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

2006－HPC－107（18）
2006／8／1

Cellプロセッサへの分子軌道法プログラムの実装と評価

林 徹生[†]　本田 宏明[††]　稲富 雄一[††]
井上 弘士[†††]　村上 和彰[††,†††]

[†] 九州大学大学院 システム情報科学府　〒816-8580 福岡県春日市春日公園 6-1
[††] 九州大学 情報基盤センター　〒812-8581 福岡市東区箱崎 6-10-1
[†††] 九州大学大学院 システム情報科学研究院　〒816-8580 福岡県春日市春日公園 6-1
E-mail : *hayashi@c.csce.kyushu-u.ac.jp*

概要　今日に至るまで種々のプロセッサ・アーキテクチャが提案され，プロセッサの計算性能は著しく向上している．現在では 1 個のチップに複数のプロセッサコアを搭載することで性能向上を図るチップマルチプロセッサ (CMP) が数多く提案されるに至っているが，高い計算性能を誇る Cell プロセッサもその一つである．また，CMP チップの用途として主にメディア処理が想定されているが，その高い計算能力を生かすことで分子軌道法計算等の科学技術計算にも利用可能と考えられる．そこで本研究では Cell プロセッサに分子軌道法計算の主たる計算部分である二電子積分計算を実装し，その性能を評価する．また，分子軌道法計算のような科学技術計算へ対する今後の CMP チップの利用可能性を考察する．
キーワード 分子軌道法計算，高性能プロセッサ，CMP，並列処理

Implementation and Evaluation of A Molecular Orbital Calculation Program on Cell Processor

Tetsuo HAYASHI[†], Hiroaki HONDA[††], Yuichi INADOMI[††],
Koji INOUE[†††] and Kazuaki MURAKAMI[††,†††]

[†] Graduate school of Information Science and Electrical Engineering, Kyushu University
[††] Computing and Communications Center, Kyushu University
[†††] Faculty of Information Science and Electrical Engineering, Kyushu University

abstract　As various architectures of processor are proposed until today, the processor performance improves remarkably. Now many chip multiprocessors that planed to improve performance by implementing some processor cores on a chip are proposed, and processor “Cell” is one of them. Though the media processing is mainly assumed as a usage of the chip, we think that we can apply their high performance to Science and Technology calculation like Molecular Orbital(MO) calculation. In this paper, we implement Two Electron Integral calculation that is core of MO calculation on Cell processor, and evaluate performance. And we consider the use possibility of chip multiprocessor for Science and Technology calculation like MO calculation.
keyword　Molecular Orbital calculation, high performance processor, CMP, parallel processing

1　はじめに

近年，1 個のチップに複数のプロセッサコアを搭載するチップマルチプロセッサ (CMP) が数多く提案されている．依存関係のない処理を各プロセッサコアで並列実行することにより性能を向上できる．特に，2005 年に開発された Cell プロセッサは，ゲーム機等のメディア処理に重きを置いた設計ではあるものの，ハイエンドな高性能汎用プロセッサと比較して極めて高い処理能力を有している [3]．例えば，同じプロセステクノロジである Pentium4 と比較した場合，Cell プロセッサは表 1 に示すようにピーク性能 (4GHz 動作時の単精度浮動小数点演算性能) は約 10 倍である．

一方，高いプロセッサ性能を必要とするアプリケーションとしては，メディア処理だけでなく，物

表 1: Cell と Pentium4 の性能比較

		Cell	Pentium4
トランジスタ数（個）		2億3400万	1億2500万
面積（mm²）		221	112
動作周波数（Hz）		4G（最大4.6G）	3.8G
ピーク性能（GFLOPS）	単精度	256（SPE×8）	15.2
	倍精度	26（SPE×8）	7.6

理現象の予測や解明，新技術の開発・制御などに用いられる科学技術計算がある．ここで，代表的な科学技術計算である非経験的分子軌道法 (以下，分子軌道法) に着目する．分子軌道法においては，タンパク質 (分子量数千〜数百万) 等の巨大分子を扱う場合，現在の高速な計算処理能力をもってしても非常に多くの計算時間が必要となる．例えば，従来の汎用プロセッサにおいてリゾチーム (分子量 14,307) に対して精度良く計算すると，およそ 27 年の計算時間を要するといった報告もある [1]．したがって，巨大分子をシミュレーション対象とする場合はより高性能な計算能力が必要である．この問題を解決する 1 つの方法として，前述した高性能 CMP である Cell を活用することが考えられる．しかしながら，Cell はメディア処理に重きを置いているため，分子軌道法プログラムを効率的に実行できるか否かは明らかにされていない．

