省電力ページング方式を実装した 次世代メモリアーキテクチャ上での並列プログラムの評価

細萱祐人[†,†††] 遠藤敏夫[†,†††] 松岡聡[†,††,†††]

近年大規模計算機の省電力化の要求が高まるにつれて，CPU だけでなくメモリの消費エネルギーの削減が重要視されつつある．メインメモリとして使用される DRAM は揮発性メモリであるため，電力コストが非常に大きい．しかし，スワップを避けるように設計される HPC では必要以上に DRAM を搭載しており，その結果多くの場合搭載された全メモリは使われてはいない．そこで，我々は DRAM の搭載容量を削減するためにメインメモリに MRAM と DRAM，スワップデバイスとして FLASH メモリを使用した低消費電力メモリアーキテクチャを提案する．本アーキテクチャはメモリアクセスを高速な MRAM に集中させ，かつ FLASH へのアクセスを減らす省電力ページング方式を実装している．本アーキテクチャをシミュレーションにより評価した結果，DRAM 搭載容量を削減した場合に，アプリケーションベンチマークの性能低下を 12%に抑え，メモリモジュールの消費エネルギーを 26%に削減できることを示した．

Performance Evaluation of Parallel Applications on Next Generation Memory Architecture with Power-Aware Paging Method

Yuto Hosogaya,[†,†††] Toshio Endo[†,†††] and Satoshi Matsuoka[†,††,†††]

With the increasing demand for low power high performance computing, reducing power of not only CPUs but also memories is becoming important. In typical HPC environments larger capacity of DRAM than needed is installed to avoid memory swapping, although not all of the memory is used in many cases. Since DRAM is volatile, such unused memory can waste a significant amount of power even in a standby state. We propose a next generation low power memory system that reduces the DRAM capacity while minimizing application performance degradation. In this architecture, both DRAM and MRAM, fast non-volatile memory, are used as a main memory, while FLASH memory is used as a swap device. Our profile-based paging algorithm optimizes memory accesses by reducing accesses with slower memories and using faster memories as much as possible. Simulated results of our architecture show that the energy consumption of memory system can be reduced to 26% by reducing DRAM capacity, with 12% performance loss of application benchmarks.

1. はじめに

近年高集積化が進んだ HPC 向け計算機システムの消費電力は非常に増加しており，その省電力化が大規模システムを設計・運用する上で最大の関心事の一つとなっている．多くの場合に最大の電力を消費する CPU については，DVS を使った研究などが広く行われている[1)]．一方，システムは高集積化・廉価化したメモリを大量に搭載するようになり，メモリが消費するエネルギーの割合が大きくなってきている．大規模システムではキャッシュやメモリコントローラを含めたメモリモジュールが全体の 41%の電力を消費しうるという指摘もある[2)]．

大規模 HPC システムの設計においては以下の理由により，メモリは必要以上に搭載される傾向が強い．様々な特性を持つ多数のアプリケーションを良好な性能で動作させるためには， HPC アプリケーションの速度性能に対して致命的なメモリスワップの発生を最小限にする必要がある．そのため，システムのメモリ容量はメモリ使用量が最大のアプリケーションを基準に決定される場合が多い．その結果，実際の運用環境では， その使用率はそれほど高くない傾向にある[3)]．

現在メインメモリとして広く使用されている DRAM

† 東京工業大学
Tokyo Insitute of Technology
†† 国立情報学研究所
National Institute of Informatics
††† JST, CREST

は揮発性メモリであり，データを保持し続けるために大きな電力コストを要する．また，HPC アプリケーションが要求するメモリサイズはますます増加しており，これに答えるようにより大容量の DRAM を搭載することは，電力コストの面や，単価が低下しているとはいえ導入コストの面から見ても現実的ではない．そのため，アプリケーション性能を維持した上で，搭載する DRAM の容量を削減することが重要になってきている．

そこで本研究では，DRAM の低容量化を実現する以下のようなメモリアーキテクチャを提案する．まず，次世代の高速かつ低電力メモリである MRAM をメインメモリとして DRAM の一部と置き換え，メモリアクセスの高速化を図る．さらに，メインメモリ容量を削減することで発生するスワップのコストを軽減するために，HDD よりもアクセス速度や電力的にも優れる FLASH メモリをスワップデバイスとして使用する．また，これらの混在したメモリの特徴を最大限活用できるような省電力ページング方式を実装する．このアーキテクチャの性能を見積もるために，メモリモジュールの性能モデルを構築し，複数のアプリケーションベンチマークを用いたシミュレーションにより評価を行った．その結果，最適な容量の場合にはスワップを起こさない十分な容量の DRAM を搭載したアーキテクチャと比較して，性能低下を 12%に抑えつつ，メモリチップの合計消費エネルギーを 26%に削減できることを示した．また，提案するアーキテクチャではスワップを使用しても妥当な性能で計算を実行できており，これは具体的な評価は行っていないが HDD をスワップデバイスとして使用するアーキテクチャより実行速度，消費エネルギーの両面で大幅によい性能である．このことから我々のアーキテクチャでは現在のメモリ資源でもより大きなサイズのアプリケーションを妥当な性能で実行できることがわかる．さらに，用いたアプリケーションのうち CG，HPL においては HPC においても消費エネルギーを最小とするためには，スワップを起こしてでも DRAM 容量を削減するほうが良いことが分かった．現在のところ評価は一台のマシンで行っているが，本研究の成果を大規模・複数アプリケーション環境に拡張し，将来の省電力大規模 HPC システムの設計に活用することをめざしている．

