Chapter 3

Processor Research Team

3.1 Members

Kentaro Sano (Team Leader)

Tomohiro Ueno (Postdosctoral Researcher)

Takaaki Miyajima (Postdosctoral Researcher)

Jens Christoph Huthmann (Postdosctoral Researcher)

Artur Podobas (Postdosctoral Researcher)

Atsushi Koshiba (Postdosctoral Researcher)

Antoniette Pangilinan Mondigo (Student Trainee)

Kohei Hijikata (Student Trainee)

Kouki Watanabe (Student Trainee)

3.2 Overview of Research Activities

3.2.1 Aim of Team

The aim of the processor research team is to explore and establish data-flow-based parallel computing models and high-performance computer architectures which are promising and necessary as next-generation computing technologies in the forthcoming post-Moore era. We are researching and developing general-purpose processor architectures with their programming model different from existing multi/many-core processors, hardware and system software for reconfigurable computing, and any specific custom computing machines, as well as system software for data/task-flow-based high-performance computing to efficiently utilize large-scale parallel machines such as the supercomputer Fugaku.

In the next decade, sooner or later, the Moore's law as planar lithography-scaling is going to end and therefore we will be no longer able to rely on two-dimensional scaling of CMOS devices. In this "post-Moore" era, transistor integration, power consumption per transistor, and relative latency of data movement to the transistor's switching speed are not sufficiently improved. Consequently, it is predicted that the conventional approaches cannot increase the performance and performance per power, which have been so far improved mainly by the semiconductor scaling. Accordingly, we will need to more efficiently and effectively utilize available hardware resources, i.e., transistors on chips, to achieve target performance. In particular, the conventional many-core architectures and large-scale systems based on the parallel computing model and global synchronization will be confronted with limitation in increasing computing performance due to the following reasons:

1) dark silicon problem where most of transistors cannot be utilized due to the upper limit of on-chip power consumption. Since power per transistor does not decrease, we need to inactivate a large portion of transistors even if we have more transistors integrated on a chip,

2) relatively-increasing latency to transistor's switching speed. Due to the increasing latency, we can no longer shorten cycle time to update memory elements used for computing, cycle time to control computation based on some decision, and synchronization time among a large number of physically-distributed processor chips,

3) inefficient data-movement among on-chip cores via a memory subsystem or through a global network in a system, and

4) a large overhead in global synchronization for large-scale parallel computation which is affected by relatively-increasing delay of data transfer through a system-wide network.

That is, the existing approaches/architectures are not designed to scale the performance under these critical conditions. For example, the von-Neumann architecture is based on two cycles of "memory-element update" and "control" which cannot be accelerated any more, and therefore parallel processing is introduced as pipelining, super-scalar, and many cores as well as latency-hiding techniques of memory hierarchy with cache memories, speculative execution with branch prediction, and simultaneous multi-threading. These additional mechanisms make semiconductor resource utilization much worse for target processing and computing. In addition, the global barrier in parallel computation degrades overall performance as more nodes are utilized.

The custom computing/reconfigurable computing/spatially-mapped computing (spatial computing) allow us to efficiently utilize hardware resources for target computation while architectural overhead for reconfiguration, which can be seen in a field-programmable gate array (FPGA) device, is also suitable for the dark silicon problem. The spatial computing with the data-driven model (or data-flow model) allows us to avoid or mitigate cycles in processing. By spreading a sequence of operations onto space on hardware with a data flow, we can avoid instruction execution cycles with memory-element update and control so that we can increase a computing throughput with fine-grain parallelism increased naturally.

The data-flow or task-flow approach can also make it easier to avoid the global synchronization in large-scale parallel computing. If we automatically schedule and control task execution based on task-flow with task dependency, we can efficiently execute tasks without global synchronization when they become ready to be executed. Thus, we believe that the custom computing/reconfigurable computing/spatial computing with these localized control and synchronization is essentially necessary for future computer architectures in the post-Moore era, and therefore we are researching them.

Since researches on these themes require broad range of expertise, we are collaborating with other research teams in R-CCS, universities in Japan (Tohoku university, University of Tsukuba, Nagasaki university, Kumamoto university, Hiroshima city university, Kyoto university, and Japan advanced institute of science and technology; JAIST), and a research institute out of Japan, such as Argonne national laboratory, US.

