様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９　（共通）

科学研究費助成事業　　研究成果報告書

令和　元 年　 ６ 月 １４ 日現在

機関番号：１２６０１
研究種目：基盤研究(C)（一般）
研究期間：2015～2018
課題番号：１５Ｋ００１６６

研究課題名（和文）密結合演算加速機構による通信と演算の融合に関する研究

研究課題名（英文）Research on Integration of Communication and Computation by Tightly Coupled Accelerators

研究代表者
塙　敏博（Hanawa, Toshihiro）

東京大学・情報基盤センター・准教授

研究者番号：３０３０８２８３

交付決定額（研究期間全体）：（直接経費）　　3,600,000円

研究成果の概要（和文）：GPUなど演算加速器間の直接通信を実現する密結合演算加速機構TCAアーキテクチャは低遅延通信効果により強スケーリングに有効である。本研究ではTCAの高速通信と演算を融合し、高効率な演算の実現を目的として、feasibility studyを実施した。主に数値計算アルゴリズムを用いて演算加速器向けのプログラムをOpenCLで記述しFPGA論理を生成して性能評価を行った結果、高度なパイプライン化を施すことで高い性能が得られた。OpenACC記述からOpenCLへの自動変換についても試みた。しかし、従来の記述から大幅な修正を行う必要があり、コンパイラによる自動変換は非常に困難であると考えられる。

研究成果の学術的意義や社会的意義
演算加速器向けのプログラミング言語であるOpenCLを用いたFPGA実装がFeasibleであることを示した。しかしGPUのようにデータ並列の記述では性能が得られず、FPGAのアーキテクチャを考慮し記述する必要がある。OpenCLのカーネルを分割し、パイプライン方式での制御に変更することで行列積については高い性能が得られた。
また、通常のソフトウェア最適化技術と逆行する、冗長な記述や、ループ中での分岐などがFPGAで有効である。今後に向けた最新インタフェースとして、CPUとキャッシュ一貫性を持つFPGA接続、3次元積層メモリに関して性能確認を行い、現状の各5倍、30倍程度のバンド幅が期待できる。

研究成果の概要（英文）：Tightly Coupled Accelerators (TCA) architecture, which realizes direct communication among accelerators such as GPUs, is effective for improving strong-scaling performance thanks to low-latency of TCA. In the present study, the feasibility study was performed for the purpose of realization of highly efficient computation by fusion of fast communication using TCA and FPGA computation. Several kernels including numerical algorithms were described for accelerator by OpenCL, and higher performance could be achieved by further modification toward highly pipelined manner. Automatic conversion from OpenACC to OpenCL was also investigated. However, since drastic modification is required from traditional description manner, it is considered that automatic optimization is too complicated.

研究分野：高性能計算

キーワード：FPGA　演算加速装置　PCI Express　OpenCL　OpenACC

様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９、ＣＫ－１９（共通）

１．研究開始当初の背景

GPU (Graphics Processing Unit) の持つ高い演算性能とメモリバンド幅に着目し、様々な HPC アプリケーションへ適用する GPGPU (General Purpose GPU)が盛んに用いられるようになった。しかしながら、GPU の演算性能の高さに対して、GPU と CPU を接続するインタフェースである PCI Express (PCIe) の性能がボトルネックとなっている。GPU を搭載する高性能計算サーバをノードとする GPU クラスタも増加する一方であり、TOP500 リストには年々多くの GPU 搭載スーパーコンピュータが名を連ねている。

しかしながら、GPU クラスタにおいては、複数ノードをまたがる GPU 間の通信が必要である。従来は、CPU メモリを介してノード間を転送する必要があったため、

1. GPU メモリ から CPU メモリ
2. CPU メモリからネットワークデバイスを経由して別ノードの CPU メモリ
3. CPU メモリから GPU メモリ

の 3 回のデータコピーが必要となっていた。特にサイズの小さなデータ転送の場合にはオーバーヘッドが顕著であり、レイテンシの増加が性能低下を引き起こす。従って、問題サイズを固定し使用する GPU 数・ノード数を増やすことで実行時間の削減を目指す Strong-scaling では、相対的に小サイズのデータ転送が増加し、通信ボトルネックになるため、スケーラビリティの向上が期待できなくなる。

