2007－ARC－174 (8)
2007／8／1

ClearSpeed製SIMD型マルチコアプロセッサにおける並列アプリケーション実行時間予測手法の検討

西川由理[†] 鯉渕道紘[††] 吉見真聡[†]
三浦謙一[††] 天野英晴[†]

本研究報告では，ClearSpeed社の開発したコプロセッサCSX600上での並列プログラムの実行時間を，演算時間と転送時間の評価結果に基づいて予測する手法の検討を行う．そのために，同社から提供されるSDK(Software Developement Kit)を用いて，各種演算命令および転送命令の実行時間を計測し，評価を行った．また，それら計測されたデータに基づき，演算と転送の時間を各々算出する式を導出し，また性能を維持しつつ計算を実行するための，演算と転送量のバランスを検討した．Monte Carlo積分と姫野ベンチマークを実行するClearSpeed向けプログラムを実装し，予測実行時間と実際の実行時間を比較したところ，演算時間は約1.60～3.62%程度の誤差で予測可能であることが明らかになった．

A Study of Time Prediction Method for Running Parallel Applications on ClearSpeed's SIMD-Based Multi-Core Processor

Yuri Nishikawa,[†] Michihiro Koibuchi,[††] Masato Yoshimi,[†]
Kenichi Miura[††] and Hideharu Amano[†]

This research focuses on a methodology for predicting execution time of parallel programs on ClearSpeed's CSX600 coprocessor. For this aim, performance of various arithmetic operations and bandwidth memory transfer operations was measured using an SDK (Software Development Kit) provided by ClearSpeed Technology. The measurements were analyzed to derivate formulae to estimate computation and data transfer time of parallel programs, and also to study a balance between calculation and data transfer. In order to examine the effectiveness of the formulae, following two parallel programs, the Monte Carlo Integration and the Himeno Benchmark, were implemented on CSX600 to compare the program execution time. As the result, execution time of a pure computation were estimated with an uncertainty between 1.60 - 3.62%.

1. はじめに

半導体製造プロセスの微細化に伴い，マイクロプロセッサの性能改善手法として，動作周波数の向上に頼る従来の手法が，性能と消費電力のバランスの点から限界を迎えつつある．そのため，近年では，複数の演算要素(コア)を用いて並列処理を行うことで，動作周波数を低く抑える性能向上手法が注目されてきている．2005年頃から登場したIntelのQuad-Coreプロセッサ(4コア)などが示した高い演算能力により，高周波数と大容量キャッシュによる従来のプロセッサに代わるアーキテクチャとして，マルチコアプロセッサが並列処理に有効であることが広く認識されるようになった．コア数の増加により演算能力の拡大を図る流れは今後も続くと考えられ，2010年には数十から数百個のコア(メニーコア)で構成されるプロセッサが実用化されると言われる．一方，多数の演算コアの動作による消費電力を低く抑えるため，SIMD技術のメニーコアプロセッサへの応用が注目されつつある．SIMD型プロセッサは内部の演算コアが同一の命令によって動作する．マルチメディア演算などデータレベルの並列性の高い処理に対し，単純化された制御機構によって優れた性能を発揮し，メニーコアプロセッサの電力性能比を高められると考えられている．

ClearSpeed社は科学技術計算，DSP，組み込みシステムなどの幅広いアプリケーションをターゲットとする高性能かつ低消費電力なSIMD型アクセラレー

† 慶應義塾大学大学院理工学研究科
Graduate School of Science and Technology, Keio University
†† 国立情報学研究所
National Institute of Informatics

タ CSX600 を製品化している．これは 1 個の制御用コアと 96 個の一次元アレイ状に接続された演算用コアを持つヘテロジニアスメニーコア型であり，特に行列積演算で優れた性能を発揮する．計算サーバとの協調処理を目的として開発された PCI-X アクセラレータボードは，CSX600 を 2 個と 1GB のメモリを搭載し，浮動小数点演算が必要なアプリケーションをコプロセッサ側で処理させることで消費電力の増大を抑えながら性能を上げ，システム全体の効率性を向上することを目標としている．このアクセラレータカードの採用例として，東工大のキャンパスグリッドの計算資源であるスーパーコンピューティングシステム TSUBAME[1] が挙げられる．アクセラレータボード 360 枚を一部のノードに接続することで，システム全体の消費電力増大は 1%程度に抑えつつ，性能が 25%向上したことが報告されている．Linpack では 47.38 TFLOPS の性能を記録しており，これは 2007 年 6 月の TOP500 において第 14 位，かつその性能は国内最高である．また，DGEMM，FFT (いずれも倍精度) などのベンチマークによる性能評価も行われており，それぞれ 25GFLOPS，10GFLOPS の性能を実現している[2]．

