DOI: 10.14529/jsfi210205

# Performance and Power Analysis of a Vector Computing System*

***Kazuhiko Komatsu***[1], ***Akito Onodera***[2], ***Erich Focht***[3], ***Soya Fujimoto***[4], ***Yoko Isobe***[4], ***Shintaro Momose***[4], ***Masayuki Sato***[2], ***Hiroaki Kobayashi***[2]

The performance of recent computing systems has drastically improved due to the increase in the number of cores. However, this approach is reaching the limitation due to the power constraints of facilities. Instead, this paper focuses on a vector processing with long vector length that has a potential to realize high performance and high power efficiency. This paper discusses the potential through the optimization of two benchmarks, the Himeno and HPCG benchmarks, for the latest vector computing system SX-Aurora TSUBASA. The architecture of SX-Aurora TSUBASA owes the high efficiency to making good of its long vector length. Considering these characteristics, various levels of optimizations required for a large-scale vector computing system are examined such as vectorization, loop unrolling, use of cache, domain decomposition, process mapping, and problem size tuning. The evaluation and analysis suggest that the optimizations improve the sustained performance, power efficiency, and scalability of both benchmarks. Therefore, it is clarified that the SX-Aurora TSUBASA architecture can achieve higher power efficiency due to its high sustained memory bandwidth paired with the long vector computing.

*Keywords: SX-Aurora TSUBASA, optimization, vector computing, power efficiency, Himeno benchmark, HPCG.*

## Introduction

The performance of recent high-performance computing (HPC) systems has been remarkably improved. One of the main factors is the increase in the number of nodes. A large number of nodes are clustered into an HPC system. For example, Supercomputer Fugaku, the top 1 system in the TOP500 ranking as of November 2020, is equipped with 158,976 computing nodes and 7,630,848 cores [5]. The large number of nodes brings the improvement of the peak performance. The other factor is the improvement of a core in a processor. The improvement of a core performance is mainly due to the improvement of vector processing. Vector processing has been adopted by various recent processors in shape of SIMD units, AVX-512 instruction architecture (ISA) for Intel Xeon, AVX-2 ISA for AMD EPYC. GPUs from NVIDIA and AMD support vectorization in the SIMT manner, while Fujitsu A64FX implements the ARM SVE as SIMD units with a vector ISA. The NEC SX dedicated vector processors implement a long vector ISA combining SIMD with pipelining.

The performance growth comes with a considerable increase in the power consumption of HPC systems. Due to the limitation of the power supply capacity of each system, the conventional approach to improve the performance by simply increasing the number of nodes is reaching the limit. For the design of future HPC systems, a paradigm shift to another new approach is essential to maximize performance within limited power constraints.

[*] This paper is an extended version of the following two papers, A. Onodera, et al., “Optimization of the Himeno Benchmark for SX-Aurora TSUBASA,” Proceedings of International Symposium on Benchmarking, Measuring and Optimizing (Bench20), 2020, and K. Komatsu, et al., “Performance Evaluation of a Vector Supercomputer SX-Aurora TSUBASA,” Proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis (SC18), 2018, by adding optimizations of HPCG and evaluations of the Himeno and HPCG benchmarks on large vector computing systems.

[1] Cyberscience Center, Tohoku University, Miyagi, Japan
[2] Graduate School of Information Sciences, Tohoku University, Miyagi, Japan
[3] NEC Deutschland GmbH, Germany
[4] NEC Corporation, Japan

This paper focuses on a computing system that uses a long vector ISA, which is one of the most promising technologies for high power efficiency. To exploit the potential of the computing system, this paper takes an approach to the enhancement of the sustained performance by the optimizations for not only a single node but also multiple nodes on a vector computing system. So far, there have been many efforts for the single node optimizations for a long vector ISA to accelerate HPC applications [10, 13, 19, 20]. Especially, focusing on a high data supply capability to cores in a vector processor, many memory-intensive HPC applications such as computational fluid dynamics simulation [22] have been accelerated. This paper examines NEC's latest vector system named *SX-Aurora TSUBASA* that adopts commodity interconnects such as InfiniBand for inter-node communication [24, 30]. Since multiple node optimizations as well as the single-node optimizations are important, this paper optimizes, two benchmark programs, the Himeno [1] and HPCG [2, 9, 17] benchmarks by applying not only the single optimizations such as vector optimizations, loop unrolling, and efficient use of the cache, but also the multiple node optimizations such as appropriate process mapping, tuning of domain decomposition. By performance evaluation and analysis in terms of sustained performance, scalability, and power consumption, the power efficiency of a large-scale SX-Aurora TSUBASA is investigated.

The contributions of this paper are the following.

1. The potential of a large-scale vector computing system SX-Aurora TSUBASA is investigated. By applying optimizations to the Himeno and HPCG benchmarks, the effectiveness of the optimizations is discussed using evaluations on two generations of SX-Aurora TSUBASA.
2. The power efficiency of the vector computing system SX-Aurora TSUBASA is quantitatively discussed by detailed power analysis.

The rest of this paper is organized as follows. Section 1 explains a large-scale vector computing system of SX-Aurora TSUBASA. Section 2 describes optimizations of the Himeno and HPCG benchmarks for SX-Aurora TSUBASA. Section 3 discusses the effectiveness of the optimization and investigates the power efficiency of SX-Aurora TSUBASA through evaluation. Section 4 introduces related work. Section 4 describes the conclusions of this paper.

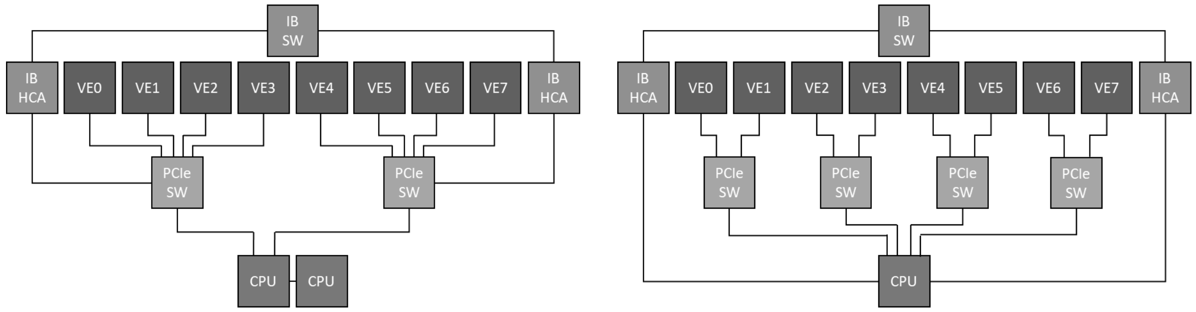
# 1. Overview of Vector Computing Systems

## 1.1. SX-Aurora TSUBASA Vector Computing System

NEC SX is a series of vector supercomputer systems that has been continuously developed since 1983. SX-Aurora TSUBASA is the latest vector computing system based on not only the long term experience and accumulated knowledge but also new strategies that improve the flexibility and usability.

Figure 1 shows two generations of SX-Aurora TSUBASA. Figure 1a sketches the first generation of SX-Aurora TSUBASA called an A300-8 model. One *Vector Host (VH) node* consists of one VH and eight Vector Engines (*VEs*). Eight of the first generation VEs are connected to one VH through two 16-lane PCI Express (PCIe) generation 3.0 switches. The eight VEs are divided into two *VE groups*. Four VEs in each VE group and one InfiniBand EDR HCA are connected to each PCIe switch.

Figure 1b depicts the second generation of SX-Aurora TSUBASA called a B401-8 model. Eight of the second generation of VEs are used in the B401-8 model. The eight VEs are divided



(a) An A300-8 model

(b) A B401-8 model

**Figure 1.** Two generations of SX-Aurora TSUBASA

into four VE groups. Two VEs in a VE group are connected to a PCIe generation 3.0 switch. The two InfiniBand HDR HCAs are connected to a CPU through 16-lane PCIe generation 4.0.

There are four paths to communicate among processes on SX-Aurora TSUBASA: communication within a VE node, communication within a VE group, communication among VE groups, and communication among VHs. Since each path has different bandwidth and latency, it is necessary to optimize parallel computing considering the difference of communication paths.

A VH is a common x86 Linux node equipped with an x86 processor such as Intel Xeon and AMD EPYC. A VE is a dedicated vector processor attached to a VH through PCI Express. An application is basically executed on a VE as a primary processor responsible for main calculations. A VH mainly manages VEs and performs OS-related tasks such as system calls from a VE. The OS-related tasks are transparently offloaded to a VH from a VE. This transparent offload enables a programmer to use the vector processor without any special effort such as the specifications of computational kernels and OS-related tasks. Furthermore, this execution model of SX-Aurora TSUBASA can reduce frequent data transfers between a VE and a VH. Such data transfers become one of the major bottleneck factors on an ordinary accelerator.

