Evaluation of Mobile ARM-Based SoCs for High Performance Computing

Andreas Selinger
Institute for Microelectronics
TU Wien
Gußhausstraße 27-29/E360
A-1040 Wien, Austria
selinger@iue.tuwien.ac.at

Karl Rupp
Institute for Microelectronics
TU Wien
Gußhausstraße 27-29/E360
A-1040 Wien, Austria
rupp@iue.tuwien.ac.at

Siegfried Selberherr
Institute for Microelectronics
TU Wien
Gußhausstraße 27-29/E360
A-1040 Wien, Austria
selberherr@iue.tuwien.ac.at

ABSTRACT
The electrical power consumption of current supercomputers has become the key limiting factor to successfully reach exascale. To further increase energy efficiency, the use of mobile system-on-chips (SoCs) has been proposed in the past. In this work we investigate four development boards equipped with different SoCs with respect to their suitability for HPC. Each of the boards is carefully benchmarked for high floating point computing performance, high memory bandwidth, low latencies, and power consumption, all of which we identified as key properties for successful HPC systems. In contrast to earlier work, we also include OpenCL-capable graphics processing units integrated in the SoC in our benchmarks.

Our benchmark results show that mobile SoCs are not ready for a successful adoption in HPC yet. While theoretical peak compute performance and memory bandwidth suggest SoCs to be a competitive with existing HPC systems, practical values are most of the time much lower. Therefore, none of the SoCs provides an attractive option for HPC in terms of lower acquisition costs or higher energy efficiency.

Author Keywords
ARM; System on Chip; Linear Algebra; HPC

ACM Classification Keywords
C.4 PERFORMANCE OF SYSTEMS

1. INTRODUCTION

The high performance computing (HPC) landscape was dominated by special-purpose vector and single instruction multiple data (SIMD) architectures in the early 1990s. In the mid- to late 1990s, microprocessors such as DEC Alpha, SPARC, and MIPS, which were used in workstations during that time, started to take over HPC. About ten years later, these reduced instruction set computers (RISC) computing processors were, in turn replaced by the x86 complex instruction set computing (CISC) architecture in commodity computers [10].

Each of these transitions was fueled by the higher cost-effectiveness of the higher-volume markets. The design, verification, and fabrication of increasingly complicated, custom-made processors for HPC and for the mass market require comparable resources. However, units sold in mass markets allow for a much better cost amortization, thinning out the per-processor cost for the design, verification, and fabrication in proportion to the difference in market size. One of the most prominent inflection points was ASCI Red, the first supercomputer to provide more than one trillion (tera) floating point operations per second (TFLOPS) [8]. ASCI Red was deployed in 1997 and consisted of Pentium Pro x86 processors, the first to provide integrated double precision floating point units. Since then, x86 systems expanded their share of systems in the TOP500 list[1] of supercomputers up to 90 percent by the end of 2015.

Today, x86 systems are possibly facing a similar threat of becoming replaced in the HPC landscape by the higher-volume architecture in mobile system-on-chips (SoCs). One such sign is that the market for commodity personal computers is currently on the decline, while the market for mobile devices is growing fast. Moreover, with the stagnation of clock frequencies due to power dissipation issues, power-efficient mobile SoCs appear to be the perfect fit for further increasing performance within a given power budget.

Former evaluations demonstrated the feasibility of HPC workloads on small clusters based on mobile SoCs [4, 7]. One of the most prominent examples is the European Mont-Blanc project with the aim of designing an exascale-architecture based on ARM cores [10, 11]. At the same time, however, these earlier studies also identified challenges yet to overcome, most notably better support for double precision arithmetic, support for error-correcting memory, and the limitations of a 32-bit address space in terms of total memory capacity. The purpose of this work is to revisit currently available SoCs for their suitability in HPC. In contrast to earlier works, we not only consider the central processing units on SoCs, but also the attached graphics processing units for general purpose computations. With the broader availability of the cross-platform heterogeneous programming model OpenCL, these units become now accessible for general HPC workloads.

[1] `http://top500.org/`

SpringSim-HPC 2016 April 3-6 Pasadena, CA, USA

A mobile SoC is optimized for low power consumption and hence provides smaller peak performance than a typical processor in conventional supercomputers. Therefore, a supercomputer based on mobile SoCs requires a higher number of nodes to achieve the same overall performance. Depending on the actual configuration, the difference in the number of nodes can be up to an order of magnitude. The additional networking hardware required in such a case must be accounted for when considering the total power budget of a supercomputer. Consequently, there has to be a clear advantage in terms of performance per Watt for mobile SoCs on the node level to be attractive for HPC. However, our study does not find such an advantage of mobile SoCs in terms of performance per Watt on the node level. Thus, even though our study focuses on a comparison on the node-level, its findings are also relevant for large-scale supercomputers with thousands of nodes.

