Performance and Efficiency: a Multi-Generational Benchmark of Modern Processors on Bandwidth-Bound HPC Applications

Balázs Drávai[1], István Z Reguly[1]

[a]*Pázmány Péter Catholic University, Faculty of Information Technology and Bionics, Práter utca 50/a, Budapest, 1083, Hungary*

Abstract

The last two years has seen the launch of a multitude of new x86 processors, in reaction to market demand. Intel has launched four families of Xeon Processors, with some novel architectural features; first the Sapphire Rapids generation which featured a version with on-package HBM, the Emerald Rapids generation, and then differentiated by releasing the performance-oriented Granite Rapids and the efficiency-oriented Sierra Forest families. In this work, we evaluate the performance and energy efficiency of CPUs from each of different generations and variants of Intel and AMD CPUs, with a particular focus on bandwidth-bound high performance computing (HPC) applications. We contrast runtime and energy consumption figures and track trends across generations. We furthermore study how enabling locality-improving optimizations increases cache reuse and overall performance, while reducing energy use.

Keywords: Benchmarking, Xeon, Performance, Energy, Cache, CFD

1. Introduction

In recent years, several generations of x86 processors were released by AMD and Intel. Intel released three generations of Xeon processors, each advancing performance and energy efficiency. AMD released two, with architectural innovations such as 3D V-Cache. In this paper we examine the performance and energy efficiency evolution of these processors. On Intel's side there was a move from the Intel 7 to the Intel 3 process, and the introduction of chiplet designs or the integration of High Bandwidth Memory (HBM). For Intel processors, we focus on Sapphire Rapids (SPR) with up to 4 compute tiles, Emerald Rapids (EMR) with up to 2 compute tiles, Granite Rapids (GNR) with up to 3 tiles, and Sierra Forest (SRF) with up to 2 compute tiles. These families are differentiated based on their core counts, cache sizes, and energy efficiency, and with the latest generation, the separation of performance-oriented (P) and efficiency-oriented (E) families. For AMD processors, we study the Milan generation, already using a chiplet architecture, as well as the Genoa family of processors with up to 12 chiplets.

Memory bandwidth is a key limiter for many applications - newer generations support higher speed DDR modules, with Milan still on DDR4-3200 MHz, and newer systems going from DDR5-4800 MHz in SPR, up to 8800 MHz with GNR using the new Multiplexed Rank DIMMs (MRDIMM), and increasing from 8 memory channels to 12 with GNR and SRF. At the same time the TDP for the highest core count models were at 350 W for SPR and EMR, and this was increased to 500 W with GNR.

Despite the advancements in GPU acceleration, many legacy codes remain CPU-only (such as nuclear security), making the evaluation of newer generation CPUs crucial. The absence of plans for further HBM-equipped CPUs raises questions about the performance of DDR-based solutions for these workloads. Our study provides an evaluation of the performance and energy efficiency of recent Intel Xeon and AMD EPYC processor generations. By examining a set of primarily bandwidth-bound applications, we shed light on the implications of these architectural innovations and inform the design of computing systems.

In this paper, we make the following contributions:

1. Establish performance baselines for the 5 Intel and 2 AMD systems to be used for architectural efficiency calculations.
2. Benchmark a variety of structured- and unstructured-mesh codes and contrast their performance in absolute terms (runtime) as well as relative (achieved architectural efficiency).
3. Measure the energy consumption of these applications running on the 6 bare-metal Intel systems, and study how energy efficiency improves across generations. For four, we report the energy consumption of the DRAM subsystem as well.
4. Enabling a cache-blocking and communication-avoiding optimization for a subset of our structured-mesh applications, we study the achieved improvement in runtime and energy and its relationship to the different CPU configurations.

2. Systems and Baseline Performance

For this study, we evaluate the following systems - with Intel machines available in the Intel Developer Cloud.

1. Intel Xeon CPU MAX 9480 Processor with HBM (Sapphire Rapids, codenamed SPR+HBM), available in the

Intel Developer Cloud. Two sockets, each with 56 cores, Hyperthreading on. 2x4 NUMA regions, with 2x64 GB HBM in HBM-only mode, with SNC4. Clock frequencies between 1.9 GHz (base frequency) - 2.6 GHz (all-core turbo), giving a theoretical 13.6-18.6 FP32 TFLOPS/s. Software: Intel OneAPI Base and HPC toolkits, 2024.1 (including Intel MPI). Benchmarked June 5, 2024.