Moreover, SX-Aurora TSUBASA supports two explicit offload mechanisms: *VH call* and *VEO (VE offload)*. VH call can offload scalar-friendly computations such as serial computation and system calls to a VH from a VE by explicitly specifying a part of an application to be offloaded. On the other hand, VEO is used for programs executed on a VH as a primary processor. By VEO, a part of an application is offloaded to a VE, and the VE acts as a secondary processor, which is close to the execution model of an ordinary accelerator. Using VEO, vector-friendly computations such as main computations are explicitly offloaded to a VE from a VH. These offloading mechanisms allow the SX-Aurora TSUBASA to support various execution models. This flexibility contributes to improvements of usability and effective usage of the computational resources by considering characteristics of applications and processors.

## 1.2. Vector Engine

A VE is a vector processor that mainly contributes to the system performance based on its vector computing capability. Figure 2 shows an architecture of a VE. The architecture of two generations of VE is the same. The VE is equipped with eight vector cores. As the core performance of VE Type 10B called VE 10B is 537.6 Gflop/s for single-precision (SP) floating-point calculations, the socket performance reaches 4.30 Tflop/s. In the case of the second generation of VE Type 20B called VE 20B, the core performance is 614.4 Gflop/s (SP), resulting in the socket performance of 4.92 Tflop/s. The vector length of each vector core is 256 double-precision

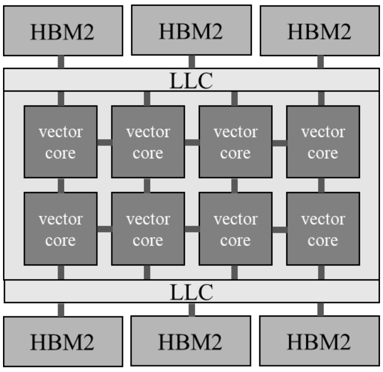


**Figure 2.** Architecture of a VE

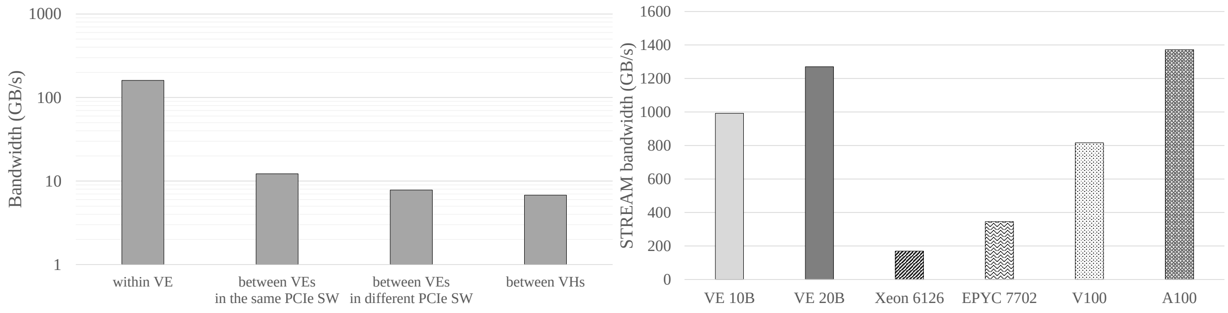
floating-point elements. It is much longer than that of recent x86 processors which have the SIMD length of 8 double-precision words in 512-bit SIMD units. The eight vector cores share a total 16 MB last level cache (LLC). Each core and the LLC are connected by a two-dimensional mesh network. Furthermore, six High Bandwidth Memory (HBM) modules work together as the main memory [18]. VE 10B and VE 20B use HBM2 [7] and HBM2E [26], respectively. As a result, their memory bandwidths reach 1.22 TB/s and 1.53 TB/s, which are much higher than those achieved with conventional DDR memory modules.

An optional configuration mode of the VE is the *partitioning mode*. In partitioning mode, vector cores, LLC, and main memory are virtually partitioned into two same capability segments. A VE can be treated as two independent partitioning nodes. Since these nodes are isolated from each other, conflicts of the communication between LLC and vector cores can be reduced. Thus, the partitioning mode is useful for an application whose bottleneck is the LLC bandwidth. On the other hand, one vector core can use only the half of memory bandwidth and capacity of the VE processor. Because of this trade-off, the partitioning mode should be used considering characteristics of target applications.

### 1.3. Multiple Levels of Bandwidths of SX-Aurora TSUBASA

In order to examine the various memory levels bandwidths of SX-Aurora TSUBASA, preliminary evaluations are conducted. Figure 3a shows the peer-to-peer network bandwidths of the four communication paths using the *osu_bw* kernel of the OSU Micro-Benchmarks [3]. The vertical axis shows the bandwidth when the message size is 512 KB. The horizontal axis shows the communication paths on the B401-8 systems. This figure shows that the network bandwidth is fast in the order of communication within a VE, within a VE group, with different VE groups, and between a VH node. Therefore, to efficiently exploit the potential of a system, it is necessary to take care of the differences in the communication bandwidths. Since the bandwidths are different among various paths, the communication that requires high bandwidth should use high bandwidth paths, e.g., by localizing communication as much as possible.

Figure 3b shows the memory bandwidth using the *triad* kernel of the STREAM benchmark [4]. The vertical axis shows the stream memory bandwidth. The horizontal axis shows the tested processor types. VE 10B, VE 20B, two sockets of Intel Xeon Gold 6126, called Xeon 6126, and two sockets of AMD EPYC 7702 are used as x86 processors. Nvidia TESLA V100 and A100 are used as GPUs. This figure shows that VE 20B and A100 achieve the highest memory bandwidths. Although the memory bandwidths of VE 10B and V100 are lower than those of VE 20B and A100, they are higher than those of two sockets of Xeon 6126 and two sockets of EPYC



(a) Peer-to-peer communication bandwidth

(b) STREAM memory benchmark

**Figure 3.** Bandwidth of SX-Aurora TSUBASA

7702. The main reason for the differences in the bandwidth comes from the memory subsystems that each processor adopts. VE 20B and A100 are HBM2E memory modules, VE 10B and V100 are the HBM2 memory modules, and Xeon 6126 and EPYC 770 are the DDR4 memory DIMMs. These differences affect the stream memory bandwidths. To take a close look at the figure, the stream memory bandwidth of A100 is about 8.0 % higher than that of VE 20B. The efficiency to the peak memory bandwidth of A100 is higher than that of VE 20B. The efficiencies of A100 and VE 20B are 88.2 % and 82.7 %, respectively. The operational frequency of HBM2E and the efficiencies lead to the differences in the memory bandwidths between VE20B and A100. As a result of the preliminary evaluation, it is essential for the optimization of applications to fully exploit the various bandwidths considering the characteristics of a single node and multiple nodes.

## 2. Optimization Techniques for a Vector Computing System

To exploit the potential of a vector computing system, optimizations for a single node and multiple nodes are essential. For target computations, this paper chooses important kernels frequently used in memory-bound HPC applications: a stencil computation and a conjugate gradient (CG) computation. There are two famous benchmark programs including these types of the kernels, the Himeno and HPCG benchmarks. In this section, by briefly investigating the characteristics of the benchmarks, optimizations such as vectorization, exploitation of memory and LLC bandwidths, domain decomposition, and process mapping are applied to these benchmark programs.

### 2.1. Optimizations for Stencil Computations

Stencil computations are one of the important kernels in the field of HPC and data sciences. The Himeno benchmark is one of the benchmark programs that measures the performance of the stencil computations. The Himeno benchmark solves the Poisson equation by the Jacobi method in the incompressible fluid analysis [1]. The main kernel named Jacobi requires high memory bandwidth because it performs stencil calculations that continuously update grid points using the values of adjacent grid points. In the Jacobi kernel, 19-point stencil calculations are performed for an array $p$ of the pressure term. By updating the array p in a triple loop in the $i$, $j$, and $k$ directions, 19 references of array p occur in one iteration.

First, to understand the characteristics of the Himeno benchmark, its code is briefly analyzed using four Bytes/Flop (B/F) ratios: *required B/F*, *actual B/F*, *memory B/F*, and *LLC B/F* [11, 24]. The required B/F ratio is defined as the ratio of bytes of the number of load and store instructions to the number of floating-point operation instructions. The required B/F ratio of the Himeno benchmark is 3.33. The actual B/F ratio is calculated from the number of actual memory accesses that take into account the actual behavior of LLC divided by the number of actual floating-point operation instructions. The actual B/F ratio of the Himeno benchmark is 2.24 in VE 10B and VE 20B. The memory and LLC B/F ratios are defined as the ratios of the peak memory and LLC bandwidths to the peak computing performance. The memory B/F ratios of VE 10B and 20B are 0.28 and 0.31, respectively. The LLC B/F ratios of VE 10B and 20B are 0.62 and 0.61, respectively. By comparing with the four B/F ratios, the Himeno benchmark is judged as a memory bandwidth-bound application even on vector computing systems equipped with high memory bandwidth.