一方，著者らはこれまでに，ClearSpeed プロセッサのチップ内ネットワークや I/O 性能など，アーキテクチャの詳細を評価するため，姫野ベンチマークを用いた性能測定とプロファイリングを行っている[3][4]．この結果によると，プログラムを実行した性能は一般的な PC (Xeon 2.80GHz) に比べて約 2.5 倍程度高いものの，これは実効ピーク性能の 1/50 以下の性能である．また実行時間のプロファイリングの結果から，コア間，メモリ間通信の待ち時間が実行時間全体の 34〜50%を占める一方，PE の稼働率が 12%程度と低いことが明らかになった．この評価結果から，従来のプログラム実装と同様の方法では，特定のアプリケーション以外では性能が低く留まり，アプリケーションの特性に対する検討が不十分な点があると言える．

そこで，性能向上を阻害する要因を解析し，プログラムの実装方法による性能限界を示す必要がある．その端緒として，CSX600 上での並列プログラムの実行時間を，演算時間と転送時間の評価結果に基づいて予測する手法の検討を行う．CSX600 は SIMD 型であるため，特に演算時間は直接的に演算量をカウントすれば，各種演算命令の実効性能により概算できる．各種演算および転送性能を計測する方法として，同社が提供する SDK(Software Developement Kit) を用いる．またその結果に基づき，性能を維持しつつ計算を実行するための演算と転送量のバランスを明らかにするとともに，実行時間を概算する式の導出を試みる．さらに，その検証のため，Monte Carlo 積分と姫野ベンチマークを実行する ClearSpeed 向けプログラムを実装し，予測実行時間と実際の実行時間を比較する．

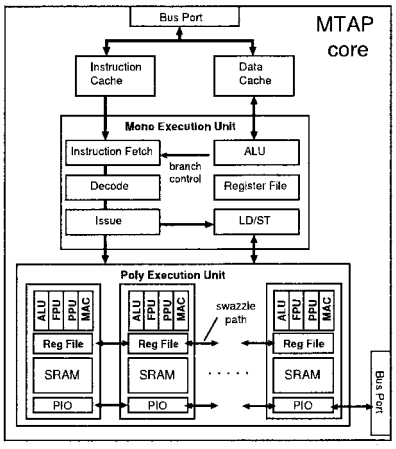

図 1　CSX600 の MTAP アーキテクチャ

以下，2 章において CSX600 とその開発環境の概要，および関連製品，関連研究を説明し，3 章にて，CSX600 の演算および転送性能の評価結果を示す．また 4 章にて，これらの実行時間を予測する手法について述べ，その実例を 5 章にて示す．最後に 6 章でまとめを述べる．

2. ClearSpeed CSX600

2.1 概　　要

ClearSpeed 社の CSX600 は，一次元アレイ状に接続された 96 個のプロセッシングエレメント (PE) で構成される SIMD 型コプロセッサである．理論性能は単精度または倍精度演算に対し 48GFLOPS，行列積の演算ルーチン DGEMM を用いた評価では，33GFLOPS の実効性能を有する[5]．

CSX600 の Multi-Threaded Array Processor (MTAP) の構造を図 1 に示す．96 個の各 PE はそれぞれ，単精度／倍精度浮動小数点乗算器および加算器，整数演算用の ALU，6Kbyte のローカルメモリ (SRAM)，16 エントリのレジスタファイル，および PE 間を接続する高速 I/O ポートを持つ．この浮動小数点演算器は 4 つの演算処理を同時に行えるベクトル演算機構を持ち，これにより理論性能 0.5GFLOPS が得られる．

また各 PE 間は swazzle パスと呼ばれる 64-bit 幅の

データパスがあり，データの通信や交換のため，互いのレジスタファイルが直結されている．これらのPEはPoly Execution Unitと呼ばれ，Mono Execution Unitが命令発行や同期処理等の全体制御を行う．

