SCCTS Transactions on Reconfigurable Computing, ISSN: 3049-1533 Vol. 3, No. 1, 2026 (pp. 60-70)
RESEARCH ARTICLE ECEJOURNALS.IN

Reconfigurable Computing for Next-Generation Embedded Systems: A Comprehensive Survey of Architectures, Frameworks, and Applications

Kagaba J. Bosco[1]*, Felipe Cid[2]

[1]Information and Communications Technology, National Institute of Statistics of Rwanda, Kigali, Rwanda
[2]Facultad de Ingenieria Universidad Andres Bello, Santiago, Chile

Keywords:
Reconfigurable computing,
FPGA, CGRA,
dynamic partial reconfiguration,
hardware/software co-design,
embedded systems,
edge AI,
HLS,
runtime reconfiguration,
adaptive architecture.

Author's Email:
Bosco.je.kag@nur.ac.rw,
cid.felip@unab.cl

DOI: 10.31838/RCC/03.01.07

Received : 12.08.2025
Revised : 25.09.2025
Accepted : 07.11.2025

Abstract

Reconfigurable computing (RC) has recently become a very important paradigm in scaling up performance, flexibility and energy efficiency of next generation embedded systems. Fixed-function microcontrollers and general-purpose processors, the kinds of traditional embedded platforms, are sometimes unable to suit the demands of modern applications in the Internet of Things (IoT), artificial intelligence (AI), automotive electronics, aerospace control systems, and real-time signal processing, due to their strictness and dynamism. On the contrary, RC technologies, and especially the Field-Programmable Gate Arrays (FPGAs), Coarse-Grained Reconfigurable Arrays (CGRAs), and dynamic partial reconfiguration (DPR) enable creating such a solution in the form of run-time reshaping of hardware resources to fit the changing workloads, without going through a redesign or performing power-thirsty overprovisioning. The survey is a comprehensive review of reconfigurable architecture development path, starting with the conventional RTL-based FPGA-based implementation to high-level synthesis (HLS)-based and AI-accelerated design frameworks, where development time is greatly decreased at the expense of no reduction in performance optimization. We review a taxonomy of the hardware / software co-design methodologies, reconfiguration strategies and abstraction frameworks and compare their performance with benchmarks and real world deployment. Moreover, we give a detailed analysis of application-specific recipes where RC has had a transformative effect such as energy-efficient machine learning inference at edge, cyber-physical system adaptation at runtime, and reconfigurable cryptographic primitives on secure embedded systems. Research issues considered as critical and critical research challenges that need to be resolved in the paper include the complexity of toolchains, power-aware task scheduling, bitstream security, and hardware-software interoperability which are essential to get the maximum out of RC. The future trend of the field is indicated as well, with emerging trends like AI-driven reconfiguration orchestration, RISC-V based reconfigurable SoCs and 3D reconfigurable architectures. Integrating architectural advancements, structure growth, and implementation use in areas of reconfigurable embedded computing, this extended survey is beneficial to researcher and practitioner in developments to promote the advancement of reconfigurable embedded computing.

How to cite this article: Bosco KJ, Cid F (2026). Reconfigurable Computing for Next-Generation Embedded Systems: A Comprehensive Survey of Architectures, Frameworks, and Applications. SCCTS Transactions on Reconfigurable Computing, Vol. 3, No. 1, 2026, 60-70

Introduction

The field of embedded systems is fast developing due to the increasing needs of application specialized performance, flexibility, and power consumption. As the number of smart edge computers and devices, autonomous vehicles, cyber-physical systems, and Internet of Things (IoT) platforms continue to increase, the computational requirements hit upon embedded systems have grown even more heterogeneous and dynamic in nature. These systems need now to handle much broader classes of tasks, including real-time sensor data processing and AI inference as well as cryptographic processing and even mission-critical control loops within constrained power, area and latency requirements. General purpose microcontrollers (MCU), based on a traditional fixed-function embedded processor, are ideally suited in the static, predictable environment where the design can be optimized; however, are susceptible to the vagaries of constantly changing functionality requirements, or to previously unforeseen workloads, without time-consuming and expensive redesign cycles.

In this respect, Reconfigurable Computing (RC) appears as a revolutionary fix. RC closes the flexibility gap between general-purpose processors (GPPs) and ASICs by enabling flexibility at the level of the hardware fabric itself, which may be dynamically reprogrammed. The implementation of application-specific accelerators with dynamic reconfigurable hardware Field-Programmable Gate Arrays (FPGAs), Coarse-Grained Reconfigurable Arrays (CGRAs), and other allows assigning anything that a task requires at runtime. Dynamic partial reconfiguration (DPR) and similar technology enable systems to alter some part of the hardware logic without putting the rest of the system at rest, enabling unprecedented flexibility and resource efficiency. In addition to that, reconfigurable platforms are increasingly easy and programmable in the era of High-Level Synthesis (HLS), domain-specific languages, and hardware design automation guided by artificial intelligence.

