SN Computer Science (2024) 5:623
https://doi.org/10.1007/s42979-024-02958-3

ORIGINAL RESEARCH

First Impressions of the Sapphire Rapids Processor with HBM for Scientific Workloads

Eva Siegmann[1] · Robert J. Harrison[1] · David Carlson[1] · Smeet Chheda[1] · Anthony Curtis[1] · Firat Coskun[1] · Raul Gonzalez[1] · Daniel Wood[1] · Nikolay A. Simakov[1,2]

Received: 4 October 2023 / Accepted: 29 April 2024

Abstract

The landscape of high performance computing (HPC) has witnessed exponential growth in processor diversity, architectural complexity, and performance scalability. With an ever-increasing demand for faster and more efficient computing solutions to address an array of scientific, engineering, and societal challenges, the selection of processors for specific applications becomes paramount. Achieving optimal performance requires a deep understanding of how diverse processors interact with diverse workloads, making benchmarking a fundamental practice in the field of HPC. Here, we present preliminary results observed over such benchmarks and applications and a comparison of Intel Sapphire Rapids and Skylake-X, AMD Milan, and Fujitsu A64FX processors in terms of runtime performance, memory bandwidth utilization, and energy consumption. The examples focus specifically on the Sapphire Rapids processor with and without high-bandwidth memory (HBM). An additional case study reports the performance gains from using Intel's Advanced Matrix Extensions (AMX) instructions, and how they along with HBM can be leveraged to accelerate AI workloads. These initial results aim to give a rough comparison of the processors rather than a detailed analysis and should prove timely and relevant for researchers who may be interested in using Sapphire Rapids for their scientific workloads.

Keywords Benchmarking · Performance · Energy usage · Intel Sapphire rapids · Fujitsu A64FX · AMD Milan · Intel Skylake · HBM · AMX · TMUL

Introduction

This paper presents our preliminary work of benchmarking and comparing performance on different processors with a special focus on the Sapphire Rapids' (SPR) [1] and their HBM feature, which promises a performance boost for memory bandwidth-limited applications. An early study using the SPR architecture reported more than 8.5x faster runtimes for multi-physics codes relative to Intel's Broadwell architecture when utilizing high bandwidth memory [2]. The paper investigated the runtime of two different hydrodynamics applications on Intel Broadwell, as well as on SPR with and without HBM. Another study examined bandwidth limitations in the SPR processor and concluded that the lack of sufficient concurrency in the cores of the processor affects bandwidth and explains why the peak is never achieved when using HBM [3]. Wang et al. [4] explored the effects of HBM on several benchmarks and applications, finding that many scientific applications benefit from it. [5] analyzed the performance of AMD Genoa and Intel Sapphire Rapids CPUs and compared them to older CPU models. The authors used the HPL, HPCC, and NAS parallel benchmarks, as well as LAMMPS, GROMACS, and NWChem for the comparison. The paper concludes that the Intel Sapphire Rapids as well as the AMD Genoa CPUs provide a significant performance boost of 20% to 50% compared to older AMD and Intel CPUs. [6] investigated the SPEChpc 2021 benchmark suite, in MPI-only mode, on Intel Ice Lake and Sapphire Rapids and analyzed the performance in terms of runtime and power/energy.

In this work, we extend the previous studies by studying a different set of benchmarks and applications and comparing the performance on a diverse set of architectures, i.e. Fujitsu A64FX, Intel Skylake, and AMD Milan. The latest AMD

✉ Eva Siegmann
eva.siegmann@stonybrook.edu

[1] Institute for Advanced Computational Science, Stony Brook University, Stony Brook, NY 11794, USA

[2] Center for Computational Research, SUNY University at Buffalo, Buffalo, NY 14203, USA

Published online: 07 June 2024

model is not included in the study due to the unavailability for experimentation or analysis within the research environment. All benchmarks and applications were compiled with full optimizations for each architecture. Selected instances were chosen to juxtapose Intel Sapphire Rapids with NVIDIA's Grace-Grace and Grace-Hopper superchips [7].

We also share the observed performance gains from using the new AMX instruction set extension to the AVX 512 ISA. SPR is the first chip to implement the tiling instructions. These tiling instructions are run on an accelerator, a tile matrix multiply unit (TMUL). This TMUL unit operates on data stored in separate 2-D registers representing a tile. [8] shows improved inference performance of BERT, a Deep Learning model, by applying quantization and operator fusion on top of Intel's AMX features. Their results are comparable to NVIDIA's T4 GPU for smaller batch sizes. In our case study, we report the performance of multiple input problems for the convolution operator and see how different types of memory and hardware support for input data types affect these results.

The results presented here are a first step and will be extended in the future. This paper is laid out in the following manner. We discuss the benchmarking protocol, micro-benchmark applications, and three science applications in Section "Materials and Methods". The next section discusses the observed performance for the aforementioned benchmarks and compares it to other systems. We also include an investigation on how changing various BIOS settings affects the performance and energy consumption of a selected application. This analysis was initially necessary to choose settings that kept the power consumption within data center limits while not decreasing the performance of the nodes. Finally, we discuss our key observations.

Materials and Methods

Benchmarking protocol: The systems evaluated in this benchmark analysis are listed in Table 1. The used compilers, MPIs, as well as processor-specific flags can be found in the appendix. Typically, each application or micro-benchmark was executed five to ten times on each system, and mean values were reported. HBM on the SPR nodes was configured in Flat Mode with Sub-NUMA Clustering 4 (SNC4). In this configuration, memory is spread across 16 NUMA regions, with regions 0–7 containing cores, and all (0–15) with both DDR5 and HBM. To preferentially utilize HBM over DDR5 memory where possible, the tested applications and micro-benchmarks were run with `numactl -preferred-many=8-15`, unless specified otherwise. This setup was shown in our tests to produce the same performance as binding applications to specific numa nodes while being less cumbersome to execute. Hereafter, SPR-HBM denotes runs on the SPR processor wherein HBM was preferentially utilized, while SPR-DDR signifies runs with default memory configurations. Energy consumption was measured using each system's ipmitool [9].

