Alqalam Journal of Medical and Applied Sciences. 2026;9(11):3604-3611
https://doi.org/10.54361/ajmas.261110

Systematic review

RISC-V Vector Processors for Edge AI and High-Performance Computing: A Focused Review of Architectures and Performance Evaluation Methods

Thuraya Rajab[1]*, Fathi Hamhoum[2]

[1]*Department of Computer Science, Faculty of Science, University of Sabratha, Libya*
[2]*Department of Computer Science, Faculty of Science, University of Zawia, Libya*
Email. *thuraya@ta.sabu.edu.ly*

Abstract
Edge artificial intelligence and high-performance computing differ in available resources, numerical precision, and workload size, making the suitability of RISC-V vector processors dependent on their conditions of use. This paper presents a focused review of selected RISC-V vector processor designs and studies evaluating their performance, aiming to analyse architectural trade-offs and determine the comparability of performance and energy-efficiency results. The review analyses and compares the selected studies according to design characteristics, evaluated workloads, software environments, baselines, and measurement boundaries, distinguishing architectural studies from platform and application evaluations. Comparisons are classified as direct quantitative, conditional quantitative, or qualitative according to the agreement between the conditions underlying the specific results. The analysis shows that kernel acceleration does not necessarily translate into an equivalent improvement in the complete application, and that reduced precision and computational specialisation require performance gains to be interpreted alongside output quality and software support. Differences in energy-measurement boundaries and implementation types also prevent a unified ranking of all reviewed designs. The review contributes by linking results to their conditions and clarifying the limits of their use in evaluating architectural choices for both domains, rather than merely comparing reported peak values. The conclusions remain limited by the scope of the selected studies, their different configurations, and the relationship of some works to shared architectural families. This paper emphasises the importance of common workloads and baselines, documented operating conditions, and combined kernel and application benchmarks to improve the reliability of future comparisons.
Keywords. Vector Processing; RISC-V; High-Performance Computing; Edge Artificial Intelligence; Energy Efficiency; Performance Evaluation.

Introduction

Computing requirements vary with the operating environment. In edge AI applications, such as image analysis within cameras or sensor-data processing, a device must execute tasks quickly within its available energy and memory limits [1]. High-performance computing, in contrast, emphasises intensive scientific calculations and the processing of large volumes of data [2]. Despite these differences, both domains share the need to use processing and memory resources efficiently. The open RISC-V instruction set enables processors with different designs and extensions, making it a suitable basis for examining these requirements [1].

Among its extensions, vector processing is particularly relevant because it applies the same operation to multiple data elements. This approach is useful in operations such as matrix multiplication and neural-network convolution, as well as scientific-computing and signal-processing algorithms. However, support for vector instructions alone does not determine processor performance. Results also depend on the organisation of execution units, the number of lanes, register capacity, memory-access speed, and the ability of software to exploit the hardware [3].

Previous studies have examined RISC-V architectures and software tools from several perspectives. The review by Kalapothas et al. surveyed cores and frameworks used in machine-learning applications [4], whereas Agosta et al. focused on hardware and software support for deep learning at the edge [1]. Silvano et al. provided a broader survey of deep-learning accelerators for heterogeneous platforms, including RISC-V-based solutions [2]. These reviews provide an important foundation for understanding the field, but they also make it necessary to define a contribution that goes beyond restating designs and their characteristics. This paper compares RISC-V vector-processing architectures and their evaluation methods in selected edge AI and HPC studies. The comparison examines not only reported performance and energy-efficiency figures but also the conditions under which they were obtained.

Differences in numerical precision, problem size, software baseline, and measurement method may prevent direct comparison [3]. Furthermore, accelerating an individual kernel does not necessarily produce the same speedup when running the complete application [3]. This review aims to relate design characteristics to workload requirements and clarify both what the study results support and what limits their generalisability. To this end, it distinguishes simulation results from measurements on physical hardware, and computational-unit evaluations from system or complete-application evaluations. Through this analysis, the paper seeks to provide a comparison that supports an understanding of architectural trade-offs in both domains and identifies aspects requiring further documentation or evaluation.

