Trends in Exascale Computing Software for Low Power

Taesang Huh, Jae-Hyuck Kwak, Myoungju Koh, and Seungwoo Rho

Abstract—The demand for low-power HPC in exascale computing is a top priority, and various improvements are being considered to reduce the clock frequency and improve the cost of data movement in an aspect of energy consumption and performance to save power. Although exascale computing is predicted to be created with a new architecture different from previous HW architectures, we focus on SW research trends for low power in achieving an exascale system in this paper.

Research Keywords—Exasacale Computing, Low Power Software, Auto-tuning, Dynamic Voltage Scaling

1 INTRODUCTION

Exascale computing, which requires low-power HPC, is not just 1,000 times the current peta-class architecture. The processor architecture has yet to be determined, and it will reach 150 MW in a simple scaling-up of the Sunway Taihulight petaflop computer in China, which has the highest performance today. US DARPA report, DOE (Department of Energy) report and Exascale Computing Project (ECP) aim at 20-30MW with the reasonable power of exaflops computer, and hardware architecture changes seem to be necessary for this [1]-[3].

To conserve power, the clock frequency will decrease and consequently, the number of processors on a single chip is expected to increase, and the exascale architecture will have a high concurrency. In terms of energy consumption and performance, data migration costs are not expected to improve as much as the cost of floating point operations, so algorithms need a programming model that minimizes data movement and allows power management. At all levels, it will be more difficult to handle the I/O systems (chip to memory, memory to I/O node, and I/O node to disk) because it cannot keep pace with the I/O bandwidth and machine speed.

Reliability and resiliency are mandatory for one billion concurrent scales. The silent error due to components and manufacturing variability failures will have a greater impact on the computational results of exascale computers than today's petascale computers.

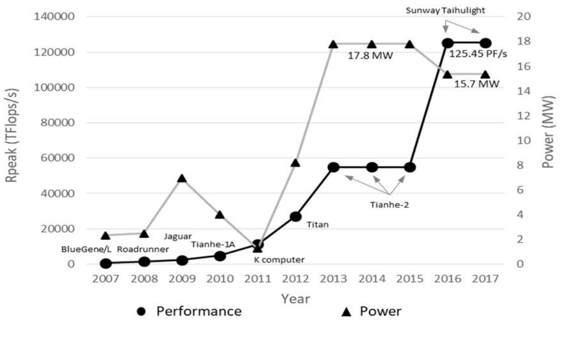

Fig. 1. Performance and energy efficiency of rank 1 systems on the top500 list (2007-2017)

Figure 1 shows Top500 trends in performance and energy efficiency of rank 1 systems for the past 10 years [4]. The 2017 rank 1 system, Sunway Taihulight, has significantly improved performance and power efficiency over the previous tianhe-2. But in order to achieve exascale computing, it is es-

- *Taesang Huh is with National Institute of Supercomputing Networking, KISTI, Yuseong-gu, Daejeon, South Korea. E-mail: tshuh@kisti.re.kr*
- *Jae-Hyuck Kwak is with National Institute of Supercomputing Networking,, KISTI, Yuseong-gu, Daejeon, South Korea. E-mail: jhkwak@kisti.re.kr*
- *Myoungju Koh is with National Institute of Supercomputing Networking, KISTI, Yuseong-gu, Daejeon, South Korea. E-mail: myju@kisti.re.kr*
- *Seungwoo Rho (corresponding author) is with National Institute of Supercomputing Networking, KISTI, Yuseong-gu, Daejeon, South Korea. E-mail: seungwoo0926@kisti.re.kr*

sential to improve performance and energy efficiency with the new and improved HW architecture and advanced SW supporting it.

Low power for exascale computing is necessary not only for HW development but also for SW development. This article discusses the research trends focusing on the SW of an exascale system for low power computing.

2 Software For Exascale system

The scope of software development in the exascale system includes the operating system (runtimes for scheduling, memory management, communication, performance monitoring, power management, and resiliency), computational library, compiler, programming language, and application framework. Countries around the world are actively enhancing SW for the exascale system by establishing roadmaps for each country.

2.1 Software Requirements for Exascale Computing

There are some indispensable SW issues to consider for exascale computing as follows:

- A scalable system with superior performance over existing supercomputers (operation rate: 6 times or more),
- 1 exaflops performance under 20MW~30MW power consumption
- Availability based on checkpoint and restart with guaranteed downtime within one week per year
- Compatible architecture to cover a wide range of workloads.