2. 不揮発性メモリ

FLASH は不揮発性メモリの中で，現在最も使用が広まっているデバイスである．100MB/s 近くの読み出し性能を持つ Solid State Disk(SSD) や，HDD に FLASH を付随させた HHDD(Hybrid Hard Disk Drive) がすでに製品化され，HDD の代替または補助的に使われ始めている．また Fusion-io[4] は PCI Express で接続される FLASH ストレージデバイス ioDrive を開発し，これは 8KB の連続読み込みで 800MB/s の性能を持っている．これらのソフトウェアからのサポートとしては，USB メモリを簡単にスワップデバイスとして拡張することが可能な Windows Vista の機能である Ready-Boost[5] などが挙げられる．今後更に，MLC(Multi-level Cell) や 3 次元構造の実現により FLASH の集積は増し，速度向上すると期待される．本研究でも FLASH をスワップデバイスとして用いることにより省電力化を図るが，主な対象は HPC アプリケーションであり，デスクトップアプリケーションとは特性が大きく異なる．

FLASH は，書き込み速度が低速であることや書き換え回数に制限があるため，頻繁にデータを書き換える可能性のあるメインメモリとして用いるのは困難である．一方，メインメモリとして有望な不揮発性メモリとして，Magnetoresistive RAM (MRAM)[6]，Phase-change RAM (PRAM)，Ferroelectric RAM (FeRAM) などの開発が行われてきている．中でも MRAM は，DRAM よりも高速で，はるかに低消費電力であることが期待される点，小容量ではあるものの 2006 年に Freescale[7] により製品が発売されている点から最も注目されている．しかし，現時点で集積度の向上や書き込み電力の削減等に課題を抱えている．また，大容量製品の量産化がなされるまではビット単価が DRAM よりも高価であることが予想されるため，HPC 用システムの全メインメモリを MRAM で構成するのは難しいと考えられる．そのため本論文では，メインメモリの一部を MRAM に置き換えるアーキテクチャを提案し，その容量と性能の関係について詳細に議論する．

3. 提案アーキテクチャ

本研究では異種メモリを混載した次世代の低電力メモリアーキテクチャを提案する (図 1)．メインメモリに DRAM のキャッシュとしてではなく，メモリ階層の同じレベルで MRAM を使用することで DRAM 容量の削減を実現している．また，スワップを考慮する本アーキテクチャでは，シークを伴う HDD よりもはるかに高速かつ低電力な FLASH をスワップデバイスとして使用しており高速なスワップを実現している．

3.1 省電力ページング方式

本アーキテクチャには，混載したメモリの特徴を最大限生かすための省電力ページング方式を実装している．混載したメモリのうち，より高速かつ低電力な MRAM にメモリアクセスを集中させるようなデータ配置を行うことにより性能向上が期待できる．現在のところ，ページング方式は以下のような情報が得られることを仮定し，利用している．まず各ページについてアプリケーション実行を通した総メモリアクセス回

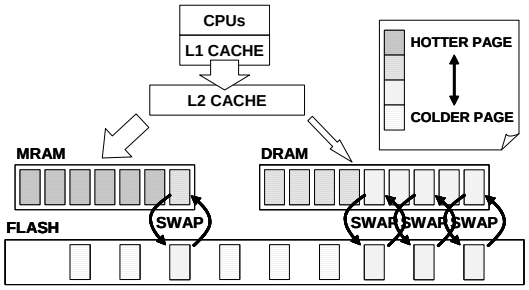

図 1 提案システムの概要

数が得られるとし，それが多いページをホットページ，少ないものをコールドページと呼ぶ．また実行中の任意の時点で，実行開始からの DRAM，MRAM それぞれへのメモリアクセス回数を得ることができるとする．前者の仮定については，繰り返しの多い HPC アプリケーションには適用可能と考えられるが，今後適用範囲を広げるために動的プロファイルを用いるなどにより仮定を弱めていきたい．

メインメモリ上のページの複製が常にスワップデバイスに存在すれば，スワップアウト時までに更新が行われなかったクリーンページを改めて書き戻す必要はない．そのためこの書き戻しを省略することでページを書き戻す I/O 処理を削減できる．この方法は一般的にも知られているが[8]，特に我々のアーキテクチャにおいては，FLASH の書き込みが読み込みより遅い傾向にあること，デバイスの書き換え回数が限られていることから，大きな利点を持つ．以下で示す方式はこの方法を前提とする．

