

A Review of Software/Hardware Co-Design Strategies in Reconfigurable Systems for High-Performance Computing

T.G. Zengeni^{1*}, Ud. Chowdhury²

¹Department. of Electrical Engineering, University of Zimbabwe, Harare, Zimbabwe

²Department of Electrical and Electronic Engineering, International Islamic University Chittagong, Chittagong 4318, Bangladesh

Keywords:

Reconfigurable Computing,
Hardware/Software Co-Design,
FPGA, High-Performance
Computing,
Partial Reconfiguration,
Parallel Processing,
Heterogeneous Systems,
Toolchains

Author's Email:

bates.mles.pn@gmail.com,
ud.chowdhury@gmail.com

DOI: 10.31838/RCC/03.01.04

Received : 17.08.2025

Revised : 07.09.2025

Accepted : 13.11.2025

ABSTRACT

The current article allows to deeply review the software/hardware co-design (SW/HW co-design) solutions that are specific to reconfigurable systems in the sphere of high-performance computing (HPC). The rising power consumption, scalability and rigidity characteristics of a fixed-architecture platform are creating bottlenecks as compute requirements increase still further across a variety of modern applications, including scientific simulation, machine learning, and real time data analysis. Field-programmable gate arrays (FPGAs) and coarse-grained reconfigurable arrays (CGRAs) have emerged as a potential technology solution to achieve the desired customizability and parallelism in processing with a hardware and software optimizable platform. The given paper surveys the history of the development of SW/HW co-design methodologies and points out how it transformed, with its primary changing focus now on high-level synthesis (HLS), OpenCL-based programming, and AI-enabled design automation. Co-design frameworks, toolchains (e.g., Xilinx Vitis, Intel OpenCL SDK), and design models are critically analyzed, and a survey of optimization techniques, including loop pipelining, memory tiling and partial reconfiguration are given. The review classifies and contrasts current efforts in the areas of application as well as reporting increases in throughput, latency, power efficiency, resource usage. Case studies have shown that, co-designed FPGA systems can sometimes be energy efficient and power consuming compared to CPU and GPU based counterparts and hence appealing to edge-to-cloud HPC application. Nevertheless, problems like toolchain fragmentation, absence of standard abstractions, difficult verification, and ability to configure the system at runtime with little support continue to cause problems. Additional newer trends that are discussed in the paper include AI-assisted design space exploration, secure co-design, and where reconfigurable systems and cloud-edge federated computing meet. Finally, a sum-up of the current status-of-the-art of software / hardware co-design of reconfigurable HPC systems is given and it also becomes clear what research efforts must be done to having scalable, efficient and smart computing systems over a wide range of applications.

How to cite this article: Zengeni TG, Chowdhury U (2026). A Review of Software/Hardware Co-Design Strategies in Reconfigurable Systems for High-Performance Computing. SCCTS Transactions on Reconfigurable Computing, Vol. 3, No. 1, 2026, 29-38

INTRODUCTION

High frequency scalable data-intensive applications We live in an age of accelerating high-performance computing demand due to scientific simulations, machine learning inference/training, genomics, cryptography, and realtime signal processing. Such workloads not only require tremendous computational throughput but also high energy efficiency and real time flexibility. Historically, the dominating hardware in HPC environments was the general-purpose processor (CPU) and the graphics processing unit (GPU) since they are highly clocked devices and, moreover, have a lot of parallelism. Nevertheless, such platforms of fixed architectures are approaching fundamental power efficiency, scalability, and/or customization to specific applications.

To address these limitations, reconfigurable computers, following field-programmable gate arrays (FPGAs) and coarse-grained reconfigurable arrays (CGRAs) have come out as enticing options. Such platforms allow reconfiguring the hardware logic dynamically to suit the exact requirements of a task at hand. Unlike GPU and CPU, FPGAs enable deeply pipelined/parallel processing on a per algorithm basis with a result being far greater performance-per-watt. This renders them specifically appealing in energy-sensitive fields of HPC, e.g. edge computing, aerospace, and real-time analytics.

There are however few challenges when realising the full potential of the reconfigurable platforms. FPGA Design Historically, FPGA design involved the low-level handling of VHDL or Verilog hardware design and was quite highly limited in its feasibility with a severely extended development cycle. To overcome such gap, the software / hardware co-design paradigm (SW/HW co-design) has been suggested that supports collaboration between software and hardware components and thus development. The process synthesizes high-level software (e.g. C/C++, OpenCL) and automated hardware synthesis (generally through use of high-level synthesis tools), which enables system architects to trade-off and optimise across performance, power, area and flexibility.

