# Energy Efficiency Impact of Processing in Memory: A Comprehensive Review of Workloads on the UPMEM Architecture

Yann Falevoz(✉) and Julien Legriel

UPMEM, 6 Pl. Robert Schuman, 38000 Grenoble, France
yfalevoz@upmem.com

**Abstract.** Processing-in-Memory (PIM) architectures have emerged as a promising solution for data-intensive applications, providing significant speedup by processing data directly within the memory. However, the impact of PIM on energy efficiency is not well characterized. In this paper, we provide a comprehensive review of workloads ported to the first PIM product available on the market, namely the UPMEM architecture, and quantify the impact on each workload in terms of energy efficiency. Less than the half of the reviewed papers provide insights on the impact of PIM on energy efficiency, and the evaluation methods differ from one paper to the other. To provide a comprehensive overview, we propose a methodology for estimating energy consumption and efficiency for both the PIM and baseline systems at data center level, enabling a direct comparison of the two systems. Our results show that PIM can provide significant energy savings for data intensive workloads. We also identify key factors that impact the energy efficiency of UPMEM PIM, including the workload characteristics. Overall, this paper provides valuable insights for researchers and practitioners looking to optimize energy efficiency in data-intensive applications using UPMEM PIM architecture.

**Keywords:** Processing in memory (PIM) · UPMEM architecture · Data-centric architectures · Workload optimization · High-performance computing · Computational efficiency · Energy consumption · Energy efficiency

## 1 Introduction

With the exponential growth of data-intensive applications, traditional compute-centric architectures have become a bottleneck for efficient and scalable processing. The demand for high-performance computing has pushed the limits of Moore's Law, leading to a growing disparity between the processing power

This paper is supported by European Union's Horizon Europe programme (HORIZON-CL4-2021-HUMAN-01 and HORIZON-EIC-2021-PATHFINDEROPEN-01), under SustainML and BioPIM projects.

The original version of this chapter was previously published without open access. A correction to this chapter is available at
https://doi.org/10.1007/978-3-031-48803-0_41

of CPUs and the bandwidth of memory. This has resulted in significant performance degradation and energy inefficiencies when dealing with large-scale data-intensive workloads.

### 1.1 Limitations of Traditional Compute-Centric Architectures for Data-Intensive Workloads

Traditional compute-centric architectures, with separate processing and memory components, face limitations with data-intensive workloads. The "memory wall" or "data movement bottleneck" arises due to the extensive data movement between processor and memory [25]. This poses challenges for applications that demand processing large data volumes, leading to high energy consumption and latencies [14]. As a result, data-intensive applications suffer from performance degradation, and the energy consumption can be unsustainable. Despite solutions like caching and prefetchers, which offer some performance improvements, they still rely on compute-centric architectures and are constrained by data movement limitations.

### 1.2 From Data Movement to Data Processing

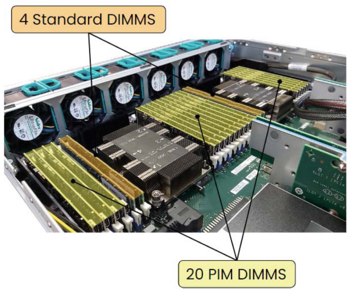
Processing in Memory (PIM) is a promising paradigm that tackles traditional compute-centric architecture limitations [18]. With PIM, processing occurs directly within the memory where data resides, eliminating data movement between processor and memory [18,25,26]. This yields significant energy savings, vital for energy-conscious data centers, and reduced latency, especially for data-intensive workloads. By reducing energy consumption, PIM can facilitates the development of more efficient, sustainable and cost effective computing systems for machine learning, data analytics, and graph processing [18]. Moreover, it unlocks new applications and breakthroughs in scientific research, drug discovery, and other areas where large-scale data analysis is required.

### 1.3 UPMEM Architecture

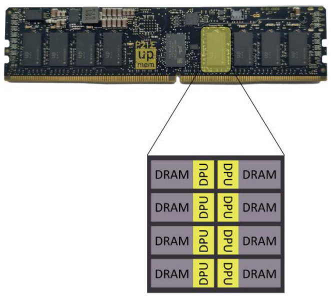
UPMEM PIM is the first commercial PIM product available on the market [11]. It offers a radical new approach to data processing by combining DRAM and logic functions in the same chip, allowing for ultra-fast data access and processing, and eliminating the need for frequent data transfers between the memory and the processor. This results in a substantial overall reduction in power consumption and latency, as well as improved scalability and performance.

