Performance Evaluation of a Vector Supercomputer SX-Aurora TSUBASA

Kazuhiko Komatsu
Tohoku University
Sendai, Miyagi 980–8578, Japan
Email: komatsu@tohoku.ac.jp

Shintaro Momose, Yoko Isobe, Osamu Watanabe, Akihiro Musa
Tohoku University, NEC Corporation
Minato-ku, Tokyo 108–8001, Japan
Email: {s-momose@ak, y-isobe@pi, o-watanabe@az, a-musa@bq}.jp.nec.com

Mitsuo Yokokawa
Kobe University
Nada-ku, Kobe 657–8501, Japan
Email: yokokawa@port.kobe-u.ac.jp

Toshikazu Aoyama
NEC Corporation
Minato-ku, Tokyo 108–8001, Japan
Email: t-aoyama@ap.jp.nec.com

Masayuki Sato, Hiroaki Kobayashi
Tohoku University
Sendai, Miyagi 980–8579, Japan
Email: {masa, koba}@tohoku.ac.jp

***Abstract*—A new SX-Aurora TSUBASA vector supercomputer has been released, and it features a new system architecture and a new execution model to achieve high sustained performance, especially for memory-intensive applications. In SX-Aurora TSUBASA, the vector host (VH) of a standard x86 Linux node is attached to the vector engine (VE) of the newly developed vector processor. An application is executed on the VE, and only system calls are offloaded to the VH. This new execution model can avoid redundant data transfers between the VH and VE that can easily become a bottleneck in the conventional execution model. This paper examines the potential of SX-Aurora TSUBASA. First, the basic performance is clarified by evaluating benchmark programs. Then, the effectiveness of the new execution model is examined by using a microbenchmark. Finally, the potential of SX-Aurora TSUBASA is clarified through evaluations of practical applications.**

I. Introduction

As supercomputing systems are now widely used in not only cutting-edge scientific research but also various industries, supercomputing systems have become important social infrastructures. To respond to the computational demands brought on by the expanded utilization of supercomputing systems, many advanced techniques are required to ensure high computational performance.

For example, recent processors and accelerators utilize many-core technology, which integrates to thousands of cores on a chip. In addition, vector calculation techniques are adopted even in scalar processors and accelerators as well as in vector processors. Thanks to these techniques, the theoretical computing performance of the world's fastest supercomputing system in the TOP 500 list has reached 122.3 Pflop/s in June 2018 [1].

However, as these techniques only enhance computational capability, there is a gap between theoretical performance and sustained performance, which is known as the *memory wall problem* [2]. Since improving memory performance is more difficult than improving computational performance, this gap is growing wider and wider. In other words, only compute-intensive applications that require high computational performance rather than memory performance stand to benefit from the high theoretical peak performance of recent supercomputing systems. Moreover, memory-intensive applications limited by the lower memory performance have lower sustained performance. Vector supercomputing systems that are designed for high memory performance can achieve high sustained performance even in memory-intensive applications [3][4].

Although vector supercomputing systems have occupied smaller portions of annual TOP500 lists, which mostly measures the theoretical peak performance, vector supercomputing systems do achieve very high sustained performance with respect to the percentage of peak performance, called efficiency, in the ranking of the HPCG benchmark [5][6][7]. HPCG was developed as a more practical benchmark closer to the characteristics of practical applications. In the benchmark, not only computational performance, but also memory and network performance are important. While the efficiencies of Intel-based and accelerator-based systems are only a few percent of the peak performance on the HPCG benchmark, the SX-ACE vector system, for example, exceeds 10%. The high memory bandwidth of vector supercomputing systems brings its high efficiency and high sustained performance.

Against such background where memory performance has started to gain more and more attention, a new vector supercomputer called *SX-Aurora TSUBASA*, which has the world's highest memory bandwidth, has been released [8][9]. SX-Aurora TSUBASA is designed to meet two key requirements: high usability and high sustained performance. To meet these requirements, SX-Aurora TSUBASA incorporates a new system architecture and new memory integration technology.

To achieve high usability, the minimum configuration, i.e., a node, of SX-Aurora TSUBASA utilizes a system architecture that consists of a vector engine (VE) equipped with a newly developed vector processor and a vector host (VH) of a standard x86 Linux node. Although the system architecture is similar to those of conventional accelerators, its execution model is completely different. The VE is responsible for executing an entire application, while the VH is used for processing system calls invoked by the application. Thanks to this

SC18, November 11-16, 2018, Dallas, Texas, USA
978-1-5386-8384-2/18/$31.00

TABLE I
SPECIFICATIONS OF THREE TYPES OF VE.

VE Type	Frequency	Number of cores	SP performance of a core	SP performance of a processor	DP performance of a core	DP performance of a processor	Memory bandwidth	Memory capacity
10A	1.6 GHz	8	614.4 Gflop/s	4915.2 Gflop/s	307.2 Gflop/s	2457.6 Gflop/s	1228.8 GB/s	48 GB
10B	1.4 GHz	8	537.6 Gflop/s	4300.8 Gflop/s	268.8 Gflop/s	2150.4 Gflop/s	1228.8 GB/s	48 GB
10C	1.4 GHz	8	537.6 Gflop/s	4300.8 Gflop/s	268.8 Gflop/s	2150.4 Gflop/s	750.0 GB/s	24 GB

new execution model, the system architecture has two major advantages over conventional accelerators. The first is that frequent data transfers between the VE and its corresponding VH, which tend to easily become a bottleneck in conventional accelerators, can be avoided, since an application is mainly executed on the VE. The second advantage is that no special programming is required. As an application is executed on the VE, explicit specification of the computational kernel to be executed on the VE is not necessary at all. In addition, programmers do not need to take care of the system calls to be executed on the VH. Generally, system calls in an application are transparently offloaded to the VH. In this way, the new execution model of SX-Aurora TSUBASA solves the problem of the conventional execution model of systems with accelerators and is able to deliver high usability.

Moreover, to achieve high sustained performance, SX-Aurora TSUBASA provides the world's highest memory bandwidth by introducing a new memory integration technology. Taiwan Semiconductor Manufacturing Company Ltd. (TSMC), Broadcom, and NEC collaborated to develop an advanced high bandwidth memory (HBM) packaging technology that enables six HBM2 memory modules to be integrated on the CPU package of a VE. As a result, it delivers the world's highest memory bandwidth, and it enables the multiple vector cores of the CPU to deliver high sustained performances in practical applications.

This paper clarifies the potential of the SX-Aurora TSUBASA vector supercomputer. Its performance is examined using fundamental benchmark programs as well as on practical application kernels. Then, its new execution model is evaluated. In particular, on a microbenchmark including vector-friendly calculations, scalar-friendly calculations, and system calls, the performance of the new execution model is compared with that of the conventional execution model of accelerators. Next, the performance of SX-Aurora TSUBASA is examined in practical applications of running tsunami simulation code and turbulent flow direct numerical simulation (DNS) code. This paper also analyzes performance bottlenecks by using different ratios involving the memory bandwidth and computation throughput. The analysis will be helpful for identifying bottlenecks and performance tuning. The results of these evaluations and analyses demonstrate that SX-Aurora TSUBASA achieves both high sustained performance and high usability through the new system architecture and its execution model.

The rest of this paper is organized as follows. Section II gives an overview of SX-Aurora TSUBASA. The system architecture of SX-Aurora TSUBASA, the architecture of the VE, and its new execution model are described. Section III starts by evaluating the performance of SX-Aurora TSUBASA by using benchmark programs and kernels. Then, the sustained memory bandwidth and sustained performance are evaluated. In Section IV, the new execution model is evaluated by using a microbenchmark. Section V discusses the sustained performance of practical applications. Section VI concludes this paper with a brief summary.