The remainder of this work is organized as follows: In Section 2 we discuss and identify key requirements for HPC systems. The hardware and software setups used for the benchmarks in our study are sketched in Section 3. Section 4 presents our benchmark results which are then followed by a discussion in Section 5. In the conclusions in Section 6 we argue that based on our results it is unlikely that mobile SoCs will play a significant role in the HPC landscape before reaching exascale.

2. REQUIREMENTS ON HPC SYSTEMS

Modern general purpose HPC systems require a careful balance of hardware features as well as a mature software stack. If suitability for general purpose workloads is not a design goal, specialized systems such as *Anton* [13, 14] for molecular dynamics simulation can be designed to achieve higher, yet problem-specific, efficiency [20]. Such specialized systems, however, serve only a relatively small community, hence we will focus on general purpose systems in the following.

HPC systems have almost fully converged with respect to the operating system and hence large parts of the software stack. The November 2015 update of the TOP500 list of supercomputers reported 494 out of 500 systems to be run on Linux; the remaining six systems run on the AIX operating system. As a consequence, certain HPC-centric software packages no longer need to support a variety of different operating systems, but can instead focus their resources on a single platform. Conversely, the dominance of Linux in HPC mandates that any new HPC hardware architecture must ensure good support by Linux distributions. This is the case for common mobile SoCs with ARM-based central processing units as well as x86-based mobile SoCs.

Since mobile SoCs are clearly ready for HPC from the software stack perspective, closer inspection is required on hardware details. The three key characteristics we consider in this work, namely high FLOPs for compute-intensive applications, high bandwidth for applications with low arithmetic intensity, and the necessity of low latency for good strong scaling properties, are discussed in the following.

2.1 Requirement 1: High FLOPS

The most prominent benchmark for measuring the performance of current supercomputers is the HPLinpack benchmark used for the TOP500. While HPLinpack is a formal specification of the computations to be measured, HPL[2] is the most commonly used implementation of the HPLinpack benchmark. The benchmark describes the Gaussian elimination process for solving a dense linear system and thus requires $\mathcal{O}(N^3)$ FLOPs for a system requiring $\mathcal{O}(N^2)$ bytes of memory. Consequently, the benchmark has high arithmetic intensity and is ultimately compute-limited provided that the caches are large enough.

The fastest system in the November 2015 edition of the TOP500 provides a HPLinpack performance of 34 Peta-FLOPS (PFLOPS) while drawing 18 MW of electrical power. On the other hand, the power consumption for a machine with one Exa-FLOPS is targeted to be 20 MW, so a 25-fold improvement in power efficiency is needed. Therefore, FLOPS/Watt rather than FLOPS is a better metric for evaluating the performance of mobile SoCs.

2.2 Requirement 2: High Memory Bandwidth

It has even been argued that the use of HPLinpack for the TOP500 encourages organizations to make poor system choices and that the TOP500 ranking consequently does not give an indication of system value [5]. Alternative benchmarks such as High-Performance Geometric Multigrid (HPGMG) [1] and High-Performance Conjugate Gradients (HPCG) [3] were proposed to provide a better overall picture of HPC system performance. Both rely on the solution of sparse systems of equations by means of iterative methods: HPGMG applies full geometric multigrid, while HPCG uses a conjugate gradient method accelerated by a geometric multigrid preconditioner.

HPCG achieves the goal of measuring system performance without relying on raw compute power only. As first results have shown, the peak performance for HPCG on current systems is primarily determined by the available memory bandwidth [6] and thus can be equally well measured using the STREAM benchmark [9]. Therefore, we select high memory bandwidth as our second figure of merit for HPC systems. Instead of using the HPCG benchmark, however, we use the much simpler and equally appropriate STREAM benchmark for assessing memory bandwidth.

2.3 Requirement 3: Low Latency

Good HPC clusters behave favorably in the weak scaling and the strong scaling limit. In the weak scaling limit, the work size per process is kept constant and the number of processes (and hence the problem size) is increased. Usually, best weak scalability is obtained by selecting the work size per process as large as possible, because any latency effects are suppressed. In the strong scaling limit, the total problem size is kept fixed and the number of processes is increased. Consequently, the work size per process decreases so that latencies ultimately limit the strong scalability.