そこで、筑波大学計算科学研究センターにおいて、HA-PACS としてコモディティ技術による大規模 GPU クラスタに加え、ノード間接続および GPU 間接続に、レイテンシとバンド幅の改善を目指した独自開発の密結合並列演算加速機構 TCA (Tightly Coupled Accelerators)を提案してきた。TCA の実現においては、ルータチップとして PEACH2 を開発した。これは FPGA を用いて PCIe パケットの中継処理、高度な DMA 転送などを全てハードワイヤード処理で行うものである。PCIe Gen2 x8 レーンのハード IP を 4 ポート分内蔵した Altera 社 Stratix IV GX を用いて注意深く実装を行ってきた結果、優れた低遅延性能を実現した。姫野ベンチマークを始め、GPU 用 QCD ライブラリ QUDA、ストリーム処理などの様々な応用に適用し、高い Strong-scaling 性能を示すことを示してきた。

さらに、プログラミング言語として、PGAS (Partitioned Global Address Space)言語 XcalableMP と演算加速器のためのプログラミング言語 OpenACC とを組み合わせた XcalableACC (XACC)を提案し、TCA に向けた実装が行われている。ユーザは逐次プログラムに指示文を追加するだけで、TCA 通信機構を意識せずに記述でき、自分で通信コードなどを記述した場合と比べて遜色ない性能を得ることができる。

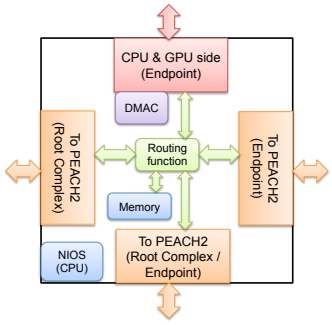

図 1 PEACH2 の内部構造

２．研究の目的

GPU などの演算加速装置を搭載したクラスタにおいて、演算加速装置間の直接通信を可能にする密結合演算加速機構 TCA (Tightly Coupled Accelerators)アーキテクチャを提案し、実証クラスタ HA-PACS/TCA において、低遅延通信の効果により Strong-scaling に極めて有効に働くことを示してきた。これらの知見を元に、本研究では TCA による加速装置間の高度な通信機構と、比較的小規模な演算とを融合することにより、演算加速器間の実効遅延をさらに減らすことを目的とする。これまでに FPGA による通信機構 PEACH2 の通信エンジンを開発してきたが、FPGA 内部の論理ブロックには余裕がある。その一方で、近年 OpenCL 言語で記述した加速装置向けのプログラムを基に FPGA 論理を生成し動作させることが可能になってきている。

そこで、これまでに開発してきた FPGA による通信機構 PEACH2 の通信エンジンに加え、プログラム中に記述されたコードから FPGA 論理を生成し、動的に読み込むことによりオフロード実行を可能にすることを目的にする。具体的には、TCA における通信と比較的小規模な演算とを融合して、演算加速装置間、また演算加速装置と CPU との間でストリーム的、パイプライン的な処理を FPGA 上にオフロードし、クラスタシステム全体として性能向上を目指す。

３．研究の方法

本研究においては、

(1) OpenCL を用いた FPGA に向けた演算機能の実現
(2) OpenACC をベースにした FPGA 向けプログラミングの検討

について実施した。

４．研究成果

(1) OpenCL を用いた FPGA に向けた演算機能の実現

実際に OpenCL カーネルが FPGA 上で動作するためには、以下のような条件が必要である。

① ホストと FPGA 間が PCI Express で接続されていること
アクセラレータとして用いるためには、GPU などと同様に、高速汎用 I/O である PCI Express を用いて接続する必要がある。

② FPGA の内部が部分再構成(Partial reconfiguration)可能であること
PCI Express インタフェース、 DDR メモリインタフェースなどの IO 回路は不変であり、特に PCI Express インタフェースが変更されてしまうと、ホストが停止してしまう。これらのインタフェースを除き、OpenCL のカーネルに相当する範囲だけを再構成する。

③ PCI Express 経由で FPGA 内部が再構成できること
OpenCL のカーネルとして動作するためには、カーネル実行前に FPGA 構成情報（コンフィグレーションデータ）をホストから FPGA にダウンロードする必要がある。その際、コンフィグレーションデータは数十 MB を超えるため、PCI Express 経由で転送し、かつデータを受信すると同時に内部を再構成するような仕組みが必要である。

これらを実現する OpenCL BSP (Board Support Package)がベンダから提供される必要がある。

本研究では、まず Bittware 社の PCI Express ボード S5-PCIe-HQ (s5phq_d5)を用いて研究を行った。このボードには Altera 社の FPGA Stratix V GS D5 が搭載され、8 GB の DDR3 オンボードメモリ、PCI Express Gen3 x8 が搭載されている (OpenCL 使用時は Gen2 x8 でしか使用できない)。ソフトウェア環境としては、Altera 社 Quartus II 16.0.1 を使用した。OpenCL コードのコンパイルを行うには、Altera Offline Compiler (実際には aoc コマンド)を実行するだけである。それによって、内部では以下のような処理が行われる。

