情報処理学会論文誌　コンピューティングシステム　Vol.5　No.1　41–52 (Jan. 2012)

3次元積層LSI向けSRAM/DRAMハイブリッドキャッシュ・アーキテクチャ

上野 伸也[1,a)]　橋口 慎哉[1,†1]　福本 尚人[1]　井上 弘士[2]　村上 和彰[2]

受付日 2011年5月11日, 採録日 2011年8月21日

概要：本稿では，3 次元積層 DRAM の利用を前提とし，大幅なチップ面積の増加をともなうことなく高いメモリ性能を達成可能な新しいキャッシュ・アーキテクチャを提案する．3 次元積層された DRAM を大容量キャッシュとして活用することで，オフチップメモリ参照回数の劇的な削減が期待できる．しかしながら，キャッシュの大容量化はアクセス時間の増加を招くため，場合によっては性能が低下する．この問題を解決するため，提案方式では，実行対象プログラムのワーキングセット・サイズに応じて 3 次元積層 DRAM キャッシュを選択的に活用する．ベンチマークプログラムを用いた定量的評価を行った結果，提案方式は動的制御方式により平均メモリアクセス時間を 15%削減した．

キーワード：3 次元積層，メモリ

SRAM/DRAM Hybrid Cache Architecture for Three-dimensional Integrated Circuits

Shinya Ueno[1,a)]　Shinya Hashiguchi[1,†1]　Naoto Fukumoto[1]　Koji Inoue[2]
Kazuaki Murakami[2]

Received: May 11, 2011, Accepted: August 21, 2011

Abstract: This paper proposes a novel cache architecture for 3D-implemented microprocessors. 3D-IC is one of the most interesting techniques to achieve high-performance, low-power VLSI systems. Stacking multiple dies makes it possible to implement microprocessor cores and large caches (or DRAM) into the same chip. Unfortunately, applying the 3D DRAM cache causes performance degradation for some programs, because increasing cache size makes access time longer. To tackle this issue, the proposed cache supports two operation modes: a fast but small SRAM cache mode and a slow but large DRAM cache mode. An appropriate operation mode is selected at run time based on the behavior of application programs. The evaluation results show that the proposed approach achieves 15% of memory performance improvement.

Keywords: 3D integration, memory

1. はじめに[*1]

半導体チップの新しい実現法として 3 次元積層技術が注目されている．従来の 2 次元実装 LSI においては，回路の大規模化にともないブロック間接続のための配線が長くなる．そのため，動作周波数の低下や消費電力の増大を招くといった問題があった．これに対し，3 次元積層 LSI では，3 次元方向へ回路を集積することで配線長を維持しつつ，回路を大規模化できるといった利点がある．また，たとえば DRAM とロジックのように異なるプロセスを経て製造

[1] 九州大学大学院システム情報科学府
Graduate School of Information Science and Electrical Engineering, Kyushu University, Fukuoka 819–0395, Japan
[2] 九州大学大学院システム情報科学研究院
Faculty of Information Science and Electrical Engineering, Kyushu University, Fukuoka 819–0395, Japan
[†1] 現在，富士通株式会社
Presently with Fujitsu Ltd.
[a)] ueno@soc.ait.kyushu-u.ac.jp

[*1] なお，本稿の概要は SACSIS で発表予定である．本稿では，文献 [1] の内容に対し，新たな動作モード決定アルゴリズムを導入することでより高い性能を実現している．

した複数のダイを積層することも比較的容易になる．

このような背景の中，3 次元積層技術を利用したプロセッサ構成法に関する研究が行われるようになってきた．その中でも特に，個別の製造プロセスを経て作成した大容量 DRAM ダイとプロセッサ・ダイを積層し，これらの間を TSV（Through Silicon Via）で接続するアプローチが大きな注目を集めている [2], [3], [4]．プロセッサ・コアと大容量メモリを 1 個の LSI チップに混載することで大容量オンチップ・メモリを実現するとともに，TSV による高オンチップ・メモリバンド幅も利用可能となる．これにより，深刻化の一途をたどるメモリウォール問題の抜本的解決策として期待できる．

文献 [2] では，積層した DRAM を L2 キャッシュとして活用する．このような DRAM キャッシュ・スタック法（以降，DRAM スタック法と略す）では，高速なタグ検索を実現するために，SRAM 実装されたタグ RAM をプロセッサ・コアと同一レイヤに有する．L1 キャッシュミスが発生した際，当該タグメモリを参照すると同時に，積層された大容量 DRAM のアクセスを開始する．DRAM スタック法では，32〜64 MB 程度の DRAM をラストレベル・キャッシュとして使用するため，オフチップ・アクセス回数を劇的に削減できる．しかしながら，その反面，キャッシュの大容量化はアクセス時間の増加を招く．そのため，小容量なキャッシュでも十分にオフチップ・アクセス回数が少ないアプリケーションでは性能が低下するといった問題が生じる．

そこで本研究では，従来の DRAM スタック法が有する問題点を解決すべく，新しいメモリ構成法として **SRAM/DRAM ハイブリッド・キャッシュ**を提案する．本方式は，高速かつ小容量な「SRAM キャッシュ・モード」，または，3 次元積層 DRAM を用いた低速かつ大容量な「DRAM キャッシュ・モード」での動作が可能である．アプリケーション特性に応じて適切な動作モードを選択することで，高速アクセスの実現とキャッシュミス率の改善を両立させ，メモリ性能の向上を可能にする．

提案するハイブリッド・キャッシュでは，アプリケーションの特性に応じて適した動作モードを選択する必要がある．一般に，メモリアクセス・パターンは入力データにより変化するため，実行前に適切な動作モードを決定するのは難しい．そこで本研究では，プログラム実行中に将来の適切な動作モードを決定する動的最適化アルゴリズムを導入する．

以下，2 章では文献 [2] で提案された DRAM スタック法を紹介し，その問題点を整理する．次に，3 章でハイブリッド・キャッシュを提案し，そのマイクロアーキテクチャ，制御方式，動作モード決定法の詳細を示す．4 章ではベンチマークプログラムを用いて定量的評価を行い，提案方式の有効性を明らかにする．最後に 5 章で簡単にまとめる．

2. DRAM スタック法

2.1 メモリ・アーキテクチャ

従来の 2 次元実装プロセッサ（以降，ベースプロセッサと呼ぶ），ならびに，文献 [2] で提案された DRAM スタック法の構成を**図 1** に示す．以降，本稿では，プロセッサ・コアが実装されたダイを**下層ダイ**，DRAM が実装されたダイを**上層ダイ**と呼ぶ．ベースプロセッサは下層ダイのみで構成され，1 個のプロセッサ・コアと L2 キャッシュが搭載されていると仮定する．一方，DRAM スタック法では，3 次元積層技術により上層ダイとして DRAM を積載し（以降，上層 DRAM と呼ぶ），これを L2 キャッシュのデータ領域として使用する．L2 キャッシュのタグ情報に関しては，高速なキャッシュ検索を可能とするために SRAM を用いて下層ダイに実装される[*2]．このタグ RAM のサイズは，プロセッサのアドレスビット長とキャッシュ構成に依存する．たとえば，上層 DRAM の容量が 64 MB，アドレス長 64 ビット，ブロックサイズ 64 B，連想度 16 の場合，そのタグ領域には約 5 MB の容量が必要となる．これは，デュアルコア・プロセッサ・チップに搭載される L2 キャッシュ容量とほぼ同程度（もしくはそれ以上）の実装面積を要する．そこで，DRAM スタック法では，ベースプロセッサにおいて L2 キャッシュとして使用していた SRAM 領域を利用してタグメモリを搭載する．

2.2 性能向上の条件

2.1 節で説明した DRAM スタック法における最大の利点は，ラストレベル・キャッシュの大容量化にともなうオフチップ・アクセス回数の大幅な削減である．その一方，一般にキャッシュヒット時間はキャッシュ容量に比例して長くなる傾向にある．そのため，DRAM スタック法では L2 キャッシュアクセス時間が増大し，ひいては性能低下をもたらす恐れがある．そこで本節では，DRAM スタック法により性能向上を達成するための条件を整理する．ベースプロセッサと DRAM スタック法に関するメモリ性能の優劣を解析するため，本節では以下の式で表される *AMAT*（Average Memory Access Time：平均メモリアクセス時間）を用いる．

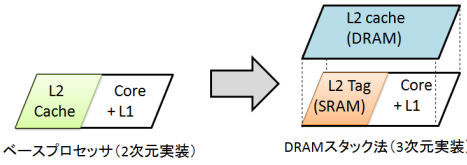

図 1　ベースプロセッサと DRAM スタック法の構成
Fig. 1　Architecture of baseprocessor & DRAM-stacking.