以下では，ページング方式について，素朴なものから順に改良することにより説明を行う．ページフォルトが発生した場合，多くの OS では LRU(実際には近似した方式) によりスワップアウトするページを決定する．しかしメモリの種類を同一視して単純に LRU を用いると，異種メモリの使い分けができない．また，以下のようなケースが考えられる．アプリケーションを通して 1 度しかアクセスされない (コールド) ページがたまたまページフォルトの起こる直前にアクセスされると，それはスワップされず，別のホットかもしれないページがスワップアウトされてしまう．

そのため，ホットページを MRAM に集めるために以下のような方式 (以後，MRAM fixed 方式と呼ぶ) を考える．ページを，プロファイルから得られる総アクセス回数が多い順に，MRAM の容量が許す限り配置する．それらはスワップアウトの対象から外し，DRAM 領域のみで LRU を行う．これにより，ホットページを常に MRAM 上に置いておくことが可能となる．しかしながら，予備実験によりこの方式はアプリケーションによっては性能が低下し，単純な LRU よりも 2 倍近く悪くなる場合があった．それは特にメモリアクセスの局所性の小さい，すなわちホットページ

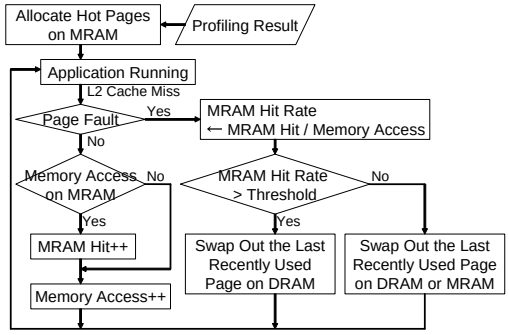

図 2 提案ページング方式

とコールドページのアクセス回数の差が小さいアプリケーションにおいて見られた．これは，ホットページをメインメモリ上に固定する利点が少ないにも関わらず，DRAM 領域のみ，つまり単純な LRU の場合より少ない領域のみを使いまわすため，スワップ回数が増加することが原因であった．

そこで，アプリケーションの性質を示す指標として，MRAM ヒット率を導入し，これを利用したページング方式を提案する．提案するページング方式の処理の流れを図 2 に示す．MRAM ヒット率は，アプリケーション実行開始から任意の時点までの，MRAM への総アクセス回数の全メインメモリへの総アクセス回数に対する比率である．プロファイルによってホットページが MRAM 上に集められているという仮定から，この値が大きいことはホットページとコールドページのアクセス頻度の差が大きいことを示している．そして，MRAM ヒット率と次式で定義される閾値を比較することでアプリケーションのメモリアクセス局所性を判断する．

$$Thr = \alpha \times MRAM\_SIZE/TOTAL\_SIZE$$

上式中の MRAM_SIZE および，TOTAL_SIZE はそれぞれ搭載した MRAM の容量，MRAM と DRAM の合計容量を示す．また，$\alpha(\approx 1)$ は調節可能な定数である．ページフォルトが発生した時点で，MRAM ヒット率が閾値を上回れば，DRAM 上のページからのみスワップアウトするページを選択し，逆に閾値を下回れば両方の RAM からスワップアウトするページを探す．この操作によりアプリケーションの性質に沿った適切なページングが可能となる．なお，α は小さいほど MRAM 上のページをメモリ上に残しやすくする．この値は予備実験の結果から 0.9 とした．

3.2 性能モデル

我々は，本アーキテクチャの性能を見積もるためにアプリケーションの実行時間とメモリチップが消費するエネルギーのモデルを構築した．

メモリアクセスによる遅延の合計は 1 アクセスあたりの遅延とアクセス回数の積によって決定する．この 1 アクセスあたりの遅延は，メモリの種類，アクセス

表 1 シミュレート環境

L2 cache サイズ	1MB
ブロックサイズ	64B
連想度	1
ページサイズ	4KB

表 2 設定したパラメータ

	DRAM	MRAM	FLASH
アクセス単位 (Byte)	64	64	4096
読み込み遅延 (ns)	22.5	15	35000
書き込み遅延 (ns)	22.5	15	64000
読み込みエネルギー (nJ)	6.24	1.36	7000
書き込みエネルギー (nJ)	6.24	2.60	12800
待機電力 (μW/MB)	867	469	0

単位によって異なる．アプリケーションの実行時間は，このメモリアクセスによる遅延とメモリアクセスと非依存な計算時間の和によって決定する．

メモリアクセスによる動的消費エネルギーの合計は，１アクセスあたりの消費エネルギーとアクセス回数の積によって決定する．メモリチップの消費エネルギーはこの動的消費エネルギーと，メモリ搭載容量と実行時間に比例して増加する待機電力による静的消費エネルギーの和によって決定する．