CSX600はMTAPのほか，チップ内部SRAM，外部DRAMインタフェースより構成され，これらを合わせて“mono”メモリ，96個のPE内メモリを“poly”メモリと呼ぶ．このmono-poly間のデータ転送は，CSX600のマルチスレッド機構を利用し，演算中にバックグラウンドで転送することが可能である．一方，swazzleパスを利用したPE間のバックグラウンド転送は行えない．

2.2 関連研究

PCに接続され，汎用プロセッサと協調動作するコプロセッサとして，天体シミュレーションを対象とするGRAPEシリーズが挙げられる．GRAPE-6[6]は，1024個の専用LSIを並列動作させることで，地球シミュレータを約2割上回るピーク性能を実現している．昨年にはFPGAを最大7個搭載するGRAPE-7が$105/GFLOPという価格対性能比を実現した[7]．

また最近では，Cell Broadband Engine[8]がゲーム用途だけでなく，高性能が要求される他のアプリケーションのアクセラレータとしても注目を集めており，単精度の高速フーリエ変換(FFT)で46.8GFLOPSの性能を達成したとの報告もある[9]．

3. 予備評価

3.1 演算性能

本研究報告では，CSX600上で実行するアプリケーションの実行時間を予測する手法について検討する．それにあたり，最初に個々の演算命令の実行時間を測定した．その結果を表1, 2に示す．実行環境は，ClearSpeed社が提供するSDK(Version 2.50)，およびプログラミング言語C^nを用いた．また加算，乗算，積和，および除算命令を256回実行した時間を計測し，ClearSpeed Advance Boardを210MHzで動作させた場合の，単位時間あたりの演算量を求めている．

ここでは，各PEのベクトル型浮動小数点演算と非ベクトル演算，またMono Processing Unitの非ベクトル演算性能を計測した．表1より，ベクトル演算機構を用いると，加算と乗算は約4倍程度，積和演算は約5〜9倍程度の性能が得られることがわかる．

3.2 転送性能

CSX600のデータパスは以下の3通りに分類できる:

- mono to poly 転送
- poly to mono 転送

表1 四則演算性能

		加算	乗算	積和	除算
Vector	float	8.351	8.351	18.67	0.551
(Poly)	double	6.936	6.936	8.905	0.347
Non-Vector	int	1.769	1.518	1.981	0.130
(Poly)	float	1.639	1.639	2.188	0.304
	double	1.393	1.406	1.803	0.132
Mono	int	1.675	1.118	1.060	0.039
	float	1.388	1.388	1.119	0.169
	double	1.383	1.383	1.116	0.130

単位:GFLOPS

表2 キャストおよびベクトル生成・分解の演算性能

	Non-vector キャスト	Vector キャスト	Vector 生成	Vector 分解
float	(int) 2.00	(int) 22.25	22.25	22.25
double	(int) 1.23	(int) 22.06	22.06	22.06
int	(float) 2.00	-	-	-
	(double) 1.83	-	-	-

単位:GFLOPS

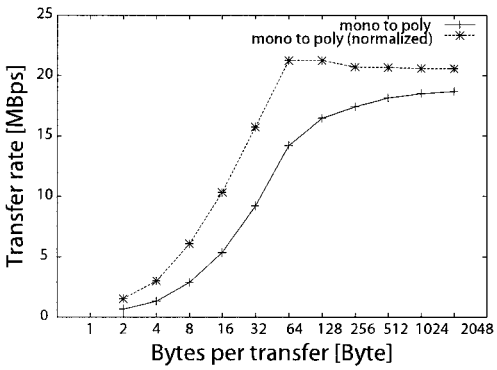

図2 Mono-Poly間のバンド幅

- swazzle 転送

バンド幅は，各データサイズについて，転送10000回に要した時間の平均を取得することで求めた．その結果を図2 - 4に示す．また，mono-poly間転送については計測プログラムのプロファイリングを行い，そこで転送機構が稼働している割合を算出し，それにより正規化したバンド幅を合わせて示す．

4. 実行時間予測手法の提案

4.1 演算時間予測

プログラム中の演算部分は，以下の手順で見積もることが可能だと考えられる．

(1) プログラム中の各種演算の回数を数える
(2) Poly演算の場合は(1)にPE数(96)をかける
(3) さらにVector演算の場合は(2)に4をかける

