Vol.2016-HPC-155 No.14
2016/8/8

HPCにおけるHSAの性能評価

石村 脩[†1,a)]　吉本 芳英[†1,b)]

概要：今日の High Performance Computing (HPC) では、処理の高速化のため、General Purpose computing on GPU (GPGPU) が頻繁に用いられている。しかし、一般的にこれらで用いられている GPU は CPU に汎用バスを介して接続されているため、CPU と GPU の間のデータ転送や処理の切り替えが遅く、粒度の細かい並列処理には向かない。一方で近年開発が進められている Heterogeneous System Architecture (HSA) では、汎用バスを介したデータ転送ではなく CPU と GPU で仮想空間を含めて統合されたメモリによるデータ共有 (Heterogeneous Uniform Memory Access) やカーネルモードへのコンテキストスイッチをせずに GPU にジョブを渡すことを可能とする機構 (Heterogeneous Queuing) が用意され、この問題への対応がなされていると主張されている。しかし、HSA が HPC において、実際にどの程度の効果を持つものであるか検証した先行研究は存在しない。そこで本研究では、HSA を採用した APU (Godavari) の性能評価を、データのアクセス遅延・バンド幅、および GPU のジョブの起動遅延に注目して行った。

キーワード：HSA APU GPGPU Godavari 性能評価

Effectiveness of Heterogeneous System Architecture in High Performance Computing

OSAMU ISHIMURA[†1,a)]　YOSHIHIDE YOSHIMOTO[†1,b)]

Abstract: Recently, General Purpose computing on GPU (GPGPU) has frequently been used as an acceleration technique in High Performance Computing (HPC). Most of the GPUs used in GPGPU are discrete GPUs which are connected to CPUs via general purpose buses. Therefore, they are not effective for fine grained parallelism because of their large latency in a context switch and slow data transfer. Heterogeneous System Architecture (HSA), which has been developed recently, provides unified memory access including the virtual space between CPU and GPU without the transfer (Heterogeneous Uniform Memory Access) and job queuing without kernel context switch (Heterogeneous Queuing) in order to address this problem. However, there is no previous research on the effectiveness of HSA in HPC. In this paper, from the viewpoint of latency and bandwidth in data access and job queuing latency, we evaluated the performance of Godavari that is an APU implements HSA.

Keywords: HSA APU GPGPU Godavari Benchmark

1. はじめに

今日、GPGPU は HPC において幅広く用いられるようになっている。2015 年 10 月の時点で、Top500 に掲載されているシステムの 100 以上が GPU をアクセラレータとして用いている。これらは弱スケーリングの問題に対しては非常に有用であり、より大きな問題を解くことが可能となっている。しかし、対象となる問題のサイズは何もデータサイズが日々大きくなるものばかりではない。そのため、今後は強スケーリングの問題をいかに効率的に解くことが可能かが重要となっている。

現在の一般的な GPGPU を用いたスーパーコンピューターは、強スケーリングの問題に対しては一つ大きな問題が存在する。それは、これら用いられている GPU の多くは外付け GPU(dGPU) であり、これらは PCI Express など

†1 現在，東京大学
Presently with University of Tokyo
a) oishimura@is.s.u-tokyo.ac.jp
b) yosimoto@is.s.u-tokyo.ac.jp

の非常に遅いバスによって CPU と繋がれている点である。表 1 は、幾つかの一般的な CPU と GPU のサポートするメモリと、その転送速度である。PCI Express 3.0 x16 の転送速度は 16GB/s と、CPU のサポートするメモリに近い転送速度であり、GPU に対しては非常に遅いことがわかる。

表 1 各デバイスにおけるメモリの種類
Table 1 Memory Support List

Device	A10-7850K	Xeon E5-4600 v3	Radeon R9 Fury X	GTX TITAN X
Manufacturer	AMD	Intel	AMD	NVIDIA
Type	CPU	CPU	GPU	GPU
Memory	DDR3-2400	DDR4-2133	HBM1	GDDR5
BW*1(GB/s)	38.4	68	512	336.5

