Vol.2010-HPC-124 No.6
2010/2/22

ベクトルアクセス機構を有するメモリモジュールによる不連続なDMAの効率化

塚本太郎[†1] 田邊昇[†2]
太田淳[†1] 中條拓伯[†1]

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．この種の 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， DMA で主記憶をアクセスする CPU へのメモリ側の連続化ハードウェアによる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．

Improving Efficiency for Discontinuous DMA Using a Memory Module with Vector Access Functions

Taro Tsukamoto,[†1] Noboru TanabeE,[†2]
Atsushi Ohta[†1] and Hironori Nakajo[†1]

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

いくつかの重要アプリケーションでは，メモリアクセスの空間的局所性が乏しく，不連続アクセスが性能ネックである．例えば NAS CG ベンチマークはリストアクセスが性能ネックである．Wisconsin ベンチマークは主記憶上にデータベースが配置された場合，等間隔アクセスが性能ネックである。これらは，キャッシュベースの CPU では例えば 128 バイトのキャッシュラインの中に有効なデータが 8 または 4 バイトしかない非効率的なアクセスが発生するため著しい性能低下があった．

上記の問題の解決のため，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[1)2)3)] および DIMMnet-3[4)] 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.)[5)6)] 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．これら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．これらの CPU ではキャッシュラインと同等の内部バス転送単位とアプリケーションの不整合の問題だけでなく、DMA 起動やバス調停のためのオーバーヘッドが存在し、キャッシュベースの CPU 以上に不連続アクセスの問題が深刻である．

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 リスト[7)] 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．

本論文では 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を提案し，その評価を東芝 Cell リファレンスセット (以下 CRS) 上で行なった．以下，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/B.E. お

†1 東京農工大学
Tokyo University of Agriculture and Technology
†2 株式会社東芝, 研究開発センター
Corporate Research and Developpment Center, Toshiba Corporation

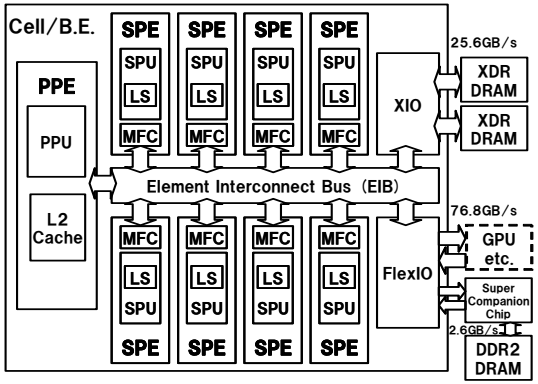

図 1 Cell Broadband Engine (Cell/B.E.) の構成
Fig. 1 Structure of Cell Broadband Engine (Cell/B.E.)

よびその開発環境である CRS 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 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 Cell Broadband Engine(Cell/B.E.) や，東芝の 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ている．図 1 に，Cell/B.E. の構成を示す．

(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 Local Strage(LS) を有す
(4) 外付けの XDR DRAM ベースの主記憶は XIO を介して接続しており，他の外部チップは I/O 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. 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．

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．

3.4 転送時のアラインメント合わせ

Cell/B.E. の DMA コントローラのように，DMA 転送を行う際のソースとデスティネーションの間でアラインメントがずれていると，直接 DMA でコピーできない実装がある．そ

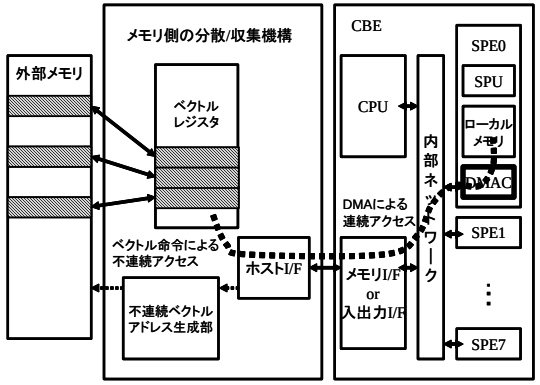

図 2 提案方式の基本コンセプト
Fig. 2 The basic concept of proposed architecture

の場合，一旦バッファ領域に転送したいデータを含むブロックを DMA 転送し，その後にロードストア命令で所望の位置にソフト的にコピーしなおす必要がある．Cell/B.E. には上記のようなオーバーヘッドが存在するため，アプリケーションのアクセスパターンによっては性能低下の原因となる．