[*2] 一般に，DRAM と SRAM は製造プロセスが大きく異なる．そのため，同一層に DRAM データ領域と SRAM タグ領域を実装することは可能であるが，その場合には製造コストが増加する．

$$AMAT = HT_{L1} + MR_{L1} \times (HT_{L2} + MR_{L2} \times MMAT) \quad (1)$$

- HT_{L1}/HT_{L2}：L1/L2 キャッシュヒット時間 [cycles]
- MR_{L1}/MR_{L2}：L1/L2 キャッシュミス率
- $MMAT$：主記憶のアクセス時間 [cycles]

ベースプロセッサと DRAM スタック法の違いは，メモリ階層における L2 キャッシュ部分にある．したがって，両者において，HT_{L1}，MR_{L1}，ならびに，$MMAT$ は同一となる．ここで，HT_{L2} と MR_{L2} に関して，それぞれ，ベースプロセッサの場合を HT_{L2_B} と MR_{L2_B}，また，DRAM スタック法の場合を HT_{L2_D} と MR_{L2_D} で表記する．厳密には回路構成やデバイス特性に異存するが，一般的には以下の関係式が成り立つ．

$$HT_{L2\_B} \leq HT_{L2\_D} \quad (2)$$

$$MR_{L2\_B} \geq MR_{L2\_D} \quad (3)$$

したがって，DRAM スタック法がベースプロセッサの性能を上回るための条件は式 (4) となる．

$$HT_{L2\_B} + MR_{L2\_B} \times MMAT > HT_{L2\_D} + MR_{L2\_D} \times MMAT \quad (4)$$

この式を変形すると，式 (5) が導き出される．

$$MR_{L2\_B} - MR_{L2\_D} > \frac{HT_{L2\_D} - HT_{L2\_B}}{MMAT} \quad (5)$$

左辺は DRAM スタック法の採用による L2 キャッシュミス率削減効果 $MR_{L2\_REDUCTION}$，右辺の $HT_{L2\_D} - HT_{L2\_B}$ は L2 キャッシュアクセス時間オーバヘッド $HT_{L2\_OVERHEAD}$ である．この式より，DRAM スタック法により性能向上を実現するためには，L2 キャッシュアクセス時間オーバヘッドを隠蔽できる L2 キャッシュミス率削減効果が必要であることが分かる．

2.3　問題点と解決すべき課題

本節では，DRAM スタック法の問題点と解決すべき課題を整理する．ここで，ベースプロセッサに関しては，2 MB の 8 ウェイ・セットアソシアティブ方式（ラインサイズは 64 B）であり，そのアクセス時間は 6 クロックサイクルとする．一方，DRAM スタック法においては 32 MB の 8 ウェイ・セットアソシアティブ方式（ラインサイズは 64 B）であり，アクセス時間は 28 クロックサイクルと仮定する．主記憶アクセス時間 $MMAT$ は 181 クロックサイクルであり，各アクセス時間の根拠は 4.1 節で示す．その他の実験環境の詳細については，4.1 節で示す内容と同じである．一般に，キャッシュヒット時間は，キャッシュ構成，ならびに，メモリ素子やデコーダ，センスアンプといった各構成要素の動作速度によって決定されるため，$HT_{L2\_OVERHEAD}$ は定数となる．これに対し，$MR_{L2\_REDUCTION}$ は実行対象となるプログラムの特性に大きく依存するため，DRAM スタック法では以下の問題点が生じる．

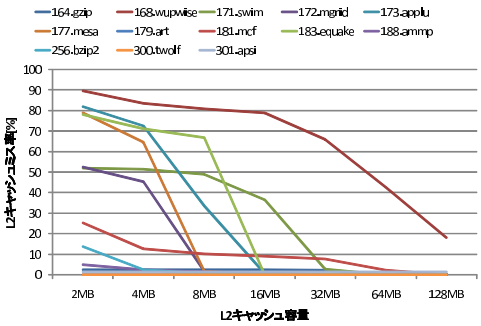

図 2　L2 キャッシュ容量と L2 キャッシュミス率：SPEC CPU 2000
Fig. 2　L2 size-missrate: SPEC CPU 2000.

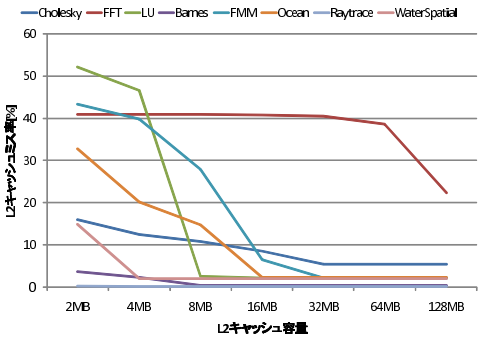

図 3　L2 キャッシュ容量と L2 キャッシュミス率：Splash2
Fig. 3　L2 size-missrate: Splash2.

- **L2 キャッシュミス率改善効果はプログラム間で異なるため，場合によっては DRAM スタック法の導入により性能が低下する．**複数ベンチマークプログラムの実行において，L2 キャッシュ容量を 2 MB から 128 MB まで変化させた場合のミス率を**図 2** ならびに**図 3** に示す．横軸は L2 キャッシュ容量，縦軸は L2 キャッシュミス率である．図 2 の *171.swim*，*172.mgrid*，*173.applu*，*183.equake* や図 3 の *LU*，*FMM* などでは，L2 キャッシュサイズを 2 MB から 32 MB に拡大することで大幅なミス率削減を達成している．これに対し，図 3 の *FFT* においては，32 MB の L2 キャッシュ容量に対してワーキングセット・サイズが十分に大きいため，L2 キャッシュミス率の改善はきわめて低い．また，図 2 の *164.gzip* や図 3 の *Raytrace* などでは，2 MB の L2 キャッシュ容量に対してワーキングセット・サイズが十分に小さいため，DRAM 積層による恩恵を受けることができない．このように，L2 キャッシュサイズの増大によるミス率削減効果は実行対象プログラムの特性に依存するため，式 (5) で示した性能向上条件を必ずしも満たすとは限らない．
  各プログラムの実行において，$MR_{L2\_REDUCTION}$ と $HT_{L2\_OVERHEAD}$ が DRAM スタック法の利得（*Profit*）に与える影響を**図 4** に示す．ここで，利得とは，大容量 DRAM キャッシュの導入に起因する

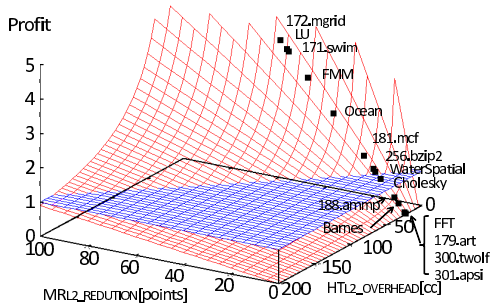

図 4　$MR_{L2\_REDUCTION}$ と $HT_{L2\_OVERHEAD}$ が $Profit$ に対して与える影響

Fig. 4　Effect of $MR_{L2\_REDUCTION}$ and $HT_{L2\_OVERHEAD}$.

