Fast, Accurate and Distributed Simulation of novel HPC systems incorporating ARM and RISC-V CPUs

Nikolaos Tampouratzis
Exascale Performance Systems - EXAPSYS Plc
Heraklion, Greece
tampouratzis@exapsys.eu

Ioannis Papaefstathiou
Exascale Performance Systems - EXAPSYS Plc
Heraklion, Greece
ygp@exapsys.eu

ABSTRACT

The growing developments of HPC systems used in a plethora of domains (healthcare, financial services, government and defense, energy) triggers an urgent demand for simulation frameworks that can simulate, in an integrated manner, both processing and network components of an HPC system-under-design (SuD). The main problem, however, is that, currently, there is a shortage of simulation frameworks that can handle the simulation of actual HPC systems, including the hardware, complete software stack and network dynamics in an integrated manner. In this work we start from the first known, open-source, fully-distributed Cloud simulation framework, COSSIM, and, as part of the RED-SEA[1] and Vitamin-V[2] European projects, we extend it so as to be able to accurately simulate HPC systems. The extended simulator has been evaluated when executing the very-widely used HPCG & LAMMPS benchmarks on both ARM & RISC-V architectures; the results demonstrate that the presented approach has up to 95% accuracy in the reported SuD aspects.

KEYWORDS

HPC Simulator, Distributed Systems Simulator, ARM, RISC-V

ACM Reference Format:
Nikolaos Tampouratzis and Ioannis Papaefstathiou. 2024. Fast, Accurate and Distributed Simulation of novel HPC systems incorporating ARM and RISC-V CPUs. In *The 33rd International Symposium on High-Performance Parallel and Distributed Computing (HPDC '24), June 3–7, 2024, Pisa, Italy.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3625549.3658816

1 INTRODUCTION

Nowadays HPC systems encompass a wide range of processing units ranging from well-known X86/ARM to modern RISC-V architectures interconnected through multiple networks. A significant challenge encountered by designers of such systems is the lack of simulation tools capable of providing comprehensive insights beyond basic functional testing. Such insights include the actual performance of the HPC nodes, accurate overall system timing, power/energy estimations, and network deployment issues. However, given the very high complexity and heterogeneity of today's HPC systems, simulating independently only parts, components, or attributes of such a system is a cumbersome, inaccurate, and inefficient approach.

In this paper, we present several extensions of an open-source framework (i.e. COSSIM [1]) so as to overcome the aforementioned constraints, that have been designed and evaluated in the framework of the RED-SEA and Vitamin-V European Projects. The COSSIM framework effectively incorporates a collection of sub-tools that simulate the computing devices of the processing nodes, as well as the network(s) of the parallel systems. It provides cycle-accurate results by simulating the actual parallel application and system software on the target hardware nodes together with the selected network topologies. In this paper we present the extension of COSSIM so as to support modern ARM and RISC-V architectures enabling the designer to accurately and efficiently simulating complete MPI applications on heterogeneous HPC systems.

The main contributions of this paper are:

- The first known open-source[3] extended simulation framework which can simulate complete heterogeneous HPC Systems supporting ARM and **RISC-V** CPUs with PCI devices.
- An innovative flow to enable the designers to simulate the complete aspects of HPC Systems (i.e. CPU and Network Environment) when executing real MPI applications coupled with open-source reference designs.
- A thorough evaluation of the end simulation system based on the very widely used HPCG & LAMMPS benchmarks.

2 RELATED WORK

Several simulators for HPC systems have been proposed each with its own approach and focus area.

Trace-driven simulators like SIM-MPI [2] and LogGOPSim [3] model computation and communication operations to predict performance, while execution-driven simulators such as PS-SIM [4] and SimHPC [5] execute MPI programs on scaled-down real clusters to simulate target systems. However, their main drawback is that they all collect the computation traces on existing hardware nodes that are the same as the SuDs, thus they cannot be utilized when the SuD comprises of different/new CPUs.

