2003－ARC－154　(24)
2003／8／6

プリフェッチ機能を有するメモリモジュール

田邊　昇[†1]　土肥康孝[†2]
中條拓伯[†3]　天野英晴[†4]

PC のメモリスロットに装着されるプリフェッチ機能を有するメモリモジュールを提案する．このデバイスは，Pentium4 などの COTS(Comercial Off-The-Shelf) 型 MPU のキャッシュアーキテクチャの弱点を軽減することで，パーソナルスーパーコンピュータに匹敵する性能を PC 上でも実現可能にすることを目指している．本報告では，プリフェッチ機構付メモリモジュールの基本コンセプトとそのソフト的対応法について述べ，DIMMnet-2 における実装案を紹介する．4 つのアプリケーションについての予備検討を通し，高速化が得られる見通しを示す．

A Memory Module with Prefetching Functions

Noboru Tanabe,[†1] Yasunori Dohi,[†2] Hironori Nakajo[†3] and Hideharu Amano [†4]

A memory module with prefetching functions plugged into a memory slot of a PC is proposed. This device mitigates the weak points of cache architecture. In this way, it will realize personal supercomputer class performance with COTS (Comercial Off-The-Shelf) type MPU, such as Pentium4. In this report, the concept and its software corresponding method of a memory module with prefetching functions, and introduction of its implementation plan on DIMMnet-2 network interface are presented. In addition, the prospects for acceleration with the preliminary examination about four applications are shown.

1. はじめに

半導体技術やアーキテクチャの進歩を背景にしたマイクロプロセッサ (MPU) の目覚しい進歩を遂げている．例えば Pentium4 などはスーパーコンピュータを凌駕する周波数で，スーパーコンピュータの 1 CPU に匹敵する演算性能をオフィスや一般家庭に提供している．量産効果に下支えされ，COTS(Comercial Off-The-Shelf) である MPU およびパーソナルコンピュータ (PC) の性能向上およびコストパフォーマンスの向上は目覚しいものがある．

これらの COTS 部品である MPU はほぼ例外なくキャッシュアーキテクチャに基づいている．キャッシュアーキテクチャは主記憶の脆弱さを隠蔽できることが多いため，低コストな PC における主記憶は，ベクトル型スーパーコンピュータのそれとは異なり，キャッシュが効かないアプリケーションに対しては演算能力にバランスしたものにはなっていない．

科学技術計算の分野では，SX-6i[1)] のようにベクトル型スーパーコンピュータの 1 CPU 分を切り出した PC と同程度のサイズのパーソナルスーパーコンピュータ製品上で，PC との性能差が数十倍に及ぶアプリケーションが多数存在することが報告されている．このようなアプリケーションにおいてはベクトル型スーパーコンピュータは依然として存在意義がある．

一方，DB2 の父と称される IBM フェローの Linsay 氏は，コンピュータ上の主記憶が増加することに伴って DBMS の重要な部分が主記憶に常駐するようになり，キャッシュミスの最小化の重要性が高まることを指摘 [2)] しており，キャッシュが効かないアプリケーションは科学技術計算にとどまるものではない．この点は将来，MRAM のような不揮発性大容量高速メモリが DRAM に置き換わるような状況になった場合，この分野におけるキャッシュミスの最小化の重要性は加速する．

キャッシュベースの MPU は記憶階層上で主記憶直前にあるキャッシュのラインサイズは伸びる傾向にある．例えば Pentium3 では L2 キャッシュの 32 バイトであるのに対し，Pentium4 では L2 キャッシュの 128 バイトである．キャッシュベースの MPU は常に上記のサイズで主記憶をアクセスする．

ところが，例えばリレーショナルデータベース処理のようにストライドの大きい不連続アクセスが必要なアプリケーションでは，真に必要なデータが 128 バイトのキャッシュラインの中に 1 バイトしか存在しないというケースもある．このように，キャッシュが存在すること自体がかえって性能低下の原因になることがある．

しかし，COTS である PC と COTS ではないパーソナルスーパーコンピュータの間には 100 倍程度の価格差があり，市場に劇的な変化がない限り後者を手軽に購入できる状態にはなり難い．よって，価格と性能の両面において，これらの両極端を埋めるコンピュータシステムの登場が望まれる．