[2] `http://www.netlib.org/benchmark/hpl/`

	Jetson TK1	Odroid XU3-Lite	Parallella	Wandboard Quad
Vendor	NVIDIA	Hardkernel	Adapteva	Community
CPU Architecture	Cortex-A15	Cortex-A15/A7	Cortex-A9	Cortex-A9
SoC	Tegra K1	Exynos 5422 Octa	Zynq 7Z010 CPU	i.MX6 Quad
CPU Cores	4	4+4	2	4
GPU/FPGA Name	NVIDIA Kepler GK 20a	ARM Mali-T628	Epiphany III	Vivante GC 2000
Main Memory	2 GB DDR3	2 GB DDR3	1 GB DDR3	2 GB DDR3

Table 1. Overview of mobile hardware used for the benchmarks in this work.

Latency is inherent to all components in a system: On the lowest level, latencies in loading data from cache or global memory are observed. Also, the synchronization of threads or processes through mechanisms such as locks or mutexes induces latency. The primary source of latency beyond a single node is due to the network communication, which is inherently limited by the speed of light.

In this work we only consider latency within a single node, as network-induced latencies are independent of the node architecture. The most important OpenMP constructs are compared, covering a wide application range: The time to enter an OpenMP parallel region, the time for entering a parallel for-loop, the overhead of an OpenMP barrier, the time for OpenMP `single` and `critical` sections, the overhead of locking and unlocking of shared resources, ordered access, atomic operations, and reductions. The latency of mobile SoCs must be competitive with current x86 processors, if they want to be attractive for HPC.

3. BENCHMARK SETUP

In the following we define the hardware and software used for obtaining the benchmark results in Section 4. Because a plethora of different hardware and software is available for ARM-based mobile SoCs, the hardware and software chosen is such that it covers a broad range of central processing units (CPUs), graphics processors or co-processors, and software used in HPC. To better compare the results, we also include results obtained on a laptop equipped with an Intel Core i3-3217U CPU (17 Watt thermal design power) to better compare with the performance provided by an x86-based platform. The laptop display was powered off during the measurements to provide a fair comparison of power consumption. A detailed description of the mobile hardware boards is given in the following.

3.1 Hardware

Mobile SoCs fabricated at high volume include good graphics processing units (GPUs). Consequently, we chose products where general purpose computations can be carried out using an OpenCL framework [15] in order to maximize the overall system's performance. An exception is the Jetson TK1 which provides a CUDA toolchain for programming the GPU. Because only some applications can take benefit from GPU acceleration, we benchmark the CPU and the GPU of each SoC separately.

An overview of the devices selected for our benchmarks is given in Table 1. The Jetson TK1 and the Odroid XU3-Lite are equipped with high-performance Cortex-A15-based CPUs [17], the latter implementing ARM's big.LITTLE concept by pairing the faster Cortex-A15-based CPUs with low-power Cortex-A7 CPUs. However, only the Cortex-A15 cores are benchmarked due to their higher performance. We note that the hardware on the Odroid XU3-Lite is very similar to the hardware used in the Mont-Blanc project mentioned in the introduction. The Parallella board and the Wandboard Quad board are equipped with Cortex-A9 CPUs, which represent a compromise between faster Cortex-A15 and lower-power Cortex-A7.

The Jetson TK1 integrates 192 NVIDIA CUDA cores (Kepler), providing 326 GFLOPS of theoretical single precision compute power. Double precision performance, however, is restricted to only 13 GFLOPS. The Mali-T628 GPU in the Odroid XU3-Lite is capable of a theoretical peak performance of 142 GFLOPS in single precision and 42 GFLOPS in double precision. The field-programmable gate array (FPGA) Epiphany III on the Parallella board provides 32 GFLOPS of theoretical peak floating point performance in single precision (no double precision capabilities). Similarly, the Vivante GC 2000 on the Wandboard Quad is able to provide a theoretical peak of 32 GFLOPS of floating point performance in single precision. However, currently only an OpenCL SDK supporting the embedded profile is available for the Vivante GC 2000, hence GPU benchmark results could not be collected for the Wandboard Quad.

All four boards provide a single DDR3 memory channel. The theoretical peak memory bandwidth of the Jetson TK1 and the Odroid XU3-lite is 14.9 GB/sec. On the Parallella board and the Wandboard the theoretical peak memory bandwidth is 12.8 GB/sec. The Parallella board provides one Gigabyte of main memory, while the other three boards provide two Gigabytes. Because all four boards implement a 32-bit CPU microarchitecture (ARMv7-A), the upper limit for main memory capacity is 4 GB.

We measured the power consumption at the wall outlet using an ELV Energy Master Basic 2 with an accuracy of one percent and 0.1 Watt in the range 0.1 to 100 Watt. To mimic the use in an actual HPC system as closely as possible, the boards were operated in headless mode without additional external connectors other than a Gigabit network connection.

