【解説論文】

Cell Broadband Engine™を用いたブレード・サーバーの設計と実装

佐貫 俊幸

The Design and Implementation of Cell Broadband Engine™ Processor based Blade Server

Toshiyuki Sanuki

IBMは, ソニー・グループ様, 東芝様と共に, Cell Broadband Engine（以下CBEと記す）と呼ばれる次世代のマイクロプロセッサーを開発した. CBEは, PowerPC®との互換性を保ちながら, SPE（Synergistic Processor Element）と呼ばれるSIMDエンジンを複数個搭載し, 高速の処理が可能となることを目指している. しかしながら, アプリケーション・プログラムからCBEの能力を活用するためには, 複数のSPEを有効活用した並列処理やデータの入出力を考慮に入れる必要がある. 我々は, このCBEを用いたブレード・サーバーCPBSを設計・試作した. 本論文では, このサーバーの基本設計を行うに当たり, CBEの性能を生かしたシステム・アーキテクチャーの提案をする. また, 実装したプロトタイプを用いて特定のアプリケーション・プログラムの評価を行ない, 大幅な処理の高速化が図れたことを報告する.

IBM, in collaboration with Sony Corporation and Toshiba Corporation, has developed a next generation microprocessor named Cell Broadband Engine (“CBE”). The CBE, while maintaining compatibility with 64-bit PowerPC®, incorporates multiple SIMD engines called the Synergistic Processor Element (“SPE”) and aims to achieve high-speed processing performance. However, in order to enable application programs to fully utilize the capability of the CBE, there is a need to take into consideration parallel processing and/or data transfer by effective use of multiple SPEs. We have designed and implemented a CBE processor based blade server (“CPBS”) that utilizes this CBE. In designing this CPBS, this paper proposes a system architecture that fully utilizes the capabilities of the CBE. Further, we would like to report a significant improvement of the processing speed through evaluation of specific application program using the implemented prototype system.

Key Words & Phrases：マイクロプロセッサー, アーキテクチャー, システム設計, デジタルメディア処理
microprocessor, architecture, system design, digital media processing

1. はじめに

PCの領域では, デジタルメディアのアプリケーションの増加に伴い, IA32におけるSSEやSSE2命令[1]に代表されるように, ベクトル命令を強化したマイクロプロセッサーの採用が進んでいる. 元来, ベクトル命令はスーパーコンピューターなどの科学技術計算の分野で活用されてきたが, アプリケーションの高度化に伴い, PC用マイクロプロセッサーにおいても標準的機能となりつつある. また, このような傾向は, PCのみならず, 組み込みシステムや, サーバーの分野でも広まりつつある[2].

IBMは, ソニー株式会社グループ様, 株式会社東芝様と共同でCell Broadband Engine™（CBE）と呼ばれる次世代マイクロプロセッサーを開発した. これは, 64bit Power Architecture™を基に, 8個のSynergistic Processor Element（SPE）と呼ばれるSIMD（Single-Instruction Multiple-Data）型演算装置をSoC（System on Chip）技術で統合したマイクロプロセッサーである. CBEは単体のマイクロプロセッサーとして, 200GFlops以上の高い浮動小数点演算を実現する[3].

デジタル・コンテンツの編集・制作, シミュレーションなどの応用分野では, 膨大な計算量とデータ量を必要とする. 実際, 単一のCPUを搭載したシステムを複数個並べ, ローカル・エリア・ネットワークで接続したクラスター・サーバーを構築することは原理的に可能である. しかし, この場合はプロセッサー間通信の帯域の狭さと遅延の増大によって, システムとしての

提出日：2005年08月31日　再提出日：2005年12月12日

性能向上が伴わないことが多い.

我々は, スケーラブル・ビデオ・サーバー[4]を開発した経験を基に, IBM Böblingen研究所およびIBM Researchと共同で, CBEを用いたブレード型のサーバー(CBE Processor based Blade Server:以下CPBS)を試作した. このサーバーは, IBM BladeCenter®の規格に基づき, スケーラブルな処理能力の向上を可能にすると共に, サーバーにおけるCBEの活用方法を評価するのが目的である.

本論文では, 筆者が代表してこのプロジェクト・チームの成果を報告する. その内容としては, このCPBSのアーキテクチャーを策定する上での設計指針を明確にし, システムの実装の考慮点を述べる. 特に複数のプログラミング・モデルをサポートするシステムの実現方法と考慮点を詳述する. 具体的には, 2章ではCBEプロセッサーの概要を述べ, 3章ではCBEのプログラミング・モデルを整理し, システム化の課題を示す. 4章では, CPBSの設計と実装に関する考慮点を議論し, 5章では, アプリケーションを用いてCPBSを評価した結果とその考察を行う. 特にIAサーバーでは負荷の大きいアプリケーションとして, 布のシミュレーションと生命情報科学でのデータ検索を例として, CPBSがどのような優位性を持つか検討する. 最後に6章で, まとめとしてCPBSの有効性を示し, 今後の課題を示す.