そのため、並列化のオーバーヘッドが大きく、粒度の小さい並列性の並列化による高速化できない。

AMD の CPU に、AMD Fusion Accelerated Processing Units(APU) と呼ばれる製品がある。これは一つのダイに CPU と GPU を統合しており、主にエントリークラスのデスクトップ PC やラップトップなどで利用されている。そして近年の APU には Heterogeneous System Architecture(HSA) と呼ばれる、CPU や GPU・DSP など様々な種類のデバイスを統合的に扱うことを目指したアーキテクチャが実装されている。HSA は並列プログラミンの敷居を取り払うことと、各デバイス間の通信レイテンシを減らすことを目標とし、Heterogeneous Unified Memory Architecture (hUMA) と Heterogeneous Queuing (hQ) と呼ばれる二つの機能を備えている。hUMA は GPU と CPU から同じシステムメモリにアクセスすることを可能とする機能である。hQ はハードウェアによるサポートにより、CPU、GPU 双方から、システムコンテキストスイッチ無しでのタスクのオフロードを可能とする機能である。

これにより、HSA が実装された APU には、HPC の高速化を行う潜在的能力があると考えた。しかし、APU が HPC ではほぼ全く使われていないこともあり、HSA の性能について評価を行った先行研究は存在しない。そこで、本研究では、HSA が実装された APU である Godavari を利用し、幾つかの基礎的なベンチマークと、実アプリケーションによるベンチマークを行い、HSA の評価を行った。

2. HSA

Heterogeneous System Architecture (HSA) は HSA Foundation によって標準化されているアーキテクチャである。The HSA Foundation は ARM, AMD, Imagination, MediaTek, Qualcomm, Samsung, TI と言った企業によって構成されている。[1]. HSA は CPU や GPU、DSP など、複数の種類のシームレスな統合を目指し、複数デバイス間のコミュニケーションのレイテンシを減らしている [2]。

*1 BW: Bandwidth

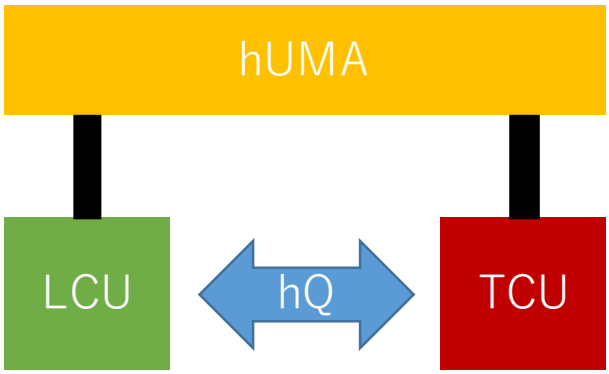

図 1 HSA の機能
Fig. 1 The Features of HSA

図 1 は HSA を構成する機能を示す。

HSA は Latency computer unit (LCU) と Throughput computer unit (TCU) と呼ばれる 2 種類の計算ユニットをサポートしている。[2] LCU は CPU を一般化したもので、CPU の命令セットと HSA intermediate language (HSAIL) の命令セットの双方をサポートする。TCU は GPU や DSP を一般化したもので、HSAIL のみをサポートする。

LCU と TCU はそれぞれ HSA の提供する Heterogeneous Unified Memory Architecture (hUMA) と呼ばれる機能を通し、同じ仮想メモリ空間にアクセスすることができる。また、Heterogeneous Queuing (hQ) と呼ばれる機能を通し、LCU から TCU、TCU から LCU また LCU 同士、TCU 同士でタスクコンテクストスイッチ無しのタスク発行が可能である。

これらの機能により、タスクオフロードの際のタスクフローが、図 2 で示される従来のものから、図 3 で示されるタスクフローへと簡略化される。

3. 性能評価

3.1 テスト環境

テストに用いた環境を表 2 に示す。従来の環境との比較を行うため、dGPU として Radeon HD7970 の環境を用意した。

HSA 及び OpenCL のコンパイラ及びランタイムは以下のものを利用した。

- HSA
  - HSA-Driver-Linux-AMD v1.6
  - HSA-Runtime-AMD v1.0.3
  - CLOC v0.9.8
- OpenCL

*2 AMD Turbo Core Technology [4] は停止している。.
*3 DDR3-2400 は JEDEC で JESD79-3F として標準化されているいない。[5]. 設定には AMD Memory Profile [6] を利用している。