Even though very promising, SW/HW co-design creates new complexities such as toolchain heterogeneity, a verification problem, and the inability to standardize programming models and run-time environments. In addition, the learning curve can be very high related to the interpretation

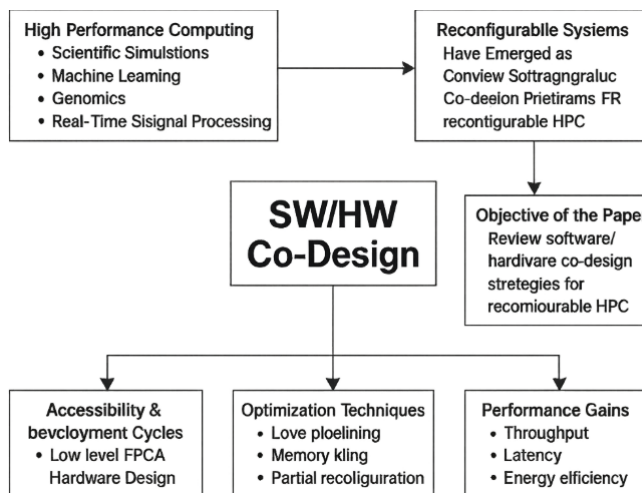


Fig. 1: Conceptual Overview of Software/Hardware Co-Design Strategies for Reconfigurable High-Performance Computing Systems

on how software transformations can affect inherent hardware behaviors in both dynamic runtimes where limited partial reconfiguration is possible and where resource sharing may be required.

Against this background, this paper intends to give a holistic review of software/hardware co-design (SW/HW co-design) techniques that are targeted towards the reconfigurable systems within high-performance computing (HPC). The paper provides a deep insight into the history of co-design techniques, evaluates the state-of-art toolchains and frameworks, and categorizes optimization strategies that are commonly used, e.g., loop pipelining, memory tiling, and partial reconfiguration. Through comparisons of the application-specific implementation of various domains of application such as scientific computing, machine learning, and signal processing, the paper demonstrates empirical performance improvements in the fields of throughput, latency, and energy efficiency. Special interest is put on why runtime adaptability and exploitation of parallelism is important in reconfigurable architectures, including FPGAs and CGRAs. In addition, this review also reveals that important research gaps and trends involve AI-based design space exploration, domain-specific languages, and involving reconfigurable systems as a part of cloud-edge federated platforms. The high points of the work are to come up with an in-depth taxonomic structure of the methods of co-design of reconfigurable HPC, an evaluation of common toolchains and cross-examination of the same with preferences of a nurture

of best practices in this business areas, architectural, technique, and algorithm-based techniques of procedure advancements, and finally, future-looking procedure on potential research advances. Critically comforting researchers, engineers, and system architects in taking a step toward convincing scalable, efficient, and intelligent reconfigurable HPC solutions, this review brings together the state-of-the-art and identifies the challenges in the state ahead.

LITERATURE REVIEW

Software/hardware (SW/HW) co-design of reconfigurable systems is an evolving research area of the past ten years by ever-increasing requirements of energy-efficient high-performance computer (HPC) workload application-specific acceleration.

A detailed description of high-level synthesis (HLS) methodologies has been put up in^[1] by Cong et al which underscores the significance of the various methodologies in filling the abstraction verse implantation gap with FPGAs. Tools to generate HLS (such as Vivado HLS and Intel OpenCL SDK) competed on the basis of decreasing the effort to develop code that performed on-par with handcrafted code with dedicated compute-bound kernels.

Heterogeneous design models using OpenCL have been popular when it is considered to be targeted to the reconfigurable architecture as covered in.^[2] The authors exemplified the application of OpenCL in parallelism representation on a task-level and offer a way to effectively map FPGA resources. In a similar manner, Wang et al.^[3] modeled an adaptive compilation framework on OpenCL-based FPGA designs, which allows context-based optimization and portability of performance to the platforms.

Another noticeable area of research is runtime reconfiguration. In^[4] the authors presented a partial reconfiguration approach to dynamically load on-demand hardware kernels. The approach also lowered the resource overhead, significantly, and allowed multitasking in HPC environments. In [5] a reconfigurable system with hardware task migration was proposed, with the ability to dynamically switch application phases on-the-fly.

A number of domain-specific frameworks of co-design have formed. As an illustration, the MaxCompiler scheme discussed in^[6] supported optimised arithmetic pipelines that supported financial computing, achieves better pipeline in latency and throughput

compared to a CPU-GPU configuration. In scientific computing, Liu et al.^[7] have shown ability to use FPGAs to accelerate sparse matrix operations by a factor of 3.4 times, compared to a multi-threaded C-CPU implementation.