RISC-V Instruction Set and Core Design

RISC-V is an open instruction set that specifies the instructions available to software and the expected behaviour of a processor [1]. How these instructions are implemented depends on the processor's internal design, including its pipeline organisation, arithmetic units, and caches. Different cores may therefore implement the same instruction set while differing

Copyright *Author (s) 2026. Distributed under Creative Commons CC BY 4.0*
Received*: 28-08-2026 -* ***Accepted****: 26-10-2026 -* ***Published****: 04-11-2026*

in performance, energy consumption, and area [3]. This separation allows designs to address different requirements, from resource-constrained devices to systems with high computational capabilities. RISC-V extensions also allow functions such as floating-point arithmetic, vector processing, and application-specific instructions to be added. However, an open instruction set does not mean that every processor based on it has an open-source implementation; this depends on the terms under which each design is made available [1].

Vector Processing and the RVV Extension

A vector instruction describes an operation applied to a group of data elements. For example, a vector addition instruction can add corresponding elements in two lists instead of issuing a separate instruction for each pair. This does not mean that all elements are processed in a single cycle: execution time depends on vector length, the number of execution units, and the ability of memory to supply data [3]. The RISC-V vector extension, known as RVV, provides a model that allows software to work with different vector lengths. Vector-register capacity, the number of active elements in an operation, and the number of lanes that physically execute calculations must be distinguished. Increasing register capacity may help retain and reuse data, while increasing the number of lanes allows more elements to be processed in parallel. However, additional resources may increase design cost without improving performance if the workload is small or memory access is the limiting factor [3]. The designs do not all use the same version of the vector instruction set. Some rely on earlier versions, some support RVV 1.0 or subsets of it, and others use nonstandard vector instructions or add custom instructions. The extension type and version are therefore essential characteristics to record when comparing designs [3].

Edge AI and HPC Requirements

In edge AI, tasks such as image classification and signal processing are executed close to the data source. Processor suitability depends on achieving the required response time within the device's available resources. Constraints may include memory, electrical power, and cost, but their severity differs between a small battery-powered device and a larger edge platform with a continuous power supply [1]. Some inference applications use low-precision data to reduce storage, data movement, and computational cost. The benefit nevertheless depends on hardware and software support for the selected precision and on the effect of quantisation on model quality. Execution speed alone is therefore insufficient without reporting numerical precision and predictive-accuracy results when available [1]. HPC encompasses diverse workloads, including scientific simulation, linear algebra, and data analysis [2]. Some workloads are limited by arithmetic-unit capacity, whereas others are more sensitive to memory bandwidth or access latency. Precision, parallelism, and problem-size requirements also vary across applications. It should therefore not be assumed that all HPC workloads use double precision or benefit equally from additional resources [3]. The two domains overlap in operations such as matrix multiplication and signal transforms [3]. A useful comparison therefore considers actual workload characteristics, including data size, memory-access patterns, and required precision, alongside the broad classification as edge AI or HPC.

Evaluation Metrics and Comparison Boundaries

Performance is measured by task execution time or the number of operations completed per unit time. It may also be expressed in cycles, although a cycle count does not determine elapsed time without the clock frequency. GOPS and GFLOPS must likewise be interpreted according to operation type, precision, and counting convention. Low-precision integer operations cannot be compared directly with double-precision floating-point operations [3]. Electrical power, measured in watts, differs from the energy required to complete a task, measured in joules. A higher-power processor may finish a task sooner, so power alone is insufficient to assess energy consumption. Metrics such as billions of operations per second per watt or inferences per joule are used, but the task and system boundaries included in the measurement must be specified [2] [3]. Design area also depends on fabrication technology and the components included. The area of a single arithmetic lane differs from that of a vector unit or an entire chip. The same applies to energy: some results cover only arithmetic units, whereas others include cores, caches, and additional components [3]. Finally, simulation results, FPGA prototype evaluations, and measurements on fabricated chips must be distinguished [2]. Kernel or model-layer evaluations must also be separated from complete-application evaluations [1]. These differences do not negate the value of the studies, but they determine the conclusions that can be drawn and the extent to which their results can be compared.

Methodology

This paper adopts a focused review of studies on RISC-V vector-processing architectures and their evaluation in edge AI and HPC applications. Studies were collected through web searches and reference tracing, with publisher websites and institutional repositories consulted to verify publication details and obtain full texts. The review is not presented as an exhaustive account of all published studies. Priority was given to studies directly relevant to the comparison that clearly described the design, workloads, or evaluation method. Previous reviews were used to understand the field and identify overlap with this paper. Studies without available full texts were excluded from detailed analysis. When multiple versions of the same work were available, they were compared to avoid treating them as independent studies.