表 2　テスト環境
Table 2　Test Platforms

Platform Name	CPU (OpenMP)	iGPU (HSA)	iGPU (OpenCL2)	dGPU (OpenCL1)
Runtime	OpenMP	HSA1.0	OpenCL2.0	OpenCL1.2
Kernel	Ubuntu 3.19.0-42	Ubuntu 4.0.0-100002	Ubuntu 3.19.0-42	Ubuntu 3.19.0-42
GPU	-	Godavari	Godavari	Radeon HD 7970
Microarchitecture	-	GCN 1.1	GCN 1.1	GCN 1.0
Manufacturing Process	-	28nm	28nm	28nm
Compute Units	-	8	8	32
Hardware Threads	-	512	512	2048
Memory Bus Type	-	DDR3	DDR3	GDDR5
Memory Bandwidth	-	38.4GB/s	38.4GB/s	264GB/s
Memory Type	-	Shared	Shared	Independent
Device Memory	-	-	2274MB	2962MB
Local Memory	-	-	32KB	64KB
Core Clock Frequency	-	866MHz[*2]	866MHz[*2]	925MHz
Peak FLOPS	-	886.784GFlop/s	886.784GFlop/s	3788.8GFlop/s
Bus interface	-	-	-	PCIe 3.0 x16
CPU	A10-7870K			
Processor Clock	3.9GHz[*2]			
System Memory	16GB (DDR3-2400)[*3]			
Cache	L1 : 64KB, L2: 4MB			
Motherboard	ASUS CROSSBLADE RANGER			
Chipset	AMD A88X FCH			

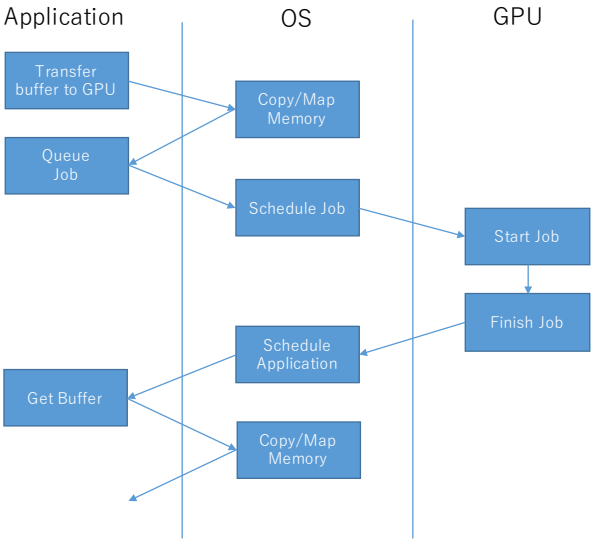

図 2　従来のタスクオフロードのフローチャート [3]
Fig. 2　Flowchart of Conventional Task Offloading [3]

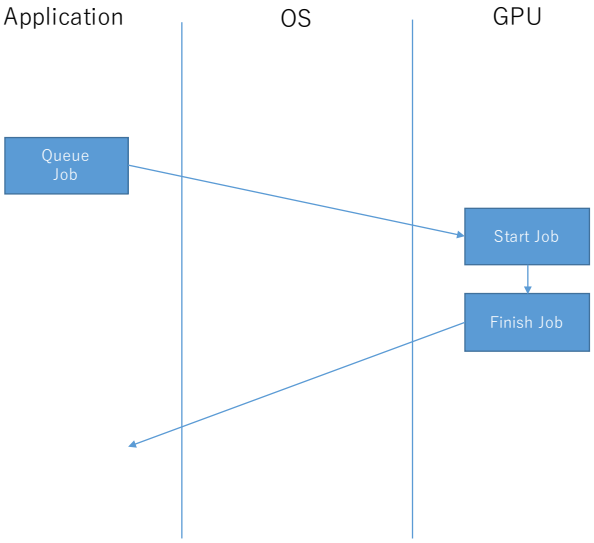

図 3　HSA でのタスクオフロードのフローチャート [3]
Fig. 3　Flowchart of HSA Task Offloading [3]

– AMD Stream SDK v3.0
– AMD Catalyst Graphics Driver v15.2

なお、現在 OpenCL2.0 を用いることで、HSA の一部機能を用いることが可能となっている。iGPU (OpenCL2) では Shared Virtual Memory として hUMA が機能する。一方、iGPU (HSA) では hUMA 及び hQ の双方が機能する。