An UPMEM PIM server is a standard application server where most of the DIMM slots have been populated with PIM DIMMs (Fig. 1a). Standard and UPMEM DIMMs coexist in the server and the firmware is made aware of the specific configuration. A typical configuration totalizes 20 PIM DRAM modules.

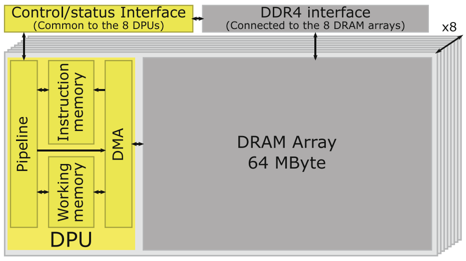
A module takes the form of a standard DDR4-2400 DIMM (Fig. 1b). It has a capacity of 8GB, and embeds 16 PIM chips on it, 8 on each side. Inside each chip, 8 processors coexist (Fig. 1b). These processors are called DPU, standing for Data Processing Unit. A typical configuration then totalises 2560 DPUs for 160 GB of PIM DRAM.



(a) UPMEM PIM server



(b) UPMEM PIM module



(c) Structure of a PIM chip

**Fig. 1.** UPMEM PIM platform

A DPU is a custom proprietary 32-bit processor, inspired by the RISC philosophy, but adapted to the specific design challenges of DRAM manufacturing processes. It shares the access with the host CPU to a DRAM bank, called Main RAM (Fig. 1c). Instruction and data caches are replaced by an instruction RAM and a Working RAM. DPUs are independent from each others' memory and run asynchronously relative to each other. 24 hardware threads can execute independently of each other in a DPU. All inter DPU communication takes place through the host CPU. This architecture allows every DPU to efficiently work within their own fragment of the dataset and even with their own programming routine if chosen to be different. This technology comes with a set of tools designed to make the porting of applications as smooth as possible.

Each DPU can be programmed with standard tools and skillset. UPMEM provides a complete software stack that enables DPU programs to be written in C and APIs in common languages for host programming. Communication libraries make it easy to configure the DPUs, organise distribution of data, scheduling, and retrieve results. The compiler for the DPU target is based on LLVM and comes with an efficient tool for debugging based on LLDB.

### 1.4 Objectives

Since the launch of UPMEM's PIM architecture, researchers have shown growing interest in exploring its potential for accelerating data-intensive workloads. Numerous papers report impressive speedups of tens or even hundreds of times compared to traditional architectures in various applications like genomics, analytics, and artificial intelligence.

However, most existing literature focuses primarily on performance gains, with little attention to energy efficiency. This paper aims to comprehensively review workloads ported to UPMEM's PIM product, quantifying PIM's impact on both performance and energy efficiency. By doing so, we aim to present a more complete picture of PIM technology's benefits and its potential for sustainable and energy-efficient computing.

## 2 Material and Methods

### 2.1 Search Strategy

We conducted a search of the literature using Google Scholar, with the search term "UPMEM". The search was conducted on April 6$^{\text{th}}$, 2023, and yielded over 100 results. For each result, we recorded the following information: the date of publication, the field, the title, the type of publication (such as paper, short paper, thesis, abstract, white paper, poster, or slides), the way in which the UPMEM product was mentioned in the publication (such as simple reference, technical presentation, actual use, or acceleration results), the link to the publication, the link to a GitHub repository if provided, and the institutes who contributed to the publication.

### 2.2 Inclusion Criteria

To ensure the relevance of our review, we limited our search to publications that met the following inclusion criteria:

1. The publication must be written in English
2. The publication must be a paper available in full-text format.
3. The publication must provide sufficient details on the experiment setup to evaluate the gain in energy efficiency.
4. The publication must provide acceleration results, not just in terms of speedup, but in terms of throughput or execution times for both PIM and reference implementations, or at least provide data to calculate this information. It will be required to evaluate the gain in energy efficiency.
5. The acceleration results must have been measured on real hardware, and on the latest version of UPMEM PIM DIMMs. This excludes publications prior to 2021, for which this version of UPMEM PIMs were not yet available. This condition is related to the requirement of knowing the power consumption of the PIM modules to evaluate the gain in energy efficiency.
6. The publication must show that efforts to optimise performance have been made, that is, that the work was not just to show that porting is feasible.