II. OVERVIEW OF SX-AURORA TSUBASA

NEC has been pursuing superior sustained performance, especially for memory-intensive scientific applications, since its rollout in 1983 of the SX series with dedicated vector processors. As the successor to SX-ACE, SX-Aurora TSUBASA inherits the design concepts from the SX vector architecture and improve its performance aiming at higher sustained performance and usability [8]. In particular, it has a new vector processor and improved efficiency in terms of both power usage and space. SX-Aurora TSUBASA has one fifth of the power consumption and takes up just one tenth of the floor space of the SX-ACE [9].

The system architecture of SX-Aurora TSUBASA is different from its predecessors in the SX series. The system mainly consists of a VH and one or more VEs. The VH is a standard x86 Linux server that provides standard operating system (OS) functions. *VE OS* is an OS for VEs that runs on a VH, and controls VEs. Each VE is implemented as a PCI Express (PCIe) card equipped with the newly developed vector processor, and it is connected to the VH. Three types of VE (Type 10A, 10B, and 10C) are provided in accordance with the clock frequency and memory configuration, as shown in Table I. Type 10A focuses on high arithmetic performance. Type 10B is a baseline model focusing on memory-intensive applications. Type 10C is an entry model.

As the system architecture of SX-Aurora TSUBASA mainly consists of a VE and a number of VHs, it has high level of flexibility when it comes to configuration. The SX-Aurora TSUBASA series includes not only a large-scale supercomputer, but also a small configuration similar to a personal computer. There are three product series: A100 series, A300 series, and A500 series. The A100 series is a workstation model with one VE per VH; it is the smallest in the SX-Aurora TSUBASA line up. The A300 series is a standard rack-mount model with the usual air cooling. Up to eight VEs can be configured with one VH. The A500 series is designed as a large-scale supercomputer. Up to eight VEs can be configured with one VH, and up to eight VHs are implemented per rack with direct liquid cooling.

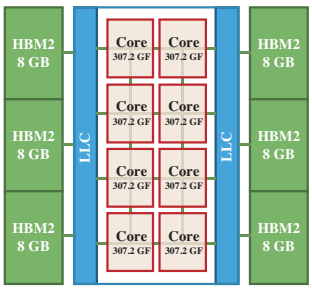

Fig. 1. Block diagram of a vector processor.

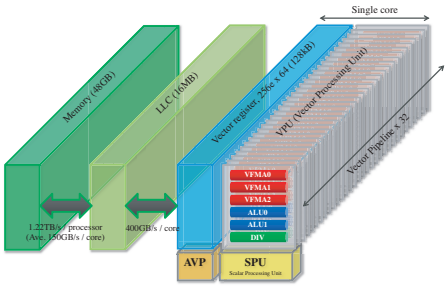

Fig. 2. Detail of a core in the VE processor.

A. Architecture of a Vector Engine

As the characteristics of scientific applications grow more diverse, sustained performance depends strongly upon not only the theoretical peak computing performance of a system but also its memory bandwidth. The design target of the VE processor is high sustained performance for applications limited by memory performance through provision of a high memory bandwidth at a reasonable power efficiency. Thus, six HBM2 modules are connected to the processor. This is the first implementation in the world, and NEC developed the implementation in collaboration with TSMC, Ltd., and Broadcom Inc. by using TSMC's chip on wafer on substrate (CoWoS) technology [10]. Thanks to the availability of such a cutting-edge technology, the processor currently provides the world's highest memory bandwidth, 1.22 TB/s per processor.

Figure 1 shows a block diagram of the VE processor. It consists of eight powerful vector cores, a 16 MB last-level cache (LLC), and six HBM2 memory modules. Each core provides 307.2 Gflop/s for double-precision (DP) and 614.4 Gflop/s for single-precision (SP) floating-point calculations. Overall, each VE processor having eight cores provides up to 2.45 Tflop/s (DP) and 4.91 Tflop/s (SP) worth of floating-point performance. The LLC is on both sides of the cores, and it is connected to each core through a two-dimensional network with a total cache bandwidth of 3 TB/s.

This processor is manufactured with 16-nm FINFET process technology. Approximately 4.8 billion transistors are integrated into the LSI, which has an area of 14.96 mm by 33.00 mm and runs at up to 1.6 GHz.

Figure 2 illustrates a single core and its registers, cache, and memory hierarchy. The core consists of three major units: a scalar processing unit (SPU), a vector processing unit (VPU), and a memory addressing vector control and processor network unit (AVP). The SPU works as a core controller. It is tightly coupled with both the VPU and the AVP. Although OS functions are provided by the VH in SX-Aurora TSUBASA, each core of the VE processor has almost the same functionality as a modern processor. Its functions include instruction fetch, decode and execution, instruction branch, detection of exception, memory protection, and so on. Most of these functions are supported by the SPU.

One of the advantages of this architecture is the VPU organization. The VPU has three vector fused multiply add (VFMA) units, which can be independently operated by different vector instructions. Each VFMA unit has 32 vector pipelines (VPP). The vector length (VL) of this architecture is 256 elements, each of which is 8 B. Therefore, one vector instruction executes 256 arithmetic operations with eight clock cycles. Unlike a wider SIMD implementation in major scalar processors, executing operations in not only spatially parallel, but also temporally parallel can hide memory access latency.

As shown in Figure 2, the number of vector registers is 64. The size of each vector register is 2 KB, because 256$\times$ 8B elements that is the same as the VL are held in a vector register. Therefore, the total size of the vector registers is 128 KB per core. This capacity is larger than a L1 cache implemented in modern scalar processors. The LLC is directly connected to the vector register, and the connection has a 400 GB/s bandwidth. Because of the high bandwidth design between memory and each core, three or four cores can use the entire memory bandwidth of this processor. This is a one of the features to accelerate memory-intensive applications.

The VE processor package is implemented on a PCIe card with a Gen3 $\times$16 connection. The card is designed so that the power consumption does not exceed 300 W.

B. Execution Model of SX-Aurora TSUBASA

The VE is entirely responsible for executing applications, while the VH provides OS functions such as process scheduling and handling of system calls invoked by the application running on a VE. Unlike a conventional accelerator, the one used by SX-Aurora TSUBASA does not require frequent data transfers between the VE and its VH. Furthermore, programmers do not need to care about interactions between the VH and VE, because they are handled transparently by system calls forwarded to a VH.

Figure 3 shows the software stack. The design and implementation of the VE OS were inspired by prior HPC operating systems for heterogeneous systems [11][12]. In these systems, system calls made by an application process are offloaded to their corresponding pseudo processes on another Linux host or core. In contrast, in the SX-Aurora TSUBASA, there is no OS kernel on a VE. Instead, VE OS modules on the VH provide the OS functionality for the VE. The VE OS modules consist of the *ve_exec* command and the *VE OS service*. The *ve_exec* loads a VE program, requests permission to create

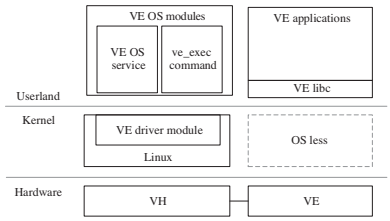

Fig. 3. Software stack of SX-Aurora TSUBASA.

a VE process, and serves the system calls and exceptions of the VE process. The VE driver installed in the VH Linux kernel space is a PCI device driver that provides VE resource accessibility and handles interrupts from the VEs.

A C library (libc) compliant with standards (e.g., ISO C99, C11, and POSIX including pthread) is ported to a VE. Application programs for the VE can use the standard libc. Therefore, programmers can write application programs in standard programming languages such as C, C++, and Fortran without having to use special programming models. SX-Aurora TSUBASA does not require codes of existing application programs to be ported as the only thing that needs to be performed is recompile.