Data extraction covered architectural characteristics, vector instruction-set version, numerical precision, memory organisation, evaluated workloads, software environment, baseline, and methods for measuring performance, energy, and area. The boundaries of the evaluated component were also recorded, distinguishing kernels, layers, and complete applications.

Copyright *Author (s) 2026. Distributed under Creative Commons CC BY 4.0*
Received: *28-08-2026 -* ***Accepted:*** *26-10-2026 -* ***Published:*** *04-11-2026*

The analysis was based on the similarity of evaluation conditions. Quantitative comparisons were used when the data were suitable, and qualitative comparisons when differences in precision, platform, or measurement boundaries prevented reliable numerical conclusions. Information absent from the source was recorded as not reported and distinguished from information whose extraction or verification remained incomplete.

Study Classification and Comparison Framework

Classification Basis

The selected studies differ in their contributions. Some introduce an architectural design, others examine memory and software effects on an existing platform, and previous reviews summarise the literature. This paper separates these roles and then relates their findings without counting an application study as a new processor. Designs are classified according to their architectural direction, recording characteristics such as vector extension, numerical precision, memory organisation, and execution flexibility. These directions overlap: a design may combine scalability with application-specific specialisation. No design is assumed to belong exclusively to edge AI or HPC; its suitability is discussed in light of the available workloads and evaluation conditions.

Included Architectural Designs

Table 1 presents design directions and comparison limitations rather than a ranking from best to worst. Including a design does not imply that it was directly evaluated in both domains. The design version is identified when extracting results because characteristics may change across versions.

Table 1. Included architectural designs, comparison dimensions, and limitations

Design	Architectural direction	Comparison focus	Main limitation
Ara2 [3]	Scalable vector processing supporting RVV 1.0	Lane count, workload size, and balance with the core and memory	Additional resources do not guarantee better performance for every workload
Vitruvius+ [5]	Vector unit supporting RVV 0.7.1 and overlapping computation with memory access	Scientific computation and execution organisation	FPGA results are separated from physical-implementation estimates
Spatz [6]	Compact vector units in a cluster with shared local memory	Register capacity and data reuse	Unit-level and cluster-level evaluation boundaries differ
Spatzformer [7]	Runtime reorganisation of core-to-vector-unit connections	Mixed workloads and synchronisation overhead	Gains depend on the execution of concurrent scalar and vector work
SPEED [8]	Custom instructions, multiprecision arithmetic, and neural-network dataflows	Precision, layer type, and data movement	Evaluation configurations differ
Sparq [9]	Vector processing specialised for low-precision convolution	Computational specialisation and custom-instruction cost	Reported area and energy estimates cover a single lane
FWHT-RVV [10]	Custom instructions and FWHT algorithm optimisation	Reduced data movement and acceleration of a specific kernel	Kernel acceleration does not equal acceleration of a complete LLM
Schmidt et al.'s chip [11]	Rocket cores with Hwacha vector units	Precision flexibility, chip measurements, and energy–frequency trade-offs	Uses a nonstandard extension rather than implementing RVV 1.0

Platform and Application Evaluation Studies

Six additional studies were used to interpret workload, software, and memory effects. These include Polara-Keras2c for deploying AI models [12], the study by Garcia et al. on inference using SG2042 [13], and the study by Martínez et al. on quantised inference on ARM and RISC-V [14]. The group also includes the study by Vizcaino et al., which examines vector length and memory[15], the Earth sciences study [16], and the study of FFT on long-vector architectures [17]. These studies help explain how design characteristics affect task performance, but they do not all represent new, independent architectures. Some also share a platform or architectural family. This relationship is considered when interpreting the breadth of the evidence, and their results are not treated as evaluations of entirely independent platforms. The reviews by Agosta [1], Kalapothas [4], and Silvano [2] are used for background and to position this paper's contribution; they are not counted as independent experiments in the results table. The two SPEED versions are treated as versions of related work, with their differences documented and repeated results not counted twice [8] [18].