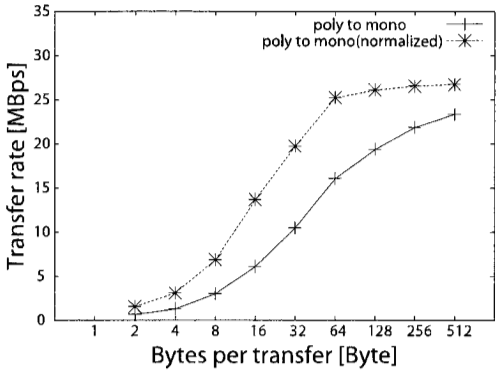

図 3 Poly-Mono 間のバンド幅

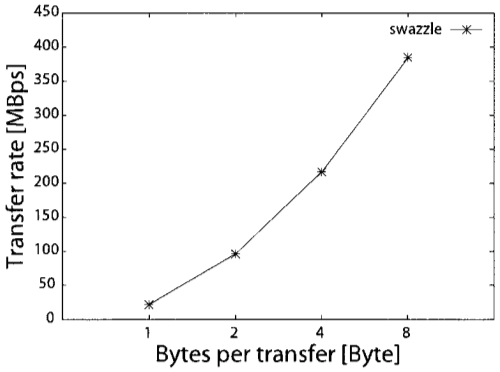

図 4 隣接 PE 間のバンド幅

(4) (3) で得られた各種演算の回数を, 対応する表 1 の GFLOPS 値で割る

実プログラムを用いた例を次章にて示す.

4.2 転送時間予測

3.2 節で示したメモリバンド幅のデータを用いることにより, プログラムの実行におけるデータ転送時間を以下の式で概算する:

$$T = T_{mp} + T_{pm} + T_{pp} \tag{1}$$

ここで T_{mp}, T_{pm}, T_{pp} は, それぞれ mono to poly 転送, poly to mono 転送, swazzle 転送に要する時間を表す. 図 2 を, データサイズにより mono to poly 間のバンド幅を求める関数 $b_{mp}(d_i)$ とすれば:

$$T_{mp} = \sum_{i=0}^{N} d_i b_{mp}(d_i) \tag{2}$$

のように, 各転送の合計時間を求められる. ここで N は転送回数, d_i はデータサイズを表す.

4.3 演算量と転送量のマッチポイント

一般に大量のデータを処理する場合, 各 PE のローカルストレージにデータが収まらず, mono-poly メモリ間の転送が必須となる. この場合, CSX600 はバックグラウンド転送機構を持ち, 演算と mono-poly 間転送のオーバーラップが可能である. したがって演算性能を維持するための転送最適化手法として double-buffering が適用できる. ここで各 PE のローカルストレージのヒープ領域を S とし, うち半分を転送に用いるものとする. ここでバンド幅を表す関数を $b(d_i)$ とすれば, 転送に要する時間 T_{trans} は

$$T_{trans} = \frac{S}{2\sum_{i=0}^{N} b(d_i)} \tag{3}$$

とおける. 一方, 単位時間あたりの演算性能が表 1 で与えられているが, このうち演算 $A_1, A_2, .., A_n$ のプログラム中で占める割合が $a_1 : a_2 : ... : a_n$, また各演算性能が $P_1, P_2, ..., P_n$ であるとする. すると, 転送時間中に可能な演算 A_i の演算回数は

$$\frac{a_i P_i}{\sum a_i} \times \frac{1}{T_{trans}} \tag{4}$$

で求められる. ここで, 実際の A_i の演算回数がこれを下回る場合, プログラム全体の演算性能が下落する.

5. 実プログラムを用いた検証

4 章で述べた手法を検討するため, 以下に述べる 2 種類の並列プログラムを CSX600 に実装し, その実行時間と予想実行時間の比較を行った.

5.1 Monte Carlo 積分

Monte Carlo 積分とは, 近似確率を利用することで, 面積分や体積分の近似解を求める手法である[10]. いま, $n \times N$ 個の乱数 $x_{[1,1]}, ..., x_{[n,N]}$ が, ある n 次元体 R の領域内に一様分布しているとする. ある関数 f で表される領域の体積 V は, n 重積分

$$\int f dV \approx \frac{V}{N} \sum_{i=1,j=1}^{n,N} f(x_{[i,j]}) \tag{5}$$

で近似できる. つまり, 関数 f に乱数値を代入し, 求める領域内にあるか否かの判定を N 回だけ反復する.