3.2.2 Overview of Research Activities FY2019

Toward the aim of our team described above, we have conducted the following researches in the fiscal year of 2019.

1. Investigation and exploration of coarse-grained reconfigurable architectures (CGRAs)

2. Research and development of FPGA cluster

3. FPGA-based Applications

The background, motivation, objectives, and achievement of each research subject follow in the next section.

3.3 Research Results and Achievements

3.3.1 Investigation and exploration of coarse-grained reconfigurable architectures (CGRAs)

As the achievement which was mainly published in [2], we conducted architectural investigation of CGRA with past related work for the last three decades. With the end of both Dennard's scaling and Moore's law, computer users and researchers are aggressively exploring alternative forms of computing in order to continue the performance scaling that we have come to enjoy. Among the more salient and practical of the post-Moore alternatives are reconfigurable systems, with CGRAs seemingly capable of striking a balance between performance and programmability.

In this work [2], we surveyed the landscape of CGRAs. We summarized nearly three decades of literature on the subject, with a particular focus on the premise behind the different CGRAs and how they have evolved. Next, we compiled metrics of available CGRAs and analyze their performance properties in order to understand and discover knowledge gaps and opportunities for future CGRA research specialized towards High-Performance Computing (HPC). We found that there are ample opportunities for future research on CGRAs, in particular with respect to size, functionality, support for parallel programming models, and to evaluate more complex applications.

As the achievement which was mainly published in [5], we also conducted architectural exploration framework by developing a CGRA simulator. CGRAs are being considered as a complementary addition to modern High-Performance Computing (HPC) systems. These reconfigurable devices overcome many of the limitations of the (more popular) FPGA, by providing higher operating frequency, denser compute capacity, and lower power consumption. Today, CGRAs have been used in several embedded applications, including automobile, telecommunication, and mobile systems, but the literature on CGRAs in HPC is sparse and the field full of research opportunities.

In this work [5], we introduced our CGRA simulator infrastructure for evaluating future HPC CGRA systems. The simulator is built on synthesizable VHDL and is highly parameterizable, including support for connectivity, SIMD, data-type width, and heterogeneity. Unlike other related work, our framework supports co-integration with third-party memory simulators, DRAMSim3, or evaluation of future memory architecture, which is crucial to reason around memory-bound applications. We demonstrated how our framework can be used to explore the performance of various kernels, showing the impact of different configuration and design-space options.

3.3.2 FPGA cluster

We have been developed an FPGA Cluster of Fig.3.1 as a research platform for high-performance reconfigurable computing with FPGAs. We have conducted the following sub-topics for the FPGA cluster.

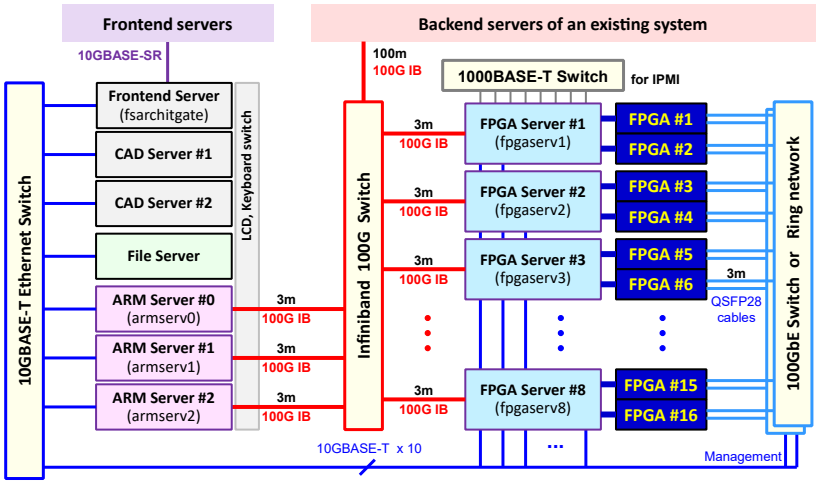

Figure 3.1: Organization of FPGA Cluster developed in Processor research team.