The analyzed metrics are:

Runtime: The runtime, or time to solution, is often the primary concern of users as it is the limiter for discovery or publication deadlines. It can be a good assessor of how scalable the combination of software and hardware is.

Energy consumption: This is primarily most important to system managers who seek to minimize overall energy consumption. But increasingly this information is passed to users to inform climate-conscious actions.

Table 1 Studied systems

Processor	Cores	Clock speed (GHz)	Memory (GB)
Intel® Xeon® CPU Max 9468 ('Sapphire Rapids')	96	2.6	256 DDR5 + 128 HBM
Fujitsu A64FX-FX700	48	1.8	32 HBM
AMD EPYC 7643 ('Milan')	96	3.2	256 DDR4
Intel® Xeon® Gold 6148 ('Skylake')	40	2.4	192 DDR4
Processor	NUMA Regions	Infinband Network	Vector Instructions
Intel® Xeon® CPU Max 9468 ('Sapphire Rapids')	16	NDR (400 Gbit/s)	AVX512, Intel DL Boost, AMX extensions
Fujitsu A64FX-FX700	4	HDR (100 Gbit/s)	SVE instructions
AMD EPYC 7643 ('Milan')	8	HDR (100 Gbit/s)	AVX2 instructions
Intel® Xeon® Gold 6148 ('Skylake')	2	FDR (56 Gbit/s)	AVX512 instructions

Efficiency: This metric is mainly interesting to system managers, who are seeking good overall system throughput and cost-efficient utilization.

Peak power consumption: The metric alone does not deliver a lot of useful information, but for a given performance/efficiency/or other target it can inform system design and is a key constraint in peak performance.

For selected benchmarks, results on Graviton3 CPUs and NVIDIA A100 GPUs are included, as those results are available from a previous study, [10].

Benchmarks

Several smaller benchmarks are investigated to test multiple attributes of the systems. They are listed and described below:

DAXPY and simple memory copy were used to analyze the bandwidth memory on SPR with and without HBM.

STREAM - We compile the standard source code for all processors except for A64FX, where a tuned version of STREAM is publicly available. The array size is chosen such that at least half of the available node memory is in use. The benchmark is run on all architectures at full subscription.

The **HPCC** (HPC Challenge) benchmark [11] combines multiple benchmarks. Here we are reporting on three of them: High Performance LINPACK (HPL), Matrix-Matrix multiplication and Fast Fourier Transform (FFT). LINPACK solves a linear system of equations using all cores in parallel. The performance is measured in Giga Floating Point Operations Per Second (GFLOPS) and corresponds to the performance of the application on all allocated compute resources. We also report GFLOPS/Core, which are the total GFLOPS divided by the number of cores.

The **HPCG** (The High-Performance Conjugate Gradients) benchmark is an alternative to the HPL benchmark (used in HPCC) and utilizes methods and patterns commonly used in many PDE solvers [12]. Unlike HPCC, HPCG does not rely on external libraries but requires vendors to optimize their own version of HPCG. Thus, for x86 machines, we used the Intel version of HPCG, and for the A64FX the version from Cray.

oneAPI Deep Neural Network Library or oneDNN, a part of oneAPI, is an open-source library providing optimized deep learning primitives for CPUs and GPUs. Many deep learning frameworks like PyTorch, and TensorFlow use oneDNN as a backend for accelerated computing on CPUs. oneDNN has the capability to detect the underlying instruction set architecture and uses Just-In-time (JIT) code generation to deploy optimized kernels at runtime. In this study, we explore the performance of new features included in the SPR processors, namely Advanced Matrix Extensions, and compare them to the best dispatch in Skylake-X and A64FX processors for the same problem, inputs, and run configuration. To prevent any unfair disadvantages in our study, we do not include the AMD Milan processor since it does not have special 512-bit vector processing elements.

Applications

Scientific applications are investigated to allow for a better understanding if, and how, the SPR with and without HBM can benefit real-life applications.

GROMACS is a software package for the simulation of biomolecular systems like proteins, membranes, DNA, and RNA [13]. It calculates how atoms move over time under a classical physics approximation by solving Ordinary Differential Equations based on Newton's second law. Three systems, consisting of 82K, 200K, and 1.2M atoms, were used as benchmark [14].

OpenFOAM is a library and a collection of applications for the numerical solution of Partial Differential Equations [15]. The test case is a calculation of incompressible airflow around a motorcycle and is based on a test included in the OpenFOAM suite (`incompressible\simpleFoam\motorBike`). We have increased the initial grid in each direction 2, 4 and 6 times to increase resolution and problem size. The grid is further refined around the obstacle, and the Navier–Stokes equations are solved on an unstructured grid. The resulting mesh consists of 2 M, 11 M, and 35 M cells. Total wall time was used as a performance metric (i.e., smaller = better).

ROMS, Regional Ocean Modeling System [16, 17], is an ocean model widely used in the scientific community. It is a free-surface, terrain-following, primitive equations ocean model. The test case we use for this work simulates the flow around the west Antarctic Peninsula [18]. ROMS was picked as a test application because it is an important local application with performance characteristics representative of other structured grid codes.

HPCC, HPGC, GROMACS, and OpenFOAM were executed on all cores of a single node (96 cores) using MPI for parallel execution.

The input parameters and automation procedures (batch job submission and monitoring, output parsing) for HPCC, HPCG, GROMACS, and OpenFOAM were adopted from the XDMoD application kernel module [10, 19].

Results

Here we focus on the runtime, memory bandwidth, as well as on energy consumption.