To exploit the bandwidth of SX-Aurora TSUBASA, the optimizations for improving utilization of the LLC, loop unrolling, domain decomposition, and process mapping are applied [27]. The first optimization is the efficient use of the LLC. Since each element in an array $p$ is used 19 times in the Jacobi kernel, 18 times of memory accesses can be reduced if the element is stored in the LLC. For the 19-point stencil calculation, three planes need to be stored in LLC to reuse an element 18 times if the size of three planes can fit the LLC. Therefore, the priority of the cache retention for array $p$ sets to be high by using a dedicated compiler directive.

The next optimization is loop unrolling to reduce the loop overhead. As the Jacobi kernel is the triple nested loop and the number of loop iterations is large, the cost of controlling the loop such as loop condition tests and increments of loop indices cannot be ignored, especially on vector computing systems. By applying loop unrolling, the overhead is reduced. As the innermost loop is used for the vectorization and the outermost loop is used for parallelization, the second loop is unrolled. As VE has more vector registers than a general-purpose processor, the number of unrolls can be large, which is more effective in general. This paper selects the best parameter of the number of unrolls by the brute-force search in the range of $2^0$ to $2^6$.

The third optimization is the tuning of the domain decomposition. For the MPI version of the Himeno benchmark, it is necessary to decompose the three-dimensional domain for parallel processing. To keep a sufficiently large vector length for the computation, the innermost loop should be carefully selected. Thus, the length in the $k$ direction should be at least 256. Moreover, the decomposed domain in the $j$ direction should be smaller than that in the $i$ direction as the memory accesses to the $i$ direction are sequential. The appropriate domain decomposition is searched by brute-force as the patterns of the domain decomposition are not so large.

The last optimization is process mapping. A halo communication between two processes that calculate adjacent domains is one of the most bandwidth-bound communication parts in the Himeno benchmark. As there is a bandwidth difference of each communication path shown in Fig. 3a, adjacent processes are carefully assigned in the same VE and the same VE group rather than the VH-VH communication and the VH groups communication. Furthermore, considering the balance of the communication load in each communication path, the process assignment is equally distributed.

## 2.2. Optimizations for the Conjugate Gradient Method

The CG method is one of algorithms to solve linear equations. The CG method is generally used for large sparse systems that are difficult to handle by direct numerical methods. The HPCG benchmark [2, 9, 17] is one of the benchmark programs that measures the performance of the CG computation. The HPCG solves a linear equation $Ax = b$ with a symmetric sparse matrix discretized by the finite element method using a multi-grid preconditioned conjugate gradient (CG) method with a symmetric Gauss-Seidel smoother. According to the CG method, the linear equation $A\boldsymbol{x} = \boldsymbol{b}$ results in finding $x$ that minimizes $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T A\boldsymbol{x} - \boldsymbol{b}^T\boldsymbol{x} + c$. The CG method solves simultaneous linear equations by the iterative method. The method is often used in large-scale sparse matrix coefficients that would require a huge number of calculations and memory in direct methods like Gauss elimination. The required B/F ratio of the reference version of the HPCG benchmark is 8.31, making it a very realistic benchmark for memory-bound applications and an ideal candidate for exploiting the large memory bandwidth of the VE.

By using the B/F ratios, the characteristics of the HPCG benchmark is briefly analyzed. The required and actual B/F ratios of the HPCG benchmark are 7.62 and 5.80, respectively. The memory B/F ratios of VE 10B and 20B are 0.28 and 0.31, respectively. The LLC B/F ratios of VE 10B and 20B are 0.62 and 0.61, respectively. By comparing with the four B/F ratios, it is clarified that the HPCG benchmark is an LLC bandwidth-bound program.

This paper uses the code optimized for SX-Aurora TSUBASA [16]. The code is based on a vectorized version of the reference algorithm that has achieved 11.2 % efficiency for the SX-ACE processors [12, 14, 21] by the following important optimizations: ELLPACK data format for the sparse matrix, hyperplanes or level scheduling ordering for vectorization, and cache retention control for variables in the SX-ACE advanced data buffer (ADB) for data reuse. Instead of the cache retention control for ADB on SX-ACE, the priority of the cache retention for the LLC on VE is controlled by the dedicated compiler directive.

Besides vectorization and optimal data access, the highest impact on the performance is reformulating the Gauss-Seidel smoother implementation to significantly reduce the number of operations. When storing the matrix $A$ in three parts, strictly lower and upper matrices $L$ and $U$ as well as the diagonal $D$, a symmetric Gauss-Seidel step can be expressed as follows.

$$(L + D)\boldsymbol{x}^{(k+1/2)} = \boldsymbol{b} - U\boldsymbol{x}^{(k)} \quad \text{(forward substitution)} \tag{1}$$

and

$$(U + D)\boldsymbol{x}^{(k+1)} = \boldsymbol{b} - L\boldsymbol{x}^{(k+1/2)} \quad \text{(backward substitution)} \tag{2}$$

with $\boldsymbol{x}^{(k)}$ being the $k^{\text{th}}$ iteration of $\boldsymbol{x}$. After computing the temporary vector $\boldsymbol{r} = U\boldsymbol{x}^{(k)}$ through a sparse matrix-vector multiplication (SpMV), the value of $\boldsymbol{x}^{(k+1/2)}$ is computed through a triangular solve (TRSV) operation by fulfilling the following equation.

$$(L + D)\boldsymbol{x}^{(k+1/2)} = \boldsymbol{b} - \boldsymbol{r}. \tag{3}$$

Thus, the right-hand side of (2) can be computed as follows.

$$\boldsymbol{b} - L\boldsymbol{x}^{(k+1/2)} = \boldsymbol{r} + D\boldsymbol{x}^{(k+1/2)}, \tag{4}$$

which leads to compute $\boldsymbol{x}^{(k+1)}$ from Eq. 2 as the result of a backward triangular solve. $A$ matrix forward and backward substitutions are replaced by the forward and backward substitutions of only the $L + D$ and $U + D$ matrices while saving almost half of the loads and operations with the expense of one fast SpMV, fast vectorizable element-wise product $D\boldsymbol{x}$, and vector additions/subtractions. Additionally, operations in the first, fine-grained smoother step of the V-shaped multi-grid are saved by using the zero initial guess $\boldsymbol{x}^{(0)} = 0$ that leads to $\boldsymbol{r} = 0$. This algorithm is also applied by other architectures such as Intel optimized HPCG provided with the MKL library and GPU HPCG implementations like rocHPCG [25, 28, 29].

In the MPI parallelized version, the matrix can be decomposed into a purely local part and a halo matrix containing domain boundary elements. This separation allows for some extent of overlap between computation and communication that improves scalability.

Furthermore, the matrix size should be appropriately specified. To perform efficient vector computing by keeping an enough long vector length, the y-axis and z-axis sizes need to be long. As the matrix size affects the convergence of the calculation results, the matrix size should be carefully selected considering the residual. This paper searches for the optimum matrix size. To reduce the search space, the matrix size suitable for a single node is searched. For a single node, $(nx, ny, nz) = (56, 216, 376)$ achieves the highest performance. Then, based on the suitable matrix size for a single node, the size for multiple nodes is searched. By fixing the value of $nz$ to 376 in order to keep the vector length, $nx$ and $ny$ are searched.

This paper also uses the partitioning mode to further exploit the potential of LLC. As the HPCG benchmark is an LLC bandwidth-bound program, the partitioning mode that reduces the contention to LLC is more suitable than the normal mode.

# 3. Evaluation

## 3.1. Evaluation Environments

**Table 1.** Computing systems used for the evaluations

| Systems | A300-8 | B401-8 | Xeon | EPYC | V100 | A100 |
|---|---|---|---|---|---|---|
| Host | 2×Xeon 6126 | EPYC 7402P | 2×Xeon 6126 | 2×EPYC 7702 | 2×Xeon 6126 | |
| Accelerator | VE 10B | VE 20B | - | - | V100 | A100 |
| # of nodes | 8 VE nodes | 576 VE nodes | 1 node | 68 nodes | 1 node | 1 node |
| Compiler | NEC 3.2.1 | NEC 3.2.0 | Intel 19.1.3.304 | Intel 19.1.2.254 | PGI 21.2-0 | |

For the evaluation, six various computing systems, SX-Aurora TSUBASA A300-8, SX-Aurora TSUBASA B401-8, Xeon, EPYC, V100, and A100, are used as shown in Tab. 1. The specifications of the processors used in the systems, VE 10B, VE 20B, two sockets of Xeon 6126, two sockets of EPYC 7702, V100, and A100, are shown in Tab. 2.