Additionally, the Structural Simulation Toolkit (SST) and the integration of Gem5 with SST [6] represent advancements in parallel simulation and cycle-accurate modeling, respectively. Other simulators, such as [7], focus on real network simulations using OMNeT++/ OMNEST, while making restrictive assumptions for the computational models. Our approach goes beyond those approaches

[1] https://www.redsea-project.eu/
[2] https://www.vitamin-v.eu/

[3] https://github.com/ntampouratzis/RED-SEA

HPDC '24, June 3–7, 2024, Pisa, Italy

ACM ISBN 979-8-4007-0413-0/24/06…$15.00
https://doi.org/10.1145/3625549.3658816

by combining successfully the simulation of both processing and network aspects of HPC applications in one framework with the same notion of time, while providing the unique capability to model non-implemented CPUs based on the RISC-V Instruction Set.

There are also several simulators [8], [9], [10] which can simulate multi-core X86, ARM and RISC-V machines with different Network-on-Chip (NoC) configurations. However all these simulators can only model multi-core devices (i.e. they model NoCs and not external networks) and not interconnected parallel distributed systems of systems, such as the ones in the HPC domain.

Finally, authors in [11] propose an approach to simulate HPC systems by simulating only 1 node using an execution-driven full system simulator and a message emulation environment. The main drawback of this approach is the huge and difficultly handled traces generated among the stand-alone simulators and the lack of synchronization and data exchange mechanisms. In addition, it requires pre-execution of the target HPC application in an existing HPC system, while they target only ARM-based systems.

3 COSSIM EXTENSIONS

Figure 1 illustrates the main components of the COSSIM simulator and its interfaces and inputs/outputs. Multiple instances of a node simulator module (i.e. a GEM5-based module) are utilized for the efficient simulation of the processing nodes of a parallel system; each simulator node can run in a different physical core (either in the same CPU or in different distributed CPUs), making the simulator fully scalable. The network that binds together the different simulated nodes is modeled in network simulation modules (i.e. OMNET++ based modules). The initial version of COSSIM supports ARM multicore processors, while in this work we extend COSSIM to support multicore RISC-V processors connected to PCI devices.

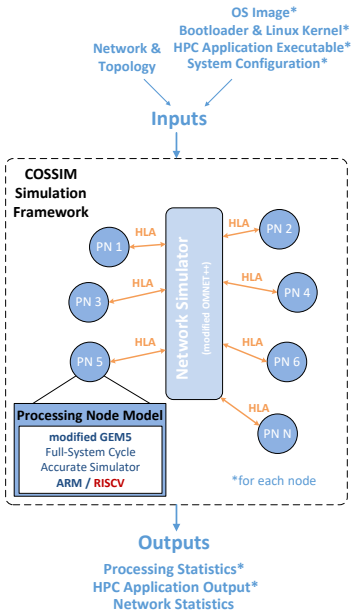

Figure 1: COSSIM Framework Extensions

3.1 Extending RISC-V Model of GEM5

The main goal of our work was to first extend the baseline GEM5 RISC-V system to support PCI interconnections so as to be able to simulate any PCI-compatible device. The hardware configuration developed, as an extension of the RISC-V GEM5 platform, is depicted in Figure 2. We utilize the bus subsystem, CPU, HiFive Platform and RISCVBoard from the conventional GEM5 and we added the modules which are denoted by the orange boxes.