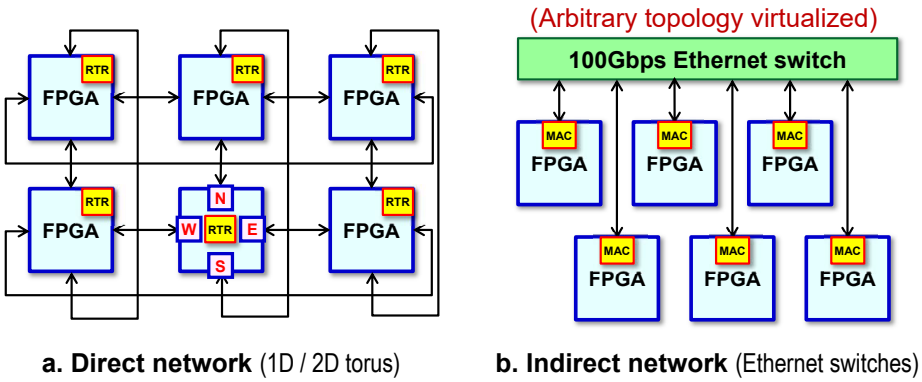

Figure 3.2: Two types of inter-FPGA netwotrs considered.

3.3.2.1 Research on inter-FPGA networks

As the achievement which was mainly published in [6], we conducted the following research on inter-FPGA networks of Fig.3.2. As FPGAs become a favorable choice in exploring new computing architectures for the post-Moore era, a flexible network architecture for scalable FPGA clusters becomes increasingly important in high performance computing (HPC). In this work [6], we introduced a scalable platform of indirectly-connected FPGAs, where its Ethernet-switching network allows flexibly customized inter-FPGA connectivity. However, for certain applications such as in stream computing, it is necessary to establish a connection-oriented data-path with back-pressure between FPGAs. Due to the lack of physical back-pressure channel in the network, we utilized our existing credit-based network protocol with ow control to provide receiver FPGA awareness and tailored it to minimize overall communication overhead for the proposed framework.

To know its performance characteristics, we implemented necessary data transfer hardware on Intel Arria 10 FPGAs, modeled and obtained its communication performance, and compared it to a direct network. Results showed that our proposed indirect framework achieves approximately 3% higher effective network bandwidth than our existing direct inter-FPGA network, which demonstrates good performance and scalability for large HPC applications.

As the achievement which was mainly published in [7], we also researched hybrid utilization approaches with an inter-FPGA network and a host server network. A tightly coupled FPGA cluster is a promising approach for large-scale parallel processing with application specialized hardware. Along with the advantages of FPGA-based custom computing, such as high power efficiency, a customized network subsystem with efficient communication through direct Inter-FPGA links allows an FPGA cluster to be an effective platform for large-scale parallel processing. However, the cluster can suffer from substantial communication costs when a cluster becomes larger to obtain higher computing performance.

In this work [7], we propose to exploit the communication capacity of a host server network to improve communication performance. Besides, we show estimations for practical communication patterns on a network model in which we efficiently use both the FPGA and the host networks.

3.3.2.2 Research on software-bridged FPGA driver

As the achievement which was mainly published in [9], we researched and developed a remoted FPGA driver as a software bridge through a general-purpose network. A heterogeneous system with FPGAs is gathering attention in High-Performance Computing (HPC) area. When FPGAs are used as an accelerator attached to the host CPU, there can be many configurations such as network topology to construct FPGA cluster. Sustained data transfer bandwidth between FPGA memory and CPU memory on a distant node is one of the most important factors to decide a topology of FPGA cluster. In order to explore the best topology, a quantitative evaluation of bandwidth is required.

In this work [9], we developed a remoted FPGA driver as a software bridge through a Infiniband EDR (100 Gbps) network, which is commonly used as a general-purpose system network in HPC systems. We conducted bandwidth measurement on two host nodes; both nodes are connected via 100 Gbps Infiniband cable and one host node has PCIe Gen3 x8-based FPGA accelerator card. We implemented a Direct Memory Access (DMA) function on an FPGA-attached node and a software bridged data transfer function to transfer data between two nodes. The result shows that DMA function and software bridged data transfer function achieve 82.2 %

and 69.6 % of the theoretical bandwidth of PCIe Gen3 x8, a bottleneck of data transfer path, respectively.