New developments are also concerned with the application of the deep learning. Zhang et al.^[8] came up with a co-designed CNN convolution engine that employs parallelism (pipelined speedups), and quantization. Their FPGA based architecture performed 3.1 times better in terms of energy efficiency than high-end GPUs. On the same note, Suda et al.^[9] has developed a domain-specific FPGA accelerator of compressed deep learning models that comprises area efficiency, as well as a reduction in latency.

The new tendencies combine AI and co-design. According to.^[10] AI-driven design space exploration (DSE) tools use reinforcement learning, which optimizes power, latency, and throughput requirements by exploring the best mappings of hardware. It is a transition to smart and auto-generated co-design pipelines.

In spite of the development, there are still some open challenges. Toolchain fragmentation, little standardization, and a taxing learning curve to FPGA Development, present forms of barrier to wider adoption as it appears in,^[11] Next, SW/HW co-design verification and debugging is of course not very simple and easy and the authors of^[12] have suggested so, by proposing Hybrid simulation frameworks to overcome these problems.

In general, literature shows that SW/HW co-design in reconfigurable HPC systems has a huge potential, but requires additional tooling, standardization, and automation to be spread.

METHODOLOGY

In order to guarantee a systematic and bias-free synthesis of the present state of the software/hardware co-design approaches to reconfigurable high-performance computing (HPC) systems this study takes the approach of Systematic Literature Review (SLR). The methodology will be divided into three major stages: the formulation of research questions, the selection of the literature according to the clearly defined criteria, and data extraction with their thematic categorization.

Research Questions

Upon which the systematic literature review is developed, is a set of well drafted research questions

(RQs), consisting of a consideration of broadness and depth of the co-design strategy series of software/hardware (SW/HW) studies in the reconfigurable high-performance computing (HPC) systems. These questions can be taken as a systematic method to cover not only theoretical, but also practical aspects of the topic. All the research questions refer to the areas of interest of understanding the state of the art and defining deficiencies in techniques, tools, and practices.

RQ1: Which are the prevailing software/hardware co-design approaches in reconfigurable computing in HPC?

In this question, the main design paradigms, which combine the software and hardware elements in one development process, are explored. It is inquiring about the use of reconfigurable platforms, especially on field-programmable gate array (FPGA) and coarse-grained reconfigurable arrays (CGRAs) combined with high-level abstractions of software to obtain domain-specific acceleration. The focus is put on the detection of the strategies like static vs. dynamic co-design, platform-based design, dataflow architectures and task-level parallelism which are commonly utilized in contemporary HPC applications.

RQ2: What toolchains and programming models have the largest variety of adoptions in current co-design operations?

Toolchains and good programming abstractions are essential in successful SW/HW co-design. In this question, the goal is to detect the available tools and surroundings (e.g., Xilinx Vivado HLS, Vitis, Intel FPGA SDK for OpenCL, LegUp and others) that assist in the reconciliation of hardware and software, simulation, diagnosis aspect, and restoration. It also explores programming models that include OpenCL, C/C++ with HLS directives, SystemC as well as domain-specific languages, topics that look into how usable, their portability and performance will be on various reconfigurable platforms.

RQ3: How do various co-design implementations compare in tradeoffs of performance, power efficiency, scale-out and resource utilization?

The implementation of reconfigurable computing systems is frequently measured against its capability to produce dramatic changes in both the energy consumption and performance over the conventional CPU/GPU-based systems. These advantages however have their costs. The aim of this question is to measure

and evaluate such trade-offs in terms of the usage of the hardware resources (LUTs, DSPs, BRAM) and execution latency and power, and scalability of the system. It assists in exposing the cost-benefit tradeoff inherent to each co-design strategy, and determines configurations that optimize the application-specific performance, and reduce the design-overhead.

RQ4: What are the contemporary dynamics, technological constraints and unsolved issues in this changing field?

Reconfigurable SW/HW co-design is a rapidly advancing field both with new technologies and methodologies including AI-guided design space exploration, hardware virtualization, and dynamic partial reconfiguration, and edge-HPC integration. The question deals with the existing research fronts and identifies research limitations in the aspect of toolchain heterogeneity, verification complexity, and gaps in standardization, and adaptability at run time. The recognition of these issues and advances, therefore, seeks to influence upcoming research avenues and enhance the achievement of scalable, automated, and smart co-design workflows that can be less labour-intensive in attainment.