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．

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．

表 1 DIMMnet-2 の主なベクトルコマンド

ロード	連続	VL
	等間隔	VLS
	リスト	VLI
ストア	連続	VS
	等間隔	VSS
	リスト	VSI

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．

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， Cell/B.E. の主記憶側 (主記憶コントローラ内または XDR DRAM の場所) に DIMMnet-2 同様の Gather 回路がある状態を仮定して，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，CRS 上で性能を評価した．表 2 に実機評価の評価環境を示す．

5.1 評価に用いたベンチマーク

本研究では，データベースの検索性能評価プログラムである Wisconsin ベンチマークを用いて，メインメモリに置かれたデータベース要素への等間隔アクセス処理の効率化についての評価を行った．

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．

表 2 評価環境

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
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) CTK ライブラリ
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 -no-finline-functions

以下は評価を行ったクエリの SQL 文である．

各クエリの SQL 文

$(Q1)$ $select\ *\ from\ tenk1\ where\ (unique2 > 301)\ and\ (unique2 < 402)$

$(Q2)$ $select\ *\ from\ tenk1\ where\ (unique2 > 647)\ and\ (unique2 < 65648)$

$(Q3)$ $select\ *\ from\ tenk1\ where\ unique2 = 2001\ and\ (t2.unique2 < 1000)$

$(Q7)$ $select\ MIN(unique2)\ from\ tenk1$

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．その際，記述には CTK(Cell Tool Kit) ライブラリを用いた．CTK による通信関数はアラインメントに関する煩雑さを関数内部に隠蔽しているので，Wisconsin ベンチマークの中で多発するアラインメントがされていない位置に不連続にならぶ小さなデータの読み出しには ctk_dma_get_small_block() 関数を用いて簡潔に実装した．これは関数名に small がついている関数なので 16KB 以下のデータを通信する際に長さ条件判断しないで行う分だけ低遅延である．最適化オプションを単純に-O3 とするとインライン展開が起き、後述するように中身をコメントアウトされた VLS 関数や VLI 関数が実質消されてしまい DIMMnet 起動用関数オーバーヘッドが測定に組み込まれなくなる恐れがある．このため，-O3 にインライン化だけ抑制する (-no-finline-functions) オプションを追加した．

5.2 アラインされないワードの Get 性能

提案方式である DIMMnet-2 と同等の Gather 回路による連続化を行った場合，不連続アクセスはキリの良いアドレスにアラインされた位置にマップされている Prefetch Window(一種のベクトルレジスタ) への連続 DMA となる．一方，上記回路を用いない場合は，Wisconsin ベンチマークの中ではアラインメントがされていない位置に不連続にならぶ 4 バイトデータの読み出し (Get) が多発する．Cell/B.E. の場合，アラインされているデータ転送しか通信ハードウェアは扱えないので，アラインされたブロックをバッファに転送した後に，あらためてロードストア命令などで正しい位置に 4 バイトをコピーする必要がある．これは後述の測定では CTK の通信関数内部の処理になりプログラマからは隠蔽されている．CTK を使う場合，ユーザのソースコード上にアラインメント補正処理は見えないが，関数内部での処理にはアラインメントによって性能に影響が出ると考えられる．このため，アラインされた位置間の Get とアラインメントがされていない位置への Get でバンド幅にどの程度の差が生じるのかを測定した．