図 2 Bittware 社 S5-PCIe-HQ

① OpenCL のプログラム構造から、パイプラインステージを切り出し、ステートマシンを構成する。

② 必要な資源を見積もる。(ロジック使用率、レジスタ使用率、メモリ使用率、DSP 使用率)

③ PCI Express や DDR3-DRAM インタフェースなどのモジュール、OpenCL を元に Verilog HDL の記述が生成される。

④ Quartus (Altera 社の FPGA 向け論理合成ツール)を起動し、論理合成、配置配線等を行う。

⑤ FPGA コンフィグレーションデータのビットストリームを含む .aocx ファイルが出力される。

本ボードを用いて、疎行列数値計算として疎行列ベクトル積および CG 法[成果⑤]、階層型行列ベクトル積[成果④]を実装した。階層型行列ベクトル積には、ppOpen-APPL/BEM ver.0.5.0 に含まれる HACApK ライブラリ利用版のリファレンス実装を用いる。ppOpen-APPL/BEM とは、JST CREST「自動チューニング機構を有するアプリケーション開発・実行環境: ppOpen-HPC」の構成要素の一つであり、境界要素法(Boundary Element Method, BEM)用のソフトウェアフレームワークである。

図 3 に CG 法における実装の性能比較を示す。単純な実装に比べ、最適化を施すことで 64 倍の高速化が見られたが、Intel Xeon CPU E5-2680v2 と比較して、14 倍程度の遅い結果となった。一方、電力を比較すると、CPU に比べて FPGA は 1/10 程度の電力で動作する。

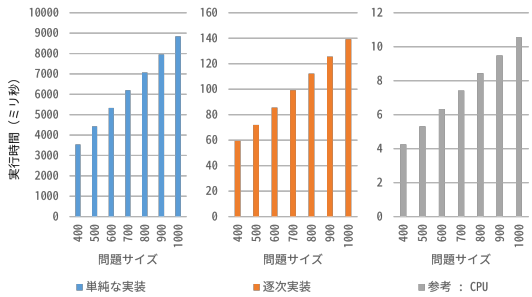

図 3 CG 法実装の比較

FPGA における実装上の問題は、一度コンパイルするたびに数時間を要する。また GPU 等の最適化技術は使えず、むしろ逆行するようなコードの方が高速化されることも多い。さらに、SIMD 化により並列処理も可能であるが、FPGA 上に収まりきらず、コンパイルの終盤になって初めてエラーになることもある。このように開発効率は極めて劣悪であり、FPGA の世代が進み、チップ当たりの回路規模が増大するとさらにコンパイル時間は増加する傾向にある。

平成 28 年度からは、Stratix V の後継である Arria10 FPGA 搭載 PCIe ボード(Terasic 社 DE5a-Net)を用いて研究を継続した。

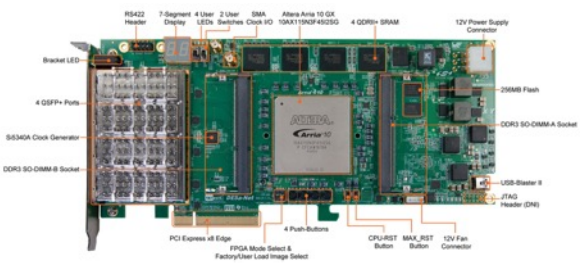

図 4 Terasic 社 DE5a-Net

これまでの階層型行列ベクトル積に加えて、階層型行列の生成、ソートアルゴリズムなどに適用を試みた[成果①③]。その結果、CPU 1 コ

アを超える性能は得られたが、全体としては 1/10 程度の性能に制約されている。性能が低い原因の 1 つは、ボードに搭載されたメモリのバンド幅が低いためであり、DE5a-Net は S5-PCIe-HQ に比べてさらにメモリバンド幅も小さい(表 1)。また、根本的に改善するためには、計算の各ステップをカーネルとして切り出し、パイプライン的に処理する必要がある。

表 1 S5-PCIe-HQ と DE5a-Net の比較

FPGA	Intel Arria 10 GX E1 (10AX115N2F45E1SG)	Altera Stratix V GS D5 (5SGSMD5K2F40C2)
#Logic units (ALMs)	427,200	172,600
#RAM blocks (M20K)	2,713	2,014
#DSP blocks	1,518 (浮動小数点)	1,590 (整数のみ)
ボード	Terasic DE5a-Net E1	Bittware S5-PCIe-HQ GSMD5
DDR メモリ容量	(4 + 4) GB	(4 + 4) GB
DDR メモリバンド幅	17.1 GB/sec	25.6 GB/sec
PCIe I/F (OpenCL の場合)	Gen3 x8	Gen2 x8