そこで本研究では，Cell プロセッサに対して分子軌道法プログラムを実装し，性能を評価する．また，Cell アーキテクチャが有する特徴がプロセッサ性能へどのような影響を与えるか解析を行う．具体的には，複数のプロセッサコアに対するタスク分配を行い，分子軌道法プログラムを 8 個のプロセッサコアで並列実行する．更に，分岐予測やアラインメント制御など，従来ハードウェアで制御してきた処理をソフトウェアで行うことが性能に与える影響を示す．そして，より高性能化を実現するための課題を明らかにする．

以下，Cell プロセッサへの分子軌道法プログラムの実装と評価に関する構成を述べる．まず，第 2 節で Cell プロセッサのアーキテクチャについての説明を行い，続いて，第 3 節で分子軌道法プログラムと Cell への実装方法を示す．第 4 節で実験結果と考察を述べ，最後に第 5 節でまとめる．

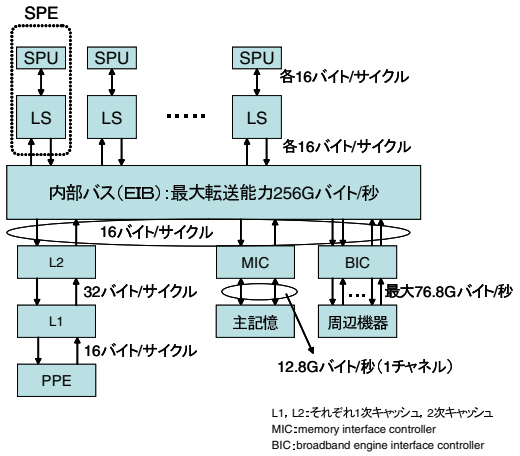

図 1: Cell のブロック図

2 Cell アーキテクチャ

2.1 内部構成

Cell には 9 個のプロセッサコアが搭載され，1 個の制御プロセッサ PPE(Power Processor Element) と 8 個の演算プロセッサ SPE(Synergistic Processor Element) から構成される．Cell の内部構成を図 1 に示す．各 SPE は 256K バイトのスクラッチパッドメモリ (プログラマが明示的に制御可能なオンチップメモリ) を内部に持つ．それぞれのプロセッサコアは時計回り 2 本，反時計回り 2 本のリング型内部バス EIB(Element Interconnect Bus) で接続される．

2.2 特徴

Cell アーキテクチャの主な特徴は，1）8 個の演算プロセッサによるスレッドレベル並列性，2）徹底したハードウェアの簡素化による消費電力の削減，が挙げられる．特に，後者により最大 4GHz という高周波数での動作を可能としている．以下，ハードウェアの簡素化を目的とした主なアーキテクチャ上の特徴をまとめる．

- ソフトウェア分岐予測の採用
  多くの汎用プロセッサはハードウェアに動的分岐予測機を搭載しており，90 %以上の確率で分岐予測が的中する．しかしながら，命令フェッチ毎に分岐予測を実施するため多くの電力を消費する．これに対し，SPE ではソフトウェア分岐予測を採用しており，回路面積の削減と低消費電力化を実現している．しかしながら，一般にソフトウェア制御分岐予測は動的分岐予測機に比べ予測精度が低下する

ためプロセッサ性能に悪影響を及ぼす可能性がある．

- ソフトウェア制御オンチップメモリ
  最近の高性能汎用プロセッサでは，ほとんどの場合においてオンチップ・キャッシュが搭載されている．そのため，プログラマはキャッシュメモリへのデータ転送を意識することなくプログラムを作成できる．しかしながら，オンチップ・キャッシュサイズの増加に伴い，その消費電力が問題となっている．これに対し，Cell において各 SPE は LS(Local Store) を搭載している．プログラマは明示的に DMA(Direct Memory Access) を制御し，LS とメモリ間でのデータ転送を実現する．SPE 間で直接は同期を取らず，独立に動作する．したがって，SPE におけるプログラマブルなローカルメモリを最大限利用した計算を実現できる．