2. CBEアーキテクチャーの概要

2.1 設計思想

マイクロプロセッサーの性能は, 技術の進歩と共に飛躍的に向上してきた. しかし, 近年今までの技術改良では, 性能向上に限界が見えてきている. その限界を生む主な要因は, 消費電力の壁, メモリーの壁, そして周波数の壁と呼ばれる課題である[6]. CBEプロセッサーはこれらの壁に対して, それぞれ, マルチコア・アーキテクチャーの採用, 主記憶とキャッシュとローカル・メモリーの3階層メモリー・モデルの採用, そして深いパイプラインを可能にするソフトウェア中心の実行制御などの技術の導入により, 飛躍的な性能向上を目指したものである[7].

2.2 CBEプロセッサーの基本構造

CBEプロセッサーは, マルチコア設計を採り, 同一パッケージ内に, 1個のPPE(Power Processor Element)と呼ばれる64bit Power Architectureを基にしたマイクロプロセッサー, 8 個のSPEと呼ばれるCBEアーキテクチャー専用に設計されたSIMD型RISCプロセッサー, それにメモリー・コントローラー(MIC), およびI/O コントローラー(BIC)が納められている. そして,

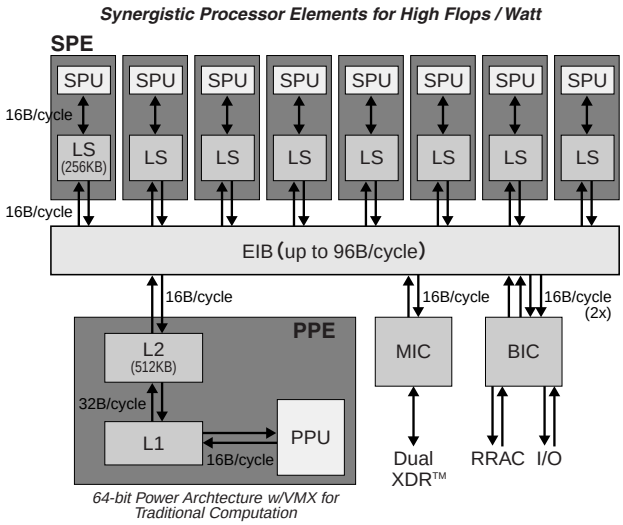

図1. CBEプロセッサーの内部構造[3]

各構成ユニットはElement Interconnect Bus(EIB)と呼ぶリング状のバスで結合されている. このプロセッサーの構成を**図1**に示す.

PPEは, 64ビットPowerPCコアを拡張し, 高速な動作周波数を達成すると共に, SIMD命令であるVector/SIMD Multimedia Extension Technology(以下VMX)のサポートや, ハードウェアによるスレッドサポート, さらにLPAR(Logical Partitioning)などの仮想化機能の導入などの先進機能を実現している.

各SPEは, それぞれLocal Store(LS)と呼ばれるローカル・メモリーと, 128本の128ビット幅のレジスター・ファイルを装備している. 1つのSPEは最大16WayのSIMDアクセラレーターとして動作する. すなわち, 各SPEはDMA経由でLSに対し16の同時アクセスを実行可能な構造を持つ. その浮動小数点演算の理論性能値は8GFlops/GHz(単精度演算)である. 動作周波数が3GHzの場合, CBEプロセッサー全体としてのピーク性能は, 8個のSPEとPPEの性能を合わせて, 200GFlops以上に達する.

3. CBEのシステム化の課題

前章で述べたように, CBEは高い演算性能を持つが, その一方でそのユニークなアーキテクチャーに起因するシステム化の課題がある. ここでは, ソフトウェアの面を中心に課題を整理する.

3.1 多様なプログラミング・モデルへの対応

PPEとSPEは, 命令セットや機能特性の面で大きく異なる. PPEは汎用(はんよう)的なプロセッサーとして, 一般的なワークロードに対応可能であるのに対して, SPEはSIMD命令に特化した最適化がなされている[8]. CBEのプログラミング・モデルを考えた場合には大別

して, PPE中心にプログラムが動作するモデル(PPE Centric Model：PCM)と, SPE中心にプログラムが動作するモデル(SPE Centric Model：SCM)の2種類が存在する[9].

