Performance of Radix Sort using All-to-all Optical Interconnection Network in an Eight-FPGA Cluster

Kenji Mizutani[1, 2*], Yutaka Urino[2], Takanori Shimizu[2], Hiroshi Yamaguchi[2], Shigeru Nakamura[2], Tatsuya Usuki[2], Kiyo Ishii[1], Ryosuke Matsumoto[1], Takashi Inoue[1], Shu Namiki[1] and Michihiro Koibuchi[3]

1 National Institute of Advanced Industrial Science and Technology (AIST), Platform Photonics Research Center, 1-1-1 Umezono, Tsukuba, Ibaraki, 305-8568, Japan

2 Photonics Electronics Technology Research Association (PETRA), Tsukuba, Ibaraki, Japan

3 National Institute of Informatics (NII)2, Tokyo, Japan

**Corresponding author: mizutani.ken@aist.go.jp*

Abstract: This paper presents a high-throughput dataflow processing using all-to-all communication with eight FPGAs. We demonstrated a parallel radix sorting throughput of 37.2 GB/s for 32-bit key range and 16-GiB data size. © 2024 The Author(s)

1. Introduction

In the data-centric era, it is important to retrieve and analyze multiple data to extract value simultaneously. In a parallel computer system, all-to-all communication performance among compute nodes is frequently performed. PC clusters using modern reconfigurable FPGAs (Field Programmable Gate Arrays) equipped with High Bandwidth Memory 2 (HBM2) have received attention as a low-power computing platform. HBM2 has many memory channels, in contrast to a traditional DRAM. Inter-FPGA networks face a significant challenge in connecting many memory channels with all-to-all communications.

We have developed OPTWEB, a network mechanism that enables dataflow processing with all-to-all communication between FPGAs equipped with HBM2s. All FPGAs are fully connected using fiber optics. The multiple memory controllers of each FPGA are then directly connected to each other to achieve highly efficient all-to-all communication [1, 2]. In this study, we demonstrated a parallel radix sort speedup using dataflow processing with all-to-all communication.

2. A Custom FPGA Cluster

Figure 1 illustrates a block diagram of OPTWEB for an eight-FPGA with a fully connected network topology on an FPGA card [1]. *M*emory controllers (M#*k*) and network ports are provided, equal to the number of FPGAs, and connected by a distributed switch in the FPGA. After the distributed switches in all FPGAs fix their paths for an all-to-all communication, static and dedicated paths from source memories to destination memories are established over the fully connected network. All 64 memory channels distributed in the eight FPGAs are directly connected to each other via the dedicated paths. For example, M#7 in FPGA #0 is directly connected to memory M#0 in FPGA #7, and so on. The fully connected network enables these dedicated memory-to-memory paths. OPTWEB dynamically switches the communication between the application logic and a memory and that between the application logic and a remote memory of a different FPGA by updating the changeover switch. The command sequencer manages this operation without congestion.

3. Parallel Radix Sort

We describe our design of a parallel radix sort on the FPGA cluster. Figure 2 shows a process flow for an X-bit radix sort that consists of X/c c-bit counting sorts, which sorts data based on the key type. Counting sort is a computationally minimal algorithm, and high throughput sorting performance can be expected on FPGAs with finite hardware resources. In the first step, a global sort, namely key exchange among the FPGAs using all-to-all communication. In subsequent steps, a local sort within the FPGA is repeated. Figure 3 shows the block diagram of the sorting kernel in each FPGA. It consists of a counting sort block and a transpose block, which are shared by global/local sort. Figure 4 illustrates the behavior of the global sort using three FPGAs. Numbers from 0 to 35 indicate the initial sequence of data to be sorted. Three types of keys are assumed and indicated by color. Within the FPGAs, the data is arranged in order from the beginning of the memory space managed by each memory controller. First, the counting sorter sorts the keys by type. Second, all-to-all communication is performed on the results, and data with the same key is collected in each FPGA. Finally, the transposer arranges the data in order in the memory space handled by each memory controller. Transposition at the end of each sorting step provides a stable sort that maintains the order of the data. This series of data processing is executed in a single memory read/write operation as a dataflow processing.