さらに，COTS ではないパーソナルスーパーコンピュータの製品サイクルと，COTS である PC の製品サイクルには歴然とした差があり，およそ 1,2 年で少なくとも演算性能だけで比較すれば PC は陳腐になってきたパーソナルスーパーコンピュータと遜色ないレベルまで進歩すると思われる．

以上のような観点から，キャッシュが効かないアプリケーショ

†1 (株) 東芝, 研究開発センター
Corporate Research and Developpment Center, Toshiba
†2 横浜国立大学
Yokohama National University
†3 東京農工大学
Tokyo University of Agriculture and Technology
†4 慶應義塾大学
Keio University

ンに対するアーキテクチャ面からの対応としては，COTS の目覚しい性能向上を容易に取り込めるものでありつつ，キャッシュアーキテクチャの欠点を補う方式の研究開発が望ましい．

本研究は以上のような背景を鑑み，PC のメモリスロットに装着されるプリフェッチ機能を有するメモリモジュールを提案する．このデバイスは，Pentium4 などの COTS 型 MPU のキャッシュアーキテクチャの弱点を軽減することで，パーソナルスーパーコンピュータに匹敵する性能を PC 上でも実現可能にすることを目指している．**表 1** に本研究の目標とする性能と価格のイメージを示す．

表 1 目標とする性能と価格のイメージ

	パーソナルスパコン	目標	PC
キャッシュが効くアプリ性能	2	1	1
キャッシュが効かないアプリ性能	20	10	1
価格	200	2	1

本研究のプロトタイプ検証は DIMM スロットに装着されるネットワークインタフェース DIMMnet-2[3] 上で行うことが予定されている．本方式を DIMMnet-2 上に適用することで，実質的に高速化された安価なノード PC が高速なネットワークで結合され，全体として巨大かつ高速アクセス可能な共有メモリを有する並列型スーパーコンピュータに見える PC クラスタが構築されることを目指している．

本報告では，2 章ではプリフェッチ機構付メモリモジュールの基本コンセプトとそのソフト的対応法について述べる．3 章ではベクトル命令で改善されるアクセスパターンについて述べ，4 章ではその DIMMnet-2 における実装案を紹介する．5 章では 4 種類のアプリケーションに基づく予備検討を行い，6 章で関連研究について述べ，7 章でまとめる．

2. プリフェッチ機構付メモリモジュール

2.1 基本コンセプト

メモリ空間にマップされたメモリモジュール側にあるバッファ(プリフェッチバッファ) へのプリフェッチコマンドを発行し，ホスト CPU から利用確率が高い状態に整えられたデータ群に対してブロックアクセスを行う．

その結果，キャッシュ・TLB・FSB・メモリバスの利用効率が向上する．メモリモジュールは着脱可能なので，CPU やチップセットを改造することなく，高性能かつ低下価格な COTS を HPC 向けコンピュータとして有効に活用できる．

なお，プリフェッチコマンドには種々の実装法がありうるが，本研究ではベクトル転送命令について検討する．

2.2 ソフトウェア的対応

本機構は，キャッシュメモリ等とは異なり，ソフト的に透過なハードではない．利用するためにはプログラムの変更またはコンパイラによるケアが必要である．

本機構を Pentium4 などのキャッシュベースの CPU から利用する際のリード時における指針を以下に示す．

(1) 主記憶の WriteBack 属性の領域にスクラッチパッド領域 A を確保する．

(2) 本機構を利用するためのプリフェッチコマンドを実際にデータを利用するより前に発行する．

(3) プリフェッチバッファから領域 A にブロックコピーを行うコードをプリフェッチが完了する頃に実行する．

(4) プリフェッチバッファのアドレスに対応するキャッシュラインを再利用前にフラッシュする命令を実行する．

(5) 領域 A(実体はキャッシュ上) にあるデータを用いて，所望の処理を行う．プリフェッチと処理の時間差が少なければキャッシュへのアクセスだけで必要なデータが得られる．

なお，SCIMA[5)6)] のように，メモリ空間上にマップされた高速アクセスが可能なオンチップメモリを有する CPU においても，本機構を使うことは可能であり，その場合は上記のスクラッチパッド領域 A およびその実体がオンチップメモリに相当し，キャッシュではないため，キャッシュラインのフラッシュは不要になるので，よりシンプルになる．

