Power Efficiency in HEP (x86 vs. arm)

Emanuele Simili, Gordon Stewart, Samuel Skipsey, Dwayne Spiteri, David Britton

School of Physics and Astronomy, University of Glasgow
Kelvin Building, University Avenue, G12 8QQ, United Kingdom

E-mail: emanuele.simili@glasgow.ac.uk

Abstract. The power consumption of computing is coming under intense scrutiny worldwide, driven both by concerns about the carbon footprint, and by rapidly rising energy costs. ARM chips, widely used in mobile devices due to their power efficiency, are not currently in widespread use on the Worldwide LHC Computing Grid. However, LHC experiments including ATLAS and CMS are increasingly able to compile their work-loads on the ARM architecture to take advantage of various HPC facilities.

The work described in this paper compares the energy consumption of various work-loads on two almost identical machines, one with an arm64 CPU and the other with a standard x86_64 CPU, operating in identical conditions. This builds on our initial study of two rather dissimilar machines, located at different UK Universities [1]; this study produced some interesting results which were at times contradictory, and raised the need for a closer comparison.

The set of benchmarks used were designed to represent a typical HEP work-loads. They include CPU-intensive, memory-intensive, and I/O-bound tasks, ranging from simple scripts to full ATLAS simulations. As they became available, we have also used the HEP-Score containerized jobs, which are actively being developed to match LHC Run 3 conditions and can target different architectures.

1. Introduction and Motivations

The power consumption of the global computing infrastructure used to process high-energy physics (HEP) data from the LHC experiments, known as the Worldwide LHC Computing Grid (WLCG [2]), is a significant operational cost.

ARM chips, used in many mobile and portable devices due to their power efficiency, could be adapted to help combat this issue, but are not yet widely adopted as capacity hardware on the WLCG. However, over the last few years, the use of ARM chips in HPC machines has encouraged the LHC experiments to compile and validate their work-loads on the arm64 architecture.

To investigate the performance and energy consumption of ARM chips, the work described in this paper compares two similar machines: one equipped with an Ampere arm64 CPU and one with a standard AMD x86_64 CPU. The objective being to demonstrate whether the power efficiency can be improved for LHC computing by deploying ARM-based hardware.

2. Methodology

In order to compare the energy consumption of the two architectures, two almost identical machines were acquired, with the following CPU specifications respectively:

- Single-socket AMD EPYC 7643 processor [3], with 48 cores / 96 threads at 2.3 GHz and a nominal Thermal Design Power (TDP) of 300 W, which will be referred to as "x86".
- Single-socket Ampere Altra Q80-30 processor [4], with 80 Neoverse N1 cores [5] at 3 GHz and a nominal TDP of 210 W, referred to as "arm".

Both machines are built within a SuperMicro chassis and configured with 256 GB of DDR4 3200 MHz RAM and a 3.84 TB SSD. It is important to note that the two machines were of almost identical price, which from a site or provider perspective, means they are directly comparable, despite having different numbers of physical cores.

In addition to the machines above, we also included in the comparison a standard worker-node representative of our Tier-2 cluster at Glasgow:

- Dual-socket AMD EPYC 7513 processors, each with 32 cores / 64 threads at 2.6 GHz and a nominal TDP of 200 W, configured with 512 GB DDR4 3200 MHz RAM and a 3.84 TB SSD drive, which will be referred to as "2*x86".

It should be noted that this latter machine is part of a larger 2-unit / 4-node Dell chassis, but by considering a scaled price, this node is also comparable to the test machines described above.

