

## Review Article

# FPGA-Based Embedded Systems: Design, Applications, and Performance Analysis

Amit Barsana<sup>1</sup>, Shubham Sharma<sup>2</sup>

<sup>1,2</sup>Student, National Institute of Technology Warangal, India.

## I N F O

**Corresponding Author:**

Amit Barsana, National Institute of Technology Warangal, India.

**E-mail Id:**

barsana10amit@gmail.in

**Orcid Id:**

<http://orcid.org/0009-0004-7793-1898>

**How to cite this article:**

Barasana A, Sharma S. FPGA-Based Embedded Systems: Design, Applications, and Performance Analysis. *J Adv Res Embed Sys* 2024; 11(2): 7-13.

Date of Submission: 2024-06-12

Date of Acceptance: 2024-07-04

## A B S T R A C T

Field-Programmable Gate Arrays (FPGAs) have emerged as a cornerstone in the design of embedded systems, offering unparalleled flexibility, high performance, and energy efficiency. These programmable hardware platforms enable designers to implement complex functionalities while adapting to evolving requirements, making them ideal for a wide range of applications in industries such as consumer electronics, automotive, aerospace, healthcare, and industrial automation. This article provides a comprehensive review of FPGA-based embedded systems, examining design methodologies that integrate hardware-software co-design, advanced verification tools, and cutting-edge techniques like High-Level Synthesis (HLS) and Dynamic Partial Reconfiguration (DPR). Diverse applications are explored, showcasing the role of FPGAs in achieving real-time processing, low-latency operation, and power efficiency in critical domains. Detailed performance considerations highlight their advantages in parallel processing, energy optimization, and customization, while addressing inherent challenges such as design complexity, initial costs, and thermal management. Key challenges and future trends, including the integration of FPGAs with artificial intelligence (AI) frameworks, heterogeneous computing architectures, and edge computing solutions, are also discussed to illuminate the evolving landscape of FPGA integration in embedded systems. By capturing the state-of-the-art and emerging possibilities, this review underscores the transformative potential of FPGAs in shaping the future of embedded technologies.

**Keywords:** Field-Programmable Gate Arrays, Dynamic Partial Reconfiguration (DPR), Heterogeneous Computing Architectures

## Introduction

Embedded systems are the backbone of modern technology, powering an extensive array of applications that span consumer electronics, automotive systems, medical devices, aerospace systems, and industrial automation. These systems are characterized by their ability to perform specific tasks with precision, reliability, and efficiency, often operating in resource-constrained environments.

In this context, Field-Programmable Gate Arrays (FPGAs) have emerged as pivotal components, offering unique advantages that make them indispensable in the design and deployment of advanced embedded systems.

FPGAs are reconfigurable hardware platforms known for their unparalleled flexibility, enabling designers to customize hardware architectures to meet application-specific requirements. Unlike traditional microcontrollers

or Application-Specific Integrated Circuits (ASICs), which are either fixed-function or require significant time and cost for development, FPGAs can be programmed and reprogrammed even after deployment. This reconfigurability, combined with their inherent ability to process tasks in parallel, makes FPGAs well-suited for applications demanding high throughput, low latency, and energy efficiency. Furthermore, their capability to implement complex algorithms in hardware allows for significant performance gains, especially in domains such as signal processing, real-time control, and machine learning.<sup>1,2</sup>

In addition to these technical advantages, FPGAs provide a pathway for rapid prototyping, making them invaluable for accelerating the design and testing of embedded systems. They enable iterative development cycles where hardware can be modified and optimized without the delays associated with ASIC fabrication. This feature not only reduces development costs but also enhances the adaptability of products to changing market demands or evolving standards.<sup>3</sup>

This article delves into the state-of-the-art in FPGA-based embedded systems, offering a comprehensive overview of the design methodologies that underpin their development. It examines diverse applications across key industries, illustrating the transformative role of FPGAs in addressing complex challenges. Moreover, it provides an in-depth performance analysis, highlighting the strengths and limitations of FPGA-based systems in comparison to traditional solutions. Finally, the article explores emerging trends and future directions, shedding light on how advancements in FPGA technology are poised to shape the next generation of embedded systems. Through this analysis, the article aims to provide a thorough understanding of the potential and evolving role of FPGAs in driving innovation across industries.

## Design Methodologies for FPGA-Based Systems

The design of FPGA-based embedded systems is a structured process that combines hardware and software development to create optimized, high-performance solutions. This process involves several critical steps, each contributing to the reliability, efficiency, and adaptability of the final system.

### Specification and Partitioning

The process begins with a detailed specification of the system's functional and non-functional requirements. Functional requirements define what the system is expected to do, while non-functional requirements address constraints like power consumption, performance, and cost. Partitioning tasks between hardware and software components is a key aspect of this phase, as it determines which functions are best executed in hardware (for performance

and parallelism) and which are better suited for software (for flexibility and ease of updates). Effective partitioning ensures an optimal balance between performance and resource utilization.