PCMは, アプリケーションがPPEで実行され, 個々のタスクはSPEにオフロードされるモデルである. SPEの使い方の違いでさらに3種類のモデルに分類できる. 最初は, タスクが逐次的な処理を必要とする場合に, 各SPEは多段階のパイプラインとして動作するモデルである. 一方, アプリケーション内のタスクを分割して, 個々のタスクを並行に処理が可能な場合には, 複数のSPEを同時並行処理させるモデルが有効である. 最後のモデルは, ほとんどのアプリケーション・プログラムをSPEに割振り, 各SPEが主記憶とのデータの管理を含めて実行責任を持つモデルである. この場合, PPEは単純なリソース管理のみ行う.

一方, SCMは, SPEがアプリケーションの実行主体となり, PPEを介さずに各SPE間でデータ転送およびコードの制御を行うことになる. PPEは単純なリソース・サーバーとして機能する.

このように, CBEでは非対称な命令セットに起因して, アプリケーションに合わせて複数のプログラミング・モデルを持つことが可能なため, 各モデルに柔軟に対応することが求められる. **図2**に示すように, PPEおよびSPEのスレッドを仮想化して, 物理リソースにマップする仕組みが必要となる.

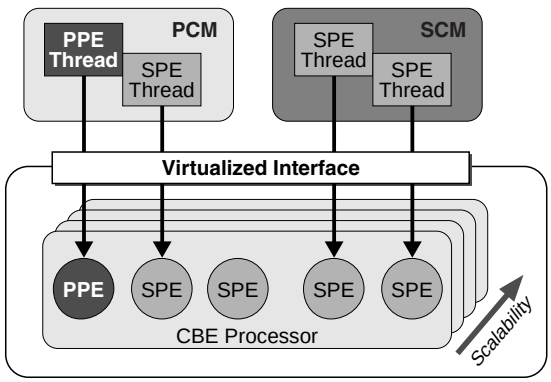

図2. 仮想化による多様なプログラミング・モデルへの対応

3.2 LSの制約

各々のSPEで実行するプログラムのアドレス空間が干渉しないように, SPEのLoad/Store命令はローカルなLSを対象とし, アドレス変換無しにアクセス可能にしている. LSと主記憶の間のデータ転送は各SPEに搭載されたDMAコントローラーによって行う. 従って, SPEの実行と並行して, 主記憶内のデータ転送をソフトウェアで制御する必要がある. また, 個々のLSの容量は256KBに限られるため, LS内にプログラムとデータを最適に配置すると共に, SPEがアイドル状態にならないように間断のないデータの供給のための工夫を必要とする.

4. CPBSの設計と実装

4.1 設計指針

我々は, 本システムの設計を行うに当たり, 以下に掲げる方針を採ることにした.

- CBEの処理能力を有効に引き出すため, メモリー帯域, 入出力帯域を最大限確保する.
- 幅広いアプリケーションに対応するため, シングル・プロセッサーおよびマルチプロセッサーの各々に対応可能な構成を採る.
- スケーラビリティを確保し, プログラムの変更なく, システムの追加でアプリケーションのワークロードの増加に対応可能とする.
- 市場において新しいアーキテクチャーが受け入れやすくするため, できるだけ既存の技術, 製品を採用する.
- オープンな環境に対応可能とするため, システム階層の各インターフェースを明確に定義して, 標準規格の採用および技術情報の公開を行う.

4.2 CPBSハードウェアの実装

4.1に掲げた指針により, 我々はIBM BladeCenterを採用することにした. CBEプロセッサーを搭載したブレードを新規に開発して, BladeCenterの筐体(きょうたい)の中に格納することにした. 既存デスクトップ, もしくはデスクサイドのマシンの採用も検討したが, サーバーとしてのスケーラビリティ実現の要請により, BladeCenterのパッケージを選択した.

1つのブレードには, **図3**に示すハードウェアの構成を採ることにした. すなわち, ブレード内は, CBEプロセッサーを2個搭載し, 2way SMP(Symmetric Multi-processor)構成を採用した. これは, プログラムの複雑化, 大規模化に伴い, 単体のCBEでは対応が難しい場合が予想されるためである. CBEプロセッサー