3.2　基礎的なベンチマーク

基礎的なベンチマークとして、メモリのバンド幅の測定及び、タスクの発行にかかる時間、タスクの発行とともに行われるメモリの初期化にかかる時間の測定を行った。

メモリのバンド幅の測定は、十分に大きな量のデータを転送することにより計測した。dGPU (OpenCL) の場合に

は、データをシステムメモリから GPU のメモリに転送し、その後カーネルがデータを読み書きを行い、再びシステムメモリに戻すという操作を行っている。そのため、ここで示されるバンド幅は PCI Express のバンド幅と GPU メモリのバンド幅の双方に律速されたものとなる。

結果は図 4 の様になった。

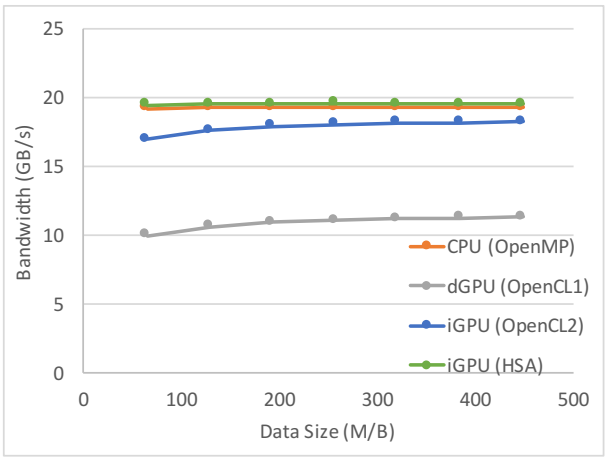

図 4 各プラットフォームにおけるメモリの読み書きのバンド幅
Fig. 4 Bandwidth (read+write) of memory copy on each platform

タスクの発行時間は、何も計算を行わないカーネルを発行し、その時間を計測した。

メモリの初期化にかかる時間は、小さな異なる量のデータ処理を行うカーネルを作成し、それらの処理にかかかる時間から、線形近似によりデータ量 0 の時間を仮想的に計算した。

各カーネルの実行時間は図 5 の様になった。図 5 に示される点線が、近似直線である。

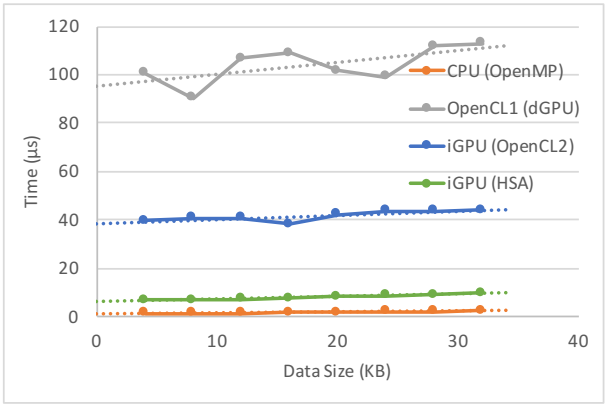

図 5 メモリ操作を含むタスクの発行時間
Fig. 5 Time Consumption of Memory Attaching Task Dispatching

以上の結果をまとめたものが、表 3 である。T_d はタスク発行にかかる時間、T_m はメモリの初期化にかかる時間、BW はメモリのバンド幅の結果を示す。なお、メモリ操作を含むタスク一つを発行する場合の時間 T は以下で示される式となる。

$$T = T_d + T_m + (datasize(read + write))/BW$$

OpenMP を利用した場合については、メモリの初期化とタスク発行の時間を分離することがでないため、表のような表記となっている。

表 3 基礎的なベンチマークの結果
Table 3 Basic Benchmark Results

Platform	CPU	dGPU	iGPU	
Runtime	OpenMP	OpenCL1	OpenCL2	HSA
T_d (μs)	$\leq$1.2	36.9	38.0	5.0
T_m (μs)	$\leq$1.2	59.6	1.6	1.2
BW (GB/s)	19.4	11.5	18.4	19.5

この結果により、HSA はタスク発行及び、メモリのバンド幅の双方の点について、大きな優位性を持つことがわかった。

3.3 実アプリケーションによるベンチマーク

基礎的なベンチマークの結果に基づいて、行列積及びハッシュ関数を実アプリケーションのベンチマークとして選択した。行列積のベンチマークの結果は図 6 のようになった。