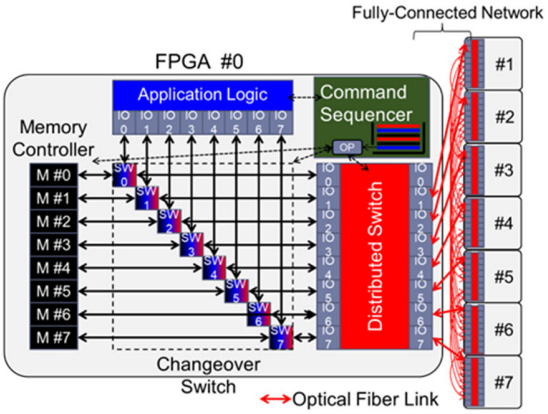

Figure 1: Block diagram of OPTWEB for a fully connected network topology on an FPGA card [3].

Figure 2: Flow of an X-bit radix sort that consists of X/c c-bit counting sorts.

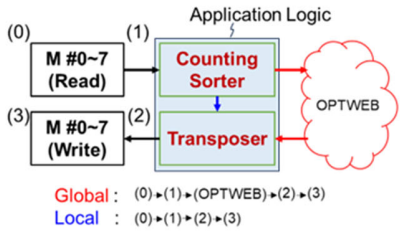

Figure 3: Block diagram of the sorting kernel.

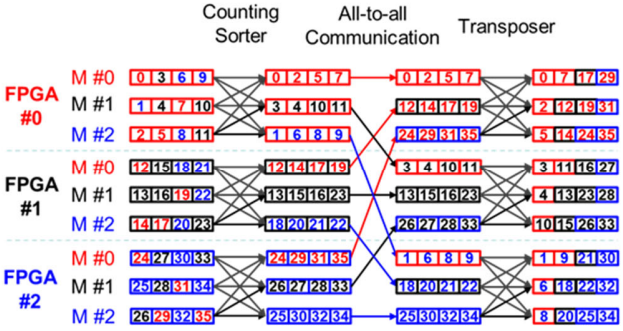

Figure 4: Behavior of the global counting sort.

4. Evaluation

The parallel radix sort circuit was implemented on an eight-node rack-server system, as shown in Figure 5 [3]. Each computing node using 1-U server has a CPU (Intel Xeon Gold 5215, 2.5 GHz) and a custom FPGA card, which has a custom FPGA (Intel Stratix10 MX2100) and eight embedded optical modules (EOMs). We used Intel Quartus Prime Pro version 19.4 to synthesize the FPGA circuit. The operating system was CentOS Linux v7.9.2009. The FPGAs are connected to each other with EOMs to a fully connected network using wavelength router technology. An optical transceiver chip in the EOM was fabricated using silicon photonics technology. The EOM has a bi-directional 100-Gbps throughput, which consists of four 25-Gbps Si photonics transceivers within a 12×12-mm^2 footprint. The EOM supports WDM in the C band, and couples to 12-core single-mode optical ribbon fiber. Eight network ports in each FPGA are connected to eight EOMs via electrical wires on the FPGA card. Wavelength routers can configure a fully connected network, using wavelength division multiplexing (WDM) technology to bundle multiple signals into a single optical fiber. A wavelength router and erbium-doped fiber amplifiers (EDFAs) are placed at the top of the rack. The optical equipment includes a WDM light source (Yokogawa AQ2200), and arrayed waveguide gratings (AWGs; NTT Electronics) for the wavelength router and (de)multiplexers.

We used a single 8-GB HBM2 memory in the evaluation. The HBM2 memory has eight memory channels. Each memory channel consists of two pseudo channels to simultaneously perform reading and writing through a memory channel. Each pseudo channel has a 256-bit width and 200-MHz clock frequency. Although the memory bandwidth is 51.2 Gbps per memory channel and 409.6 Gbps at maximum, any two FPGAs are connected to each other with a 50-Gbps link in our implementation, giving a 400-Gbps aggregate bandwidth per FPGA. The number of ALMs is not large enough to support 100-Gbps links and sorting kernels.

The key data to be sorted were 32-bit length unsigned integers without a payload. Namely, the maximum key range X was 32 bits. The number of bits c to be sorted for each Counting Sort was set to 4. We measured the sorting throughput by varying the key ranges from 4 bits to 32 bits. We also measured the sorting throughput by varying the number of keys from 2^{27} to 2^{32}. Namely, the key data size was from 0.5 GiB to 16 GiB (64 MiB to 2 GiB per FPGA). The key distribution was uniform, and the initial order was random. The keys were initially stored in HBM2 memories on the FPGAs and they were sorted to appropriate areas in the HBM2 memories on the FPGAs according to their key values.