Copyright *Author (s) 2026. Distributed under Creative Commons CC BY 4.0*
Received*: 28-08-2026 -* ***Accepted****: 26-10-2026 -* ***Published****: 04-11-2026*

https://doi.org/10.54361/ajmas.261110

Comparison Framework

The comparison is conducted at two related levels: design characteristics and the conditions underlying experimental evidence. Particular attention is given to result comparability and to whether kernel-level gains translate into layer or complete-application performance when such data are available. The first level covers the vector extension, numerical precision, execution organisation, and memory. The second covers workload type and size, compiler and libraries, baseline, evaluation method, and the boundaries of the measured component. It distinguishes measurements on physical hardware from simulation results and estimates based on models or assumptions. To organise the analysis, comparisons are divided into three categories. A comparison is direct quantitative when influential evaluation conditions agree or are controlled within a common experiment.

It is conditional quantitative when disclosed differences remain that limit interpretation, without providing a basis for a general ranking of designs. Qualitative comparison is used when the data do not support a reliable numerical comparison. Information not reported in the source is distinguished from information whose extraction or verification remains incomplete. Comparisons within a study are also separated from comparisons across studies because using the same metric does not ensure equivalent evaluation conditions. This framework supports the interpretation of trade-offs rather than reliance on peak figures alone. A higher operation rate may reflect lower precision or greater computational specialisation [8], while energy efficiency may reflect a low-frequency operating point or measurement of only part of a system [11]. The application-level effect of accelerating a specific kernel may also remain limited if that kernel accounts for only a small fraction of execution time [10]. Each result is therefore linked to its conditions and scope before its relevance to edge AI or HPC requirements is discussed.

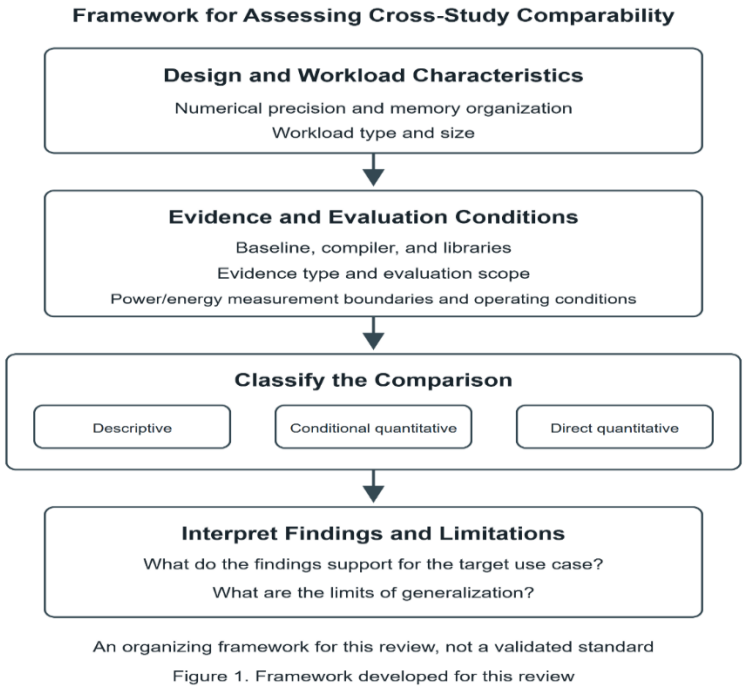


Figure 1. The comparison framework followed in this review.

Results

From Kernel Acceleration to Application Performance

The selected studies show that evaluation scope affects the interpretation of performance gains. Accelerating a kernel or a set of layers does not necessarily yield the same speedup for a complete application. The extent to which the gain carries over depends on the optimised portion's share of execution time, the cost of the remaining operations, data transfers, and coordination between units [10]. SPEED provides an example of this difference within a single evaluation configuration. Using INT8, VGG16 convolution layers achieved a speedup of approximately 6.11 times over the Ara baseline, while the complete application achieved 5.84 times [8].

For MobileNetV2, the speedup was 144.25 times for convolution layers and 100.81 times for the complete application [8]. These results indicate that operations outside the optimised portion limit the overall gain. The values remain tied to the study's simulation configuration and baseline and do not represent a general comparison with all vector processors. FWHT-RVV illustrates the importance of distinguishing a result's scope from the method used to obtain it. The study reported an average speedup of 2.55 times for FWHT kernels over the vector baseline, whereas the estimated overall performance improvement for one Llama2-7B layer was approximately 1.1 times [10]. However, the layer estimate partly relies on timings obtained from LLMCompass and is therefore not treated as a measurement of complete-model inference on a physical chip [10]. The two values are also not used to calculate a unified speedup-transfer ratio because their evaluation scopes and methods differ. For HPC workloads, the Earth sciences study reports speedups of up to 32.6 times for batched matrix-