- アラインメントを取らない LS
  従来の汎用プロセッサでは，メモリからのデータ読み出し，またはデータ書き込みにおいて，ハードウェアでアラインメントを取っていた．一方，Cell では 128 ビット単位で Load/Store を実行しているが，ハードウェアでのアラインメント制御を省略してアクセス時間を削減している．代わりに，置換・シャッフル命令などソフトウェアで制御をする．

3 分子軌道法プログラムの実装

3.1 分子軌道法プログラムの特徴解析

表 2 に 5 種類のペプチド分子を用いたときの分子軌道法計算における各ステップ別計算時間を示す [2]．多くの計算時間が必要となる分子軌道法計算だが，表 2 が示すように，全計算時間の 95 %以上を占めるのが二電子積分計算とフォッグ行列の生成ステップである．そして，二電子積分計算とフォック行列計算の計算時間は共に基底関数の数 N の 4 乗に比例する．したがって，計算の対象分子と精度にも依存するが，巨大分子系に関しては二電子積分計算を如何にして速く解くかが分子軌道法計算の高速化につながる．

二電子積分計算のアルゴリズムはいくつか知られているが，本論文では高精度かつ高速な新・小原のアルゴリズムを採用する [4]．図 2 において新・小原のアルゴリズムは外側 4 重ループの内にある

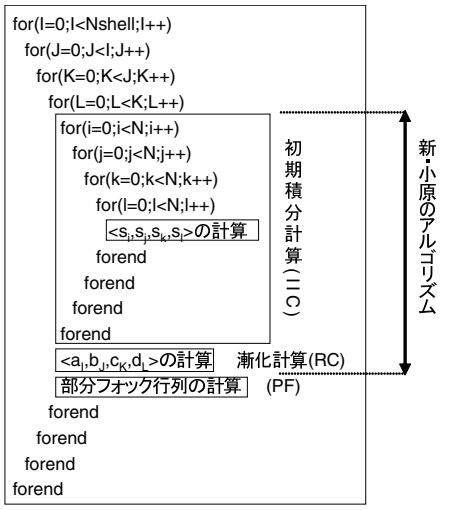

図 2: 新・小原のアルゴリズムを使用したフォック行列の計算

初期積分計算 (IIC:Initial Integral Calculation) と漸化計算 (RC:Recurrence Calculation) で構成されており，同じループ階層で部分フォック行列の計算 (PF:Partial Fock) が行われる．また，図 2 の全ての計算により全フォック行列が生成される．

初期積分計算は図 2 が示すように内部に 4 重のループ構造を持つ．その特徴としてループ 1 回あたりの演算数は非常に少なく，命令レベルの並列性も低い事が挙げられる．更には浮動小数点除算，開平逆数演算，指数関数演算, 及び誤差関数演算など多くの複雑な計算を $< s_i, s_j, s_k, s_l >$ に含み，かつそれがクリティカル・パス上に存在する．

一方の漸化計算の $< a_I, b_J, c_K, d_L >$ は，非常に高い並列性を持つ多数の積和演算命令のみから構成される．そのため，複数の積和演算回路を用いた並列計算による高速化が期待できる．

3.2 二電子積分計算の Cell への実装方針

Cell の CMP 構造を生かすにはタスク分配が重要となる．CMP 全体として高い性能とするには，SPE 間の通信を削減し，かつ，各 SPE の総処理時間を平均化する必要がある．

そこで本研究では，タスク分配の粒度については同期処理がほとんど発生しない観点から選択した．具体的には IIC と RC，PF を 1 つのタスク分配のための粒度とした．また，タスク分配の方法としては PPE-SPE 間通信が少ないことを優先事項と設定した．以上に従い，表 3 に図 2 を参考にしたタスク分配法の評価を示す．RC は並列性が非常に高いた