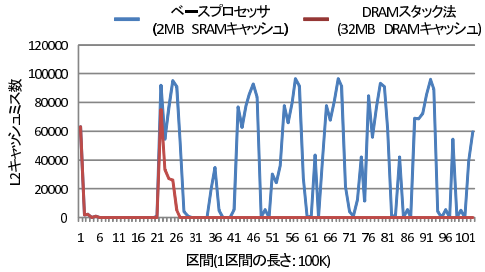

図 5　L2 キャッシュアクセス 10 万回あたりの L2 キャッシュミス数：Ocean

Fig. 5　L2 cache misses per 100,000 accesses: Ocean.

L1 ミスペナルティ削減効果を表す指標であり，以下のように定義する．

$$Profit = \frac{MR_{L2\_REDUCTION} \times MMAT}{HT_{L2\_OVERHEAD}} \qquad (6)$$

$Profit$ の値が 1.0 より大きい場合は DRAM スタック法の導入により性能改善が期待できることを意味する．図 4 には，$Profit$ が 1.0 以下のベンチマークプログラムが多くあるため，DRAM スタック法の導入により性能低下が発生する可能性があることが分かる．

- **L2 キャッシュミス率改善効果は単一プログラム実行中にも変動するため，DRAM スタック法の潜在能力を十分に引出すことができない．**一般に，プログラムの実行において，メモリ参照の振舞いは時々刻々と変化するため，それにともないキャッシュミス率も変動する．*Ocean* ベンチマークプログラムを実行した際の L2 キャッシュミス発生頻度の変動を図 **5** に示す．横軸は L2 キャッシュアクセス 10 万回を 1 区間とする時間経過を表しており，縦軸は各区間において発生した L2 キャッシュミス回数である．図 5 より，プログラム実行中に L2 キャッシュミス発生頻度が大きく変化していることが分かる．特に，キャッシュサイズを 2 MB から 32 MB へ増大させることにより，ミス発生回数削減効果が大きい区間（たとえば，区間 41 から 47）と小さい区間（たとえば，区間 1 から 20）が存在することが分かる．また，L2 ヒット時間も考慮した DRAM スタック法によるメモリ性能改善効果の変動

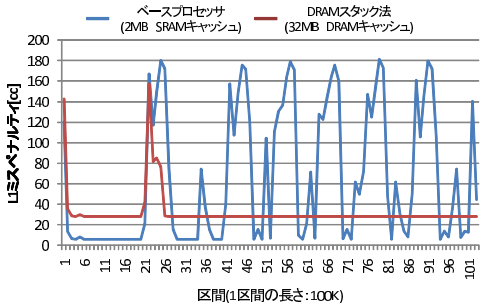

図 6　プログラム実行中の各区間における L1 ミスペナルティ：Ocean

Fig. 6　Run-time variation of L1 misspenalty: Ocean.

を図 **6** に示す．縦軸は各区間における L1 ミスペナルティ（$HT_{L2} + MR_{L2} \times MMAT$）であり，横軸は図 5 と同様に区間を表す．これらの実験結果から，プログラムの実行において，DRAM スタック法の適用により性能低下が生じる区間が存在することが分かる．

3.　3 次元積層プロセッサ向け SRAM/DRAM ハイブリッド・キャッシュ

3.1　基本概念

2.3 節で述べたように，従来の DRAM スタック法では，十分な L2 キャッシュミス削減率を達成できなければ性能が低下する．この問題を解決する単純な方法として，ベースプロセッサと同様，SRAM で構成される L2 キャッシュを下層に実装し，上層 DRAM を L3 キャッシュとして活用することが考えられる．しかしながら，この場合，2.1 節で説明したように，下層に実装されるタグメモリはベースプロセッサの L2 キャッシュと同程度の容量が必要となるため，下層ダイの面積オーバヘッドが大きくなる．そこで本研究では，大幅な面積の増加をともなうことなく，DRAM スタック法の問題を解決し，さらなる高性能化を実現する新しいメモリ構成法として図 **7** に示す **SRAM/DRAM ハイブリッド・キャッシュ**を提案する．本方式では以下の 2 つの動作モードを有する．

- **SRAM キャッシュ・モード：**下層 SRAM が通常の L2 キャッシュとして動作する．このとき，上層 DRAM は使用せず，消費電力削減のために電源供給を遮断する．
- **DRAM キャッシュ・モード：**下層 SRAM が上層 DRAM 用のタグ RAM として動作する．このとき，従来の DRAM スタック法と同様に，上層 DRAM は L2 キャッシュのデータ（キャッシュライン）保存用メモリとして使用される．

SRAM/DRAM ハイブリッド･キャッシュでは，L2 キャッシュミス率削減効果が十分に得られる場合にのみ積層 DRAM を活用する．それ以外の場合には通常の L2 キャッシュメモリとして動作し，上層 DRAM へのアクセスは行わない．このように，選択的に大容量 DRAM を活用する

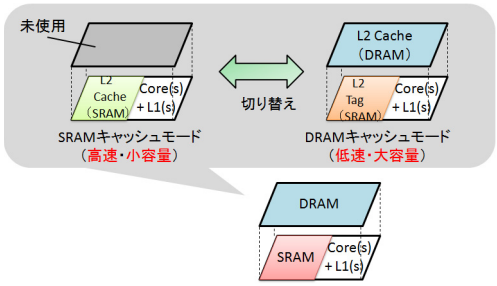

図 7　SRAM/DRAM ハイブリッド・キャッシュ
Fig. 7　SRAM/DRAM Hybrid Cache Architecture.

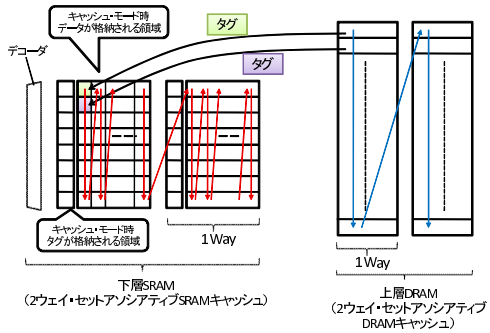

図 8　DRAM キャッシュモード時のタグのマッピング
Fig. 8　DRAM-tag mapping for SRAM of bottom layler.

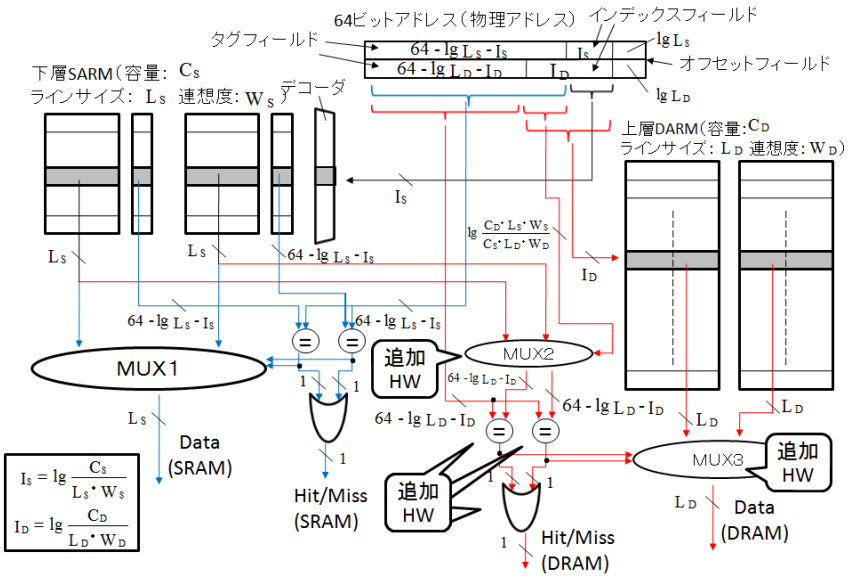

図 9　ハードウェア・アーキテクチャ
Fig. 9　HW-support.

