードオフを定量的に評価するために用いられる指標である．電圧制御を用いて省電力化を試みる場合の評価に適している[8]．ED^2P は，高速・低速な領域がそれぞれ何回ずつ使用されたかをカウントし，その回数とそれぞれのアクセス毎に消費するエネルギー，リーク電流による消費エネルギー，プログラム実行に要した時間から求めることとした．高速・低速な領域は 70nm のテクノロジを想定し，アクセス時のエネルギーとリーク電流は CACTI4.1[12] を利用して求める．高速・低速な領域のしきい値電圧には CACTI の定める値を用い，それぞれ $0.19V$，$0.29V$ である．高速な領域の電源電圧は，ITRS のロードマップ[10] から 70nm のハイパフォーマンス向けトランジスタで用いる電源電圧を参照し，$1.1V$ とした．今後は電源電圧のスケーリング幅が小さくなることを考慮して，低速な領域の電源電圧は $1.0V$ とした．CACTI から求めた一回あたりのアクセスエネルギーを表 3，リーク電流を表 4 にそれぞれ示す．表には 1 ウェイあたりの結果を示す．なお，データ入れ替え時に消費するエネルギーは各領域への読み出し・書き込み分のエネルギーの合計とした．データの重要度決定にクリティカルパス情報を用いる場合，PIT を用いる．時間的局所性を用いる場合は LRU 情報を用いる．

表 2 ベンチマークプログラム
Table 2 Benchmark Programs

Benchmark	*Input Set*
gzip	input.source
vpr	net.in arch.in
gcc	166.i
parser	ref.in
vortex	lendian1.raw
bzip	input.source
mcf	inp.in

表 3 アクセス当たりのエネルギー (pJ)
Table 3 Energy per Access(pJ)

		Tag	Data
L1	Fast	0.65	11.83
	Slow	0.53	9.78
	Swap	2.36	43.21
L2	Fast	0.42	43.36
	Slow	0.21	21.64
	Swap	1.27	130.00

6.2 L1 キャッシュメモリへ適用した場合

まず省電力キャッシュアーキテクチャを L1 キャッシュメモリへ適用する場合のヒット率を図 7 に示す．図は，クリティカルパス情報に基づいてアクセスする領域の順番を決定する場合の結果である．図中の縦軸はキャッシュアクセスの内訳とヒット率を表す．横軸は各ベンチマーク毎の結果とその平均を示している．ラベルの

表 1 プロセッサ構成
Table 1 Processor Configuration

Clock Frequency	6 GHz
Fetch Bandwidth	8 instructions
Branch Predictor	1K-set 4-way set-associative BTB, 4K-entry 12-history-length gshare predictor, 64-entry return address stack, 6-cycle miss penalty, updated at commit stage
Insn. Windows	128-entry instruction queue, 64-entry load/store queue
Issue Width	8 instructions
Commit Width	8 instructions
Functional Units	6 Int, 3 FP, 4 Ld,St
Latency (total/issue)	iALU 1/1, iMUL 8/1, iDIV 32/1, fADD 4/1, fCMP 4/1, fCVT 4/1, fMUL 4/1, fDIV 32/1, fSQRT 32/1, Ld/St 2/1
Register Files	32 32-bit Int registers, 32 32-bit FP regsters
Insn. Cache	32KB, 4-way, 32B blocks, 1-cycle hit latency, 18-cycle miss penalty
Data Cache	32KB, 4-way, 32B blocks, 4-port, write-back, non-blocking load, hit under miss, 3-cycle fast hit latency, 5-cycle slow hit latency, 8-cycle swap penalty
L2 Cache	unified, 1MB, 8-way, 64B blocks, 18-cycle fast hit latency, 22-cycle slow hit latency, 40-cycle swap penalty
Main memory	80-cycle latency

表 4 リーク電流 (mW)
Table 4 Leakage Current(mW)