To execute a VE program, users invoke *ve_exec* with a program path and its arguments. *ve_exec* traverses the header of an executable and linkable format (ELF) and the program header; then it reads each segment of the program into the buffer on *ve_exec*. Next, *ve_exec* requests the VE OS service to create a VE process and to send the segments to the VE process memory. At the same time, *ve_exec* creates an alternative process on the VH, called a *pseudo process*. Basically, a virtual address space of a VE process is homologized to that of a pseudo process. Whenever any memory manipulation is requested, the pseudo process first allocates a page on the virtual address space and then requests the VE OS service to allocate a page on the physical address space on the VE. Each VE core has an address translation buffer to handle virtual address translation and protection. The VE OS service appropriately sets the page table entry to the buffer.

Handling exceptions on the VE process involves some overhead compared with a traditional system call, since a system call, which is a kind of exception, is executed in the VH through the PCI bus. Since a PCI read is much slower than a PCI write from a latency point of view, PCI reads should be avoided during system calls in order to minimize overhead. Thus, a special hardware instruction called the *store host memory (SHM)* instruction is implemented. The VE process writes the ID of a system call and its arguments on the corresponding pseudo process memory by using the SHM instruction. Then, it invokes a monitor call (MONC) instruction, which stops the VE core and sends an interrupt to the VH. Then, the interrupt handler of the VE driver wakes up the pseudo process corresponding to the VE process. The pseudo process determines the cause of the exception. If it is caused by an illegal operation, the pseudo process sends an appropriate signal to itself and aborts both the VE process and the pseudo process. If the cause is the MONC instruction, the pseudo process determines which system call has been requested by reading the memory written by the VE process. Then, the pseudo process handles the system call and returns the result to the VE process. In the case that a socket connecting the VE card is different from the one executing the VE OS service, system calls require additional overhead due to inter-socket communication. However, this overhead is negligible, because the latency between sockets is much smaller than that of the PCI bus.

III. Performance Evaluation of SX-Aurora TSUBASA using Benchmark Programs

This section describes the performance of a node of the new vector supercomputer through evaluations of fundamental benchmark programs and application kernels. The analysis uses four different ratios involving the number of floating point operations to the amount of data, known as the bytes per flop (B/F) ratio: *memory B/F*, *LLC B/F*, *code B/F*, and *actual B/F*. The memory B/F ratio is defined as the memory bandwidth of the system divided by the theoretical computing performance. The LLC B/F ratio is defined as the LLC bandwidth of the system divided by the theoretical computing performance. The code B/F ratio is defined as the necessary data in bytes divided by the floating operations in a code. The code B/F ratios are computed by counting the number of memory operations and floating-point operations in an object code. The actual B/F ratio is calculated from the number of block memory accesses and actual floating-point operations that can be obtained only from the profile data [4]. The actual B/F ratio reflects the actual behavior of the system.

A. Experimental Environments

An A300-2 was used for the evaluation of one VE, while an A300-8 was used for the evaluation of multiple VEs. The specifications of the VEs (Type 10B) are listed in Table II. As the frequency of the VE is 1.40 GHz, the theoretical computing performance of its eight vector cores is 2.15 Tflop/s in terms of DP floating-point operations. The six HBM2 memories in each VE provide a 1.22 TB/s memory bandwidth. Each VE is equipped with a 16 MB LLC with a 2.66 TB/s cache bandwidth. The memory B/F and LLC B/F ratios are $\frac{1.22TB/s}{2.15Tflop/s} = 0.57$ and $\frac{2.66TB/s}{2.15Tflop/s} = 1.24$, respectively.

The VH of SX-Aurora TSUBASA is an Intel Xeon processor of the Skylake generation. As listed in Table II, the Intel Xeon Gold 6126 has a theoretical computing performance of 998.4 Gflop/s in DP floating-point operations. The VH is also equipped with six DDR4 DRAMs with a 128 GB/s memory bandwidth. The memory B/F of the VH is 0.13.

Table III lists the software environment of SX-Aurora TSUBASA. NEC compilers are used for generating object code for the VEs. The compilers basically apply vectorization to parallelisms of the innermost loop. If a vector is not long enough, the compilers try to enlarge the vector by collapsing the outer loops.

TABLE II
SPECIFICATIONS OF PROCESSORS USED IN THE EVALUATIONS.

Processor	Vector Engine Type 10B	Intel Xeon Gold 6126	SX-ACE
Frequency	1.40 GHz	2.60 GHz	1.0 GHz
Performance / core	537.6 Gflop/s (SP) 268.8 Gflop/s (DP)	166.4 Gflop/s (SP) 83.2 Gflop/s (DP)	64.0 Gflop/s (SP/DP)
Number of cores	8	12	4
Performance / socket	4.30 Tflop/s (SP) 2.15 Tflop/s (DP)	1996.8 Gflop/s (SP) 998.4 Gflop/s (DP)	256 Gflop/s (SP/DP)
Memory subsystem	HBM2 × 6 modules	DDR4-2666 DIMM × 6 channels	DDR3-2000 DIMM × 16 channels
Memory bandwidth	1.22 TB/s	128 GB/s	256 GB/s
Memory capacity	48 GB	96 GB	64 GB
LLC bandwidth	2.66 TB/s	N/A	1.0 TB/s
LLC capacity	16 MB shared	19.25 MB shared	1 MB private

Processor	NVIDIA Tesla V100	Intel Xeon Phi 7290	Intel Xeon Phi 7210
Frequency	1.245 GHz	1.50 GHz	1.30 GHz
Performance / core	27.343 Gflop/s (SP) 13.671 Gflop/s (DP)	96 Gflop/s (SP) 48 Gflop/s (DP)	83.2 Gflop/s (SP) 41.6 Gflop/s (DP)
Number of cores	5120	72	64
Performance / socket	14 Tflop/s (SP) 7 Tflop/s (DP)	6.912 Tflop/s (SP) 3.456 Tflop/s (DP)	5.324 Tflop/s (SP) 2.662 Tflop/s (DP)
Memory subsystem	HBM2	MCDRAM + DDR4-2400 × 6 channels	MCDRAM + DDR4-2133 × 6 channels
Memory bandwidth	900 GB/s	N/A (MCDRAM), 115.2 GB/s(DDR)	N/A(MCDRAM), 102 GB/s(DDR)
Memory capacity	16 GB	16 GB(MCDRAM), 96GB(DDR)	16 GB(MCDRAM), 96GB(DDR)
LLC bandwidth	N/A	N/A	N/A
LLC capacity	L2 6 MB shared, L1 128 KB private	1 MB shared by two cores	1 MB shared by two cores

TABLE III
SOFTWARE ENVIRONMENTS OF SX-AURORA TSUBASA.

Operating System	CentOS Linux release 7.3.1611
NEC Fortran compiler for Vector Engine	nec-nfort 1.3.0
NEC C/C++ compiler for Vector Engine	nec-nc++ 1.3.0
NEC VE OS	veos 1.2.1
NEC MPI	NEC MPI Version 1.1.1
Intel compiler	18.0.2 20180210

Table II also lists the specifications of the previous SX-ACE for performance comparison. SX-ACE has a theoretical computing performance of 256 Gflop/s with a 256 GB/s memory bandwidth. Each of the four vector cores in one SX-ACE node has a 1 MB assignable data buffer (ADB) with a 1 TB/s bandwidth between the core and the ADB. The memory B/F ratio is 1.00. Furthermore, the table also lists the specifications of the NVIDIA Tesla V100 and Intel Xeon Phi (KNL) 7290 and 7210. In particular, the theoretical computing performances of V100, KNL 7290, and KNL 7210 in DP operations are 7 Tflop/s, 3.456 Tflop/s, and 2.662 Tflop/s, respectively. The V100 provides a 900 GB/s memory bandwidth by HBM2, and its memory B/F is 0.13. The KNL 7290 and KNL 7210 have MCDRAM that provides a memory bandwidth of more than 450 GB/s.