ことで，高速アクセスとキャッシュミス率改善の両立を可能にし，3 次元積層 DRAM の潜在能力を最大限に引き出す．

3.2　マイクロ・アーキテクチャ

ハイブリッド・キャッシュでは，下層 SRAM は通常のデータキャッシュならびにタグ RAM としての動作切替えが可能でなければならない．DRAM キャッシュ・モード動作時，下層 SRAM のデータアレイに上層 DRAM キャッシュ用タグを格納する．ここで，下層 SRAM の切替えを実現するためには以下の 2 点を考慮する必要がある．

- **タグ情報のマッピング：**一般に上層 DRAM は下層 SRAM より大きな容量を有する．したがって，ラインサイズが同じ場合には，上層 DRAM と比較して，下層 SRAM キャッシュの総キャッシュライン数は少なくなる．そこで，DRAM キャッシュ・モード時，下層 SRAM では 1 個のキャッシュラインに複数タグを格納する．連想度が 2 の場合の例を**図 8** に示す．上層 DRAM の各ウェイに対応するタグに関して，下層 SRAM へ垂直方向にマッピングする．なお，本マッピングは連想度が 2 以外の場合でも対応可能である．
- **ハードウェア・サポート：**ハイブリッド・キャッシュのマイクロ・アーキテクチャを**図 9** に示す．ここで，下層 SRAM ならびに上層 DRAM に関して，それぞれの容量を C_S と C_D，ラインサイズを L_S と L_D，連想度を W_S と W_D で記す．アドレス長は 64 ビットと仮定する．連想度とラインサイズが同じ場合，下層 SRAM キャッシュのセット数は上層 DRAM のそれと比較して少ない．そのため，前述したように，下層 SRAM は上層 DRAM のタグを垂直方向に折り畳んで格納する．したがって，データアレイより読み出したデータの中から該当するタグを選択するための専用マルチプレクサが必要となる（図 9 の MUX2 ならびに MUX3）．また，タグ比較を行うための比較回路などが追加される．これらの追加構成要素は小規模かつ容易に実装でき，アクセス時間や実装面積に与える影響は無視できるほど小さいと考えられる．

DRAM キャッシュ・モード時のデータ読み出しは以下

の流れで行われる．インデックスは上位 $I_D - I_S$ ビットと下位 I_S ビットの 2 つに分割される．まず，下位 I_S ビットで下層 SRAM キャッシュの 1 セットに格納されている全タグを読み出す．次に，上位 $I_D - I_S$ ビットで 1 つのセットに格納されているすべてのタグから W_D 個のタグを選択する．以降は通常のキャッシュと同様，メモリ参照アドレスより得たタグと比較し，一致するものが存在すればヒット，存在しなければミスとなる．ヒットの場合には，インデックスにより上層 DRAM から読み出されたデータから当該タグに対応するデータを選択する．ただし，図 8 で示したタグ情報のマッピング，ならびに，図 9 のマイクロアーキテクチャを前提とした場合，下層 SRAM が正しく動作するためには以下の条件を満足する必要がある．

- 下層 SRAM と上層 DRAM の連想度が 2 のべき乗である：この条件を満たさない場合，インデックスの上位ビットではビット数が不足し，1 つのキャッシュラインに格納されているタグから選択できないためである．
- $I_S \leq I_D$（$\frac{C_S}{L_S W_S} \leq \frac{C_D}{L_D W_D}$）：この条件を満たさない場合には，DRAM キャッシュ・モード動作時において，上層 DRAM のタグ読み出しのため，下層 SRAM キャッシュの複数セットにアクセスする必要が生じる．
- $\frac{C_D W_S}{C_S W_D}(64 - lgL_D - lgI_D) \leq L_S$：この条件を満たさない場合は，上層 DRAM のすべてのタグを下層 SRAM キャッシュに格納できない．

3.3　動的動作モード決定法

3.3.1　動作モード決定方針

SRAM/DRAM ハイブリッド・キャッシュでは，いかに適切な動作モードを選択できるかがきわめて重要になる．適切な動作モード（より高い性能を達成できる動作モード）は，プログラム実行におけるメモリ参照の振舞い，ならびに，SRAM と上層 DRAM のハードウェア特性に大きく依存する．ハイブリッド・キャッシュでの動作モード決定においては，動作モード決定時期ならびに動作モード設定時期に関して，プログラム実行前または実行中の選択肢が考えられる．本稿では，サーバやデスクトップ PC といった汎用システムでの応用を想定している．この場合，実行コードの互換性はきわめて重要となる．また，組込みシステムとは異なり，多くの場合において，実行時の振舞いを事前に解析することは難しい．これらの理由により，本稿ではプログラム実行時にハードウェアレベルで動作モードを決定する選択肢を採用する．ハイブリッド・キャッシュにおいて，プログラム実行中の動作モード切替えは以下の手順で行われる．

(1) L2 キャッシュへのアクセスを禁止する．プロセッサ側から L2 キャッシュへのアクセス要求があれば，動作モード切替えが終了するまでプロセッサはストールする．
(2) 現在使用している L2 キャッシュ（つまり，動作モード切替え前に使用している L2 キャッシュ）をフラッシュする．動作モードの切替えにより L2 キャッシュのデータ/タグ情報の記憶領域が変更されるために必要となる．
(3) データ/タグ情報記憶領域へのアクセスパスを変更する（詳細は 3.2 節を参照）．これにより動作モードの切替えが完了する．
(4) L2 キャッシュへのアクセスを再開する．

動的に動作モードを切り替えることで大きな性能向上が期待できる一方で，上述したようにキャッシュをフラッシュする必要があるため，性能向上阻害要因として以下の 2 つの問題が生じる．

- **ライトバックにともなう性能オーバヘッド：**動作モード切替え前の L2 キャッシュ内のダーティ・ラインをすべて主記憶へ書き戻す必要がある．そのため，メモリバンド幅を圧迫し性能低下が発生する可能性がある．
- **初期参照ミスの増加：**動作モード切替え直後, L2 キャッシュは空の状態である．そのため，見かけ上の初期参照ミス（動作モード切替えが原因で新たに発生する初期参照ミス）が増加する．

したがって，これらの性能オーバヘッドを考慮した適切な動作モードの決定が必要となる．

さらに，適切な動作モードの決定には L1 ミスペナルティの変化に対する追従性が重要である．追従性が高いと，素早く動作モードの切替えを行うことができるため性能向上を実現できる．しかしながら，適切な動作モードの変化が多いプログラムでは，動作モードの切替え回数が増加する．これによりオーバヘッドの総和が増加し，性能が低下する恐れがある．追従性が低い場合には，これとは逆のことが起こる．したがって，適切な動作モードを決定するためには，オーバヘッド以外に，L1 ミスペナルティの追従性を考慮する必要がある．

3.3.2　動的動作モード決定アルゴリズム

3.3.1 項で述べたように，次の動作モードを決定するうえで，1) 動作モード切替えのオーバヘッドの見積もり方法と，2) L1 ミスペナルティの変化に対する追従性が重要となる．アプリケーション実行中に動作モードを決定するために，本アルゴリズムでは，アプリケーション実行を一定回数の L2 キャッシュアクセスで分割し（これを区間と呼ぶ），各区間ごとに適切な動作モードを決定する．現在実行している区間と，次に実行する区間の各動作モードにおける平均メモリアクセス時間の差は同程度であると想定する．この想定を基に，次の動作モードを決定する．しかしながら，現在の動作モードと異なる動作モードの平均メモリアクセス時間を求めるのは困難である．そのため，3.3.3 項で説明するキャッシュミス率推測法を用いる．

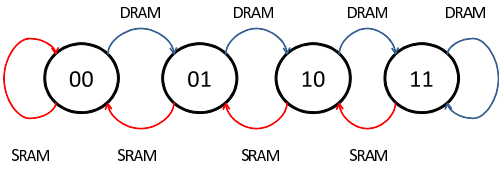

図 10　2 bit カウンタを用いた動作モード決定法
Fig. 10　Dynamic method for determination by 2 bit counter.