For the compiler, the proprietary NEC compiler for VEs is used. The compile option "-O4 -msched-block" is used, which allows the compiler to automatically optimize and schedule the instruction in a basic block. For the general-purpose processors such as Xeon and EPYC, the Intel compiler collection is used. For V100 and A100, the PGI compiler is used.

For the Himeno benchmark, the MPI versions are used. For the evaluation on multiple nodes, the weak scale version is developed. For Xeon 6126 and EPYC 7702, only the optimization for the domain decomposition is applied to the reference codes. For V100 and A100, the parameter tuning of system parameters is optimized [23].

For the HPCG benchmark on Xeon 6126 and EPYC 7702, only the optimization for the size tuning is applied to the reference codes. For A100, the HPCG code in the NVIDIA HPC-Benchmark 21.2 is used.

To measure the power consumption of processors, Vector Engine MMM-Command, Intel SoC Watch, and NVIDIA SMI are used. For the power consumption of the whole system,

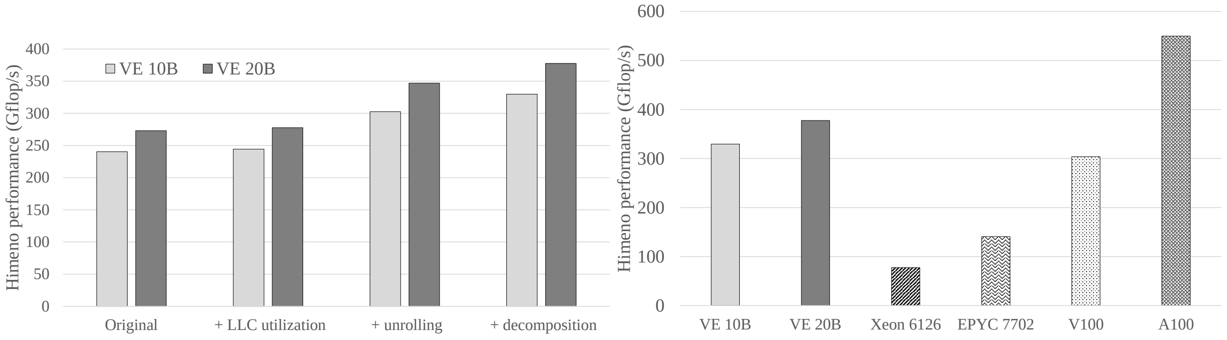
**Table 2.** Specification of processors used for evaluation

| | VE 10B | VE 20B | Xeon 6126 | EPYC 7702 | V100 | A100 |
|---|---|---|---|---|---|---|
| Number of cores | 8 | 8 | 12 | 64 | 5120 | 6912 |
| Peak SP (Tflop/s) | 4.30 | 4.92 | 1.766 | 4.096 | 14 | 19.5 |
| Peak DP (Tflop/s) | 2.15 | 2.46 | 0.883 | 2.048 | 7 | 9.7 |
| Memory | 6×HBM2 | 6×HBM2E | 6×DDR4 | 8×DDR4 | 4×HBM2 | 6×HBM2E |
| Mem. BW (GB/s) | 1228 | 1536 | 128 | 204.8 | 900 | 1555 |
| Mem. Cap. (GB) | 48 | 48 | 192 | 256 | 32 | 40 |
| LLC BW (TB/s) | 2.66 | 3.00 | - | - | 2.70 | 6.88 |
| LLC Cap. (MB) | 16 | 16 | 19.25 | 256 | 6 | 40 |

Supermicro IPMICFG is used. The execution times of the Himeno and HPCG benchmark are set to 10 minutes and two minutes, respectively.

## 3.2. Evaluation Results

### *3.2.1. Evaluation of the Himeno benchmark*



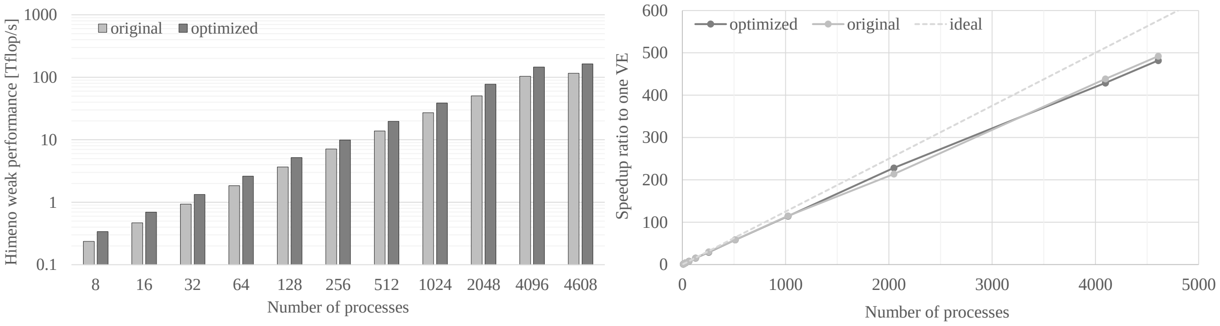
(a) Effects of the optimizations on a VE

(b) Performance of various processors

**Figure 4.** Performance of the Himeno benchmark on a single node

First, to examine the effects of the optimizations on a single VE node, Fig. 4a shows the performance on VE 10B and VE 20B. The vertical axis represents the Himeno performance. The horizontal axis represents each optimization. "+LLC utilization", "+unrolling", and "+decomposition" indicate that each optimization is applied in the order of LLC utilization, loop unrolling, and tuning of decomposition parameters from "Original". In the original code, the decomposition parameter is set to $(i, j, k) = (2, 2, 2)$.

Figure 4a shows that each optimization improves the Himeno performance. In particular, the loop unrolling has a great impact on the performance improvement. The loop unrolling achieves about 23.9 % and 24.9 % performance improvements on VE 10B and 20B, respectively. This is because the loop unrolling reduces loop overheads that is one of the large overheads. Moreover, the tuning of the decomposition parameters improves about 9.0 % and 8.7 % on VE 10B and VE 20B, respectively. The sequential memory access along the $i$ direction by the tuning of the domain decomposition contributes to the performance improvement. As a result, the single-node optimizations achieve about 37.3 % and 38.3 % sustained performance improvements on VE 10B and VE 20B compared to the original code, respectively. Figure 4a also shows that the performances of VE 20B are higher than those of VE 10B. This performance improvement is



(a) Performance

(b) Scalability

**Figure 5.** Weak scale performance of the Himeno benchmark on the B401-8 system

mainly brought by the improvement of the computational capability and the memory bandwidth of VE 20B.

To compare the Himeno performance of VE 10B and VE 20B with other processors, Fig. 4b shows the Himeno performance on various processors. This figure shows that A100 achieves the highest performance. A100 achieves about 43.3 % higher performance than VE 20B even though the peak memory bandwidth of VE 20B and A100 are almost the same. One of the reasons is that the reduction operation is heavy for vector computing. Since the vector length of a VE is long, the cost for the vector reduction becomes large. The other reason is that the high bandwidth of VE cannot be exploited due to the single-precision floating-point data. A packed memory load operation that treats two single-precision floating-point elements in a one load operation is not efficiently performed. Thus, A100 achieves higher performance compared with VE 20B. Compared with Xeon, EPYC, and V100, VE 10B and VE 20B achieve high performance. Since the memory and LLC bandwidths of VEs are the highest among them, VEs can achieve the highest performance.

In the aspect of the efficiency, the ratio of the sustained performance to the peak performance, VE 10B, VE 20B, Xeon 6126, EPYC 7702, V100, and A100 are 7.7 %, 7.7 %, 2.2 %, 1.7 %, 2.2 %, and 2.8 %, respectively. VE 10B and VE 20B achieve the highest efficiencies. Since VEs are carefully designed considering a balance between the sustained memory performance and sustained computational performance for memory-intensive applications, VEs can achieve the highest performance.

Next, the performance and weak scalability on a large-scale SX-Aurora TSUBASA are examined. Figure 5a shows the Himeno performance of the weak scaling on the B401-8 system. The problem size assigned to each process is fixed to the L size of $256 \times 256 \times 512$. This size is the maximum size for the memory capacity when eight processes are assigned to a VE. The vertical axis shows the sustained performance in the log scale. The maximum number of processes is 4608 that is equivalent to 576 VE nodes or 72 VH nodes. This figure shows that the optimized version achieves higher performance than that of the original version. About 43 % on average and about 53 % at maximum performance improvements are obtained by the optimizations. These results indicate that the proposed optimizations are essential to exploit performance on the large-scale vector computing system.