4. 評　　価

本研究は次世代のメモリアーキテクチャを想定しており，現在では我々が期待する性能を持った MRAM および FLASH を入手困難なため，シミュレーションによって評価を行った．シミュレーションにあたり，まず HPC アプリケーションのメモリアクセスのトレースファイルを作成した．このトレースは L2 キャッシュまでをシミュレートするように改造した Valgrind[9] を用いて行った．表 1 にシミュレートした環境を示す．トレースの結果，時系列に従って L2 キャッシュミスを伴うアクセスのアドレスと実行される処理 (READ または WRITE) がファイルに出力される．

モデルに当てはめる各メモリのパラメータを表 2 に示す．表のアクセス遅延，消費エネルギーは DRAM と MRAM は L2 キャッシュのブロックサイズ，FLASH はページサイズへの 1 アクセスあたりの値である．DRAM の値は一般的な 64MB チップの DDR3 の性能を基準に決定した．FLASH の値は 58MB/sec の読み出し速度と 32MB/sec の書き込み速度を持つ Samsung[10] の SSD を基準とし，速度、電力共に今後数年で 2 倍の性能となると仮定して設定した．また，我々のアーキテクチャでは FLASH を大量に搭載することを仮定しているため，その待機電力は無視した．

現在製品化されている MRAM は DRAM よりも低速であるため，その速度は利用しなかった．代わりに，現在，NEC[11] は SRAM と同等のスピードを持つ MRAM チップを開発しているため，MRAM の速度を DRAM の 1.5 倍高速に設定した．アクセス時の電力は読み込み時に 182mW，書き込み時に 346mW 消費する Freescale の MRAM チップを基準にし，これは今後数年で半減できるとして値を設定した．上で設定したアクセス遅延と電力の積から表に示すエネルギーを求めた．待機電力は以下に示す仮定から見積もった．まず，現在 60mW である Freescale の待機電力は同様に半減できるとする．また，DRAM のチップが 64MB であるのに対し，MRAM のチップは 0.5MB であり，集積度の差は大きいが，将来 MRAM の集積度も DRAM 並に増加するとする．これらの仮定が今後何年で実現するかどうかの明言は困難だが，このパラメータは今後の MRAM および FLASH の技術動向を考慮して再考していきたい．

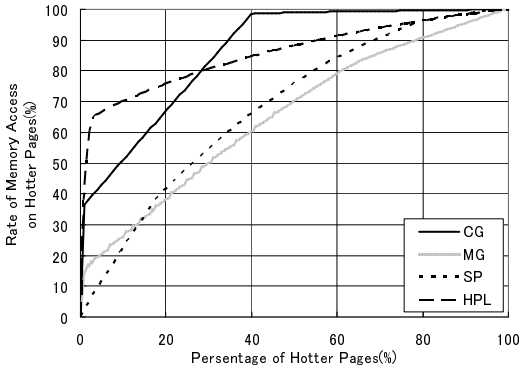

図 3 メモリアクセス局所性の違い

評価に用いたベンチマークは Nas Parallel Benchmark3.2[12] の CG(class B)，MG(class A)，SP(class B) と HPL[13] で，それぞれ 1 ノードで実行した．トレースファイルのサイズの関係上，NPB の各アプリケーションは Iteration5 回で行い，HPL は Matrix size を 8192，Block size を 256 に設定し行った．その結果，CG，MG，SP，HPL のメモリ使用量は 412MB，448MB，344MB，544MB であった．実環境の HPC システムのメモリ使用量は全体の約 3〜5 割ほどである傾向にあるため[3]，これらのアプリケーションを動かす典型的なアーキテクチャとして，DRAM のみ 1GB 搭載したものを想定し，主な比較対象とする．

これらのアプリケーションのメモリアクセスの局所性は，本アーキテクチャの性能に非常に大きな影響を与える．図 3 はホットページへのアクセスが全体のアクセスに対してどの程度占めているかをグラフ化したものである．このグラフから CG，HPL はメモリアクセスの局所性が大きく，一方 MG，SP は全使用メモリにほぼ同様なアクセスがあることがわかる．