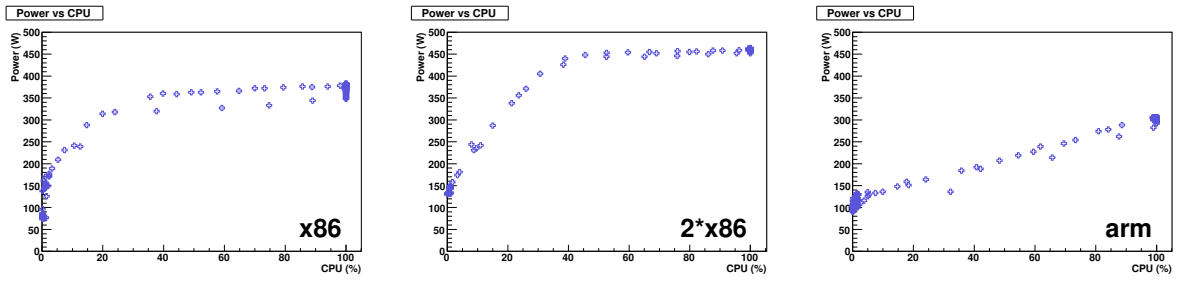
The power consumption was measured using the open-source IPMItools utility [6] that provides remote management of server hardware via the Intelligent Platform Management Interface (IPMI) protocol. This tool is already integrated into our site monitoring system [7], displaying real-time power consumption for the entire cluster, among other metrics.

For this study, we used a custom script to export power consumption, CPU and RAM usage at ten second intervals. The script starts collecting metrics one minute before running the work-load and continues for one minute after the work-load has completed.

All metrics are exported to a CSV file for later analysis, i.e., to extract the average power $\overline{P} = \frac{1}{n}\sum_{i=0}^{n} P_i$ and the total energy E, calculated as the integrated power over time: $E = \sum_{i=0}^{n} P_i \times \delta t_i$ and expressed in kW×h.

With such metrics we can make a scatter plot of power versus fractional CPU usage for an identical work-load run on different machines (see fig. 1). It is interesting to note that the power consumption of the arm shows a linear increase with CPU load, while for both x86 machines the power saturates at around 50% CPU usage, reflecting the onset of hyper-threading.

The actual work-load in this instance is an ATLAS simulation producing 1,000 $t\bar{t}$ events (see section 3.2 for more details).



(a) (b) (c)

Figure 1: Power vs. CPU usage for a typical work-load executed on our three machines (left to right: x86, 2*x86, arm).

2.1. IPMI Validation

The accuracy of the power readings obtained through IPMItools was validated with external meters connected to the servers.

Two meters were used, one for each of the server's dual power supplies, and the integrated power consumptions were summed and compared with the IPMI readings.

Test	Duration	Arch	IPMI (kWh)	Plug (kWh)	Δ (kWh)	Δ%
idle	30 min	x86	0.04766	0.046 (=0.021+0.025)	+0.00166	-3.5
idle	30 min	arm	0.05668	0.054 (=0.024+0.030)	+0.00268	-4.7
load	40 min	x86	0.25729	0.263 (=0.128+0.135)	-0.00571	-2.2
load	30 min	arm	0.13418	0.134 (=0.064+0.070)	+0.00018	+0.1

Table 1: Total energy calculated from IPMI readings compared to values taken from the external meters. The HEP work-load used to measure the power usage under load lasted longer on the x86 than the arm, thus the difference in duration for the load test.

Depending on the machine and the state (loaded / idle), the obtained results were in agreement to within a few percent (see table 1), with the largest difference being 4.7%. Therefore, we assume an error margin of ±5% on all our measurements.

3. Benchmarks and Results

To compare performance and energy consumption, this set of benchmarks was used:

1) Idle: the power consumption measured while servers are idle.
2) ATLAS work-load: a full Geant4 multi-threaded simulation (sim-digi) executed within the Athena Software framework [8].
3) HEP-Score: containerized HEP-specific work-loads built for benchmarking purposes [9], only builds available for both x86_64 and arm64 architectures were used.

All benchmarks were executed under identical conditions on both architectures to ensure that the results were consistent and unbiased. Each benchmark was run multiple times, and the average results over these runs were calculated.

3.1. Idle Consumption

In this benchmark, we used the IPMI exporter script to measure the power consumption of the x86 and arm when idling for 1 hour. The arm shows a higher baseline and larger oscillations between power states compared to the x86, leading to a higher average consumption (see fig. 2).