Figure 5a also shows that the increase in the number of processes improves the weak-scale performance. To examine the weak scalability in detail, Fig. 5b shows the weak scalability of the Himeno benchmark on the B401-8 system. The vertical axis shows the speedup ratio to a single VE node. The horizontal axis shows the number of processes. This figure shows that

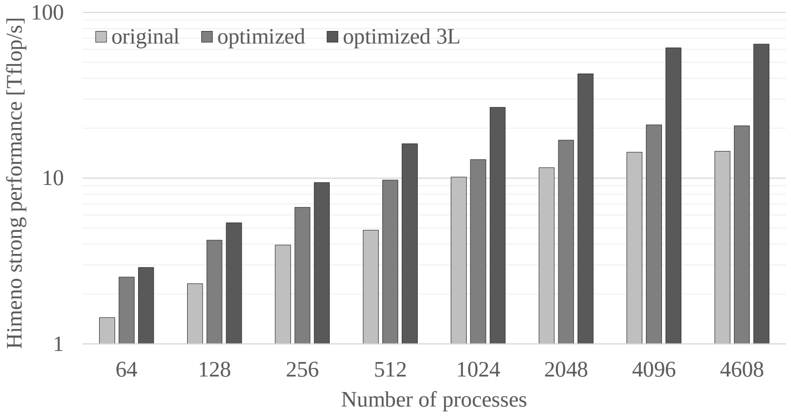
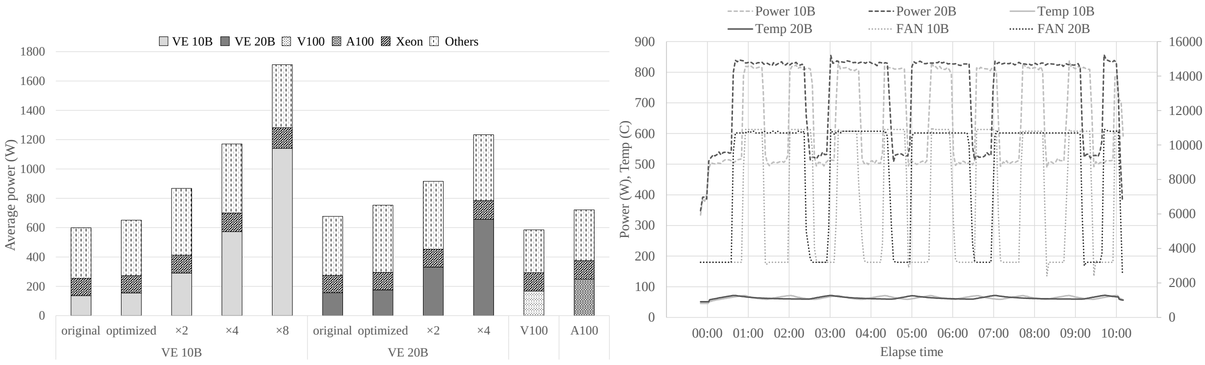


**Figure 6.** Strong scale performance of the Himeno benchmark on the B401-8 system



(a) Average power consumption

(b) Power consumption, fans, and temperature

**Figure 7.** Power consumption of the Himeno benchmark

good scalabilities are obtained in the original and optimized versions. Although the collective communication for residual in the Himeno benchmark leads to the decrease in scalabilities, the parallel efficiencies of the original and optimized versions reach 85.4 % and 83.7 % even when the number of processes is 4608.

The strong scalability on a large-scale SX-Aurora TSUBASA is examined. Figure 6 shows the Himeno performance of the strong scaling on the B401-8 system. The problem size is the XL size of $512 \times 512 \times 1024$ and the 3L size of $1024 \times 1024 \times 2048$. The vertical axis shows the sustained performance. This figure shows that the optimized version achieves higher performance than the original version. This is because the optimizations contribute to the performance improvements. In particular, when the number of processes is large, multi-node optimizations such as domain decomposition and process mapping impact the sustained performance.

Moreover, the performance of the 3L size is much higher than that of the XL size, especially when the number of processes is large. This is because the parallelisms for vectorization and parallelization in the XL size are not enough for the large-scale execution. As the number of processes increases, the performance differences between the XL size and the 3L size become large. Even the 3L size is not enough when the number of processes is large. To exploit the full potential of the system, the larger size needs to be selected according to the system size.

To clarify the power efficiency, the power consumption of SX-Aurora TSUBASA is examined. Since one of the limiting factors in the scale of computing systems is power consumption of each system, power efficiency and/or sustained performance per power is very important today and future. Figure 7 shows the power consumption when the weak scale performance of the Himeno benchmark is measured. Figure 7a shows the breakdown of the average power consumption of VEs, Xeon, GPUs, and "Others." "Others" includes the power consumption of the cooling fans, the memory modules of a VH, the power units, and the other server components. "Others" is calculated by subtracting the power consumption of the processor from the total power consumption. The horizontal axis represents the various processors of the original and optimized versions.

First, taking look at single VE cases, the power consumptions of the optimized version increases by comparing the original and optimized versions on VE 10B and VE 20B. This is because the cores and memory in VEs are fully operated by the optimizations. Moreover, VE 20B consumes about 13.4 % more power than VE 10B. This is due to the difference in the operating frequency of VE 20B and VE 10B. As the operating frequency of VE 20B is 1.6 GHz while it is 1.4 GHz in VE 10B, VE 20B is running at about 14.2 % faster frequency than VE 10B. As the power consumption is in proportion to the frequency, VE 20B consumes more power than VE 10B.

Compared with A100, the power consumptions of VE 10B and VE20B are low. However, the total power consumption of the A100 system is lower than those of the VE 10B and VE 20B systems. This is because the fine-grain fan control can be performed in the A100 system. Thus, the power consumption by the cooling fan in the A100 system becomes low.

The power consumption of "Others" in cases of a single VE occupies about 60 % of the total power consumption. One of the power consuming components is the cooling fans, especially when the fans rotate at very high speed when the temperature of VEs increases. The power consumption of "Others" on VE 20B is higher than that on VE 10B. This is because the temperature of VE 20B easily increases compared with VE 10B due to the high frequency of VE 20B.

To investigate the relationship between the power consumption and the cooling fans, Fig. 7b shows the total power consumption, the number of rotations of fans, and the temperature on VE 10B and VE 20B. The vertical axis in the left shows the power consumption and the temperature of VEs. The vertical axis in the right shows rotations per second of the cooling fan. The horizontal axis shows the elapsed time. This figure suggests that the cooling fan runs at the high speed from when the temperature of VEs rises to 70 degrees until when it drops to 60 degrees. This figure also shows that the fan of VE 10B often runs high rotations compared to VE20. This result implies that VE10 is easier to be cold enough to stop the fans than VE20 because the operating frequency of VE 10B is lower than that of VE 20B.

Second, in the cases of multiple VEs, the power consumption of VEs almost proportionally increases as the number of VEs increases in both cases of VE 10B and VE 20B. Since most computation is performed on VEs, the power consumption increases according to the number of VEs while those of Xeon and others slightly increase.

Figure 8 shows the power efficiency that divides the sustained performance by the average power. "Processor" indicates the power efficiency of the processor, i.e., the performance divided by the average power of only the processor. "System" indicates the power efficiency of the whole system. First, taking look at single node cases, this figure shows that the optimizations contribute to the power efficiencies as well as the sustained performance. The efficiencies of VE 10B and VE 20B are improved by about 18.0 % and 18.7 %, respectively. Since the increase in the power consumption can be amortized by the increase in the sustained performance, the power