動作モード切替えによるオーバヘッドは現在使用しているキャッシュの情報を用いて予測する．現在使用しているキャッシュ内にあるデータは次の区間で必ず使用すると想定すると，式 (7) のように Valid ライン数 N_{Valid} と主記憶アクセス時間 $MMAT$ を用いてオーバヘッド OH を見積もることが可能である．

$$OH = N_{Valid} \times MMAT \tag{7}$$

式 (7) を用いると，SRAM キャッシュ・モードから DRAM キャッシュ・モード，DRAM キャッシュ・モードから SRAM キャッシュ・モードへ変化する条件式は 1 区間の L2 キャッシュアクセス数を ACC_{L2} とすると，以下の式 (8)，式 (9) で表せる．なお，HT_{L2_B}，HT_{L2_D}，ならびに，$MMAT$ はハードウェア特性にのみに依存しており，チップ製造後には基本的に既知と考えることができる．

$$MR_{L2\_B} - MR_{L2\_D} > \frac{HT_{L2\_D} - HT_{L2\_B} + \frac{OH}{ACC_{L2}}}{MMAT} \tag{8}$$

$$MR_{L2\_B} - MR_{L2\_D} < \frac{HT_{L2\_D} - HT_{L2\_B} - \frac{OH}{ACC_{L2}}}{MMAT} \tag{9}$$

区間を短くすると，L1 ミスペナルティに対する追従性は高くなり，区間を長くすると低くなる．3.3.1 項で述べたように両方に利点・欠点があるが，本稿では，**図 10** のようなカウンタを用いて比較的高い追従性を維持し，かつ頻繁な動作モードの切替えを防止する．現在の区間が DRAM キャッシュ・モードが高性能であると予測した場合，カウンタを 1 増加する．逆の場合はカウンタを 1 減少する．カウンタが 11（00）になったときに次の区間で DARM（SRAM）キャッシュ・モードへの切替えを行う．ただし，動作モードを切り替えた直後はウォームアップ区間として動作モード切替えやカウンタの増減を禁止する．このとき，1 区間の長さを短くすれば，L1 ミスペナルティに対する追従性は高くすることができる．また，動作モードの切替えには，カウンタの値を 11（00）にする必要があるため，頻繁に適切な動作モードが変化するプログラムでは動作モードの切替えを防止することが可能である．

3.3.3　キャッシュミス率推測法

キャッシュミス率はタグ情報のみから得ることができる．そこで，DRAM キャッシュ・モード時には，下層 SRAM のタグメモリが未使用になることを利用し，ここに SRAM

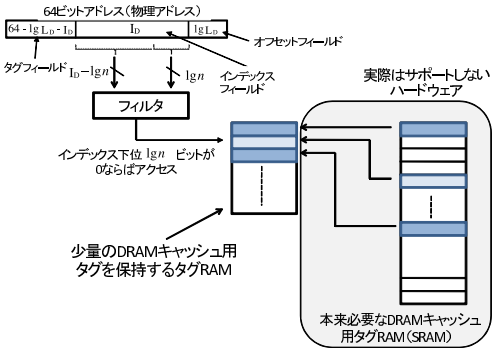

図 11　DRAM キャッシュミス率推測のためのハードウェア・サポート
Fig. 11　HW support for DRAM cache miss rate estimation.

キャッシュ・モードを想定してタグ情報を格納する．これにより，DRAM キャッシュ・モード時においても SRAM キャッシュ・モードでのヒット/ミス回数を求めることが可能となる．同様に，SRAM キャッシュ・モード時においても，DRAM キャッシュに格納可能なライン数分のタグ情報を保存するメモリを搭載すれば，DRAM キャッシュ・モードでの L2 ミス率を測定することができる．しかしながら，この場合にはきわめて大きな面積オーバヘッドが発生する．たとえば，32 MB の DRAM キャッシュにおいてラインサイズを 64 B と仮定するとタグ領域は 2.5 MB 程度必要となる．この問題を解決するため，文献 [5] で提案された手法と同様，いくつかのセットに対応する少数のタグ情報のみを保存するアプローチを採用する．

図 11 に，少数のタグを保持し，DRAM キャッシュのミス率を推測するハードウェア・サポートを示す．本来必要な容量を持つタグ RAM とは別に，キャッシュセットをサンプリングし本来必要なタグ総数の $1/n$ のみ保持する小容量のタグ RAM を下層ダイに配置する．そして，L2 キャッシュアクセスが発生した際，タグを保持すべきと判断された場合にのみ実際にそのタグ RAM へのアクセスを行う．具体的には，インデックス下位 $lg(n)$ ビットを参照し，0 であるならば別途設けたタグ RAM へアクセスする．このとき，タグ RAM に対するアクセス数，ヒット数をカウントすることでこのタグ RAM のキャッシュミス率を算出し，これを DRAM キャッシュのミス率とする．1/32 程度のタグのみ保持する場合でも高い精度でキャッシュミス率を推測することが可能である．

4.　評価実験

4.1　評価環境

本評価では，ミシガン大学で開発された M5 シミュレータ [6] を用いてトレースを取得し，メモリ性能値を算出した．評価指標は式 (1) で表す AMAT（平均メモリアクセス時間）である．提案方式においては，動作モード切替え時に発生するライトバック時間も含んでいる．具体的には，

表 1　L1/L2 キャッシュに関する設定
Table 1　Configuration of L1/L2 cache.

	L1$	SRAM$モード	DRAM-STACK DRAM$モード
サイズ	32 kB	2 MB	32 MB
連想度	2	8	8
ラインサイズ	64 B	64 B	64 B
アクセス時間 [ns]	0.640	1.962	9.292
アクセス時間 [cc]	2	6	28

表 2　主記憶に関する設定
Table 2　Configuration of main memory.

規格	DDR3 DRAM
サイズ	2 GB
バンク数	8
ページサイズ	4 KB
アクセス時間 [ns]	60.22
アクセス時間 [cc]	181

8 GB/s のメモリバンド幅を想定し，全ダーティ・ラインのデータ転送に要する時間を用いて近似した．プロセッサはシングルコア，インオーダ・命令発行，動作周波数 3 GHz を想定した．ハイブリッド・キャッシュにおいて，各メモリのアクセス時間は動作モードの選択にきわめて重要な設計パラメータとなる．そこで，**表 1**，**表 2** に示す L1 キャッシュ，SRAM キャッシュ，DRAM キャッシュ，主記憶の各種アクセス時間の決定方法について説明する．L1 キャッシュ，L2SRAM キャッシュに関しては CACTI [7] から得られる値をそのまま用いる．しかしながら，主記憶はプロセッサ・コアとオフチップ・バスで接続されているため，バスの速度も考慮する必要がある．なお，DRAM キャッシュはプロセッサ・コアと TSV による接続を仮定しており，これも 1 種のバスと考えることができる．DRAM キャッシュと主記憶のアクセス時間，特に読み出し時間のモデルを式 (10) に示す．

$$T_{access} = T_{request} + T_{read} + T_{reply} \tag{10}$$

- T_{access}：プロセッサがメモリにデータを要求してからデータが返るまでの時間 [s]
- $T_{request}$：プロセッサからメモリへデータ要求信号が届くまでの時間 [s]
- T_{read}：プロセッサがアクセスしたメモリがアドレスに従いデータを出力するまでの時間 [s]
- T_{reply}：出力データがメモリからプロセッサに届くまでの時間 [s]

式 (10) を用いて主記憶のアクセス時間を算出する．文献 [8] によれば 130 nm プロセスにおいて $T_{request} = 10$ ns である．理想的には伝播遅延はプロセス・ルールに比例するので，45 nm では $T_{request} = 3.46$ ns である．T_{read} は CACTI により算出可能であり，表 2 に示す値では，$T_{read} = 48.76$ ns である．T_{reply} はさらに式 (11) で表される．

$$T_{reply} = \frac{data\_size}{bandwidth} \tag{11}$$

- $data\_size$：転送データサイズ [B]
- $bandwidth$：プロセッサと接続バスのバンド幅 [B/s]