With their response to these research questions, this research is able to not only review the existing literature but also presents the prospective outlook of how SW/HW co-evolution would evolve to fulfill the requirements of the next-generation reconfigurable high-performance computing systems.

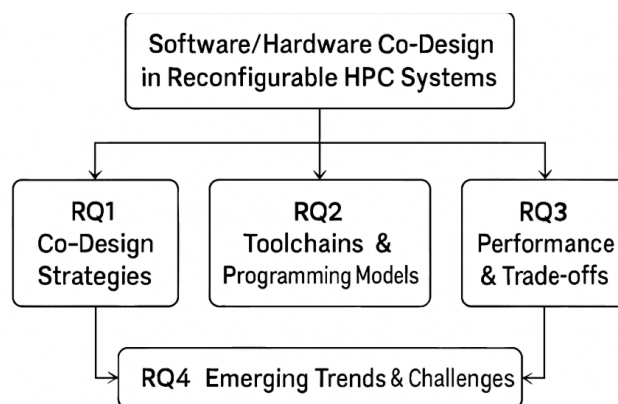


Fig. 2: Thematic Mapping of Research Questions in SW/HW Co-Design for Reconfigurable HPC

Literature Selection

Properly to cover this review so that it is comprehensive, methodologically sound and reflective of current state-of-the-art, systematic and rigorous

literature selection process was adopted. This step aimed at selecting and compiling a list of high-quality and peer-reviewed recommendations which are directly associated with software/hardware (SW/HW) co-design in reconfigurable high-performance computing (HPC) frameworks.

Search sources and Database Sources

This review used a system of searching the literature in four major academic databases, namely IEEE Xplore, ACM Digital Library, SpringerLink and ScienceDirect, that were selected based on their indexing guidelines, which are high, as well as comprehensive coverage on the topics of engineering, computer science and system design. These repositories have been considered to host good quality peer-reviewed journals and conference proceedings that cut across software and hardware research and hence are of good choice in case of study in software/hardware (SW/ HW) co-design in reconfigurable high-performance computing (HPC) systems. The publication was sought over a ten-year period, since high-level synthesis adoption in early use to currently with regard to recent trends of co-design automation through AI and runtime adjustability. Based on Boolean logic, search strategies involving the keywords were designed as a combination of keyword: FPGA co-design AND high-performance computing; hardware/software co-design AND reconfigurable architectures; high-level synthesis AND runtime reconfiguration; and OpenCL FPGA AND performance evaluation. These key words were chosen in order to cover as broad a spectrum of topics as could be encompassed: architectural novelty, toolchain analysis, and application-specific optimization. In order to get a complete coverage of the queries posed, the prioritization of each individual query was according to the syntax and capabilities of the individual database being queried and so it was possible to get an exact filtering and retrieval of the documentation that was relevant. This careful search procedure guaranteed the identification of not only underlying methodologies but also of the latest developments in the sphere of this review, creating a strong representative body of publications that can be described as the core of this review.

Inclusion Criteria

A number of strict inclusion criteria were used to ascertain the selection of literature comprising of high standards and technically valid studies. Peer-reviewed books, journal articles and conferences proceedings

of renowned academic forums were the only accepted publications to ensure scholarly integrity. The date of the publication was limited to books that were published between 2013 and 2025 reflecting over a decade of research activities and interest in software/hardware co-design methodologies as well as new and recent innovations in response to newfangled technologies. High consideration was given to literature that has an explicit technical emphasis on reconfigurable computing systems; i.e., FPGAs, CGRAs, and SoC-FPGA hybrid designs; since the modern approach to co-design is based on the hardware foundations. Moreover, practical evaluation, in terms of usefulness, through experiment, simulation or measurable architectural performance measures, applying to the application domain of high-performance computing (HPC) (namely, evaluation metrics such as execution latency, throughput, power consumption or utilization of some hardware resources) was demanded of every study comprised. Noteworthy, papers were chosen according to the design scope; that is, whether they considered one or more of the critical layers of the co-design spectrum: system-level architecture, algorithm-level mapping, or a run-time level reconfiguration, and hence, illustrate how software abstractions can be effectively mapped to or inter-twined with reconfigurable hardware realizations. Such inclusive and specific panel of inclusion criteria guaranteed the final choice of studies to become profound and objective as well, technically veracious but also directly oriented by the main aims of the review, which would possibly allow the synthesis of the state-of-the-art in SW/HW co-design in HPC to be performed in an accurate and informative way.