Table 2 Benchmarks: STREAM, HPCC(DGEMM, Linpack and FFT)

CPU/System	Cores	STREAM TRIAD, GB/s	Matrix Multiplication		LINPACK		FFT	
			GFLOPS	GFLOPS/ Core	GFLOPS	GFLOPS / Core	GFLOPS	GFLOPS/Core
ARM Fujitsu A64FX	48	838.96	1978	41.2 ± 0.2	1177 ± 19	24.5	24.4 ± 0.9	0.51
Intel Skylake	40	149.51	1559	39.0 ± 8.1	981 ± 109	24.5	33.4 ± 2.4	0.84
AMD Milan	96	333.54	2775	28.9 ± 0.9	1493 ± 16	15.6	42.6 ± 1.0	0.44
Intel Sapphire Rapids	96	388.31	4787	49.9 ± 2.7	2211 ± 182	23.0	129.0 ±15.1	1.34
Intel Sapphire Rapids HBM	96	1360.54	5392	56.2 ± 4.2	2862 ± 36	29.8	143.1 ±24.4	1.49

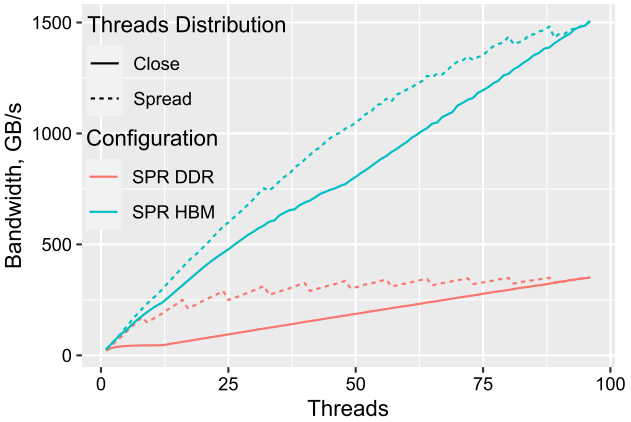


Fig. 1 Memory copy routines showing memory bandwidth patterns

Benchmarks

Memory bandwidth. STREAM TRIAD demonstrates the sustained HBM bandwidth of SPR being 1,360.5 GB/s, which is 3.5 times higher than DDR bandwidth on the same node (Table 2, STREAM Column). This is consistent with the results obtained by Wang et al. [4]. We also studied the bandwidth versus the number of threads and their distribution using the memory copy benchmark (Fig. 1). Impressively, increasing the number of threads up to full subscription on a node improves the memory bandwidth utilization almost linearly with HBM, while in DDR mode, the bandwidth saturates quickly. The cumulative single-node memory copy bandwidth on SPR was more than 4.2x higher for SPR-HBM than for SPR-DDR (Fig. 1). Similarly, using the `spread` thread distribution configuration consistently outperformed `close` thread placement except at the highest number of threads tested.

The **HPCC** benchmark utilizes BLAS and FFT libraries, often used in scientific applications. Matrix-matrix multiplication (DGEMM) is one of the few practical calculations capable of approaching theoretical FLOPS, in part due to memory-efficient algorithms, which significantly reduce the memory bandwidth requirements. As can be seen from Table 2 (matrix multiplication column), SPR demonstrates the highest per core and per node performance with HBM bringing an additional 13% improvement. In the LINPACK test, the memory bandwidth requirements are higher and HBM systems (A64FX and SPR-HBM) show better performance (Table 2, LINPACK column). SPR-DDR has a similar per-core performance to the Intel Skylake-X CPU, and HBM brings an additional 29% improvement making SPR-HBM the fastest per core and per node system. FFT exhibits even higher memory bandwidth requirements than LINPACK and DGEMM and systems with faster memory show better performance (Table 2, FFT column). SPR shows the fastest per core and per node results with HBM responsible for an 11% improvement.

Table 3 Performance in HPCG Benchmark

CPU/System	Cores	HPCG	
		GFLOPS	GFLOPS /Core
ARM Fujitsu A64FX	48	64.4 ± 2.8	1.34
Intel Skylake	40	36.4 ± 0.3	0.91
AMD Milan	96	53.0 ± 2.0	0.55
Intel Sapphire Rapids DDR	96	83.6 ± 1.1	0.87
Intel Sapphire Rapids HBM	96	197.5 ± 2.1	2.06

Table 4 benchDNN: input problems

Problem number	Problem
1	mb256ic64ih56oc256oh56kh1ph0n
2	mb256ic64ih56oc64oh56kh1ph0n
3	mb256ic128ih28oc128oh28kh3ph1n
4	mb256ic128ih28oc512oh28kh1ph0n
5	mb256ic256ih14oc256oh14kh3ph1n
6	mb256ic1024ih14oc256oh14kh1ph0n
7	mb256ic512ih7oc512oh7kh3ph1n
8	mb256ic2048ih7oc512oh7kh1ph0n
9	mb256ic256ih56oc128oh56kh1ph0n
10	mb256ic512ih14oc512oh7kh3sh2ph1n

The **HPCG** benchmark reflects the performance of many PDE solvers. SPR-DDR exhibits similar performance to previous generations. Turning on HBM brings an impressive 2.4x increase in performance (Table 3).

oneDNN is publicly available via GitHub, and we chose the latest release v3.2 for this work. To benchmark, we compare the performance of the convolution driver with `benchdnn`, a benchmarking harness provided by oneDNN. We selected 10 sample inputs as mentioned in Table 4 for the convolution driver. The naming convention follows a descriptor-size combination. Here mb stands for

mini batch size; ic and ih stand for input channel and height respectively; oc and oh stand for output channel and height respectively; kh, sh and ph stand for kernel, stride, and padding heights respectively. For example, mb256ic64ih56oc256oh56kh1ph0n will mean that we have a mini batch size of 256, 64 input channels, input height 56, output channel 256, output height 56, kernel height 1, and padding height 0. Next, we selected 3 input configurations to test, i.e., the input data type to the operator. We test with signed and unsigned 8-bit integers (important for inference), 32-bit floating point inputs, and lastly BrainFloat-16 (BF16) data types (both important for training as well as inference). The low precision data types are chosen since SPR has native support for them and they can leverage the TMUL accelerator via AMX instructions.

For the SPR chips, we utilize `numactl` to map memory regions explicitly to those nodes that have HBM and DDR corresponding to the CPUs used for running the benchmark. This helps us understand how HBM plays a role in the observed performance. Each associated data type has a CPU dispatch control (except for A64FX), which can control the ISA to be used. Here we select the default, which is the best available for the input configurations. This is described in Table 5.