3.2 Software

Vendor-provided (community-provided for the Wandboard Quad) customized versions of Ubuntu 14.04 were used on all four boards for running the benchmarks. By using a common operating system base, any deviations in the performance results due to differences in the software stack are minimized.

To quantify the performance of the GPUs as well as the FPGA on the boards, we used the OpenCL-based auto-tuning framework for linear algebra kernels available in ViennaCL [12, 16]. For the Jetson TK1 we compared with the performance of the vendor-provided cuBLAS library.

The compute performance of the CPU cores was evaluated using OpenBLAS [19] for the FLOP-intensive matrix-matrix multiplication kernel. We used the vector triad in the STREAM benchmark for measuring memory bandwidth. For measuring the power draw at different arithmetic intensities we used a custom code in which three double precision values are loaded from main memory and then the necessary number of floating point operations were carried out to obtain the specified arithmetic intensity. We measured latency of different OpenMP features with the EPCC OpenMP micro-benchmark suite[3] [2]. The GNU compiler collection was used on all systems to minimize the effect of compiler-specific differences in the benchmarks. Reported values are median values of 10 benchmark runs, within which each result is the average of 20 repeated measurements.

4. RESULTS

The performances obtained for dense matrix-matrix multiplication for the CPUs and GPUs of the SoCs from Table 1 as well as the x86-based laptop are given in Figure 1. For the Vivante GC 2000 GPU on the Wandboard Quad we could not obtain measurement results because support for the OpenCL Full Profile is currently not available. Similarly, the OpenCL SDK on the Parallella board resulted in extremely low performance and could not compile the automatically generated kernels. The performances obtained on the CPUs are almost constant for matrices larger than 200-by-200. In contrast, the performance degrades for matrices with a size above 1200-by-1200 on the GPUs of both the Jetson TK1 and the Odroid XU3-Lite.

The performance for matrix-matrix multiplication on the Parallella board can be estimated from prior work [18]: Each core is capable of achieving about 1 GFLOPS for local matrices of size 32-by-32, provided that the data is already available on the chip. For the 16-core Epiphany III FPGA this results in about 16 GFLOPS for matrices of size 128-by-128. Larger matrices, however, are multiplied with much lower performance, because memory transfers to and from the Epiphany III chip suffer from a low memory bandwidth of only 0.15 GB/sec. This results in floating point rates below 10 GB/sec for matrices larger than 128-by-128.

The memory bandwidth obtained with the CPUs as well as the GPUs for the different SoCs is depicted in Figure 2. The peak bandwidth on the CPU is obtained for vectors with less than $100\,000$ elements, indicating beneficial caching effects. This caching effect is particularly pronounced on the x86 CPU, where the effective bandwidth increases up to 25 GB/sec. On GPUs, however, the kernel launch latency results in peak bandwidth to be attained for vectors with more than a million entries. On the Epiphany III FPGA the external memory bandwidth to DRAM was reported to be only 0.15 GB/sec [18], which could not be verified because of stability problems with the OpenCL SDK.

In order to better evaluate the contribution of memory transfers to and from main memory to the total power consumption, we used synthetic workloads and recorded the respective power draw, cf. Figure 3. For simplicity we only measured a load on the CPU. This is sufficient because GPUs on SoCs share the same memory with the CPU. The benchmark results identify the highest power consumption at an arithmetic intensity of about 1 FLOPS/Byte. At higher arithmetic intensity the transactions to and from main memory become less frequent, which is reflected by a reduced power consumption of one to two Watt. In other words, the largest share of the total power consumption can be attributed to the SoCs rather than the main memory.

The time required for OpenMP thread synchronization is given in Table 2. Timings span from only 0.02 to 8.6 microseconds for the different OpenMP features. Overall, there is no clear winner in terms of OpenMP overhead: The Wandboard provides low overhead for parallel regions, barriers, single as well as critical regions. On the other hand, the Jetson TK1 provides the fastest times for locking, ordered and atomic operations as well as reductions, but shows the largest overheads when entering parallel regions.

5. DISCUSSION

A successful HPC machine based on mobile SoCs must outperform existing clusters in at least one category out of energy efficiency for compute-intensive tasks, energy efficiency for memory-intensive tasks, or acquisition cost. In the following we discuss these options in light of the benchmark results from Section 4.

The CPUs on the four boards provide about 1 GFLOPS of raw compute performance per Watt in double precision (cf. Figure 1). Up to 10 GFLOPS/Watt are obtained in single precision on the Jetson TK1, but the performance degrades for larger matrices. Somewhat ironically, the good ratio is obtained on the GK 20a GPU, which is a scaled-down version of graphics processors already used for HPC. With 1 GFLOPS/Watt mobile SoCs are not attractive for current supercomputers; the x86-based laptop achieves 2 GFLOPS/Watt in single precision, whereas the best systems in the TOP500 achieve about 2 GFLOPS per Watt in double precision.