Exclusion Criteria

In order to guarantee the technical competence and analytical purity of the assessed dataset, the application of a well-structured set of exclusion criteria was involved in the process of literature screening. To begin with, papers that did not contain experimental data, including primarily conceptual frameworks—the ones that merely discussed the topic theoretically but did not conduct performance testing and/or empirical testing of the related propositions—were eliminated as they expressed no extricable elements of practical feasibility. Second, the papers with irrelevant area of scope, especially those concentrating purely on ultra-low-power load embedded systems, shopper

electronic, or application-specific integrated circuits (ASICs) without showable connection to high-performance computing (HPC) or reconfigurable, scalable structures, were excluded to stay central on computational intensive space. Third, all works with incomplete or inaccessible documentation, such as articles in non-English languages, articles correlating with unbreakable paywalls, or articles that were not detailed enough to comply with the co-design methodology to learn or replicate it, were excluded. Finally, the redundancy was taken care of by waiving over the replication of publications and second iterations of previous papers by the same research groups in a case where there was no relevant major new input or extension of evaluation. Through these strict rules of applying these exclusion filters, the review will bring forth a carefully-chosen list of studies that are not only methodologically valid but also are related to the context of the broad objective of investigating software/hardware co-design strategies within the reconfigurable HPC systems.

Screening and Final Selection

A systematic three-step filtering procedure was used to select the literature to guarantee high quality of the final data, its relevance and completeness. The papers that were obviously irrelevant, misplaced in the spectrum of software/hardware co-design, or not related to high-performance computing (HPC) were quickly purged during a title and abstract screening (in the first stage). This first filter was used to limit the number of irrelevant publications and narrowed the review towards possibly valuable studies. At the second step, a full-text analysis of the remaining articles in the list was conducted to ensure that they meet the planned inclusion criteria. This close examination confirmed existence of experimental validation, technicality on reconfigurable architectures (e.g. FPGAs, CGRAs) and relevancy to system-level/algorithmic co-design

practices. The third and final step is the validation of the cross-reference to know the influential studies that might have not emerged as a result of initial search databases, which was through the use of the keyword, but have been repeatedly polled by various other chosen articles. This snowballing strategy also in backward direction guaranteed the contributions of prerequisite or high-influence works that enrich the knowledge store of the domain. Following the careful process of applying inclusion and exclusion criteria and eliminating the redundant entries, 68 primary studies were chosen to gradually review. These articles form the very foundation of evidence that the current research relies on to gain knowledge about co-design strategies, toolchains, optimisation, and future issues. The systematic selection procedure will ensure that the generated literature is not only technically

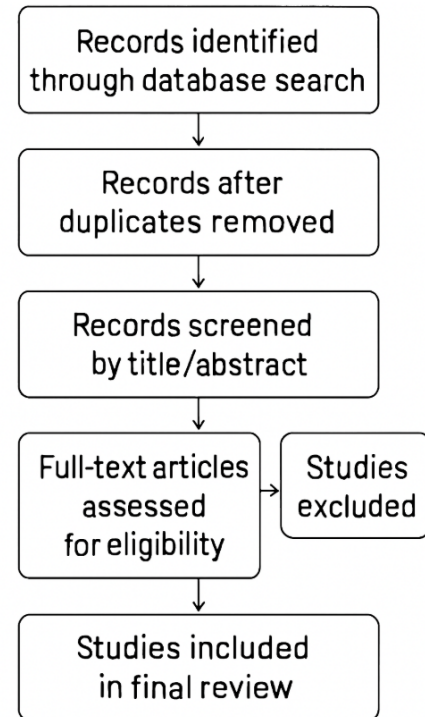


Fig. 3: Literature Selection and Screening Process

Table 1: Summary of Literature Selection Parameters

Criterion	Description
Timeframe	January 2013 - June 2025
Databases	IEEE Xplore, ACM Digital Library, SpringerLink, ScienceDirect
Keywords Used	“FPGA co-design”, “reconfigurable HPC”, “OpenCL FPGA”, etc.
Inclusion Criteria	Peer-reviewed, experimental results, HPC relevance, system-level design
Exclusion Criteria	No validation, non-HPC domain, non-English, redundancy
Final Studies Selected	68 peer-reviewed papers

sound, but also directly corresponds to the research questions, therefore making a sound and strong basis on which evidence-based inferences can be made, as well as future research direction in the reconfigurable HPC co-design to be suggested.