We see the performance difference between the convolution operator with the same inputs on all 3 architectures that have a 512-bit vector length implementation. In Fig. 2, while running the samples in forward mode with 32-bit floating point data types typically used during training, we see that SPR-HBM is up to 1.7x and 3.5x faster than Skylake-X and A64FX. Improvements with respect to Skylake-X for this problem show the general benefits of running on newer Intel hardware as the same dispatch (AVX512_CORE) for jit'd kernels is used in both cases.

We also bring special focus to forward mode runs with the BrainFloat-16 data type commonly used on GPUs. Figure 3 shows the performance gains from using BF16 data type compared with other architectures. A64FX is omitted because the JIT kernels do not have a BF16 implementation.

Table 5 benchDNN: best CPU dispatch available

Architecture	Mode	Data type	Best CPU dispatch
A64FX	FWD_B	fp32	SVE-512
	FWD_B	bf16	–
	FWD_I	int8	SVE-512
Skylake-X	FWD_B	fp32	AVX512_CORE
	FWD_B	bf16	AVX512_CORE
	FWD_I	int8	AVX512_CORE
Sapphire Rapids	FWD_B	fp32	AVX512_CORE
	FWD_B	bf16	AVX512_CORE_AMX
	FWD_I	int8	AVX512_CORE_AMX

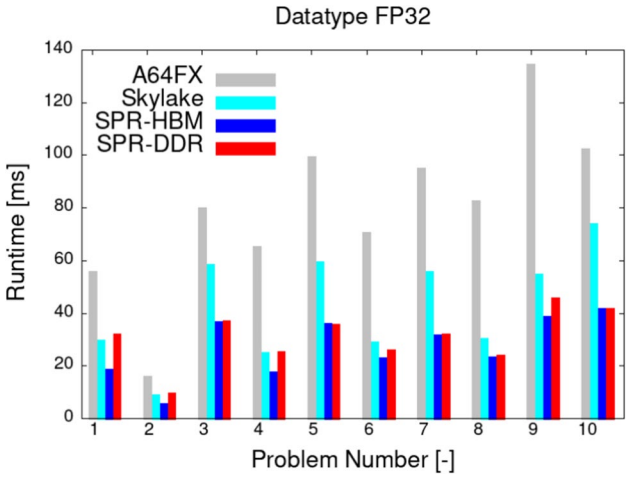


Fig. 2 FWD_B mode (forward mode with bias) used in training with 32-bit floating point input and output

Comparing to results obtained by SPR-HBM in fp32 mode, we see speedup ranging from 1.79x - 9.2x. This result is crucial as bf16 data type can be used for mix-precision training available in Deep Learning frameworks and can accelerate workloads on CPU architectures with native support. Skylake-X suffers from massive performance degradation even compared to fp32 mode because of the lack of intrinsic support for BF16 instructions. Additional details can be found in the Appendix.

Lastly, we show the performance of the same inputs in forward mode for inference with 8-bit integer data types in Fig. 4. We see speedups of up to 16.1x with the SPR processor over Skylake-X and up to 17.5x speedup over A64FX when using SPR-HBM. There are also general benefits of using HBM vs DDR on the SPR processor with speedups

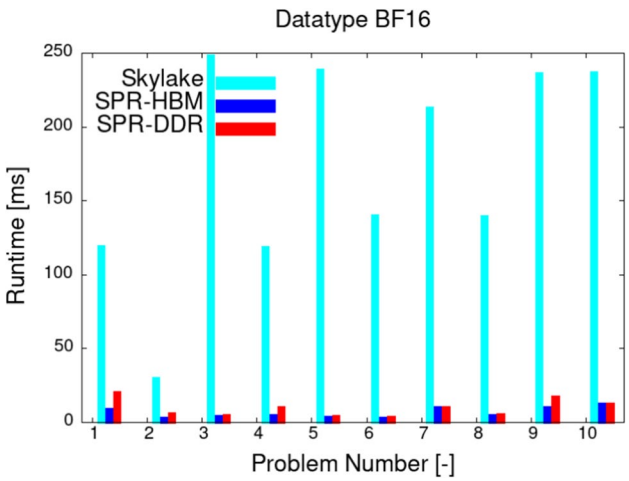


Fig. 3 FWD_B mode (forward mode with bias) with 16-bit BrainFloat data type for input and output

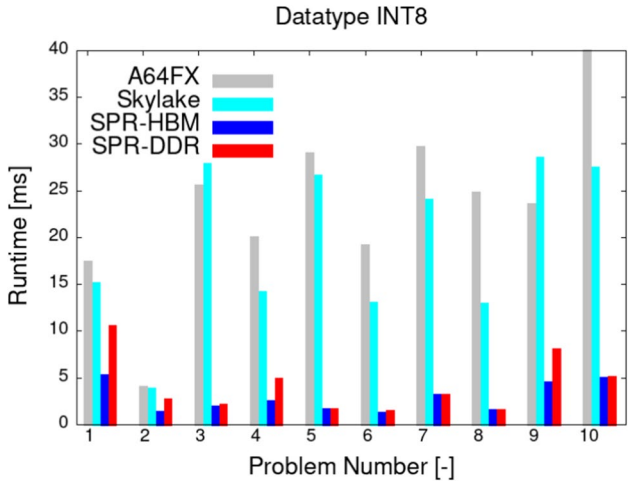


Fig. 4 FWD_I mode (forward pass in inference) with 8-bit integer data type input and output

of up to 2x. However, there are some cases across all figures where there was no speedup and insignificant loss in performance.

Applications

All our benchmarks show significant improvement over the previous CPU generation, with HBM bringing a significant performance boost for some applications. We therefore wanted to determine if improvements in core libraries and system capabilities will translate to real-world applications.