このアルゴリズムを CSX600 に実装する場合, 各 PE は, 乱数生成, $f(x_{[i,j]})$ の演算, および大小判定 ($f(x_{[i,j]})$ の値が求める領域の内側にあるか否か) の 3 点のみを行えば良いため, N 回の反復演算中には PE 間, および mono-poly 間のデータ通信がほぼ発生しない.

演算時間予測手法を検証するため, モンテカルロ積分により, 多次元球 ($N = 2$ の場合は円) の体積を求

表 3　モンテカルロ積分の総演算回数の内訳

Operations		Dimensions		
		$d=n$	$d=2$	$d=3$
Non	加算	NP	960M	960M
-vector	積和	nNP	1920M	2880M
	キャスト	nNP	1920M	2880M
Vector	乗算	$(2n+1)NP$	4800M	6720M
	積和	$(n-1)NP$	960M	1920M
	キャスト	NP	960M	960M
	生成	nNP	1920M	2880M
	分解	NP	960M	960M

N: 反復回数, P: PE 数

表 4　モンテカルロ積分の予測実行時間の内訳

Operations		Dimensions	
		$d=2$	$d=3$
Non	加算	0.689 s	0.689 s
-vector	積和	1.064 s	1.597 s
	キャスト (i→d)	1.049 s	1.573 s
Vector	乗算	0.692 s	0.968 s
	積和	0.108 s	0.215 s
	キャスト (d→i)	0.043 s	0.044 s
	生成	0.087 s	0.131 s
	分解	0.043 s	0.044 s
	合計 (予測実行時間)	3.775 s	5.262 s
	実測値	3.643 s	5.179 s

i→d: int 型から double 型への変換
d→i: double 型から int 型への変換

めるプログラムを実装した. 反復回数は $N = 10^7$ 回とし, これは CSX600 全体で 9.6×10^8 の反復数に等しい. 浮動小数点演算には倍精度型を用い, 線形合同法で乱数を生成した. ClearSpeed 社の SDK では, より質の高い乱数生成ライブラリも提供されているが, 四則演算の回数を厳密にカウントするため, 乱数を直接生成している.

上記の演算の内訳を表 3 に示す. ベクトル命令では同時に 4 つの浮動小数点演算が行われるため, ベクトル演算の項では命令発行数の 4 倍を演算回数として提示している.

ここで, 3.1 節で示した手法に従い, 表 3 の値を表 1 の性能値で割った値を表 4 に示す. この結果より, $d=2$ のとき 3.62%, $d=3$ のとき 1.60% の誤差で求められることが明らかになった.

5.2　姫野ベンチマーク

姫野ベンチマークとは, 非圧縮流体解析コードの性能評価のために考案された, ポアソン方程式をヤコビの反復法で解く際の主要な処理の演算速度を計測するもので, 科学技術計算に対するシステム性能評価に広く用いられるベンチマークプログラムである.

同種のベンチマークと比べてシステムに対する要求条件が厳しく, 計算機のメモリバンド幅やキャッシュ

表 5　ループ毎の転送量

	S		M	
	転送回数	データ量 (bytes)	転送回数	データ量 (bytes)
mono to poly	186	640	2268	640
	186	408	756	184
poly to mono	62	624	756	624
	62	392	252	168
swazzle	15624	8	128016	8
転送時間	0.01216 s		0.1232 s	

表 6　姫野ベンチマークの転送時間

	S	M
実行時間	7.473 s	60.21 s
稼働率	31.84%	30.18%
転送時間	2.379 s	18.17 s
予測転送時間	1.216 s	12.32 s

の性能によって, 結果が大きく変わることが知られている. 一回の演算ループにおいて三次元的に隣接する全ての要素の値を必要とするため, 演算量と転送量の計算オーダが等しく, メモリ間通信に高いメモリバンド幅が求められるためである.

姫野ベンチマークの主要な処理部では, 三次元配列 (p とする) に対する浮動小数点演算が反復して行われ, 最も内側のループでは 34 回の浮動小数点演算 (加減算および乗算) が行われる. 性能測定は, プロセスの反復回数 N に基づき, 単位時間当たりの浮動小数点演算数を求めることによって行う.