Matrix-matrix multiplication, the Stream benchmark [19], and the Himeno benchmark [20] were used to evaluate the performance of the SX-Aurora TSUBASA node. Furthermore, as a more practical evaluation of performance, kernels of actual applications that were run on the supercomputing systems at Tohoku University [3] were used. Table IV gives an overview of the six kernels including their methods, memory access patterns, mesh sizes, and code B/F ratios. As the kernels include a wide variety of code B/F ratios, they are suitable for the evaluating supercomputing systems.

B. Performance of Benchmark Programs

Figure 4 shows the sustained performance of SP and DP matrix-matrix multiplications (SGEMM and DGEMM, respectively). SGEMM and DGEMM can be used as measures of the efficiency of the theoretical computing performance. The x-axis indicates the number of threads. The left y-axis indicates the sustained performance, and the right axis shows efficiency as a percentage of peak theoretical performance.

As shown in the figure, the sustained performances for SGEMM and DGEMM are almost the same as the theoretical peak performances in all threads. As the number of threads increases, almost ideal core scalability is realized. This is because the computational performance of the SX-Aurora TSUBASA can be fully exploited, and this results in very high efficiencies of 97.8% to 99.2%.

Figure 5 shows the sustained bandwidths of SX-Aurora TSUBASA, SX-ACE, and Skylake on Triad in the Stream benchmark. The x-axis indicates the number of threads. The y-axis indicates the stream memory bandwidth. The maximum number of threads is limited by the number of cores integrated in each processor. The array size in the Stream benchmark was selected to be sufficiently larger than the cache size.

As shown in the figure, SX-Aurora TSUBASA achieves a 995 GB/s stream memory bandwidth at maximum, while SX-ACE and Skylake achieve 211 GB/s and 85 GB/s, respectively. In addition, the stream bandwidths of V100 and KNL 7290 were measured. V100 and KNL 7290 achieves 727 GB/s and 446 GB/s, respectively. The stream bandwidth of the SX-Aurora TSUBASA is thus 4.72, 11.7, 1.37, and 2.23 times higher than those of SX-ACE, Skylake, V100, and KNL 7290, respectively. The high memory I/F to six HBM2 modules and its new implementation technology contribute to high sustained memory bandwidth.

Moreover, the efficiency as a percentage of the theoretical memory bandwidth of the SX-Aurora TSUBASA, SX-ACE,

TABLE IV
OVERVIEW OF TOHOKU UNIVERSITY KERNELS.

Applications	Fields	Methods	Memory access characteristics	Mesh size	Code B/F	Actual B/F
Land mine [13]	Electromagnetic	FDTD	Sequential memory access	100 × 750 × 750	6.22	5.79
Earthquake [14]	Seismology	Friction Law	Sequential memory access	2047 × 2047 × 256	6.00	2.00
Turbulent flow [15]	CFD	Navier-Stokes equation	Sequential memory access	512 × 16384 × 512	1.91	0.35
Antenna [16]	Electromagnetic	FDTD	Sequential memory access	252755 × 9 × 97336	1.73	0.98
Plasma [17]	Physics	Lax-Wendroff	Indirect memory access	20,048,000	3.02	0.075
Turbine [18]	CFD	LU-SGS method	Indirect memory access	480 × 80 × 80 × 10	0.77	0.0086

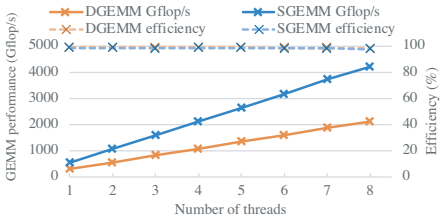

Fig. 4. Performance of GEMM.

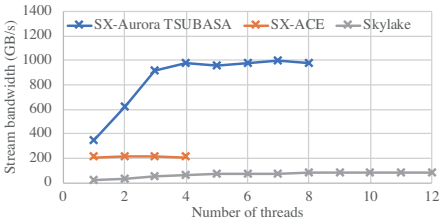

Fig. 5. Performance of the Stream benchmark.

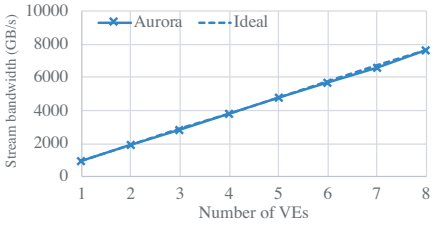

Fig. 6. VE-level scalability of the Stream benchmark.

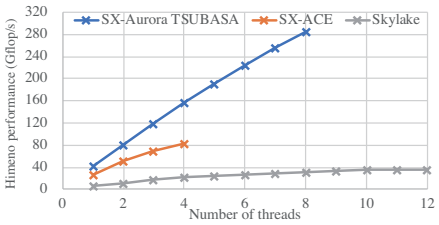

Fig. 7. Socket performance of Himeno.

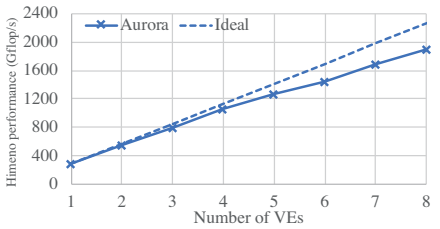

Fig. 8. Scalability of Himeno using multiple VEs.

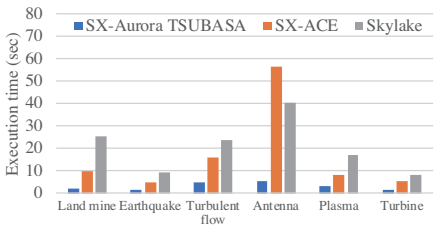

Fig. 9. Execution time of Tohoku University kernels.

Skylake, and V100 are 79%, 83%, 66%, and 81%, respectively. The vector supercomputers are more efficient than Skylake. Having a design that strongly focuses on memory performance is one of the factors contributing to the high sustained memory performance of the vector supercomputers.

Figure 6 shows the VE-level scalability of SX-Aurora TSUBASA. The x-axis indicates the number of VEs. The y-axis indicates the stream memory bandwidth. The stream bandwidth was measured by using the MPI version in the Stream benchmark. This figure shows good VE-level scalability according to the number of VEs. As the number of VEs increases, the aggregated memory bandwidth increases.

Figure 7 shows the performance on the Himeno benchmark with the XL data size of 1024 × 512 × 512. The x-axis indicates the number of threads. The y-axis indicates the performance. The Himeno benchmark measures the performance of solving the Poisson equation by using the Jacobi iteration method. In this benchmark, 19-point stencil-like accesses to the array p and accesses to the coefficient and work arrays a, b, c, bnd, $wrk1$, and $wrk2$, are necessary for each loop iteration.

As shown in the figure, SX-Aurora TSUBASA outperforms SX-ACE and Skylake. In the case of the maximum number of threads, SX-Aurora TSUBASA achieves 3.4 and 7.96 times higher Himeno performance than SX-ACE and Skylake, respectively. Moreover, the Himeno performances of V100 and KNL 7290 were measured. V100 and KNL 7290 achieve 305 Gflop/s and 137 Gflop/s, respectively. As the performance of SX-Aurora TSUBASA is 286 Gflop/s, SX-Aurora TSUBASA achieves 0.93 and 2.1 faster than V100 and KNL 7290, respectively. Although the peak performance of V100 is 3.26 times higher than that of SX-Aurora TSUBASA, SX-Aurora TSUBASA achieves almost the same performance of V100. The high memory bandwidth of SX-Aurora TSUBASA contributes the high sustained performance since the code B/F ratio of the Himeno benchmark is 3.73. However, as the stream bandwidth of SX-Aurora TSUBASA is higher than that of V100, there might be still room for further improvement of the performance in the Himeno benchmark.

Figure 8 shows the VE-level scalability of SX-Aurora TSUBASA on the Himeno benchmark. The x-axis indicates the number of VEs. The y-axis indicates performance as measured using the MPI version in the Himeno benchmark. Here, performance is almost proportional to the number of VEs for up to four VEs. For four or more VEs, the overhead of the reduction calculation and the insufficient vector length adversely affect the VE-level scalability.