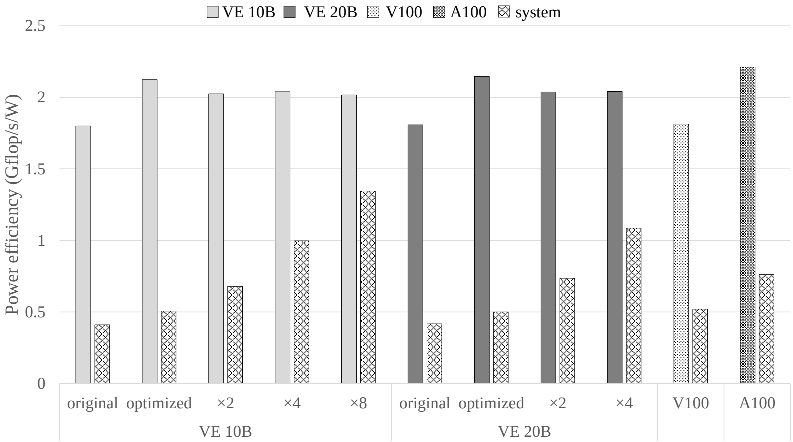
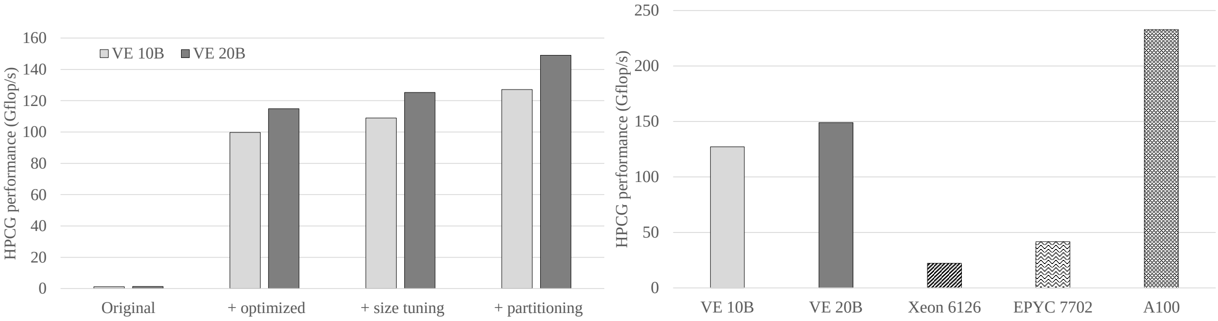


**Figure 8.** Power efficiency



(a) Effects of the optimizations on a VE

(b) Performance of various processors

**Figure 9.** Performance of the HPCG benchmark on a single node

efficiencies can be improved. Moreover, the power efficiencies of VE 10B, VE 20B, and A100 are similar. Even the previous generation of VE 10B achieves high power efficiency. Although the performances of VE 10B and VE 20B are lower than that of A100, the power consumption of VE 10 and VE 20B are lower than that of A100. As a result, VE 10B and VE 20B achieve a similar power efficiency to A100.

In the cases of multiple VEs, the power efficiencies of “processors” in VE 10B and VE 20B gradually decrease as the number of VEs increases. This is because the sustained performance does not ideally scale according to the number of VEs, although the power consumption increases according to the number of VEs. On the other hand, the power efficiencies of “system” increase as the number of VEs increases. As the total power consumption does not proportionally increase to the number of VEs, the increase in the total power consumption can be amortized by the increase in the sustained performance.

The process technologies used in VE 10B and VE 20B are TSMC 16 nm while the process technology of A100 is TSMC 7 nm. Even though VEs use the two-generation old process technology, the power efficiencies of VEs are similar to A100 that uses the latest process technology. If VEs use the same process technology, the power efficiencies should be much higher than that of GPUs. This result clarified that the vector architecture is a power efficient architecture that is one of important features as the power constraints become stricter in the future.

### *3.2.2. Evaluation of the HPCG benchmark*

To examine the effects of the optimizations, Fig. 9a shows the performance of the optimizations on VE 10B and VE 20B. The vertical axis represents the sustained performance of the HPCG benchmark. The horizontal axis represents each optimization. “Original” indicates the reference version of HPCG. “+ optimized” indicates the version for the vector optimizations. “+ size tuning” indicates tuning of the problem size to $(56 \times 216 \times 376)$ from the initial problem size of $(104 \times 104 \times 104)$. “+ partitioning” uses the partitioning mode. The optimizations are applied in the order of the vector optimizations, the tuning of the matrix size, and the partitioning mode from left to right in the figure.

This figure shows that each optimization improves the HPCG performance. The optimized version that applies the ELL data format, hyperplanes or level scheduling ordering for vectorization, and cache retention for the LLC, and reduction in the instructions significantly improve the performance. The main reason is due to the increase in the vectorization ratio and the average vector length. By the optimizations, the vectorization ratio is improved to 99.2 % from 73.7 % and the average vector length is drastically improved to 236.2 from 27.9. Therefore, the performance improves by 86.8 times compared with the original version.

Figure 9a also shows that the tuning of the matrix size further improves the performance by about 9 %. As the matrix size affects the size of hyperplane slices, the average vector length is improved to 241.2.

Furthermore, the partitioning mode further improves the performance. By the partitioning mode, the execution time of the load instruction becomes about 10 % shorter than that of the normal mode. The short execution time of the load instruction by the reduction in the network conflicts between LLC and the memory further results in about 19 % performance improvement.

Moreover, Fig. 9a shows that the performances of VE 20B are about 15 to 17 % higher than those of VE 10B. The main reason for the improvement is that the LLC bandwidth of VE 20B is higher than that of VE 10B. Since the LLC bandwidth of VE 20B is improved by about 12.8 %, it contributes to the higher performance.

Compared with the other processors, Fig. 9b shows the performance on VE 10B, VE 20B, two sockets of Xeon 6126, two sockets of EPYC 7702, and A100. The horizontal axis represents processors. This figure shows that A100 achieves the highest performance. The performance of A100 is 1.56 times faster than that of VE 20B. Compared with Xeon and EPYC, VE 10B and VE 20B achieve much high performance. One of the reasons is that the LLC bandwidth of A100 is higher than that of VE 20B. The theoretical LLC bandwidth of A100 is 6.88 TB/s [8], while it is 3.00 TB/s on VE 20B. As the LLC bandwidth of A100 is more than 2.29 times higher, the LLC bandwidth improves the HPCG performance on A100. The other reason is that VE 20B cannot fully exploit its high memory bandwidth due to the memory latency by the indirect memory accesses. As a result, the difference of the HPCG performance between VE 20B and A100 becomes larger than that in the case of the stream memory bandwidth.

On the other hand, the efficiencies of VEs are the highest. The efficiencies of VE 10B, VE 20B, Xeon 6126, EPYC 7702, A100 are 5.9 %, 6.1 %, 1.3 %, 1.0 %, and 2.4 %, respectively. Due to the balanced architecture of SX-Aurora TSUBASA for bandwidth-bound applications, VE 10B and VE 20B achieve higher efficiencies than the other processors.

Figure 10 shows the scalability of the HPCG benchmark on the B401-8 and EPYC systems. The vertical axis represents the speedup ratio to single process performance. The horizontal axis represents the number of processes. This figure shows that a good scalability of VE 20B

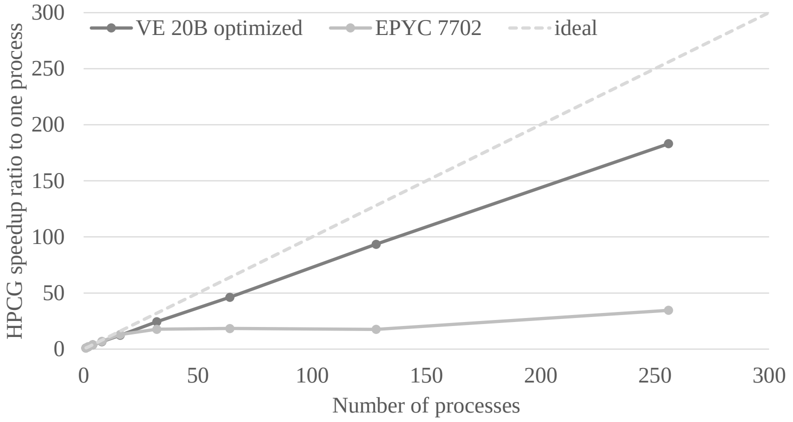
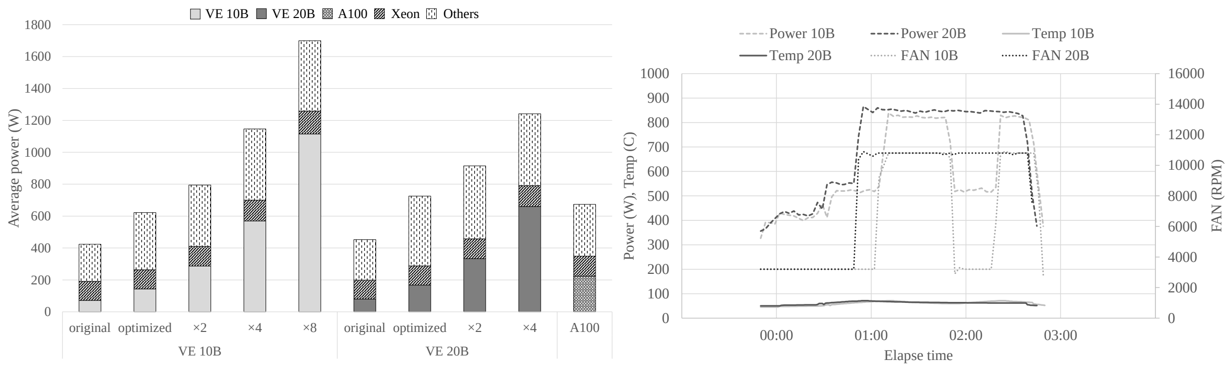


**Figure 10.** Scalability of the HPCG benchmark



(a) Average power consumption

(b) Power consumption, fans, and temperature

**Figure 11.** Power consumption of the HPCG benchmark

is obtained. When the number of processes is 256, the parallel efficiency is about 71.6 %. As the HPCG benchmark is weak scaling, the performance scales fine even when the number of processes is large.