一方，ライト時における高速化の指針は DIMMnet-1 における BOTF における Window メモリへの CPU からの書き込みの高速化指針 [4] とほぼ同様であり，Window メモリにコピーし終えたら，送信を指示するキック操作の代わりにベクトルストアコマンドを起動する．

3. ベクトル命令で改善されるアクセスパターン

3.1 連続アクセス

キャッシュアーキテクチャでもタイリングのアルゴリズムを適用するなど技巧を要するが，工夫することで，連続アクセスについては対応できることが多いと考えられる．

本方式ではベクトル連続アクセス命令をサポートする．アドレス計算の負担がメモリモジュール側にオフロードされる分，本方式はさらにアプリケーションが高速化する可能性がある．

3.2 等間隔アクセス

キャッシュアーキテクチャでは等間隔アクセス時にはキャッシュラインの利用効率が低下する．とりわけ，少ないバイト数からなるデータ型で間隔値 (ストライド) がキャッシュラインサイズを超える場合は最悪の状況になり，殆どの時間が主記憶との間の無駄なデータの転送に費やされてしまう．

これに対し，等間隔ベクトルアクセス命令によれば，アドレス計算の負担がメモリモジュール側にオフロードされるのみならず，ベクトルレジスタ上に必要なデータのみ圧縮格納されるので，それを CPU から連続アクセスとして扱うことができ，キャッシュラインやメモリバスや FSB の効率的な利用が可能になる．

3.3 リストアクセス

キャッシュアーキテクチャではリストアクセス時にもキャッシュラインの利用効率が低下する．キャッシュライン内の有効データ比率の問題は等間隔アクセスの場合と同様だが，さらにリスト（ポインタまたはインデックス) のためにもキャッシュやバスが消費されるという問題もあるため，さらに性能低下を引き起こす．

これに対し，リストベクトルアクセス命令によれば，アドレス計算の負担がメモリモジュール側にオフロードされる点，ベクトルレジスタ上に必要なデータのみ圧縮格納される点のみならず，リスト（ポインタまたはインデックス) のための CPU 側の資源を消費することが無くなる点で性能向上が期待できる．

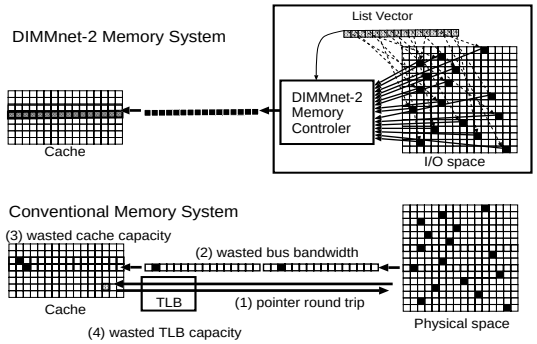

図 1 リストアクセスにおけるキャッシュ挙動の違い

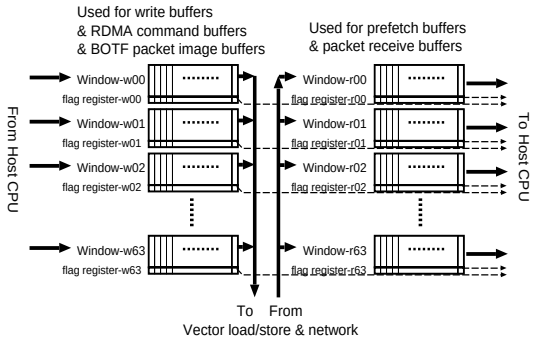

図 2 プリフェッチ機能付き Window メモリ

4. DIMMnet-2 における実装

4.1 資源モデル

1 個の DIMMnet-2 上のプログラマから見える資源を分類すると以下のようになる．

(1)SO-DIMM 領域 (4GB)
(2)LLCM 領域 (数百 KB)
(3)WW : Write Window(64bit 幅 64 ワード× N 本)
(4)PW : Prefetch Window(64bit 幅 64 ワード× N 本)
(5)SR : Scalar Register(7bit 幅 8 ワード)
(6)WWSR : Write Window Status Register(64bit 幅 N ワード)
(7)PWSR : Prefetch Window Status Register(64bit 幅 N ワード)
(8)CMR : Command Register(64bit 幅 16 ワード)
(9)CMSR : Command Status Register(16bit 幅 1 ワード)
(10)CMSR : Command Count Register(8bit 幅 16 ワード)
(11)CPR : Current Page Register(16bit 幅 4 ワード)
(12)CPMR : Current Page Mask Register(16bit 幅 4 ワード:カーネル領域にマップ)
(13) その他制御レジスタ