2. Intel Xeon Platinum 8480+ Processor (Sapphire Rapids, codenamed SPR+DDR), available in the Intel Developer Cloud. Two sockets, each with 56 cores, Hyperthreading on. 16x32 GB DDR5-4800MHz RAM. Clock frequencies between 2.0 GHz (base frequency) - 3.0 GHz (all-core turbo), giving a theoretical 14.3-21.5 FP32 TFLOPS/s. Software: Ubuntu 22.04, Intel OneAPI Base and HPC toolkits, 2024.1 (including Intel MPI). Benchmarked June 13, 2024.
3. Intel Xeon Platinum 8592+ Processor (Emerald Rapids, codenamed EMR), available in the Intel Developer Cloud. Two sockets, each with 64 cores, Hyperthreading on. 16x64 GB DDR5-5600MHz RAM. Clock frequencies between 1.9 GHz (base frequency) - 2.9 GHz (all-core turbo), giving a theoretical 15.5-23.7 FP32 TFLOPS/s. Software: Ubuntu 22.04, Intel OneAPI Base and HPC toolkits, 2024.1 (including Intel MPI). Benchmarked June 15, 2024.
4. Intel Xeon Platinum 6960P Processor (Granite Rapids, codenamed GNR), available in the Intel Developer Cloud. Two sockets, each with 3 NUMA nodes and 72 cores total, Hyperthreading on. 24x32 GB DDR5-8800MHz (MRDIMM) RAM. Clock frequencies between 2.7 GHz (base frequency) - 3.8 GHz (all-core turbo), giving a theoretical 24.8-35 FP32 TFLOPS/s. Software: Centos 9 Stream, Intel OneAPI Base and HPC toolkits, 2024.1 (including Intel MPI). Benchmarked June 2, 2024.
5. Intel Xeon 6740E Processor (Sierra Forest, codenamed SRF 96c), available in the Intel Developer Cloud. Two sockets, each with 96 cores, no Hyperthreading. 16x64 GB DDR5-6400MHz RAM. Clock frequencies between 2.4 GHz (base frequency) - 3.2 GHz (all-core turbo), giving a theoretical 3.6-4.9 FP32 TFLOPS/s. Software: Centos 9 Stream, Intel OneAPI Base and HPC toolkits, 2024.1 (including Intel MPI). Benchmarked June 29, 2024.
6. Intel Xeon (Unnamed) Processor (Sierra Forest, codenamed SRF 192c), available in the Intel Developer Cloud. Two sockets, each with 192 cores, no Hyperthreading. 24x64 GB DDR5-6400MHz RAM. Clock frequencies between 2.4 GHz (base frequency) - 3.2 GHz (all-core turbo), giving a theoretical 7.3-9.8 FP32 TFLOPS/s. Software: Centos 9 Stream, Intel OneAPI Base and HPC toolkits, 2024.1 (including Intel MPI). Benchmarked July 18, 2024.
7. AMD EPYC 7763 (codenamed Milan), available in the Komondor supercomputer. Two sockets, each with 64 cores available, Hyperthreading off. 2x4 NUMA regions, with 16x16 GB DDR4 RAM. Clock frequencies between 2.45 GHz (base frequency) - $\sim$ 3.0 GHz, giving a theoretical 8.75-10.75 FP32 TFLOPS/s. Software: Cray 16.0.1. Benchmarked Oct 12, 2024.
8. AMD EPYC 9B14 (codenamed Genoa), available as a C3D virtual machine in Google Cloud. Two sockets, each with 90 available cores, Hyperthreading off. 2x2 NUMA regions, with 24x64 GB DDR5 RAM, of which 1440 is available. Clock frequencies between 2.6 GHz (base frequency) - $\sim$ 3.6 GHz (turbo), giving a theoretical 19.9-27.5 FP32 TFLOPS/s. Software: Ubuntu 22.04, GCC 11.3. Benchmarked June 6, 2023.

Note that RAPL counters were only available on the Intel systems and the earlier generation AMD Milan system. Cloud VMs do not expose these counters due to associated security vulnerabilities.

Table 1 shows detailed architectural information on the studied platforms. Most relevant to the present study is the achieved bandwidth as measured by BabelStream Triad [1] with OpenMP, running on one or both sockets of each tested platform, noting the highest achieved bandwidth in Last Level Cache (LLC) and main memory (DDR or HBM), as well as the speedup of main memory bandwidth in comparison to SPR DDR. Subsequent generations of Xeon chips have larger Last Level Caches (LLC): 105 MB per socket for SPR, 320 MB for EMR, and 432 for GNR, but only 96 MB for SRF (96 core variant, 192 MB for the 192 core variant) - the tested Milan CPU had 256 MB, and the Genoa CPU had 384 MB LLC per socket. Relative differences between maximum cache bandwidth and maximum DRAM bandwidth are also noteworthy: SPR, EMR and SRF (96 core) support 8 channels of DRAM per socket, and have a ratio of 11-15×, GNR, SRF 192 core, and Genoa have 12 channels per socket, yet GNR only has an 6.4× difference, and Genoa has a 18× difference. We also see a 2.96× increase in main memory bandwidth in the span of 1.5 years, going from SPR to GNR. SPR+HBM (the Intel Xeon MAX CPU) has a much higher bandwidth, but limited size HBM memory (64 GB per socket), and only a 2.4× difference between LLC and HBM bandwidth

The Intel machines were bare metal systems accessed in the Intel Developer Cloud, the Milan machine was in the Komondor supercomputer, whereas the Genoa machine is a VM in Google Cloud. While VMs are known to have inefficiencies over bare-metal servers [2], these tend to be under 10% for most applications comparable to our benchmarks (bandwidth and compute intensive).

3. Applications and Parallelizations

We consider a number of primarily bandwidth-bound codes, which are implemented on top of two domain specific languages: OPS [3] (for structured meshes) and OP2 (for unstructured meshes) [4]. Prior work has shown that these implementations match or outperform the original hand-written implementations, and therefore the use of a DSL does not impede performance [5, 6, 7, 8], but it does enable automatic parallelization with MPI as well as a variety of shared-memory parallel programming models.

Specifically we evaluate the following structured-mesh applications:

	Compute					Memory				Cache		Misc		
Processor	Cores	Threads	BF[1] (GHz)	ATF[2] (GHz)	TFLOPS/s	Size (GB)	Transfer Rate (MT/s)	Type	BW[3] (GB/s)	LLC[4] (MB)	BW[3] (GB/s)	TDP[5] (W)	Process	Release
Intel Xeon CPU MAX 9480 (SPR+HBM)	2x56	2x112	1.9	2.6	13.6-18.6	2x64	3200	HBM2E	1475	112	3481	350	Intel 7	Q1 2023
Intel Xeon Platinum 8480+ (SPR+DDR)	2x56	2x112	2.0	3.0	14.3-21.5	16x32	4800	DDR5	388	112	4340	350	Intel 7	Q1 2023
Intel Xeon Platinum 8592+ (EMR)	2x64	2x128	1.9	2.9	15.5-23.7	16x64	5600	DDR5	542	120	8149	350	Intel 7	Q4 2024
Intel Xeon Platinum 6960P (GNR)	2x72	2x144	2.7	3.8	24.8-35	24x32	8800	DDR5	900	144	7346	500	Intel 3	Q3 2024
Intel Xeon 6740E (SRF 96c)	2x96	2x96	2.4	3.2	7.3-9.8	16x64	4800	DDR5	396	96	4139	250	Intel 3	Q2 2024
Intel Xeon (Unnamed) (SRF 192c)	2x192	2x192	2.4	3.2	14.7-19.6	24x64	6400	DDR5	667	192	7627	350	Intel 3	Pending
AMD EPYC 9B14 (Genoa)	2x90	2x90	2.6	~ 3.6	22.4-31.1	24x64	4800	DDR5	529	768	9534	360	TSMC 5	Q4 2022
AMD EPYC 7763 (Milan)	2x64	2x64	2.45	~ 3.6	7.5-11.1	16x16	3200	DDR4	235	256	4808	280	TSMC 7	Q1 2021