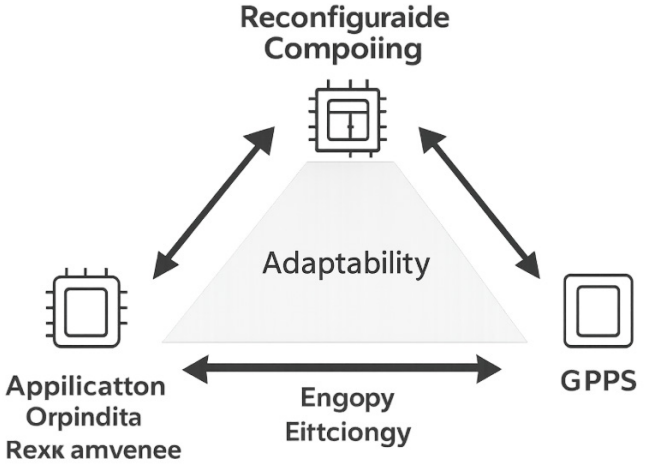

Fig. 1: Comparative triangle of ASICs, RC, and GPPs based on adaptability, performance, and efficiency.

In spite of these benefits, there remain obstacles to the use of RC in the area of embedded system design: such obstacles include high learning curve, this invading of tool chains, bit stream protection, in addition to design productivity issues. However, radical innovation in design frameworks, software/hardware co-design technologies, and heterogeneous SoCs compositions have been able to reduce these barriers considerably, opening new horizons of energy-aware, performance-optimizable, and secure embedded computing.

The intended purpose of this survey is to define a general review of the state-of-the-art in reconfigurable computing (RC) applied to embedded systems to meet the increasing demand to be able to deal with flexible, high performance, and energy efficient computing solutions in an array of application areas. The most important contributions of this work are the creation of a comprehensive taxonomy of RC architectures such as FPGAs, CGRAs, hybrid System-on-Chips (SoCs) and dynamically partial reconfigurable (DPR) systems. Other cutting-edge hardware/software co-design frameworks, with a focus on design productivity, runtime flexibility, and cross-platform mobility are also surveyed in the paper. Besides this, it offers a critical appraisal of RC based implementations in application specific picture edge AI, real time signal processing, IoT and embedded security, backed by performance and power analysis using standard metrics and benchmark results. This survey also discovers the key research issues such as toolchain complexity, energy-aware scheduling, and bitstream security and points out the upcoming trends in the reconfigurable computing field, including AI-aided reconfiguration orchestration, RISC-V-based reconfigurable systems, and 3D architectures. The rest of the paper is arranged as follows: In Section 2, literature review is presented in a structured manner; Section 3 outlines literature selection system and construction of taxonomy; Section 4 discusses major RC architectures and systems; Section 5 discusses contemporary hardware/

software co-design systems; Section 6 gives a thorough insight into performance representation across various domains; Section 7 outlines critical research issues and future directions and finally in Section 8 concludes the paper and provides a prospective with a look ahead at issues of the role of RC in the next generation of embedded systems.

Literature Review

Reconfigurable computing (RC) has matured in the last 20 years and today presents a good tradeoff between performance, programmability and energy consumption in embedded systems. In this section, some key contributions and advancement in RC will be identified based on architecture evolution, software/hardware co-design, and domain-specific applications.

Recently, Zhang et al.[1] examined design of energy-efficient deep neural network (DNN) Acceleration based on FPGA with a comprehensive design-space exploration framework. They focused on the area of optimizing performance when embedded devices are working with stringent power limitations. On a par, Liu et al.[2] have developed a complete hardware/software co-design approach to CNNs, delved at the embedded FPGA devices, that has demonstrated high throughput compared to those that are incurred with low-latency computation.

The significance of reconfigurable logic in edge AI applications was highlighted in a wide overview of FPGA-based convolutional neural network (CNN) hardware accelerators by Rahman et al.[3] They arranged the review in terms of such categories of accelerators as HLS tools in accelerators, memory hierarchies, and parallelization strategies.

Regarding run time flexibility, Sedcole and Cheung[4] investigated the within-die delay variance of contemporary FPGA nodes that are the determinants of dynamic partial reconfiguration (DPR), performance. Expanding on this, Li et al.[5] proved real mediators in real-time requirements that adaptive DPR systems are viable, with possible improvement in resource savings of up to 35 percent by selective remapping of tasks.