		Tag	Data
L1	Fast	0.36	6.60
	Slow	0.24	4.44
L2	Fast	0.99	101.11
	Slow	0.76	77.77

`f`，`s` の前の数字はそれぞれ高速・低速なウェイの数を示している．例えば `1f3s` は高速ウェイ数が 1，低速ウェイ数が 3 であることを表す．**ca** はクリティカルアクセス，**nca** はノンクリティカルアクセス，**fast** は高速ウェイヒット，**slow** は低速ウェイヒット，**miss** はキャッシュミスという意味である．望ましいキャッシュヒットである **ca_fast_hit**，**nca_slow_hit** は高速・低速なウェイ数に関係なくそれぞれ約 21%と約 42%である．一方，データの移動が必要になる **ca_slow_hit**，**nca_fast_hit** はそれぞれ約 14%と約 17%であった．**ca_slow_hit**，**nca_fast_hit** が起こっていることから，同一セットにアクセスする命令の重要度に変化があることがわかる．高速・低速なウェイ数が変わっても各ヒット率，ミス率に大きな変化はみられない．ヒット率に変化が見られない点，低速なウェイが多い方がリーク電流に起因する電力の削減効果を考慮して，今後は高速なウェイが 1，低速なウェイが 3 の L1 キャッシュについて評価を行っていくこととする．

図 7 に示した結果では，**ca_slow_hit**，**nca_fast_hit** がそれぞれ約 14%と約 17%あることから，データの移動に要するペナルティの影響が懸念される．そこで，スワップ頻度の低減のためにノンクリティカルアクセス時に高速な領域でヒットした場合，低速な領域へデータの移動を行わないようにした．これは以下の理由による．クリティカルアクセスされたデータは直ちにプロセッサへ供給されないと性能へ悪影響を与える．このため，低速な領域にクリティカルアクセスがあった場合，次回のアクセスからは当該データを高速な領域から供給するために当該データは移動されなければならない．しかし，ノンクリティカルアクセスされるデータが高速に供給されても，性能へ与える影響は小さいことが期待できるからである．これによりデータの移動が必要なキャッシュアクセスは **ca_slow_hit** 時のみとなる．この方式とデータの重要度を時間的局所性で決する方式を適用した場合の結果を図 8 に示す．図において横軸はベンチマーク毎の結果と平均を，縦軸はアクセスの内訳を表している．`CPI-normal` は図 7 に示した改良前の結果を，`CPI-remain` は **nca_fast_hit** 時にデータを低速なウェイに移動させなかった場合の結果をそれぞれ表し，`LRU` は時間的局所性を用いてデータの重要

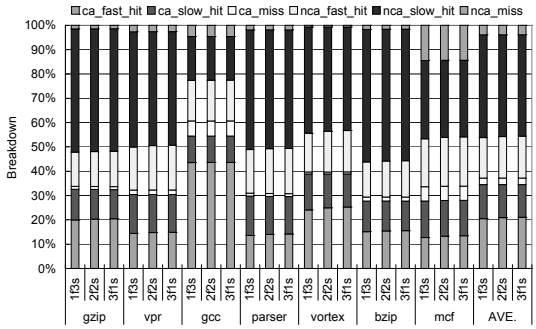


図 7 クリティカルパス情報を用いたアクセスをした場合の内訳
Fig. 7 Access Breakdown When Using Critical Path Information

度を決定した場合の結果を表している．改良前と比較して CPI-remain では，**ca_slow_hit**，**nca_slow_hit** の割合が減って，**ca_fast_hit**，**nca_fast_hit** の割合が増えている．改良前に平均で全アクセスの 21%に対して必要だったデータの移動が改良後には平均で全アクセスの 8%にまで抑えられている．LRU の場合，高速な領域のデータがアクセスされると当該データは高速な領域に留まり，低速な領域のデータがアクセスされると直ちに高速な領域へ移動される．このため，アクセスの傾向はデータの移動を抑制した場合と似ている．移動が必要なアクセスは平均で約 3%と非常に少ないが，**nca_fast_hit** の割合が非常に多い．クリティカルパス情報を用いてアクセスする領域を決める場合，アクセスは各領域に対して順番に行われる．従って **nca_fast_hit** は，低速な領域→高速な領域の順にアクセスされる．この時間が大きいとプログラムのクリティカルパスが入れ替わってしまい，処理性能へ影響を与える可能性がある．