その結果を図 3 に示す．横軸は転送サイズで、16 で割り切れるアドレスからのソースのオフセット（バイト数）を 0,4,8,12,16 と変化させて測定したものである．その結果，CTK を用いた場合，サイズだけでなく，オフセットによってバンド幅は変動する．特に 4 バイトの get の際のバンド幅に着目すると、16 バイト境界にアラインした場合の 21.5MB/s に比べて，アラインされないワードを DMA で Get する際のバンド幅は全て平均 8.2MB/s に過ぎない．アラインメントにより 4 バイトという小さいサイズのバンド幅は 1/2.62 に低下してしまうことがわかった．

一方，図 3 のオフセット＝ 0 のラインが提案方式の Prefetch Window のサイズを変化させた時の実効バンド幅に相当する．その実測値は DIMMnet-2 と同サイズの 512 バイトの時に 2.44GB/s，その 32 倍の 16KB の時に 14.5GB/s となった．これはアラインされないワードを DMA で Get する際のバンド幅の各 298 倍，1768 倍である．この差が Wisconsin ベンチマークを提案方式で実行する場合の加速を生み出すものと考えられる．

5.3 ゼロ遅延モデルでの検索性能

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

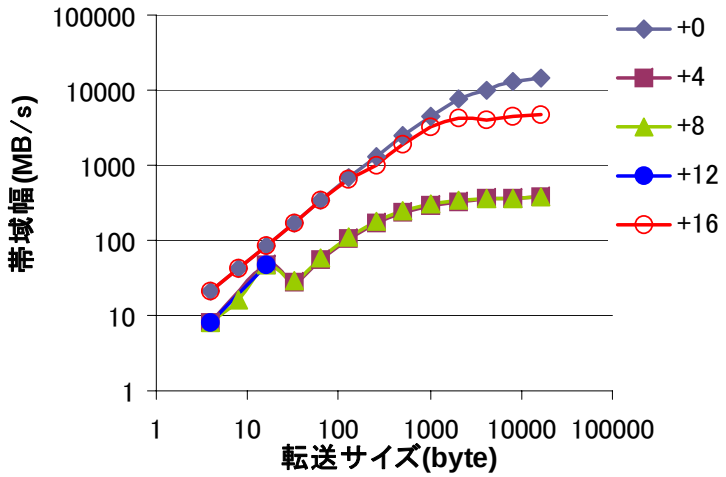

図 3 サイズとアラインメントによるバンド幅の変化 (ctk_dma_get_small())

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．この測定ではタプル数は 1000 と少ない状態であるが全てのタプルへのアクセスが LS には入りきらずに主記憶への DMA になる状態を再現し，アラインメントは常にアラインしているものと近似して get を行って時間測定した．

上記クエリー Q1,Q2,Q3,Q7 に関して，ゼロ遅延モデルの評価を行った結果を図 4 で示す．測定結果は，クエリー Q1,Q2,Q3,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 DIMMnet-2 と同等の 512B，つまり 4 バイトのデータを 128 個分に固定している．前節の実験結果から例えば Prefetch Window のサイズを 16KB に増やすなどすれば，さらに性能向上する伸びしろがある．

加速率に 10 倍強のクエリ（Q1,Q2) と 20 倍強のクエリ (Q3,Q7) があるが，これらは計算部分と DMA 部分の比率の違いにより生じているものであると考えられる．オリジナルのプログラムの全処理時間に占める DMA 時間の割合は Q1 が 91%，Q2 が 92%，Q3 が 97%，Q7 が 98%であった．提案方式は DMA 部分を加速するが，計算部分は基本的に加速しないため，計算の重さが加速率の差となって現れる．

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コンパイルオ

プション-O3 -no-finline-functions の場合では最大 23.0 倍 (Q3) の加速率が得られた．

なお，この測定ではアラインメントは常にアラインしているものと近似して get を行って時間測定している．一方，実際の加速率はオリジナルのプログラムの DMA が 4 回に 1 回だけアライン，3 回はミスアラインであるべきであるため，アラインメントまで考慮した加速率は図 4 で示された値よりも 2 倍弱程度大きくなるものと考えられる．オリジナルのプログラムの全処理時間に占める DMA 時間の割合は Q1 が 91〜98%とクエリ毎に異なるが大半を占めているため，DMA 時間へのミスアラインを考慮した補正率 21.5/(21.5/4+8.2*3/4) = 1.86 倍がほぼストレートに全体の処理時間の短縮に寄与すると考えられる．