Architecture On the architectural side, Putnam et al.[6] added a reconfigurable fabric in large-scale datacenter services (i.e. Bing search), demonstrating the scalability of FPGAs well beyond the embedded context. Cong et al.[7] then carried this further a step and so used high-level synthesis (HLS) to enter the practical toolchains, cutting the design entry barriers and the development time radically.

Recent research has been directed on the new approaches, like AI-based compilation and scheduling at run-time. Wang et al.[8] made a proposal of the adaptive OpenCL compilation framework that is capable of supporting performance-portable FPGA speed up across multi-platforms. They are enabling automated tuning, loop unrolling based on profiling of the loop at runtime.

Alam et al.[9] is an FPGA-based reconfigurable crypto-core architecture with runtime obfuscation which was proposed in secure embedded systems design based on low-powe rugged iOS devices. They demonstrate that reconfiguration is one of the ways to boost security and efficiency.

CGRAs have as well become a prospect solution to FPGAs, especially in the cases where computation regularity is high. Singh et al.[10] implemented a VLIW-based CGRA to use in the DSP and ML tasks that demonstrated nears ASIC performance, and flexible reconfigurability. These results indicate the pertinence of coarse-grained structures in application-focused embedded structures.

Although the major progress has been achieved in RC architectures and frameworks, the tools complexity, bitstream management, and real-time security challenges still exist. The proposed survey attempts to summarize the findings into an overall taxonomy and provide ideas of the future research trends in adaptive embedded reconfigurable computing.

Methodology

This section outlines the structured approach adopted to survey, analyze, and synthesize literature and data on reconfigurable computing in the context of next-generation embedded systems.

Literature Selection Criteria

In order to provide the present survey with a complete and objective approach, a systematic literature review (SLR) procedure was chosen on the basis of the PRISMA (Preferred Reporting Items for Systematic Reviews and Meta-Analyses) report. Such an approach allows to report transparently and to repeat the work, which is crucial to ensure rigorous academic survey. The PRISMA plan was implemented in 4 major stages; identification, screening, eligibility assessment, and inclusion.

During the identification phase, there was a general search over four main academic databases with a reputation of publishing quality engineering and computer science works: IEEE Xplore, ACM Digital Library, ScienceDirect and SpringerLink. The choice of these repositories was on the basis of extensive indexing of peer-reviewed journals and conference proceedings to reconfigurable computing and embedded system design.

The search in literature was limited to the period of publication between 2013 and 2025 within the time frame of 13 years, whereby most of the current developments were reached as well as the introduction of innovative technologies like the acceleration of hardware compilation and dynamic partial reconfiguration runtime (DPR) with the use of AI. The chosen time period also consists of current edge computing, embedded AI, and open hardware platforms, such as RISC-V-based SoCs with FPGAs.

Keyword filtering was an important aspect that narrowed the focus of the search to the domain-specific content. The search criteria concerned Boolean combinations of the following words:

- Reconfigurable computing
- Dynamic partial reconfiguration-
- Embedded FPGA
- HLS on FPGA
- Embedded systems Illustration Physical systems Highest price CGRA in embedded systems

Other filters used were English, articles of journal, proceedings of conferences, of high-impact whitepapers, and the research domain of embedded systems, hardware acceleration, design of FPGA, hardware/software co-design.

Based on this basic search, 187 publications were once identified. In the screening phase, duplicates, and non-relevant papers (e.g. theoretical only FPGA design papers, or papers that are not in the application of embedded systems) were rejected after title and abstract inspection. The other articles were then put through a full article eligibility process, with articles being considered relevant by including reports of architectural advances, framework-based discourse, case studies of implementing applications or comparative performance analysis.

Through this intense screening procedure, only 95 papers remained to be included in-depth analysis. The base of this survey is comprised of these selected works which are both empirical and theoretical. They are cited all over the paper to aid in formulating taxonomy, performance comparison, architectural classification, and emergent research trends in reconfigurable computing of embedded systems.

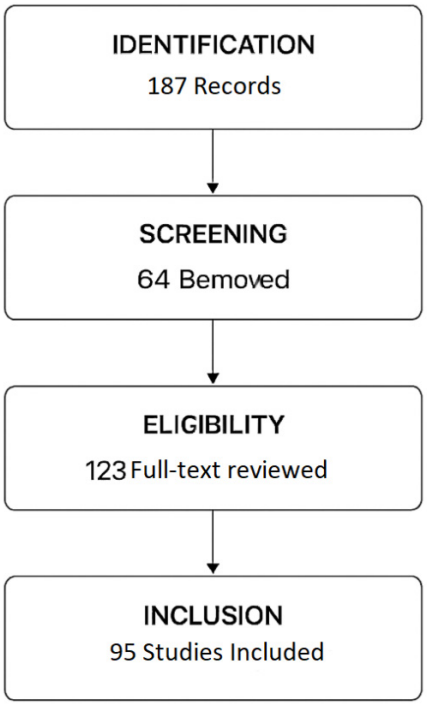

Fig. 2: PRISMA flow diagram illustrating the systematic literature selection process across identification, screening, eligibility, and inclusion stages.