これらの研究から得られた知見としては、OpenCL からのコンパイルに関して基本的に容易に実現はできるが、ソフトウェアとしての最適化技術とは全く異なる最適化が必要であり、特にデータの流れを意識して OpenCL のカーネルを分割しパイプライン処理を記述する、といった実装上の工夫が必要であることがわかった。例えば、Intel によってパイプライン処理に最適化された OpenCL カーネルによる単精度行列積プログラムを用いた場合、DE5a-Net では、10x16 の Systolic array で A: 10240 x 4096, B: 4096 x 16384, C: 10240 x 16384 の行列を用いた結果、853.5 GFLOPS の性能を得た。これは GPU の性能には敵わないものの、同世代の CPU (Xeon Broadwell)と比べると良い結果であり、電力効率を考えると 10 倍程度向上する。このような最適化は従来のコンパイラ技術では到底達成できるものではなく、ライブラリやマクロなどを援用する他ないと考える。

また、使用した DE5a-Net ボードでは OpenCL の記述内容によって回路合成ができず、不具合を解消することができなかった。ボードベンダーからのサポートも得られなかったため、DE5a-Net の使用は途中で諦めざるを得なくなった。

そこで平成 30 年度には HBM2 メモリを持つ最新 Stratix10MX 搭載ボード (Intel Stratix 10 MX FPGA 評価キット)、さらにはベンダーの異なる Xilinx 社 Virtex UltraScale+搭載ボードを導入し、研究を継続している。

これまでの FPGA の利用は、GPU などのアクセラレータと同様に、PCI Express を介してホストに接続し、CPU からの処理をオフロードする形態であった。しかし近年では、Intel 社により Xeon CPU と Arria 10 FPGA を QPI で接続した製品が発表されたり、今後 Xeon プロセッサと FPGA が UPI で接続され 1 チップ化された製品も登場してきた。IBM Power9 プロセッサでは、NVLink2 インタフェースにより NVIDIA Volta GPU が直接接続可能になっている一方で、CAPI, OpenCAPI のインタフェースも持っており、同じく CAPI, OpenCAPI をサポートする Xilinx FPGA が接続できる。これらの環境では CPU と FPGA との間でキャッシュコヒーレントなメモリ転送が可能になり、低オーバヘッドで FPGA へのオフローディングが実現できる。例えば、Intel Xeon Broadwell プロセッサと Arria 10 が QPI で接続された Intel HARP2 プラットフォームでメモリバンド幅を測定すると、27 GB/秒と、PCIe 接続のシステムの 5 倍以上の性能が得られており[成果②]、今後は CPU や GPU の不得手な処理を、比較的細粒度に FPGA 上で実行できるようになると考えられる。

また、3 次元積層メモリ HBM2 を用いることでメモリバンド幅も飛躍的に向上する。Stratix 10 MX 評価ボードでは、理論バンド幅 512 GB/秒に対して 370 GB/秒程度の性能が得られることがわかった。

今後も基盤(B)「再構成可能システムと GPU による複合型高性能計算プラットフォーム」において知見を活かしていく。

(2) OpenACC をベースにした FPGA 向けプログラミングの検討

FPGA に向けた高位記述言語(高位合成ツール: HLS)としては、C または C++をベースにしたもの、ハードウェア記述言語をベースにした SystemVerilog、演算加速装置向けの OpenCL を用いたものとがある。本研究では、既存のソフトウェアからの親和性が高く、Intel(旧 Altera)社、Xilinx 社両者で共通性が高いと考えられる、OpenCL を用いた実装を考える。実際に Intel 社 FPGA 向けの開発環境としては、(1)で報告したように性能の観点を除けば完成度は高く、プログラムから実際に動作する FPGA 論理に変換することが可能であると考えられる。

OpenACC から OpenCL への変換を実現する手法としては、いくつかのパスが考えられる。図 5 にフローを示す。逐次のソースコードに OpenACC の指示文を追加したコードに対して、それを OpenACC 対応コンパイラでコンパイルすることで、ホストコード、OpenCL カーネルに分離する。