We excluded duplicate publications and publications that did not meet these criteria.

### 2.3 Estimation of Energy Efficiency Gain

Less than half of the resulting papers provide insights on the impact of PIM on energy efficiency. However, these papers differ in their evaluation methods, making it challenging to compare and draw conclusions from the results. To provide a comprehensive overview of the energy efficiency impact of PIM, we have decided to propose data center level approach. To estimate the energy efficiency of the PIM system and compare it with the baseline system:

1. For the reference servers, we gathered power data (TDP) from datasheets and specifications provided by manufacturers for main parts of the systems, that is processors and DIMMs (Table 1). We have added up these figures and, to ensure accuracy, we also added a base consumption of 115W for the chassis, fans, common modules and PSU efficiency loss, to calculate the power consumption of each system. The resulting value has been verified to be relevant with various online power calculators [3,9] for different configurations. For the PIM server, in addition to the server's power consumption calculated using the preceding method, we accounted for the measured power consumption of the PIMM DIMMs, which is 23.22W.
2. We applied a Power Usage Effectiveness (PUE) [20] factor to take into account the power consumed by the whole data center infrastructure including servers, network switches, and cooling facilities. We used a factor of 1.55, which was the average in 2022 [16].
3. To calculate the energy per execution, we multiplied the power consumption of each system by the execution time given in the paper or divided it by the throughput, depending on what is available in the paper.
4. Finally, we compared the energy per execution of the PIM system with that of the baseline system, considering it is dedicated to the specific workload, to evaluate the gain in energy efficiency.

**Table 1.** Power consumption

| Component | TDP (W) |
|---|---|
| Intel Xeon Silver 4110/4215/E5-2630 v4 | 85 [5,7,8] |
| Intel Xeon Gold 5120 | 105 [4] |
| Intel Xeon E5-2697 v2 | 130 [6] |
| 32GB DDR4-2666 RDIMM | 5 |
| 64GB DDR4-3200 RDIMM | 5.5 |
| NVIDIA A100 | 300 [2] |

As an example, a standard UPMEM PIM server with 2 x INTEL Xeon Silver 4215 processors, 4 x 32 GB DDR4-2666 RDIMMs and 20 x UPMEM PIM DIMMs results in a power consumption of 770 W. Taking into account the PUE, this leads to 1193 W. Considering a workload with an execution time of 10 ms on that server, the energy per execution would be of 11,93 J.

While this rudimentary approach offers a good initial trend for evaluating energy efficiency gain, it cannot replace direct measurements on a real PIM system and may introduce some unfavorable bias. For instance, it assumes the CPU power consumption remains the same in the PIM system, even though it is significantly less solicited due to offloading computations to the PIM cores. Additionally, we did not consider the substantial reduction in data transfers between the CPU and DRAM, which can have a notable impact on energy consumption. In particular, read operations in DRAM consume approximately 100 and 1000 times more energy than multiplication and addition, respectively [27]. Neglecting these aspects can impact energy consumption significantly and underestimate the PIM system's true energy efficiency gain.

## 3 PIM Implementations of Modern Workloads

### 3.1 Mathematical Functions

**Item et al.** [23] discusses the problem of the limited instruction sets of PIM architectures and the need for efficient support for **transcendental functions** and other hard-to-calculate operations in PIM systems. The authors present TransPimLib, a library that provides CORDIC-based and Lookup Table-based methods for trigonometric functions, hyperbolic functions, exponentiation, logarithm, square root, etc. They implement TransPimLib for the UPMEM PIM architecture and evaluate its methods in terms of performance and accuracy, using microbenchmarks and three full workloads (Blackscholes, Sigmoid, Softmax).

Blackscholes that use Multiplication-based Fuzzy Lookup Table (M-LUT) and LDEXP-based Fuzzy Lookup Table (L-LUT) versions are, respectively, within 75% and 82% the performance of the 32-thread CPU baseline. The fixed-point L-LUT version is 62% faster than the 32-thread CPU baseline. In this last case, the speedup achieved does not translate into a gain in terms of energy efficiency. In fact, the power consumption of the PIM server is significantly higher compared to the traditional server, which translates into a 58% increase in energy costs, i.e. 69.8 additional Joules per operation. Nonetheless, as the authors point out, TransPimLib's techniques can curtail the need for data transfer from PIM cores to the CPU for applications using neural networks and machine learning algorithms, resulting in a speedup of 6 to 8 times compared to executing them on the host CPU.