データサイズはキャッシュラインサイズ（64 B），バスのバンド幅は 8 GB/s を仮定すると，$T_{reply} = 8$ ns となる．したがって，T_{access} は

$$T_{access}[\text{ns}] = 3.46 + 48.76 + 8 = 60.22 \tag{12}$$

となる．次に，DRAM キャッシュのアクセス時間を算出する．積層 DRAM キャッシュはプロセッサ・コアと TSV による接続を仮定している．TSV は広バンド幅が実現可能であり [9]，各層間を短い配線長で接続可能である．したがって，ここでは $T_{request}$，T_{reply} ともに 1 クロックサイクルの遅延と仮定する．プロセッサ・コアの動作周波数は 3 GHz であるので，$T_{request} = T_{reply} = 0.333$ ns である．T_{read} は CACTI により算出可能であり，$T_{read} = 8.632$ ns である．以上により，

$$T_{access}[\text{ns}] = 0.333 + 8.632 + 0.333 = 9.292 \tag{13}$$

となる．ベンチマークプログラムは，SPEC-CPU2000 [10]（入力は train）ならびに SPLASH-2 [11]（入力は *Cholesky*：*tk29.0*，*FFT*：*4M data points*，*LU*：*1024* × *1024 matrix*，*Barnes*：*32K particles*，*FMM*：*64K particles*，*Ocean*：*258* × *258 ocean*，*WaterSpatial*：*4096 molecules*）から 14 個選択した．評価対象モデルは以下のとおりである．

- **2D-SRAM**：2 次元実装で小容量の SRAM（2 MB）を用いる従来の方式．
- **DRAM-STACK**：大容量 DRAM（32 MB）をスタックする従来の DRAM スタック法（図 1）．
- **HYBRID-IDEAL**：理想ハイブリッド・キャッシュ．各区間においてつねに適切な動作モードを選択でき，かつ，動作モード切替えにともなう性能オーバヘッドは発生しないと仮定する．なお，1 区間は L2 キャッシュアクセス 100 k 回である．ただし，これはハイブリッド・キャッシュの上限値を示すものではなく，ハイブリッド・キャッシュを指定した区間長で実行した場合の上限値である．
- **HYBRID**：1 区間の長さを 200 k 回の L2 アクセスで固定とするハイブリッド・キャッシュ．動作モード切替えにともなうオーバヘッドを考慮して動作モードの決定を行う．実行開始時の動作モードは SRAM キャッシュ・モードであり，SRAM キャッシュ・モード時の DRAM キャッシュミス率推定用タグは本来必要な量の 1/32 を搭載している．このモデルはカウン

タが 1 bit の場合に相当する．

- **HYBRID-2 bit**：1 区間の長さを 100 k 回の L2 アクセスで固定とするハイブリッド・キャッシュ．動作モード切替えにともなうオーバヘッドを考慮し，図 10 に示す 2 bit カウンタを用いて動作モードの切替えを制御する．実行開始時の動作モードは SRAM モードで，2 bit カウンタの初期値は 01 としている．
- **HYBRID-3 bit**：1 区間の長さを 200 k 回の L2 アクセスで固定とするハイブリッド・キャッシュ．動作モード切替えにともなうオーバヘッドを考慮し，3 bit カウンタを用いて動作モードの切替えを制御する．実行開始時の動作モードは SRAM モードで，3 bit カウンタの初期値は 011 としている．

HYBRID-IDEAL, HYBRID, HYBRID-2 bit, HYBRID-3 bit における区間の長さの決定理由に関しては，4.2 節で説明する．

4.2　評価結果

最初に，HYBRID-IDEAL，HYBRID，HYBRID-2 bit，HYBRID-3 bit において，1 区間の長さを 100 k，200 k，500 k，ならびに，1 M 回の L2 アクセスに固定した場合の性能評価結果を**図 12**，**図 13**，**図 14**，**図 15** に示す．図 12〜図 15 の縦軸は，各ベンチマークで区間の長さが 100 k の場合の AMAT を 1 として正規化を行っている．これらの図から分かるように，各モデルにおいて平均で最も性能が高くなるのは，HYBRID-IDEAL では 100 k，HYBRID では 200 k，HYBRID-2 bit では 100 k，HYBRID-3 bit では 200 k である．以下の性能評価では各モデルの 1 区間の長さとして，これらの値を用いている．性能評価結果を**図 16** に示す．縦軸は，各ベンチマークプログラムで 2D-SRAM の AMAT を 1 として正規化した値である．

まず，ハイブリッド・キャッシュの潜在的な性能向上について考察する．HYBRID-IDEAL はオーバヘッドを考慮せずに，区間ごとに適切な動作モードを選択可能である．このため，2D-SRAM や DRAM-STACK に比べすべてのベンチマークプログラムにおいて，性能が向上する．平均では，2D-SRAM，DRAM-STACK に比べ，約 20%，約 17%の AMAT を削減している．

次に，実行時動作モード決定アルゴリズムに基づくハイブリッド・キャッシュについて考察する．傾向として，L1 ミスペナルティの追従性が低くなる（カウンタの bit 数が増える）につれて AMAT が増加するプログラムと減少するプログラムの 2 通りがある．これは，プログラムごとに適した L1 ミスペナルティの追従性が異なるためである．平均すると，2 bit カウンタを用いた HYBRID-2 bit が最も低く，2D-SRAM と比較して，平均で約 15%の AMAT を削減している．

以下は，各ベンチマークプログラムについて各動的動作モード決定法と AMAT の変化について考察する．各動的動作モード決定法の AMAT には，適切な動作モードを選択した区間の割合と SRAM キャッシュ・モードと DRAM キャッシュ・モードの性能差が大きく影響している．**図 17** は，各動的動作モード決定法における，適切な動作モードを選択した割合を表している．この割合は以下の式を用いて求めている．

$$R = \frac{N_{correct}}{N_{interval}} \tag{14}$$

- R：適切な動作モードを選択した割合
- $N_{correct}$：適切な動作モードを選択した区間数
- $N_{interval}$：総区間数

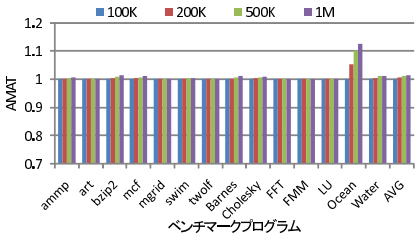

図 12　区間長が性能に与える影響（HYBRID-IDEAL）
Fig. 12　Effect of interval length: HYBRID-IDEAL.

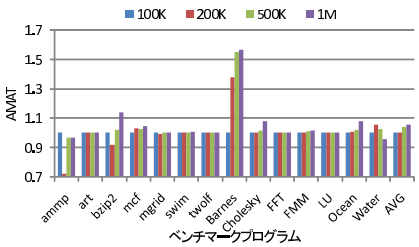

図 13　区間長が性能に与える影響（HYBRID）
Fig. 13　Effect of interval length: HYBRID.

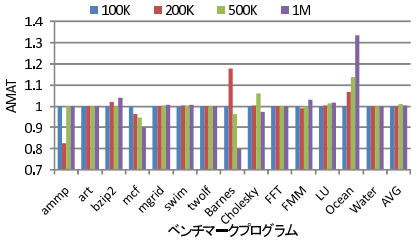

図 14　区間長が性能に与える影響（HYBRID-2 bit）
Fig. 14　Effect of interval length: HYBRID-2 bit.

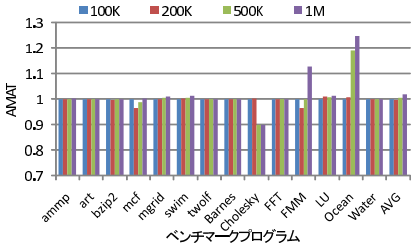

図 15　区間長が性能に与える影響（HYBRID-3 bit）
Fig. 15　Effect of interval length: HYBRID-3 bit.