4.1 ページング方式の評価

まず，提案したページング方式の評価を行う．図 4

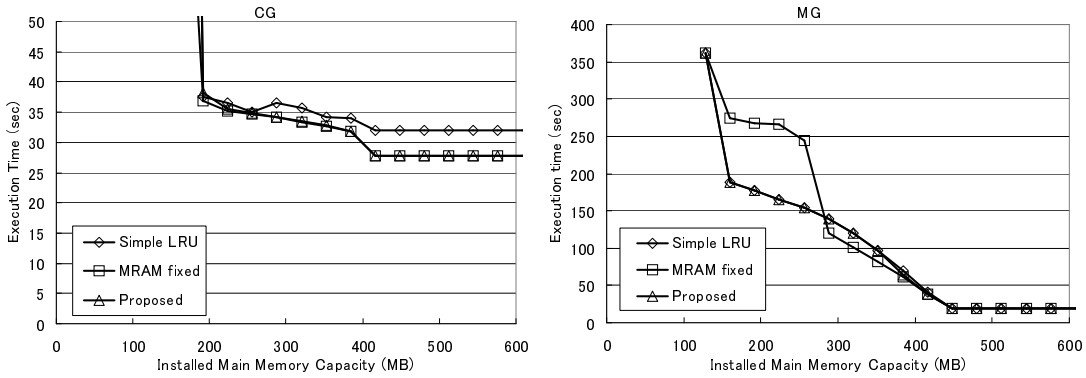

図 4 ページング方式の評価 - MRAM 128MB

は搭載 DRAM 容量に従う CG，MG の実行時間の推移を示している．各系列はページング方式の違いを示しており，それぞれ単純な LRU を用いた方式，MRAM fixed 方式，提案ページング方式である．グラフは 128MB の MRAM を搭載した場合の結果であるが，搭載 MRAM 容量が変わっても同様な挙動を見せた．また，搭載する DRAM，MRAM 容量が等しい場合消費エネルギーはほぼ実行時間にのみ依存するため，消費エネルギーのグラフは示していない．

CG，MG ともに提案するページング方式がほとんど場合で最も性能がよい結果となった．CG は高いメモリアクセス局所性を持つため，プロファイルを行い MRAM 上にホットページを置くことで，より多くのメモリアクセスを高速な MRAM に集めることができ，最大 15%ほど性能を改善している．一方その局所性から，ほとんどの場合 MRAM ヒット率が閾値を上回り MRAM 上のページは固定されたため，提案方式と MRAM fixed が同等の良好な性能を示した．一方，メモリアクセスの局所性が小さい MG は，搭載メインメモリ容量が小さい時，MRAM fixed 方式では，3.1 節で指摘した弊害が発生していることが分かる．閾値を導入し，MRAM からスワップするかどうかを動的に決定することでこの弊害を防いでいることがわかる．

4.2 アプリケーション実行時間の評価

図 5 は提案ページング方式のもとで搭載するメインメモリ容量に従う，アプリケーションの実行時間の推移を示している．グラフの各系列は搭載する MRAM 容量を示している．グラフから各アプリケーションは，スワップが発生すると顕著に性能が低下するアプリケーションと，性能がある程度維持できるアプリケーションに分かれることがわかる．例えば，CG，HPL はメインメモリ容量を削減してもある程度は性能が維持できているのに対して，MG，SP は少しのスワップの使用で性能が顕著に低下している．

これは先に述べた各アプリケーションのメモリアクセスの局所性に起因している．メインメモリ上にページがホットページから順番に乗っていると仮定し，メインメモリ容量がアプリケーションのメモリ使用量の 40%であるとする．この時，単純に考えると，図 3 の 40%の垂直線のグラフより上の分のアクセスがページフォルトを伴うアクセスである．このページフォルトを伴うアクセスはメモリアクセスの局所性の大きい CG は全体の 2%以下であるのに対し，局所性の小さい MG は約 40%である．このようにメモリアクセスの局所性の小さいアプリケーションでは少しでもメインメモリ容量が不足するとスワップ回数の急激に増加し，顕著に性能が低下してしまう．なお CG を除いて，MRAM 容量を変化させる影響は実行時間には非常に小さい．

4.3 消費エネルギーの評価

図 6 は提案ページング方式のもとで搭載するメインメモリ容量に従う，アプリケーション実行時にメモリチップで消費するエネルギーの推移を示している．待機電力による静的消費エネルギーはメインメモリ容量に比例するため，搭載メインメモリ容量を増加させると消費エネルギーは増加する．一方，スワップが発生すると FLASH へのアクセスによる動的消費エネルギーが増加することに加え，実行時間が増加するため長い時間システムを稼動することになり，待機電力による静的消費エネルギーも増加する．

メインメモリにおける MRAM の比率が高くなると実行時間はほとんど変化無い一方，消費エネルギーは減少することがわかる．また，SP を除いた各アプリケーションにおいて，消費エネルギーが極小となる 2 つのメインメモリ容量が存在している．1 つはスワップが起こらない最小の容量であり，もう 1 つはスワップが発生する容量の中にある．特にメモリアクセス局所性の大きい CG，HPL に注目するとスワップが起こらない最小の容量よりもスワップを伴う容量の方が，消費エネルギーが小さい．このことから消費エネルギーの観点から見た場合，HPC でもアプリケーションに応じてスワップを使用することが有効であることがわかる．