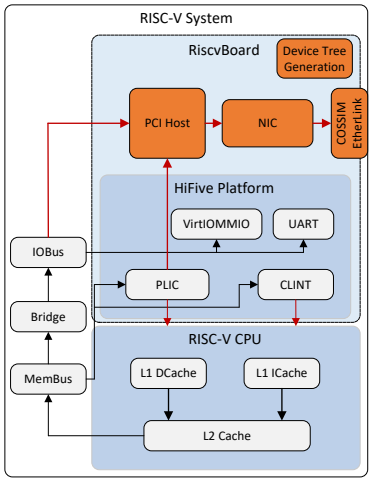

Figure 2: PCI Integration in RISC-V Model

In more detail, in GEM5, system configurations are organized into container classes called platforms. A Platform class is a hierarchical extension of a parent class that provides a standardized set of utility functions and peripherals that can be used to tailor the configuration to a particular board or system. For RISC-V, the HiFive platform has recently been introduced, within GEM5, which aligns with SiFive's HiFive series of boards. In our work, and in order to be able to evaluate the accuracy of our approach as demonstrated in the results' section, the memory map conventions and peripheral addresses are fully inline with the specifications outlined in the GEM5 RISCVMatchedBoard which is based on SiFive Freedom U740 multi-core processor. However, our work can easily be extended and be used by other SoCs as long as the memory map and peripheral address tables are altered accordingly.

In order to support multiple interconnected RISC-V nodes, we added a PCI host model which is attached to the generic RISC-V system. Specifically, we implemented a GenericRiscvPciHost object that inherits the methods of the base GEM5 GenericPciHost object. The main purpose of the custom PCI host is to map the correct source interrupt number from the PCI host to the RISC-V Platform-Level Interrupt Controller (PLIC). In our implementation, the base interrupt source number is added to the PCI interrupt number; thus, the base source number for the PCI host is 0x10, and the source numbers for the PCI interrupts are 0x11, 0x12, 0x13 and 0x14.

Due to the absence of distinction, within GEM5, between requests pertaining to configuration, I/O, and memory space, it becomes necessary to allocate distinct configuration, I/O, and memory ranges for the simulated PCI devices. In our implementation, we used the address ranges from the widely-used ARM Vexpress GEM5 V1 platform; 256MB (address range 0x30000000 - 0x3fffffff), 16MB (address range 0x2f000000 - 0x2fffffff), and 1024MB (address range 0x40000000- 0x7fffffff), are assigned for the PCI configuration space, I/O space, and Memory space, respectively. DRAM is mapped from 2GB to 512GB while all PCI memory regions are uncashable.

Another challenge that needed to be addressed was the right generation of the Device Tree so that the operating system's kernel can use and manage the newly added PCI component.

In order to provide a more easily used RISC-V simulation environment we configured an (OpenSBI)[12] bootloader and a Linux kernel binary that works with GEM5 full system simulations; as a result the bootloader and the kernel binaries are completely independent. OpenSBI is a reference implementation of RISC-V SBI (Supervisor Binary Interface), and it can act as a first-stage bootloader setting up the environment before jumping to the start of the main runtime/OS, a Linux kernel in this case. In our configuration, after the first stage booting is done by OpenSBI, it will jump to the instruction in which the Linux kernel is located, and then the actual simulation starts.

The only network interface card that has been implemented, tested, and verified in the publicly accessible repositories of GEM5 is an Ethernet adapter based on the Intel 8254x. It is provided as a PCI GEM5 network device using the e1000 Linux driver. In order to extend our base simulation framework, we modified GEM5 so as to support the Intel 8254x network card and thus allowing RISC-V CPUs' PIO ports for PCI devices to be connected to the master ports of the IOBus while the DMA ports can be connected to the slave ports of the IOBus master.

3.2 HPC Run-Time Environmnet

In order to efficiently support real HPC applications, our extended system can simulates parallel application/program in combination with any OS including the required libraries (e.g. MPI). In order to facilitate the use of our simulator by any interesting HPC stakeholder, Ubuntu 22.04 LTS GEM5 compatible images, utilizing Linux Kernels v6.5.5 and bootloaders, have been created and configured for both ARM and RISC-V architectures and are available in open-source.