表 2: 5 種のペプチド分子におけるステップ別計算時間の分布

Types of Peptide Molecules	G	GA	GAQ	GAQM	GAQMY
No. of Atoms	10	20	37	58	75
No. of Atomic Orbitals	55	110	207	316	427
	Computation Time (sec.)				
Setup	0.1	0.1	0.1	0.1	0.3
Initial Orbital Set	0.1	0.6	4.4	18.9	57.3
ONE-EI Calculation	0.1	0.3	1.5	5.0	10.1
TWO-ERI Calculation & Fock-Matrix Generation	22.9 (96.6%)	269.4 (98.7%)	1871.0 (98.6%)	8482.1 (98.8%)	23284.3 (98.6%)
Fock-Matrix Diagonalization	0.2	1.7	11.0	60.9	211.7
Property Calculation	0.1	0.3	2.2	9.2	27.5
Total CPU Time	23.7	272.9	1892.7	8584.9	23614.5
No. of Iterations	12	14	15	16	19

[a]4-31G basis function set, using GAMESS, on Pentium-III@500MHz with 512MB Main Memory

表 3: タスク分配の選択肢

IIC	RC	PF	評価	備考
PPE	PPE	PPE	×	逐次処理
		SPE	×	RCの並列性を生かせない
	SPE	PPE	△	ERIC[1]の方針
		SPE	△	中間積分値転送の必要性
SPE	PPE	PPE	×	RCの並列性を生かせない
		SPE	×	
	SPE	PPE	○	PPEが部分フォック行列計算
		SPE	○	SPEが部分フォック行列計算

め，SPE で計算する．また，RC に必要なデータは IIC で生成される．したがって，PPE-SPE 間の通信量を削減するため，IIC も SPE で計算する．残る PF については，LS サイズの制約を考慮に入れ PPE で計算させる．

3.3 処理の流れ

PPE は自身のメモリ上に各 SPE に対応する同期等のプログラム通信用の領域を確保する．SPE に渡す引数やプログラムはこの領域に格納され，同じく格納されている flag の値によって SPE の状態をプログラム制御して計算を進めていく．PPE から関数呼び出しの形でタスク分配された SPE は，DMA を用いてメモリからデータを受け取り処理を開始する．flag の値は PPE，SPE 共に書き換えることができる．尚，SPE の遷移状態は図 3 のようになっている．

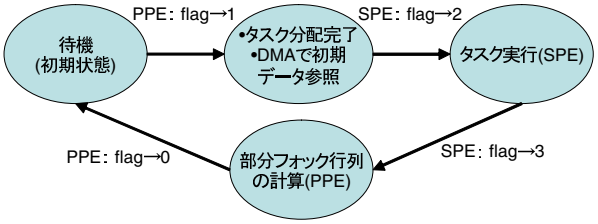

図 3: flag の値と SPE の遷移状態

4 性能評価

4.1 評価対象モデルと実験環境

Cell に分子軌道法計算プログラムを実装して性能を評価する．評価対象モデルは以下の通り．

- **BASE**:従来の Cell.
- **PBP(Perfect Branch Predection)**:BASE モデルにおいて，分岐予測ヒット率は 100 %と仮定したモデル.
- **EDP(Extended Double Process)**:BASE モデルにおいて，倍精度浮動小数点演算 (double) 性能が単精度浮動小数点演算 (float) 性能と等しいと仮定したモデル．実装上は，double で宣言している定数や変数，関数を float に置き換えている．
- **EDP+PBP**：EDP と PBP を組み合わせたモデル.
- **PentiumD**:動作周波数 3.2GHz の PentiumD プロセッサ.