3.3.2.3 Research on high-level synthesis (HLS) compiler for FPGA

As the achievement which was mainly published in [3], we researched and developed an extension of an existing HLS compiler with visualization and profiling tool. The recent maturity in High-Level Synthesis (HLS) has renewed the interest of using FPGAs FPGAs to accelerate High-Performance Computing (HPC) applications. Today, several studies have shown performance- and power-benefits of using FPGAs compared to existing approaches for a number of application kernels with ample room for improvements. Unfortunately, modern HLS tools offer little support to gain clarity and insight regarding why a certain application behaves as it does on the FPGA, and most experts rely on intuition or abstract performance models.

In this work [3], we hypothesize that existing profiling and visualization tools used in the HPC domain are also usable for understanding performance on FPGAs. We extend an existing HLS tool-chain to support Paraver – a state-of-the-art visualization and profiling tool well-known in HPC. We describe how each of the events and states are collected, and empirically quantify its hardware overhead. Finally, we practically apply our contribution to two different applications, demonstrating how the tool can be used to provide unique insights into application execution and how it can be used to guide optimization.

As the achievement which was mainly published in [4], we also researched OpenMP-based task offloading for FPGA with an HLS compiler. Next to GPUs, FPGAs are an attractive target for OpenMP device offloading, as they allow to implement highly efficient, application-specific accelerators. However, prior approaches to support OpenMP device offloading for FPGAs have been limited by the interfaces provided by the FPGA vendors' HLS tool interface or their integration with the OpenMP runtime, e.g., for data mapping.

This work [4] presents an approach to OpenMP device offloading for FPGAs based on the LLVM compiler infrastructure and the Nymble HLS compiler. The automatic compilation flow uses LLVM IR for HLS-specific optimization and transformation and for the interaction with the Nymble HLS compiler. Parallel OpenMP constructs are automatically mapped to hardware threads executing simultaneously in the generated FPGA accelerator and the accelerator is integrated into `libomptarget` to support data-mapping. In a case study, we demonstrate the use of the compilation flow and evaluate its performance.

3.3.3 FPGA-based Applications

3.3.3.1 Highly-pipelined stream computation of Tsunami simulation with a ringed FPGAs

As the achievement which was mainly published in [1], we researched stream computation of Tsunami simulation with multiple FPGAs connected with a ring network. Since the hardware resource of a single FPGA is limited, one idea to scale the performance of FPGA-based HPC applications is to expand the design space with multiple FPGAs. In this work [1], we present a scalable architecture of a deeply pipelined stream computing platform, where available parallelism and inter-FPGA link characteristics are investigated to achieve a scaled performance.

For a practical exploration of this vast design space, a performance model was presented and verified with the evaluation of a tsunami simulation application implemented on Intel Arria 10 FPGAs. Finally, scalability analysis was performed, where speedup is achieved when increasing the computing pipeline over multiple FPGAs while maintaining the problem size of computation. Performance was scaled with multiple FPGAs; however, performance degradation occurred with insufficient available bandwidth and large pipeline overhead brought by inadequate data stream size.

Tsunami simulation results showed that the highest scaled performance for 8 cascaded Arria 10 FPGAs is achieved with a single pipeline of 5 stream processing elements (SPEs), which obtained a scaled performance of 2.5 TFlops and a parallel efficiency of 98%, indicating the strong scalability of the multi-FPGA stream computing platform.

3.3.3.2 Highly-pipelined stream computation of fluid simulation with a ringed FPGAs

As the achievement which was mainly published in [10], we researched stream computation of Fluid simulation with multiple FPGAs connected with a ring network. Stream computing is a suitable approach to improve both performance and power efficiency of numerical computations with FPGAs. To achieve further performance gain, temporal and spatial parallelism were exploited: the first one deepens and the latter duplicates pipelines of streamed computation cores. These two types of parallelism were previously evaluated with Arria 10 FPGA. However, it has not been verified if they are also effective for the latest FPGA, Stratix 10, which has a larger amount of logic elements (i.e., 2.4x of Arria 10) and is equipped with a new feature to improve the maximum

clock frequency (i.e., HyperFlex architecture). To show the scalability for such state-of-the-art FPGAs, in this paper, we firstly implemented a streamed fluid simulation accelerator with both parallelism types for Stratix 10.