In order to fully simulate a parallel program/application on a network of computers via MPI, an MPI Cluster is set up. In this respect, the rsh server has also been successfully imported into all GEM5 nodes so as to allow the launching of commands on remote nodes. In addition, MPICH v4.0 has been imported and successfully evaluated using both simple and collective communication MPI routines, while a *hosts* file (/etc/hosts) is configured by each GEM5 node OS so as to map hostnames to IP addresses. Finally, a *host_file* is created to declare the number of MPI-ranks that will be executed in each COSSIM node. All those configurations are also available in open-source together with the corresponding instructions.

Code Segment 1 presents the MPI run-time environment for the HPCG execution on 3 gem5 nodes using 16 MPI ranks per node. The user can modify the model of both the computing node and the interconnection network of the target HPC system using COSSIM's user friendly graphical interface. Finally, since COSSIM supports full Ubuntu simulated OS, the user may add any cloud-based HPC mechanisms (such as SLURM/REST interfaces) in order to determine the number and type of nodes assigned to a submitted job.

4 VALIDATION AND PERFORMANCE ANALYSIS

This section presents the experimental results that validate the accuracy and scalability of the extended COSSIM simulator using

Code Segment 1 MPI run-time environment for 3 gem5 nodes

```
1: hostname node0 #Define the hostname for node0 (localhost)
2: rsh 192.168.0.3 hostname node1 #Define the hostname for node1
3: rsh 192.168.0.4 hostname node2 #Define the hostname for node2
4: echo "node0:16 node1:16 node2:16" >> host_file
5: m5 resetstats #reset the gem5 statistics before mpi execution
6: #Run the HPCG benchmark on 3 nodes (16 MPI ranks/node)
7: mpirun -launcher rsh -n 48 -f host_file ./hpcg
8: m5 dumpstats #dump the gem5 statistics
```

the widely-used HPCG and LAMMPS [13] benchmarks. In all our experiments, we executed our simulations in a server with an AMD Ryzen 9 7950X @ 4.50GHz and 128GB of RAM while we have used the exact same configurations and compiler optimizations for all the experiments involving ARM and RISC-V CPUs.

4.1 COSSIM Accuracy

The widely used HPCG v3.1 benchmark has been fully ported on RISC-V and ARM CPUs and we compare the results reported by our simulation framework with those actually measured in two real systems: Dibona-TX2 cluster [14] comprising of ARM CPUs and HiFive Unmatched board [15] that contain SiFive Freedom U740 RISC-V SoC.

First of all, we configured the extended COSSIM to simulate from 2 up to 64 ARM cores in each GEM5 instance using the Armv8 processor ISA clocked at 2GHz & with DDR4 memory so as to perfectly match the characteristics of the Dibona Cluster. In Figure 3 we can see the HPCG performance comparison as reported by the extended COSSIM simulator and as measured in the Dibona Cluster. In the left y-axis there are the GFLOPS (reported by the simulator and measured), while in the right y-axis we can see the performance deviation and accuracy error; the performance difference (e.g. deviation) between the GFLOPs reported by the extended COSSIM simulator and those actually measured in the Dibona Cluster is below 5%, while the accuracy error is zero.

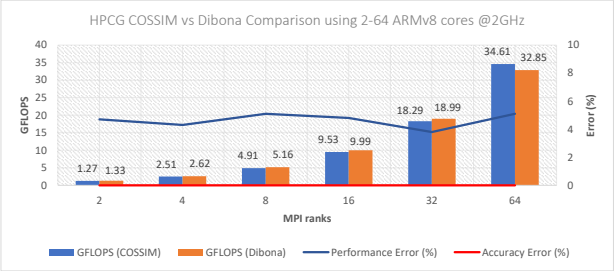

Figure 3: Simulation Results for 2-64 ARMv8 cores

In addition, we configured the extended COSSIM to simulate 4 RISC-V cores, in each GEM5 instance, with the exact same architectural characteristics as those of SiFive's U740 system. In Figure 4 there are the GFLOPs reported by the simulator and those triggered in SiFive's system. As we can see the performance deviation is below 9% (8.3%) for 4 MPI ranks (U740 is equipped with a 4-core RISC-V CPU), while the accuracy error is zero. In both cases (1st and 2nd column) an unoptimized version of the HPCG is used, while we achieve 2.7x higher performance (0.504 GFLOPS), when we applied several compiler optimizations while the deviation remains virtually the same.