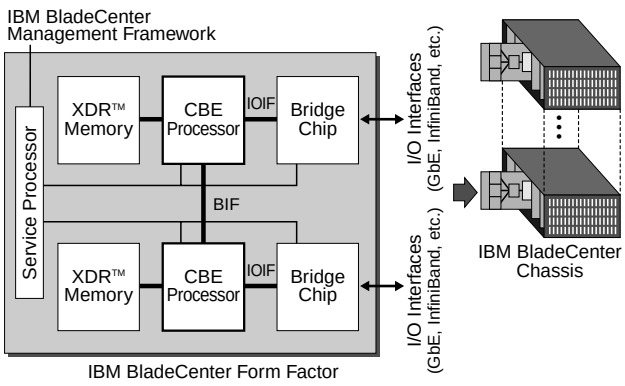

図3. CPBSブレードのハードウェア構成

のコヒーレントI/Oを提供するBIF（Broadband Processor Interface）を介して，2つのCBEプロセッサーを接続し，対称なマルチプロセッサー構成を取ることにした．これによって，各ブレードには，最大16個のSPEをアプリケーションから利用可能となる．

その一方，2つのCBEが各々独立して動作をするモードもサポートすることにした．これは，リアルタイム・アプリケーション［10］を想定した場合，単一のCPUとしてシステムのリソースを明示的に割り当て，予測可能な処理応答速度を提供することを考慮したためである．

メモリーに関しては，従来のDDRメモリーに比べ5倍以上高速なアクセスが可能なXDRTM DRAM［11］を各CBEプロセッサーのメモリー・インターフェースに接続した．現時点でブレード当たりのメモリー容量は1GBである．将来大容量XDRメモリーが利用可能になれば，この容量はさらに増加させることが可能である．

入出力に関しては，CBEプロセッサーのIOIF（I/O Interface）を通じて，Bridge Chipに接続した．このBridge Chipを経由して，PCI-Express，PCI，InfiniBand，Giga Bit Ethernet，USBなどの入出力インターフェースを持たせることにした．

さらに，BladeCenterの管理フレームワーク［12］の下で，各CPBSブレードを適切に管理対象とするため，Service Processorと呼ぶ管理専用のプロセッサーを各ブレードに搭載した．これによって，システムの構成管理，ブレード上の構成部品のモニターなどの機能を提供し，運用における可用性を高めることを可能にした．

本ブレードは，BladeCenter HS40と同様に，通常のブレードの2倍の幅を持たせた．これは，現行のBladeCenterの電源容量および放熱仕様に準拠するためである．従って，BladeCenterの筐体当たり1枚から7枚までのCPBSブレードが搭載可能となる．
同一のBladeCenter筐体内のブレードおよび，他の筐体のブレードとは，BladeCenterに標準で備わるGigabit Ethernetスイッチもしくは，外部に配置されたInfiniBandスイッチによって接続される．InfiniBandを介した場合には，より高帯域かつ低い遅延時間でブレード間における双方向の通信が可能となる．

4.3 ソフトウェアの実装

4.1の指針に従い，CPBSのソフトウェアは，BladeCenterと互換性を持たせ，かつ標準的な規格をできるだけ採用することにした．具体的には，**図4**に示すソフトウェア・スタックの構造を採った．すなわち，OSのロードとシステムの初期化や状態管理を行うFirmware（FW）と，Linuxカーネルおよびサービス機能群，そして，アプリケーション・プログラムに機能を提供するツールおよびライブラリー群を配し，その上にアプリケーション・プログラムが乗る構造にした．また，アプリケーション・プログラムに対しては，インターフェース（API），およびプログラムの交換規約であるApplication Binary Interface（ABI）を明確に定め，互換性の維持を実現した．

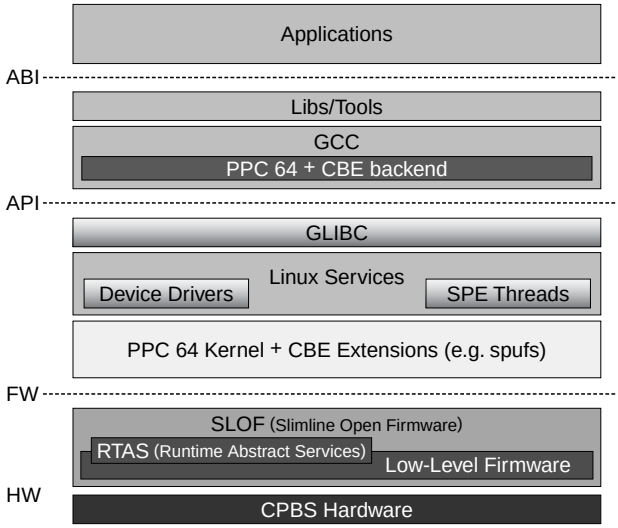

図4. CPBSのソフトウェア・スタックの構造

4.3.1 Firmwareの実装

FWのインターフェースとして，IEEE 1275の規格［13］に従ったSlimline Open Firmware（SLOF）を導入した．このSLOFの下に，ハードウェア構成や状態情報を提供するRuntime Abstraction Services（RTAS）と，CPBSハードウェア・コンポーネントの初期化を行うLow-Level Firmware（LLFW）を配した．