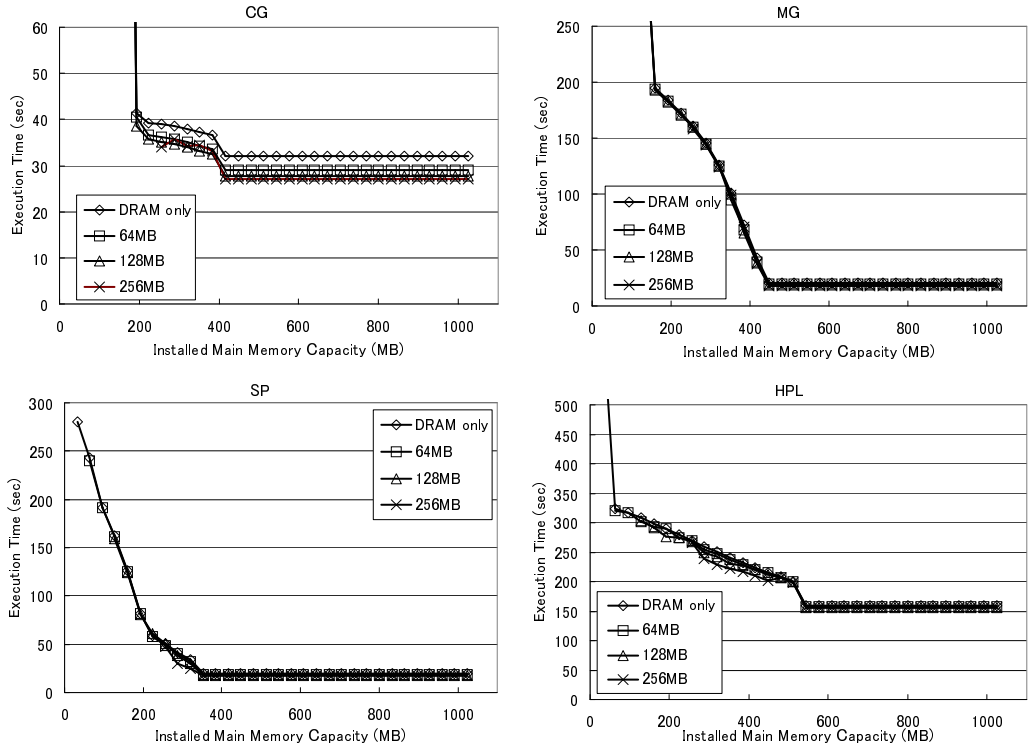

図 5 実行時間の推移

実行時間と合わせて評価すると，MRAM 128MB, DRAM 96MB の提案アーキテクチャと DRAM 1GB のアーキテクチャ上で CG を実行した結果を比較すると，性能低下を 12%に抑えつつ，消費エネルギーは 26%まで削減が可能になることがわかる．このようにメモリアクセスの局所性の高いアプリケーションでは本アーキテクチャが非常に有効であることがわかった．一方，メモリアクセスの局所性の低いアプリケーションではスワップが発生しない程度の DRAM 容量の削減は消費エネルギーの削減につながるものの，スワップの発生と同時に急激な実行時間の増加に伴い消費エネルギーも大きく増加してしまう．このような違いがあるため，アプリケーションの特性に応じて割り当てるメモリ量を決めるべきと考えられ，これは今後の課題の一つである．

本アーキテクチャにおいて，より大容量の MRAM を搭載することで性能は向上するが，先に述べたように MRAM の導入コストは大きいため，その搭載容量は制限される．CG のグラフによると，MRAM を 0 から 64MB に変化させたときのエネルギー削減は大きく，この場合少量の MRAM でも我々の目的上効果的であると言える．

また，今回の評価ではメモリチップの消費エネルギーしか考慮していないが，実際は他のコンポーネントを合わせて評価する必要がある．消費エネルギーの大きな部分を占める CPU は，実行時間の増加により静的消費エネルギーは増加する一方，メモリアクセス遅延の増加によりワークロードは低下するため，動的消費エネルギーは減少すると考えられる．

5. 関 連 研 究

メモリモジュールの省電力化の研究はすでに複数のアプローチからなされている．メモリ階層の上位階層のキャッシュ効率を上げることは下位階層へのアクセスを削減し動的消費エネルギーを減らすと同時に，実行時間を短縮できるため静的消費エネルギーも削減できる．近藤らはソフトウェア制御可能なオンチップメモリを用いることで，キャッシュヒット率を向上させている[14]．Hung らや Lebeck らは DRAM チップが休眠状態や低電力状態を持つことを仮定し，実行時にメモリの状態を変化させることで省電力化を実現している[15)16)]．状態の切り替えはページ単位でなくチップ単位で行われるため，多くのチップを休眠させることが可能なページング方式が提案されている．Cai らも同様にアクティブなメモリサイズを調節するが，HDD 休眠までのタイムアウト時間を調節することにより，メモリと HDD の合計消費電力を最小にすることを目的としている[17]．しかしこれらの研究では近藤らを除き，デスクトップアプリケーションやサーバを用いて