Table 1: Overview of processor specifications highlighting compute power, memory bandwidth, cache capacity, and power efficiency for selected server processors. Notes: [1] Base Frequency, [2] All-Core Turbo Frequency, [3] Bandwidth, [4] Last Level Cache, [5] Thermal Design Power.

1. CloverLeaf 2D/3D [9] – developed by the UK mini-app consortium, CloverLeaf is a proxy for nuclear security codes, simulating shockwaves using a structured-mesh Eulerian hydrodynamics setup. The code is largely bandwidth-bound, with some operations on faces/edges that may be latency bound. Double precision, 7680^2 (2D), 408^3 (3D) problem size, 50 iterations.
2. OpenSBLI SA & SN [6] – a shock-capturing Navier-Stokes solver developed at the University of Southampton for studying aeroacoustic phenomena around aircraft. The code has 2 variants – Store All (SA), which is bandwidth-bound, and Store None (SN), which recomputes derivatives on the fly, thereby trading compute for data movement, but is still mostly bandwidth bound. Double precision, 320^3 problem size, 20 time iterations.
3. RTM - computational geophysics proxy code, implementing the forward pass of a Reverse Time Migration algorithm. Uses an 8th order finite difference stencil. Due to the large stencils, it is sensitive to cache locality and vectorization, has large communications volume over MPI. Single precision, 320^3 problem size, 10 time iterations.
4. Acoustic – a proxy extracted from Devito [10], it is a structured-mesh high-order (8th) finite difference acoustic wave propagation solver. Bandwidth and cache locality bound, with large communications volume over MPI. Single precision, 1000^3 problem size, 30 time iterations.
5. MG-CFD [11] – a proxy for the Rolls-Royce CFD simulation code, implementing an unstructured mesh finite volume Euler equations solver with multi-grid. Bound by latencies and indirect memory accesses. Double precision, NASA Rotor37 case with 8 million vertices, 25 iterations.
6. Volna [12] – unstructured mesh finite volume Nonlinear Shallow Water Equations solver. Also sensitive to indirect memory accesses as MG-CFD, but less so. Indian ocean case with 30 million vertices, 200 time iterations.
7. miniBUDE [13] – proxy molecular docking code representative of the University of Bristol's BUDE. Compute and latency bound. bm1 test case, 30 iterations.

In this work, we evaluate pure MPI parallelizations (both 1 process per physical core and 1 process per SMT logical core, where HyperThreading was enabled), as well as a hybrid MPI+OpenMP parallelization, where we use one process per NUMA region. For all computational loops in structured-mesh applications OpenMP's `collapse` is used for the whole loop nest, with the innermost loop instructed to vectorize using `omp simd`. For unstructured mesh codes we study auto-vectorizing implementations that explicitly pack and unpack vector registers, with the vector length adjusted for the platform (256 bit for SRF, 512 for the rest). In prior work, we studied SYCL parallelization on CPUs [14], and showed that it delivers lower performance compared to MPI/MPI+OpenMP - since this has not improved significantly since, we do not report performance with MPI+SYCL here.

4. Results

We perform four runs of each benchmark configuration and average the results - each run reports both total wall time (excluding initial setup) as well as detailed per-subroutine timings with achieved effective bandwidth. To initialize data and TLB caches, the first time iteration in the applications is excluded from the measurements. Bandwidth is calculated as the sum of array sizes accessed by a given sweep across the grid (multiplied by 2 if read and written), divided by execution time, and it includes MPI communications. When discussing a fraction of peak bandwidth achieved, we compare this "effective bandwidth" value to the achieved peak DDR (or HBM) bandwidth, as reported in Table 1. The variation in measured results were consistently below 5%. We benchmarked performance with a pure MPI configuration, as well as a hybrid MPI+OpenMP configuration (1 process per NUMA domain), which we compare on Figure 2, but in other figures we show the best.

Figure 1 shows the wall times for different applications normalized to SPR DDR, Figure 2 shows the performance of MPI