Mobile SOCs are not attractive for HPC from an acquisition point of view either: All four boards were priced between 100 and 200 USD at the time of purchase. Current HPC processors carry a price tag of about $1\,000$ to $10\,000$ USD, but also provide a factor of 10 to 50 higher perfor-

[3] `https://www.epcc.ed.ac.uk/research/computing/performance-characterisation-and-benchmarking/epcc-openmp-micro-benchmark-suite`

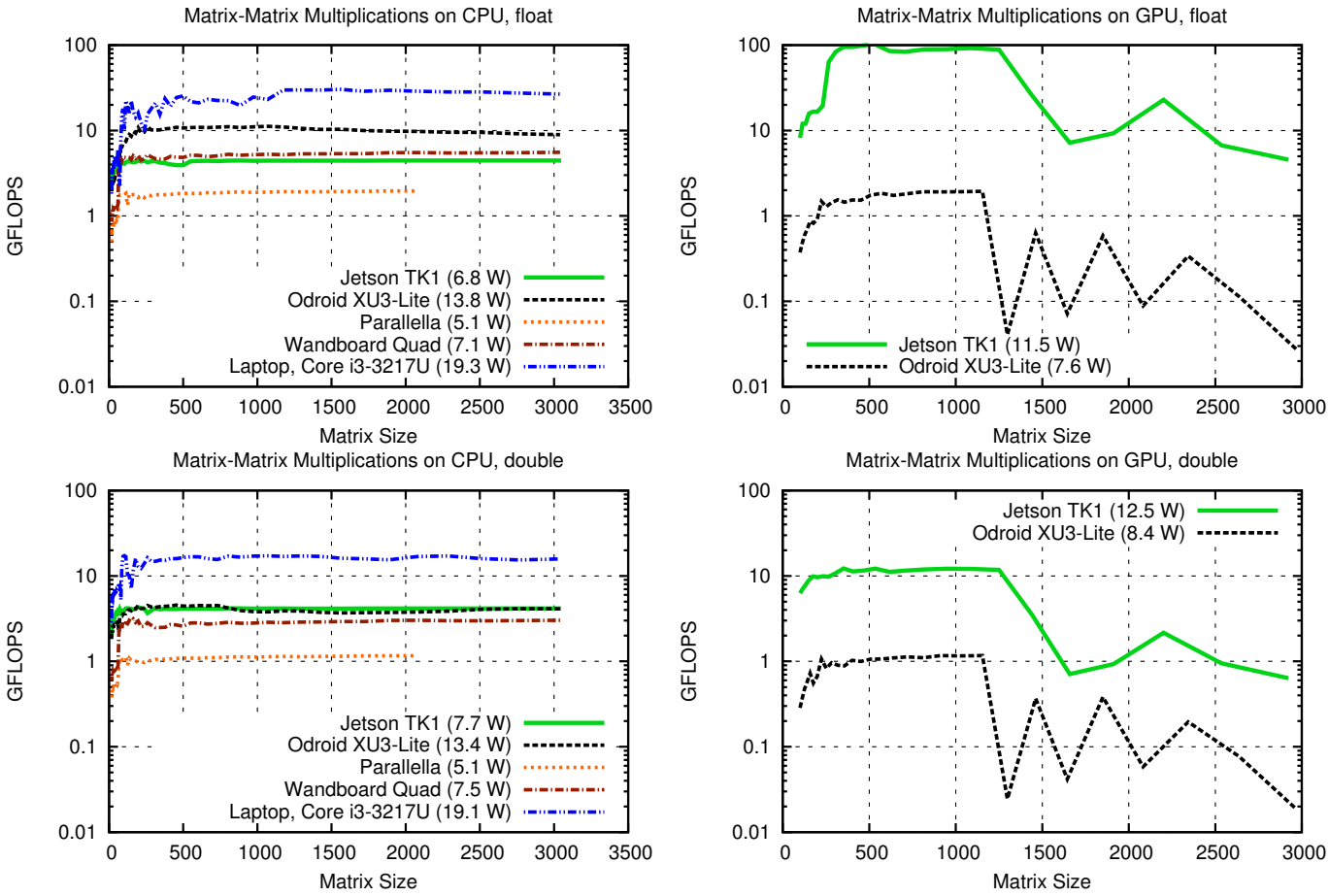

Figure 1. Performance for dense matrix-matrix multiplication in single precision (top) and double precision (bottom) for CPUs (left) and GPUs (right). The power draw during each of the runs is given in parenthesis.