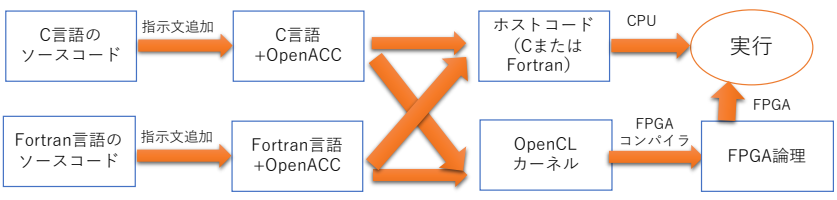

図 5 FPGA 実装に向けたフロー

ここでは、Fortran で記述されたコードを OpenCL で実現する手段を考える。

① まず、PGI Fortran コンパイラで OpenACC コードをコンパイルする[1]。その際、NVIDIA

GPU 向けに CUDA のコードを出力する -ta=tesla:keepgpu,nonvvm オプションを指定する。それによって Filename.n001.gpu のようなファイルが生成され、CUDA C のコードが格納されている。

② CU2CL によって CUDA から OpenCL に変換する[3]。

この手順により、Fortran による OpenACC 記述から OpenCL カーネルが生成できることを確認した。

問題点としては、ホストコードが可読性のある状態では取得できず、別途ホストコードを作成する必要があることである。XACC コンパイラにおいては、実装のベースになっている Omni OpenACC コンパイラで OpenCL に類似した PezyCL への変換が可能であるため[3]、OpenCL への対応も比較的容易であると考えられるが、今後の課題である。また、(1)で述べたように、単に OpenCL から FPGA 向けに論理合成できたとしても、全く性能を引き出せない可能性が高く、何らかの変換を行うか、あるいは元の記述に大きな制約をかける必要がある。

一方、AMD が開発している HIP において、様々な環境に対応した移植性の高い C++言語環境とツール群について設計・開発されている[4]。今後 HIP のような言語を用いてプログラミングを行うのも一つの方向性として検討する必要がある。

参考文献：

[1] Bittware, S5-PCIe-HQ, https://www.bittware.com/fpga/s5phq/
[2] PGI Community Edition, https://www.pgroup.com/products/community.htm.
[3] CU2CL: Automating CUDA-to-OpenCL Translation, http://chrec.cs.vt.edu/cu2cl/index.php
[4] A. Tabuchi, Y. Kimura, S. Torii, H. Matsufuru, T. Ishikawa, T. Boku, M. Sato, Design and Preliminary Evaluatino of Omni OpenACC Compiler for Massive MIMD Processor PEZY-SC, Intl. Workshop on OpenMP (IWOMP 2016), 2016.
[5] HIP：C++ Heterogeneous-Compute Interface for Portability, https://gpuopen.com/compute-product/hip-convert-cuda-to-portable-c-code/

５．主な発表論文等

〔雑誌論文〕（計　0 件）

〔学会発表〕（計　5 件）

① 塙 敏博、伊田 明弘、星野 哲也、OpenCL を用いた FPGA による階層型行列計算、情報処理学会ハイパフォーマンス研究会、Vol. 2018-HPC-163, No. 26, pp. 1-8, 2018 年 3 月
② Toshihiro Hanawa, Taisuke Boku, Design Experience on Intel HARP2 Platform using OpenCL, 18th SIAM Conference on Parallel Processing for Scientific Computing, March 2018.
③ 塙 敏博、伊田 明弘、星野 哲也、階層型行列計算の FPGA への適用、情報処理学会ハイパフォーマンス研究会、Vol. 2017-HPC-161, No. 10, pp. 1-10, 2017 年 9 月
④ 塙 敏博、伊田 明弘、大島 聡史、河合 直聡、FPGA を用いた階層型行列ベクトル積、情報処理学会ハイパフォーマンス研究会、Vol. 2016-HPC-155, No. 40, pp. 1-9, 2016 年 8 月
⑤ 大島 聡史, 塙 敏博, 片桐 孝洋, 中島 研吾、FPGA を用いた疎行列数値計算の性能評価、情報処理学会ハイパフォーマンス研究会、Vol. 2016-HPC-153, No. 1, pp. 1-9, 2018 年 3 月

〔その他〕
ホームページ等 https://www.cspp.cc.u-tokyo.ac.jp/hanawa/

６．研究組織

(1)研究分担者

研究分担者氏名：

ローマ字氏名：

所属研究機関名：

部局名：

職名：

研究者番号（8 桁）：

(2)研究協力者
研究協力者氏名：
ローマ字氏名：

※科研費による研究は、研究者の自覚と責任において実施するものです。そのため、研究の実施や研究成果の公表等については、国の要請等に基づくものではなく、その研究成果に関する見解や責任は、研究者個人に帰属されます。