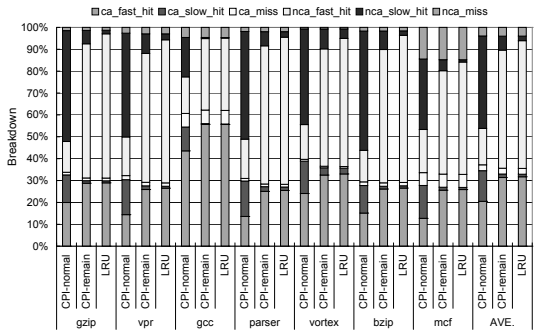


図 8 アクセスの比較
Fig. 8 Access Comparison

次に，検討するアーキテクチャが処理性能に与える影響を見る．処理性能を図 9 に示す．全てのウェイが高速な場合の結果で正規化されており，グラフが高いほど処理性能が高いことを示している．Slow は全てのウェイが低速な場合を示す．全てのウェイが高速，低速なキャッシュへのアクセスは，従来通りタグ領域とデータ領域へ並列アクセスを行う．データの入れ替えを抑制しない CPI-normal は，クリティカルパス情報に従った領域選択によるアクセスペナルティとデータの入れ替えペナルティによって Slow よりも処理性能が低下してしまう．一方 CPI-remain と LRU の場合，**nca_fast_hit** 時にはデータ供給までにペナルティがかかるものの，どちらも処理性能低下を抑えることができている．これは，これらのデータ配置方法がデータの移動にかかるペナルティを抑制できることと，クリティカルパスに与える影響を小さくできることによる．つまり，**nca_fast_hit** 時にかかるペナルティはクリティカルパスに悪影響を与えほど大きくない．またどちらの場合も **ca_fast_hit** が増加していることから，クリティカルアクセスされたデータが高速な領域から供給される機会が増している．これが処理性能によい影響を与え，**nca_fast_hit** の増加という弊害を小さくできている．

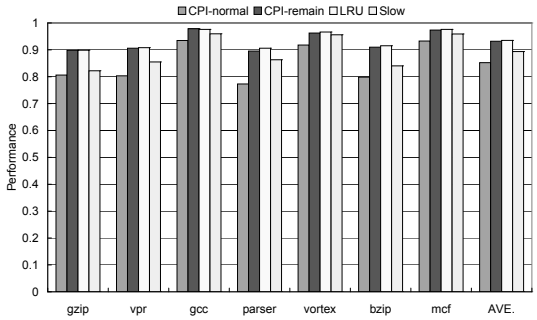


図 9 クリティカルパス情報を用いたアクセスをした場合の処理性能
Fig. 9 Performance Result When Used Critical Path Information for Cache Access

次にエネルギーについて見る．図 10 にエネルギーの結果を示す．全てのウェイが高速な場合の結果で正規化されており，グラフが低いほど消費エネルギーが少ないことを示している．横軸はベンチマークプログラム毎にデータの入れ替え方法を変化させた場合の結果を表し，縦軸はエネルギーの総量とその内訳を表す．縦軸の Access はキャッシュメモリにアクセスした時に消費したエネルギーを，Swap はデータの入れ替え時に消費したエネルギーを，Leak はリーク電流により消費したエネルギーをそれぞれ表す．Slow は全てのデータ領域へアクセスを行っているため，全ての場合と比較してアクセス時に消費するエネルギーが大きい．また CPI-remain，LRU の場合よりも処理性能が低下しているため，リーク電流によるエネルギーも CPI-remain，LRU の場合より大きい．CPI-normal から CPI-remain，LRU へとデータ入れ替え方法が変わると，キャッシュメモリへのアクセス時に消費するエネルギーが増加する．これはクリティカルパス情報に従った領域選択を行うことに起因する **nca_fast_hit** 増加のペナルティである．しかし，データ入れ替え時に消費するエネルギーは減少している．これは，データ入れ替えを抑制する CPI-remain，LRU の効果である．また CPI-remain，LRU では，データ入れ替え時にかかる処理性能へのペナルティも抑えることに成功