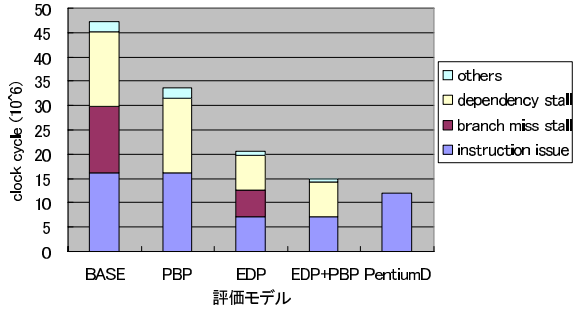

図 4: 項目別の全クロックサイクル数と処理の内訳

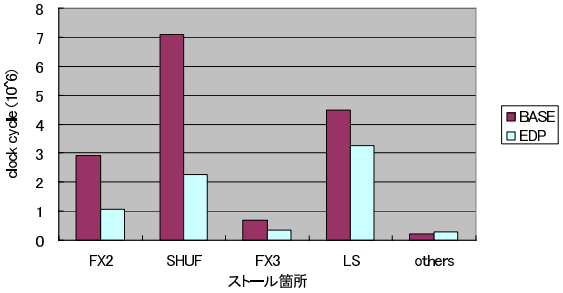

図 5: dependency stall のストール箇所

本実験ではシミュレータ mambo を用いた [6]．また，コンパイラには mambo と共に提供される ppu-gcc と spu-gcc(コンパイラオプション：-O2 -g) を用いた．分子軌道法プログラムの実行には，仮想上の分子である C_4(全ての積分計算タイプに対して計算を行う) を入力データとした．基底関数は DZV(4-31G と同カテゴリ)，基底関数の数 N は 20 とした．一方，PentiumD の評価環境 (対象分子以下は全て Cell と同じ) では，コンパイラに gcc(-O2 -g)，性能評価ライブラリに PAPI を用いた．

4.2 結果

実験結果を図 4 に示す．図中において，各棒グラフの内訳は以下の通り．

- **instruction issue**：全命令の実行に要する総クロックサイクル数 (ストールサイクルを除く)．
- **branch miss stall**：分岐予測ミスに起因するストールサイクル数．
- **dependency stall**：データ依存に起因するストールサイクル数．
- **other**：その他の理由によるストールサイクル数．

まず，BASE と PentiumD モデルを比較する．BASE モデルでは，分岐予測ミスとデータ依存によるストールサイクル数が実行時間のそれぞれ 1/3 程度ずつを占めている．この結果，PentiumD モデルに比べ 4 倍もの実行時間を要している．BASE と PentiumD モデルのピーク性能を考えると，BASE モデルの方がストールサイクル数を除いても遅い．その主な原因は，大容量レジスタを用いたループアンローリングといった Cell の特徴に合ったコード最適化が不十分であるためだと考えられる．

次に，ソフトウェア分岐予測による性能への影響を議論する．BASE と PBP モデルを比較すると，BASE モデルでは 60 %程度の分岐予測ヒット率であった．分岐予測ヒット率を向上させる (PentiumD モデルにおけるヒット率 99 %と同等にする) ことが可能ならば，PBP モデルのように約 30 %の計算性能の向上が得られることが分かる．

倍精度浮動小数点演算性能に関しては，BASE と EDP モデルを比較する．EDP モデルは BASE モデルに比べ，各内訳に対するクロックサイクル数が半減されている．倍精度/単精度の浮動小数点演算の演算レイテンシ (加算や乗算でそれぞれ 13/6 clock cycle) に比例した instruction issue と branch miss stall の削減を実現している．一方，dependency stall に関しては前者に比べてクロックサイクル数の削減率がやや劣る．

そこで，図 5 に示す BASE と EDP モデルにおいて，データ依存がパイプライン処理のどこで起こっているか (ストール箇所) に着目する．FX2 と FX3 は浮動小数点計算であり，SHUF は Load 命令直後または Store 命令の直前に発行されレジスタに対するデータの並び替えを処理し，LS は Load/Store 命令を処理する．今回用いたアルゴリズムは中間積分を保持するため Load/Store 命令が多く [5]，図 5 が示す通り Load/Store 命令に関係するデータ依存ストールが多く現れている．FX2 と FX3 は倍精度/単精度の浮動小数点演算の演算レイテンシが直接影響するため，instruction issue と同様にレイテンシに比例したストールサイクル数の削減を示している．

一方，SHUF と LS に関しては，ソフトウェアでアラインメントを取っている事が関係していると考えられる．具体的なコードを見ると，Load/Store 命令の発行時に，マスク生成命令，0 との論理和，回転命令，シフト命令，シャッフル命令のいずれか，また