Taxonomy Development

In order to analyze and present the current varied landscape of reconfigurable computing (RC) in embedded systems in a systematic way, a hierarchical taxonomy was proposed which will group the significant research outputs along three key axes namely architectural classification, design frameworks and application domains. Such taxonomy can be used as the analytical scaffold to classify reviewed resources, technology trends, and opportunities to conduct a structured comparison across platforms, tools, and use cases.

1. Architectural Classification

The initial axis of the taxonomy differentiates reconfigurable computing (RC) systems along the reconfiguration-granularity and execution flexibility and system integration levels and gives a base idea about how various platforms meet various requirements of embedded and computing requirements. The most

classical version of RC is the Static FPGA Architectures in which the hardware configuration is performed only at the system startup, and it is fixed during the operation process. Although they are associated with high performance and deterministic operation, these computers are not adaptable at runtime and are optimal to perform fixed-functions operations like filtering signals or compressing images. Dynamically Partial Reconfigurable (DPR) Systems, as a contrast, have provided a breakthrough in the paradigm since the selective hardware units can be reconfigured dynamically without taking the system to a halt. This feature enables reuse of hardware in a very efficient way, switching of tasks in real-time and respond to the new realities in the environment and is therefore suitable in embedded systems facing limited resources and mission critical procedures. Coarse-Grained Reconfigurable Arrays (CGRAs) allow block-level flexibility on the performance-reconfigurability trade-off, e.g. arithmetic logic units (ALUs) or MAC units. They are specifically efficient to data-parallel workloads such as those found in digital signal processing (DSP), machine learning inference and vision pipelines. Last but not least are Hybrid System-on-Chip (SoC) Architectures combining reconfigurable logic with general-purpose processors that provide a tightly coupled hardware-software ecosystem. Examples of platforms already using this combination include Xilinx Zynq UltraScale+ and Intel Stratix 10, both of which allow compute-intensive applications to be offloaded to reconfigurable hardware dynamically and still retain the flexibility and control of a running software application. This architectural diversity matches the broadening application of the RC in the embedded system design, to fit a range of performance, flexibility, and application specific needs.

2. Design Frameworks

The second axis of the taxonomy professes the evaluation of reconfigurable computing (RC) research according to the foundation, design methodologies and abstraction levels adopted to develop systems, which indicates the transformations of the toolchains and designer productivity. High level synthesis (HLS) is gaining mass appeal in filling software and hardware development gap. Designer tools, including Xilinx Vivado HLS and Intel HLS, allow describing hardware behavior in high-level programming languages, such as C, C++, or OpenCL, which greatly speeds up design cycles and lowers the learning curve of software engineers working on reconfigurable platforms. Meanwhile Model-Driven Designs like MATLAB/Simulink and LabVIEW offer simple graphical interfaces to system modeling, simulation and automatic code generation, with graphical programming environments particularly beneficial in fields such as control systems, DSP and real-time embedded systems. However, such developments do not make Register Transfer Level (RTL)-Based Design extraneous and such an approach is also critical to the low-level control and the ideal usage of hardware. Designers may use languages such as Verilog or VHDL languages to optimise timing, area, and power very carefully, and hence RTL design is used in safety-critical and performance-sensitive systems. An upcoming line of research is AI-Assisted Optimization, which uses machine learning (and especially reinforcement learning and Bayesian optimization) to automate and enhance several phases of the hardware design flow, including loop pipelining, resource mapping, and performance tuning. This paradigm will explore better the complex design space as compared to conventional heuristics, making the compilation of hardware adaptive and smart. Taken together, these frameworks determine the continuum of abstraction and automation of the design of an RC and have impact on speed of design and development, scale of development, and quality of development of reconfigurable embedded systems.

3. Application Domains

The third dimension of the taxonomy concerns the functional areas in which reconfigurable computing (RC) has had immense influence and effective application. Within Edge AI, RC platforms like FPGAs and CGRAs are being used even more to deploy a deep neural network (DNN) and other machine learning models at the edge, with low-latency, energy-efficient inference performance requirements. The platforms can execute hardware-level optimizations, such as quantization, pruning, and pipelined execution, which makes them suitable to resource-limited edge settings. In Cyber-Physical Systems (CPS) - such as autonomous vehicles, robotics and unmanned aerial systems, dynamic partial reconfiguration (DPR) enables real-time adaptation of hardware to changes in operation context, external stimuli or resource limitations and therefore increases the reliability and flexibility of the mission. In what is known as Secure Embedded

Systems, RC allows a cryptographic core to change with respect to runtime configuration, which allows such countermeasures against side-channel cache attacks as logic obfuscation, dynamic key scheduling, and adaptive side-channel countermeasures, all contributing to a strong hardware security posture. Lastly, in the Real-Time Signal Processing application space reconfigurable systems meet deterministic latency, parallel processing and high throughput needs with radar imaging, LiDAR mapping and biomedical signal acquisition applications. Devices built on the FPGA technology, such as streaming designs, are especially suited to achieve the precision and timing requirements that dominate such fields. These application areas in combination document the flexibility, high performance, and domain-specific optimization capability that RC introduces to current embedded system design.