しているため，リーク電流による消費エネルギーも抑えることができている．

次に ED^2P の結果を見る．図 11 に結果を示す．全てのウェイが高速な場合の結果で正規化されており，グラフが低いほど省電力であることを示している．`Slow` では消費エネルギーは抑えられていたが，それに伴う処理性能の低下の影響で ED^2P は悪化している．`CPI-normal` ではデータの移動にかかるエネルギーとペナルティの影響で ED^2P が悪化している．データの移動を抑制した場合は，大きく省電力化ができているものとそうでないものがある．これは，検討しているアーキテクチャの構造上どうしても起きてしまうデータの移動がクリティカルパスへ影響を与える頻度とアクセス時のエネルギー削減効果に依存している．`CPI-remain` と `LRU` で結果に大きな差は見られない．しかし，時間的局所性を適用する `LRU` の方が `CPI-remain` よりも単純なハードウェアでデータの重要度を決定できる．よって L1 キャッシュメモリは，時間的局所性を利用してデータの重要度を決定することにする．

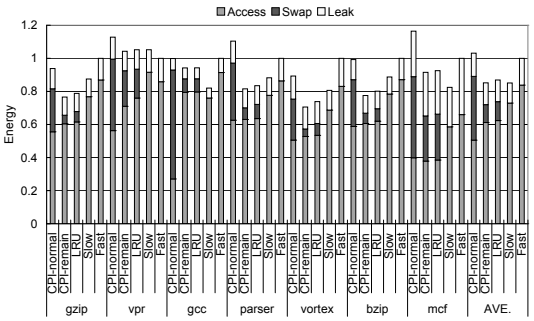


図 10 重要度決定方法ごとのエネルギー
Fig. 10 Energy Results

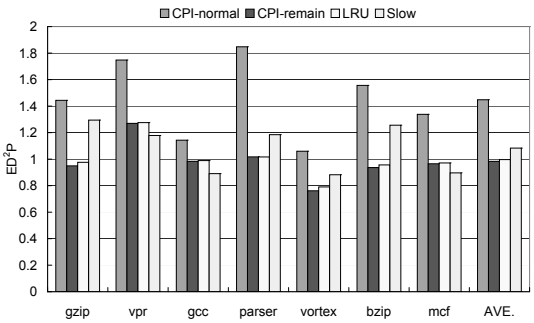


図 11 重要度決定方法ごとの ED^2P
Fig. 11 ED^2P Results

最後に，各領域へのアクセス方法を変えた場合の処理性能，エネルギー，ED^2P について見る．結果をそれぞれ図 12，図 13，図 14 に示す．図において，**CP** はクリティカルパス情報を用いてアクセスする領域を決めた場合を，**Phased** は 5.2 節で説明した段階的アクセスをした場合を，**F-Phased** は 5.3 節で説明した高速型段階的アクセスをした場合をそれぞれ示している．前述したように，**Phased** と **F-Phased** は **CP** に比べ，早い段階でミスを特定できる．また，全セットのタグを同時に読み出してデータが配置されてる場所を特定するので，**CP** 時に必要な領域毎のアクセスをする必要がない．このため，アクセス方式に **CP** を用いた時よりも高い処理性能を示している．また処理性能低下も `Slow` より小さい．特に高速な領域のデータをタグ比較と同時に読み出す **F-Phased** では，処理性能低下を平均で 2.7%にまで抑えることができている．次にエネルギーについて見る．**CP** では `Fast` や `Slow` の場合よりも悪化する場合も見られたが，**CP** と比べてアクセスする領域が少ない **Phased** と **F-Phased** は，アクセス時に消費するエネルギーを小さくすることに成功している．読み出すデータが低速な領域に存在する場合もタグ比較と同時に高速な領域のデータ領域を読み出す **F-Phased** は，**Phased** よりもアクセス時の消費エネルギーが大きい．ED^2P の場合も，必要なウェイにしかアクセスしないことで消費エネルギーを抑えることに成功した **Phased** と **F-Phased** は，全ての領域にアクセスする可能性のある **CP** と全てのウェイが低速な `Slow` よりも大幅な省電力化を実現できている．特に遅延が最も小さかった **F-Phased** の場合，平均で 54%ED^2P を削減できている．