Figure 9 shows results for the six kernels of the practical applications listed in Table IV. The x-axis indicates the six kernels in descending order of the code B/F ratio. The y-axis indicates the execution time. As shown in the figure, SX-Aurora TSUBASA achieves the shortest execution time on all six kernels.

To investigate the performance of these kernels in more detail, actual B/F ratios calculated from the profile data are utilized in addition to the code B/F ratios obtained from the object codes of the kernels. For example, when data stored in LLC are transferred to a processor, the number of memory

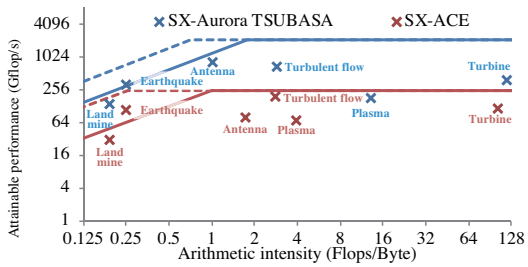

Fig. 10. Analysis of Tohoku University kernels by using the roofline model.

accesses decreases. In this way, the actual B/F ratio calculated from the number of actual memory accesses can take into account the actual behavior of LLC.

As listed in Table IV, the actual B/F ratios of SX-Aurora TSUBASA for *Land mine*, *Earthquake*, *Plasma*, *Turbulent flow*, *Antenna*, and *Turbine* are 5.79, 2.00, 0.075, 0.35, 0.98, and 0.0086, respectively. All actual B/F ratios are lower than the corresponding code B/F ratios. This relative reduction implies that the LLC of SX-Aurora TSUBASA helps to reduce the number of memory accesses.

Figure 10 shows the roofline model. The x-axis indicates the arithmetic intensity. The y-axis indicates the attainable performance. The red and blue plots indicate the sustained performance and arithmetic intensities determined from the actual B/F ratios of the kernels on SX-Aurora TSUBASA and SX-ACE, respectively. The sustained performances of SX-Aurora TSUBASA are higher than those of SX-ACE for all kernels because of its improved computational and memory performance. The arithmetic intensity of SX-Aurora TSUBASA for the *Antenna* kernel is to left of that of SX-ACE. This result is due to the efficient use of the shared LLC. The arithmetic intensity of SX-Aurora TSUBASA for *Plasma* kernel is to right, since divisions in SX-Aurora TSUBASA are calculated by reciprocal approximation, the arithmetic intensity increases.

By comparing the four B/F ratios, the bottlenecks of the kernels can further be analyzed. Table V classifies the bottlenecks in terms of the B/F ratios. Note that other factors such as memory latency are not considered in the analysis, although they may be needed if the kernel to be analyzed is one that dominates indirect memory accesses.

When the code B/F ratio is lower than the LLC B/F ratio and the actual B/F ratio is lower than the memory B/F ratio, the numbers of data requests to the LLC and memory are low. In other words, when computing demand for arithmetic units is high, the kernel is considered computation-bound.

Similarly, when the code B/F ratio is lower than the LLC B/F ratio and the actual B/F ratio is higher than the memory B/F ratio, the number of data requests to memory is high while that to LLC it is low. In that case, the kernel can be considered memory-bound.

Furthermore, when the code B/F ratio is higher than the LLC B/F ratio and the actual B/F ratio is lower than the memory B/F ratio, the number of data requests to LLC is high, while the number of memory requests is low. In this case, the LLC bandwidth becomes a bottleneck.

TABLE V
BOTTLENECK CANDIDATES ANALYZED USING FOUR DIFFERENT B/F RATIOS.

B/F ratio	Actual $<$ Memory	Actual $>$ Memory
Code $<$ LLC	Computation-bound	Memory-bound
Code $>$ LLC	LLC-bound	LLC-bound or memory-bound

Finally, when the code B/F ratio is higher than the LLC B/F ratio and the actual B/F ratio is higher than the memory B/F ratio, the data requests to LLC or memory are high. As a bottleneck cannot be identified in this case, further analysis involving comparing the code B/F ratio with the actual B/F ratio is required.

In the case that

$$Code\,B/F\,ratio > \frac{LLCBW}{MemoryBW} \times Actual\,B/F\,ratio, \quad (1)$$

the number of data requests to LLC is higher than that to memory. Thus, the kernel can be considered LLC-bound.

In the case that

$$Code\,B/F\,ratio < \frac{LLCBW}{MemoryBW} \times Actual\,B/F\,ratio, \quad (2)$$

the number of data requests to memory is higher than that to LLC. Thus, the kernel can be considered memory-bound.

The above analysis enables us to classify the bottlenecks of kernels whose memory access patterns in SX-Aurora TSUBASA are sequential. The *Earthquake* and *Turbulent flow* kernels are bound to the LLC bandwidth. The *Land mine* and *Antenna* kernels are bound to the memory bandwidth.

The sustained performances on SX-Aurora TSUBASA for the *Earthquake* and *Turbulent flow* kernels are 4.54 and 3.18 times higher than those on SX-ACE, respectively. As the ADB bandwidth of SX-ACE is 1 TB/s, the maximum speedup should be 2.66 if these kernels are memory-bound. Although these kernels are bound to the LLC bandwidth in SX-Aurora TSUBASA, they are bound to memory in SX-ACE. This suggests that the speedup ratios of the SX-Aurora TSUBASA could be increased by three times or more. The performance of the *Land mine* kernel is bound to the memory bandwidth in both SX-Aurora TSUBASA and SX-ACE.

SX-Aurora TSUBASA dramatically outperforms the SX-ACE on the *Antenna* kernel. The profile data indicates that the number of memory accesses decreases by about 43% as the number of threads increases. The main reason for this decrease is that sharing the LLC by all cores in the VE reduces the number of memory accesses. Moreover, the *Antenna* kernel is bound to computations in the case of SX-ACE, whereas the memory bandwidth is the bottleneck in SX-Aurora TSUBASA. As a result, SX-Aurora TSUBASA shows a significant performance improvement over SX-ACE.

The speedup ratios for the *Plasma* and *Turbine* kernels that include indirect memory accesses are 2.68 and 2.85, respectively. In the case of the *Turbine* kernel, memory latency is the main performance limitation. In the case of the *Plasma* kernel,

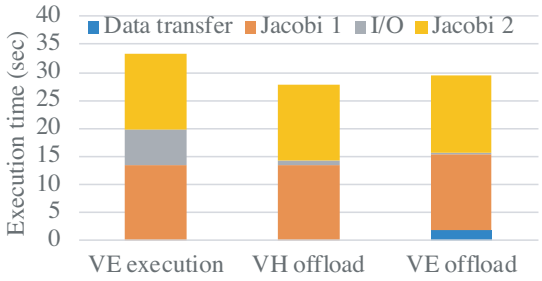

Fig. 11. Performance comparison of *VE offload* and *VH offload*.

the division calculations and memory latency are performance bottlenecks.

These performance evaluations using the fundamental benchmark programs demonstrate that SX-Aurora TSUBASA achieves high performance even with kernels of practical applications. Although the memory B/F ratio of SX-Aurora TSUBASA is lower than that of SX-ACE, the average performance improvement is a factor of 4.6. These results clearly indicate the capability of SX-Aurora TSUBASA at accelerating memory-intensive applications.

IV. Evaluation of the Execution Model of SX-Aurora TSUBASA

This section evaluates the execution model. In SX-Aurora TSUBASA, a program is basically executed on the VE. Only system calls invoked from a program running on the VE are transparently offloaded to the VH. By utilizing this execution model, frequent data transfers between the VH and the VE, which is a bottleneck of the conventional accelerator execution model, can be avoided. In addition, as data transfers and kernels to be executed on the VE do not have to be explicitly specified in a program, existing programs can be ported to SX-Aurora TSUBASA simply by recompiling them for the VE.