Copyright *Author (s) 2026. Distributed under Creative Commons CC BY 4.0*
Received: *28-08-2026 -* ***Accepted:*** *26-10-2026 -* ***Published:*** *04-11-2026*

multiplication tests at selected sizes, whereas the improvement after integrating the changes into the computeLocalIntegration region was approximately 1.81 times [16]. The two results use different scopes and baselines. It is therefore inappropriate to calculate a direct transfer ratio between them or to interpret either as a speedup for the complete SeisSol application. They nevertheless demonstrate the need to trace an optimisation's effect after integration into a broader software context. Accordingly, this review separates kernel-level, layer or software-region, and complete-application results. Numerical comparisons across these levels are used when their conditions agree, while interpretation remains qualitative when baselines or evaluation methods differ. This separation prevents local gains from being generalised to broader uses that the study did not evaluate.

Effects of Numerical Precision and Specialisation on Performance and Output Quality

The selected studies identify numerical precision and computational specialisation as important factors in interpreting performance. Designs targeting quantised inference, such as SPEED and Sparq, benefit from representing data with few bits and organising arithmetic units around the targeted operations [9] [8]. However, comparing their operation rates with designs supporting double-precision floating-point arithmetic, such as Ara2 and the included Spatz version [3] [6], is insufficient to assess efficiency. The meaning of an operation, the required resources, and output-quality requirements differ. Sparq illustrates the combined effects of quantisation and architectural specialisation. In convolution tests, speedups were approximately 1.7 times for four-bit data and 3.2 times for two-bit data relative to an optimised convolution implementation using 16-bit data [9]. The performance tests include runtime data-packing overhead, which must be considered when evaluating compressed representations. These results nevertheless concern only the tested convolutions; they do not independently demonstrate complete-model acceleration or preserved model accuracy after quantisation. The area and energy estimates also concern a single lane after removal of the floating-point unit, so differences in cost cannot be attributed to data width alone. SPEED provides results that connect performance with inference quality in specific cases. In its model-evaluation table, VGG16 Top-1 classification accuracy decreased from 79.33% with FP32 to 78.33% with INT8, a reduction of one percentage point [8].

For MobileNetV2, accuracy decreased from 78.00% to 77.33%, a reduction of 0.67 percentage points [8]. These values show that performance gains are accompanied by changes in output quality within the study's configuration. These accuracy results cannot be extended to INT4 tests, nor can the reduction be assumed acceptable for every application. Suitability depends on task requirements and the quantisation and evaluation conditions. Garcia et al. show that a smaller numerical representation does not guarantee shorter execution time when it is poorly matched to the hardware and libraries. On the evaluated SG2042 platform, insufficient BF16 support required alternative execution paths, and converting some models to FP32 could improve performance despite increasing representation size [13]. This finding remains specific to the platform and software environment studied, but it highlights the need to record actual numerical-precision support rather than nominal data size alone. In HPC applications, the trade-off also concerns the numerical-accuracy requirements of scientific computation. FP64 support in the included designs enables the evaluation of workloads that differ from quantised inference, but it does not independently establish the accuracy of a complete scientific application. Conversely, high INT4 or INT8 operation rates do not demonstrate that a specialised design is a suitable alternative for calculations requiring FP64. Suitability is therefore discussed according to workload type and required precision, without assuming that all edge AI applications use low precision or that all HPC applications require the same precision. This review therefore relates performance results to data width and type, accumulation precision when reported, output quality, and hardware and software support. It also distinguishes the effects of reduced precision from those of custom instructions and changes to arithmetic units. When a study does not isolate these factors, the improvement is interpreted as their combined outcome. This allows the value of specialisation to be assessed against task requirements rather than treating the highest operation rate as sufficient evidence of superiority.