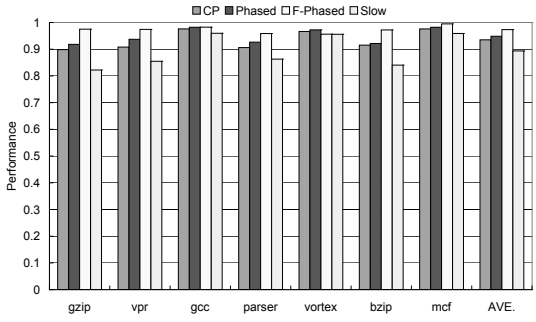


図 12 アクセス方法ごとの処理性能の結果
Fig. 12 Performance Result of Each Acccess Technique

6.3 L2 キャッシュメモリ

省電力アーキテクチャを L2 キャッシュメモリへ適

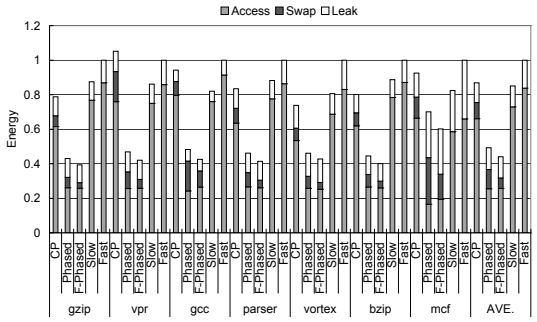


図 13 アクセス方法ごとのエネルギーの結果
Fig. 13 Energy Result of Each Acccess Technique

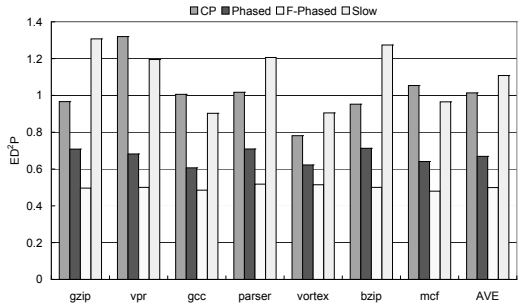


図 14 アクセス方法ごとの ED^2P の結果
Fig. 14 ED^2P Result of Each Acccess Technique

用した場合の結果を示す．3 節で述べたように，L2 キャッシュでのデータの重要度は LRU で決定される．アクセスの方法は，L1 キャッシュメモリへ適用して良い結果が得られた高速型段階的アクセスを用いる．L1 キャッシュメモリは高速な領域と低速な領域の割合が 1 : 3 で，時間的局所性を用いてデータを配置し，高速型段階的アクセスをされるとした．図中のラベルのうち，`1f7s` などは L2 キャッシュメモリの高速，低速な領域が持つウェイの数を表している．`0f8s` は L1，L2 キャッシュメモリの全ての領域が低速な場合を示している．全ての結果は L1，L2 キャッシュメモリの全ての領域が高速な場合の結果で正規化されている．

最初に処理性能について見る．図 15 に処理性能の結果を示す．高速な領域にウェイを 1 つしか持たない場合は，高速な領域から供給されるデータが少ないこととデータの入れ替えによるペナルティにより `0f8s` よりも処理性能が低下することがある．この処理性能低下は，高速な領域に存在するウェイ数の増加に伴い改善されていく．

次に消費エネルギーを見る．図 16 に消費エネルギーの結果を示す．全ての場合でリーク電流による消費エネルギーが大きいことがわかる．高速な領域のウェイ数が増加すると処理性能は改善するが，高速な領域の増加に伴いリーク電流が増加する．このため，リーク電流による消費エネルギーは減少しない．逆にタグ比較時に読み出されるデータ領域が増加するため，アクセス時のエネルギーが増加していく．エネルギーを示すグラフからは，`1f7s` と `2f6s` の場合が消費エネルギーが小さいことが分かる．

最後に ED^2P の結果を見る．図 17 に ED^2P の結果を示す．前述したように，高速な領域に置かれるウェイ数が増加するほど処理性能低下は抑制できたが，消費エネルギーは増加した．この消費エネルギー増加分を抑制できた処理性能低下でカバーできれば，高速な領域の増加に伴い ED^2P も改善するはずである．しかし，グラフから分かるように，多くのプログラムで `2f6s` の場合が最も ED^2P を改善できている．その後高速な領域に置かれるウェイ数が増加するにつれ，緩やかに ED^2P は悪化していく．