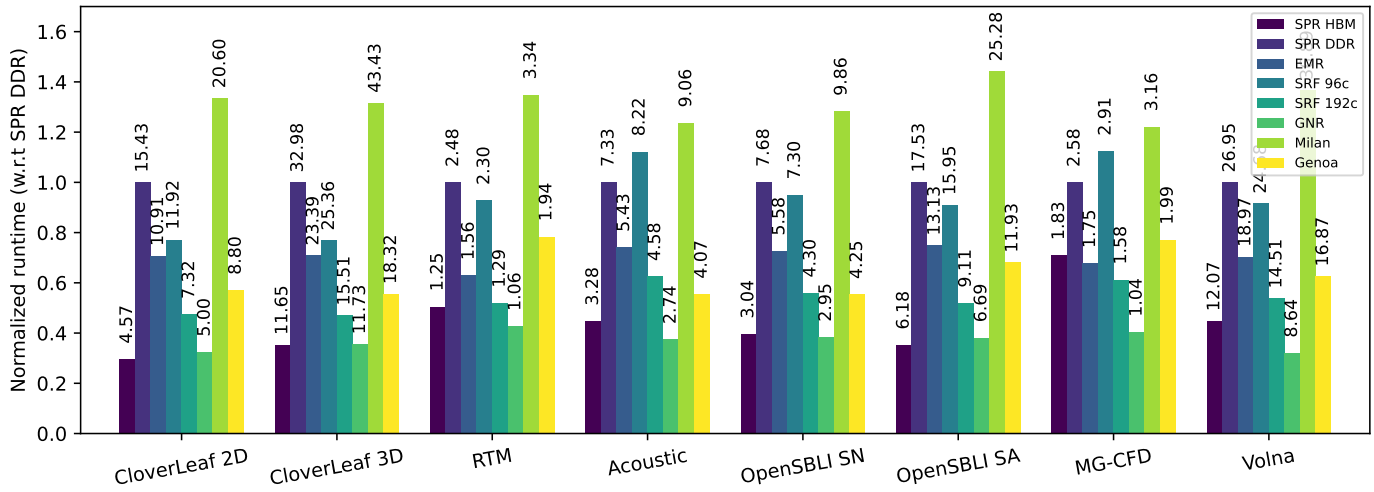

Figure 1: Execution time of different benchmarks on the test systems, normalized to SPR DDR. Data labels are absolute runtimes.

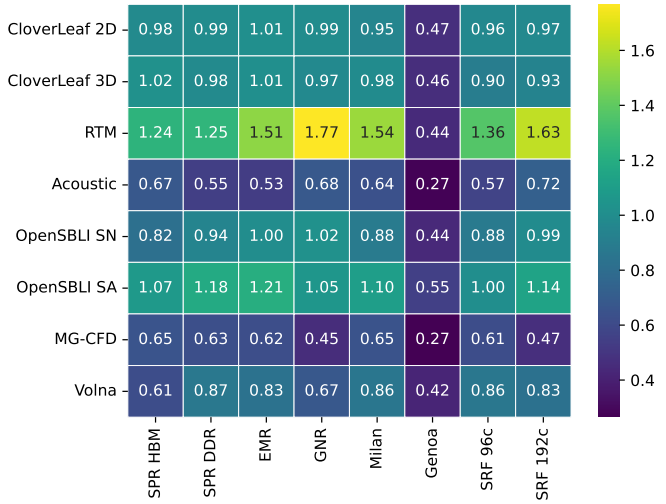

Figure 2: Relative performance of hybrid MPI+OpenMP compared to pure MPI - values above 1.0 indicate MPI+OpenMP performing better.

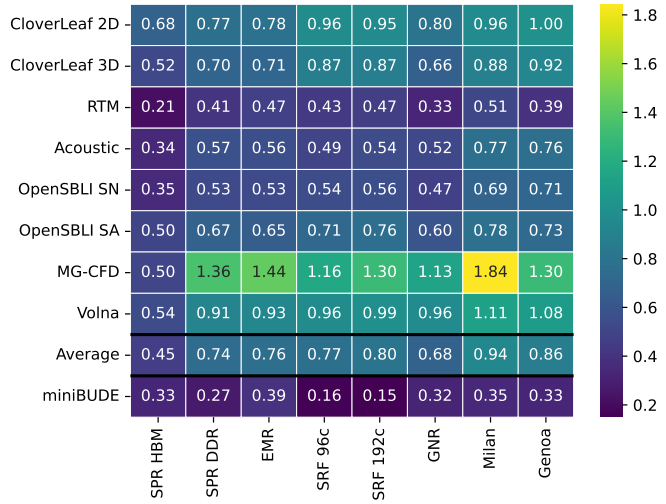

Figure 3: Fraction of peak bandwidth achieved, and their average by platform. Fraction of peak FP32 compute shown separately for miniBUDE

+OpenMP compared to pure MPI, and Figure 3 shows the hardware efficiency as calculated by the fraction of peak bandwidth (from Table 1) and achieved effective bandwidth. miniBUDE is reported separately in Figure 3 showing fraction of peak FP32 compute achieved.

The first immediate observation is that the SPR HBM system closely competes with GNR on the structured-mesh benchmarks; these applications all have regular memory access patterns and low to moderate amounts of computational intensity, and therefore the primary factor determining performance should be the available bandwidth. As Figure 3 shows, on SPR HBM only 45% of peak bandwidth is achieved on average, as shown in prior work [15] this is largely due to a shift towards latency limitations, as well as a lack of concurrency capable of saturating the available bandwidth [16]. Indeed, on average SPR HBM spends 26% of total runtime doing MPI communications, compared to 18-22% on the other Intel platforms. On unstructured mesh applications, which heavily rely on indirect memory accesses, SPR HBM falls behind GNR.

The simplest application, CloverLeaf 2D exhibits the highest fraction of peak bandwidth achieved, and is the most consistent across different platforms too. Due to its simpler access patterns, it even has cross-loop data reuse, resulting in close to 100% utilization thanks to caching effects. CloverLeaf 2D also has a small MPI overhead; as little as 1.2% on EMR, up to 20% on Genoa (average at 7%). MG-CFD also exhibits excellent memory locality across loops, especially at coarser multigrid levels, hence the high efficiency values. For both codes we can see a trend to higher efficiencies with larger cache sizes.

At the other end of the spectrum, RTM and Acoustic employ high-order (8^{th}) stencils, and especially the former has complex control flow and significant amounts of compute, therefore we observe lower bandwidth utilization. These applications are especially communication-intensive, with 34% and 44% of time spent in MPI respectively.

In-between, with the OpenSBLI applications, the Store All

(SA) version achieves on average 65% bandwidth utilization, and spends only 10% of runtime in MPI, and the more computationally intensive Store None (SN) version achieves 53% of peak bandwidth and has a 25% MPI overhead - nevertheless SN is actually 2.3× faster than SA, showing that it is well worth trading computations for a reduction in memory movement. The Volna unstructured-mesh code does not have multi-grid, yet achieves good cache locality in the indirect-access loops (of which there are fewer compared to MG-CFD).

The compute-intensive miniBUDE benchmark on the other hand is limited by compute throughput and control latency to a smaller some extent. we observe 27-39% efficiency, steadily increasing with subsequent compute-oriented generations. The Sierra Forest platforms are built with efficiency cores, with fewer and smaller (256-bit AVX2 instead of AVX-512) vectors, they achieve a lower fraction of peak theoretical compute.