Effects of Memory and Software on Vector-Processing Utilisation

The selected studies show that increasing vector resources does not guarantee better performance unless the workload provides sufficient work and an appropriate flow of data. In Ara2, lane utilisation depends on the work available per lane and the balance with the scalar core and memory [3]. On an FPGA platform with selected HPC workloads, the study by Vizcaino et al. shows that changing vector length, memory latency, and memory bandwidth changes execution behavior [15]. Scaling effects therefore cannot be explained by the number of arithmetic units alone, and the superiority of a particular vector length cannot be generalised to all workloads. The importance of software is evident in the FFT and Earth sciences studies, where vector-processing benefits interact with algorithm selection, data layout, and the ability of compilers and libraries to exploit hardware. The RISC-V baseline in the FFT study uses a non-vectorised FFTW implementation [17], and the OpenBLAS configuration in the Earth sciences study does not provide vectorisation for the evaluated version [16]. The reported speedups therefore combine software optimisation with vector processing rather than isolating architectural superiority alone. For inference, Martínez et al. show that optimising multiplication operations may expose the cost of other operations, including Softmax, LayerNorm, GELU, and quantisation [14]. Together with Garcia et al.'s findings on libraries and arithmetic intensity [13], this evidence indicates that design suitability requires examination of the actual execution path. Hardware characteristics are therefore related to software configurations and workload size, and software changes are distinguished from architectural changes whenever the experiments permit.

Copyright *Author (s) 2026. Distributed under Creative Commons CC BY 4.0*
Received: *28-08-2026 -* ***Accepted:*** *26-10-2026 -* ***Published:*** *04-11-2026*

Limits of Comparing Performance and Energy Efficiency

The available evidence includes design simulation, FPGA evaluation, and measurements on fabricated chips [2]. These methods answer different questions: simulation enables exploration of architectural configurations, while physical measurements characterise execution on a specific platform. Implementation type alone, however, does not determine comparability, which also requires agreement in workload, precision, baseline, measurement boundaries, and operating conditions [3]. This issue is particularly apparent in energy results. Schmidt et al. measure cluster energy at different voltage and frequency points [11], whereas Polara-Keras2c uses an estimate based on power assumptions from a previous study [12]. In Martínez et al., energy measurements apply only to the ARM platform [14]. These results therefore cannot support a unified ranking of RISC-V processor energy efficiency. Unreported energy must not be treated as zero or inferred from execution time alone. Examining the SPEED configurations shows that the best performance or efficiency figures may concern a configuration different from the main comparison, and some comparisons involve projections across fabrication technologies [8]. The operating point, design configuration, and estimation method must therefore accompany each result, and peak values from separate configurations must not be combined into a single description. The available evidence supports conditional conclusions about design trade-offs, but not the designation of one processor as the best for all edge AI and HPC requirements.

This application shows that numerical values alone are insufficient for a reliable quantitative comparison. Differences in precision, workload scope, baseline, or energy-measurement boundaries may change what a result supports, making conditional or qualitative interpretation more appropriate than ranking designs by their reported figures.

Table 2. The application of the comparison framework to selected cases from the included studies. Classification concerns the specific comparison and its conditions, rather than the entire study or its overall quality.

Specific comparison	Comparison type	Classification basis and limits of inference
Convolution-layer versus complete-application speedup in SPEED for the same model at INT8	Direct quantitative within the study, with the two evaluation scopes distinguished	The study reports both results against the Ara baseline within its specified configuration. The comparison identifies differences in gains across the two scopes without generalising to other models or platforms. [8]
Low-precision convolution in Sparq versus an optimised 16-bit implementation	Conditional quantitative	Speedup ratios reflect performance differences when precision and computational implementation change together. They do not isolate architecture alone or establish equivalent complete-model quality after quantisation. [9]
FWHT kernel speedup versus estimated LLM-layer improvement in FWHT-RVV	Qualitative across evaluation scopes	The workloads and methods used to obtain the results differ, and the layer estimate partly depends on LLMCompass. The case illustrates the limits of generalising kernel gains without calculating a direct transfer ratio to the layer. [10]
Batched matrix-multiplication speedup versus computeLocalIntegration-region improvement in the Earth sciences study	Qualitative across the two scopes	Work boundaries and baselines differ. The ratios are therefore not used to calculate a unified transfer of speedup, and neither measures complete SeisSol performance. [16]
Energy-efficiency results for Schmidt et al.'s chip versus energy estimates in Polara-Keras2c	Qualitative with respect to evaluation method	The evidence types and measured system boundaries differ: the former includes cluster-energy measurements, while the latter relies on power assumptions. These data do not permit a direct numerical ranking of the two platforms' efficiency. [11] [12]