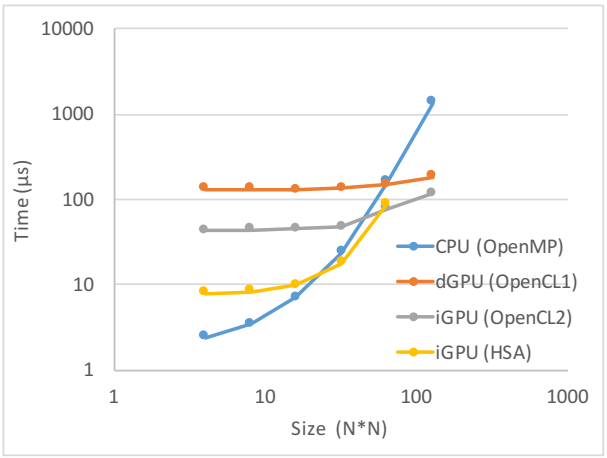

図 6 行列積計算時間
Fig. 6 Matrix Product Benchmark

この結果により、特定のデータサイズにおいて少しではあるが、iGPU(HSA) が有用性を示すことがわかる。しかしながら、iGPU(HSA) の結果にはおかしな挙動が見られる。それは、データサイズが 32*32 から 64*64 の部分における傾きが iGPU(OpenCL2) と比較しかなり急である点である。本来であれば、どちらも同じ計算能力を持つハードウェアを利用しているため、この部分の傾きは同程度になり、緩やかに iGPU(HSA) の値と iGPU(OpenCL2) の値が収束してゆくはずである。

ハッシュ関数のベンチマークの結果は図 7 のようになっ

た。なお、ここで用いているハッシュ関数は 512bit を入力し、128bit を出力する MD5 関数である。この結果では、iGPU(HSA) が他のプラットフォームに対して優位性を示す状況は存在しない。しかし、この結果でも iGPU(OpenCL2) と比較し、行列積同様、異常な挙動が見られる。

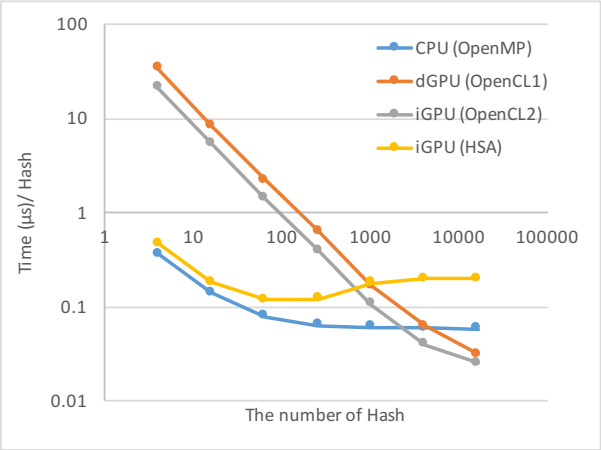

図 7　ハッシュ計算時間
Fig. 7　Hash Calculation Benchmark

4. 関連研究

初期の HSA が実装されていない APU に関する研究としては以下のような先行研究が存在する。

2011 年に M. Daga, A. Aji, 及び W. chun Feng らによって、第一世代 APU である AMD Zacate APU パフォーマンスが計測され、データ量が十分に大きい時にはバンド幅の効果が発揮されるが、データ量が小さい時には DMA の初期化などが高く、効果はあまりないという結果が示されている [7]。2012 年には K. L. Spafford, J. S. Meredith, S. Lee, D. Li, P. C. Roth, 及び J. S. Vetter らが第一世代 APU である Llano を用いたパフォーマンス計測を行い、CPU-GPU 間のメモリ転送が非常に多い場合のみ良いパフォーマンスを発揮するという結果が示された [8]。2013 年には J. Jansson によって第二世代 APU の Trinity を用いた計測を行いデータ量が十分に大きい場合にデータの転送が不要である利点が発揮され、良いパフォーマンスを発揮するという結果が示されている [9]。これらの結果は HSA が実装された APU において小さなタスクの並列化が効率化されるという結果とは対照的なものである。