The x86 consumes an average of 81.8 W and the arm an average of 103.8 W, resulting in a difference $\Delta P_{idle} = 22$ W. This would suggest that ARM servers may be slightly less power efficient during periods of inactivity or I/O-bound operations. However, it is important to note that these results are specific to the machines used for this study and that other differences between the hardware may have affected power efficiency.

3.2. Full ATLAS Simulation

The ATLAS simulation framework Athena version 23.0.3 is available for both x86_64 and arm64 architectures via the CernVM File System (CVMFS [10]). This is a stable release that has been thoroughly tested and validated by the ATLAS collaboration, making it an ideal candidate for benchmarking.

We chose a $t\bar{t}$ simulation, run via a customized version of the standard $t\bar{t}$ script [11], specifying the number of events to be processed. This job involves a multi-threaded full ATLAS detector

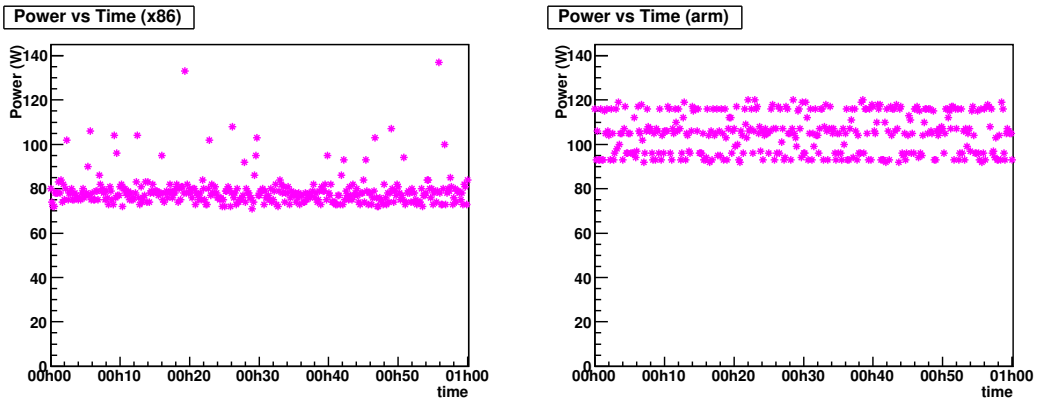


Figure 2: Idle power of the x86 (left) and the arm (right).

simulation built on Geant4, and uses as input an existing $t\bar{t}$ gen-events file, also taken from CVMFS. The number of threads is set to the maximum number available on each CPU to fully utilize the avaiable resources. Processing 10,000 $t\bar{t}$ events takes between five and eight hours, depending on the machine.

To evaluate the performance of our machines, we conducted three simulation runs on each. The average execution time and energy consumption are summarised in table 2, together with their standard deviations.

Arch	Threads	Time	δT	E (kWh)	δE	Idle (W)	Max (W)	W×h/evt
x86	96	07:46:14	0.3%	2.8966	0.6%	85	382	0.000290
2*x86	128	05:44:25	0.4%	2.5843	0.6%	133	463	0.000258
arm	80	05:19:07	2.2%	1.5853	2.5%	110	309	0.000159

Table 2: Results of the ATLAS simulations. Each line is the average result of three runs (10,000 $t\bar{t}$ simulated events each); the standard deviation of energy and time are given in the δ columns.

Notably, the arm exhibited superior energy efficiency and faster execution times compared to both x86 machines. This shows that for a typical HEP work-load, ARM machines can be faster than their x86 counterparts while reducing energy consumption per event by almost half.

3.3. HEP-Score Containers

The HEP-Score benchmark suite is a collection of work-loads from the LHC experiments used to evaluate the performance of HEP workflows. While the full software suite had not been released for ARM at the time of this study, a number of experimental work-loads were available as standalone containers.

We used five such HEP-Score containers from the Container Registry [12], which included a combination of event generation, detector simulation and track reconstruction tasks from two major experiments (ATLAS and CMS). Table 3 summarises the results obtained on our three machines for each work-load.

Since HEP work-loads are designed to scale the number of parallel jobs with the number of available threads, we could not compare the power consumption directly among machines with a different number of threads, as the machine with more threads would have completed more work. To address this issue, the energy usage was scaled by the amount of work done.