### Hardware Design

In the hardware design phase, developers use Hardware Description Languages (HDLs) such as VHDL or Verilog to define the logic circuits that will be implemented on the FPGA. This phase focuses on creating a design that meets performance and power consumption targets while utilizing FPGA resources efficiently. Modern tools and methodologies, including High-Level Synthesis (HLS), allow developers to design hardware using high-level programming languages like C++ or OpenCL, significantly reducing the complexity and time required for hardware development.<sup>4</sup>

### Software Development

FPGA-based systems often require a software layer to function effectively. This layer typically includes device drivers, middleware, and application-level code. Software development is usually carried out in high-level programming languages, such as C or Python, to simplify the coding process. It often involves integrating third-party libraries, configuring hardware-software interfaces, and optimizing software for real-time performance.

### Verification and Validation

Ensuring the correctness and robustness of the system is critical, and this is achieved through rigorous verification and validation processes. Tools like ModelSim, Vivado, and Quartus Prime are commonly used to simulate hardware behavior and verify that the design meets specified requirements. This step includes functional testing, timing analysis, and power consumption estimation to ensure that the design is robust and reliable before deployment. Validation extends to verifying that the entire system, including hardware and software, operates as expected under real-world conditions.

### Implementation and Testing

Once the design is verified, it is synthesized and programmed onto the FPGA. Implementation involves mapping the HDL or HLS design onto the FPGA fabric, optimizing the layout to maximize performance and minimize resource usage. Following implementation, extensive testing is conducted in simulated and real-world environments to identify and resolve any issues. Testing ensures that the system performs correctly under various conditions, including stress testing and edge-case scenarios, to validate its robustness and reliability.

### Emerging Trends in FPGA Design

The field of FPGA design is evolving rapidly, driven by advancements in tools and techniques that simplify develop-

ment, enhance functionality, and improve overall system performance. Among these, High-Level Synthesis (HLS) and Dynamic Partial Reconfiguration (DPR) stand out as transformative technologies that are reshaping how FPGAs are utilized in embedded systems and other applications.<sup>5</sup>

### High-Level Synthesis (HLS)

HLS has emerged as a groundbreaking approach that abstracts the complexity of traditional FPGA development. Traditionally, developers relied on Hardware Description Languages (HDLs) like VHDL or Verilog, which required deep knowledge of low-level hardware design. HLS eliminates this barrier by enabling developers to design hardware using high-level programming languages such as C, C++, or OpenCL. These high-level designs are automatically translated into HDL, significantly reducing the time and expertise needed for development. HLS not only accelerates the design process but also facilitates rapid iteration and prototyping, allowing designers to explore multiple design configurations efficiently. Additionally, HLS tools often include features like automated optimization for latency, throughput, and resource utilization, which can result in designs that are as efficient as those created manually using HDLs. The adoption of HLS is particularly beneficial in domains like AI, signal processing, and scientific computing, where complexity and performance demands are high.

### Dynamic Partial Reconfiguration (DPR)

DPR is another transformative trend that extends the flexibility of FPGAs. With DPR, it is possible to reconfigure specific sections of an FPGA at runtime without disrupting the operation of other sections. This capability enables multi-tasking, where different tasks can be loaded and executed dynamically based on current requirements. For example, in an industrial automation system, one part of the FPGA could be dedicated to monitoring sensors, while another section could be reconfigured on the fly to perform diagnostics or adapt to changing production requirements. DPR also enhances resource utilization by allowing hardware resources to be shared among multiple functions over time, reducing the need for oversized or underutilized FPGA resources. This feature is particularly advantageous in applications with dynamic workloads, such as adaptive filtering in signal processing, real-time image processing, and reconfigurable AI accelerators.

### Future Directions in HLS and DPR

The integration of HLS with DPR is a promising frontier, enabling developers to dynamically reconfigure FPGA hardware using high-level languages, further simplifying the design process and increasing system flexibility. Additionally, advancements in toolchains and frameworks are expected to improve the interoperability of HLS and DPR with AI development environments and heterogeneous

computing platforms. For instance, HLS-generated designs may increasingly be optimized for interaction with CPUs and GPUs, while DPR can enable seamless transitions between AI models or data processing tasks in real-time.

### Applications of FPGA-Based Embedded Systems

FPGA-based embedded systems have become integral to a wide range of industries, leveraging their unique combination of reconfigurability, high performance, and parallel processing capabilities. Their adaptability and efficiency make them invaluable in applications requiring real-time processing, precision, and robustness. Below is an extended exploration of their application across key sectors, highlighting specific use cases and advantages.

#### Consumer Electronics

FPGAs play a critical role in consumer electronics, particularly in applications requiring digital signal processing (DSP). They are extensively used in audio and video devices, such as smart TVs, home theater systems, and high-fidelity audio equipment, where they enable real-time processing with minimal latency. For example, FPGAs can process 4K or 8K video streams, apply filters, or enhance image quality in real time. This capability is essential for delivering superior user experiences in devices where responsiveness and processing speed are paramount. Moreover, their programmability allows manufacturers to update features or improve device performance post-deployment, enhancing product longevity.