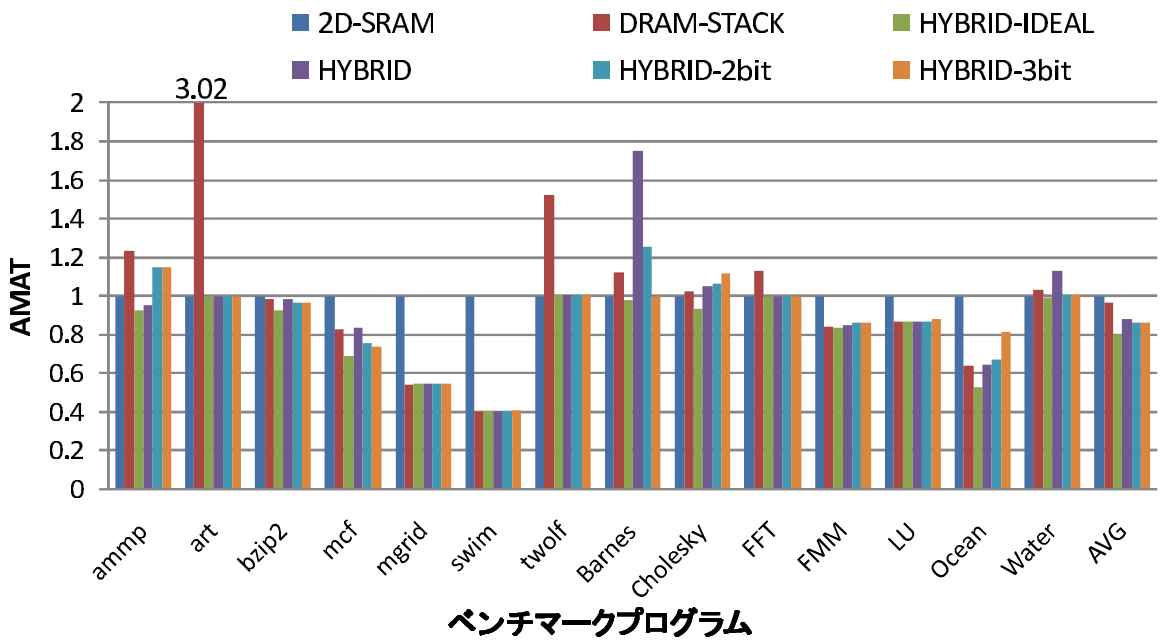

図 16　性能評価結果

Fig. 16　Performance result.

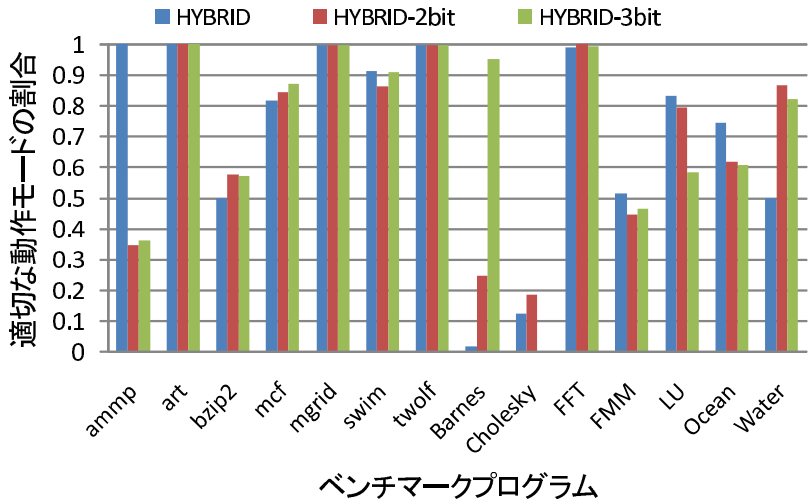

図 17　実行中に適切な動作モードを選択した割合

Fig. 17　Ratio of high performance mode.

また，ここでいう適切な動作モードを選択した区間とは，以下の条件を満たした区間のことである．

$$if SRAM cache mode$$
$$MP_{SRAM}(i) + MP_{OH}(i) < MP_{DRAM}(i) \quad (15)$$
$$if DRAM cache mode$$
$$MP_{DRAM}(i) + MP_{OH}(i) < MP_{SRAM}(i) \quad (16)$$

- $MP_{OH}(i)$：区間 i のオーバヘッドによる L1 ミスペナルティの増加量
- $MP_{DRAM}(i)/MP_{SRAM}(i)$：区間 i のオーバヘッドを考慮していない DRAM/SRAM キャッシュ・モードの L1 ミスペナルティ

式 (15)，式 (16) において，左辺が実際に各モデルを実行した際のトレースから得られる値，右辺が事前実行のトレースから得られる値である．

art や *mgrid*，*swim*，*twolf*，*FFT*，は，ほとんどの区間で適切な動作モードを選択することができている．そのため，HYBRID-IDEAL と AMAT の差はとても小さい．一方，*bzip2* や *Cholesky* は適切な動作モードを選択している割合は小さいが，2D-SRAM と DRAM-STACK の AMAT 差が小さい．このため，これらのプログラムでは，AMAT の増加が小さい．*ammp* や *mcf*，*Barnes* は 2D-SRAM と DRAM-STACK の AMAT 差が大きく適切な動作モードを選択した割合の増減に応じて，AMAT も大きく変化している．**図 18** は *Barnes* を実行したときの L1 ミスペナルティの変化を示している．図 18 から HYBRID と HYBRID-IDEAL の差が大きいことが分かる．このため，HYBRID

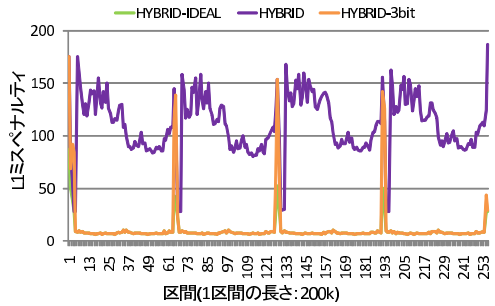

図 18　プログラム実行中の L1 ミスペナルティの変化：Barnes
Fig. 18　Run-time variation of L1 misspenalty: Barnes.

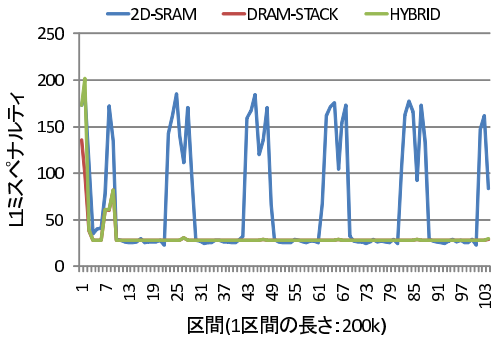

図 19　プログラム実行中の L1 ミスペナルティの変化：FMM
Fig. 19　Run-time variation of L1 misspenalty: FMM.