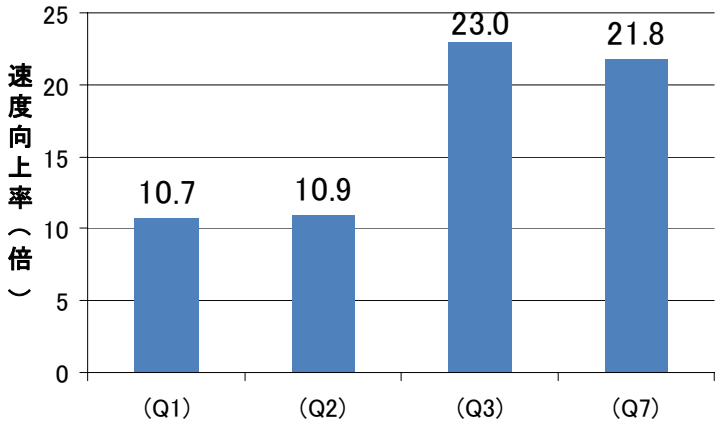

図 4 ゼロ遅延モデルでの検索性能 (ミスアライン効果補正前)

5.4 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疑似遅延時間を挿入する．疑似遅延時間の生成には，1 μ秒未満の遅延は空ループで，1 μ秒以上の遅延は CTK のライブラリにある ctk_usleep 関数を用いた．遅延時間を 0.1 μ秒〜40 μ秒の間で変化させて測定を行った．Prefetch Window を 1 枚用いる場合 (PW1) と 2 枚用いる場合 (PW2) について測定した．各クエリの測定結果は図 5 〜図 8 に示す．

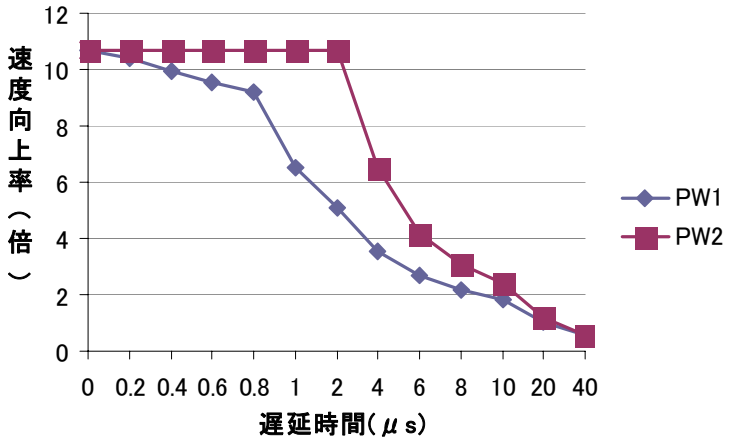

図 5 プリフェッチの遅延を変動させた時のクエリ Q1 の速度向上率

その結果，全てのクエリに対して 512 バイトのデータのプリフェッチにかかる時間が 2 μ秒以下ならば十分に高い加速率が得られることがわかった．外部メモリの種類の違い (DDR, DDR2，DDR3，XDR など) やそのバンク数やアクセスパターンによって不連続データのプリフェッチにかかる時間が変動する．具体的には 100MHz の DDR ベースの 2 バンクのメモリである DIMMnet-2 上での 60 バイト間隔の等間隔アクセスは 1 μ秒強であることが Verilog レベルでのシミュレーションや実機上で観測されている．DIMMnet-2 の後継機である DIMMnet-3 では周波数やバンク数がそれぞれ 2 倍以上に向上しているので，上記のアクセスは 1 μ秒以下になることが予想される．

さらに，DIMMnet-3 などを外に付加するのではなく，CPU のメモリコントローラを改良する場合については Cell/B.E. に採用されている XDR DRAM のアクセス遅延やバンド幅をプリフェッチ遅延のベースとして考えることができる．XDR DRAM などの高速なメモリでは，DDR2 ベースの DIMMnet-3 より大幅に少ない遅延を想定できるため，本方式