処理性能，消費エネルギー，ED^2P のそれぞれの結果から，高速な領域がウェイを 2 つ持つ場合が最適な構成であるといえる．このとき，平均で約 10%の処理性能低下で約 14%の ED^2P 削減を達成できている．ただしプロセッサ全体を考慮すると，10%の処理性能低下によりキャッシュメモリ以外のリーク電流による消費エネルギーの増加が懸念される．しかし，現在のプロセッサではキャッシュメモリの消費するエネルギーが支配的となっている[18] ことから，プロセッサ全体でも省電力化が期待できる．

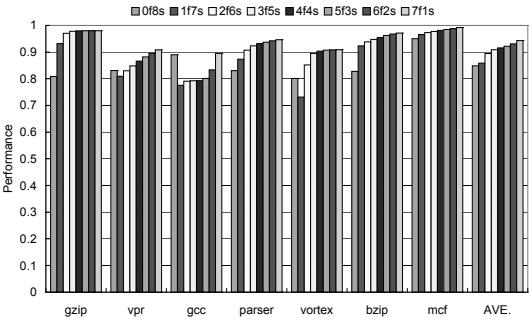


図 15 L2 キャッシュメモリへ適用した場合の処理性能
Fig. 15 The Performance Result of L2 Cache

7. 関 連 研 究

Abella ら[1] と Balasubramonian ら[2] は，クリティカルパス情報を用いたデータの配置とアクセスを行うことで，L1 データキャッシュで消費される動的消費電力と静的消費電力を削減することを提案している．