2.2 Exascale Computing Software

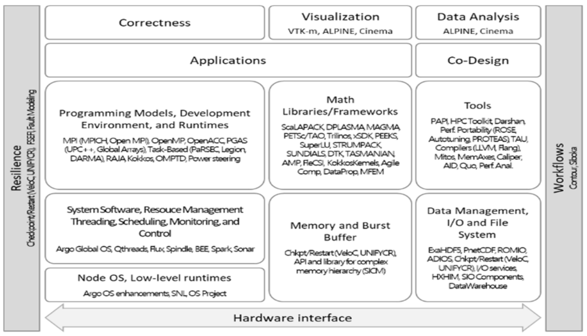
Correctness
Visualization
VTK-m, ALPINE, Cinema
Data Analysis
ALPINE, Cinema
Applications
Co-Design
Programming Models, Development Environment, and Runtimes
MPI (MPICH, Open MPI), OpenMP, OpenACC, PGAS (UPC++, Global Arrays), Task-Based (PaRSEC, Legion, DARMA), RAJA, Kokkos, OMPTD, Power steering
Math Libraries/Frameworks
ScaLAPACK, DPLASMA, MAGMA, PETSc/TAO, Trilinos, xSDK, PEEKS, SuperLU, STRUMPACK, SUNDIALS, DTK, TASMANIAN, AMP, FleCSI, KokkosKernels, Agile Comp, DataProp, MFEM
Tools
PAPI, HPC Toolkit, Darshan, Perf. Portability (ROSE, Autotuning, PROTEAS), TAU, Compilers (LLVM, Flang), Mitos, MemAxes, Caliper, AID, Quo, Perf. Anal.
System Software, Resouce Management Threading, Scheduling, Monitoring, and Control
Argo Global OS, Qthreads, Flux, Spindle, BEE, Spark, Sonar
Memory and Burst Buffer
Chkpt/Restart (VeloC, UNIFYCR), API and library for complex memory hierarchy (SICM)
Data Management, I/O and File System
ExaHDF5, PnetCDF, ROMIO, ADIOS, Chkpt/Restart (VeloC, UNIFYCR), I/O services, HXHIM, SIO Components, DataWarehouse
Node OS, Low-level runtimes
Argo OS enhancements, SNL OS Project
Hardware interface
Resilience
Checkpoint/Restart (VeloC, UNIFYCR), FSEFI, Fault Modeling
Workflows
Contour, Sibola

Fig. 2. Exascale Computing SW Stack by ECP

Exascale computing software is aimed at using highly parallel applications based on the exascale architecture. It is well described in figure 2 exascale computing software stack and should reflect the following application requirements:

- Programming model like task-based RTS and concurrency from applications
- Integrating SWs using different approaches such as on-node parallelism
- Development and integration of co-designed topic-based community components
- Coupling both algorithms and SWs
- Integration of sensitivity analysis, data assimilation and quantification techniques of uncertainty
- Understanding the requirements of data analytic computing methods and applications [5].

3 Trend of software research for low power computing

3.1 Compiler Algorithm for Energy Reduction

A compiler is an implementation of architecture-independent algorithms that can map to high-quality code and a lightweight mechanism that interfaces with runtime layers and architectures. For example, dynamically map specific execution contexts to code to improve high performance and power efficiency [6]. The research challenges in optimization are to make parallel programming mainstream, to write compilers capable of self-improvement, and support optimization for parallel code.

3.2 Dynamic Voltage Scaling (DVS)

To reduce CPU power and increase energy efficiency, the DVS algorithm is being studied to adjust CPU frequency after CPU-boundedness detection so that the program can run on non-peak CPU frequency. For programs where the execution time is limited to the performance of the peripheral rather than the CPU speed, applying DVS to the program can contribute to reducing power consumption [7].

There are the research challenges in optimization as follows:

- Source code instrumentation for performance profiling
- Execution with profiling
- Determination of appropriate processor frequencies for each phase
- Source code instrumentation for DVS scheduling.

3.3 Auto-tuning

Since meeting the real-time simulation requirements of up to 1,000 times per core is a difficult task in a power-efficient design, an automatic optimization framework is being developed to optimize the code and reduce the computational burden, and the automatic tuner is used to systematically apply compiler optimizations [8]. The main ideas of auto-tuning is to achieve to optimize memory access for many-core CPUs, Multi-core CPUs and GPUs. Auto-tuning gives us code generation for optimization candidate, search functions for the best candidate and automatic execution for optimization. Through automated code tuning/compiler/runtime optimization, refinement loops can enhance not only but also power efficiency.