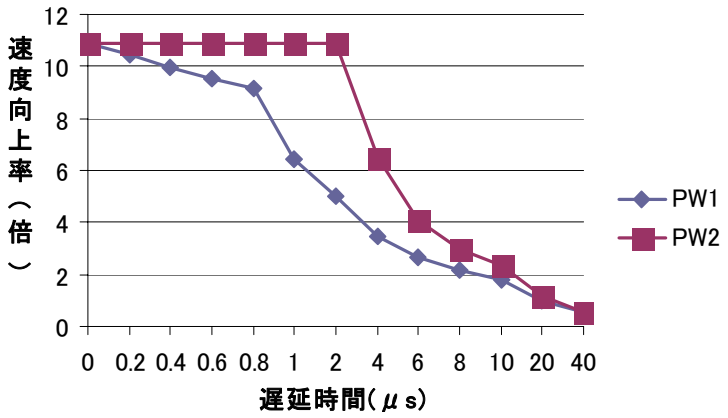

図 6 プリフェッチの遅延を変動させた時のクエリ Q2 の速度向上率

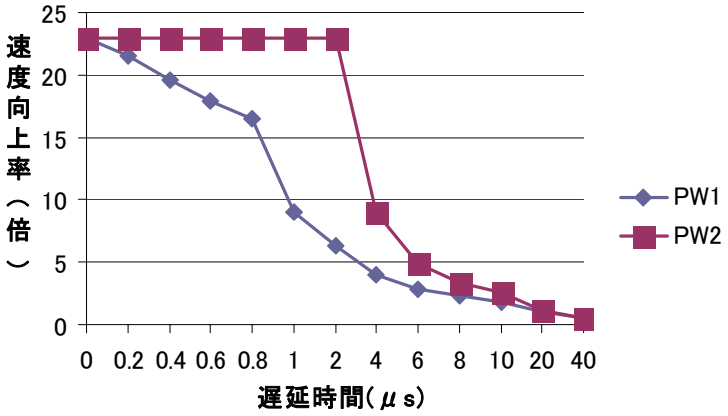

図 7 プリフェッチの遅延を変動させた時のクエリ Q3 の速度向上率

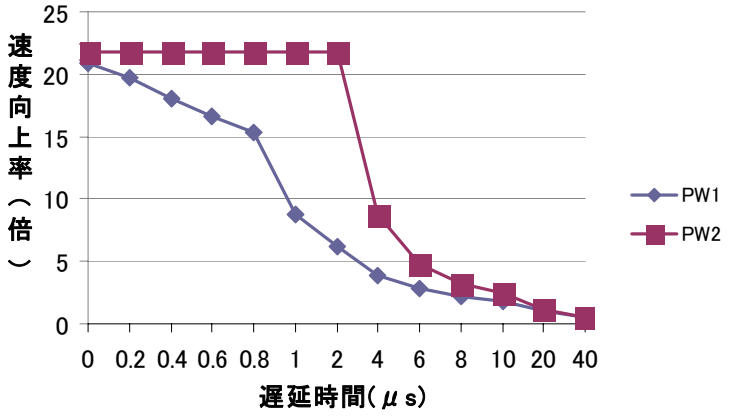

図 8 プリフェッチの遅延を変動させた時のクエリ Q7 の速度向上率

は有望であると言える．

また，キャッシュベースの CPU における評価結果[2] 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．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[8]，SDT[9] 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[1] や，Wisconsin ベンチマークによる等間隔アクセスの高速化[2] 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．それに対して，本論文で評価している DMA ベースで主記憶をアクセスする CPU ではそのようなオーバーヘッドは本質的に存在しないので，不連続アクセスの連続化を行う意義が大きい．本論文はその効果を定量的に示している．さらに，上記の従来研究は DIMMnet-2 という別ハードウェアを CPU の外側に付加する構成に限定しているが，本論文では CPU と同一チップ上にあるメモリコントローラの改善に関する提案と評価を行っている．