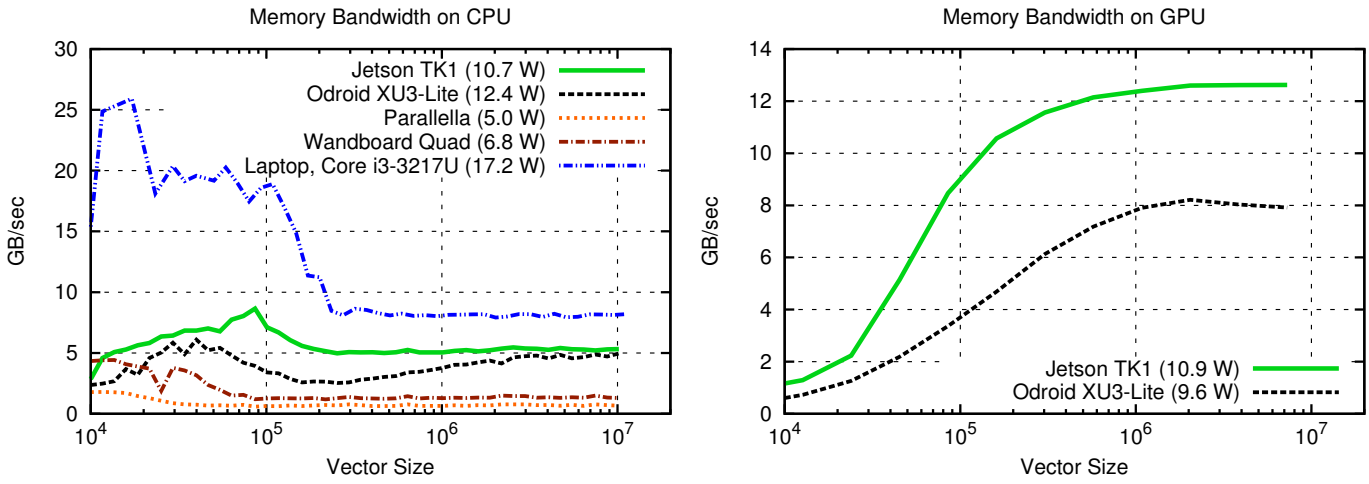

Figure 2. Memory bandwidth obtained for the STREAM benchmark using the CPU (left) and the GPU (right). The power draw during each of the runs is given in parenthesis.

OpenMP Benchmark	Jetson TK1	Odroid XU3-Lite	Parallella	Wandboard Quad	Core i3-3217U
PARALLEL	8.6	5.0	3.3	**2.3**	2.5
FOR	4.5	1.9	0.9	**0.7**	1.3
PARALLEL FOR	8.8	5.1	3.5	2.8	**2.5**
BARRIER	4.4	1.9	0.9	**0.6**	1.2
SINGLE	5.1	2.3	0.9	**0.6**	1.2
CRITICAL	0.9	0.5	0.29	**0.19**	0.3
LOCK/UNLOCK	**0.07**	0.6	0.28	0.19	0.3
ORDERED	**0.02**	1.0	0.7	0.5	1.3
ATOMIC	**0.04**	0.29	0.09	0.08	0.13
REDUCTION	**0.7**	5.1	3.4	2.3	2.6

Table 2. Time in microseconds for OpenMP thread synchronization for entering and leaving parallel regions as well as short reductions. The shortest execution time for each benchmark is printed in bold.

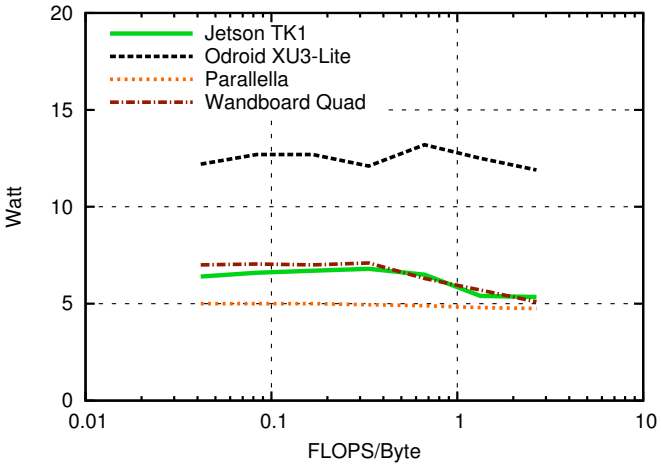

Figure 3. Power drain for CPU loads with different arithmetic intensities.

mance as well as other useful features for HPC. Even ultra low-cost boards such as the Raspberry Pi Zero for 5 USD do not offer a substantially better GFLOPS/USD ratio than current CPUs used in HPC. The new NEON vector unit in the new ARMv8 instruction set will improve energy efficiency in terms of GFLOPS/Watt, but this may not be enough to excel in HPC.