タスクのオフロードのコストを減らすアプローチとしては、アクセラレータと CPU をより密に結合させるという HSA のアプローチ以外にも、幾つかの手法が存在する。一つは、単純により大きな帯域を持つバスを用意するという方法だ。NVIDIA の NVLink [10] [11] は、最大 160GB/s の帯域を持ち、CPU との通信も可能とされている [12]。しかし、このように新しい規格のバスでデバイスを繋ぐという手法は、繋がれるデバイス全てがその規格に対応しなければならず、規格が一般化するまではコストが高くなりがちである。また別の手法の一つとして、汎用演算も含めてアクセラレータにオフロードするという手法である。これにより、既存のシステムのバスに左右されず、アクセラレータ内で自由に設計を行うことが可能となり、タスクのオフロードのコストを下げることも可能となる。PezySC2 ではこのアプローチが取られるようである [13]。

5. 終わりに

この論文では HSA の評価を行った。基礎的なベンチマークの結果によると、HSA が有用である領域が存在することことがわかった。しかしながら、実アプリケーションのベンチマーク結果は、基礎的なベンチマークから推測される結果と大きく異なり、顕著な速度向上はなされなかった。これはコンパイラやランタイムが原因ではないかと推察される。しかし、この原因の究明及び解決は今後の課題とする。

本研究では、AMD の APU および GPU のみで性能評価を行った。しかし、HPC で主に用いられる GPU は NVIDIA の GPU である。また、GPU 以外のアクセラレータとして Intel Xeon Phi や Pezy SC なども利用されている。これらのとの比較なしに、APU の HPC における有用性を語ることはできないであろう。また、Intel の iGPU の性能も近年急成長しており、これらと APU の比較も非常に興味深い。以上の点についても今後の課題とする。

謝辞　本論文を作成するにあたり、実験に関して様々なアドバイスをくださった平木敬教授。、機材の準備にご協力いただいた泊久信さん、及び平木研究室の皆様へ心から感謝の気持ちと御礼を申し上げ、謝辞にかえさせていただきます。

参考文献

[1] HSA Foundation: HSA Foundation ARM, AMD, Imagination, MediaTek, Qualcomm, Samsung, TI, http://www.hsafoundation.com/. (Visited on 01/22/2016).
[2] HSA Foundation: *Heterogenious System Architecture: A Technical Review* (2012).
[3] Bratt, I.: HSA Queuing (2013).
[4] Advanced Micro Devices, Inc.: AMD Turbo Core Technology, http://www.amd.com/en-gb/innovations/software-technologies/turbo-core. (Visited on 01/21/2016).
[5] JEDEC: *DDR3 SDRAM standard (revision F)* (2012).
[6] Advanced Micro Devices, Inc.: AMD Radeon Memory - AMP Technology, http://www.radeonmemory.com/amp_technology.php (2014). (Visited on 01/21/2016).
[7] Daga, M., Aji, A. and chun Feng, W.: On the Efficacy of a Fused CPU+GPU Processor (or APU) for Parallel Comput-

ing, *Application Accelerators in High-Performance Computing (SAAHPC), 2011 Symposium on*, pp. 141–149 (online), DOI: 10.1109/SAAHPC.2011.29 (2011).

[8] Spafford, K. L., Meredith, J. S., Lee, S., Li, D., Roth, P. C. and Vetter, J. S.: The Tradeoffs of Fused Memory Hierarchies in Heterogeneous Computing Architectures, *Proceedings of the 9th Conference on Computing Frontiers*, CF '12, New York, NY, USA, ACM, pp. 103–112 (online), DOI: 10.1145/2212908.2212924 (2012).

[9] Jansson, J.: Integrated GPUs : how useful are they in HPC? (2013).

[10] NVIDIA Corporation: NVLink, Pascal and Stacked Memory: Feeding the Appetite for Big Data — Parallel Forall, https://devblogs.nvidia.com/parallelforall/nvlink-pascal-stacked-memory-feeding-appetite-big-data/. (Accessed on 07/13/2016).

[11] NVIDIA Corporation: Whitepaper NVIDIA Tesla P100.

[12] International Business Machines Corporation: IBM POWER8 CPU and NVIDIA Pascal GPU speed ahead with NVLink - IBM Systems Blog: In the Making, https://www.ibm.com/blogs/systems/ibm-power8-cpu-and-nvidia-pascal-gpu-speed-ahead-with-nvlink/. (Accessed on 07/13/2016).

[13] Pezy Computing: 4,096 コア規模で、8TFLOPS の次世代メニーコアプロセッサ「PEZY-SC2」開発の計画を発表。2016 年末にもタワーサーバーラック 1 台で 5PetaFLOPS の HPC システムを構成可能に (2015).