For the simpler, lower-order stencil codes (CloverLeaf, OpenSBLI) there is little difference between MPI and MPI+OpenMP, as shown in Figure 2 - with the two usually within 5-10% of each other - but nevertheless this difference is significant enough that practitioners need to evaluate both, and pick the better configuration. For the high-order RTM code, moving to MPI+OpenMP cuts down significantly on communication costs. Comparing unstructured mesh codes MG-CFD and Volna is not entirely fair, as the pure MPI version vectorizes well, and thus significantly outperforms MPI+OpenMP. Notably the Genoa platform performs poorly with MPI+OpenMP; this is because the VM did not expose the multiple NUMA domains (only 1 per socket), therefore NUMA issues were still observed.

4.1. Performance evolution across generations of Xeons

It is revealing to analyze the performance improvement of the subsequent generations of Xeon processors, given the architectural improvements over time. Designs moved from 4 chiplets with CPU cores in SPR to 2 chiplets with EMR, and to 3 with GNR, last level cache size increased from 105 MB per socket from SPR to 432 MB with GNR. The introduction of energy-efficient designs with SRF also significantly moved the architectural trade-offs.

Considering that most of our benchmarks are mainly bandwidth-limited, we calculate the ratio between the observed speedup over SPR DDR to the ratio of available bandwidth compared to SPR DDR: $(runtime_X/runtime_{SPR\ DDR})/(bandwidth_X/bandwidth_{SPR\ DDR})$. This gives us a sense of how "balanced" the architectural evolution is in terms of improvements in bandwidth and other factors (cache, compute, latency).

The most obvious case is that of the SPR HBM, which is architecturally the same as SPR DDR with the exception of the high-bandwidth memory, and therefore our benchmark applications experience a shift from being purely bandwidth limited to being more constrained by latency and cache behavior, only improving overall by 64% compared to the improvement in bandwidth.

The Emerald Rapids (EMR) platform saw a 15% improvement in core count, a 3× increase in cache size, a move from 2 chiplets to one, and a 40% improvement in available bandwidth. Overall these improvements have been balanced with applications improving overall within 2% of the bandwidth increase, meaning EMR is approximately 40% faster than SPR DDR on these benchmarks. The Granite Rapids (GNR) platform presents an interesting further step in this direction: we have another 12.5% increase in core count, a 1.35× increase in cache size, and 2.13× improvement in bandwidth compared to Emerald Rapids. This further massive lift in available bandwidth is slightly less balanced however, with 91% speedup compared to the improvement in bandwidth. Nevertheless, GNR in absolute terms is still on average 2.71× faster than SPR DDR and 1.93× faster than EMR.

The Sierra Forest family of CPUs represents a different set of architectural trade-offs with their energy-efficient design. The 96 core variant has 8 memory channels per socket, just like SPR and EMR, while the 192 core variant has 12 channels, similarly to GNR. While core counts are 1.3-3.4× higher, cache sizes are smaller, and the available bandwidth is 4% (96 core) or 70% (192 core) higher compared to SPR DDR. Yet SRF is on average 5% and 10% faster compared to the increase in bandwidth for the 96 core and 192 core variants respectively. Thanks to the higher amount of concurrency, it can more efficiently saturate the available bandwidth - and despite the higher core counts it does not suffer from significantly higher MPI overheads (19-23% on average, vs. 18-21% on SPR/EMR/GNR). The 192 core SRF platform represents a balanced improvement over the 96 core one, with a 1.65× increase in bandwidth, yielding a 1.73× improvement in the overall performance of our benchmarks (except miniBUDE, which has a 2x improvement, directly proportional to available cores).

The AMD CPUs are not generationally comparable to these Intel CPUs, therefore we only calculate improvements from Milan to Genoa: there is a 1.5× improvement in the number of cores, a 3× increase in cache size, and a 2.25× increase in available bandwidth. Genoa achieves a 91% efficiency compared to Milan, mainly due to the cost of MPI operations not reducing at the same rate as compute and bandwidth increase.

4.2. Energy efficiency

Intel has built the Running Average Power Limit [17] counters and controls into their chips since the Sandy Bridge generation, which among other things has made it possible to accurately measure power consumption at the millisecond-scale. While energy counters may be available for the whole system, and components such as integrated graphics, main memory, and CPU package, on the studied Intel platforms we were able to measure the consumption of the CPU+memory system, and just the memory subsystem on EMR, SRF, and GNR. We were able to access these counters on the Milan system, but on Genoa these counters were not available due to poteintial security vulnerabilities in cloud VMs.

Figure 6 gives an overview of the reported energy consumption (CPU and memory together, and memory separately, for SPR HBM only on-package memory is included, external DRAM is not). The trends paint a similar picture to the execution times, since most of these CPUs have the same 2×350 W TDP (SPR,