We then thoroughly evaluated it by obtaining computational performance (FLOPS), power efficiency (FLOPS/W), resource utilization, and maximum clock frequency (Fmax). From the results, we found that this implementation excessively used DSP blocks due to inefficient mapping of floating-point operations, which reduced Fmax and the number of pipelined cores. To improve the scalability, we optimized the implementation to reduce the DSP block usage by utilizing a Multiply-Add function in a single DSP block. As a result, the optimized fluid simulation achieves 1.06 TFLOPS and 12.6 GFLOPS/W, which is 1.36X and 1.24X higher than the non-optimized version, respectively. Moreover, we estimate that the fluid simulation with Stratix 10 could outperform GPU-based implementation with Tesla V100 by optimizing it for HyperFlex architecture.

3.3.3.3 Scalable N-body stream computation with a ringed FPGAs

As the achievement which was mainly published in [8], we researched N-body stream computation with multiple FPGAs connected with a ring network. FPGAs offer a fairly non-invasive method to specialize custom architectures towards a specific application domain. Recent studies has successfully demonstrated that single-node FPGAs can be a rival to both CPUs and GPUs in performance. Unfortunately, most existing studies limit themselves to using a single FPGA devices, and their scalability requires more investigation.

In this work [8], we practically demonstrated how to scale the important n-body problem across a comparatively large FPGA cluster. Our design composed of up to 256 processing elements achieved near-linear strong scaling, with performance-levels comparable to that of custom Application-Specific Integrated Circuits (ASICs).

We further developed an analytical performance model, which we use to predict the performance of our solution onto future upcoming Intel Agilex FPGAs. Our system reached up to 47 Giga-Pairs/second, and using our performance model we predicted that we can reach up-to 0.142 Tera-Pairs/second peak performance with next-generation FPGAs.

3.4 Schedule and Future Plan

In addition to the researches and development done in FY2019, we are planning to conduct the following researches in the next fiscal year, some of which are newly started and some are continuous work to the present subjects.

1. Further exploration of CGRAs, and development of its place-and-route compiler. We will extend CGRAs and evaluate their performance by benchmarking with some computing kernels. For this, we will develop a compiler for our CGRAs.

2. Research and development of system hardware for FPGA cluster. We will develop a system-on-chip (SoC) on an FPGA device in the cluster, which is called AFU Shell. The developed AFU Shell will support fundamental data-movement among a host CPU and FPGA, and inter-FPGA networks of a direct and an indirect topologies. For the indirect topologies, SoC will provide a virtualized circuit switching mechanism on the top of the packet switching mechanism of 100 Gbps Ethernet.

3. Research and development of system software for FPGA cluster. We will develop an FPGA-object class library as a hardware-abstraction layer which allows us to easily use FPGAs. We are also developing a resource manager software for FPGA resources in the cluster. With this resource manager, we will be able to exclusively utilize a part of FPGA resources in a system with configuration of the Ethernet-based inter-FPGA network based on a request from a user program.

4. Research on utilization of FPGA cluster with an existing HPC machines without FPGAs. We will experimentally connect the FPGA cluster to the supercomputer Fugaku by using 100 Gbps Infiniband network and the software bridge of FPGA driver described in Section 3.3.2.2, so that we can offload tasks to the FPGA cluster from MPI processes running on Fugaku. This also demonstrates that the FPGA cluster with the software bridge is very flexible and available to extend no-FPGA machines.

5. Researches on more FPGA-based applications. We are going to develop and evaluate the following applications and benchmarks for FPGA cluster: 3D FFT, Genome sequence matching, stream computing of Fluid simulation in a different parallelism from deeper pipelining, breadth first search of a graph, and so on. We will mainly use Intel's HLS compiler to implement them.

3.5 Publications

3.5.1 Articles/Journal

[1] Antoniette Mondigo, Tomohiro Ueno, Kentaro Sano, and Hiroyuki Takizawa, "Scalability Analysis of Deeply Pipelined Tsunami Simulation with Multiple FPGAs," IEICE Transactions on Information and Systems(Special Section on Reconfigurable Systems), Vol.E102-D, No.5, pp.1029-1036,May. 2019.

[2] Artur Podobas, Kentaro Sano, and Satoshi Matsuoka, "A Survey on Coarse-Grained Reconfigurable Architectures from a Performance Perspective," IEEE Access, Vol.8, pp.146719-146743, DOI:10.1109/ACCESS.2020.3012084, 2020.