HPC clusters for applications limited by memory bandwidth may benefit from many SoCs with one memory channel each. Dual-, triple-, or quad-channel configurations in desktop and workstation machines come at a higher price tag than two, three, or four low-cost boards equipped with SoCs. However, current SoCs are unable to use the full memory bandwidth with their CPU. Instead, at most half of the bandwidth is obtained with Cortex-A15-based CPUs, whereas Cortex-A9-based CPUs only provide about a tenth of the theoretical peak performance. GPUs on mobile SoCs may not reach full memory bandwidth either, as the Odroid XU3-Lite demonstrates (Figure 2). On the other hand, the Jetson TK1 reaches full memory bandwidth only when using the GPU. Mobile SoCs may become very attractive for memory-bandwidth-limited tasks as soon as the CPU fully saturates the main memory channel. At the same time, memory bandwidth per Watt is also determined by the energy cost of moving data. The direct comparison with the x86-based laptop shows that the Jetson TK1 and the Odroid XU3-Lite offer the same memory bandwidth per Watt. This is a strong indication that new memory technology, such as on-chip high bandwidth memory, rather than a different processor architecture is needed to improve power efficiency.

The evaluation of latencies is positive: Mobile SoCs are already competitive with x86 processors in terms of thread synchronization cost. A direct comparison with an x86 processor in Table 2 shows latencies of mobile SoCs competitive with today's HPC machines. Still, an HPC system consisting of mobile SoCs is likely to suffer from higher overall network latencies, because the smaller raw performance on each node can only be compensated with a higher number of nodes. A higher number of nodes, however, inevitably leads to higher latency for global all-to-all communication.

Overall, the Jetson TK1 with its high compute power in single precision and the possibility to fully saturate the memory channel is the most attractive for use in HPC among the devices considered in our benchmark. However, its attractiveness stems from the scaled-down GPU rather than from the ARM-based CPU. Therefore, further improvements to ARM-based CPUs are necessary in order to gain any traction in the HPC landscape. At least some are addressed by the new ARMv8 instruction set, but it is not clear whether that is enough to become competitive for HPC without negative side-effects for the mobile sector.

6. CONCLUSION

In this work we evaluated the suitability of mobile SoCs for the HPC landscape. After considering practical performances obtained for raw floating point compute power, memory bandwidth, and latencies, we conclude that mobile SoCs are only competitive with respect to latencies. Even if the cost of additional networking infrastructure required for building a hypothetical cluster based on mobile SoCs is ignored, the CPUs on the SoCs still fall behind most existing systems in HPC.

The current migration to the new ARMv8 instruction set in mobile SoCs is expected to resolve limitations in memory capacity by transitioning to a 64 bit address space. Also, better double precision capabilities through the NEON vector instructions are expected. However, the lack of support for error correcting code (ECC) main memory will remain a significant hindrance for adoption in HPC. Nevertheless, ARM-based cores will remain an attractive option of chip makers who do not have access to an x86 license.

ACKNOWLEDGMENTS

Karl Rupp acknowledges support by the Austrian Science Fund (FWF), project P23598.