In order to evaluate this execution model, a microbenchmark consisting of vector-friendly computations, scalar-friendly computations for I/O, and system calls was developed. In this microbenchmark, the Jacobi kernel in the Himeno benchmark is executed, and the results of the Jacobi kernel are written as formatted output. The computations for the formatted output, such as casts, are not suitable for vector processing, but work fine for scalar processing. The output to a file is a system call. Finally, the Jacobi kernel is executed again.

The microbenchmark is executed in three ways. In the first way, it is executed on a VE, and only a system call is transparently offloaded to the VH; this is called *VE execution*. This is the standard way of executing a program on SX-Aurora TSUBASA. In the second way, called *VH offload*, the benchmark is executed on the VE, and scalar-friendly computations and a system call are explicitly offloaded to the VH. If programmers know that some parts of a computation is suitable for scalar processing, they can explicitly offload it to the VH. In the third way, called *VE offload*, the benchmark is executed on the VH, and only the Jacobi kernel is offloaded to the VE. This execution model is close to the conventional accelerator execution model.

Figure 11 compares the execution times of the three ways. The x-axis indicates the execution ways, and the y-axis indicates the execution time, showing breakdowns of the different parts of the computation. The scalar computations in the VE execution take longer than the other ways. This is because all the computations including the scalar-friendly computations are executed on the VE. In the case of *VH offload*, as such scalar computations are explicitly offloaded to the VH, the formatted output can be executed on the VH. As a result, the execution time of the microbenchmark is the shortest. SX-Aurora TSUBASA achieves the highest performance by executing vector-friendly computations on the VE and offloading the scalar-friendly computations appropriately to the VH. *VE offload* needs extra overhead for transferring data from the VH to the VE. As a result, the total execution time becomes longer than that of *VE offload*.

This comparison clarifies that the execution model of SX-Aurora TSUBASA minimizes data transfers between the VH and VE by executing vector-friendly computations on the VE and appropriately offloading scalar-friendly computations and the system call to the VH. Therefore, the execution model fulfills potential of the SX-Aurora TSUBASA while maintaining high usability.

V. Performance Evaluation of SX-Aurora TSUBASA on Practical Applications

A. Tsunami Simulation Code

This section evaluates the performance of a tsunami numerical simulation code called *TUNAMI (Tohoku University's Numerical Analysis Model for Investigation of near-field tsunamis)* [21] as a practical application. The TUNAMI code solves non-linear shallow water equations, and its numerical scheme uses the staggered leap-frog 2-D finite difference method. It is a standard method for predicting tsunami inundations approved by the United Nations Educational, Scientific and Cultural Organization (UNESCO) [22]. The TUNAMI code is utilized in a real-time tsunami inundation forecast system [23][24]. The code plays a key role, and the simulation needs to be done within five minutes. This system is designed to help local governments in Japan assess where the possibility of damage from a tsunami is high. Therefore, besides having high performance, the system should be small and energy-efficient.

The TUNAMI code is a vectorizable and parallelizable program. It has several double-nested loops in the latitude and longitude directions, and all inner loops are vectorized. The Fortran compilers of SX-Aurora TSUBASA and SX-ACE vectorize the inner loops automatically. In contrast, the Intel compiler does not vectorize any loops, because it incorrectly estimates that their vectorization would not improve performance. Important loops that are closely related to performance are not vectorized. Thus, the compiler directive *vector always* is inserted in these loops so that they will be vectorized by the Intel compiler. The TUNAMI code is parallelized

Fig. 12. Core and socket performance of the TUNAMI code.

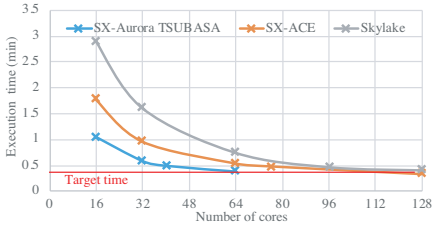

Fig. 13. Parallel performance of the TUNAMI code.

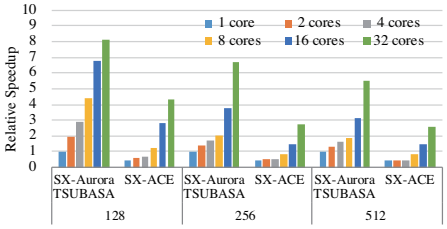

Fig. 14. Relative speedup of the DNS code to the computation time on a core of SX-Aurora TSUBASA for each problem size

using a domain decomposition method with MPI. A 1-D (y-direction) domain decomposition is used instead of a 2-D (x- and y-directions) one, because the loops of the x-direction are vectorized, where long loop lengths need to be kept in order to determine the performance of vector architectures.

The performances of a single core and a single socket of SX-Aurora TSUBASA, SX-ACE, Skylake, and KNL 7210 were compared. The simulation area was of a coastal region of Japan (1,244 $\times$ 826 km) with an 810 m resolution. The code should be executed within 30 seconds, because the problem size is one tenth the size of the actual simulation. Figure 12 shows the speedup ratios of the core and socket performance on KNL 7210, Skylake, and SX-Aurora TSUBASA relative to that on SX-ACE along with the efficiencies of each system. The purple and gray bars indicating core performance illustrate the effectiveness of vector processing for this code. The core performances of KNL 7210 and Skylake with vectorization are respectively 5.3 and 1.8 times higher than those without vectorization. This means that vector processing is an important way of obtaining higher performance for the TUNAMI code.

Regarding the core performances depicted in Figure 12, the computational efficiency of the SX-Aurora TSUBASA is higher than those of KNL 7210 and Skylake and about a quarter that of SX-ACE. Thus, it outperforms KNL 7210, Skylake, and SX-ACE by 11.4, 4.1 and 2.4 times in terms of computational efficiency, respectively. Its socket performance is 2.5, 1.9 and 3.5 times higher than those of KNL, Skylake and SX-ACE. The computational efficiency of SX-Aurora TSUBASA is slightly lower than that of Skylake. This is because the performance of the TUNAMI code is bound to the LLC bandwidth in SX-Aurora TSUBASA.

Parallel performance was evaluated in a simulation of phenomena occurring three hours after an earthquake. Figure 13 shows the execution times on 11 sockets of Skylake, 32 sockets of SX-ACE, and eight sockets of SX-Aurora TSUBASA. The execution time of 16 cores on SX-Aurora TSUBASA is 63 seconds. As SX-Aurora TSUBASA achieves the highest core performance in the four systems, it has the short execution time on the smallest number of cores. Meanwhile, the overhead of the MPI communication gradually decreases the scalability of SX-Aurora TSUBASA for 32 or more cores. As shown in the figure, it needs fewer cores compared with SX-ACE and Skylake to complete the simulation within the target time.

These results demonstrate that SX-Aurora TSUBASA can exploit the potential of its powerful vector operation capability in comparison with other vector functions such as SIMD and AVX of KNL 7210 and Skylake. As the SX-Aurora TSUBASA occupies one tenth the floor space and has one fifth the power consumption of the SX-ACE, it is a compact platform for executing the TUNAMI simulation code.

B. Direct Numerical Simulation Code of Turbulent Flows

Solving turbulent flows is essential in science and engineering research fields because practical applications always include turbulent phenomena. DNSs of turbulence with high Reynolds number Re, however, are difficult to carry out because the number of grid points approximately proportional to $Re^{9/4}$ is necessary. The larger Re is, the more the number of grid points in discretization increases. Therefore, computing systems with higher calculation speeds are required. The recent developments of supercomputing systems make them a powerful means to elucidate such phenomena.

In 2002, high resolution DNSs of incompressible homogeneous turbulent flows with up to 4096^3 grid points were performed on the Earth Simulator by implementing an efficient DNS code based on the Fourier spectral method [25]. A lot of illuminating results were obtained from those simulations. More recently, DNSs with up to 12288^3 grid points were carried out on the K computer [26]. Higher performance supercomputers, however, still need to be developed.