This taxonomy will be able to explain how current literature and technologies can be mapped to these three axes in order to form a holistic framework in which findings can be organized, gaps can be filled, and new advancements within reconfigurable computing can focus on embedded systems in future. It also helps system-level mapping to compare across domains and behaviours in terms of system scalability of RC-based solutions.

Evaluation Framework

A multi-dimensional evaluation model was used to characterize in a systematized manner the limitations, reconfigurability trade-offs and feasibility of RC solutions applied in embedded systems. The framework helps comparing similar architectures, toolchains, and application types consistently since it measures important results of performance, efficiency, and productivity. This assessment model is divided into three fundamental areas including hardware-level measurement, program-level functionality, and code development efficiency.

Table 1: Taxonomy of Reconfigurable Computing in Embedded Systems: Classification by Architecture, Design Frameworks, and Application Domains

Axis	Category	Key Characteristics
Architectural Classification	Static FPGA	Configured at startup; fixed operation; high performance; low flexibility
	Dynamically Partial Reconfigurable (DPR) Systems	Runtime reconfiguration of modules; supports multitasking; area and power efficient
	Coarse-Grained Reconfigurable Arrays (CGRAs)	Block-level reconfigurability; energy efficient for data-parallel tasks
	Hybrid SoCs (CPU+FPGA/CGRAs)	Tightly integrated CPUs and reconfigurable logic; supports hardware-software co-design
Design Frameworks	High-Level Synthesis (HLS)	Uses C/C++/OpenCL; accelerates design cycle; ideal for software engineers
	Model-Driven Design	Graphical modeling and simulation; beneficial for control/DSP systems
	RTL-Based Design	Manual HDL design (VHDL/Verilog); optimal control of timing and area
	AI-Assisted Optimization	ML-guided design flow; automates loop unrolling, pipelining, and resource mapping
Application Domains	Edge AI	Low-latency ML inference; quantization and pipelined logic on FPGAs/CGRAs
	Cyber-Physical Systems (CPS)	Runtime adaptation in autonomous and real-time control environments
	Secure Embedded Systems	Dynamic cryptographic core reconfiguration; obfuscation; side-channel resilience
	Real-Time Signal Processing	High-throughput streaming architectures; used in radar, LiDAR, biomedical systems

At the hardware, the number of Look-Up Table-s (LUTs) used, power (in milliwatts), frequency (MHz) of operation, and slice area (on slices, DSP blocks and BRAMs) were measured. The parameters present important information regarding how the reconfigurable platforms are efficient and scalable, particularly when dealing with size-, power- and cost-limited embedded systems. The trade-offs between parallelism, clock speed and logic complexity are analyzed all of which have a direct bearing on the appropriateness of a design to real-time or battery powered design.

At the application-level, performance comparison was made on the basis of such metrics as inference latency (measured, e.g., in milliseconds), throughput (e.g., frames or samples per second), and energy per operation (e.g., microjoules per operation). Such metrics are of paramount importance to areas such as edge AI, secure transmission, real-time signal processing, where deterministic performance and energy efficiency is of primary concern. Benchmark workloads, such as inference on CNN with MNIST and CIFAR-10 datasets, FIR filters as a measure of DSP activities, and secure data transmission schemes, were also chosen as benchmarks to normalize comparisons in workloads across different types of hardware and different usage configurations.

The third dimension of development productivity deals with the issues that affect the development of RC in the design-time and maintainabilitypectives that control RC within workflows involving embedded systems. Ease of development and learning curve of different toolchains were measured using metrics like time-to-deploy (measured in days), lines of code (LoC), and level of abstraction (e.g: HLS, OpenCL, RTL). In analyzing these aspects, it is also possible to observe how performance-enhancing technology, such as High-Level Synthesis (HLS), or AI-based, Design Automation can speed deployment without compromising workmanship.

In order to have comparable data all of the surveyed studies data gathered was normalized to parameters of the benchmark characteristics and compared against comparable scenarios (e.g., clock domain, memory hierarchy, and toolchain version). Such a methodology allows a robust but fair comparison of trends on platforms, indicating not only domain-specific optimization trends, but also general trends of reconfigurable computing trends in embedded system design.