GROMACS and other Molecular Dynamics applications are responsible for a significant portion of HPC system utilization. SPR performs the fastest among pure CPU systems (Table 6). However, the use of HBM does not have a significant effect on the performance across the tested problem sizes. For small MEM systems (82k atoms), the SPR CPU is 2.5 times faster than the Milan CPU, but for larger systems (RIB, 2 M atoms, and PEP 12 M atoms), SPR is only 31–37% faster. We speculate that due to the smaller size of the MEM system, it fits mostly within CPU

caches, allowing more efficient utilization of AVX-512 instructions. This results in a single SPR node performance for MEM being only 15% slower than the NVIDIA A100 GPU performance. For larger problems, SPR is 40–50% slower than A100 (for a more extensive comparison to modern GPUs, see [7]). Another notable point is that the old Skylake CPU still has strong per-core performance, and multi-node execution can alleviate lower core per-node count.

OpenFOAM tests demonstrate an increased performance of SPR for larger problems compared to previous Intel and AMD CPUs (Table 7). For the smallest problem, which has 2 M cells, SPR performs similarly to Milan, with little effect from HBM. For the larger problems (11 M and 35 M cells), SPR is 25% faster than Milan. An additional 21–25% improvement can be achieved by using HBM. Interestingly, HBM has no significant effect on the meshing step. Most of the performance improvement comes from the solving part. This is most likely due to intensive memory allocation/deallocation during the meshing process where DDR and HBM perform similarly.

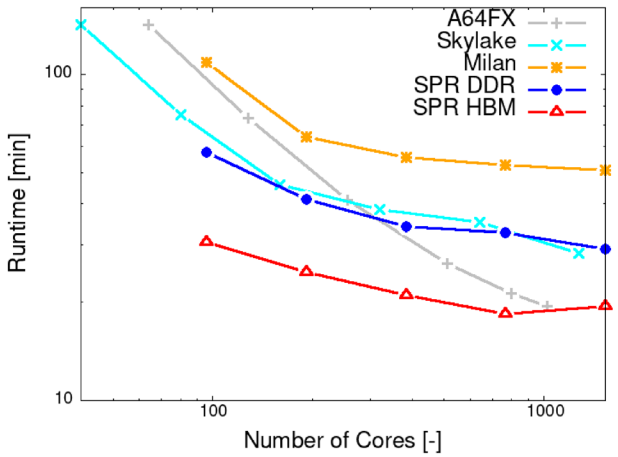
ROMS - The benchmark results in terms of runtime, scaling, and energy consumption are shown in Fig. 5. On SPR, HBM is nearly 2 times faster than using DDR (Fig. 5a). The performance of Skylake and SPR-DDR is very similar. A64FX shows poor performance when

Table 6 Performance for GROMACS. GPU performance is given for comparison, see [10] for more details. The GPU systems had four and two GPUs but only one was used. The power measurements also include idle GPUs

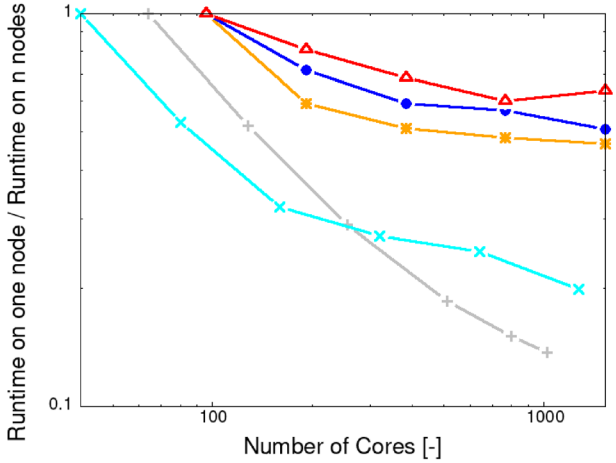
		Simulation Speed, ns/day	Energy Efficiency, ns/kWh	Power, W
MEM, 82K Atoms				
ARM Fujitsu A64FX	48	22.8 ± 0.3 (10)	9.1 ± 0.4 (10)	105 ± 5 (10)
Intel Skylake	40	51.4 ± 1.2 (10)	8.8 ± 0.4 (9)	245 ± 9 (9)
AMD Milan	96	81.3 ± 11.2 (10)		
Intel Sapphire Rapids DDR	96	203.6 ± 4.8 (22)	9.6 ± 0.4 (11)	853 ± 35 (11)
Intel Sapphire Rapids HBM	96	206.1 ± 5.2 (10)	9.5 ± 0.4 (10)	859 ± 32 (10)
Intel IceLake/NVIDIA A100	64	242.6		
Intel IceLake/NVIDIA A100	54	236.5 ± 10.8 (11)	13.9 ± 0.8 (11)	707 ± 9 (11)
RIB, 2M Atoms				
Intel Skylake	40	4.8 ± 0.01 (8)	0.86 ± 0.02 (7)	230 ± 5 (7)
AMD Milan	96	10.1 ± 0.21 (20)		
Intel Sapphire Rapids DDR	96	13.88 ± 0.05 (10)	0.58 ± 0.01 (10)	997 ± 17 (10)
Intel Sapphire Rapids HBM	96	14.49 ± 0.05 (10)	0.62 ± 0.01 (10)	972 ± 8 (10)
Intel IceLake/NVIDIA A100	64	21.41		
PEP, 12M Atoms				
AMD Milan	96	0.9 ± 0.01 (20)		
Intel Sapphire Rapids DDR	96	1.18 ± 0.01 (12)	0.049 ± 0.002 (12)	1008 ± 39 (12)
Intel Sapphire Rapids HBM	96	1.2 ± 0.02 (10)	0.053 ± 8e-04 (10)	953 ± 7 (10)
Intel IceLake/NVIDIA A100	64	2.42		