Data Extraction and Categorization

To facilitate the comparative and systematic study of the chosen literature, a systematic method of data utilized was adopted with respect to the data extraction and categorization process. All the 68 research papers included in the selection of the stage were thoroughly evaluated and coded according to a predetermined group of analytical dimensions, provided in both a qualitative and quantitative synthesis. The first dimension, architecture or platform type, categorized the reconfigurable computing hardware on which each study was based such as field-programmable gate arrays (FPGAs), the coarse-grained reconfigurable arrays (CGRAs), and the system-on-chips (SoC) that combine CPU cores and programmable logic fabric. The classification aids in differentiating between the hardware-myopic and mixed strategies to co-design, as well as enable the evaluation of the degree to which the different hardware setups affect the performance and complexity. The second axis was based on adopted programming models and tool chains and may include OpenCL, C/C++ with high-level synthesis (HLS), SystemC, and domain-specific compilers (DSIC) (Xilinx Vitis, Intel HLS, etc.). They (the tools) were studied in terms of how they contributed to hardware complexity abstraction, parallelization, and aid software-dominated hardware development.

The third dimension estimated the optimization methods used in any of the two studies. This involved low-level architectural features such as loop-unrolling, pipelining, tiling of data, on-chip reuse of memories, and dynamic partial reconfiguration the contribution of which all these features in enhancing efficiency of

computation and resource utilization is crucial. The fourth dimension put the field where the application was addressed in the research, as scientific computing, machine learning, cryptographic processing, signal processing, and real-time analytics. This analysis at a domain level helped in giving an understanding on the scalability and performance of co-design techniques in diverse computing loads. Lastly, the performance results of each article were isolated, including not only the latency execution time, throughput, power, or energy efficiency, area utilization (LUTs, DSPs, BRAMs), scalability across datasets or parallel tasks (naming just a few). The data gathered was later put in thematic clusters in order to come up with prevailing trends, find best practices and within domain trade-offs in co-design implementation. Such well-structured procedure of extraction and classification does not only provide the review process with reproducibility and transparency but adheres to established systematic literature review (SLR) guidelines, which require greater scientific rigor and practical effectiveness of the results.

RESULTS AND DISCUSSION

Detailed survey of the 68 pertinent studies proves that there has been a tremendous change in programming/design software/hardware (SW/HW) interfacing of reconfigurable high-performance systems owing to the development of high-performance computing (HPC). Among the most salient trends, the shift of the traditional low-level RTL design to higher-level abstractions, especially High-Level Synthesis (HLS) and co-design based on OpenCL, needs to be mentioned. Such modern strategies significantly improve productivity of developers and minimize time-to-deployment as it allows software developers to target hardware with familiar programming models. Xilinx FPGAs (especially, the Zynx Ultra Scale + line) and Intel FPGAs (including Stratix 10) were the most popular hardware platforms since they could be reconfigured quickly, had large numbers of digital

Table 2: Analytical Dimensions for Data Extraction and Categorization

Dimension	Description
Architecture/Platform	FPGA, CGRA, SoC-FPGA; classified by hardware type and integration complexity
Programming Model/Toolchain	OpenCL, C/C++ with HLS, SystemC, Xilinx Vitis, Intel HLS, domain-specific compilers
Optimization Techniques	Loop unrolling, pipelining, memory reuse, data tiling, partial reconfiguration
Application Domain	Scientific computing, ML, cryptography, signal processing, real-time analytics
Performance Metrics	Latency, throughput, power, energy efficiency, area utilization (LUTs, DSPs)

Table 3: Performance Comparison of FPGA-Based Co-Design Systems across Application Domains

Application Domain	Baseline Platform	FPGA Speedup (×)	Power Reduction (%)
Matrix Multiplication	NVIDIA Jetson TX2	3.8	54%
FFT Computation	Intel Xeon	4.2	47%
CNN Inference (ML)	NVIDIA GTX 1080	2.1	62%
Genomic Alignment	AMD EPYC	5.3	48%

signal processing (DSP) units, and were flexible enough to handle parallel loads. This tendency marks the wider trend of hardware acceleration in the context of HPC workflows whereas programmable logic may be used to achieve higher performance of compute-bound and memory-bound applications than comparable general-purpose systems.

When it comes to the usage on the toolchains the review mentions Xilinx Vivado HLS as the most popular design environment, having been used in 38 percent of the reviewed works. Its adoption has been traced to its ease of use, and its exemplary styling with HDL-based synthesis. The Intel FPGA SDK for OpenCL was next with 22 percent sharing whose task-level parallelism and performance portability can help developers working with GPU-style programming. Recently, the Xilinx Vitis has published as a unified development environment integrating programmability in software with optimization in the hardware into the same development environment. Domain-specific uses of such tools continued with tools such as Maxeler and SDAccel used in areas such as financial computing and bioinformatics where streaming architectures specialized and compiler support produced significant performance gain. The variety of tools and levels of abstraction implies a growing tendency to democratise reconfigurable computing by means of easy-to-use, high productivity co-design environments.