Evaluating the performance of DNS codes on SX-Aurora TSUBASA is quite important to determine whether higher resolution DNSs with more than 12288^3 grid points are possible or not. This section compares the results of running DNS code on SX-ACE and SX-Aurora TSUBASA.

The incompressible Navier-Stokes equations and a continuum equation were discretized using a Fourier spectral method. The nonlinear terms are convolution sums, and therefore, three-dimensional fast Fourier transforms (3D-FFT) can be applied to the convolution to reduce the number of arithmetic operations. However, aliasing errors arise in the FFT calculation. These errors can be removed by applying the phase shift method and by taking the maximum wavenumber $k_{\max}$ as $(\sqrt{2}/3)N$ in the implementations, where N is the number of grid points in each of the Cartesian coordinates in physical space.

In computations by MPI parallelization of the 3D-FFT, data transfers using all-to-all communications among all MPI processes easily becomes a bottleneck. Thus, the number

of all-to-all communications should be avoided as many as possible. Some parallel DNS codes in Fortran for high-resolution turbulence simulations was developed for the Earth Simulator in 2002 and on the K computer in 2014. The code on the Earth Simulator is parallelized in one direction by slab decomposition. Then, two-directional parallelization, or pencil decomposition, is used so that the memory capacity in an MPI process can be reduced as much as possible.

Computation times for one time step of the RK integration for the slab decomposition DNS code with grid points of 128^3, 256^3, and 512^3 were measured by changing the number of MPI processes from 1 to 32 on SX-Aurora TSUBASA, and from 1 to 32 on SX-ACE. Each MPI process was allotted to one core in all the measurements.

The relative speedups of calculation time are shown in Figure 14. The calculation time of one core on SX-Aurora TSUBASA is taken as a reference value in order to compare speedup ratios for three different grid points. For the grid points with 128^3, 256^3, and 512^3, the speedups of 32-core calculations on SX-Aurora TSUBASA are approximately 8.14, 6.73, and 5.48 times compared with the one-core calculation on it, respectively. The speedups of any number of cores on SX-ACE are relatively low and does not change so much as the number of cores of SX-ACE increases.

The reason is that the LLC hit rates in the cases of 256^3 and 512^3 grid points become lower than that of 128^3 grid points in SX-Aurora TSUBASA. While the LLC hit rate is about 30% in the case of 128^3 grid points, the LLC hit rates decrease by about 1 to 6% in the cases of 256^3 and 512^3 grid points. Complex numbers of the Fourier coefficients are represented with two different DP variables, which represent the real parts and imaginary parts of the complex numbers, in the code. Since the performance of this implementation greatly affects memory performance, performance tuning such as modification of variable structure and cache blocking can further improve the hit rates.

Approximately 10% of the execution time is occupied by bank conflicts in all cases. Stride and indirect accesses in the innermost loop seem to affect the memory performance. Further optimization and careful tuning to memory accesses lead to a high sustained performance, and thereby can exploit the potential of SX-Aurora TSUBASA.

VI. Conclusions

A new vector supercomputer, SX-Aurora TSUBASA, with the world's highest memory bandwidth was released with the aim of achieving high sustained performance while maintaining the high usability of vector supercomputing systems. SX-Aurora TSUBASA has a new system architecture that consists of several VEs and a VH. The VEs are responsible for executing an application, and the VH is used for processing system calls invoked by the application. As system calls are transparently offloaded to the VH, programmers do not need to worry about system calls. Therefore, programmers can write application programs in standard programming languages without any special languages, which leads to high usability. Furthermore, SX-Aurora TSUBASA has the world's highest memory bandwidth thanks to using six HBM2 memory modules that are tightly connected to the VE on a silicon interposer made with a 3D die stacking technology. Therefore, high sustained performance can be expected, especially for memory-intensive applications.

This paper clarifies the potential of SX-Aurora TSUBASA by describing performance evaluations of various programs including standard benchmark programs, a microbenchmark, kernels of practical applications, and entire practical applications. The results show that the high sustained memory bandwidth of SX-Aurora TSUBASA leads to t high sustained performance in various applications. Furthermore, the new execution model especially designed for SX-Aurora TSUBASA overcomes the drawback of the conventional execution model of accelerators. This paper also analyzes causes of bottlenecks by using four different B/F ratios: code B/F, actual B/F, LLC B/F, and memory B/F.

The evaluations show the importance of high sustained memory performance. The SX-Aurora TSUBASA opens up the field of supercomputing systems to more than just increasing theoretical computing performance. The balance between memory performance and computing performance is crucial to achieving high sustained performance. A new methodology to utilize the execution model and new optimization techniques for the SX-Aurora TSUBASA will be addressed in future work.

Acknowledgments

The authors would like to thank Hiroyuki Takizawa, Ryusuke Egawa, Yasuhisa Masaoka, and Souya Fujimoto for useful discussions. The authors also thank Association for Real-time Tsunami Science (ARTS) for use of the tsunami simulation code. The authors also thank the SX-Aurora TSUBASA beta program for early access to SX-Aurora TSUBASA. This research was partially supported by Grants-in-Aid for Scientific Research(S) #17H06108, Research(B) #16H02822, Research(C) #18K11322, Research(C) #18K11325, and Joint Usage/Research Center for Interdisciplinary Large-scale Information Infrastructures in Japan (Project ID: jh180040-NAH).

References

[1] TOP500 Supercomputer Sites. [Online]. Available: http://www.top500.org/

[2] W. A. Wulf and S. A. McKee, "Hitting the memory wall: Implications of the obvious," *SIGARCH Comput. Archit. News*, vol. 23, no. 1, pp. 20–24, Mar. 1995. [Online]. Available: http://doi.acm.org/10.1145/216585.216588

[3] T. Soga, A. Musa, Y. Shimomura, R. Egawa, K. Itakura, H. Takizawa, K. Okabe, and H. Kobayashi, "Performance evaluation of NEC SX-9 using real science and engineering applications," in *Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis*, ser. SC '09, 2009, pp. 28:1–28:12. [Online]. Available: http://doi.acm.org/10.1145/1654059.1654088

[4] R. Egawa, K. Komatsu, S. Momose, Y. Isobe, A. Musa, H. Takizawa, and H. Kobayashi, "Potential of a modern vector supercomputer for practical applications: performance evaluation of SX-ACE," *The Journal of Supercomputing*, vol. 73, no. 9, pp. 3948–3976, Sep 2017. [Online]. Available: https://doi.org/10.1007/s11227-017-1993-y

[5] K. Komatsu, R. Egawa, Y. Isobe, R. Ogata, H. Takizawa, and H. Kobayashi, "An approach to the highest efficiency of the HPCG benchmark on the SX-ACE supercomputer," in *Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis (SC15), Poster*, Nov 2015, pp. 1–2.

[6] J. Dongarra, M. A. Heroux, and P. Luszczek, "High-performance conjugate-gradient benchmark," *Int. J. High Perform. Comput. Appl.*, vol. 30, no. 1, pp. 3–10, Feb. 2016. [Online]. Available: http://dx.doi.org/10.1177/1094342015593158

[7] HPCG Benchmark. [Online]. Available: http://www.hpcg-benchmark.org/

[8] NEC SX-Aurora TSUBASA - Vector Engine. [Online]. Available: https://www.nec.com/en/global/solutions/hpc/sx/vector_engine.html

[9] NEC releases new high-end HPC product line, SX-Aurora TSUBASA Engine. [Online]. Available: https://www.nec.com/en/press/201710/global_20171025_01.html

[10] S. Y. Hou, W. C. Chen, C. Hu, C. Chiu, K. C. Ting, T. S. Lin, W. H. Wei, W. C. Chiou, V. J. C. Lin, V. C. Y. Chang, C. T. Wang, C. H. Wu, and D. Yu, "Wafer-level integration of an advanced logic-memory system through the second-generation CoWoS technology," *IEEE Transactions on Electron Devices*, vol. 64, no. 10, pp. 4071–4077, Oct 2017.