3.4 Hardware-Software Co-design

Energy efficiency could be improved by tuning the hardware according to the auto-tuning software. In addition, using the SW-HW optimization combination within the system design cycle, it enables both system validation and development time reduction [9]][8].

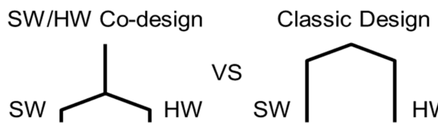

Fig. 3 Concurrent Engineering Techniques in Designing Complex SW/HW Products

Co-design enables to meet time-to-market since developed software can be verified much earlier and overall system performance, reliability, and cost effectiveness and defects found in hardware can be corrected before its implementation, compared with classic design in figure 3.

4 Conclusions

The compilation algorithms and DVS for low-power exascale computing introduced above can contribute to reducing the power consumption of the operation. The RAMP (Research Accelerator for Multiple Processors), which is An FPGA-based architecture simulator for multiprocessors, allows for near-real-time prototype development and testing. With this function, it is possible to test the application code and advanced software development used in the system, and exascale SW-HW auto-tuning to improve the performance of existing SW and HW as well as the design of low power computing.

Acknowledgment

This work is supported by the KISTI (K-18-L01-C01-S01) and the IITP (Grant No. R0190-15-2012).

References

[1] P. Messina, "The Exascale Computing Project," Computing in Science & Engineering, vol. 19, no.3, pp. 63-67, 2017.
[2] M. Lee, “World-wide trend analyses for building exascale supercomputer and suggested strategy for Korea,” CKIISE, vol.34, no.2, pp. 9-21, 2016
[3] P. Messina, “The U.S. D.O.E. Exascale Computing Project – Goals and Challenges,” available at https://www.nist.gov, 2018.
[4] Top500 list statistics, available at http://www.top500.org, 2018.
[5] ECP, Exascale Computing Project, available at: http://www.exascaleproject.org. 2017.
[6] R. Nobre, L. Reis, and J.M. Cardoso, "Compiler phase ordering as an orthogonal approach for reducing energy consumption," Proc. 19th Workshop on Compilers for Parallel Computing, 2016.
[7] N. Parihar, N. Goel, A. Chaudhary, and S. Mahapatra, "A modeling framework for NBTI degradation under dynamic voltage and frequency scaling," IEEE Electron Devices, vol. 63, no.3, pp. 946-953, 2016.
[8] D. Brayford, C. Bernau, C. Guillen, and C. Navarrete, “Node level power measurements on a petaflop system,” HPEC2016, pp. 1-6, 2016.
[9] V. Cavé, R. Clédat, P. Griffin, A. More, B. Seshasayee, S. Borkar, and J. Fryman, "Traleika Glacier: A hardware-software co-designed approach to exascale computing," Parallel Computing64, pp. 33-49, 2017.

Taesang Huh received the B.S. degree in Electric, Electronic and Computer Engineering from Sungkyunkwan University in 2000 and the M.S. degree in Information and Communications Engineering from Gwangju Institute of Science and Technology in 2002, respectively. In 2002, he has joined KISTI and works as a senior researcher of NISN at KISTI. He also received the Ph. D degree in Computer Engineering from Paichai University in 2017. His research interests include Metadata Catalog, Distributed Computing, Cloud Storage, e-science and information system.

Jae-Hyuck Kwak received the B.S. degree in Information and Computer Engineering from Ajou University in 2001 and the M.S. degree in Electrical and Computer Engineering from Seoul National University in 2003. He joined KISTI in 2003. Currently, he is a senior researcher of NISN at KISTI. His research interests lie primarily in the issues concerning High Performance Computing, Distributed Computing Technology, and Data-intensive Computing.

Myoungju Koh received the B.S. degree in Systems Management Engineering from Sungkyunkwan University (SKKU), in 2005 and M.S degree and Ph. D. degree in Industrial Engineering from SKKU in 2007 and 2014, respectively. She worked in Science and Technology Policy Institute (STEPI) for six years from 2009. In 2015, she has joined Korea Institute of Science and Technology Information (KISTI) and has been a senior researcher at KISTI. Her current research interests include research policy and strategy management of Korea National Supercomputing Center.

Seungwoo Rho received the B.S., M.S in electrical and computer engineering from University of Seoul, Korea in 2009, 2011 respectively. Since then, he has been with NISN at KISTI. His main research interests include supercomputer, cloud, distributed computing, and grid computing.