In the case of EPYC 7702, the scalability is not good compared with VE 20. The memory bandwidth of EPYC 7702 scales up to 32 processes, and then, the memory bandwidth is saturated when the number of processes is 64 or more, As the HPCG performance on EPYC 7702 is limited by the memory bandwidth, the scalability of EPYC 7702 also becomes saturated when the number of processes is 64 or more.

To examine the power consumption, Fig. 11a shows the breakdown of the average power consumption. This figure shows that the power consumption of a VE 10B and a VE 20B increases by the optimizations. As the optimized version with the partitioning mode efficiently uses VEs, the power consumption increases. On the other hand, as the original version cannot exploit the performance of VEs, the power consumption is low.

Furthermore, the power consumption of A100 is higher than those of VE 10B and VE 20B although the total power consumption of the A100 system is lower than those of the VE 10B and VE 20B systems. This is because of the difference of the fan control mechanism between the A100 system and the VE systems, which is also discussed in the Himeno benchmark. As the fine-grain fan control can be performed in the A100 system, the power consumption by the cooling fan becomes low.

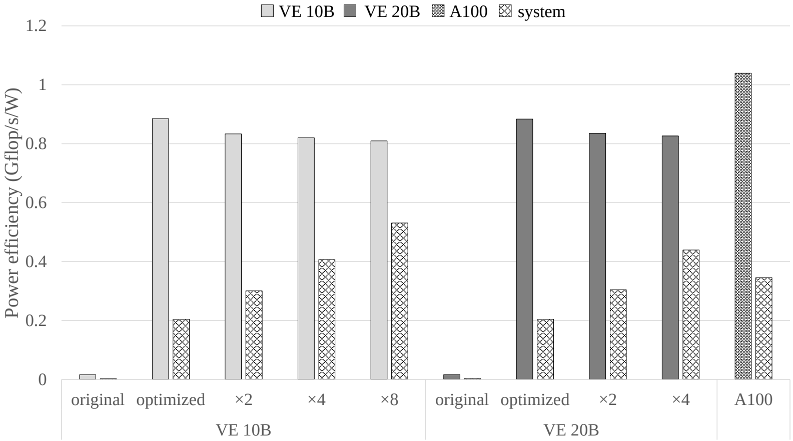


**Figure 12.** Power efficiency

Figure 11a also shows the power consumption on VE 20B is higher than those on VE 10B. To investigate the reason, Fig. 11b shows the total power consumption, the number of rotations of fans, and the temperature on VE 10B and VE 20B. The vertical axis in the left shows the power consumption and the temperature of VEs. The vertical axis in the right shows rotations per second of the cooling fan. The horizontal axis shows the elapsed time. This figure shows that the total power consumption of VE 20B becomes high because the cooling fans of VE 20B rotate more often than that of VE 10B. This is because VE10 is easier to be cold enough to reduce the number of rotations of the fan than VE20. Furthermore, the fan in the HPCG benchmark runs at the high rotation more often than that in the Himeno benchmark. Since the characteristics of the benchmarks differ from each other, it affects the frequency of the high rotations of the fan.

Figure 12 shows the power efficiency of the HPCG benchmark. The power efficiencies of a VE 10B and a VE 20B are almost the same even though the performance of VE 10B is lower than that of VE 20B even in the HPCG benchmark as well as the Himeno benchmark. As the power consumption of VE 10B is lower than that of VE 20B, the power efficiency of VE 10B equals to that of VE 20B. Moreover, the power efficiency of A100 is higher than those of VE 10 and VE 20B. Although the power consumption of VE 10 and VE 20B is lower than that of A100, the sustained performance of A100 is much higher than those of VE 10B and VE 20B. As a result, A100 achieves higher power efficiency than VE 10B and VE 20B. The reason is that the process technologies used in VE 10B and VE 20B are two-generation old compared with A100. If VEs use the same process technology, it is expected that VEs can achieve higher power efficiency than A100.

In the cases of multiple VEs, the power efficiencies of VE 10B and VE 20B gradually decrease as the number of VEs increases. This is because the sustained performance does not ideally scale according to the number of VEs, although the power consumption increases according to the number of VEs. On the other hand, the power efficiencies of “system” increase as the number of VEs increases. As the total power consumption does not increase in proportion to the number of VEs, the increase in the total power consumption can be amortized by the increase in the sustained performance.

## 4. Related Work

The performance optimization and evaluation of vector computing systems have been continuously conducted [11, 12, 15, 24]. Komatsu et al. have evaluated the first generation of SX-

Aurora TSUBASA using benchmarks including the Himeno benchmark. From the evaluation, it is clarified that the first generation of SX-Aurora TSUBASA has advantages of memory-intensive benchmark compared with Xeon Skylake and SX-ACE. However, the performance on multiple nodes is not evaluated. This paper extends the optimization of the Himeno benchmark for multiple nodes such as domain decomposition and processing mapping considering the bandwidth and clarifies the performance and scalability on multiple nodes of SX-Aurora TSUBASA.

Furthermore, the performance evaluation on the second generation of SX-Aurora TSUBASA has been reported [11]. It clarifies that the performance and power efficiencies of HPCG and HPL of VE 20B are higher than those of Xeon and EPYC. However, the detailed analysis of performance and power efficiency is not conducted. This paper further optimizes the HPCG benchmark for the large-scale vector computing systems by the size tuning. Moreover, this paper deeply evaluates and analyzes the Himeno and HPCG benchmark performances on the large-scale vector computing systems in terms of the effects of optimizations, scalability, average power consumption, power efficiency. As a result of the evaluation and deep analysis, it can be clarified that the power efficiency of a vector architecture is high and promising for the future HPC systems.

Hartwig et al. have evaluated the memory bandwidth of the A100 and the performance of sparse and batched computations [6]. This paper evaluates the performance and the power efficiency of V100 and A100 with the Himeno benchmark. It shows higher performance of A100 than that of V100. By comparing the performance of GPUs and VEs, this paper clarifies the characteristics of these processors.

## Conclusions

The peak performance of recent HPC systems has been remarkably improved by the increase in the number of nodes. This approach to improve the performance has also brought the increase in the power consumption of the HPC systems. However, due to the limitation of the power budget for each facility, the conventional approach of simply increasing the number of nodes to improve the performance is not realistic in near future. Therefore, a paradigm shift to a new approach is essential to keep improving the performance within the limited power constraints for the design of future HPC systems.

This paper focuses on a vector computing system adopting a long vector processing that has a potential to realize high performance with high power efficiency under the strict power constraints. To achieve high power efficiency, this paper improves the sustained performance by the optimizations for vector computing system. As the target programs, this paper selects two benchmark programs, the Himeno and HPCG benchmarks, and applies vector optimizations for a vector computing system in aspect of a single node and multiple nodes.

By deep analysis through the performance evaluation, the sustained performance, the scalability, and the power consumption, the power efficiency of a large-scale SX-Aurora TSUBASA are clarified. For the Himeno benchmark, VE 10B and VE 20B achieve about 37.3 % and 38.3 % performance improvements by a single node optimizations compared to the original code, respectively. VE 10B and VE 20B achieve about 7.7 % efficiency, which is the highest efficiency among various processors. For the HPCG benchmark, VE 10B and VE 20B can achieve about 112 and 113 times performance improvements by single node optimizations compared to the reference code, and about 5.9 % and 6.1 % efficiencies, respectively, which also are the highest efficiencies among various processors. Furthermore, it is clarified that SX-Aurora TSUBASA could achieve the highest power efficiencies among the latest processors such as an Intel processor, an EPYC processor, and a GPU even though VEs adopt the previous generation of the process technology. This fact suggests that the vector computing with a long vector length can achieve a high power efficient computing and the vector architecture could be most efficient if it used the latest process technology. Therefore, this paper clarified that the vector computing is one of the promising ways to survive in the design of the future computing system with the strict power constraints.