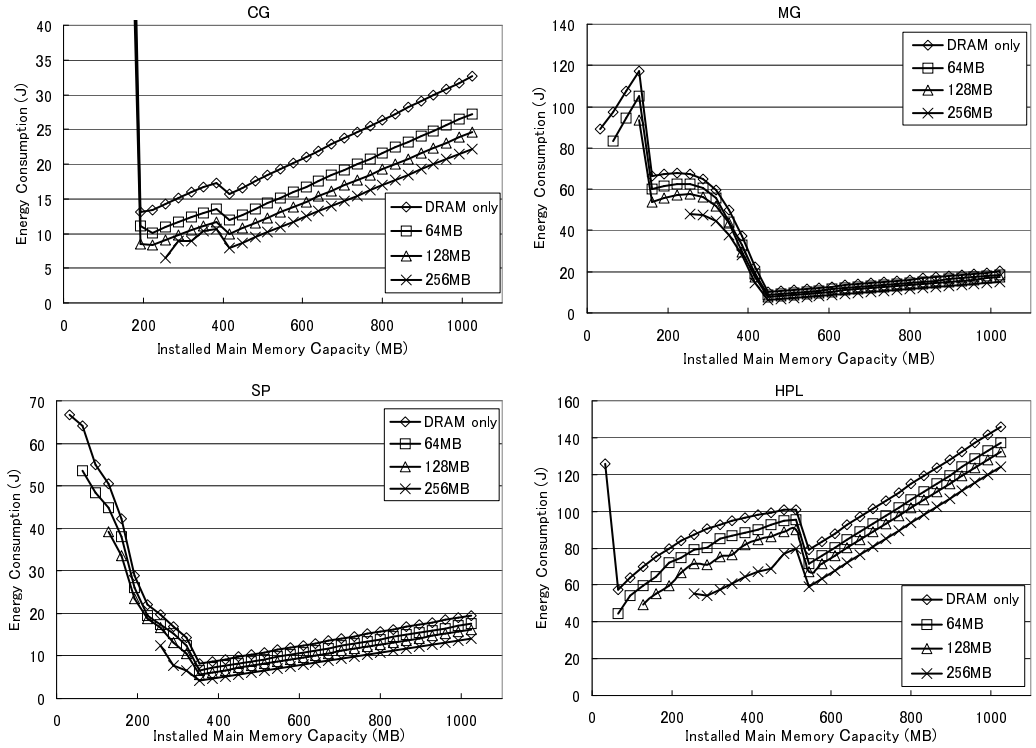

図 6 消費エネルギーの推移

評価しており，HPC における有効性は示されていない．また，これらの方式自体は大容量メインメモリの導入コストを軽減するものではない．

HDD の代わりにリモートマシンのメモリをスワップデバイスとして利用する研究は長く研究されている[18]．近年の高速なネットワークを用いて，InfiniBand 上[19]や 10GbE 上[8] における評価が報告されており，HDD よりも高性能なスワップ処理が実現されている．しかしこの手法はローカルマシンとリモートマシンの合計 DRAM 容量を削減するものではなく，また消費電力の評価も行われていない．とはいえ，複数アプリケーションを動作させる大規模システムにおいては，マシン間のメモリ使用量の平滑化が可能と考えられる．そのため，大規模システムの合計メモリ容量を削減するために，リモートスワップと本研究で用いた FLASH へのスワップの併用が有望だと考えられる．

上述した研究ではメインメモリは均一だが，組み込みシステムにおいては，消費電力や実装面積の制限のために SRAM と DRAM などの異種メモリが階層型ではなく並列に配置される場合がある．この場合，各データのメモリ上の配置が性能に大きく影響を与えるため，Avissar らはコンパイル時に最もメモリアクセス遅延が小さくなる配置を決定している[20]．しかし彼らの方式では，HPC において頻繁に用いられる大規模配列や動的に確保される領域への適用が困難である．一方本研究では，動的に MRAM と DRAM への配置を行うページング方式を提案し，HPC アプリケーションを用いて評価した．

6. ま と め

本論文では，次世代の省電力メモリを用いた低電力メモリアーキテクチャを提案した．我々のアーキテクチャは電力コストの大きい DRAM の搭載容量を削減するために，メインメモリの一部を不揮発性メモリ MRAM で置き換え，またスワップデバイスとして大容量ストレージとして期待される FLASH を使用している．そして異種のメモリを効率的に活用するページング方式を提案した．本アーキテクチャの性能を評価するため，アプリケーションの実行時間とメモリモジュールの消費エネルギーを見積もる性能モデルを構築した．本アーキテクチャを，複数の HPC アプリケーションベンチマークの挙動をシミュレートすることによって評価した．評価の結果，MRAM を 128MB 搭載した提案アーキテクチャ上で DRAM を 96MB まで削減することで，DRAM を 1GB 搭載したアーキテクチャと比較して，CG の性能低下を 12%に抑え，メモリチップの消費エネルギーを 26%に削減できることを示した．さらにメモリアクセスの局所性の高いアプリケーションについては，スワップを起こしてでも