REFERENCES

1. Adams, M. F., Brown, J., Shalf, J., Van Straalen, B., Strohmaier, E., and Williams, S. HPGMG 1.0: A Benchmark for Ranking High Performance Computing Systems. Tech. Rep. LBNL-6630E, LBNL, 2014.
2. Bull, J. M., Reid, F., and McDonnell, N. A Microbenchmark Suite for OpenMP Tasks. In *Proc. International Conference on OpenMP in a Heterogeneous World (IWOMP)*, Springer (2012), 271–274.
3. Dongarra, J., Heroux, M. A., and Luszczek, P. High-Performance Conjugate-Gradient Benchmark: A New Metric for Ranking High-Performance Computing Systems. *International Journal of High Performance Computing Applications* (2015).
4. Göddeke, D., Komatitsch, D., Geveler, M., Ribbrock, D., Rajovic, N., Puzovic, N., and Ramirez, A. Energy Efficiency vs. Performance of the Numerical Solution of PDEs: An Application Study on a Low-Power ARM-based Cluster. *Journal of Computational Physics 237* (2013), 132 – 150.
5. Kramer, W. Top500 Versus Sustained Performance: The Top Problems with the Top500 List - and What to Do About Them. In *Proc. International Conference on Parallel Architectures and Compilation Techniques (PACT)*, ACM (2012), 223–230.
6. Liu, Y., Yang, C., Liu, F., Zhang, X., Lu, Y., Du, Y., Yang, C., Xie, M., and Liao, X. 623 Tflop/s HPCG run on Tianhe-2: Leveraging Millions of Hybrid Cores. *International Journal of High Performance Computing Applications* (2015).
7. Maqbool, J., Oh, S., and Fox, G. C. Evaluating ARM HPC Clusters for Scientific Workloads. *Concurrency and Computation: Practice and Experience 27*, 17 (2015), 5390–5410.
8. Mattson, T. G., and Henry, G. An Overview of the Intel TFLOPS Supercomputer. *Intel Technology Magazine 2* (1998).
9. McCalpin, J. D. Memory Bandwidth and Machine Balance in Current High Performance Computers. *IEEE Computer Society Technical Committee on Computer Architecture Newsletter* (1995), 19–25.
10. Rajovic, N., Carpenter, P. M., Gelado, I., Puzovic, N., Ramirez, A., and Valero, M. Supercomputing with Commodity CPUs: Are Mobile SoCs Ready for HPC? In *Proc. Supercomputing*, ACM (2013), 40:1–40:12.
11. Rajovic, N., Vilanova, L., Villavieja, C., Puzovic, N., and Ramirez, A. The Low Power Architecture Approach towards Exascale Computing. *Journal of Computational Science 4*, 6 (2013), 439 – 443.
12. Rupp, K., Rudolf, F., and Weinbub, J. ViennaCL - A High Level Linear Algebra Library for GPUs and Multi-Core CPUs. In *Proc. Intl. Workshop on GPUs and Scientific Applications (GPUScA)* (2010), 51–56.
13. Shaw, D. E., Deneroff, M. M., Dror, R. O., Kuskin, J. S., Larson, R. H., Salmon, J. K., Young, C., Batson, B., Bowers, K. J., Chao, J. C., Eastwood, M. P., Gagliardo, J., Grossman, J. P., Ho, C. R., Ierardi, D. J., Kolossváry, I., Klepeis, J. L., Layman, T., McLeavey, C., Moraes, M. A., Mueller, R., Priest, E. C., Shan, Y., Spengler, J., Theobald, M., Towles, B., and Wang, S. C. Anton, a Special-purpose Machine for Molecular Dynamics Simulation. *Communications of ACM 51*, 7 (2008), 91–97.
14. Shaw, D. E., Dror, R. O., Salmon, J. K., Grossman, J. P., Mackenzie, K. M., Bank, J. A., Young, C., Deneroff, M. M., Batson, B., Bowers, K. J., Chow, E., Eastwood, M. P., Ierardi, D. J., Klepeis, J. L., Kuskin, J. S., Larson, R. H., Lindorff-Larsen, K., Maragakis, P., Moraes, M. A., Piana, S., Shan, Y., and Towles, B. Millisecond-scale Molecular Dynamics Simulations on Anton. In *Proc. Supercomputing*, ACM (2009), 65:1–65:11.
15. Stone, J. E., Gohara, D., and Shi, G. OpenCL: A Parallel Programming Standard for Heterogeneous Computing Systems. *IEEE Design & Test 12*, 3 (2010), 66–73.
16. Tillet, P., Rupp, K., Selberherr, S., and Lin, C.-T. Towards Performance-Portable, Scalable, and Convenient Linear Algebra. In *Proc. USENIX Workshop on Hot Topics in Parallelism* (2013), 1–8.
17. Turley, J. Cortex A-15 "Eagle" Flies the Coop. *Microprocessor Report 24* (2010), 1–11.
18. Varghese, A., Edwards, B., Mitra, G., and Rendell, A. P. Programming the Adapteva Epiphany 64-Core Network-on-Chip Coprocessor. In *Proc. IEEE International Parallel & Distributed Processing Symposium Workshops (IPDPSW)*, IEEE Computer Society (2014), 984–992.
19. Wang, Q., Zhang, X., Zhang, Y., and Yi, Q. AUGEM: Automatically Generate High Performance Dense Linear Algebra Kernels on x86 CPUs. In *Proc. Supercomputing*, ACM (2013), 25:1–25:12.
20. Young, C., Bank, J. A., Dror, R. O., Grossman, J. P., Salmon, J. K., and Shaw, D. E. A 32x32x32, Spatially Distributed 3D FFT in Four Microseconds on Anton. In *Proc. Supercomputing*, ACM (2009), 23:1–23:11.