Table 7 Performance in OpenFOAM application

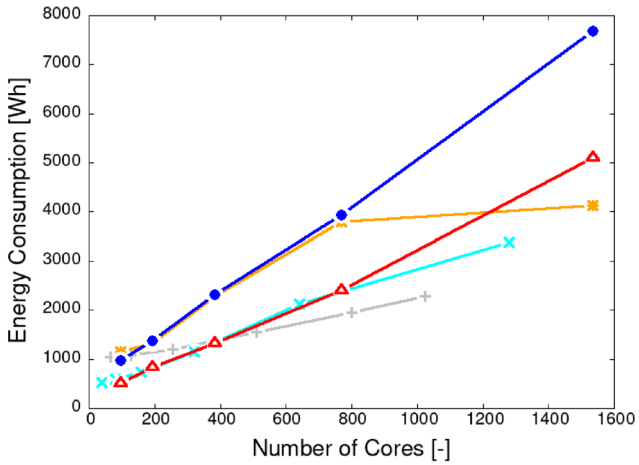
CPU/System	Cores	Run time, seconds, (smaller better)			Power, W	Energy, Wh
		Total	Meshing	Solving		
motorBikeD, 1.9M Cells						
Intel Skylake	40	386 ± 3 (4)	118 ± 3 (4)	248 ± 4 (4)	343 ± 20 (4)	40.4 ± 1.7 (4)
AMD Milan	96	234 ± 33 (4)	111 ± 21 (4)	83 ± 6 (4)		
Intel Sapphire Rapids DDR	96	254 ± 19 (10)	130 ± 12 (10)	83 ± 5 (10)	867 ± 42 (2)	78.7 ± 5.4 (2)
Intel Sapphire Rapids HBM	96	244 ± 14 (10)	129 ± 10 (10)	72 ± 4 (10)		
motorBikeQ, 11M Cells						
Intel Skylake	40	2367 ± 94 (5)	688 ± 88 (5)	1616 ± 21 (5)	391 ± 5 (5)	260.7 ± 14.3 (5)
AMD Milan	96	1406 ± 35 (4)	429 ± 54 (4)	898 ± 50 (4)	620 ± 26 (4)	245.8 ± 8.8 (4)
Intel Sapphire Rapids DDR	96	1071 ± 67 (6)	407 ± 51 (6)	588 ± 22 (6)	976 ± 22 (6)	293.2 ± 20.5 (6)
Intel Sapphire Rapids HBM	96	846 ± 40 (6)	379 ± 37 (6)	386 ± 12 (6)	962 ± 21 (6)	228.7 ± 11.1 (6)
motorBikeX6, 35M Cells						
Intel Sapphire Rapids DDR	96	3207 ± 119 (5)	1000 ± 119 (5)	2064 ± 16 (5)	1003 ± 15 (5)	897.5 ± 45.6 (5)
Intel Sapphire Rapids HBM	96	2379 ± 91 (5)	1019 ± 86 (5)	1211 ± 24 (5)	974 ± 13 (5)	646.7 ± 30.8 (5)



(a) Runtime vs number of cores.



(b) Runtime ratio (one node / n nodes)



(c) Energy consumption per simulation vs number of cores.

Fig. 5 Results for running ROMS on different CPUs

using low core counts, but outpaces the other CPUs when increasing the number of used cores. On the A64FX nodes, the code scales well, whereas this is not the case for other processors. Especially on SPR, both HBM and DDR, and Milan, the scaling is poor. The scaling was evaluated as the ratio of runtime on a single node versus on multiple nodes (Fig. 5b). The energy consumption is depicted in Fig. 5c. SPR, having a peak power consumption of more than 1000W, exhibits a substantial energy consumption, especially for higher core counts. A64FX, with a peak power consumption of around 120W, on the other hand, is very energy-efficient. The energy efficiency of the A64FX has also been indicated by the leadership of the Fugaku supercomputer in the Green500 benchmark in 2019. Summarized, ROMS showcases both positive and negative aspects: the favorable performance gain from HBM and, on the other hand, the hard fact that poor multi-node scaling inevitably leads to poor energy efficiency.

Investigating the Effect of BIOS Changes

This section focuses on the effect of changes in the system's BIOS on performance and energy consumption. We investigated two workload profiles: the default HPC profile and our custom profiles (Table 8). The default HPC profile prioritizes the performance and disables the CPU's C-states. The latter disables the ability of the CPU to go into a deeper idle state. To figure out the energy savings and effects on performance, we tested our custom profile setting where we use balanced performance and allow up to C6 idle state. This allows higher energy savings when the node is unused. Our measurements show that the custom profile reduces the idle power consumption from ~ 450 Watt to ~ 310 Watts. The BIOS settings were tested on ROMS, GROMACS, and OpenFOAM.

Four nodes were configured with each workload profile, and the ROMS application was run 5 times for each configuration on 1, 2, and 4 nodes. The results of runtime and energy consumption are shown in Fig. 6.

Table 8 Differences in the HPC and the custom profile

Setting	HPC Profile	Custom Profile
Power Regulator	Static high performance	Dynamic power savings
Min. Processor Idle Power Core C-state	No C-states	C6
Min. Processor idle power package C-state	No C-states	Package C6 Retention
Energy Performance Bias	Max Performance	Balanced Performance
UPI Link Power Management	Disabled	Enabled
Uncore Frequency Shifting	Maximum	Auto
Collaborative Power Control	Disabled	Enabled
Energy-Efficient Turbo	Disabled	Enabled
Optimized Power Mode [Hidden Setting]	Disabled	Enabled