## Acknowledgements

The authors would like to thank Christie Alappat from the University of Erlangen-Nuremberg for the assistance and inspiring discussions on HPCG algorithm tuning. The authors also thank large-scale HPC challenge of Cyberscience Center, Tohoku University for the large-scale executions of the supercomputing systems. This research was partially supported by MEXT Next Generation High-Performance Computing Infrastructures and Applications R&D Program, entitled “R&D of A Quantum-Annealing-Assisted Next Generation HPC Infrastructure and its Applications,” Grants-in-Aid for Scientific Research (A) #19H01095, Grants-in-Aid for Scientific Research (C) #20K11838, and Japan-Russia Research Cooperative Program between JSPS and RFBR, Grant number JPJSBP120214801.

## References

1. Himeno benchmark. `http://i.riken.jp/en/supercom/documents/himenobmt/`, accessed: 2021-05-31

2. HPCG benchmark. `https://www.hpcg-benchmark.org/`, accessed: 2021-05-31

3. MVAPICH: MPI over InfiniBand, Omni-Path, Ethernet/iWARP, and RoCE. `http://mvapich.cse.ohio-state.edu/benchmarks/`, accessed: 2021-05-31

4. STREAM: Sustainable Memory Bandwidth in High Performance Computers. `https://www.cs.virginia.edu/stream/`, accessed: 2021-05-31

5. TOP500 Supercomputer Sites, `http://www.top500.org/`

6. Anzt, H., Tsai, Y.M., Abdelfattah, A., *et al.*: Evaluating the performance of NVIDIAs A100 ampere GPU for sparse and batched computations. In: 2020 IEEE/ACM Performance Modeling, Benchmarking and Simulation of High Performance Computer Systems (PMBS). pp. 26–38. IEEE (2020). `https://doi.org/10.1109/PMBS51919.2020.00009`

7. Cho, J.H., Kim, J., Lee, W.Y., *et al.*: A 1.2V 64Gb 341GB/S HBM2 stacked DRAM with spiral point-to-point TSV structure and improved bank group data control. In: 2018 IEEE International Solid - State Circuits Conference - (ISSCC). pp. 208–210. IEEE (2018). `https://doi.org/10.1109/ISSCC.2018.8310257`

8. Choquette, J., Gandhi, W.: NVIDIA A100 GPU: Performance innovation for GPU computing. In: 2020 IEEE Hot Chips 32 Symposium (HCS). pp. 1–43. IEEE (2020). https://doi.org/10.1109/HCS49909.2020.9220622

9. Dongarra, J., Heroux, M.A., Luszczek, P.: High-performance conjugate-gradient benchmark: A new metric for ranking high-performance computing systems. The International Journal of High Performance Computing Applications 30(1), 3–10 (2016). https://doi.org/10.1177/1094342015593158

10. Egawa, R., Komatsu, K., Takizawa, H.: Designing an open database of system-aware code optimizations. In: 2017 Fifth International Symposium on Computing and Networking (CANDAR). pp. 369–374. IEEE Computer Society (2017). https://doi.org/10.1109/CANDAR.2017.102

11. Egawa, R., Fujimoto, S., Yamashita, T., *et al.*: Exploiting the potentials of the second generation SX-Aurora TSUBASA. In: 2020 IEEE/ACM Performance Modeling, Benchmarking and Simulation of High Performance Computer Systems (PMBS). pp. 39–49. IEEE (2020). https://doi.org/10.1109/PMBS51919.2020.00010

12. Egawa, R., Komatsu, K., Isobe, Y., *et al.*: Performance and power analysis of SX-ACE using HP-X benchmark programs. In: 2017 IEEE International Conference on Cluster Computing (CLUSTER). pp. 693–700. IEEE Computer Society (2017). https://doi.org/10.1109/CLUSTER.2017.65

13. Egawa, R., Komatsu, K., Kobayashi, H.: Designing an HPC refactoring catalog toward the exa-scale computing era. In: Resch, M.M., Bez, W., Focht, E., Kobayashi, H., Patel, N. (eds.) Sustained Simulation Performance 2014. pp. 91–98. Springer (2015). https://doi.org/10.1007/978-3-319-10626-7_8

14. Egawa, R., Komatsu, K., Momose, S., *et al.*: Potential of a modern vector supercomputer for practical applications: performance evaluation of SX-ACE. The Journal of Supercomputing 73(9), 3948–3976 (2017). https://doi.org/10.1007/s11227-017-1993-y

15. Egawa, R., Momose, S., Komatsu, K., Isobe, Y., Musa, A., Takizawa, H., Kobayashi, H.: Early evaluation of the SX-ACE processor. In: The poster at International Conference for High Performance Computing, Networking, Storage and Analysis (SC14) (2014)

16. Focht, E.: HPCG Performance Efficiency on VE at 5.99%. https://sx-aurora.github.io/posts/hpcg-tuning/ (2019), accessed: 2021-06-09

17. Heroux, M.A., Dongarra, J., Luszczek, P.: HPCG benchmark technical specification (2013). https://doi.org/10.2172/1113870

18. Hou, S.Y., Chen, W.C., Hu, C., *et al.*: Wafer-level integration of an advanced logic-memory system through the second-generation CoWoS technology. IEEE Transactions on Electron Devices 64(10), 4071–4077 (2017). https://doi.org/10.1109/TED.2017.2737644

19. Komatsu, K., Egawa, R., Hirasawa, S., *et al.*: Migration of an atmospheric simulation code to an OpenACC platform using the Xevolver framework. In: 2015 Third International Symposium on Computing and Networking (CANDAR). pp. 515–520. IEEE Computer Society (2015). https://doi.org/10.1109/CANDAR.2015.102

20. Komatsu, K., Egawa, R., Hirasawa, S., *et al.*: Translation of large-scale simulation codes for an OpenACC platform using the Xevolver framework. International Journal of Networking and Computing 6(2), 167–180 (2016). `https://doi.org/10.15803/ijnc.6.2_167`

21. Komatsu, K., Egawa, R., Isobe, Y., *et al.*: 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. pp. 1–2 (2015)

22. Komatsu, K., Egawa, R., Takizawa, H., *et al.*: Exploring system architectures for next-generation CFD simulations in the postpeta-scale era. Journal of Fluid Science and Technology 9(5), JFST0073–JFST0073 (2014). `https://doi.org/10.1299/jfst.2014jfst0073`

23. Komatsu, K., Kishitani, T., Sato, M., *et al.*: An appropriate computing system and its system parameters selection based on bottleneck prediction of applications. In: 2019 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW). pp. 768–777. IEEE (2019). `https://doi.org/10.1109/IPDPSW.2019.00127`

24. Komatsu, K., Momose, S., Isobe, Y., *et al.*: Performance evaluation of a vector supercomputer SX-Aurora TSUBASA. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis. pp. 54:1–54:12. SC '18, IEEE Press (2018). `https://doi.org/10.1109/SC.2018.00057`

25. Liu, Y., Yang, C., Liu, F., *et al.*: 623 Tflop/s HPCG run on Tianhe-2: Leveraging millions of hybrid cores. The International Journal of High Performance Computing Applications 30(1), 39–54 (2016). `https://doi.org/10.1177/1094342015616266`

26. Oh, C.S., Chun, K.C., Byun, Y.Y., *et al.*: 22.1A 1.1V 16GB 640GB/s HBM2E DRAM with a Data-Bus Window-Extension Technique and a Synergetic On-Die ECC Scheme. In: 2020 IEEE International Solid- State Circuits Conference - (ISSCC). pp. 330–332. IEEE (2020). `https://doi.org/10.1109/ISSCC19947.2020.9063110`

27. Onodera, A., Komatsu, K., Fujimoto, S., *et al.*: Optimization of the himeno benchmark for SX-Aurora TSUBASA. In: Wolf, F., Gao, W. (eds.) Benchmarking, Measuring, and Optimizing. Lecture Notes in Computer Science, vol. 12614, pp. 127–143. Springer (2021). `https://doi.org/10.1007/978-3-030-71058-3_8`

28. Park, J., Smelyanskiy, M., Vaidyanathan, K., *et al.*: Optimizations in a high-performance conjugate gradient benchmark for IA-based multi- and many-core processors. The International Journal of High Performance Computing Applications 30(1), 11–27 (2016). `https://doi.org/10.1177/1094342015593157`

29. Phillips, E., Fatica, M.: Performance analysis of the high-performance conjugate gradient benchmark on GPUs. The International Journal of High Performance Computing Applications 30(1), 28–38 (2016). `https://doi.org/10.1177/1094342015599239`

30. Yamada, Y., Momose, S.: Vector engine processor of NEC's brand-new supercomputer SX-Aurora TSUBASA. In: International symposium on High Performance Chips (Hot Chips2018) (2018)