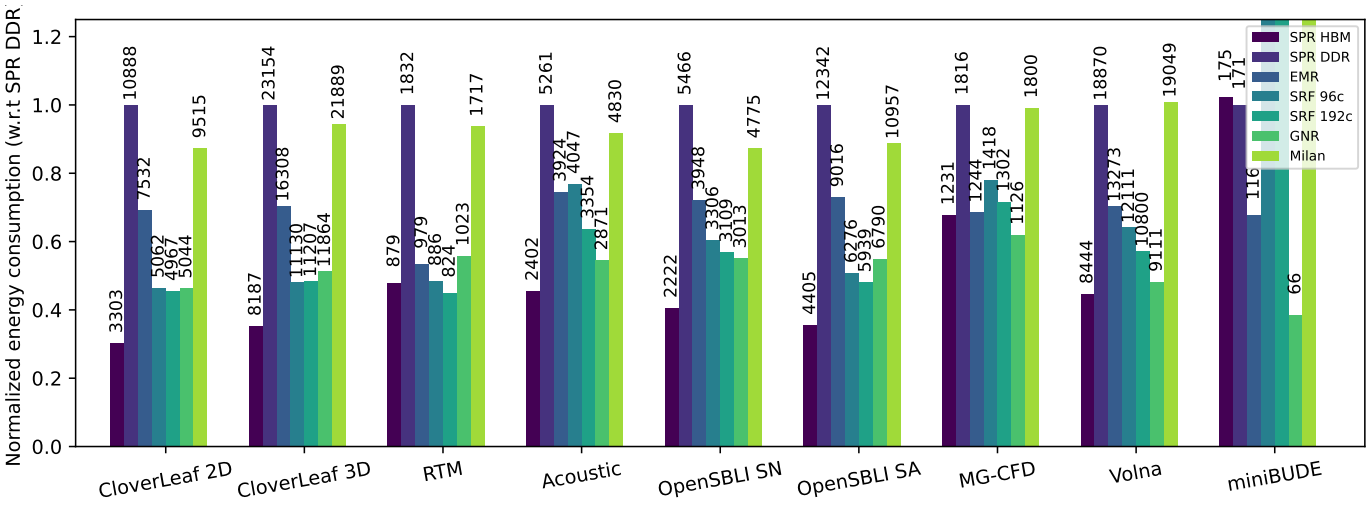

Figure 4: Energy consumption of different benchmarks on different Intel CPU platforms, normalized to SPR DDR. Data labels are Joules.

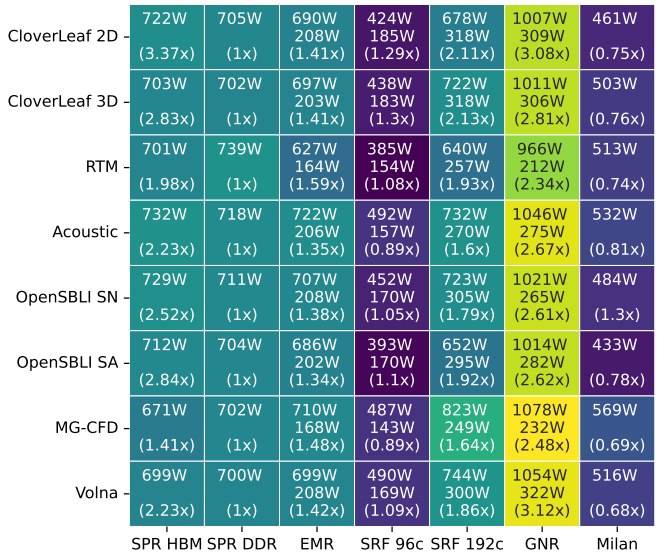

Figure 5: Average power draw of the CPU and memory, the memory separately (where available), and speedup compared to SPR DDR

EMR, SRF 192c), but notably the SRF 96c system has 2×250 W, and the GNR has 2×500 W (these TDP figures apply to the CPU only and exclude off-chip memory). This is also shown in Figure 5 that presents the average power (energy consumption divided by execution time), as well as the speedups in execution time compared to SPR DDR. We can see SPR HBM running on average 2.4× faster than SPR DDR, but at the same power, yielding a corresponding 2.4× improvement in energy efficiency. The EMR system on average consumes 20 W less power than SPR DDR, but runs 1.42× faster, and therefore is 1.46× more energy efficient. The SRF 96 core system runs at only 445 Watts, yet achieves a 1.09× speedup, and therefore is 1.75× more energy efficient. The SRF 192 core system increases TDP, but also bandwidth and overall performance - it is 1.87× faster than SPR DDR, and is 1.88× more energy efficient. The GNR system further increases TDP and effective power to 1025 Watts on average - while being 2.72× faster than SPR DDR, it is also 1.88× more energy efficient. Power figures for the memory system are also included in Figure 5 where we were able to measure them; on both EMR and SRF 96c they account for 28% of total power - they both have 16 DIMMs installed, though the former has 5600 MHz modules, while the latter has 6400 MHz ones. The SRF 192 core system has 24 DDR5-6400MHz modules installed, and the memory system represents 40% of total power on these benchmarks. The GNR system has 24 DDR5-8800 MHz modules installed, accounting for 27% of total power.

While in terms of raw performance Milan was lagging behind even the SPR DDR platform, it is built with a better process, and is more energy efficient in almost all applications, with the notable exception of the compute-bound miniBUDE, where due to a lack of support for AVX512, it performs comparatively worse.

The SRF and GNR systems nicely demonstrate the available trade-offs between energy efficiency and performance in relative terms to SPR DDR:

1. SRF 96 core: Slightly higher performance (1.09×) but at a 1.4× lower power
2. SRF 192 core: Significantly higher performance (1.87×), at the same power
3. GNR: Even higher performance (2.72×), but at 1.44× more power

4.3. Improving locality

The OPS library supports transforming the execution in a way that improves cache locality across a sequence of computational loop nests using a sparse tiling scheme, and also avoids communications across MPI utilizing an overlapped tiling scheme [18]. We enable this optimization for the CloverLeaf and the OpenSBLI benchmarks - with RTM and Acoustic we have a single high-order loop repeating, resulting in excessive increases in the overlapped regions, rendering the optimization counterproductive. For each application, we evaluate