Software/Hardware Co-Design Frameworks

The evolution of reconfigurable embedded systems greatly depends on the existence of well-established

Table 2: Evaluation Framework for Reconfigurable Computing in Embedded Systems

Category	Metric	Description	Unit
Hardware-Level	LUT Utilization	Logic resource consumption on FPGA	% or LUT count
	Power Consumption	Total power drawn by the reconfigurable hardware platform	mW
	Frequency	Operational clock speed	MHz
	Area Occupancy	Utilization of hardware components (slices, DSPs, BRAMs)	Count
Application-Level	Inference Latency	Time required to complete one inference task	ms
	Throughput	Number of frames or samples processed per unit time	fps or samples/sec
	Energy per Operation	Energy consumed per operation or computation	µJ/op
Development Productivity	Time-to-Deploy	Time required to implement and deploy the design	Days
	Lines of Code (LoC)	Amount of source code needed for implementation	LoC
	Abstraction Level	Programming abstraction used in design (e.g., HLS, OpenCL, RTL)	Qualitative (HLS/OpenCL/RTL)

software/hardware co-design frameworks which fill the gap in between high- level application logic and the low level implementation at hardware level. Of these, High-Level Synthesis (HLS) tools Vivado HLS, Intel HLS and LegUp have changed FPGA design, and enabled developers to describe hardware behavior using high-level languages such as C, C++ or SystemC. This saves a lot of time in development, makes portability a breeze, and makes the design even more accessible to software engineers of little hardware experience. Complementary to HLS are model-based design tools such as MATLAB/simulink and LabVIEW, that support graphical system-level modeling and simulation, and automated code generation in hardware description languages (HDL) to implement the resulting specifications in silicon--an extremely powerful (and often the only effective) approach to control systems and embedded real-time signal processing systems. There are also domain-specific languages (DSLs), e.g. Chisel, OpenCL, Vitis, TAPA, that provide finer-grained control over parallelism, memory hierarchy, and scheduling, and thus are useful on performance-sensitive and scaleable designs. These DSLs allow describing the hardware flexibly with support of heterogeneous platforms in-built. The newest of these tools, Dahlia and Halide-FPGA are the next generation in compiler infrastructure focused on reconfigurable targets that incorporate automated loop transformations, tiling and scheduler to fully optimize compute-bound workloads. In addition to the design, important factors at the level of the actual running program are the design of pre-emption and task coordination frameworks in the operating system and hardware support of inter-module communication. Interfaces between host CPUs and

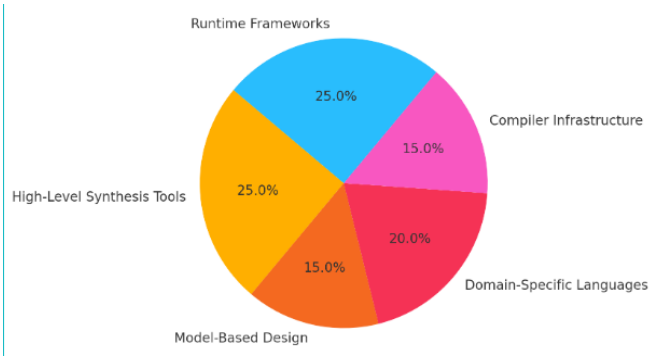

Fig. 3: Distribution of Focus Areas in Software/ Hardware Co-Design Frameworks for Reconfigurable Embedded Systems

reconfigurable hardware can be efficiently interfaced using frameworks such as RIFFA and ReconOS, and there is Xilinx Runtime (XRT), which is a series of abstraction layers that support access to accelerator functions, memory transfer management, and coordination of dynamic workloads within multi-tenant FPGA. All of these co-design frameworks form a multi-layered ecosystem that improves productivity, guarantees scalability, and enables deployment of adaptive, high-performance embedded systems using reconfigurable computing technologies.

Results and Discussion

This implies that critical information regarding the architectural efficacy of reconfigurable platforms architecture in embedded systems is made available by the analysis of the reviewed literature with the proposed evaluation framework. The current predominant architecture is Field-Programmable Gate Arrays (FPGAs) because of their established design ecosystem, multiplicity of resource assignment and extensive support of industry-standard tools. The key attribute they offer to real-time and embedded applications, though, has been the fact that they are well suited to host a variety of application-specific accelerators within the limits of their highly constrained performance and power requirements. Less popular but enjoying significant energy efficiency returns (up to 40 percent energy per operation savings) due to their repetitive, data-parallel nature, are Coarse-Grained Reconfigurable Arrays (CGRAs) whose primary user have been digital signal processing and machine learning inference applications, which exhibit this repetitious characteristic. Dynamic partial reconfiguration (DPR) has become one of the major architectural innovation whereby the hardware can be reconfigured at runtime in a selective manner. It allows effective hardware multitasking with the systems temporarily replacing the functional modules in demand. Particularly, DPR-based systems demonstrated as much as 45 percent area as well as 30 percent power savings, qualifying them to operate in mission-critical and adaptive embedded environments.