4.3.2 OSにおけるCBEのサポート

OSの階層では，64Bit PowerPC Linuxをベースにして，SPEおよびCBE固有の機能をサポートするための拡張を，カーネル・レベルおよびサービス・レベルで行なった［14］．特に，アプリケーション・プログラムからSPEに対して，どのようなアクセスを提供し，どのようなリソース管理を行うか，プログラムの性能と共に，新しいアーキテクチャーの普及にも影響を与える．そこで，カーネル・レベルでのSPEのサポートは，SPEごとにspufsと呼ぶ仮想ファイルシステム（VFS）を与え，仮想的なファイルとしてSPEのリソースにアクセス可能なインターフェースを設けた［15］．具体的には，LS，メールボックス，SPEレジスターなどを，仮想的なディレクトリとファイルとして扱えるようにした．一方，サービス・レベルのSPEのサポートは，このspufsの上に，タスクベースの抽象化を行った．PPEスレッド・プリミティブとしてSPEのプログラムを管理することに

した. SPEスレッドのプリミティブとして, DMA, イベント, メールボックスなどの機能を用意し, アプリケーションから容易に利用可能にするため, SPEに対するAPIとして機能を提供することにした.

これ以外のアプローチとして, デバイス・ドライバーによるSPEインターフェースの提供や, Linuxカーネルにおけるタスク管理システムをSPEに拡張することも検討したが, 仮想化およびPowerPCプログラムとの互換性の観点で採用には至らなかった.

4.3.3 開発ツールにおけるCBEの対応

CBEのアプリケーション・プログラムを効率良く開発するためには, SPEの命令セットに対応したコンパイラーおよびPPE/SPEの資源を活用するライブラリーの提供が重要な役割を担う. コンパイラーに関しては, C/C++言語規格からCBE固有の機能をサポートするための拡張インターフェース仕様[16]を定義して, GNU Cコンパイラー(GCC)およびIBM XL Cコンパイラーから SPEのコードを生成可能にした. また, ライブラリーに関しては, 基本的なシステム資源の管理だけでなく, スレッドの管理や, LSの操作, 数値演算, 行列演算, 他多くの機能をプログラミング言語から利用可能にした[17].

5. 評価と考察

5.1 布のシミュレーション

布をコンピューター・グラフィックスで実感的に表現することは, 複雑な数式の計算と, 大規模な行列計算を必要とするため, 一般的に従来のアーキテクチャーでは対応の難しい問題の1つであった. 特に, 布の変形過程を正確に表現するためには, 膨大な履歴情報を管理しながら行列演算と微分方程式を繰り返すため, リアルタイムに処理することは難しかった[18]. 今回, **図5**に示すように, Alias Systems社と共同で, 布のシミュレーションをリアルタイムで行うアプリケーションをCPBS上に試作した.

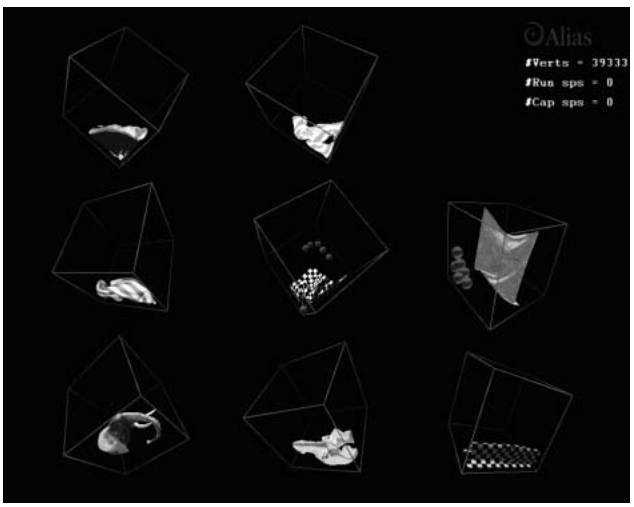

図5. 布のリアルタイム・シミュレーション結果[19]