は HYBRID-IDEAL に比べ大きく AMAT が増加している．しかしながら，HYBRID-3 bit と HYBRID-IDEAL の差は小さくなっており，AMAT の差は非常に小さくなっている．*FMM*，*LU* は 2D-SRAM と DRAM-STACK の AMAT 差は大きく，さらに，適切な動作モードの割合も比較的少ない．**図 19** に *FMM* を実行したときの L1 ミスペナルティの変化を示す．図 19 から分かるように，実行中に 2D-SRAM と DRAM-STACK の L1 ミスペナルティ差が小さい区間が多い（たとえば区間 10～22）．これらの区間で適切な動作モードの条件を満たしていないため，適切な動作モードで実行した区間の割合は小さい．しかしながら，2D-SRAM と DRAM-STACK の L1 ミスペナルティ差が大きい区間（たとえば区間 23～30）では適切な動作モードを選択している．このため，全体の AMAT では，HYBRID-IDEAL とほぼ同じとなっている．

5.　おわりに

本稿では，3 次元積層 DRAM を活用する新しいメモリ構成法として SRAM/DRAM ハイブリッド・キャッシュを提案した．評価実験の結果，SRAM/DRAM ハイブリッド・キャッシュは，従来手法と比較し平均約 15%の性能が向上し，有効であることを確認した．本稿では，主記憶アクセスに要する時間は 60.22 ns と仮定した．しかしながら，この値は DRAM コントローラの設計や実装基板上での信号伝搬遅延などに大きく依存する．そこで，今後，主記憶アクセス時間の変化が提案方式の性能に与える影響を詳細に解析する予定である．また，本稿の評価では，プロセッサはインオーダ・命令発行を想定している．しかしながら，アウトオブオーダ・命令発行のプロセッサを用いた場合やノンブロッキングキャッシュを搭載する場合，メモリアクセス時間が隠ぺいされる可能性がある．このとき，提案手法によるメモリアクセス時間の削減効果は小さくなる可能性がある．そのため，シミュレータを用いたサイクルレベルのシミュレーションを行い，提案手法の詳細な評価を行う予定である．さらに，提案手法の発展として，区間長を可変にした動的制御方式やプログラムごとに動作モードを決定および設定する静的制御方式の検討，消費エネルギーについての評価も行う予定である．

謝辞　日頃から御討論いただいております九州大学安浦・村上・松永・井上・石原研究室ならびにシステム LSI 研究センターの諸氏に感謝します．本研究は主に九州大学情報基盤研究開発センターの研究用計算機システムを利用しました．なお，本研究は，独立行政法人新エネルギー・産業技術総合開発機構（NEDO）若手グラントの支援による．

参考文献

[1] 橋口慎哉，福本尚人，井上弘士，村上和彰：3 次元積層 SRAM/DRAM ハイブリッド・キャッシュ，技術報告 (2011).
[2] Black, B., Annavaram, M., Brekelbau, N., DeVale, J., Jiang, L., Loh, G.H., McCauley, D., Morrow, P., Nelson, D.W., Pantuso, D., Reed, P., Rupley, J., Shankar, S., Shen, J. and Webb, C.: Die Stacking (3D) Microarchitecture, *MICRO 39: Proc. 39th Annual IEEE/ACM International Symposium on Microarchitecture*, pp.469–479, IEEE Computer Society (online), DOI: 10.1109/MICRO.2006.18 (2006).
[3] Loh, G.H.: 3D-Stacked Memory Architectures for Multicore Processors, *SIGARCH Comput. Archit. News*, Vol.36, No.3, pp.453–464 (online), DOI: 10.1145/1394608.1382159 (2008).
[4] Puttaswamy, K. and Loh, G.H.: Implementing Caches in a 3D Technology for High Performance Processors, *ICCD '05: Proc. 2005 International Conference on Computer Design*, pp.525–532, IEEE Computer Society (online), DOI: 10.1109/ICCD.2005.65 (2005).
[5] Qureshi, M.K. and Patt, Y.N.: Utility-Based Cache Partitioning: A Low-Overhead, High-Performance, Runtime Mechanism to Partition Shared Caches, *MICRO 39: Proc. 39th Annual IEEE/ACM International Symposium on Microarchitecture*, pp.423–432, IEEE Computer Society, Washington, DC, USA (2006).
[6] Binkert, N.L., Dreslinski, R.G., Hsu, L.R., Lim, K.T., Saidi, A.G. and Reinhardt, S.K.: The M5 Simulator: Modeling Networked Systems, *IEEE Micro*, Vol.26, No.4, pp.52–60 (online), DOI: 10.1109/MM.2006.82 (2006).
[7] Thoziyoor, S., Muralimanohar, N., Ahn, J.H. and Jouppi, N.P.: CACTI5.1, Technical Report, HP Lab (2008).
[8] Loi, G.L., Agrawal, B., Srivastava, N., Lin, S.-C., Sherwood, T. and Banerjee, K.: A thermally-aware performance analysis of vertically integrated (3-D) processor-memory hierarchy, *DAC '06: Proc. 43rd Annual De-*

sign Automation Conference, pp.991–996, ACM, New York, NY, USA (online), DOI: 10.1145/1146909.1147160 (2006).
[9] Kgil, T., D'Souza, S., Saidi, A., Binkert, N., Dreslinski, R., Mudge, T., Reinhardt, S. and Flautner, K.: PicoServer: Using 3D Stacking Technology To Enable A Compact Energy Efficient Chip Multiprocessor, *SIGOPS Oper. Syst. Rev.*, Vol.40, No.5, pp.117–128 (online), DOI: 10.1145/1168917.1168873 (2006).
[10] Henning, J.L.: SPEC CPU2000: Measuring CPU Performance in the New Millennium, *Computer*, Vol.33, pp.28–35 (online), DOI: 10.1109/2.869367 (2000).
[11] Woo, S.C., Ohara, M., Torrie, E., Singh, J.P. and Gupta, A.: The SPLASH-2 Programs: Characterization and Methodological Considerations, *ISCA '95: Proc. 22nd Annual International Symposium on Computer Architecture*, pp.24–36, ACM (online), DOI: 10.1145/223982.223990 (1995).

上野 伸也

昭和 61 年生．平成 21 年九州大学工学部電気情報工学科卒業．平成 23 年同大学大学院システム情報科学府修士課程修了．平成 23 年同博士後期課程に進学，現在に至る．3 次元積層技術を用いたプロセッサ・アーキテクチャに関する研究に従事．

橋口 慎哉

昭和 61 年生．平成 21 年九州大学工学部電気情報工学科卒業．平成 23 年同大学大学院システム情報科学府修士課程修了．同年富士通（株）入社，現在に至る．汎用計算機に関する研究開発に従事．平成 22 年山下記念研究賞受賞．

福本 尚人　（学生会員）

昭和 59 年生．平成 19 年九州大学工学部電気情報工学科卒業．平成 21 年同大学大学院システム情報科学府修士課程修了．平成 21 年同博士後期課程に進学，現在に至る．マルチコア・プロセッサに関する研究に従事．

井上 弘士　（正会員）

昭和 46 年生．平成 8 年九州工業大学大学院情報工学研究科修士課程修了．同年横河電機（株）入社．平成 9 年より（財）九州システム情報技術研究所研究助手．平成 11 年の 1 年間 Halo LSI Design & Device Technology, Inc. にて訪問研究員としてフラッシュ・メモリの開発に従事．平成 13 年九州大学にて工学博士を取得．同年福岡大学工学部電子情報工学科助手．平成 16 年より九州大学大学院システム情報科学研究院助教授．平成 19 年 4 月より同大学准教授，現在に至る．高性能/低消費電力プロセッサ/メモリ・アーキテクチャ，ディペンダブル・アーキテクチャ，3 次元積層アーキテクチャ，性能評価，等に関する研究に従事．電子情報通信学会，ACM，IEEE 各会員

村上 和彰　（正会員）

昭和 35 年生．昭和 59 年京都大学大学院工学研究科情報工学専攻修士課程修了．同年富士通（株）入社．汎用大型計算機の研究開発に従事．昭和 62 年九州大学助手．平成 6 年九州大学助教授．現在，九州大学大学院システム情報科学研究院情報理学部門教授，情報基盤研究開発センター長，情報統括本部長．計算機アーキテクチャ，並列処理，システム LSI 設計技術，等に関する研究に従事．工学博士．平成 3 年情報処理学会研究賞，平成 4 年情報処理学会論文賞，平成 9 年坂井記念特別賞，平成 12 年日経 BP 社 IP アワード，平成 12 年情報処理学会創立 40 周年記念論文賞，平成 14 年電子情報通信学会業績賞をそれぞれ受賞．