Also, the work-load scores (WL-Scores) produced by the software varies widely among different tasks. For a meaningful comparison, scores were normalised separately for each

Arch	work-load	E (kWh)	E_{scaled}	WL-Score	$Score_{norm}$	E/Score
x86	atlas-gen_sherpa	0.04576	0.04576	113.3959	0.8786	0.0004
x86	atlas-sim_mt	0.51478	0.51478	0.4068	0.7214	1.2655
x86	cms-gen-sim-run3	0.27235	0.27235	3.7347	0.7084	0.0729
x86	cms-digi-run3	0.07224	0.07224	15.0538	0.6822	0.0048
x86	cms-reco-run3	0.16447	0.16447	6.4072	0.7097	0.0257
2*x86	atlas-gen_sherpa	0.06509	0.04882	125.0605	0.9689	0.0005
2*x86	atlas-sim_mt	0.6109	0.45818	0.5639	1.0000	1.0834
2*x86	cms-gen-sim-run3	0.3319	0.24893	5.1025	0.9678	0.0650
2*x86	cms-digi-run3	0.08937	0.06703	22.0662	1.0000	0.0041
2*x86	cms-reco-run3	0.19734	0.14801	9.0284	1.0000	0.0219
arm	atlas-gen_sherpa	0.02856	0.03427	129.0706	1.0000	0.0002
arm	atlas-sim_mt	0.26156	0.31387	0.5486	0.9729	0.4768
arm	cms-gen-sim-run3	0.12691	0.15229	5.2721	1.000	0.0241
arm	cms-digi-run3	0.04008	0.04810	18.7561	0.8500	0.0021
arm	cms-reco-run3	0.09574	0.11489	7.4504	0.8252	0.0129

Table 3: Summary of the HEP-Score runs on our three machines for each work-load.

container to the highest value obtained for such work-load. (In the final HEP-Score release, WL-Scores will be normalised against those obtained from a reference machine nominated by the HEP-Score developers.)

Our analysis shows that, on average, the arm ran approximately 20% faster and used around 35% less energy per HEP task than the equivalent x86 processors. These findings suggest that the arm64 architecture may offer a promising platform for more power-efficient HEP computing.

Results from all containers are summarized in table 3. Figure 3 illustrates the energy usage per unit WL-Score of our three machines.

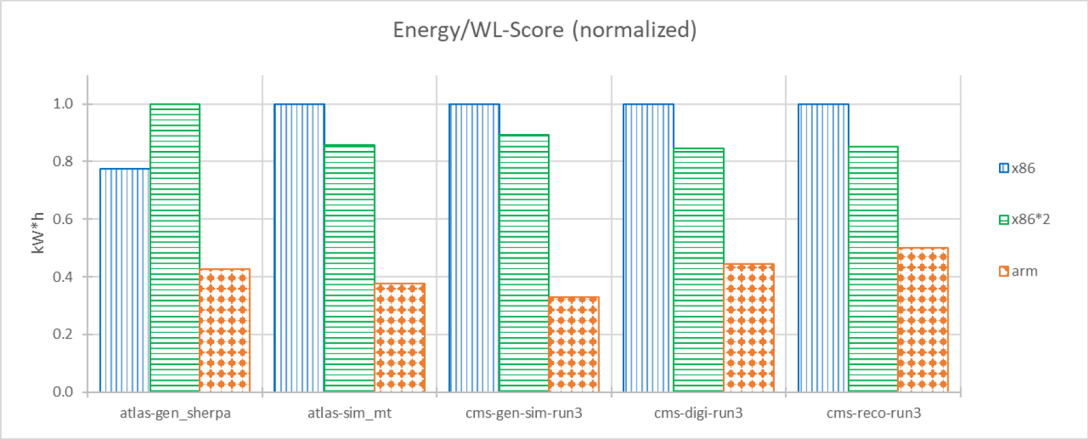


Figure 3: Comparison of the energy usage per normalised score among the three machines (x86, 2*x86, arm), calculated for all five HEP work-loads used in this study. These results are normalised to 1 to improve readability.