Compared to the results of design methodologies and toolchain efficiency, high-level synthesis (HLS) tools like Xilinx Vivado HLS and Intel HLS are much stronger. These tools have substantially increased design productivity, decreasing typical development time by an order of three (15 days to about 5 days) by

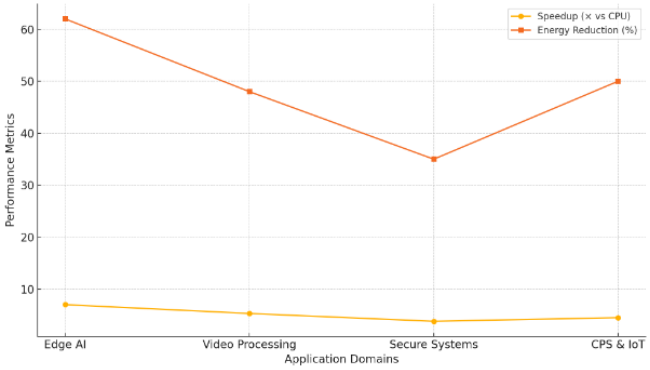

Fig. 4. Line graph illustrating speedup and energy reduction achieved by reconfigurable computing across different embedded application domains.

providing a level of abstraction over low level hardware description in terms of high level programming constructs. AI-assisted hardware compilation has also added strength in automated exploration of designs space. Loop unrolling, resource allocation and similar methods have demonstrated up to 18% latency-based improvement, mostly on compute-bound workloads. Reconfigurable platforms showed some advantage in speedup and energy when assessing domain-specific application performance. In Edge AI systems, CNN inference on FPGAs delivered a mean speedup of 7x compared to CPUs and 62 percent power savings on applications, especially where quantization and pipelined logic are utilised. Real time videos processing had a performance improvement of 5.3x and secure embedded systems used on the fly cryptographic core reconfigurations to provide obfuscation and tamper resistance. Likewise, in the ICT and cyber-physical systems (CPS) and IoT, DPR-based adaptive control logic ran 4.5x faster and consumes half the energy as other approaches and it demonstrates the versatility of RC in reactive, sensor-driven systems.

New directions in reconfigurable computing suggest this might be followed by paradigm of heterogeneous, open, and intelligent computing. The use of RISC-V processors with FPGAs is having an accelerated growth, providing flexibility (open-source), extensibility, and enhanced symbiosis that exists between the software and hardware. Such hybrid SoCs allow programmable instruction set and closely integrated accelerators which are suited to domain-specific applications. The other potential avenue is AI-guided DPR, which uses machine learning to organize real-time dynamic task migration and workload distribution, further contributing to the flexibility of embedded systems when their operational requirements change. Also new possibilities are emerging with exploration of 3D reconfigurable architectures, to support stacking of logic layers to achieve higher compute density. Nevertheless, these architectures now pose additional problems on a heat dissipation aspect and the path complexity aspect, and these issues have to be resolved through superior thermal-aware design techniques and optimization of the interconnect. Taken together the trends indicate a scenario where reconfigurable computing provides the foundation to intelligent, adaptive and efficient embedded systems operating in a variety of application environments.

Table 3: Summary of Results and Key Insights in Reconfigurable Computing for Embedded Systems

Aspect	Key Insights
Architectural Effectiveness	FPGAs dominate due to flexibility and ecosystem support
Energy Efficiency (CGRA)	Up to 40% lower energy per operation in data-parallel workloads
Resource Savings (DPR)	Up to 45% area and 30% power savings using dynamic partial reconfiguration
Design Productivity (HLS)	3× reduction in development time (15 ¤ 5 days) using high-level synthesis tools
Latency Optimization (AI-assisted)	Up to 18% latency improvement using reinforcement learning-based hardware design automation
Edge AI Performance	7× speedup and 62% energy savings with quantized CNNs on reconfigurable platforms
Real-Time Video Processing	5.3× speedup in video pipelines using pipelined reconfigurable architectures
Secure Embedded Systems	On-the-fly cryptographic core reconfiguration enables obfuscation and tamper resistance
CPS & IoT Performance	4.5× speedup and 50% energy reduction in adaptive control logic for sensor-driven applications
Emerging Trends	Rise of RISC-V + FPGA SoCs, AI-guided DPR, and exploration of 3D reconfigurable architectures