CBEプロセッサーの8個のSPE各々に布のシミュレーションのインスタンスを割り当て, 実行した. プログラミング・モデルとしては, PCMの並行動作モデルを用いた. メッシュデータ構造において, 演算関数を実行するコードをベクトル化することで, 大幅なパフォーマンスの向上が可能になった. 動作周波数を同一条件にした場合, IA32プロセッサーに比べて, CBEでは約7倍以上の性能の向上が得られた. **図6**は, 本アプリケーションをSSE2命令を付加したIA32で実行した場合,およびCBEでSPEの数を増やした場合におけるシミュレーションのフレーム速度を基準にした処理速度の向上を示している. このグラフから分かるように, SPEの増加によって処理速度は線形で向上する結果が得られた[19]. ただし, 1つのSPEの場合IA32の1.8倍を示すが, 8SPEの場合には, 約7.5倍の性能向上に留まった. 単純計算では8SPEの場合, 約14倍の向上が期待されるが, SPEとPPEの間の制御およびデータ交換のオーバーヘッドが起因したものと考えられる.

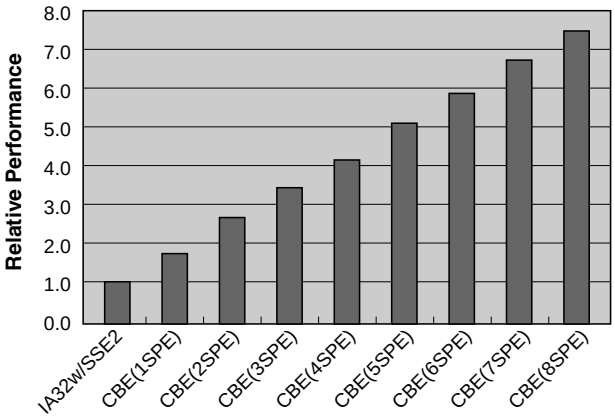

図6. SPEによるシミュレーション速度向上の寄与[19]

また, CPBSブレード上の2つのCBEプロセッサーを用いて, 16個のSPEにシミュレーションのインスタンスを同時に稼動させ, 正しく動作することが確認できた.

5.2 検索アプリケーションにおける評価

High Performance Computingの分野の1つに, 生命情報の検索がある. 大量の遺伝子の塩基配列やタンパク質のデータから, 類似性を見いだすことで, 新たな知見を得ることが盛んに行なわれている. この中で, Smith-Waterman [20]と呼ばれるアルゴリズムは, このような検索に一般的に用いられている. このアルゴリズムは, **図7**に例示するように, 検索する塩基配列のパターン(A-C-Q-R-X-K...)とデータベースに格納されているデータ列(A-C-Q-L-P-X-S-R-X-K...)

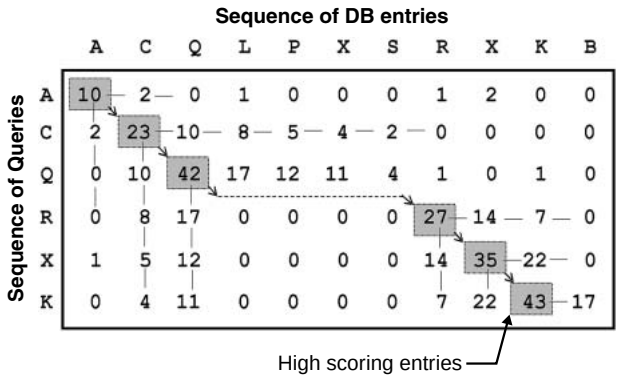

図7.　Smith-Waterman アルゴリズムの概念

の中で高いスコアのパターンから類似性を見出し，該当の塩基の特性を判別するものである．しかし，検索対象のデータの増加に伴い，スコアの判定を行う部分の処理が増大し，処理時間の長大化が生じる．

現在，このアルゴリズムはSIMD命令を用いたベクトル化により，高速化が図られることが知られており［21］，筆者らは，今回このアルゴリズムをCBEの複数のSPEにマップして，処理時間の向上を調べた．塩基配列のデータ列を，複数のSPEで判別し，判定結果をPPEに返すモデルを用いた（**図8**参照）．

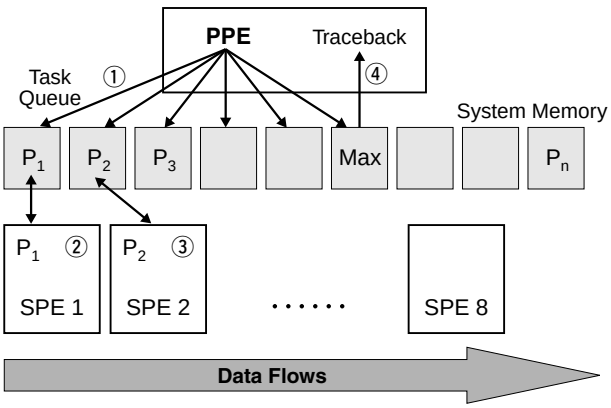

図8.　Smith-Waterman アルゴリズムのCBEへの適用

この結果，CPBSではIA32に比べ約30倍，SSE2を用いたIA32と比べても約6倍の処理速度の高速化が図れた．これはSPEのSIMD命令が有効に働いただけでなく，SPE間の処理の並列化により，データのパイプラインが間断なく行なわれたためであると思われる．この測定結果を，相対処理能力として示したのが**図9**である．