(1) は CMR に書き込まれたコマンドにより WW または PW を介してアクセスする．つまり，ホストのメモリ空間には基本的にはマップされておらず，DIMM スロットに割り当てられた最大物理メモリ容量の壁を越える一種の I/O 空間 (大域仮想アドレス空間) に存在する．大域仮想アドレス空間は rank(14bit),page(4bit),offset(32bit) の合計 50bit で指定されるノード間にまたがる共有空間である．この領域は大域仮想アドレス空間にマップされているので，後述の転送命令でリモートノードからもアクセスが可能である．ただし，各ユーザは CPMR により許可された page にしかアクセスできない．

(3)(4)(6)(7) は DIMMnet-2 における書き込み用と読み出し用のバッファとして考案された図に示されるプリフェッチ機能付き Window メモリの構成要素であり，これらがベクトルレジスタとして用いられる．

(2)〜(13) はホストの DIMM スロットに割り当てられた物理アドレス空間にマップされ，通常のメモリアクセスによりアクセス可能とする．さらに，これらもプロテクション上問題のないものは大域仮想アドレス空間にマップされているので，後述の転送命令でアクセスが可能である．つまり PW もリモートストアの書き込み対象となる．これは，超並列計算機 TS/1 におけるリモートのベクトルレジスタ間のチェイニングを行うプロセッサ間チェイニング [7] の変形である．

4.2 ベクトル転送命令

DIMMnet-2 におけるベクトル転送命令の概要を以下に示す．**図 3** に等間隔ロード・ストア命令，**図 4** に間接ロード・ストア命令、**図 5** に移動・複製命令の動作を示す．命令は 64bit 形式であり，ホストから書き込まれる．Packed と通常の間接アクセスの違いは Packed は命令中で指定する 1〜4 個の要素に対する命令であるが，通常の方は Window 上のリストベクトルで要素を指定する点である．移動と複製の違いは，ステータスレジスタのフラグがクリアされるか否かの違いである．VT 命令のみ直後に書き込まれた命令と組み合わせてリモートアクセスを行う命令と解釈される．

以下，XW は WW と PW の総称，type はデータ型，iteration は繰り返し回数，displace は XW 上での開始位置，top は大域仮想空間上の offset(8byte 境界)，stride は要素間間隔 (byte 数)，indexXW は index ベクトルを保持する window である．

(1) ベクトル連続ロード:
VL(type,iteration,displace,PWi,top)
(2) ベクトル等間隔ロード:
VLS(type,iteration,displace,PWi,top,stride)
(3) ベクトル間接ロード:
VLI(type,iteration,displace,PWi,top,indexXWj)
(4)Packed ベクトル間接ロード:
VLPI(type,iteration,displace,PWi,top,index1,index2,index3)
(5) ベクトル連続ストア:
VS(type,iteration,displace,PWi,top)
(6) ベクトル等間隔ストア:
VSS(type,iteration,displace,PWi,top,stride)
(7) ベクトル間接ストア:
VSI(type,iteration,displace,PWi,top,indexXWj)
(8)Packed ベクトル間接ストア:
VSPI(type,iteration,displace,PWi,top,index1,index2,index3)
(9) ベクトル移動:
VM(type,iteration,displace,sourceXWi,destinationXWj)
(10) ベクトル複製:
VC(type,iteration,displace,sourceXWi,destinationXWj)

(11) ベクトル転送拡張:
VT(rank,page)

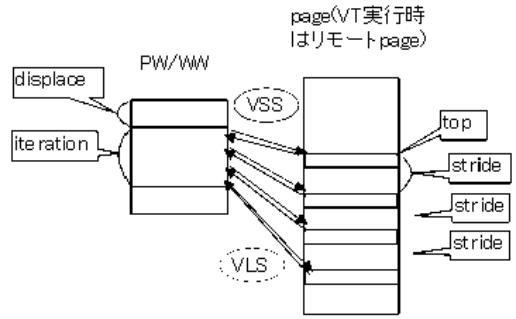

図 3 等間隔ロード・ストア命令

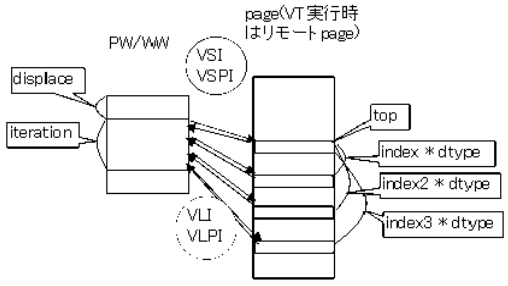

図 4 間接ロード・ストア命令

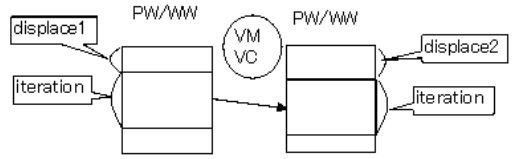

図 5 移動・複写命令

5. アプリケーションに基づく予備検討

5.1 リレーショナルデータベース

リレーショナルデータベースの処理は実行プラットフォーム上での主記憶容量の増加とともに主要な部分が主記憶に常駐するようになるため，今後はキャッシュのミスを最小化することが重要になってくる．

ハードウェアによる等間隔アクセスの連続アクセス化のサポートが主記憶データベースの問い合わせ処理に対して有効であることが，北陸先端大の SDT(Stride Data Transfer) の研究によって明らかになっている．SDT はホスト MPU 内の FIFO をベクトルレジスタと見なすことで一種のベクトルロード命令と解釈することができる．このため，SDT は本研究の方式と類似しているため，本方式においても同様に効果が期待できる．

さらに，北陸先端大はポインタが CPU に戻りキャッシュを汚すことをメモリコントローラ側で抑制する TPDT(Two-Phase-Data-Transfer) というスカラ型のリストアクセス機構が，内容比較を伴う主記憶データベースの問い合わせ処理に対して有効であることを示している．

一方，本方式ではリストベクトルは主記憶からベクトルレジスタまでしか来ないために，TPDT 同様にホスト CPU 上のキャッシュを汚すことはなく，基本的には同様の効果が期待できる．

市場規模から IA32 アーキテクチャに基づく MPU が現時点での最も COTS のメリットを享受できる MPU であり，本研究では，COTS のメリットを最大限に生かすという基本方針から，IA32 アーキテクチャに基づく COTS 型 MPU の使用を仮定する．

北陸先端大の研究は SPARC アーキテクチャによる評価であり，COTS 型 MPU ではなくバッファとして特殊な MPU 上のキャッシュの一部を FIFO 化したものを用いており，我々のターゲットとはいくつかの実装上の違いがある．その違いによる性能差の検証については今後の課題である．

5.2 NAS CG

NAS CG は NASA 提供のハイパフォーマンスコンピュータ評価用ベンチマークの一つで，共役勾配法による疎行列の最小固有値を求めるプログラムである．

NAS CG(シリアル版) のカーネル部分は以下の通りであり，殆どの処理時間がこの部分で消費されるため，この部分の高速化が重要である．その部分にはリストベクトル colidx(k) による間接参照がある．

```
1: do j=1, n
2:     do k=rowstr(j),rowstr(j+1)-1
4:     sum = sum + a(k)*p(colidx(k))
5:     enddo
6:     q(j) = sum
7: enddo
```

CG では係数行列ａ（ｋ）および解候補配列 p（k）のデータ型は倍精度浮動小数 (8 バイト) である．よって Pentium4 上では行方向の非零要素の間隔が 16 を超えるような疎らな係数行列の場合，p（k）に対するキャッシュライン上の有効なデータは 128/8 = 16 個に 1 個しか含まれず，キャッシュアーキテクチャであるがゆえに 16 倍の 16 × 8byte/2flop 実効バンド幅を要求する．さらに係数行列 a(k) のためのメモリバンド幅が 8byte/2flop とリストベクトル colidx(k) のためのメモリバンド幅が 4byte/flop がさらに必要になる．これらは基本的には連続アクセスとなるため有効データが少ないという問題はない．

一方，リストベクトルアクセス命令により圧縮がかかる本方式の場合，連続領域へのアクセスと等化になり，リストベクトル colidx(k) は CPU には転送されないので 8byte/2flop のメモリバンド幅で済む．

以上をまとめるとキャッシュサイズから溢れる大きさのデータサイズを有する ClassB 以上の CG ベンチマークでは

Pentium4 のみの場合：64+4+2 = 70 バイト/flop

DIMMnet-2 付加の場合：4+4+2 = 10 バイト/flop

の実効メモリバンド幅が必要である．DDR400 ベースの DIMM スロットに搭載された DIMMnet-2 を想定すると 1 個あたり 3.2GB/s のバンド幅が割り当てられるので DIMMnet-2 を 1 個装着の場合およそ 320MFLOPS 程度，2 個装着の場

合 640MFLOPS 程度の実効性能が期待できる．これに対し，Pentium4 のみの場合は DDR400 シングルチャネルの PC 上で 45MFLOPS，デュアルチャネルの PC 上でも 90MFLOPS 程度しか期待できない．

以上のように，CG の高速化についてはメモリシステムの改良はクラスタによる並列化よりも低コストで高い効果が得られることが予想される．

上記の荒い見積もりは，DIMMnet-2 上のメモリ (複数チャネルの DDR SO-DIMM) がリストベクトルアクセスに対してシングルチャネル分のデータをベクトルレジスタに対して供給できることを仮定しているので，その性能をキープできるようなメモリ制御部を設計することが今後の課題である．

5.3 回路シミュレーション

Spice 等の回路シミュレーションは回路行列の本質的なランダムスパース構造と反復法による収束性の悪さから，フィルイン (後天的な非零要素) 発生を抑えるオーダリングを行った行列に対する LU 分解による解法がとられることが一般的である．

その代表である Spice は，行列の構造決定後にその LU 分解の過程をループフリーのスカラ命令列に翻訳して実行するが，大規模な行列では命令キャッシュから溢れるため，命令コードフェッチのためにメモリバンド幅を消費する．キャッシュアーキテクチャではさらにランダムな要素へのアクセスになるため上記の CG の場合と同様に 16 倍に割り増しされたメモリバンド幅を消費することになり，メモリバスネックによって演算性能が上がらない．

一方，LU 分解に要する基本的な演算は対角要素 (ピボット) による同一行に属する上三角行列の非零要素に対する除算と，その行による下三角行列の更新演算 (乗算と加算の組み合わせ) の二種類のみであり，並列ピボッティング法 (PPM)[10)11)]、MVA[12)] などの非零要素の配置に応じたデータ依存関係の解析によって並列処理できる同様の演算のグループに分類することで，ベクトル処理 (または並列処理) が可能になる．その場合，係数行列は一次元配列に圧縮して格納して，リストベクトルによってアクセスすることもできる．

上記のリストベクトルアクセスについて，本方式ではリストベクトルアクセス命令によってホスト CPU に対して連続データとしてデータを供給できる．こうしてメモリバスは効率的に使用されることになり，単純なキャッシュに頼る方式よりも一桁程度の性能向上が期待できる．

なお，回路シミュレーションにおけるリストベクトルの作成は行列の非零要素の配置が固定であるために，最初に 1 度だけですむ静的な問題構造である．このため，従来からスーパーコンピュータで用いられているリストベクトルアクセス機構で対応が可能と考えられる．

5.4 ボリュームレンダリング

ボリュームレンダリングは MRI の診断結果の可視化などのために高速化が要望されるアプリケーションの一つである．こちらは三次元配列で表現される物体 (ボリュームデータ) に対する視線の変動によりスクリーンに投影される二次元画像を生成することになる．

このため，上記の回路シミュレーションとは異なり，視線と交差するボリュームデータのインデックスは視線の移動に伴って動的に生成する必要がある．

これをキャッシュアーキテクチャ上でナイーブに実行すると，視線の位置によっては最悪ケースではキャッシュライン 128 バイト中 1 バイトしか使われないという状態に陥り，視線の位置によってはメモリバスネックとなる．

一方，本方式では，ボリュームデータへのアクセスをリストベクトルアクセスにより，ホスト CPU からは連続アクセスになるように変換することで，視線の位置によらず安定したメモリバンド幅消費を行うようになり，スムースな映像化を実現することが可能と考えられる．

ただし，視線が常時動いているとリストベクトルの再利用が行われないので，ホスト CPU 上で生成されたインデックスを随時ベクトルアクセス機構側に渡しつつ，ベクトルレジスタに圧縮ロードが完了したボリュームデータの連続読み出しを並行して行えるようになることが望ましい．

DIMMnet-2 で予定されている Packed リストベクトルアクセス命令は，このような動的なリスト構造へのアクセスにおいて，並行動作を促進して少ないオーバーヘッドで実現することを意図して考案されており，そのアプリケーションによる性能評価は今後の課題である．

6. 関 連 研 究

Impulse[13)14)] はエイリアス上の連続アクセスを等間隔ベクトルアクセスや間接ベクトルアクセスに変換する点で機能が類似している．しかし，ホスト CPU 上の仮想空間上に設けられる実体と同サイズのエイリアスへのアクセスとなるために，メモリコントローラ側でのアドレス変換が多段階でオーバーヘッドが増加し，かつ 32bit CPU ではその空間の狭さに縛られる．さらに，ホスト CPU の FSB(Front Side Bus) に接続されるためホスト側のメモリコントローラを変更しない限り実装できない．このため，COTS のマザーボードを利用することはできない．また PC クラスタに拡張できる広大なサイズの大域的仮想空間の概念を持たない．

SDT[8)9)] はホストからのコマンドによって起動される等間隔ロードを高速化するためのメモリコントローラ上の機構である．本方式の等間隔ベクトルロード命令と類似しているが，ストアや種々のデータ型には対応していない．さらに COTS のマザーボードを利用することはできない点や大域的仮想空間の概念を持たない点は Impulse と同様である．

TPDT[8)9)] はポインタによるスカラ型の間接参照ロードを高速化するためのメモリコントローラ上の機構である．既にできあがった間接参照インデックスまたはポインタ配列がメモリ上に存在する場合はインデックスやポインタが CPU とメモリの間を動くことになるので，本方式のベクトル間接ロード命令と比べると性能が低くなると思われる．また本方式の Packed 間接ロード命令と目的が類似するが，TPDT は 1 個が対象であるのに対し，本方式は 4 個までの間接参照を 1 回のリクエストでこなすことができる点で機能や期待できる性能が高い．さらに COTS のマザーボードを利用することはできない点や大域的仮想空間の概念を持たない点は Impulse と同様である．

SCIMA[5)6)] は等間隔アクセスを命令でサポートする CPU だが，それは独立したメモリコントローラではなくオンチッ

プメモリへのプリフェッチを行う CPU の命令として提案されている．この命令はリストのような等間隔ではない不連続アクセスに対応していない．本方式と組み合わせることでラインのフラッシュ操作が無い，よりシンプルなソフト的対応で同様の効果が期待できる．COTS のマザーボードを利用することはできない点や大域的仮想空間の概念を持たない点は Impulse 等と同様であり，さらに COTS の CPU も利用できない．

7. ま と め

本報告では PC のメモリスロットに装着されるプリフェッチ機能を有するメモリモジュールを提案した．このデバイスは，Pentium4 などの COTS 型 MPU のキャッシュアーキテクチャの弱点を軽減することで，パーソナルスーパーコンピュータに匹敵する性能を PC 上でも実現可能にすることを目指している．

本報告ではプリフェッチ機能を有するメモリモジュールの基本コンセプトを示した．さらに DIMM スロットに装着されるネットワークインタフェース DIMMnet-2 におけるベクトル転送命令による本方式の実装例について示した．

この実装例では，プロテクションのためのオーバーヘッドを減らしつつ，ローカルとリモートの差を極力排除した統一的な命令により高速アクセス可能な，最も COTS の恩恵を享受できる 32bit CPU のアドレス空間の不足を超越した巨大な共有メモリ構築のための一つの方法論が示されている．

さらに，リレーショナルデータベース・NAS CG・回路シミュレーション・ボリュームレンダリングの 4 つのアプリケーションについての予備検討を行い，本方式により Pentium4 等の COTS 型 MPU を単純に用いる場合と比べ，メモリバンド幅の有効利用がはかられ，高速化が得られる見通しを得た．これらの詳細な評価は今後の課題である．

本方式はネットワークインタフェース上に必ずしも実装される必要は無く，1CPU のシステムにおいて有効である．つまり，並列処理とは別の原理により，本方式を内在する DIMMnet-2 は装着した安価なノード PC の実行性能を高速化する可能性を持っているという点で，従来にない画期的な特質を有するネットワークインタフェースであることが示された．これが，通信のみの高速化を実現する DIMMnet-1 との根本的な違いである．

今後は，DIMMnet-2 上の外部メモリの構成や，その他の部分の設計を進め，本方式の有効性をシミュレーションやハードウェアプロトタイプ作成により評価を行う予定である．

謝辞 本研究は総務省戦略的情報通信研究開発制度の一環として行われたものである．SCIMA についてご教授ご議論いただいた豊橋技術科学大学の中島教授，東京大学の中村助教授，藤田氏，キャッシュアーキテクチャ上でのプリフェッチに関する現状をご教授いただきました早稲田大学の笠原教授に感謝いたします．DIMMnet-2 の開発に関する議論にご参加いただいている和歌山大学の国枝教授，上原講師，齋藤助手，東京農工大学の並木助教授，浜田氏，三橋氏，荒木氏，木立氏，森氏，慶應義塾大学の西氏，渡辺氏，大塚氏，伊豆氏，北村氏，横浜国立大学の中武氏，箱崎氏に感謝いたします．

参 考 文 献

1) 萩原, 梅澤 : “パーソナルスーパーコンピュータ SX-6i”, 情報処理, Vol.44,No.3, pp.277-282 (Mar. 2003)
2) B.Lindsay : “DB2 の父ブルース・リンゼイ博士が語る、データベース・テクノロジーの現在と未来”, http://db2.jp/interview/special/lindsay.html
3) 田邊, 濱田, 中條, 天野 : “メモリスロット装着型ネットワークインタフェース DIMMnet-2 の構想”, 情報処理学会計算機アーキテクチャ研究会, 2003-ARC-152, pp.61-66 (Mar. 2003)
4) 田邊, 山本, 濱田, 中條, 工藤, 天野 : “DIMM スロット搭載型ネットワークインタフェース DIMMnet-1 とその高バンド幅通信機構 BOTF”, 情報処理学会論文誌, Vol.43, No.4, pp.866-878 (Apr. 2002)
5) 中村, 近藤, 大河原, 朴 : “ハイパフォーマンスコンピューティング向けアーキテクチャSCIMA”, 情報処理学会論文誌ハイパフォーマンスコンピューティングシステム, Vol.41 No.SIG5, pp.15-27 (Aug. 2000)
6) 近藤, 中村, 朴 : “SCIMA における性能最適化手法の検討”, 情報処理学会論文誌ハイパフォーマンスコンピューティングシステム, Vol.42 No.SIG12, pp.37-48 (Nov. 2001)
7) 田邊 : “超並列テラフロップスマシン TS/1 における並列処理〜プロセッサ間チェイニングとその応用〜”, 情報処理学会論文誌, Vol.36, No.3, pp.55-60 (Mar. 2003)
8) 府川, 田中, 宮崎 : “主記憶データベース向け高機能メモリコントローラの実現方式”, 情報処理学会計算機アーキテクチャ研究会, 2003-ARC-150, pp.77-82 (Nov. 2002)
9) 府川, 田中, 宮崎 : “主記憶データベース向け高機能メモリコントローラの性能評価”, 情報処理学会計算機アーキテクチャ研究会, 2003-ARC-152, pp.85-90 (Mar. 2003)
10) 田邊, 土肥 : “連想スイッチによる疎行列用計算機の構成”, 電子情報通信学会論文誌,Vol.J70-D,No.12,pp.2393-2401 (Dec. 1987)
11) 田邊, 石坂, 村越, 泉谷, 土肥 : “並列パイプライン式疎行列専用計算機 RAMP の構成”, 電子情報通信学会論文誌,Vol.J71-D,No.10,pp.1939-1948 (Oct. 1988)
12) Yamamoto, Takahashi : “Vectrized LU Decomposition Algorithms for Large-Scale Circuit Simulation”, IEEE Transaction on CAD, Vol. CAD-4, No.3, pp.232-239 (Mar. 1985)
13) Carter, Hsieh, Stoller, Swanson, Zhang, Brunvand, Davis, Kuo, Kuramkote, Parker, Schaelicke and Tateyama : “Impulse : Building a Smarter Memory Controller”, International Symposium on High Performance Computer Architecture (HPCA-5), pp.70-79 (Jan. 1999)
14) Mathew, McKee, Carter, Davis : “Design of a Parallel Vector Access Unit for SDRAM Memory Systems”, International Symposium on High Performance Computer Architecture (HPCA-6), pp.39-48 (Jan. 2000)