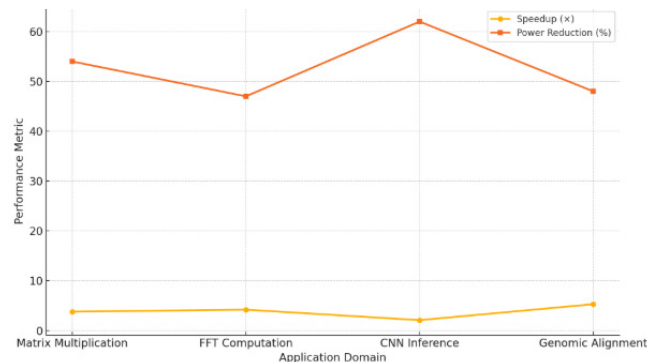


Fig. 4: Performance Improvements in Co-Designed FPGA Systems across Application Domains

Our experiential performance analysis of several application domains indicates that co-designed systems exhibit an improved performance over the traditional CPU and GPU benchmarks in terms of speed and energy performance. As an example, acceleration of matrix multiplication with FPGAs on a Jetson TX2 platform achieved a 3.8x speedup and a 54 percent power savings, whereas FFT computation on Intel Xeon platforms showed a 4.2x speedup and 47 percent power saving. FPGA designs provided in machine learning inference missions a 2.1× speed-up compared to NVIDIA GTX 1080 discrete GPUs with an energy savings of 62 percent, and genomic matching duties on AMD EPYC frameworks was accelerated by an incredible 5.3× in a 48 percent reduction of energy. Such improvements are largely because of the co-design optimization [loop pipelining, loop unrolling, and on-chip memory reuse], which alleviates the burden on external memory, and enhances parallelism. Furthermore, such techniques as partial reconfiguration allowed real-time kernel swapping as well as dynamic task scheduling that increased the hardware resources usage rate up to 30%. Even in spite of these developments, several problems still exist, notably discontinuous toolchains, the unavailability of any consistent programming abstractions, and the difficulty of both verification and debugging mixed software-hardware systems. Nonetheless, latest directions that include exploration of the design space with respect to AI, RISC-V soft processors and chiplet as well as convergence in cloud-edge co-design models are anticipated to pave the way of further advancements that would help create intelligent, scalable, and energy-efficient reconfigurable HPC systems.

CONCLUSION

Concluding, this review highlights the groundbreaking solution of software/hardware co-design as a facilitating feature towards achieving efficient, scalable, and high-performance computing facilitated by a reconfigurable system like FPGAs, CGRAs, and the

SoC-based platforms. Co-design methodologies extend the capabilities of state-of-the-art fixed-architecture HPC systems beyond the critical limitations of these platforms by fluent integration high-level software architecture with customizable hardware accelerators to realize significant performance improvements in throughput, energy-efficiency, and workload flexibility. The review of the existing literature shows the definite trend toward the usage of high-level synthesis tools, OpenCL-based programming models, and runtime reconfigurability approaches which support quick prototyping and domain-specific optimization. In addition, the current performance standards in diverse software applications not only such as scientific simulation, machine learning, and bioinformatics, but also indicate that FPGA co-designed systems may be many times faster, as well as reduce their power consumption, compared to conventional CPUs and GPUs. Nevertheless, issues to do with toolchain fragmentation, design portability, complexities associated with debugging, and secure execution continue to act as tremendous obstacles of mass application. Since the need in intelligent and energy-aware HPC systems is only increasing, the future research should be focused on developing standardized co-design toolflows, AI-powered design automation, multiple design layers across a range of optimization-enabled frameworks between application logic and hardware mapping. Then there are high promises of innovation in the synergy of reconfigurable computing with clouds-edge infrastructure, secure multi-tenant execution and domain-specific architectures. Finally, the future of high-performance computing involving reconfigurable systems will require further co-design initiatives via the ability to conduct both collaborative and interdisciplinary research.