3.5.2 Conference Papers

[3] Jens Huthmann, Artur Podobas, Lukas Sommer, Andreas Koch, and Kentaro Sano, "Profiling and Visualizing Performance of FPGAs in High-Performance Computing Environments," Proceedings of IEEE International Conference on Cluster Computing (CLUSTER), pp.371-380, DOI: 10.1109/CLUSTER49012.2020.00047.

[4] Jens Huthmann, Lukas Sommer, Artur Podobas, Andreas Koch, and Kentaro Sano, "OpenMP Device Offloading to FPGAs using the Nymble Infrastructure," Proceedings of the 16th International Workshop on OpenMP (IWOMP), Vol.12295, pp.265-279, 2020.

[5] Artur Podobus, Kentaro Sano, and Satoshi Matsuoka, "A Template-based Framework for Exploring Coarse-Grained Reconfigurable Architectures," Proceedings of the 31st IEEE International Conference on Application-specific Systems, Architectures and Processors (ASAP), pp.1-8, DOI: 10.1109/ASAP49362.2020.00010, 2020.

[6] Antoniette Mondigo, Tomohiro Ueno, Kentaro Sano, and Hiroyuki Takizawa, "Comparison of direct and indirect networks for high-performance FPGA clusters," Applied Reconfigurable Computing. Architectures, Tools, and Applications (ARC 2020), Lecture Notes in Computer Science, Vol.12083, 2020.

[7] Tomohiro Ueno, Takaaki Miyajima, Antoniette Mondigo, Kentaro Sano, "Hybrid Network Utilization for Efficient Communication in a Tightly Coupled FPGA Cluster," Proceedings of 2019 International Conference on Field-Programmable Technology (FPT), pp. 363-366, December, 2019.

[8] Jens Huthmann, Shin Abiko, Artur Podobas, Kentaro Sano, and Hiroyuki Takizawa, "Scaling performance for N-Body Stream Computation with a ring of FPGAs," Proceedings of the International Symposium on Highly-Efficient Accelerators and Reconfigurable Technologies (HEART), Article No.10, 6 pages, 2019.

[9] Takaaki Miyajima, Tomoya Hirao, Naoya Miyamoto, Jeongdo Son, and Kentaro Sano, "A Software Bridged Data Transfer on a FPGA Cluster by Using Pipelining and InfiniBand Berbs," Proceedings of the International Symposium on Highly-Efficient Accelerators and Reconfigurable Technologies (HEART), Article No.11, 6 pages, 2019.

3.5.3 Posters / Papers with Abstract Review

[10] Atsushi Koshiba, Kouki Watanabe, Takaaki Miyajima, and Kentaro Sano, "Performance Evaluation and Power Analysis of Teraflop-scale Fluid Simulation with Stratix 10 FPGA," Proceedings of 28th ACM/SIGDA International Symposium on Field-Programmable Gate Arrays (FPGA2020), abstract for poster, 1 page, Feb 2020.

[11] Takaaki Miyajima, Tomohiro Ueno, Atsushi Koshiba, Jens Huthmann, Kentaro Sano, "High-Performance Custom Computing with FPGA Cluster as an Off-loading Engine," Proceedings of HPCAsia2020 (1 page), 2020.

[12] Atsushi Koshiba, Kentaro Sano, "System Software Support for Fast and Flexible Task Management on a Large-scale FPGA cluster," Proceedings of HPCAsia2020 (1 page), 2020.

[13] Takaaki Miyajima, Tomohiro Ueno, Atsushi Koshiba, Jens Huthmann, Kentaro Sano, and Mitsuhisa Sato, "High-Performance Custom Computing with FPGA Cluster as an Off-loading Engine", International Conference for High Performance Computing, Networking, Storage and Analysis, SC'19, Denver, CO, USA RESEARCH POSTER, 2019.

3.5.4 Invited Talks (keynote, plenary talk, invited talk, panelist position talk)

[14] Kentaro Sano, "Data-flow Compiler for Stream Computing Hardware on FPGA," LSPANC, RIKEN R-CCS, Kobe, Jan 29, 2020.

[15] Kentaro Sano, "Networks of FPGA Cluster with High Flexibility of Resource Allocation," SC19 Booth talk of University of Tsukuba, Colorado Convention Center, Nov 20, 2019.

[16] Kentaro Sano, "May Pipelining Be with You," SC19 Panel:Reconfigurable Computing in HPC: Success Stories Today and Future?, Colorado Convention Center, Nov 19, 2019.