Discussion

Implications for Design Selection

The comparison indicates that a vector processor's suitability depends on how its capabilities match the target workload and operating environment. For edge AI inference, response time, memory, energy, and output quality are important. Specialisation for quantised arithmetic can be beneficial when supported by the tools and when the required quality is maintained [1] [8]. In the HPC workloads examined, data flow, vector length, data reuse, and floating-point support are important [6] [15]. These requirements overlap: the domain label alone determines neither suitable precision nor the best architectural organisation. In practice, evaluating a design begins by identifying the task, its size, and its output-quality requirements, then seeking evidence obtained under similar conditions. A complete-application result is more directly relevant to that application's performance than an individual-kernel result, while kernel tests help explain strengths and bottlenecks. When precision, libraries, or energy-measurement boundaries differ, conclusions must remain conditional on those differences. The available evidence therefore does not support a general recommendation of one design for every use, but it helps narrow the choices and identify what requires further testing.

Copyright *Author (s) 2026. Distributed under Creative Commons CC BY 4.0*
Received: *28-08-2026 -* ***Accepted:*** *26-10-2026 -* ***Published:*** *04-11-2026*

Analytical Contribution of the Review

The analytical contribution of this review is to connect design characteristics with the scope of results and the conditions under which they were obtained. Rather than combining peak performance and energy-efficiency figures into one ranking, the analysis clarifies whether a result concerns a kernel, a layer, or an application; whether it comes from simulation or physical measurement; and what the baseline and energy-measurement boundaries include. This connection identifies cases permitting numerical comparison and those useful primarily for interpreting trade-offs. Precision, memory, and software are not presented as newly discovered factors; they have been discussed in original studies and previous reviews [1] [3]. The value of the present synthesis lies in tracing their effects on the interpretation of selected evidence across edge AI and HPC, while distinguishing application studies from new designs and related versions from independent evidence. The direct, conditional, and qualitative comparison categories remain an organisational tool for this review and have not been validated as an independent standard for assessing study quality.

Conclusion

This paper presented a focused review of RISC-V vector processor designs and their evaluation studies in edge AI and HPC, relating architectural characteristics to the conditions underlying published evidence. The analysis showed that design suitability depends on workload characteristics, numerical precision, memory organisation, and software support, and that reported operation rates or energy efficiency alone are insufficient to judge performance in the intended use. The comparison highlighted the need to distinguish kernel acceleration from complete-application improvement, and physical measurements from simulation results and estimates, while specifying baselines and energy-measurement boundaries. The selected studies support analysis of trade-offs under specific conditions, but they do not provide an equivalent basis for ranking all designs or identifying a single processor as the best for both domains. The value of this review lies in organising these conditions and explaining how they affect result interpretation and the evaluation of architectural choices. The conclusions remain limited to the selected studies, whose configurations differ and some of whose works belong to shared architectural families. Future comparisons would benefit from common workloads and baselines, documented tool versions, precision, operating points, and energy boundaries, and the evaluation of output quality alongside performance. Combining kernel tests with complete-application evaluations also helps establish how far architectural improvements translate into practical benefits.