DRAM 容量を削減するほうが消費エネルギーを低減可能な場合があることを示した．

今後の改良点としては，現在メモリモジュールに限られている性能モデルを CPU 等他のコンポーネントを考慮したものにすること，LRU ベース以外のページング方式の検討などがあげられる．また，本研究では MRAM が DRAM よりも速度，電力ともに優れていることを仮定したが，今後の技術動向によっては，メモリ階層の再検討や他の不揮発メモリの導入の検討を行っていきたい．

本研究の成果を，将来の省電力大規模 HPC システムの設計，運用に活用することを目標としているが，そのためには以下の課題が挙げられる．本研究の実験で見つかったような，アプリケーションの消費エネルギーを極小にするようなメモリ量を動的に見つける手法が必要である．また複数アプリケーションの合計エネルギーを低減するための性能モデルの構築やジョブスケジューラとの連携，またリモートスワップ技術やメモリチップの動的な休眠技術との連携を検討していきたい．

謝　　辞

本研究の一部は JST-CREST「ULP-HPC: 次世代テクノロジのモデル化・最適化による超低消費電力ハイパフォーマンスコンピューティング」および Microsoft Technical Computing Initiative の援助による．

参 考 文 献

1) Nandini Kappiah, Vincent W.Freeh, and David K.Lowenthal. Just in time dynamic voltage scaling: Exploiting inter-node slack to save energy in mpi programs. In *ACM/IEEE SC 2005 Conference (SC'05)*, p. 33, 2005.
2) Charles Lefurgy, Larthick Rajamani, Freeman Rawson, Wes Felter, Michael Kistler, and Tom W. Keller. Energy management for commercial servers. *Computer*, Vol. 36, No. 12, pp. 39–48, 2003.
3) TSUBAME Grid Cluster, Tokyo Institute of Technology. `http://www.gsic.titech.ac.jp/~ccwww/`.
4) Fusion-io. `http://www.fusionio.com/`.
5) Microsoft Corporation. Windows pc accelerators. `http://www.microsoft.com/japan/whdc/system/sysperf/perfaccel.mspx`, November 2006.
6) Saied Tehrani, Jon M. Slaghter, Mark Deherrera, Brad N. Engel, Nicholas D. Rizzo, John Salter, Mark Durlam, Renu W. Dave, Jason Janesky, Brian Butcher, and Greg Grynkewich Ken Smith. Magnetoresistive random access memory using magnetic tunnel junctions. In *Proceedings of IEEE*, Vol. 91, pp. 703–714, 2003.
7) Freescale. `http://www.freescale.com/`.
8) 後藤正徳, 佐藤充, 中島耕太, 久門耕一. 10gb ethernet 上の rdma を用いた遠隔スワップメモリの実装. 電子情報通信学会技術研究報告 CPSY, Vol. 106, No. 287, pp. 7–12, 2006.
9) Valgrind. `http://valgrind.org/`.
10) Samsung. `http://www.samsung.com/`.
11) NEC. `http://www.nec.com/`.
12) Nas parallel benchmarks. `http://www.nas.gov/software/NPB`.
13) Antoine Petitet, R. Clint Whaley, Jack Dongarra, and Andrew Clearyb. Hpl - a portable implementation of the high-performance linpack benchmark for distributed-memory computers. `http://www.netlib.org/benchmark/hpl`.
14) Masaaki Kondo, Shinichi Tanaka, Motonobu Fujita, and Hiroshi Nakamura. Reducing memory system energy in data intensive computations by software-controlled on-chip memory. In *Int'l Workshop on Compilers and Operating Systems for Low Power*, 2002.
15) Hai Hung, Padmanabhan Pillai, and Shin Kang G. Design and implementation of power-aware virtual memory. In *USENIX 2003 Annual Technical Conference*, pp. 57–70, 2003.
16) Alvin R. Lebeck, Xiaobo Fan, Heng Zeng, and Carla Ellis. Power aware page allocation. In *ASPLOS-IX: Proceedings of the ninth international conference on Architectureal support for programming languages and operating systems*, pp. 105–116, 2000.
17) LeCai and Yung-Hsiang Lu. Joint power management of memory and disk. In *Proceedings of the conference on Design, Automation and Test in Europe (DATE'05)*, Vol. 1, pp. 86–91, 2005.
18) Tia Newhall, Sean Finney, Kuzman Ganchev, and Michael Spiegel. A network swapping module for linux cluster. In *Proceedings of Euro-Par '03 International Conference on Parallel and Distributed Computing*, pp. 1160–1169, 2003.
19) Shuang Liang, Ranjit Noronha, and Dhabaleswar K. Panda. Swapping to remote memory over infiniband: An approach using a high performance network block device. In *Proceedings of the IEEE Cluster Computing (Cluster2005)*, 2005.
20) Oren Avissar, Rajeer Barua, and Dave Strwart. Heterogeneous memory management for embedded system. In *International Conference on Compilers, Architecture and Synthesis for Embedded Systems(CASES)*, pp. 34 – 43, 2001.