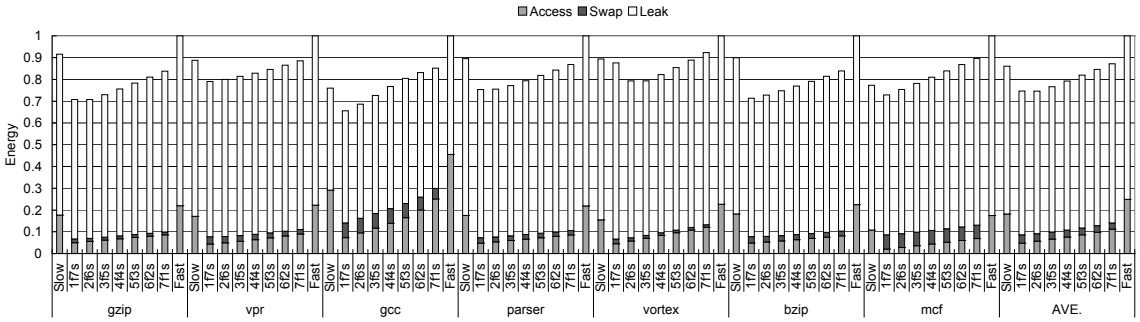


図 16 L2 キャッシュメモリへ適用した場合のエネルギー
Fig. 16 The Energy Result of L2 Cache

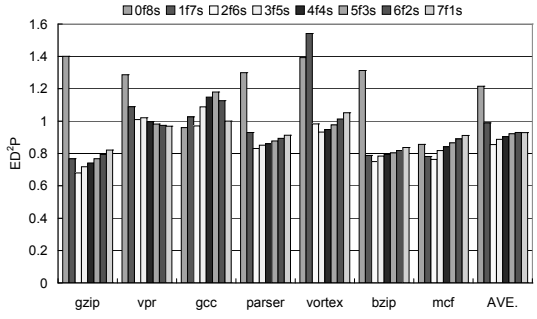


図 17 L2 キャッシュメモリへ適用した場合の ED^2P
Fig. 17 The ED^2P Result of L2 Cache

Abella らの提案する L1 キャッシュメモリは二つの独立したキャッシュメモリから構成されている．一つは高速にアクセスできるが高消費電力なキャッシュで，もう一方はアクセスは低速だが省電力なキャッシュである．アクセスは高速・低速なキャッシュへ並列に行い，タグ領域とデータ領域を読み出す．クリティカルパス上の命令によってアクセスされたデータが低速なキャッシュにあった場合，そのデータの複製が高速なキャッシュへ作られる．クリティカルパス上にない命令のアクセスするデータが高速なキャッシュにあった場合は，何も行われない．クリティカルパス上の命令によるアクセスがミスした場合は，高速なキャッシュからデータが供給される．そうでない命令がミスした場合は，低速なキャッシュからデータが供給される．データの複製を作るため，高速，低速なキャッシュ間でコヒーレンシをとる必要があるが，その方法や複製に必要なレイテンシは検討されていない．キャッシュヒットの場合もロード命令のレイテンシが一定ではなくなるため，ロード命令に続く命令のスケジューリング方法を検討する必要があるが，その方法については言及されていない．また，L2 キャッシュメモリの省電力化については考慮していない．

Balasubramonian らは高速な領域と低速な領域を持つ L1 キャッシュメモリを提案している．配置予測器と呼ばれる予測器に従って，それらの領域へデータを配置する．データ毎にアクセスされた命令がクリティカルであったか否かをカウンタに記録しており，データが追い出される際にカウンタの値に従って配置予測器を更新する．また，ロード命令がアクセスする領域の選択にバンク予測器を用いる．バンク予測器は供給されたデータが置かれていた領域の情報で更新される．予測がはずれた場合に備え，タグ読み出しは全ての領域で行われる．彼らは領域間でのデータの移動は検討していない．ロード命令に続く命令のスケジューリングについては言及されていないが，バンク予測器の予測結果を利用するものと推測される．また，L2 キャッシュメモリの省電力化については考慮していない．

8. ま と め

深刻な問題となっているプロセッサの動的・静的消費電力を削減するための省電力キャッシュアーキテクチャを検討した．本稿で検討したキャッシュアーキテクチャは，データの重要度を利用して高速だが高消費電力・低速だが省電力な領域にデータを配置する．これらの領域に対し，全ての領域のタグと高速な領域のデータを読み出し，その後低速な領域のデータを読み出すアクセス方法でアクセスを行った．評価の結果，データの重要度決定には時間的局所性を用いる方が良いこと，全ての領域が高速な L1，L2 キャッシュメモリと比較して，平均で約 10%の処理性能低下で約 14%の ED^2P 削減を達成できることがわかった．