### 3.2 Databases

**Baumstark et al.** [13] presents an approach to integrating PIM technology into a **query processing engine** of a database management system (DBMS) by leveraging adaptive query compilation. PIM technology moves computation directly to memory, which can improve the performance of DBMSs. The evaluation on UPMEM system was conducted using the Social Network Benchmark (SNB) dataset from the Linked Data Benchmark Council (LDBC). The experiment setup used two Intel Xeon Silver 4215R processors, 512 GB of DRAM, and four UPMEM DIMMs. The evaluation used the first three Interactive Short Read queries from the SNB dataset.

The results of the evaluation showed that the proposed approach speeds up query compilation time up to 30 times compared to the baseline compiler, which has resulted in a reduction of up to 1/3 in the overall query execution time. Given the power consumption of the two configurations, the gain in speed translates into a 17% gain in energy efficiency, that is, the PIM server saves up to 66 J per operation.

**Baumstark et al.** [12] discusses the benefits of PIM technology for improving the performance of **graph database traversal**. The authors used the Social Network Benchmark (SNB) dataset from the Linked Data Benchmark Council (LDBC) to evaluate the performance of this approach in a graph database management system (DBMS) and used a graph database engine called Poseidon, which supports both a persistent memory storage engine and an in-memory mode. The experiments were performed on a system with two Intel Xeon Silver 4215R, 512 GB of DRAM, and 4 UPMEM DIMMs.

The benchmarks shows that UPMEM PIM technology can outperform the runtime of a comparable CPU execution, especially when high parallelism is used. The inclusion of table transfer time between the host and the DPU results in a 30% reduction of the total table scan execution time. It is worth noting that this performance improvement is achieved despite a limited number of DPUs. The similar power consumption of the two configurations leads to a 11% gain in energy efficiency, that is, 26 J saved per operation.

**Kang et al.** [24] focuses on designing a **practical ordered index** for PIM, which can achieve low communication cost and high load balance regardless of the degree of skew in data and queries. The authors present PIM-tree, a practical ordered index that leverages the strengths of host CPU and PIM nodes to overcome load imbalance problems. The authors conducted experiments using UPMEM PIM architecture with 16 UPMEM DIMMs and compared the results with those of two traditional indexes performed on a machine that has two Intel Xeon E5-2630 v4 CPUs.

The results show that the PIM-tree outperforms state-of-the-art indexes, with more than 2x speedup compared to state-of-the-art indexes on a machine with similar performance for an INSERT operation. Given the power consumption of the two configurations, this translates into a 20% reduction in energy consumption, i.e. up to 40 J saved per operation.

## 3.3 Artificial Intelligence (AI)

**Gómez-Luna et al.** [21] evaluates the potential of PIM architectures in accelerating the **training of machine learning (ML) algorithms**, which is a computationally intensive process frequently memory-bound due to large training datasets. The authors implemented four representative classical ML algorithms, including linear regression, logistic regression, decision tree, and K-Means clustering, on the UPMEM PIM architecture. They rigorously evaluated and characterized the algorithms in terms of accuracy, performance, and scaling, comparing them to their counterparts on CPU and GPU, using the Criteo dataset.

The experimental results show that the PIM implementation of decision tree is 73.4× faster than a state-of-the-art CPU version, and 5.3× faster than a state-of-the-art GPU version. This translates into an energy gain of almost 30x and 4.2x respectively, that is 159 kJ and 17.8 kJ per execution. K-Means clustering on PIM is 2.7× and 2.76× faster than state-of-the-art CPU and GPU versions, respectively. This results into an energy gain of 10% and 2.2x respectively, i.e. 3.7 and 62.8 kJ saved per execution.

**Das et al.** [15] discusses the implementation of two **Convolutional Neural Networks (CNNs)**, YOLOv3 and eBNN, on the UPMEM PIM system, and compare the performance of their implementation with several other recently proposed PIM architectures.

Results showed that the performance speed-up of eBNN inference performance with respect to traditional CPU-based system scales up linearly as more parallel processing elements (DPU) are incorporated in the UPMEM system, reaching 92x for a complete PIM server whith 2560 DPU. Despite the large difference in power consumption between the PIM server and the reference server, the high acceleration factor results in a significant gain in energy efficiency, with a reduction factor of 36.6x, i.e. 199 µJ per execution.