これにより，複数のSPEを持つCBEの構造は，浮動小数点を主体とした数値計算だけでなく，データ解析や検索の分野においても，CBEの並列性を有効に活用可能であることが分かった．

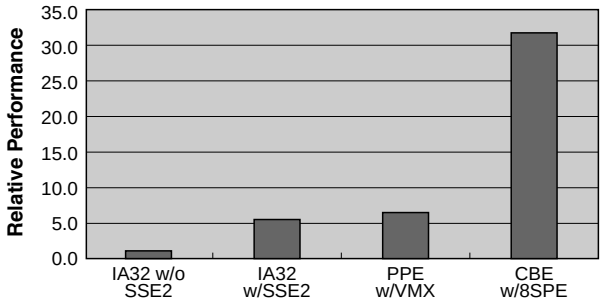

図9.　Smith-Waterman によるSPEの効果

6．おわりに

CBEは，従来のマイクロプロセッサーにはない高い演算処理能力を持つ，新しいアーキテクチャーのマイクロプロセッサーである．この能力を引き出すためには，SPEをいかに有効に動作させるかが鍵となる．我々は，このCBEを用いてIBM BladeCenterの規格に準拠したブレード・サーバーCPBSを試作した．この目的は，デジタルメディアや他の分野における負荷の大きなワークロードをCPBSで処理することで，アプリケーション全体の高速化を図り，CBEの適用分野を検討する点にある．今回，CBEをSMP構成にしたブレードを新たに開発した．このブレードのハードウェアは，XDRメモリーおよび高速インターコネクト接続機能を搭載し，CBEの高い性能を生かす設計にした．また，ソフトウェアとして，Linux OSを拡張し，CBEのリソース，特にSPEをプログラムから容易にアクセス可能にする機構をカーネル・レベルおよびユーティリティー・サービスの階層で実現した．このような対応の結果，CPBSは標準的なPowerPCのブレードと同等に使えるようになり，加えてLinuxの上でSPEの高速処理が利用可能なシステムとなった．

CBEは，汎用的なPowerPCとSIMD処理に特化したSPEの組み合わせで構成されており，非対称な命令セットを持つマルチプロセッサー構成になっている．限定されたLSのサイズにSPEのプログラムとデータを格納し，またPCMもしくはSCMのプログラミングモデルの適切な選択という，従来のSMPシステムにはないプログラミングの考慮をしなくてはならないが，その課題を克服することにより，今までのシステムでは処理困難なアプリケーションを切り開くことが期待される．今回我々は，様々なアプリケーションのワークロードをこのCPBSにマップして，その有効性を評価した．従来リアルタイム処理が困難であった布のシミュレーションは，SSE2命令を持つIA32のシステムに比べて7倍以上の性能向上が見られた．また，データ検索のアプリケーションでは，同IA32のシステムに比べて，約6倍の向上が得られた．複数のSPEを活用することでCBE

の高速な処理を具現化することができた.

今回, コンパイラーの未整備で他の命令セットとの詳細な比較検討や複数のCPBSブレードを組み合わせた性能評価には至らなかった. しかし, 今後は複数のブレードを組み合わせたシステムの構築を行い, 大規模アプリケーションを対象とした評価を進めたい. これからもCPBSの評価を続け, CBEの適用分野の明確化を検討すると共に, MPI（Message Passing Interface）[22]やOpenMPなどの並列処理のためのプログラミング環境をSPEに拡張し, 広範なアプリケーション開発を容易にするソフトウェア環境の拡充を図って行きたいと考える.

謝辞

本システムの技術検討と試作は, IBM Böblingen研究所, IBM Research, STI Design Centerとの緊密な協力の中で行なわれた. この活動に携わった, エンジニア, 研究者そしてマネージメントの各位の協力とサポートに感謝する. 特に, Lisa T. Su, Robert E. Hanson, James A. Kahle, Michael N. Day, Peter H. Hofstee, Robert Putney, Theodore R. Maeurer, Wen-Tzer T. Chen, Gottfried Goldrian, Roland Seiffert, Otto Wohlmuth, Albert Kopp, Marcus Breuer, James R. Moulic, Bruce D'Amora, Ashwini Nanda各位の多大な尽力に感謝する. また, パートナーとして支援頂いた, 株式会社ソニー・コンピュータエンタティンメント, 株式会社東芝, Alias Systems Corp.社の皆様の絶大なるサポート無しには, システムの実現や評価は不可能であった. 改めて, このプロジェクトに関わった皆様に深謝する.