同ベンチマークは CSX600 向けに特化したデータフローにより実装しているが[4], 今回は性能見積もりの単純化のため, 演算と転送が排他的に行われるようにプログラムを実装した。アプリケーション内における転送量を表 5 に示す.

これより, 4 章で述べた手法に基づき, 1 回の反復ループにおける転送時間を求め, 表 5 の「転送時間」とした. これを検証するため, 姫野ベンチマークの反復ループを 100 回実行したときの実行時間, およびその転送機構の稼働率を求め, 転送時間の実測値とした. それを表 6 に示す.

この結果より, 予測転送時間が実際より 32〜48% 下回ることが分かり, (1) 式のみでは正確に予測できないことが明らかになった. この原因として, 実プログラム中では計測プログラムに比べ, 転送完了処理等のオーバヘッドが大きいことなどが予想され, さらなる予測手法の検討が必要である.

6. まとめ

本稿では, ClearSpeed 社製コプロセッサ CSX600 上でのプログラムの実行時間を予測する手法について検討を行った. 演算時間の予測には, プログラム中の各種演算回数をカウントし, 各々の実効性能を照合して見積もる手法を提案した. また転送時間には, 各データパスのメモリバンド幅の計測データに基づき, プログラム中の転送データサイズおよび回数から時間を概算する手法を試みた.

これらの妥当性を検証するため, CSX600 上に実装した Monte Carlo 積分および姫野ベンチマークのプログラムを元に, それぞれ演算時間, 転送時間を見積もり, 実際の実行時間と比較した. それによると, 演算時間は 1.60〜3.62%と小さな誤差範囲内で予測可能であったが, 実実行時間に比べ予測転送時間が 32〜48%程度に留まる結果となった. 転送時間は必ずしも実効性能が現れるとは限らず, 転送のオーバヘッド等による影響が大きいことが想定される.

今後は, 転送のオーバヘッドの原因を特定すると共に, それを考慮した転送時間予測手法を検討する. また, 演算と転送時間を合わせた実行時間の定式化を試みる予定である. さらに, これらの式は NAS Parallel Benchmarks のような並列アプリケーションなど, 複数のアプリケーションを用いて妥当性の検証を行う予定である.

謝辞

本研究の一部は, 国立情報学研究所共同研究「高性能チップ内ネットワークに関する研究」による.

参考文献

1) "遠藤敏夫, 松岡聡, 橋爪信明, 長坂真路, 後藤和茂": "ヘテロ型スーパーコンピュータ TSUBAME の Linpack による性能評価", 情報処理学会研究報告 2006-HPC-107 (pp43-48,July 31) (2006).
2) ClearSpeed Whitepaper: Accelerating the Intel Math Kernel Library: `http://www.clearspeed.com/`.
3) Y.Nishikawa, M.Koibuchi, M.Yoshimi, K.Miura and H.Amano: Performance Improvement Methodology for ClearSpeed's CSX600, *The 2007 International Conference on Parallel Processing(ICPP-07) to appear* (2007).
4) 西川由理, 鯉渕道紘, 吉見真聡, 天野英晴: ClearSpeed 製コプロセッサの並列ベンチマークによる性能評価と性能向上手法の提案, 第 13 回「ハイパフォーマンスコンピューティングとアーキテクチャの評価」に関する北海道ワークショップ (HOKKE-2007), 情報処理学会 (2007).
5) ClearSpeed Whitepaper: CSX Processor Architecture: `http://www.clearspeed.com/`.
6) Makino, J.: The GRAPE project , *Computing in Science and Engineering* (2006).
7) K&F Computing Research: `http://www.kfcr.jp/`.
8) 林宏雄, 斎藤光男, 増渕美生: Cell Broadband Engine の設計思想 , 東芝レビュー, Vol. 61, No. 6 (2006).
9) Chow, A., Fossum, G. and Brokenshire, D.: A Programming Example: Large FFT on the Cell Broadband Engine, *Proceeding of the 2005 Global Signal Processing Expo* (2005).
10) S.A.Teukolsky, W.H.Press and W.T.Vetterling: *Numerical Recipes in C, Second Edition*, Cambridge University Press, pp. 304–308 (1992).