## 3.4 Genomics

**Diab** [17] propose Alignment-in-Memory (AIM), a framework for high-throughput **sequence alignment** using PIM, and compares the performance of UPMEM PIM with 3 conventional CPU systems in accelerating memory-bound sequence alignment workloads.

The authors observe that UPMEM outperforms the CPU systems in the majority of cases, even when data transfer time is included. **Needleman-Wunsch (NW)** speedup is low with speedups achieved of 1.1x, 1.78x and 1.5x over the three reference systems respectively and do not translates into energy gain. **Smith-Waterman-Gotoh (SWG)** and **GenASM** show a moderate acceleration (2.8x–6.2x) that translates in a slight energy gain (1.4x–2.7x). In the case of SWG, however, this moderate gain translates into a significant absolute value per execution (240 kJ–1.9 MJ) due to the long duration of this workload. **Wavefront algorithm (WFA)** and **WFA-adaptive** show strong accelerations compared to system 1 and 2 (30.1x–36.5x), that translate into significant energy gain (13.1x–14.5x) and absolute energy per execution that range from 5 to 5.5 kJ.

## 4 Discussion

### 4.1 Moving to a Larger Scale: Implications for Data Center Operations

The present study highlights the significant variation in speedup and energy efficiency gains across different workloads. To facilitate a proper comparison, it is imperative to move to a larger scale for a comprehensive understanding of the implications. Therefore, the energy saved/wasted in GWh by 1000 PIM servers operating as a small data center was determined by running the same number of run executions as the baseline server over a period of three years (Fig. 2), which is the standard depreciation period for a server. This approach provides a valuable perspective on energy efficiency gains and highlights the potential for significant savings in energy consumption by adopting more energy efficient computing techniques.

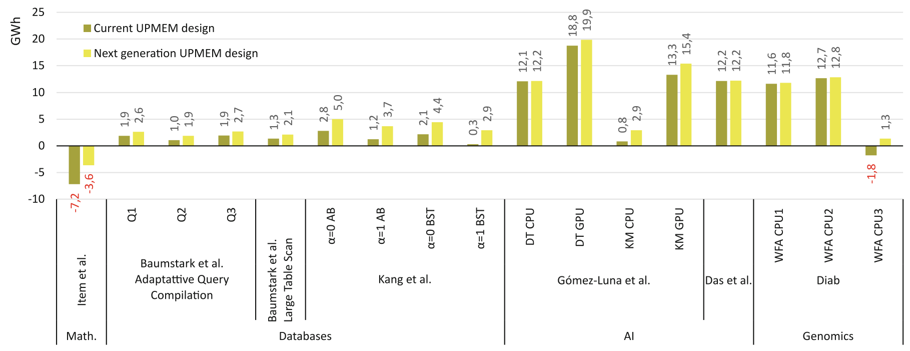


**Fig. 2.** Comparison of energy savings and wastage in GWh over 3 years for 1000 PIM servers vs. 1000 baseline servers in a small data center, for the current and next generation of UPMEM PIM DIMMs.

On one hand, savings can be significant for workloads with energy efficiency gains, reaching up to 12.7GWh over the typical 3-year depreciation period compared to CPU configurations and up to 18.8GWh compared to GPU configurations. At electricity prices of \$0.192/kWh [1], this translates to a cost reduction of up to \$2.4M and \$3.6M respectively on operating expenses (OPEX) over the same period. On the other hand, for certain workloads, such as transcendental functions, the acceleration does not compensate for the additional power induced by PIM DIMMs. This results in an increased OPEX charge. This highlights the importance of carefully considering workload characteristics when designing and optimising computing systems, as the potential for energy savings can vary significantly depending on the specific workload being performed.

The natural next step would be to assess the impact in terms of Total Cost of Ownership (TCO), but this is beyond the scope of this paper.

## 4.2 Best Suited Applications to Run on UPMEM PIM

Certain types of applications are particularly well-suited to run on UPMEM PIM processors. These include applications that exhibit: (1) data-intensive workloads, which are memory-bound on compute-centric architectures [22], and for which DPUs compute time is largely dominant over data transfer time; (2) workloads with data-level parallelism (i.e. little or no communication across DPUs) [22], allowing every DPUs to efficiently work within their own fragment of the dataset; (3) applications with irregular data access patterns; and (4) applications with algorithms difficult to vectorise.