Figure 6 (a) shows the measured sorting throughput of the parallel radix sort as a function of the key range on an FC topology. The sorting throughput is defined as the total data size of keys divided by the sorting processing time. While the key range varies from 4 to 32 bits, the key length is always 32 bits. The sorting throughput was 215.1 GB/s

for the 4-bit key range and 37.2 GB/s for the 32-bit key range. Figure 4(b) shows the measured throughput of the parallel radix sort as a function of the key data size. In this case, the key range was constant, 32 bits. The sorting throughput is almost constant regardless of the data size.

To the best of our knowledge, our parallel radix sort is the only one using many FPGAs with high-bandwidth memory networks. It is difficult to compare it with existing sorters in terms of architecture. Thus, we compared the levels of performance and support of multiple nodes of state-of-the-art sorters with CPUs, GPUs, and FPGAs, as listed in [4]. Two of them were distributed sorters with multiple CPU and GPU nodes, and the others were single-node sorters. Table 1 summarizes sorting throughputs per node. FPGA Bonsai is top-ranked with 5.814 GB/s/node. Ours is fourth-ranked with 4.646 GB/s/node and top-ranked within sorters with multi-node support. Using many FPGAs poses difficulties for both weak and strong scaling of sorters, because of the network-bandwidth bottleneck in accessing remote memories across multiple FPGAs. We optimized the sorting logic for OPTWEB to efficiently use the inter-FPGA networks. We achieved a high throughput per node comparable to that of a hardware sorter for a single FPGA.

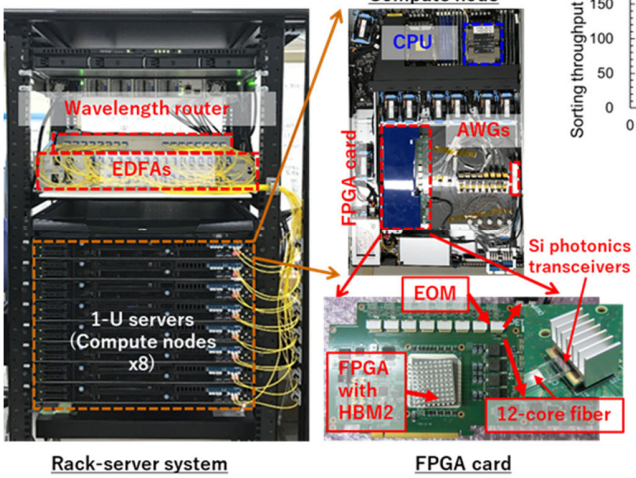

Figure 5: Eight-node server-rack system and custom FPGA card.

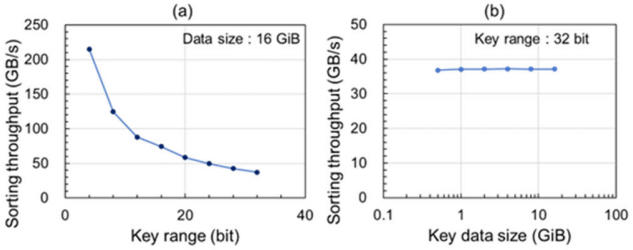

Figure 6: Sorting throughput
(a) as a function of key range,
(b) as a function of key data size.

Table 1: Quantitative comparison with existing sorters.

Sorter	Throughput (GB/s/node)	Parallel support
FPGA Bonsai	5.814	
GPU HRS	4.808	
FPGA SampleSort	4.651	
FPGA distributed (our work)	4.646	O
CPU PARADIS	2.755	
CPU distributed	2.146	O
GPU distributed	0.344	O
FPGA TerabyteSort	0.294	

5. Summary

We presented a parallel radix sort, which made the best use of all-to-all optical interconnection network on an eight-FPGA cluster. We obtained a parallel radix sorting throughput of 37.2 GB/s for a 32-bit key range and 16-GiB data size on the FPGA cluster.

[1] K. Mizutani et al., “OPTWEB: A Lightweight Fully Connected Inter-FPGA Network for Efficient Collectives”, IEEE Transactions on Computers, Volume: 70, Issue: 6, 849-862, 2021.

[2] K. Mizutani et al., “Accelerating parallel data processing using optically tightly coupled FPGAs”, Journal of Optical Communications and Networking, vol.14, no.2, pp.A166-A179, 2022.

[3] T. Shimizu et al., “Error-Free Operation for Fully Connected Wavelength-Routing Interconnect among 8 FPGAs with 2.8-Tbit/s Total Bandwidth”, Proc 2021 European Conference on Optical Communication (ECOC), pp.1-4, 2021.

[4] N. Samardzic et al., “Bonsai: High-performance adaptive merge tree sorting,” Proc. ACM/IEEE International Symposium on Computer Architecture (ISCA), pp. 282–294, 2020.