参考文献

[1] A. Klimovitski, "Using SSE and SSE2: Misconceptions and Reality", Intel Developer Update Magazine, March 2001 Issue, pp.3 - 8, 2001.
[2] K. Krewell, "Chips, Software, and Systems", Microprocessor Report, January 31, 2005, pp.1 - 3, 2005.
[3] D. Pham et al., "The Design and Implementation of a First-Generation CELL Processor", Proceedings of the 2005 IEEE International Solid-State Circuits Conference, pp.184 - 187, 2005.
[4] T. Sanuki and Y. Asakawa, "Design of video-server complex for interactive television", IBM Journal of Research and Development, Vol. 42, No.2, pp.199-218, 1998.
[5] B. Gibbs et al., "The IBM eServer BladeCenter JS20", IBM Redbooks Form No. SG24-6342-1, 2005.
[6] J. L. Hennessy and D. A. Patterson, "Computer Architecture - A Quantitative Approach 3rd edition", Morgan Kaufmann Pub., 2003.
[7] 鈴置雅一他,“ 9個のプロセッサを集積した次世代汎用MPUを開発" 日経エレクトロニクス No. 894（2005.2.28）, pp.111-117, 2005.
[8] IBM Corp., "Synergistic Processor Unit (SPU) Instruction Set Architecture Version 1.0", 2005 http://www.ibm.com/developerworks/power/cell/
[9] IBM Corp., "Cell Broadband Engine Programming Tutorial Version 1.0", 2005 http://www.ibm.com/developerworks/power/cell/
[10] J.W. Liu, "Real-time Systems", Prentice-Hall, 2000.
[11] Rambus Inc., "XDRTM DRAM System Design Overview" High Performance Memory Solution, Rambus, 2005.
[12] D. Watts et al., "IBM eServer xSeries and BladeCenter Server Management", IBM Redbook IBM Form No. SG24-6495, 2005.
[13] http://www.openfirmware.org/
[14] Barcelona Supercomputer Center (BSC), "Linux on Cell systems", http://www.bsc.es/projects/deepcomputing/linuxoncell/
[15] A.Bergmann, "Spufs: The Cell Synergistic Processing Unit as virtual file system", IBM DeveloperWorks, 2005. http://www.ibm.com/developerworks/library/pa-cell/
[16] IBM Corp., "SPU C/C++ Language Extensions Version 2.0", http://www.ibm.com/developerworks/power/cell/
[17] IBM Corp., "Cell Broadband Engine SDK Libraries Overview and Users Guide Version 1.0", 2005 http://www.ibm.com/developerworks/power/cell/
[18] H.N. Ng and R.L. Grimsdale, "Computer Graphics Techniques for Modeling Cloth", Computer Graphics and Applications, Vol.16, No.5, pp.28-41, 1996.
[19] Alias Systems Corp., "Alias Cloth Technology Demonstration for the Cell Processor Whitepaper" Issued by Alias Systems Corp., May 2005.
[20] T.F. Smith and M.S. Waterman, "Identification of Common Molecular Subsequences", Journal of Mol. Biol., Vol.147, pp.195-197, 1981.
[21] F. Sanchez et al., "Parallel Processing in Biological Sequence Comparison Using General Purpose Processors", Proceedings of the 2005 IEEE International Symposium on Workload Characterization, pp.99-108, 2005.
[22] http://www-unix.mcs.anl.gov/mpi/

日本アイ・ビー・エム株式会社
Senior Technical Staff Member
大和システム開発研究所

佐貫 俊幸 Toshiyuki Sanuki, Ph.D.

［プロフィール］
1983年日本アイ・ビー・エム株式会社サイエンスインスティチュート（現東京基礎研究所）入社．プログラミング言語の研究，大規模Video On Demand（Vod）やコンテンツ配信などのデジタルメディア・システムの構築，サーバー製品の開発などに従事．1997年米国IBMに出向．ソフトウェアならびにソリューションの技術企画を担当．2000年帰国後，エマージング・ビジネス（EBO）を推進．現在，CBEおよび次世代マイクロプロセッサー応用システムの企画，開発を担当．情報処理学会，IEEEの各会員．