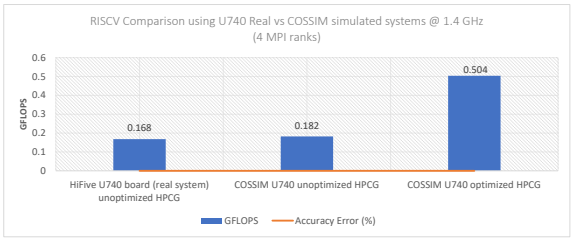

Figure 4: COSSIM comparison using 4 SiFive U740 cores

4.2 COSSIM Scalability

Apart from the widely-used HPCG benchmark, we also ported the LAMMPS benchmark, which is developed and updated in the context of the RED-SEA European project, to the extended COSSIM simulator for both ARM & RISCV architectures, in order to measure and verify the COSSIM scalability on multiple nodes.

We configure each COSSIM node to simulate from 2 to 8 ARM & RISC-V cores, with the same architectural characteristics (e.g clocked at 1.4 GHz with 16 GB DDR4 memory, 32 kB L1 instruction and data caches and 2MB L2 cache size), in order to compare the reported performance. Moreover, for our experiments we created a star network topology in OMNET++, characterized by a central node (Node0) that is connected to all other nodes through an ethernet switch, thereby ensuring efficient and centralized management of the whole simulation environment from Node0. In all LAMMPS experiments 10 steps with 32000 atoms have been used.

Figures 5 and 6 present the Performance reported when simulating the LAMMPS benchmark on the RISC-V and ARM CPUs from 1 to 16 nodes. As we can see the ARM processor outperforms by approximately 140% the RISC-V processor in the single node setup, in all ranks, while the accuracy difference is zero. Moving to multiple CPU nodes (Figure 6) implemented in multiple distributed COSSIM nodes, the performance gap for ARM architecture increases to even 262%. The main reason for the performance difference among the architectures is that the basic RISC-V ISA is less efficient than the ARMv8 one when executing HPC benchmarks/workloads. It should be stressed that due to the fully distributed feature of COSSIM, the simulation time when simulating 16 nodes is only 15% higher than in the case of the single node simulation, clearly demonstrating the high scalability of our approach.

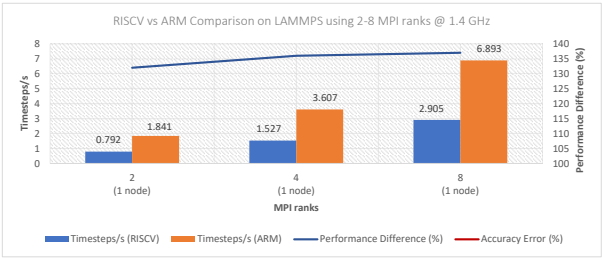

Figure 5: RISCV vs ARM Performance using 2-8 MPI ranks

5 CONCLUSIONS

In this paper we present several extensions of an open-source integrated simulation framework which allows for the accurate simulations of complete heterogeneous HPC Systems comprising of ARM and RISC-V CPUs and different intercommunication network topologies. Specifically, we extended the COSSIM simulator to support PCI devices interconnected to RISC-V cores and created an HPC run-time environment allowing for the seamlessly simulation