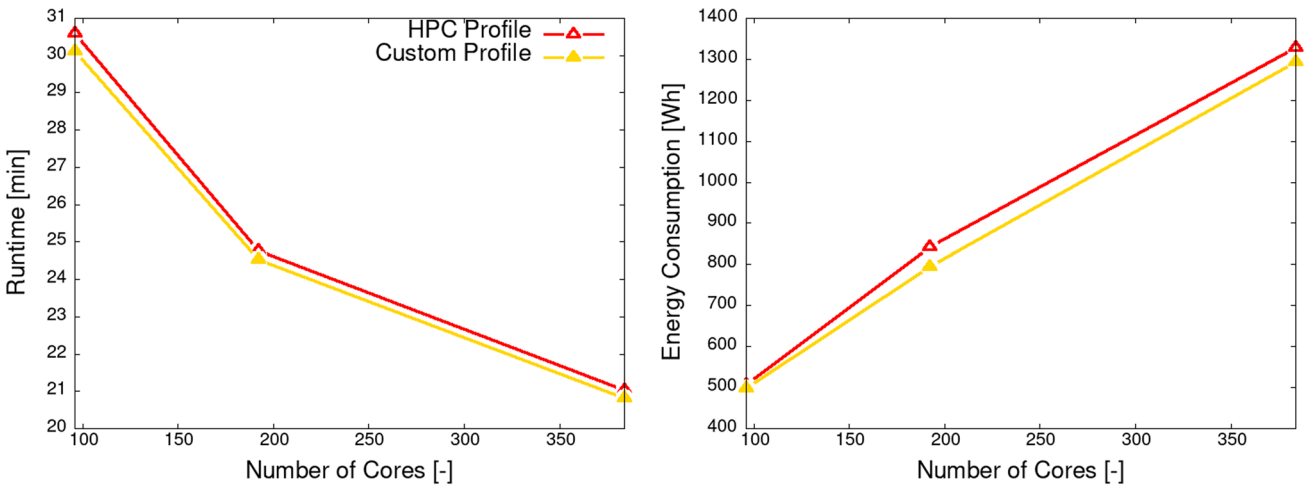


Fig. 6 Results of ROMS with different BIOS configurations. Left: Runtime in minutes vs number of cores, Right: energy consumption per simulation vs number of cores

Interestingly the runtime as well as the energy consumption show a slight decrease under the custom profile, indicating that those settings have a measurable impact. The peak power consumption decreased from 1150 to 1034 W when using the custom BIOS profile.

GROMACS and OpenFOAM show a performance degradation of 3% and 6%, respectively, when being run with the Custom BIOS. Still, for underutilized machines, the energy savings might outweigh performance degradation. Future work will investigate this in more detail.

Discussion and Conclusion

In this work, we evaluated the new Intel SPR CPU with optional HBM memory and AMX instructions. HBM holds the promise of enhancing the performance of memory-bound applications. Our results with multiple benchmarks demonstrate promising performance improvements with the new SPR (DDR mode). This is evident in the performance improvement for ROMS, GROMACS, and OpenFOAM.

HBM brings a further and often substantial improvement in the benchmarks (2.4 times in HPCG) and real applications (almost doubling for ROMS and 21–25% for OpenFOAM). However, some applications like GROMACS do not benefit from the HBM featured in this processor.

The AMX extension to the Intel AVX512 ISA shows significant speedups in our tests, positioning the SPR CPU as a potential option for mix-precision deep-learning training and inference workloads with appropriate data types.

In centers with diverse workloads, HBM-enabled SPR nodes can offer significant performance enhancements for specific applications such as ROMS. Adjusting the BIOS profile can assist in maintaining low power consumption without compromising performance. Future work will involve further BIOS investigations with additional use cases. Furthermore, we plan to extend our analysis of SPR to encompass additional scientific applications and larger, computationally more demanding test cases.

Appendix

See Tables 9, 10.

Performance Degradation of BF16 kernels on Skylake-X Processor

As seen in Fig. 3, Skylake-X suffers from poor runtimes when the input and output data type is set to BF16 because this architecture does not possess intrinsic support for BF16 instructions and uses what is available i.e., AVX512_CORE instructions that are aimed for single and double precision. We verified this behaviour by controlling the environment variable `ONEDNN_MAX_CPU_ISA` via values (`AVX512_CORE, AVX512_CORE_BF16`) on Skylake-X and SPR and observing the JIT dump. Skylake-X cannot

Table 9 Compilers, MPIs, and flags used for all presented benchmarks and applications

Processor	Compiler	MPI	CPU-specific flags
Intel® Xeon® CPU Max 9468	GCC12.1.0 Intel2023.2	OpenMPI4.1.5 IntelMPI2021.9	-march=sapphirerapids -mtune=sapphirerapids -xsapphirerapids
Fujitsu A64FX-FX700	Fujitsu4.8	FujitsuMPI	-KSVE
AMD EPYC 7643	aocc4.0	OpenMPI4.1.5	-march=native
Intel® Xeon® Gold 6148	GCC12.1.0 Intel2023.2	OpenMPI4.1.5 IntelMPI2021.9	-march=skylake -mtune=skylake -skylake
Amazon ARM Graviton3	GCC11.3.0	OpenMPI4.1.4	spack defaults
Nvidia A100	CUDA 12.0	N/A	app defaults

Table 10 Additional flags used for the STREAM benchmark

Processor	Compilation flags
Intel® Xeon® CPU Max 9468	-O3 -xsapphirerapids -ffreestanding -fiopenmp -mcmodel=large -shared-intel -qopt-streaming-stores=always -fno-builtin -qopt-zmm-usage=high
Fujitsu A64FX-FX700	-Kfast,preex -Kopenmp -Kzfill -Kstriping=4
AMD EPYC 7643	-O3 -ffreestanding -march=native -fopenmp -mcmodel=large -fno-builtin -ffp-contract=fast -fnt-store
Intel® Xeon® Gold 6148	-O3 -xCORE-AVX512 -ffreestanding -fiopenmp -mcmodel=large -shared-intel -qopt-streaming-stores=always -fno-builtin -qopt-zmm-usage=high

take advantage of efficient instructions included in the BF16 extensions to AVX512 ISA, unlike SPR, and therefore utilizes AVX512_CORE single precision instructions to "emulate" BF16 execution. BF16 extension instructions include VCVTNE2PS2BF16, VCVTNEPS2BF16 and VDPBF16PS. The first two instructions deal with converting SIMD registers with single precision values to BF16 and the latter deals with performing SIMD dot-product on BF16 pairs. These additional instructions are valuable since they reduce the zmm register usage up to ten-fold for some input problems on SPR.

Acknowledgements The authors would like to thank Stony Brook Research Computing and Cyberinfrastructure, and the Institute for Advanced Computational Science at Stony Brook University for access to the innovative high-performance Ookami computing system, which was made possible by a $5 M National Science Foundation grant (#1927880), and for access to the high-performance SeaWulf computing system, which was made possible by National Science Foundation grants (#1531492 and #2215987).

Data availability The data supporting the findings of this study are available upon reasonable request.