It is worth noting that applications that only use the operations supported by the hardware show much greater performance and energy efficiency gains [22]. For example, Giannoula et al. [19] showed that their sparse matrix vector multiplication implementation on UPMEM PIM outperforms CPU implementation for data types for which the multiplication is supported by the hardware, while performance for data types for which the multiplication is software emulated can be significantly lower, and event worse than CPU implementation. Das et al. [15], as well as Gómez-Luna et al. [21], overcame the computational limitation of the UPMEM PIM related to the fact that FPs are emulated by software and not directly implemented in hardware by using quantified versions of the algorithms studied.

## 4.3 Limitations and Future Perspectives

While this approach offers an initial trend for assessing energy efficiency gains, it cannot replace direct measurements on real systems and introduce biases leading to an underestimation of the true PIM system's energy efficiency gain. A more sophisticated method considering reduced CPU activity and fewer transfers between CPU and DRAM would provide a more accurate estimate. Measuring power consumption on real systems remains the most reliable approach.

While this study presents a comprehensive review of existing literature on the UPMEM PIM architecture and its performance benefits, some publications that did not meet the selection criteria may still present interesting results. In particular Gómez-Luna et al. [22] covers workloads and fields of application that were not included in this review. The paper discusses the performance of UPMEM PIM systems compared to a CPU and a GPU for various benchmarks, but lacks the necessary data for assessing energy impact, as execution times do not include transfer times, and speedups compared to CPU and GPU are given without corresponding execution times. The PIM system outperforms the CPU and GPU for most of the benchmarks, being on average 23.2×, and up to 629.5× faster than the CPU, and on average 2.54×, and up to 57.5× faster than the GPU. Moreover, as the UPMEM PIM system is still a relatively new technology, there may be other studies and applications that have not yet been explored.

As the UPMEM PIM architecture continues to evolve [10], there are several improvements that could enhance its performance and energy efficiency. The next version of the architecture includes improvements to the DPU's performance and programmability, such as a frequency increased to 466 MHz or 30% power reduction at same frequency, and direct access to WRAM from the host while the DPU owns the bank. This enables faster processing and more efficient resource utilization. Figure 2 shows the impact of the reduced power consumption of this new design operating at the same frequency. Future versions may support additional operations, expanding the range of applications that can benefit from the PIM approach.

## 5 Conclusions

This paper reviews the use of UPMEM PIM in different computing domains and finds that it can enhance the performance and energy efficiency of certain applications. However, the benefits vary depending on the workload. The characteristics of suitable applications and implications for data centers are discussed, providing valuable insights into the potential of UPMEM PIM as a high-performance and energy-efficient computing solution.

## References

1. Electricity prices. https://www.globalpetrolprices.com/electricity_prices/. Accessed 15 Apr 2023
2. GPU NVIDIA A100. https://www.nvidia.com/en-us/data-center/a100/
3. Intel power calculator. https://servertools.intel.com/power-calculator/
4. Intel Xeon gold 5120. https://www.intel.com/content/www/us/en/products/sku/120474/intel-xeon-gold-5120-processor-19-25m-cache-2-20-ghz/specifications.html
5. Intel Xeon processor e5-2630 v4. https://www.intel.com/content/www/us/en/products/sku/92981/intel-xeon-processor-e52630-v4-25m-cache-2-20-ghz/specifications.html
6. Intel Xeon processor e5-2697 v2. https://www.intel.com/content/www/us/en/products/sku/75283/intel-xeon-processor-e52697-v2-30m-cache-2-70-ghz/specifications.html
7. Intel Xeon silver 4110. https://www.intel.com/content/www/us/en/products/sku/123547/intel-xeon-silver-4110-processor-11m-cache-2-10-ghz/specifications.html?wapkw=%20Intel%20Xeon%20Silver%204110
8. Intel Xeon silver 4215. https://www.intel.com/content/www/us/en/products/sku/193389/intel-xeon-silver-4215-processor-11m-cache-2-50-ghz/specifications.html?wapkw=Silver%204215
9. Outervision® power supply calculator. https://outervision.com/power-supply-calculator
10. Safari live seminar: Fabrice devaux, 2 Feb 2022. https://safari.ethz.ch/safari-live-seminar-fabrice-devaux-feb-2-2022/. Accessed 27 Apr 2023
11. UPMEM tech paper. https://www.upmem.com/
12. Baumstark, A., Jibril, M.A., Sattler, K.U.: Accelerating large table scan using processing-in-memory technology. In: König-Ries, B., Scherzinger, S., Lehner, W., Vossen, G. (eds.) BTW 2023. Gesellschaft für Informatik e.V. (2023). https://doi.org/10.18420/BTW2023-51