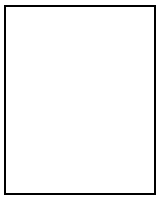
謝　　辞

本研究の一部は，文部科学省科学研究費補助金 (No. 16300019, No.176549) の援助によるものです．

参 考 文 献

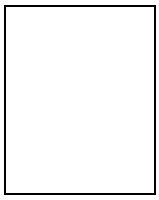
1) Jaume Abella and Antonio Gonzalez. Power efficient data cache designs. In *Proceedings of the International Conference on Computer Design*, pp. 8–13, October 2003.
2) Rajeev Balasubramonian, Viji Srinivasan, Sandhya Dwarkadas, and Alper Buyuktosunoglu. Hot-and-cold: Using criticality in the design of energy-efficient caches. In *Proceedings of the Workshop on Power Aware Computer Systems*, pp. 180–195, December 2003.
3) Doug Burger and ToddM. Austin. The Simplescalar Tool Set Version 2.0. Technical Report CS-TR-97-1342, University of Wisconsin, June 1997.
4) John H. Edmondson, Paul I. Rubinfeld, Peter J. Bannon, Bradley J. Benschneider, Debra Bernstein, Ruben W. Castelino, Elizabeth M. Cooper, Daniel E. Dever, Dale R. Donchin, Timothy C. Fischer, Anil K. Jain, Shekhar Mehta, Jeanne E. Meyer, Ronald P. Preston, Vidya Rajagopalan, Chandrasekhara Somanathan, Scott A. Taylor, and Gilbert M. Wolrich. Internal organization of the alpha 21164, a 300-mhz 64-bit quad-issue cmos risc microprocessor. *Digital Technical Journal*, Vol.7, No.1, pp. 119–135, January 1995.
5) Brian A Fields, Shai Rubin, and Rastislav Bodik. Focusing processor policies via critical-path prediction. In *Proceedings of the International Symposium on Computer Architecture*, pp. 74–85, July 2001.
6) Koji Inoue, Tohru Ishihara, and Kazuaki Murakami. Way-predicting set-associative cache for high performance and low energy consumption. In *Proceedings of International Symposium on Low Power Electronics and Design*, pp. 273–275, August 1999.
7) Koji Inoue, Vasily G. Moshnyaga, and Kazuaki Murakami. Trends in high-performance, low-power cache memory architectures. *IEICE Transactions on Electronics*, Vol. E85-C, No.2, pp. 304–314, February 2002.
8) Margaret Martonosi, David Brooks, and Pradip Bose. Modeling and analyzing cpu power and performance: Metrics, methods, and abstractions. Tutorials, SIGMETRICS 2001 / Performance 2001, June 2001.
9) Michael D. Powell, Amit Agarwal, T. N. Vijaykumar, Babak Falsafi, and Kaushik Roy. Reducing set-associative cache energy via way-prediction and selective direct-mapping. In *Proceedings of International Symposium on Microarchitecture*, pp. 54–65, December 2001.
10) Semiconductor Industry Association. International technology roadmap for semiconductors, 2005 edition, process integration, devices and structures, Janurary 2006.
11) Kimming So and Rudolph N. Rechtschaffen. Cache operations by mru change. *IEEE Transactions on Computers*, Vol.37, No.6, pp. 700–709, June 1988.
12) David Tarjan, Shyamkumar Thoziyoor, and NormanP. Jouppi. Cacti 4.0. Technical Report HPL-2006-86, Hewlett-Parckard Laboratories, June 2006.
13) Eric Tune, Dongning Liang, Dean M. Tullsen, and Brad Calder. Dynamic prediction of critical path instructions. In *Proceedings of the International Symposium on High-Performance Computer Architecture*, pp. 185–196, January 2001.
14) Don Weiss, John J. Wuu, and Victor Chin. The on-chip 3mb subarray based 3 level cache on an itanium microprocessor. *IEEE International Solid-State Circuits*, Vol. 37, No. 11, pp. 1523–1529, November 2002.
15) 小林良太郎, 安藤秀樹, 島田俊夫. データフロー・グラフの最長パスに着目したクラスタ化スーパースカラ・プロセッサにおける命令発行機構. 並列処理シンポジウム, pp. 31–38, June 2001.
16) 千代延昭宏, 佐藤寿倫. メモリアクセス命令の重要度を利用したキャッシュメモリの省電力化. 情報処理学会九州支部 火の国情報シンポジウム, March 2004.
17) 千代延昭宏, 佐藤寿倫, 有田五次郎. 低消費電力プロセッサアーキテクチャ向けクリティカルパス予測器の提案. 並列／分散／協調処理に関するサマー・ワークショップ, 情処研報 2002-ARC-149-2, pp. 7–12, August 2002.
18) Simon Segars. Low-power design techniques for microprocessors. Tutorials, International Solid State Circuits Conference, February 2001.

(平成 18 年 10 月 10 日受付)
(平成 19 年 1 月 23 日採録)



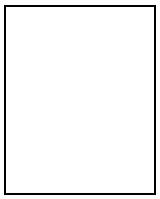
千代延昭宏（学生会員）

2002 九州工業大学情報工学部卒．2004 同大学大学院情報工学研究科博士前期課程修了．同年，九州産業大学情報科学部実習助手．2005 より 日本学術振興会特別研究員．現在，九州工業大学大学院情報科学研究科博士後期課程在籍．プロセッサアーキテクチャの研究に従事．IEEE 会員．



藤井誠一郎

2004 九州工業大学情報工学部卒．2006 同大学大学院情報工学研究科博士前期課程修了．同年，日立アドバンスドデジタル入社．



佐藤　寿倫（正会員）

1989 京都大学工学部卒．1991 同大学大学院工学研究科修士課程修了．同年，株式会社東芝入社．マルチプロセッサアーキテクチャ，消費電力見積り手法，および組み込み用マイクロプロセッサの研究開発に従事． 九州工業大学情報工学部助教授を経て，現在，九州大学システム LSI 研究センター教授．博士 (工学)．マイクロプロセッサアーキテクチャ，VLSI 設計手法に興味を持つ．電子情報通信学会，ACM，IEEE 各会員．