#### Automotive Systems

The automotive sector heavily relies on FPGA-based systems for Advanced Driver-Assistance Systems (ADAS) and autonomous driving technologies. These systems perform high-speed data processing from sensors like LiDAR, cameras, and radar, enabling safety-critical operations such as collision avoidance, lane-keeping, and adaptive cruise control. For instance, FPGAs facilitate sensor fusion, integrating data from multiple sources to create a cohesive understanding of the vehicle's surroundings in real-time. Their ability to handle massive data streams with low latency ensures reliable performance in safety-critical scenarios. Additionally, the reconfigurability of FPGAs allows for future-proofing, as automotive systems can be updated to accommodate evolving standards or functionalities.

#### Medical Devices

In the healthcare industry, FPGAs are at the forefront of innovation in medical imaging and diagnostics. Their ability to execute real-time image processing makes them ideal for applications in MRI, CT scans, and ultrasound equipment. For example, FPGAs can process high-resolution images

rapidly, enabling clearer visualization of internal structures and facilitating precise diagnosis. Their precision and reliability are crucial in critical diagnostic procedures, where errors could have significant consequences. Moreover, FPGAs' low power consumption makes them suitable for portable medical devices, enhancing accessibility and usability in various clinical settings.

### Industrial Automation

FPGAs are widely used in industrial automation systems for tasks such as predictive maintenance, machine vision, and process control. Their robustness makes them well-suited for harsh environments, such as factories with high temperatures, vibrations, or electromagnetic interference. For instance, in predictive maintenance applications, FPGAs can process sensor data in real time to identify potential equipment failures before they occur, minimizing downtime and reducing maintenance costs. Additionally, in robotic systems, FPGAs enable high-speed motor control and real-time processing of vision data, enhancing precision and efficiency in automated processes.

### Aerospace and Defense

In aerospace and defense, FPGAs are crucial for applications such as radar systems, electronic warfare, and signal intelligence. They provide the computational power needed for real-time signal processing, enabling accurate detection, tracking, and analysis of targets. For example, FPGAs in radar systems can perform fast Fourier transforms (FFTs) to process radar signals quickly and accurately. Their reconfigurability ensures flexibility in adapting to changing mission requirements or deploying new algorithms during the lifecycle of a system. Additionally, their ability to operate reliably in extreme conditions, such as high altitudes or space, makes them indispensable in these critical applications.

### Performance Analysis

The performance of FPGA-based embedded systems is a significant factor driving their adoption in a wide range of applications. By leveraging the unique architectural features of FPGAs, designers can achieve high levels of processing efficiency, energy savings, and adaptability, making them ideal for tasks requiring high throughput and low latency. Below is a detailed analysis of the key performance aspects and metrics associated with FPGA-based systems.<sup>6,7</sup>

#### Processing Speed

FPGAs excel in scenarios requiring high processing speeds due to their unparalleled parallelism. Unlike sequential processing in CPUs, FPGAs can execute multiple operations simultaneously, making them highly effective for high-throughput tasks. For example, in video processing, FPGAs can handle multiple streams or execute complex image enhancement algorithms in real time without bottlenecks.

The fine-grained parallelism of FPGAs also allows for significant acceleration of compute-intensive algorithms, such as matrix operations in machine learning, making them competitive with GPUs for specific workloads.

### Power Efficiency

A critical advantage of FPGAs lies in their power efficiency. For application-specific workloads, FPGAs often consume significantly less power than general-purpose processors, including CPUs and GPUs, while delivering equivalent or superior performance. This energy efficiency is especially important in applications like IoT devices, edge computing, and portable medical equipment, where power consumption is a critical constraint. By tailoring hardware circuits to specific tasks, FPGAs minimize energy wastage, ensuring optimal performance per watt consumed.

### Flexibility

The reconfigurable nature of FPGAs adds a level of flexibility that is unmatched by fixed-function hardware like ASICs. This flexibility enables rapid prototyping, allowing designers to test and iterate on hardware designs quickly without the long lead times associated with ASIC fabrication. Additionally, FPGAs can be reprogrammed to adapt to new standards, protocols, or functionalities, extending the product lifecycle and reducing the need for hardware replacement. This is particularly advantageous in dynamic industries such as telecommunications, where standards like 5G are continuously evolving.

### Cost Considerations

While FPGA-based designs involve high initial costs due to the need for development tools, expertise, and potentially expensive FPGA chips, they are often more cost-effective than ASICs for low- to mid-volume production. The high upfront costs of ASIC development make them viable only for large-scale production, whereas FPGAs provide a more economical solution for applications with variable or limited production volumes. Additionally, the ability to update FPGA-based systems post-deployment can offset initial costs by eliminating the need for hardware redesign in response to changing requirements.

### