社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2008－ARC－177 (2)
2008－HPC－114 (2)
2008／3／5

DMAで主記憶をアクセスするCPUにおける不連続アクセスの連続化

太田 淳[†] 金 美善[†]
田邊 昇[††] 中條拓伯[†]

Cell Broadband Engine(Cell/B.E.) のようにキャッシュを持たない単純な CPU コアを多数内蔵するマルチコア CPU が注目されている．このような CPU コアではキャッシュの代わりに小容量のローカルメモリを持ち，主記憶とローカルメモリの間を DMA 転送によりデータ転送することでキャッシュの代用をさせる．上記のアーキテクチャによるチップ内演算能力の向上とは裏腹に，これに見合ったデータ供給能力の実現が実効性能の鍵を握る．本報告では，DIMMnet-2 と同様の連続化ハードウェアによる DMA で主記憶をアクセスする CPU への不連続アクセスの連続化の効果について，主記憶データベースに対する Wisconsin ベンチマークを用いた性能評価に基づいて論じる．

Conversion from Non-continuous Access to Continuous Access on a CPU with DMA-type Main Memory Accessing

Atsushi Ohta,[†] Bizen Kin,[†] Noboru Tanabe[††] and Hironori Nakajo [†]

A multicore CPU including multiple simple CPU's with no cache such as Cell Broadband Engine(Cell/B.E.) is currently attractive. Instead of cache, such CPU core has small sized local memory which plays a role of cache with data transferring between main memory and local memory with DMA. Contrary to growing performance in a chip by multicore architecture, a key technology of effective performance is realizing data supplying capacity corresponding to the performance. In this report, with hardware which has been also implemented in DIMMnet-2, effectiveness of sequencing discontinuous accesses to a CPU which accesses main memory with DMA is shown based on performance evaluation using a Wisconsin benchmark program for main memory database.

1. はじめに

消費電力や発熱量の増加の問題から，単一 CPU では以前ほどの性能向上を望めなくなった．そこで，各 CPU メーカは更なる性能向上への打開策として，複数のコアで処理を並列実行できるマルチコア CPU の開発へと向かい始めた．その結果，チップあたりの演算性能は向上し，相対的にメモリアクセス性能に関する演算性能とのバランスの維持が重要になってきている．

とりわけ，NAS CG ベンチマークや Wisconsin ベンチマークに代表されるいくつかの重要アプリケーションでは，アプリケーション上での転送単位は 8 バイトまたは 4 バイトの不連続アクセスとなる．このため，キャッシュベースの CPU アーキテクチャでは有効なデータが 8 バイトまたは 4 バイトしかないキャッシュライン単位の非効率的なメモリアクセスが発生してしまい，著しい性能低下があった．

筆者らは以上の状況を鑑み，従来の CPU では不連続アクセスによって大幅な性能低下が生じてしまうアプリケーションの高速化を可能とするアーキテクチャを開発することを目的として研究を進めている．

上記の目的の達成のため，これまで筆者らはキャッシュベースの COTS の CPU やマザーボードをそのまま用いることが可能でメモリスロットに装着可能なベクトル型のプリフェッチ機能を有するメモリモジュールである DIMMnet-2 および DIMMnet-3 の研究開発を行ってきた．キャッシュベースの COTS の CPU にこれらを適用する場合にはキャッシュライン無効化のオーバーヘッドがかかり，これが性能向上の足かせとなっていた．

一方，Cell Broadband Engine(Cell/B.E.) や SpursEngine のようにキャッシュを持たない単純な CPU コアを多数内蔵するマルチコア CPU が注目されている．このような CPU コアではキャッシュの代わりに小容量のローカルメモリを持ち，主記憶とローカルメモリの間を DMA 転送によりデータ転送することでキャッシュの代用をさせる．上記のアーキテクチャによるチップ内演算能力の向上とは裏腹に，これに見合ったデータ供給能力の実現が実効性能の鍵を握る．

DMA で主記憶をアクセスする CPU における不連続アクセスの高速化に関する従来研究は数少ないが，Cell/B.E. における DMA list はその一つである．しかし，特にバースト長が小さい不連続アクセスが支配的なアプリケーションにおいては効果が限定的である．よって，さらなる高効率を実現できるアーキテクチャの開発が望まれる．

本報告では DMA で主記憶をアクセスする CPU における不連続アクセスに伴う上記の課題の解決方法の提案するとともに，その性能評価を東芝 Cell リファレンスセット 上で行なった．

本報告では，第 2 章で DMA で主記憶をアクセスする CPU とその一例である Cell Broadband Engine およびそのシステム開発環境である東芝 Cell リファレンスセットの概要について紹介する．第 3 章で上記アーキテクチャをとる CPU の不連続アクセスにおける課題を述べる．第 4 章で上記の課題の解決法を提案する．第 5 章では提案方式の性能評価について述べ，第 6 章で関連研究について述べ，第 7 章でまとめる．

2. DMA で主記憶をアクセスする CPU

DMA で主記憶をアクセスする CPU としては以下に示す

† 東京農工大学
Tokyo University of Agriculture and Technology
†† (株) 東芝，研究開発センター
Corporate Research and Developpment Center, Toshiba

Cell Broadband Engine(Cell/B.E.) や SpursEngine などがある．これらは，CPU コアを単純化して多数チップ内に内蔵することにより，チップ内の演算性能を向上させるとともに，データ転送をキャッシュと比較してプログラマから制御しやすいものとすることで，実行性能チューニングの可能性を高めたアーキテクチャを採用している．

Cell Broadband Engine(Cell/B.E.) は IBM，ソニー，ソニー・コンピュータエンタテインメント，東芝が共同で開発したマルチコア・アーキテクチャを採用した高性能プロセッサである．

図 1 では，Cell/B.E. の構成を示している．

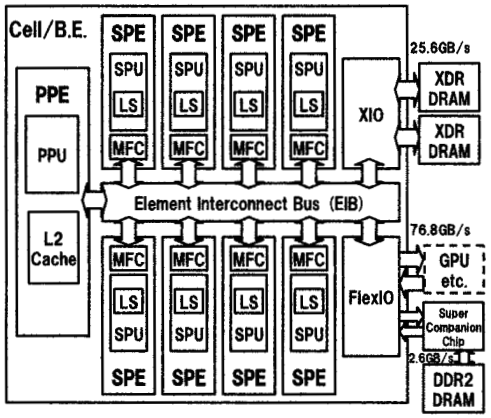

図 1 Cell Broadband Engine (Cell/B.E.) の構成

(1) マルチコア・アーキテクチャ・デザインを採用
(2) 8 個の演算に特化したコア SPE と，1 個の汎用コア PPE を搭載
(3) 各 SPE は SIMD 型演算処理ユニット，128 個の 128 ビットレジスタファイル，および 256KB のローカルストレージを有す
(4) 外付けの主記憶 XDR は，XIO を介して接続しており，他の外部チップは，IO Interface(FlexIO) を介して接続
(5) PPE，8 個の SPE，主記憶，および他の外部チップの相互間データ転送には，超高速データ転送バス Element Interconnect Bus(EIB) が用いられる

Cell/B.E. はマルチコアというものの，インテルの Core 2 Duo などのマルチコア・プロセッサとメモリの扱いが異なる．Cell/B.E. が搭載する 8 個の SPE は，それぞれ LS を持ち，実行するコードやデータをすべて LS に格納する．しかし，SPE は直接主記憶にアクセスできないため，必要に応じて，演算に必要なデータなどを主記憶から LS へ DMA(Direct Memory Access) 転送しなければならない．

これは，コアごとにキャッシュをもつシステムと類似しているが動作はまったく異なる．キャッシュの場合，キャッシュの内容は，基本的にハードウエアによって自動的に主記憶との整合性が保たれるに対し，Cell/B.E. の場合，LS とメイン・メモリの内容は直接的には何の関係もない．あくまでも，プログラマが責任を持って，必要なデータを主記憶から SPE に渡さなければ，つまり明示的に DMA 転送を実行させなければならない．

3. 解決すべき課題

本章では DMA で主記憶をアクセスする CPU における不連続アクセスに関連する課題について Cell/B.E. を例に述べる．

3.1 DMA コマンドオーバーヘッド

DMA コマンドを発行するには少なからずソフトウェアオーバーヘッドが存在するので，細粒度の DMA 転送が頻繁に発生するアプリケーションの性能は制約される．この問題は Cell/B.E. にも実装されている DMA リストを用いることによりある程度軽減することが可能である．

3.2 内部バスの調停オーバーヘッド

Cell/B.E. のように調停回路から内部バスのアクセス権利を取ってから DMA 転送を行なう種類の CPU では，少なからず調停オーバーヘッドが存在するので，細粒度の DMA 転送が頻繁に発生するアプリケーションの性能は制約される．この問題は Cell/B.E. にも実装されている DMA リストを用いても軽減することができない．

3.3 内部バスの転送単位との兼ね合い

Cell/B.E. では前述の調停オーバーヘッドとの兼ね合いからも長めのバースト転送における内部バスの転送効率を向上させるために，内部バスの最低転送単位を 128 バイトに設定されている．ところが，NAS CG ベンチマークや Wisconsin ベンチマークに代表されるいくつかの重要アプリケーションではアプリケーション上での転送単位は 8 バイトまたは 4 バイトの不連続アクセスとなる．このため，DMA リストを用いて DMA コマンドオーバーヘッドを軽減したとしても，このようなアプリケーションにおけるバスの実効バンド幅は 8/128 または 4/128 に低下してしまう．

4. 提案方式

本章では上記の問題を解決するための解決策として，DMA で主記憶をアクセスする CPU への外付けハードウェア追加，そのコンパニオンチップへの改良および同 CPU への改良について提案する．

4.1 提案方式の基本コンセプト

DMA で主記憶をアクセスする CPU における前章での課題の解決策として，DIMMnet-2 と同様の連続化ハードウェア（分散/収集機構）を外部メモリに近い場所に追加することを提案する．提案方式の基本コンセプトを**図 2** に示す．

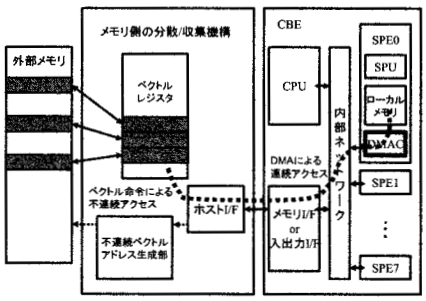

図 2 提案方式の基本コンセプト

表 1 に DIMMnet-2 の主なベクトル型アクセスコマンドを

示す．このうち，等間隔ロード/ストア，リストロード/ストアのコマンドが不連続アクセスの連続化を実行するものである．ロード系が外部メモリから一種のベクトルレジスタであるPrefetch Windowへの収集(Gather)処理を行い，ストア系が一種のベクトルレジスタであるWrite Windowからの外部メモリへの分散(Scatter)処理を行なう．

本提案は，このような分散/収集処理が可能な追加ハードウェアをDMAで主記憶をアクセスするCPUのメモリサイドに設けることにより，CPU内部での細切れなDMA転送コマンド発行を抑制し，それに伴う内部転送資源の浪費や効率低下に伴う性能低下を抑制するものである．

表1 DIMMnet-2の主なベクトルコマンド

ロード命令	
・連続ロード	VL(src, dst, DTYPE, iter)
・等間隔ロード	VLS(src, dst, DTYPE, iter, stride)
・リストロード	VLI(src, dst, DTYPE, iter, index)
ストア命令	
・連続ストア	VS(src, dst, DTYPE, iter)
・等間隔ストア	VSS(src, dst, DTYPE, iter, stride)
・リストストア	VSI(src, dst, DTYPE, iter, index)

4.2 ハード的な実装方式

4.2.1 DIMMnet装着による方式

DIMMnet-2やDIMMnet-3はCOTSのCPUやチップセット(コンパニオンチップ)，マザーボードに改造をすることなく，メモリスロットに後付けで装着することで不連続アクセスの連続化機能を追加することができる．

現状のDIMMnet-3はCRSのSO-DIMMスロットに装着可能な子基板を有しており，前述の基本コンセプトをCRSに実現可能である．ただし，CRSのSO-DIMMスロットはピークバンド幅が2.56GB/sに留まっており，その十倍のバンド幅であるXDR DRAMによる主記憶に比べてバンド幅が低いので，その効果は限定的であるものと考えられる．

一方，CRS上ではXDR DRAMがメインボード上に直接実装されているが，XDR DRAM自体は技術的にはメモリモジュールの形態での実装が可能である．よってXDR DRAMのメモリスロットを装備したCell/B.E.関連機器においては，XDR DRAMインタフェースを有するDIMMnet子基板を開発することで，高い主記憶バンド幅を背景にした基本コンセプトを実現できる可能性がある．

4.2.2 コンパニオンチップ改良による方式

Cell/B.E.自体にはFlexIOという上記のSO-DIMMスロットよりも高いバンド幅を有する入出力ポートが存在する．ノースブリッジに相当するコンパニオンチップやマザーボードの新規開発が必要になるが，FlexIOインタフェースで動作する連続化ハードウェアと拡張メモリを実装することで，Cell/B.E.自体には改造を加えることなく，前述の基本コンセプトを実現可能である．

4.2.3 CPUチップ改良による方式

東芝によるSpursEngineやIBMによるRoadRunner向けCPUなど，Cell/B.E.の派生製品である改良型CPUの開発事例がいくつかある．このようなケースではCPUチップにマイナーな改造を加えることで，従来のCell/B.E.に付加価値を加えることができる．

そのような派生CPUの開発の際に，本提案のハードウェアを主記憶コントローラや，入出力コントローラの中に実装することで，本提案のコンセプトを高性能に実現することが可能であると考えられる．

4.3 ソフト面での改造方法

上記の提案方式におけるソフトウェアの改造においては以下のような方針で行なう．

(1) 主記憶とローカルメモリの間で細かいデータサイズで行なわれる多数回のDMAコマンドの繰り返しを，主記憶との間でPrefetch Windowに収集/Write Windowから分散する少数回のベクトルロードコマンドと，Windowとローカルメモリの間で基本的にはWindowサイズで行なわれる少数回のDMAコマンドに変更する．

(2) DMAで主記憶をアクセスするCPUにはキャッシュがないため，Pentium4等のキャッシュベースのCPU向けの改造の際に必要になるキャッシュライフフラッシュ命令の挿入は不要である．

5. 性能評価

本章では，性能評価を述べる．まずは，CRS実機での評価，つまり，Cellプログラミングでは欠かせないDMA転送の性能を測定し，次に，実際のアプリケーションにおけるベクトル命令の効果を把握するために，ベンチマークプログラムによるCRS実機上でのエミュレーション評価を行った．

5.1 評価環境

表2は実機評価の評価環境を示している．

表2 評価環境(CRS実機)

モデル	Cell リファレンスセット
CPU	Cell B.E. / 3.2GHz
チップセット	TOSHIBA Super Companion Chip
主記憶	XDR 512MB ECC 対応
グラフィック，IO用メモリ	SO-DIMM 512MB
基本ソフトウェア	ハイパーバイザ OS "Beat" ゲスト OS "Level2 Linux" IO マネージメント SPE マネージメント
ソフトウェア開発環境	Eclipse 統合開発環境 (コンパイラ，デバッガ， バイナリユーティリティ， パフォーマンスモニタを含む)

表3はコンパイル環境を示している．

表3 コンパイル環境

GCC バージョン	ppu-gcc / spu-gcc 3.4.1
コンパイルオプション	PPE: -O3 -m32 SPE: -O3

5.2 評価方法と結果

5.2.1 DMA転送バンド幅

ここでは，DMA転送の時間を測定し，その転送データサイズによるバンド幅を計算した結果を示している．これは主記憶からLSメモリへGetする際のDMA転送バンド幅測定結果である．データサイズは4Bから16KBまで測定する． 図3で示してるように，SPEを1個だけ使う場合のバンド幅は最大13.6GB/sになる．しかし，SPEを2個使う場合は，外部メモリインタフェースの衝突が発生するため，データサイズが

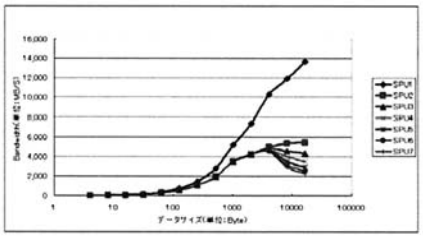

図 3　GetDMA バンド幅測定結果

大きい時は，バンド幅が半分弱に減る．データサイズが小さい時は，データ転送バス Element Interconnect Bus(EIB) の転送単位が 128 バイト刻みであるために転送効率が低下する点と，そのバス調停に伴うオーバーヘッドが顕在化する点の相乗効果により，バス効率の悪さが，外部メモリインタフェースの衝突による影響を上回るため，バンド幅に大きな変化はない．SPE3〜7 個使う場合も同様である．

そこで，プログラマが DMA 転送のタイミングをどんなにうまく調整することができるかによって，プログラムの実行時間が大きく変わってくる．図 4 が示してるように，LS メモリから主記憶に DMA 転送 (Put) するときも，主記憶から LS に DMA 転送するときと同じ理由で，SPE の個数が増えるほど，各 SPE に対するバンド幅は減る．Put の場合，最大のバンド幅が 21.1GB/ s である．

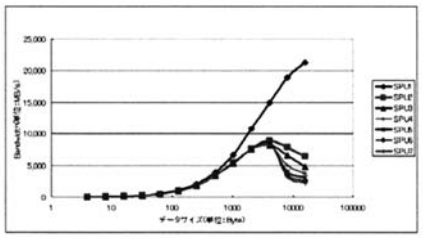

図 4　PutDMA バンド幅測定結果

5.2.2　DMA リストを使ったギャザーのバンド幅

Cell で SPE は，DMA リストと呼ばれる機能を使って，図 5 で示してるように，一度の DMA 転送で主記憶上の複数の領域のデータをやり取りすることができる．この機能をデータのギャザーと呼ぶ．この機構を利用して，DMA 転送の起動

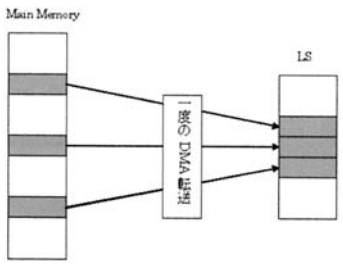

図 5　DMAlist ギャザー

にかかるコストを減らすことが可能である．図 6 は，バースト長を 4 バイトに指定している DMA コマンドのリストを 16 個有する DMA リストをブロッキング DMA 転送した場合のバンド幅の測定結果である．グラフの横軸になっている data

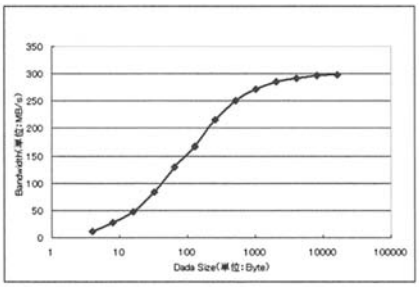

図 6　DMAlist ブロッキング測定結果

size は，DMA 転送されるトータルのデータサイズである．このグラフでは，データサイズが 16KB であると，4 バイトのバースト長をしている DMA コマンドのリストを 16 個有する DMA を 256 回繰り返して実行することになり，その時のバンド幅が 298MB/ s で，単純な 4 バイトの DMA 転送のバンド幅 23MB/s より，13 倍までの性能向上が見られる．図 7 では，DMA リスト間のブロッキングがなしの DMA 転送のバンド幅を測定している．グラフでわかるように，DMA リストノンブ

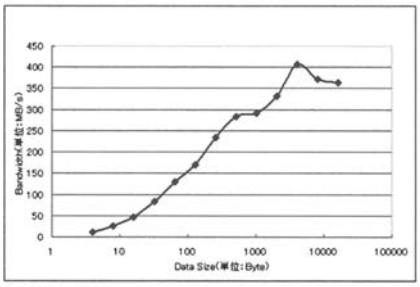

図 7　DMAlist ノンブロッキング測定結果

ロッキングのときの，最大のバンド幅は 406MB/ s で，単純な 4 バイトの DMA 転送のバンド幅 23MB/s より，18 倍までの性能向上が見られる．DMA リストノンブロッキングで，もっと高い性能が現れるのが，その原因として，前の DMA 転送の完了を待たずに次の DMA 転送が始まるからである．

これが，CRS 自体の等間隔アクセスで出せる最大の性能であり，次節では，DIMMnet-3 の装着のようにハードウェア側に直接ベクトルアクセス機構を用いるようにした場合の性能に関して評価を行った．

5.2.3　Wisconsin ベンチマークによる評価

本研究では，Wisconsin ベンチマークを用いて等間隔アクセスによる主記憶データベースの高速化の評価を行った．

Wisconsin ベンチマークでは 15 個の属性からなるタプルが 10K 個，または 1K 個から構成されるデータベースが検索対象になる．1 個のタプルは 15 個の属性が 4 バイトのデータまたはポインタからなる (合計 60 バイト)．本評価ではタプル数を

1K 個にして，データが全部 Cell 上の 1 個の SPE のローカルメモリ 256KB のローカルメモリに入る状態で評価を行った．

```
typedef struct {
   int unique1;
   int unique2;
   int two;
   int four;
   int ten;
   int twenty;
   int onePercent;
   int tenPercent;
   int twentyPercent;
   int fiftyPercent;
   int unique3;
   int evenOnePercent;
   int oddOnetPercent;
   char *string;
   char *string;
} Tuple;
```

このようなデータベースのある属性に対する検索処理を行なう際には等間隔アクセスとなってしまう．Wisconsin ベンチマークでは，ストライド値 60 バイトで不連続に格納されている 4 バイトデータに対する等間隔アクセスとなる．

例えば，Cell/B.E. だと，DMA list を用いたとしても 1 回の DMA 起動で 4 バイトのデータを 16 個しか転送することができない．内部バス上では 128 バイト単位で転送されるため，個々の 4 バイト転送が 128 バイトの転送と同じ内部バスバンド幅を消費する．しかし，例えば DIMMnet-3 の Prefetch Window だとサイズが 512 バイトであるため 4 バイトのデータを 128 個まで集めて，1 回の内部バストランザクションで DMA 転送することが可能である．

本評価では以下のクエリーを用いる．クエリーの番号は参考文献 [5] において遅延変動の評価に用いられていたものと合わせる．

(Q7) select MIN(unique2) from tenk1

評価に際しては上記クエリーを Cell/B.E. 上の PPU と 1 個の SPU で動作する C 言語で記述し，これをオリジナルプログラムとした．これを DIMMnet のベクトルコマンドを用いるように改造して，比較評価を行った．ハードウェアで実行する部分を記述したエミュレーション版を作成し，その実行結果を確認し，改造の妥当性をチェックした．

(1) ゼロ遅延モデルでの検索性能

まずは，ハードウェアによる外部メモリアクセスが理想的で，ベクトル等間隔ロード関数コール直後 1 回目のコマンド完了フラグチェックまでに Prefetch Window へのロードが終わってしまうほど十分にハードウェアが低遅延な場合 (これをゼロ遅延モデルと呼ぶことにする) に相当する加速率を測定する．本測定においてはハードウェア部の記述をしている関数の中身をコメントアウトすることで，ハードウェア部の性能不足に起因する遅延がゼロになった状態の実行時間を再現して，ベクトルコマンド起動に関するオーバーヘッドを含んだ性能を測定した．

上記クエリー Q7 に関して，ゼロ遅延モデルの評価を行った結果を図 8 で示す．測定結果は，クエリー Q7 の性能を，オリジナルのベンチマークの実行時間と比較したときの相対値である加速率で示している．ここで，Prefetch Window のサイズは 512B，つまり 4 バイトのデータを 128 個分に固定している．PW1 は Prefetch Window を単純に 1 枚用いた場合の加速率で，PW2 は Prefetch Window を 2 枚用いた場合で，original list は Cell/B.E. 自身のデータギャザー機構を使った場合の加速率である．

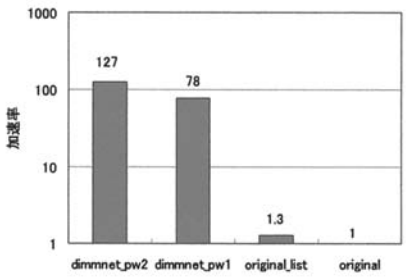

図 8 ゼロ遅延モデルでの Wisconsin Q7 検索性能

その結果，オリジナルに比べ，Prefetch Window を 1 枚だけ用いた場合は 78 倍，Prefetch Window を 2 枚用いた場合は 127 倍の加速率が得られた．そして，Cell 自身のデータギャザー DMAlist 機構を使った場合に比べても，Prefetch Window を 1 枚だけ用いた場合は 60 倍，Prefetch Window を 2 枚だけ用いた場合は 98 倍の加速率が得られた．

(2) プリフェッチにかかる遅延時間の影響

DIMMnet 上の外部メモリから Prefetch Window までの等間隔ベクトルロード実行にかかる時間を変化させたときの性能の変化を測定した．仮想的に変動させる遅延は，ベクトルコマンドが実行する処理を記述した関数をコメントアウトし，代わりにベクトルコマンドの実行が消費する時間に対応する回数の空きループの反復回数によって変化させた遅延を挿入することで実現する．ここで，空きループと遅延時間の関係はあらかじめパフォーマンスカウンタによる予備実験で確認しておく．本測定における対象もクエリー Q7 である．その結果を図 9 で示す．

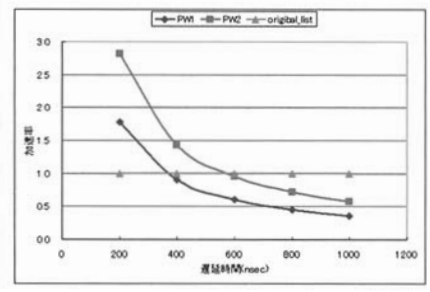

図 9 プリフェッチの遅延時間を変動させた時の Q7 処理時間

その結果，少なくとも本実験の規模の小規模データベースの検索においては，プリフェッチの遅延時間が加速率に敏感であり，実装の際にはこのチューニングが大切であることがわかった．また，キャッシュベースの CPU における評価結果 [5] と

同様に，Prefetch Window が 1 枚に比べ 2 枚の場合，等間隔ロードコマンド実行にかかる時間に対する耐性が強かった．使用する Prefetch Window を 1 枚増やすごとに 0.2 μ s 程度の遅延を隠蔽できることがわかった．タプル数 1K という小規模なデータベースでの実験では，Prefetch Window のサイズや枚数を増やす効果は少ないと思われるが，よりタプル数が多いデータベースを検索する場合には，これらの測定パラメータの変更により効率向上が期待できると考えられる．

6. 関連研究

メモリコントローラを改善することによる不連続アクセスの高速化に関する従来研究には Impulse[6]，SDT[7] がある．しかし，これらは Cell/B.E. のような DMA で主記憶をアクセスする CPU への適用を提案するものでもない上，その種の CPU と組み合わせた場合における効果を評価したものでもない．

キャッシュベースの CPU における不連続アクセスの高速化に関する従来研究には筆者等が行なった DIMMnet-2 を用いた研究がある．NAS CG によるリストアクセスの高速化 [4] や，Wisconsin ベンチマークによる等間隔アクセスの高速化 [5] が評価されている．しかし，これらは DMA で主記憶をアクセスする CPU における評価ではない．また，キャッシュベースの CPU に適用した場合は，キャッシュラインの無効化が必要であり，性能向上は無効化処理によって効果が半減してしまう．

DMA で主記憶をアクセスする CPU における不連続アクセスの高速化に関する従来研究は数少ないが，Cell/B.E. における DMA list[3] はその一つである．しかし，DMA list では内部バス調停オーバーヘッドが回避できないことや，内部バスの最小転送単位が 128 バイトであるためキャッシュベースの転送における転送効率の悪化と同様に有効なデータの割合が低い状況に陥るので，特にバースト長が小さい不連続アクセスが支配的なアプリケーションにおいては効果が限定的である．

7. おわりに

本論文では，メモリサイドに配置された Gather ハードウェアによる，DMA で主記憶をアクセスする CPU 上での不連続アクセスの連続化を提案し，その効果を東芝 Cell リファレンスセット上で測定した．

まずは，Cell/B.E. 自身が持っている DMA リストという機構を利用して等間隔アクセスの性能を測定した．次に Cell/B.E. の主記憶に DIMMnet-3 が装着されている状態を仮定して，等間隔アクセスを主体とする処理として Wisconsin ベンチマークを用い，東芝 Cell リファレンスセットの実機上でのソフトウェアエミュレーションと人工的遅延挿入によって性能を評価した．Wisconsin ベンチマークでは，データベースのある属性に対する検索処理を行う際には等間隔アクセスとなる．本研究では，DIMMnet-3 の等間隔アクセス命令 VLS を検索処理に適用した．

その結果，プリフェッチがプリフェッチ完了フラグ確認より前に終わる実装を仮定した場合，最小値を検索する問合せ処理 Q7 が Prefetch Window を 1 枚用いることで DMA リストを用いるプログラムに対して 60 倍程度高速化され，Prefetch Window を 2 枚用いると 99 倍高速化されることがわかった．

また，プリフェッチにかかる遅延を変動させた場合は，少なくとも本実験のタプル数 1K という小規模データベースの検索においては，プリフェッチの遅延時間が加速率に敏感であり，実装の際にはこのチューニングが大切であることがわかった．また 512 バイトの Prefetch Window を 1 枚増やすごとに 0.2 μ s 程度の遅延を隠蔽できることがわかった．

Prefetch Window 枚数を増やす実装は長いベクトル長においては有効と考えられる．また通常の DMA 転送を用いた場合のバンド幅の挙動から，Prefetch Window のサイズを増やす実装も有効と考えられる．ベクトルアーキテクチャはスループットが確保される適切な実装を行なえば，長いベクトル長の処理については遅延時間の影響を大幅に軽減できる。以上を考慮すれば，短いベクトル長で遅延が加速率に敏感であった結果をあまり悲観視する必要はないと思われる。以上の評価の結果から，提案方式は DMA で主記憶をアクセスする CPU 上での不連続アクセスにおいても，有効性があることがわかった．

ローカルメモリには納まりきらない主記憶上の大きなデータベースに対する性能評価は今後の課題である．また，Prefetch Window のサイズや枚数を増やし，DMA 転送効率の向上や，プリフェッチにかかる時間をさらに隠蔽することも課題として挙げられる．また，今回は Cell/B.E. に 8 個内蔵されている演算に特化しているプロセッサである SPU を 1 個だけ使って，性能評価を行っている．今後の課題としては，8 個の SPU を使って性能評価を行うことが挙げられる．

参考文献

1) 東芝セミコンダクター社：“Cell Broadband Engine”, http://www.semicon.toshiba.co.jp/product/micro/cell/index.html
2) Cell User's Group：“Cell 関連情報”, https://www.cellusersgroup.com/modules/product/
3) M. Kistler, M. Perrone, and F. Petrini: "Cell Multiprocessor Communication Network: Built for Speed" IEEE Micro, vol. 26(3) pp. 10-23, May-June 2006.
4) 田邊 昇, 安藤 宏, 箱崎 博孝, 土肥 康孝, 中條 拓伯, 天野 英晴: “プリフェッチ機能を有するメモリモジュールによる PC 上での間接参照の高速化”, 情報処理学会論文誌コンピューティングシステム,Vol. 46, No. SIG12 (ACS11), pp. 1-12(Aug. 2005).
5) 田邊 昇, 羅 徹哲, 中條 拓伯, 箱崎 博孝, 安藤 宏, 土肥 康孝, 宮代 具隆, 北村 聡, 天野 英晴: プリフェッチ機能を有するメモリモジュールによる等間隔アクセスの高速化, ハイパフォーマンスコンピューティングと計算科学シンポジウム (HPCS2006).
6) 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)
7) 宮崎 純, 府川 智治, 田中 清史: ストライドデータアクセスによる主記憶データベースの問合わせ処理の評価, 日本データベース学会 Letters, Vol.3, No.2.
8) Jim Gray. The Benchmark Handbook. Morgan Kaufmann, 1993.