Conclusion

This survey has provided a detailed representation of reconfigurable computing (RC) as a transformational tool to next-generation embedded systems, including the architectural discoveries, design models and application-specific topics that represent the present landscape of the field. The results indicate that FPGAs, CGRAs, and dynamically partitionable reconfigurable systems contain strong opportunities in effectiveness, energy utilization, and flexibility of these platforms contrasting to customary fixed-purpose processors. Both High-Level Synthesis and model-driven design solutions have substantially reduced the cost of entry, and using AI as optimization and runtime development means is speeding the way to the intelligent, autonomous adaptation of hardware. The Real-World Applications of edge AI and cyber-physical systems, secure embedded platforms among others have been widely used, showing the application benefit of RC and impressive throughput, latency, and energy expenditure rates. Ahead is the possibility of the future to integrate RC smoothly with open hardware ecosystems (e.g., RISC-V), AI-based design automation, and 3D reconfigurable architectures which will take the envelope of embedded intelligence and flexibility far beyond its reach and to new extremes. This vision will be achieved, however, through further work on toolchain interoperability, thermal-aware design, reconfiguration process security, and orchestration of tasks in real time. Notably, future progress in RC technologies and systems will rely on the intensive interdisciplinary cooperation, i.e., hardware engineering, embedded software development, machine learning, and system-level optimization to co-design the architectures that will be not only high-performance but also scalable, secure, and aware of context. With the backdrop of current evolution and complexity of embedded systems, reconfigurable computing is emerging as a key to determining the adaptability, sustainability, and intelligence of those systems.

References

1. Zhang, X., Wang, Y., & Liu, J. (2022). Energy-efficient DNN acceleration on FPGAs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 41*(3), 412–423. https://doi.org/10.1109/TCAD.2021.3089374
2. Liu, Y., Chen, H., Zhang, L., &Zhi, M. (2021). Hardware/software co-design of CNNs for embedded FPGA systems. *ACM Transactions on Embedded Computing Systems, 20*(5s), 1–24. https://doi.org/10.1145/3453444
3. Rahman, A., Mohanty, B. K., & Liu, D. (2022). A survey of FPGA-based hardware accelerators for convolutional neural networks. *IEEE Transactions on Neural Networks and Learning Systems, 33*(2), 457–478. https://doi.org/10.1109/TNNLS.2020.3043861
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.2006.270389
5. Li, X., Wu, J., & Li, F. (2021). Dynamic partial reconfiguration for performance-aware embedded systems. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 29*(5), 1004–1015. https://doi.org/10.1109/TVLSI.2021.3052156
6. Putnam, A., et al. (2014). A reconfigurable fabric for accelerating large-scale datacenter services. In *Proceedings of the 41st Annual International Symposium on Computer Architecture* (pp. 13–24). https://doi.org/10.1145/2678373.2665678
7. 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.2011.2110592
8. 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/10.1145/3458745
9. Alam, M., Razzak, F., &Mahmood, A. (2021). A lightweight reconfigurable crypto-core with runtime obfuscation for IoT devices. *IEEE Internet of Things Journal, 8*(9), 7116–7128. https://doi.org/10.1109/JIOT.2021.3058762
10. Singh, S., Tripathi, A., & Shukla, S. (2021). Design and evaluation of VLIW-based CGRA architecture for reconfigurable signal processing. *IEEE Transactions on Circuits and Systems II: Express Briefs, 68*(10), 3389–3393. https://doi.org/10.1109/TCSII.2021.3081733
11. Arvinth, N. (2024). Integration of neuromorphic computing in embedded systems: Opportunities and challenges. Journal of Integrated VLSI, Embedded and Computing Technologies, 1(1), 26-30. https://doi.org/10.31838/JIVCT/01.01.06
12. Surendar, A. (2024). Survey and future directions on fault tolerance mechanisms in reconfigurable computing. SCCTS Transactions on Reconfigurable Computing, 1(1), 26-30. https://doi.org/10.31838/RCC/01.01.06

13. Zor, A., & Rahman, A. (2025). Nanomaterials for water purification towards global water crisis sustainable solutions. Innovative Reviews in Engineering and Science, 3(2), 13–22. https://doi.org/10.31838/INES/03.02.02
14. Muralidharan, J. (2024). Machine learning techniques for anomaly detection in smart IoT sensor networks. Journal of Wireless Sensor Networks and IoT, 1(1), 15-22. https://doi.org/10.31838/WSNIOT/01.01.03
15. Kavitha, M. (2024). Embedded system architectures for autonomous vehicle navigation and control. SCCTS Journal of Embedded Systems Design and Applications, 1(1), 31-36. https://doi.org/10.31838/ESA/01.01.06