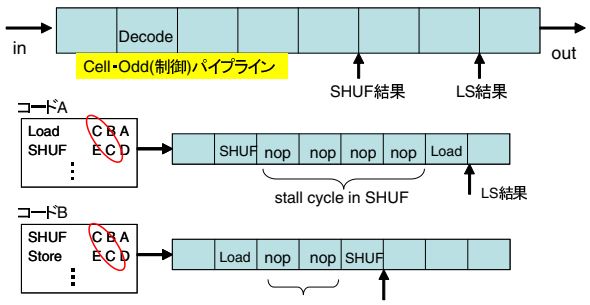

図 6: LS と SHUF におけるレイテンシのイメージ

は複数とペアを組んでアラインメントを取っていた．この Load/Store 命令とこれらの SHUF 命令のレイテンシの差 (Load/Store の方が長い) が SHUF と LS のストール時間削減量の差に現れていると考えられる．図 6 にレイテンシの差のイメージ図を示す．EDP モデルでは BASE モデルに比べ SHUF 命令が少なくなると考えられる．その結果，図 6 のコード A のような SHUF におけるストールサイクルは減少する．一方，Load/Store 命令数に変化はほとんどないため，コード B のようなストールサイクルの削減はあまり期待できない．したがって，Load/Store 命令がネックとなり，極端な dependency stall の削減は困難である．

最後に PBP+EDP と PentiumD モデルについて述べる．両者を比べると，PentiumD モデルの方がクロックサイクルが短い．前述したように，dependency stall をこれ以上削減するのは難しい．一方で，instruction issue は SIMD 演算を組み込むこと等により削減を期待できる．この結果，PentiumD モデルの計算時間を超えることも可能である．

5 おわりに

本稿では Cell プロセッサに二電子積分計算を実装して性能を評価した．分岐予測ヒット率に着目したモデルに対して評価し，ソフトウェア制御の分岐予測ヒット率の向上に伴う性能向上の可能性を示した．また，データ依存によるストール時間の差異に対して原因の解析を試みた．

本研究では，Cell がハードウェアの簡素化のために分岐予測とアラインメントの制御をソフトウェアで行うことを述べた．そして，コンパイラだけでは隠せなかった簡素化の影響が，ストールという形で実行時間に現れることを示した．また，分子軌道法計算のように依存性の高いデータを内部に持つ計算をする際，Load/Store 命令の削減が重要であることを示した．

今後の課題としては，ソフトウェア制御分岐予測に関連して用いた関数を変更するなどして，そのヒット率の変動とおよその上限を調べる必要がある．また，データ依存を減らすべく SIMD 演算の適応を含めたコード最適化を行う必要がある．また，今後は実際の Cell を使用してタスク分配法やネットワーク部分を含めたチップ全体としての性能を評価する予定である．

謝辞

本論文をまとめるにあたり，共にご議論頂いた九州大学システム LSI 研究センターならびに安浦・村上・松永・井上研究室の皆様に感謝します．なお，本研究は一部，科学研究費補助金 (学術創成研究費:課題番号 14GS0218，若手研究 A:課題番号 17680005) による．

参考文献

[1] 中村健太，“計算機科学専用プロセッサアーキテクチャに関する研究”，九州大学大学院 博士論文，2006 年 4 月

[2] 中村健太，本田宏明，梅田宏明，小松晴信，村上和彰，“分子軌道計算専用計算機用 LSI(ERIC) の開発”，The Journal of Computer Chemistry, Japan，Vol.4，NO.4，pp.155-164(2005)

[3] 村上和彰，井上弘士，吉松則文，“アーキテクチャ・徹底的なソフト化で性能／電力を向上”，日経マイクロデバイス，No.237，pp.44-47，2005 年 3 月

[4] H.Honda，T.Yamaki，and S.Obara，“Molecular integrals evaluated over contracted Gaussian functions by using auxiliary contracted hyper-Gaussian functions”，J.Chem.Phys，ll7，pp.1457-1469(2002)

[5] 稲富雄一，小原繁，長嶋雲兵，“2 電子積分計算ルーチンの性能評価”，the journal of computer chemistry，japan，vol.4，no.4，pp.175-178(2005)

[6] http://www.bsc.es/projects/deepcomputing/linuxoncell/cbexdev.html

[7] Sony Computer Entertainment Inc，“CBEA JSRE Series: SPU C/C++ 言語拡張”，公開資料