13. Baumstark, A., Jibril, M.A., Sattler, K.U.: Adaptive query compilation with processing-in-memory (2023). https://doi.org/10.1109/ICDEW58674.2023.00035
14. Boroumand, A., et al.: Google workloads for consumer devices: mitigating data movement bottlenecks. SIGPLAN Not. **53**(2), 316–331 (2018). https://doi.org/10.1145/3296957.3173177
15. Das, P., Sutradhar, P.R., Indovina, M., Dinakarrao, S.M.P., Ganguly, A.: Implementation and evaluation of deep neural networks in commercially available processing in memory hardware. In: 2022 IEEE 35th International System-on-Chip Conference (SOCC), pp. 1–6 (2022). https://doi.org/10.1109/SOCC56010.2022.9908126
16. Davis, J., et al.: Uptime institute global data center survey results 2022 (2022). https://uptimeinstitute.com/resources/research-and-reports/uptime-institute-global-data-center-survey-results-2022. Accessed 15 Apr 2023
17. Diab, S., Nassereldine, A., Alser, M., Gómez-Luna, J., Mutlu, O., Hajj, I.E.: A framework for high-throughput sequence alignment using real processing-in-memory systems (2023). https://doi.org/10.48550/arXiv.2208.01243
18. Fujiki, D., Wang, X., Subramaniyan, A., Das, R.: In-/Near-Memory Computing, vol. 16. Morgan & Claypool Publishers (2021). https://doi.org/10.1007/978-3-031-01772-8
19. Giannoula, C., Fernandez, I., Gómez-Luna, J., Koziris, N., Goumas, G., Mutlu, O.: Sparsep: towards efficient sparse matrix vector multiplication on real processing-in-memory systems (2022). https://doi.org/10.48550/arXiv.2201.05072
20. Gyarmati, L., Trinh, T.A.: Energy efficiency of data centers. In: Kim, J.H., Lee, M.J. (eds.) Green IT: Technologies and Applications, pp. 229–244. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22179-8_12
21. Gómez-Luna, J., et al.: An experimental evaluation of machine learning training on a real processing-in-memory system (2023). https://doi.org/10.48550/arXiv.2207.07886
22. Gómez-Luna, J., Hajj, I.E., Fernandez, I., Giannoula, C., Oliveira, G.F., Mutlu, O.: Benchmarking a new paradigm: experimental analysis and characterization of a real processing-in-memory system. IEEE Access **10**, 52565–52608 (2022). https://doi.org/10.1109/ACCESS.2022.3174101
23. Item, M., Gómez-Luna, J., Guo, Y., Oliveira, G.F., Sadrosadati, M., Mutlu, O.: Transpimlib: a library for efficient transcendental functions on processing-in-memory systems (2023). https://doi.org/10.48550/arXiv.2304.01951
24. Kang, H., Zhao, Y., Blelloch, G.E., Dhulipala, L., Gu, Y., McGuffey, C., Gibbons, P.B.: PIM-tree: a skew-resistant index for processing-in-memory. Proc. VLDB Endow. **16**(4), 946–958 (2022). https://doi.org/10.14778/3574245.3574275
25. Mutlu, O., Ghose, S., Gómez-Luna, J., Ausavarungnirun, R.: A modern primer on processing in memory. In: Aly, M.M.S., Chattopadhyay, A. (eds.) Emerging Computing: From Devices to Systems, pp. 171–243. Springer, Singapore (2023). https://doi.org/10.1007/978-981-16-7487-7_7
26. Mutlu, O., Ghose, S., Gómez-Luna, J., Ausavarungnirun, R.: Processing data where it makes sense: enabling in-memory computation. Microprocess. Microsyst. **67**, 28–41 (2019). https://doi.org/10.1016/j.micpro.2019.01.009
27. Sze, V., Chen, Y.H., Yang, T.J., Emer, J.S.: Key metrics and design objectives. In: Sze, V., Chen, Y.H., Yang, T.J., Emer, J.S. (eds.) Efficient Processing of Deep Neural Networks, pp. 43–58. Springer, Cham (2020). https://doi.org/10.1007/978-3-031-01766-7_3