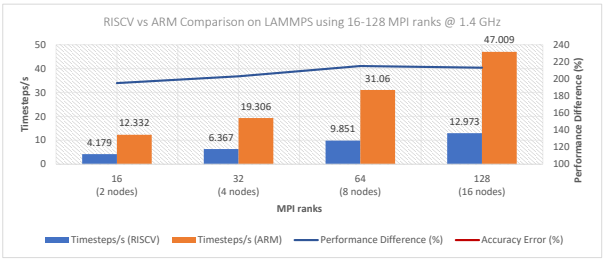

Figure 6: RISCV vs ARM Performance using 16-128 MPI ranks

of parallel multi-node ARM and RISC-V systems. Our approach enables the designers to simulate the complete aspects of HPC Systems (i.e. CPU and Network Technologies and Features) using real, full MPI applications within a single simulation framework with zero functional accuracy error. Moreover, the complete extended simulation framework as well as all the configurations (e.g. OS, libraries etc) are distributed in open-source while, as demonstrated the presented simulator produces very accurate performance results when simulating widely used HPC benchmarks.

ACKNOWLEDGMENTS

This work is supported by the RED-SEA and Vitamin-V EuroHPC and Horizon Europe projects under grant agreements No 955776 and No 101093062 respectively.

REFERENCES

[1] Nikolaos et.al. Tampouratzis. A novel, highly integrated simulator for parallel and distributed systems. *ACM Trans. Archit. Code Optim.*, 17(1), mar 2020.
[2] Jidong Zhai, Wenguang Chen, Weimin Zheng, and Keqin Li. Performance prediction for large-scale parallel applications using representative replay. *IEEE Transactions on Computers*, 65(7):2184–2198, 2016.
[3] Torsten Hoefler, Timo Schneider, and Andrew Lumsdaine. LogGOPSim - Simulating Large-Scale Applications in the LogGOPS Model. In *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing*, pages 597–604. ACM, Jun. 2010.
[4] Rajesh P. et.al Barnwal. Ps-sim: A framework for scalable simulation of participatory sensing data. In *2018 IEEE International Conference on Smart Computing (SMARTCOMP)*, 2018.
[5] Yi Liu, Yu-Zhe Zhi, Xin Zhang, He Li, and Lin et. al. Jiao. Simhpc: An execution-driven simulator for high-performance computers. *Chinese Journal of Computers*, 36:738–746, 04 2014.
[6] Mingyu et.al Hsieh. Sst + gem5 = a scalable simulation infrastructure for high performance computing. pages 196–201, 03 2012.
[7] Cyriel et.al. Minkenberg. *End-to-End Modeling and Simulation of High- Performance Computing Systems*, pages 201–240. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.
[8] T. E. Carlson, W. Heirman, and L. Eeckhout. Sniper: Exploring the level of abstraction for scalable and accurate parallel multi-core simulation. In *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, Nov 2011.
[9] Lilia Zaourar, Mohamed Benazouz, Ayoub Mouhagir, Fatma Jebali, and Sassolas. Multilevel simulation-based co-design of next generation hpc microprocessors. In *2021 International Workshop on Performance Modeling, Benchmarking and Simulation of High Performance Computer Systems (PMBS)*, pages 18–29, 2021.
[10] Boria Perez, Alexander Fell, and John D. Davis. Coyote: An open source simulation tool to enable risc- v in hpc. In *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 130–135, 2021.
[11] Fang Lin, Yi Liu, Xin Wang, and Xueyan Gai. Leveraging simulation of high performance computing systems with node simulation using architecture simulator. *CCF Transactions on High Performance Computing*, 5, 11 2023.
[12] Risc-v open source supervisor binary interface (opensbi). https://github.com/riscv-software-src/opensbi, 2023.
[13] Aidan P. Thompson, H. Metin Aktulga, and Richard et.al. Berger. LAMMPS - a flexible simulation tool for particle-based materials modeling at the atomic, meso, and continuum scales. *Computer Physics Communications*, 271, February 2022.
[14] F. Banchelli, M. Garcia, M. Josep, F. Mantovani, and J. Morillo et.al. MB3 D6.9 – Performance analysis of applications and mini-applications and benchmarking on the project test platforms. Technical report, 2019.
[15] Hifive unmatched board. https://www.sifive.com/boards/hifive-unmatched.