[17] Kentaro Sano, "FPGA Cluster as Off-loading Engine for Existing Machines," SC19 BOF:Reconfigurable/FPGA Clusters for High Performance Computing, Colorado Convention Center, Nov 20, 2019.

[18] 佐野 健太郎, "柔軟な資源割当てを可能とするFPGAクラスタシステムとそのネットワーク," 第12回FPGAエクストリームコンピューティング (FPGAX12), 東工大大岡山キャンパスくらまえホール, November 12, 2019.

[19] Kentaro Sano, "High-Performance Custom Computing with FPGA Cluster as Off-loading Engine for Supercomputers," 11th International Symposium on Discovery, Fusion, Creation of New Knowledge by Multidisciplinary Computational Sciences (held by CCS, University of Tsukuba), Tsukuba International Congress Center, Tsukuba, Japan, Oct 15, 2019.

[20] 佐野 健太郎, "FPGAを用いたカスタムコンピューティングと高性能計算の将来", 第12回総合科学を考えるセミナー, 東北大学大学院情報科学研究科, 宮城県仙台市, 9月27日, 2019.

[21] Kentaro Sano, "FPGA Cluster as Off-loading Engine for Supercomputers," 1st International Workshop on Reconfigurable High-Performance Computing (ReHPC), in conjunction with FPL, BSC in Barcelona, Spain, Sep 13, 2019.

[22] Kentaro Sano, "FPGA Cluster as Custom Computing Engine for Supercomputers," 5th workshop on Programming Abstractions for Data Locality (PADAL), INRIA in Bordeaux, France, Sep 9-11, 2019.

[23] Kentaro Sano, "FPGA-based High-Performance Custom Computing based Dataflow Approach," Workshop on Post Moore's Law HPC Computing in conjunction with ISC'19 June 20, 2019.

[24] Kentaro Sano, "Stratix10 FPGA Cluster as Off-loaded Custom Computing Engine for Supercomputers," Workshop of Intel eXtreme Performance Users Group (IXPUG) in conjunction with ISC'19 June 20, 2019.

3.5.5 Other Publication Articles

[25] 佐野 健太郎, "ソフトなハードで高性能," 日刊工業新聞, 朝刊19面, 2020年2月3日.

[26] 上野知洋, 佐野 健太郎, "FPGAクラスタとその相互結合網の研究動向," 電子情報通信学会誌 解説記事, vol.103，no.4, pp.421-425, 2020.

[27] 佐野 健太郎, "ピンチの中に勝機あり," 産經新聞連載エッセイ 科学の中身, 2019年4月20日.

3.5.6 Oral Talks (with non-reviewed papers)

[28] 小柴 篤史, 上野 知洋, 佐野 健太郎, "Stratix 10 FPGAクラスタにおける格子ボルツマン法のパイプライン並列化と性能評価," 電子情報通信学会リコンフィギャラブルシステム研究会 信学技法, Vol.120, No.168, pp.7-12, Sep 10-11, 2020.

[29] 土方 康平, 上野 知洋, 江川 隆輔, 滝沢 寛之, 佐野 健太郎, ”ベクトルプロセッサからFPGA へのタスクオフロードに関する一考察,” 電子情報通信学会リコンフィギャラブルシステム研究会 信学技法, Vol.119, No.373, pp.7-11, Jan 22-23, 2020.

[30] 小柴 篤史, 佐野 健太郎, ”サーバレスコンピューティングにおけるハードウェアアクセラレータ仮想化機構の初期検討,” 第31回コンピュータシステム・シンポジウム(ComSys2019), poster paper, 2 pages, 2019.

[31] Jens Huthmann, Auter Podobas, Takaaki Miyajima, Atsushi Koshiba, and Kentaro Sano, ”Multi-threaded High-Level Synthesis for Bandwidth-intensive Application,” 電子情報通信学会リコンフィギャラブルシステム研究会 信学技法, Vol.119, No.208, pp.51-56, Sep 19-20, 2019.

[32] 上野 知洋, 佐野 健太郎, 土方 康平, 滝沢 寛之, ”RDMAを用いた密結合FPGAクラスタのメモリ間通信性能,” 電子情報通信学会リコンフィギャラブルシステム研究会 信学技法, Vol.119, No.18, pp.7-10, May 9-10, 2019.