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 リスト[7] はその一つである．しかし，DMA リスト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，メモリ側に配置された 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 (CRS) 上で測定した．

Cell/B.E. の主記憶側 (主記憶コントローラ内または XDR DRAM の場所) に DIMMnet-2 同様の Gather 回路がある状態を仮定して，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，CRS 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-2 と同様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低遅延なメモリシステム実装を仮定した場合，最小値を検索する問合せ処理が単純な DMA を繰り返す場合に比べ，Prefetch Window を 1 枚だけ用いた場合は 10.7〜23.0 倍の加速率が得られた．

一方，上記の加速率はアライメントに伴うオーバーヘッドを加味していない．16 バイト境界のアライメントをずらした場合，合わせた場合に比べて 4 バイト Get のバンド幅は 1/2.62 に低下することがわかった．これは 91〜98%が DMA 時間で占められるオリジナルの性能を 2 倍弱低下させる結果，加速率をさらに 2 倍弱上げることが予想される．

また，プリフェッチにかかる遅延を変動させた場合は，512 バイトのデータのプリフェッチにかかる時間が 2 μ秒以下ならば十分に高い加速率が得られることがわかった．100MHz の DDR ベースの 2 バンクのメモリである DIMMnet-2 上では 1 μ秒強であり，Cell/B.E. に採用されている XDR DRAM などのより高速なメモリでは，それより大幅に少ない遅延を想定できるため，本方式は有望であると言える．

さらに，Prefetch Window のサイズを 512B から 16KB に増やした場合，実効バンド幅は 2.44GB/s から 14.5GB/s に向上した．Wisconsin ベンチマークによる評価は 512B の時のものであるので，更なる性能向上の可能性を有していると言える．

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，複数の SPU を使って性能評価を行うことが挙げられる．処理速度が上がる分，データ供給能力も上げる必要があり，それに対応できる不連続アクセススループットが高いメモリシステムの設計も今後の課題である．

また，本論文で明らかになった有効性を背景に，Cell/B.E. と同様な SPU を 4 個内蔵して PCI express を有する SpursEngine と PCI express に対応した DIMMnet-3 を組み合わせた大規模データ可視化装置[11] への応用が予定されている．その詳細設計と試作・評価は今後の課題である．

謝　辞

本研究の一部は総務省戦略的情報通信研究開発推進制度 (SCOPE) の一環として行われたものである．

参考文献

1) 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).
2) 田邊 昇，羅 徴哲，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), p.55-62(Jan. 2006).
3) 北村 聡，濱田 芳博，宮部 保雄，伊澤 徹，宮代 具隆，田邊 昇，中條 拓伯, 天野 英晴:

DIMMnet-2 ネットワークインターフェースコントローラの設計と実装，情報処理学会論文誌，Vol. 46, No. SIG12 (ACS11), pp.13-26 (Aug. 2005).
4) 田邊 昇，北村 聡, 宮部 保雄, 宮代 具隆，天野 英晴, 羅 徴哲，中條 拓伯: 主記憶以外に大容量メモリを有するメモリ/ネットワークアーキテクチャ, 情報処理学会計算機アーキテクチャ研究会，2007-ARC-172-27, pp.157-162 (Mar. 2007).
5) 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
6) Cell User’s Group：“Cell 関連情報”, https://www.cellusersgroup.com/modules/product/
7) 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.
8) 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)
9) K.Tanaka, T.Fukawa：“Highly Functional Memory Architecture for Large-Scale Data Applications”, International Workshop on Innovative Architecture for Future Generation High-Performance Processors and Systems (IWIA2004), pp.109-118 (Jan. 2004)
10) Jim Gray. The Benchmark Handbook. Morgan Kaufmann, 1993.
11) 田邊 昇，佐々木 愛美，中條 拓伯，城 和貴：“大容量データ向け対話的実時間遠隔可視化装置の実現性検討”, 電子情報通信学会コンピュータシステム研究会，CPSY2008-18, pp.43-48 (Aug. 2008).