Conflict of interest. Nil

References

1. Agosta G, Galimberti A, Zoni D. Deep learning on RISC-V platforms at the edge: a perspective on the hardware and software support. ACM Comput Surv. 2025 Nov;58:121:1–121:37.
2. Silvano C, Ielmini D, Ferrandi F, Fiorin L, Curzel S, Benini L, Conti F, Garofalo A, Zambelli C, Calore E, Schifano S, Palesi M, Ascia G, Patti D, Petra N, De Caro D, Lavagno L, Urso T, Cardellini V, Cardarilli GC, Birke R, Perri S. A survey on deep learning hardware accelerators for heterogeneous HPC platforms. ACM Comput Surv. 2025 Jun;57(11):286:1–286:39.
3. Perotti M, Cavalcante M, Andri R, Cavigelli L, Benini L. Ara2: exploring single- and multi-core vector processing with an efficient RVV 1.0 compliant open-source processor. IEEE Trans Comput. 2024 Jul;73(7):1822–36.
4. Kalapothas S, Galetakis M, Flamis G, Plessas F, Kitsos P. A survey on RISC-V-based machine learning ecosystem. Information. 2023 Jan 21;14(2):1–20.
5. Minervini F, Palomar O, Unsal O, Reggiani E, Quiroga J, Marimon J, Rojas C, Figueras R, Ruiz A, Gonzalez A, Mendoza J, Vargas I, Hernandez C, Cabre J, Khoirunisya L, Bouhali M, Pavon J, Moll F, Olivieri M, Kovac M, Kovac M, Dragic L, Valero M, Cristal A. Vitruvius+: an area-efficient RISC-V decoupled vector coprocessor for high performance computing applications. ACM Trans Archit Code Optim. 2023 Feb;20(2):28:1–28:25.
6. Perotti M, Riedel S, Cavalcante M, Benini L. Spatz: clustering compact RISC-V-based vector units to maximize computing efficiency. IEEE Trans Comput Aided Des Integr Circuits Syst. 2025 Jul;44(7):2488–502.
7. Perotti M, Raeber M, Sinigaglia M, Cavalcante M, Rossi D, Benini L. Spatzformer: an efficient reconfigurable dual-core RISC-V V cluster for mixed scalar-vector workloads. In: 2024 IEEE 35th International Conference on Application-specific Systems, Architectures and Processors (ASAP); 2024; Hong Kong. p. 172–3.
8. Wang C, Fang C, Wu X, Wang Z, Lin J. SPEED: a scalable RISC-V vector processor enabling efficient multiprecision DNN inference. IEEE Trans Very Large Scale Integr VLSI Syst. 2025 Jan;33(1):207–20.
9. Dupuis T, Fournier Y, AskariHemmat M, El Zarif N, Leduc-Primeau F, David JP, Savaria Y. Sparq: a custom RISC-V vector processor for efficient sub-byte quantized inference. In: 2023 21st IEEE Interregional NEWCAS Conference (NEWCAS); 2023; Edinburgh, United Kingdom. p. 1–5.
10. Lv L, Chao N, Xiong D, Huang K. FWHT-RVV: a RISC-V vector processor with FWHT instruction extension for edge LLM inference. IEICE Electron Express. 2026 Jun 25;23(12):1–6.
11. Schmidt C, Wright J, Wang Z, Chang E, Ou A, Bae W, Huang S, Milovanović V, Flynn A, Richards B, Asanović K, Alon E, Nikolić B. An eight-core 1.44-GHz RISC-V vector processor in 16-nm FinFET. IEEE J Solid-State Circuits. 2022 Jan;57(1):140–52.
12. El Zarif N, Askari Hemmat M, Dupuis T, David JP, Savaria Y. Polara-Keras2c: supporting vectorized AI models on RISC-V edge devices. IEEE Access. 2024 Nov 14;12:171836–52.
13. Marques Garcia A, Malenza G, Birke R, Aldinucci M. Inference performance of large language models on a 64-core RISC-V CPU with silicon-enabled vectors. Future Gener Comput Syst. 2026 Apr;177:1–15.
14. Martínez H, Catalán S, Castelló A, Mestre JI, Quintana-Ortí ES. Latency-critical quantized inference with transformer decoders on ARM and RISC-V CPUs. IEEE Internet Things J. 2025 Jul 1;12(13):25676–90.

Copyright *Author (s) 2026. Distributed under Creative Commons CC BY 4.0*
Received: *28-08-2026 -* ***Accepted:*** *26-10-2026 -* ***Published:*** *04-11-2026*

15. Vizcaino P, Ieronymakis G, Dimou N, Papaefstathiou V, Labarta J, Mantovani F. Short reasons for long vectors in HPC CPUs: a study based on RISC-V. In: Proceedings of the SC '23 Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis; 2023; Denver, CO, USA. p. 1543–9.
16. Banchelli F, Jurado D, Garcia-Gasulla M, Mantovani F. Exploring RISC-V long vector capabilities: a case study in Earth Sciences. Future Gener Comput Syst. 2026 Jan;174:1–15.
17. Vizcaino P, Mantovani F, Ferrer R, Labarta J. Acceleration with long vector architectures: implementation and evaluation of the FFT kernel on NEC SX-Aurora and RISC-V vector extension. Concurr Comput Pract Exp. 2023 Sep 10;35(20):1–14.
18. Wang C, Fang C, Wu X, Wang Z, Lin J. A scalable RISC-V vector processor enabling efficient multi-precision DNN inference. In: 2024 IEEE International Symposium on Circuits and Systems (ISCAS); 2024; Singapore. p. 1–5.

Copyright *Author (s) 2026. Distributed under Creative Commons CC BY 4.0*
Received*: 28-08-2026 -* ***Accepted****: 26-10-2026 -* ***Published****: 04-11-2026*