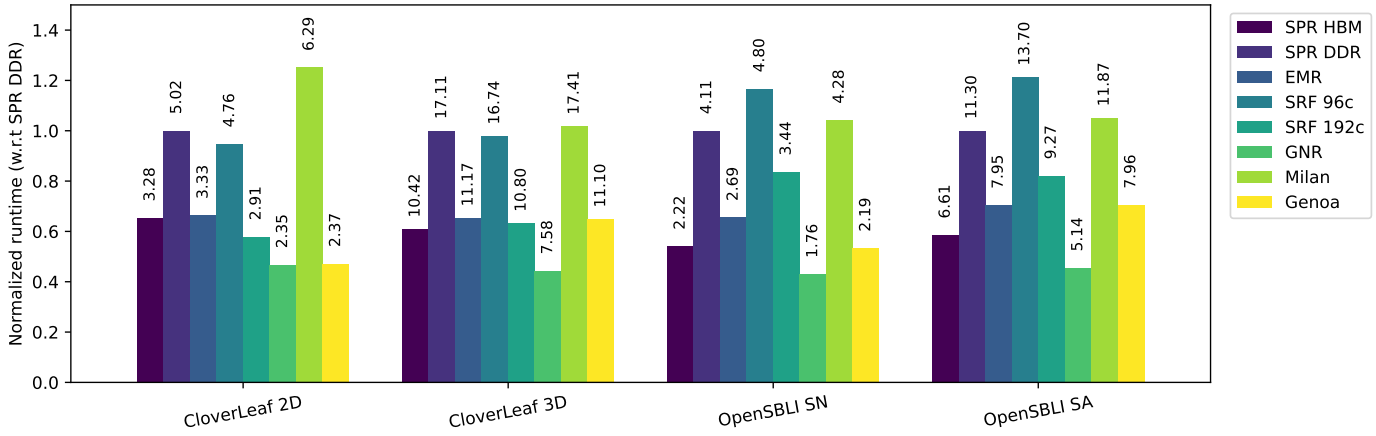

Figure 6: Normalized execution time (wrt SPR) of key applications when enabling cache-blocking tiling

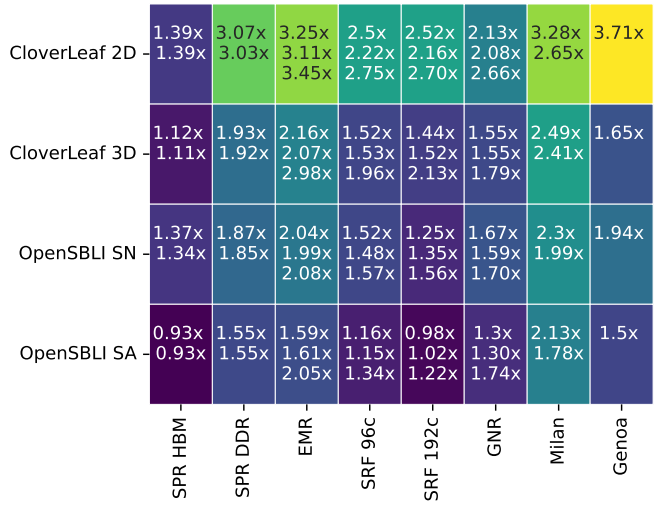

Figure 7: Improvements when enabling cache-blocking tiling: overall runtime (first row), reduction in overall energy consumption (second row), reduction in the energy consumption of the memory system (third row)

4 values of the tile size parameter, and pick the best performing configuration with respect to execution time.

Figure 7 shows the speedups and energy improvements over the baseline implementations. In the first row of each cell, we have the execution time speedup, the second shows the reduction in overall energy consumption, and the third row the reduction in the energy consumption of the memory system, where available. This tiling optimization is the most effective on the structurally simplest application, CloverLeaf 2D, with speedups ranging from 1.39× to 3.71× - indeed the speedups correlate well with the difference between maximum cache bandwidth and main memory bandwidth (smallest on the SPR HBM at 2.4×, and all the way to 18× on Genoa). On average, communication times decreased by 3.8× for CloverLeaf 2D. The 3D applications benefit less from this optimization due to more complex tile shapes and improved, but still relatively expensive communications: on average by 2.8×. For these four benchmarks, SPR HBM was 4% faster than GNR on average without tiling, however, with tiling enabled GNR is now 33% faster overall.

Studying the overall energy consumption with the tiling optimization enabled shows that on average it is well in line with the overall execution time reduction; 1.8× speedup and 1.7× less energy on average, but what is very revealing is the reduction in the memory consumption of the memory subsystem, which is much higher, on average 2.1×. This demonstrates that significantly fewer memory requests have to go out to main memory, and are instead fulfilled from cache - leading to the reduced power consumption of the memory system, and a higher power consumption of the CPU itself.

5. Conclusions

This paper evaluated the performance and energy efficiency of AMD's and Intel's recent generations processors released over the past two years, from the Sapphire Rapids generation, through Emerald Rapids, and to the recently differentiated energy-efficient Sierra Forest and performance-oriented Granite Rapids generations on Intel's side, and Milan and Genoa on AMD's. We focus on a number of bandwidth-bound structured mesh and unstructured mesh applications.

For the traditional performance-oriented Xeons these generations brought a 3-4× improvement in cache size and available main memory bandwidth, but only a 38% increase in core counts. Overall, our applications' performance on Granite Rapids improved by 2.7× compared to Sapphire Rapids, and at the same time improved energy efficiency by 1.88×, despite the higher TDP of 500 W per socket. The newly introduced energy-efficient Sierra Forest family of CPUs in contrast have much higher core counts (96-192 per socket), and are more energy efficient; delivering 9% more performance than Sapphire Rapids with 75% less energy at only 250 W per socket (96 core version), or 87% higher performance at the same 350 W per socket, resulting in 87% less energy consumed (192 core version).

Despite the significant advances in available memory bandwidth, it is still a bottleneck for many applications, including ours - therefore we evaluated the memory locality improving optimization (cache blocking tiling) on the CloverLeaf and

OpenSBLI applications, yielding significant overall speedups of 1.75×. While energy consumption decreases proportionally, we show that there is a significant shift with the power of the memory system reducing, and the power of the CPU increasing.

Our work underlines the importance of architectural advancements in improving the performance and energy efficiency of HPC systems. The rapid evolution from Sapphire Rapids to Granite Rapids brought substantial benefits, particularly for bandwidth-bound applications. The introduction of the energy-efficient Sierra Forest family further demonstrates the potential for balancing performance and energy consumption in server CPUs. These findings provide valuable insights for optimizing HPC workloads and guiding the development of next-generation processors.