Declarations

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Nassif et al. Sapphire Rapids: The Next-Generation Intel Xeon Scalable Processor. In 2022 IEEE International Solid- State Circuits Conference (ISSCC), 2022;65:44–46. https://doi.org/10.1109/ISSCC42614.2022.9731107.
2. Shipman Galen M, Swaminarayan Sriram, Grider Gary, Lujan Jim, Zerr R Joseph. Early performance results on 4th Gen Intel ® Xeon ® Scalable Processors with DDR and Intel ® Xeon ® processors, codenamed Sapphire Rapids with HBM. arXiv preprint arXiv:2211.05712, 2022.
3. McCalpin John D. Bandwidth limits in the Intel Xeon Max (Sapphire Rapids with HBM) Processors. In High Performance Computing, 2023:403–413. Cham. Springer Nature Switzerland. ISBN 978-3-031-40843-4https://doi.org/10.1007/978-3-031-40843-4_30.
4. Wang Yinzhi, McCalpin John D, Li Junjie, Cawood Matthew, Cazes John, Chen Hanning, Koesterke Lars, Liu Hang, Lu Chun-Yaung, McLay Robert, Milfield Kent, Ruhela Amit, Semeraro Dave, Zhang Wenyang. Application performance analysis: A report on the impact of memory bandwidth. In High Performance Computing, 2023:339–352, Cham. Springer Nature Switzerland. ISBN 978-3-031-40843-https://doi.org/10.1007/978-3-031-40843-4_25.
5. Cuma Martin. AMD Genoa and Intel Sapphire Rapids review. 2023. www.chpc.utah.edu/documentation/white_papers/cpus_may2023_v3.pdf.
6. Afzal Ayesha, Hager Georg, and Wellein Gerhard. SPEChpc 2021 benchmarks on Ice Lake and Sapphire Rapids Infiniband clusters: A performance and energy case study. In Proceedings of the SC '23 Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis, SC-W '23, 2023:1245-1254, New York, NY, USA. Association for Computing Machinery. ISBN 979840070785https://doi.org/10.1145/3624062.3624197.
7. Nikolay A. Simakov, Matthew D. Jones, Thomas R. Furlani, Eva Siegmann, and Robert J. Harrison. First impressions of the NVIDIA Grace CPU Superchip and NVIDIA Grace Hopper Superchip for scientific workloads. HPCAsia '24 Workshops, page 36–44, New York, NY, USA. Association for Computing Machinery. ISBN. 2024;9798400716522. https://doi.org/10.1145/3636480.3637097.
8. Gao Xiang, Lin Xiancheng, and Liu Rongkai. Comparable GPU: Optimizing the BERT model with AMX feature. In 2023 IEEE 3rd International Conference on Computer Communication and Artificial Intelligence (CCAI), 2023:158–162. https://doi.org/10.1109/CCAI57533.2023.10201262.
9. GitHub - ipmitool/ipmitool: An open-source tool for controlling IPMI-enabled systems. https://github.com/ipmitool/ipmitool.
10. Simakov Nikolay A, Deleon Robert L, White Joseph P, Jones Matthew D, Furlani Thomas R, Siegmann Eva, and Harrison Robert J. Are we ready for broader adoption of ARM in the HPC community: Performance and energy efficiency analysis of benchmarks and applications executed on high-end ARM systems. In Proceedings of the HPC Asia 2023 Workshops, HPC Asia '23 Workshops, 2023:78-86, New York, NY, USA. Association for Computing Machinery. ISBN 9781450399890https://doi.org/10.1145/3581576.3581618.
11. Luszczek P, and Dongarra J. Introduction to the HPC Challenge benchmark suite, ICL technical report ICL-UT-05-01, University of Tennessee - Knoxville, 2005. https://icl.utk.edu/files/publications/2005/icl-utk-223-2005.pdf.
12. Dongarra Jack, Heroux Michael A, and Luszczek Piotr. High-performance conjugate-gradient benchmark: A new metric for ranking high-performance computing systems. The International Journal of High Performance Computing Applications, 2016;30(1):3–10. https://doi.org/10.1177/1094342015593158.
13. Páll S, Zhmurov A, Bauer P, Abraham M, Lundborg M, Gray A, Hess B, Lindahl E. Heterogeneous parallelization and acceleration of molecular dynamics simulations in GROMACS. J Chem Phys. 2020;153(13): 134110. https://doi.org/10.1063/5.0018516.
14. Kutzner C, Páll S, Fechner M, Esztermann A, de Groot BL, Grubmüller H. Best bang for your buck: GPU nodes for GROMACS biomolecular simulations. J Comput Chem. 2015;36(26):1990–2008. https://doi.org/10.1002/jcc.24030.
15. Jasak Hrvoje, Jemcov Aleksandar, and Tukovic Zeljko. OpenFOAM: A C++ library for complex physics simulations. In International workshop on coupled methods in numerical dynamics, volume 1000, 2007:1–20. https://api.semanticscholar.org/CorpusID:35226827.
16. Shchepetkin Alexander F, and McWilliams James C. A method for computing horizontal pressure-gradient force in an oceanic model with a nonaligned vertical coordinate. J. Geophys. Res., 2003;108(C3). https://doi.org/10.1029/2001JC001047.

17. Shchepetkin AF, McWilliams JC. The regional ocean modeling system: A split-explicit, free-surface, topography following coordinates ocean model. Ocean Model. 2005;9:347–404.
18. Herman Rachael, Borowicz Alex, Lynch Maureen, Trathan Tom, Hart, and Lynch Heather. Update on the global abundance and distribution of breeding Gentoo Penguins (Pygoscelis papua). Polar Biology, 2020;43(12):1947-1956. https://doi.org/10.1007/s00300-020-02759-3.
19. Simakov Nikolay A, White Joseph P, DeLeon Robert L, Ghadersohi Amin, Furlani Thomas R, Jones Matthew D, Gallo Steven M, Patra Abani K. Application kernels: HPC resources performance monitoring and variance analysis. Concurrency and Computation: Practice and Experience, 2015;27(17):5238–5260.https://doi.org/10.1002/cpe.3564.