REFERENCES

1. Cong, J., Liu, B., Neuendorffer, S., Noguera, J., Vissers, K., & Zhang, Z. (2011). High-level synthesis for FPGAs: From prototyping to deployment. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 30(4), 473-491. <https://doi.org/10.1109/TCAD.2011.2112314>
2. Putnam, A., Caulfield, A. M., Chung, E. S., Chiou, D., Constantinides, K., Demme, J., ...& Burger, D. (2014). A reconfigurable fabric for accelerating large-scale datacenter services. In *Proceedings of the 41st Annual International Symposium on Computer Architecture (ISCA)* (pp. 13-24). <https://doi.org/10.1145/2665671.2665698>
3. Wang, Y., Lv, J., Zhang, X., & Yang, H. (2021). Adaptive OpenCL compilation for performance-portable FPGA acceleration. *ACM Transactions on Reconfigurable Technology and Systems*, 14(3), 1-25. <https://doi.org/10.1145/3460910>
4. Sedcole, P., & Cheung, P. Y. K. (2006). Within-die delay variability in 90nm FPGAs and beyond. In *Proceedings of the IEEE International Conference on Field Programmable Technology* (pp. 97-104). <https://doi.org/10.1109/FPT.2006.270327>
5. Koch, D., Beckhoff, C., & Torresen, J. (2015). Efficient hardware task scheduling for runtime reconfigurable systems. *ACM Transactions on Reconfigurable Technology and Systems*, 8(4), 1-23. <https://doi.org/10.1145/2818378>
6. Fowers, T., Brown, K., Cooke, P., & Stitt, G. (2012). A performance and energy comparison of FPGAs, GPUs, and multicores for sliding-window applications. In *Proceedings of the ACM/SIGDA International Symposium on Field Programmable Gate Arrays (FPGA)* (pp. 47-56). <https://doi.org/10.1145/2145694.2145705>
7. Liu, Y., Yang, C., & Cong, J. (2017). Memory access scheduling for parallel sparse matrix-vector multiplication. In *Proceedings of the International Conference on Field Programmable Logic and Applications (FPL)* (pp. 1-7). <https://doi.org/10.23919/FPL.2017.8056815>
8. Zhang, C., Li, P., Sun, G., Guan, Y., Xiao, B., & Cong, J. (2015). Optimizing FPGA-based accelerator design for deep convolutional neural networks. In *Proceedings of the ACM/SIGDA International Symposium on Field-Programmable Gate Arrays (FPGA)* (pp. 161-170). <https://doi.org/10.1145/2684746.2689060>
9. Suda, N., Chandra, V., Dasika, G., Mohanty, P., Ma, Y., Vrudhula, S., ...& Chakradhar, S. (2016). Throughput-optimized OpenCL-based FPGA accelerator for large-scale convolutional neural networks. In *Proceedings of the ACM/SIGDA International Symposium on Field-Programmable Gate Arrays (FPGA)* (pp. 16-25). <https://doi.org/10.1145/2847263.2847276>
10. Zhao, J., Mao, Z., & Chen, D. (2020). AutoDSE: Enabling adaptive design space exploration for FPGA HLS using reinforcement learning. In *Proceedings of the International Conference on Computer-Aided Design (ICCAD)* (pp. 1-9). <https://doi.org/10.1145/3400302.3415731>
11. Neuendorffer, S., & Williams, J. (2015). Challenges in standardizing high-level synthesis flows. In *Proceedings of the IEEE International Conference on Reconfigurable Computing and FPGAs (ReConFig)* (pp. 1-6). <https://doi.org/10.1109/ReConFig.2015.7393323>
12. Chhabra, S., & Tessier, R. (2013). Hybrid simulation models for hardware/software co-design verification. *IEEE Transactions on Computers*, 62(6), 1220-1233. <https://doi.org/10.1109/TC.2012.113>

13. Vishnupriya, T. (2025). Wireless body area network (WBAN) antenna design with SAR analysis. *National Journal of RF Circuits and Wireless Systems*, 2(1), 37-43.
14. Jeon, S., Lee, H., Kim, H.-S., & Kim, Y. (2023). Universal Shift Register: QCA Based Novel Technique for Memory Storage Modules. *Journal of VLSI Circuits and Systems*, 5(2), 15-21. <https://doi.org/10.31838/jvcs/05.02.03>
15. Yeonjin, K., Hee-Seob, K., Hyunjae, L., & Sungho, J. (2023). Venting the potential of wirelessly reconfigurable antennas: Innovations and future directions. *National Journal of Antennas and Propagation*, 5(2), 1-6.
16. RANGISETTI, R., & ANNAPURNA, K. (2021). Routing attacks in VANETs. *International Journal of Communication and Computer Technologies*, 9(2), 1-5.
17. Kavitha, M. (2023). Beamforming techniques for optimizing massive MIMO and spatial multiplexing. *National Journal of RF Engineering and Wireless Communication*, 1(1), 30-38. <https://doi.org/10.31838/RFMW/01.01.04>