4. Limitations

Despite active development within all LHC experiments, at the time of writing, support for architectures other than x86_64 was limited. However, the situation has rapidly improved, with

all major experiments having released arm64 builds of their software.

Another issue to note was the difficulty in procuring ARM-based hardware, due to a lack of ARM support from major vendors. However, this too is likely to change over time as demand for ARM systems increases.

While the ARM machine shows better performance in terms of speed and energy efficiency on most benchmarks, there are still areas that require further investigation. For instance, we have run custom ATLAS simulations and other custom benchmarks without paying much attention to the validity of the actual output. A run was deemed successful as long as the execution terminated without errors. Therefore, in future studies, we plan to rely solely on validated code, such as the HEP-Score work-loads which are under active development.

Although we trust our IPMI power readings with a $\pm 5\%$ accuracy, a more precise measurement could be achieved by using a metered PDU with remote readings on each plug. Moreover, error propagation was not performed in this study; the $\pm 5\%$ error on IPMI reading was taken to be the major source of uncertainty.

In conclusion, while our study highlights the potential benefits of ARM machines for power-efficient HEP computing, all quantitative results presented in this paper must be considered preliminary. More in-depth measurements are currently underway.

5. Conclusions and Outlook

With its promising combination of speed and power efficiency demonstrated in this study, there is great potential for the arm64 architecture to become a widespread platform for HEP computing, especially as more experiments release ARM builds of their software, and ARM-based hardware becomes more readily available. This could lead to great improvements in energy-efficiency in HEP, which is an important consideration given the large-scale computing needs of the field.

We started collaborating with the HEP-Score task force to develop an energy plug-in for a more standardized and accurate evaluation of power usage across HEP work-loads. To continue this line of research, the set of benchmarks will be extended also to GPUs.

This study provides a first step towards a comprehensive evaluation of hardware solutions in HEP computing, which considers power efficiency as important as raw performance. We believe that our findings, along with those from future work, will help guide researchers and institutions in selecting the best hardware and architecture for their computing needs.

References

[1] E.Simili, et al. "Power Efficiency: x86 versus ARM", GridPP47 / SWIFT-HEP (2022), https://indico.cern.ch/event/1128343/contributions/4787174/

[2] The Worldwide LHC Computing Grid WLCG (Oct. 2022), https://wlcg.web.cern.ch/

[3] AMD EPYC™ 7003 Series Processors (Oct. 2022), https://www.amd.com/en/processors/epyc-7003-series

[4] Ampere® Altra® Multi Core Server Processors (Oct. 2022), https://amperecomputing.com/processors/ampere-altra

[5] Neoverse N1 CPU (Oct. 2022), https://www.arm.com/products/silicon-ip-cpu/neoverse/neoverse-n1

[6] IPMItools (Oct. 2022), https://github.com/ipmitool/ipmitool

[7] E.Simili, et al. "A Hybrid System for Monitoring and Automated Recovery at the Glasgow Tier-2 Cluster", *EPJ Web of Conferences* **251**, 02047 (2021). (doi: 10.1051/epjconf/202125102047)

[8] ATLAS Software Framework Athena (Oct. 2022), https://atlassoftwaredocs.web.cern.ch/athena/

[9] D.Giordano, et al. "HEPiX Benchmarking Solution for WLCG Computing Resources", *Comput Softw Big Sci* **5**, 28 (2021). doi: https://doi.org/10.1007/s41781-021-00074-y

[10] CernVM File System CVMFS (Oct. 2022), https://cernvm.cern.ch/fs/

[11] ATLAS Run3 Full Geant4 $t\bar{t}$ simulation script (2022), /cvmfs/atlas-nightlies.cern.ch/repo/sw/master_AthSimulation_x86_64-centos7-gcc11-opt/2022-01-29T2101/AthSimulation/22.0.53/InstallArea/x86_64-centos7-gcc11-opt/bin/test_RUN3_FullG4_ttbar_2evts.sh

[12] HEP Container Registry (Oct. 2022), https://gitlab.cern.ch/hep-benchmarks/hep-workloads-sif/container_registry