[11] M. Shimizu and A. Yonezawa, "Remote process execution and remote file I/O for heterogeneous processors in cluster systems," in *2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, May 2010, pp. 145–154.

[12] B. Gerofi, M. Takagi, A. Hori, G. Nakamura, T. Shirasawa, and Y. Ishikawa, "On the scalability, performance isolation and device driver transparency of the IHK/McKernel hybrid lightweight kernel," in *2016 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, May 2016, pp. 1041–1050.

[13] M. Sato, T. Kobayashi, Z. Zeng, G. Fang, and X. Feng, "High resolution GPR system for landmine detection," in *Proceedings of International Conference Requirements and Technologies for the Detection, Removal and Neutralization of Landmine and UXO*, 01 2003, pp. 548–553.

[14] K. Ariyoshi, T. Matsuzawa, and A. Hasegawa, "The key frictional parameters controlling spatial variations in the speed of postseismic-slip propagation on a subduction plate boundary," *Earth and Planetary Science Letters*, vol. 256, no. 1, pp. 136 – 146, 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0012821X07000374

[15] T. Tsukahara, K. Iwamoto, and H. Kawamura, "Evolution of material line in turbulent channel flow," in *Proceedings of the Fifth International Symposium on Turbulence and Shear Flow Phenomena*, 01 2007, pp. 549–554.

[16] H. Sato, Y. Takagi, and K. Sawaya, "High gain antipodal Fermi antenna with low cross polarization," *IEICE transactions on communications*, vol. 94, no. 8, pp. 2292–2297, Aug 2011. [Online]. Available: https://ci.nii.ac.jp/naid/10030187090/

[17] Y. Katoh, T. Ono, and M. Iizima, "Numerical simulation of resonant scattering of energetic electrons in the outer radiation belt," *Earth, Planets and Space*, vol. 57, no. 2, pp. 117–124, 2005.

[18] Y. Sasao and S. Yamamoto, "Numerical prediction of unsteady flows through turbine stator-rotor channels with condensation," *ASME Fluids Engineering Division Summer Meeting*, vol. 1, Symposia, Parts A and B, pp. 855–861, 2005.

[19] J. D. McCalpin, "Memory bandwidth and machine balance in current high performance computers," *IEEE Computer Society Technical Committee on Computer Architecture (TCCA) Newsletter*, pp. 19–25, Dec. 1995.

[20] Himeno benchmark. [Online]. Available: http://accc.riken.jp/en/supercom/himenobmt/

[21] S. Koshimura, H. Yanagisawa, and F. Imamura, "Developing fragility functions for tsunami damage estimation using numerical model and post-tsunami data from Banda Aceh, Indonesia," vol. 51, pp. 984–51, 09 2009.

[22] C. Goto, Y. Ogawa, N. Shuto, and F. Imamura, "Numerical method of tsunami simulation with the leap-frog scheme," *Intergovernmental Oceanographic Commission of UNESCO*, 1997. [Online]. Available: https://ci.nii.ac.jp/naid/20001358454/

[23] A. Musa, H. Matsuoka, O. Watanabe, Y. Murashima, S. Koshimura, R. Hino, Y. Ohta, and H. Kobayashi, "A real-time tsunami inundation forecast system for tsunami disaster prevention and mitigation," in *Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis (SC15), Poster*, Nov 2015, pp. 1–2.

[24] A. Musa, T. Abe, T. Inoue, H. Hokari, Y. Murashima, Y. Kido, S. Date, S. Shimojo, S. Koshimura, and H. Kobayashi, "A real-time tsunami inundation forecast system using vector supercomputer SX-ACE," *Journal of Disaster Research*, vol. 13, no. 2, pp. 234–244, March 2018.

[25] M. Yokokawa, K. Itakura, A. Uno, T. Ishihara, and Y. Kaneda, "16.4-tflops direct numerical simulation of turbulence by a Fourier spectral method on the earth simulator," in *Supercomputing, ACM/IEEE 2002 Conference*, Nov 2002, pp. 50–50.

[26] T. Ishihara, K. Morishita, M. Yokokawa, A. Uno, and Y. Kaneda, "Energy spectrum in high-resolution direct numerical simulations of turbulence," *Phys. Rev. Fluids*, vol. 1, p. 082403, Dec 2016. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevFluids.1.082403

APPENDIX

A. Additional application evaluations

An additional study on the new execution model of SX-Aurora TSUBASA was carried out by evaluating I/O performance of system calls, which are invoked on the VE and then executed on the VH. The DNS code in Subsection V-B is used.

Snapshots of the velocity fields in the spectral space are stored at an appropriate interval in time integration of the DNS as checkpoints so that the simulation with the same parameters can be restarted to continue at the time when the snapshot is taken. In particular, since DNSs need long time to reach a quasi-stationary state, and a CPU resource allocated for each batch job is generally limited in a normal operation at any computer center, the checkpoint restart function plays an important role for an actual computing environments.

The amount of output data in a binary format for a snapshot of the velocity field is given by a formula

$$\begin{aligned} &16\,bytes\,(double\,precision\,complex\,variable) \\ &\times 3\,arrays\,(velocities) \times N \times N \times \frac{N}{2}, \end{aligned} \tag{3}$$

where N is the number of grid points in each direction of the three-dimensional Fourier spectral space. The velocity in a physical space is real and the conjugates of Fourier modes can be omitted in the spectral space. Therefore, the number of complex variables in the spectral space becomes a half.

In this study, seven cases of grid points N =128, 160, 192, 256, 320, 384, and 512 are considered. These numbers can be factorized by prime numbers of 2, 3, and 5 so that standard FFT can be applied in the DNS code. The amounts of snapshots of a file for each case are 48 MiB, 94 MiB, 162 MiB, 384 MiB, 750 MiB, 1296 MiB, and 3072 MiB, respectively. The snapshot is distributed among processes, each of which performs checkpointing. For example, in the case of eight MPI processes, the amount of a snapshot per process becomes one eighth of the whole snapshot. The interval of checkpoints should be determined properly by considering CPU and I/O performances. During the interval when I/O system calls are off-loaded to the VH, the calculations of the next several time steps that are approximately equivalent to the checkpoint time can be performed.

The performance of a check point function in the DNS code was measured for each case on A300-2. As the checkpoint includes WRITE system calls, it is transparently off-loaded to the VHs from the VEs.

Figure 15 shows the WRITE performances for the seven cases by changing the number of MPI processes. It is found that the WRITE performance increases as the number of MPI processes increases. The performances of 16 MPI processes for any cases are 2x or more than those of eight processes. In the case of 16 processes, two VEs are utilized, and then two VE processes and their corresponding pseudo processes work simultaneously. As a result, almost twice improvement of I/O performance can be achieved when the number of VEs is two.

This figure also shows that 1.7 GB/s and 3.4 GB/s I/O performances are obtained in the case of 8 processes and 16 processes at maximum, respectively. Although the data transfers between the VH and the VEs and additional buffer copy are necessary, the I/O performances can achieve about 80% of the I/O performance of the VH. These results indicate that SX-Aurora TSUBASA achieves sufficient I/O performance even on the new execution model.

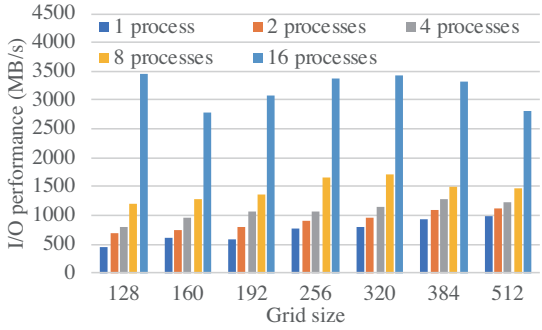

Fig. 15. I/O Performance of checkpointing the DNS code.