Acknowledgment

We are grateful for the support of the OneAPI Innovator program, and the advice and assistance of Rupak Roy, Omar Toral, Sri Ramkrishna at Intel in particular.

This research was supported by National Research, Development and Innovation Fund of Hungary (FK 145931), under the FK 23 funding scheme, as well as by Rolls-Royce plc., and by the UK EPSRC (EP/S005072/1 – Strategic Partnership in Computational Science for Advanced Simulation and Modeling of Engineering Systems – ASiMoV). This study was also partially funded by the National Research, the Development and Innovation Office in Hungary (RRF-2.3.1-21-2022-00004).

References

[1] T. Deakin, J. Price, M. Martineau, S. M. Smith, Evaluating attainable memory bandwidth of parallel programming models via babelstream, International Journal of Computational Science and Engineering 17 (2018) 247. doi:10.1504/IJCSE.2018.095847.

[2] S. Giallorenzo, J. Mauro, M. G. Poulsen, F. Siroky, Virtualization costs: Benchmarking containers and virtual machines against bare-metal, SN Computer Science 2 (2021) 404. doi:10.1007/s42979-021-00781-8.
URL https://doi.org/10.1007/s42979-021-00781-8

[3] I. Z. Reguly, G. R. Mudalige, M. B. Giles, D. Curran, S. McIntosh-Smith, The ops domain specific abstraction for multi-block structured grid computations, in: 2014 Fourth International Workshop on Domain-Specific Languages and High-Level Frameworks for High Performance Computing, IEEE, 2014, pp. 58–67.

[4] I. Reguly, Op2: An active library framework for solving unstructured mesh-based applications on multi-core and many-core architectures, in: 2012 Innovative Parallel Computing (InPar), IEEE, 2012, pp. 1–12.

[5] I. Reguly, A. Mallinson, W. Gaudin, J. Herdman, Performance analysis of a high-level abstractions-based hydrocode on future computing systems, in: High Performance Computing Systems. Performance Modeling, Benchmarking, and Simulation: 5th International Workshop, PMBS 2014, New Orleans, LA, USA, November 16, 2014. Revised Selected Papers 5, Springer, 2015, pp. 85–104.

[6] C. T. Jacobs, S. P. Jammy, N. D. Sandham, OpenSBLI: A framework for the automated derivation and parallel execution of finite difference solvers on a range of computer architectures, Journal of Computational Science 18 (2017) 12–23. doi:10.1016/j.jocs.2016.11.001.

[7] A. Owenson, Towards the use of mini-applications in performance prediction and optimisation of production codes, Ph.D. thesis, University of Warwick (2020).

[8] G. R. Mudalige, I. Z. Reguly, A. Prabhakar, D. Amirante, L. Lapworth, S. A. Jarvis, Towards virtual certification of gas turbine engines with performance-portable simulations, in: 2022 IEEE International Conference on Cluster Computing (CLUSTER), IEEE, 2022, pp. 206–217.

[9] A. Mallinson, D. A. Beckingsale, W. Gaudin, J. Herdman, J. Levesque, S. A. Jarvis, Cloverleaf: Preparing hydrodynamics codes for exascale, The Cray User Group 2013 (2013).

[10] M. Louboutin, M. Lange, F. Luporini, N. Kukreja, P. A. Witte, F. J. Herrmann, P. Velesko, G. J. Gorman, Devito (v3. 1.0): an embedded domain-specific language for finite differences and geophysical exploration, Geoscientific Model Development 12 (3) (2019) 1165–1187.

[11] A. Owenson, S. A. Wright, R. A. Bunt, Y. Ho, M. J. Street, S. A. Jarvis, An unstructured cfd mini-application for the performance prediction of a production cfd code, Concurrency and Computation: Practice and Experience 32 (10) (2020) e5443.

[12] I. Z. Reguly, D. Giles, D. Gopinathan, L. Quivy, J. H. Beck, M. B. Giles, S. Guillas, F. Dias, The volna-op2 tsunami code (version 1.5), Geoscientific Model Development 11 (11) (2018) 4621–4635.

[13] A. Poenaru, W.-C. Lin, S. McIntosh-Smith, A performance analysis of modern parallel programming models using a compute-bound application, in: B. L. Chamberlain, A.-L. Varbanescu, H. Ltaief, P. Luszczek (Eds.), High Performance Computing, Springer International Publishing, Cham, 2021, pp. 332–350.

[14] I. Z. Reguly, Evaluating the performance portability of sycl across cpus and gpus on bandwidth-bound applications, in: Proceedings of the SC '23 Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis, SC-W '23, Association for Computing Machinery, New York, NY, USA, 2023, p. 1038–1047. doi:10.1145/3624062.3624180.
URL https://doi.org/10.1145/3624062.3624180

[15] I. Z. Reguly, Comparative evaluation of bandwidth-bound applications on the intel xeon cpu max series, in: Proceedings of the SC '23 Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis, SC-W '23, Association for Computing Machinery, New York, NY, USA, 2023, p. 1236–1244. doi:10.1145/3624062.3624195.
URL https://doi.org/10.1145/3624062.3624195

[16] J. D. McCalpin, Bandwidth limits in the Intel Xeon Max (Sapphire Rapids with HBM) Processors, in: ISC 2023 IXPUG Workshop, 2023, pp. 1–4.

[17] H. David, E. Gorbatov, U. R. Hanebutte, R. Khanna, C. Le, Rapl: memory power estimation and capping, in: Proceedings of the 16th ACM/IEEE International Symposium on Low Power Electronics and Design, ISLPED '10, Association for Computing Machinery, New York, NY, USA, 2010, p. 189–194. doi:10.1145/1840845.1840883.
URL https://doi.org/10.1145/1840845.1840883

[18] I. Z. Reguly, G. R. Mudalige, M. B. Giles, Loop tiling in large-scale stencil codes at run-time with ops, IEEE Transactions on Parallel and Distributed Systems 29 (4) (2017) 873–886.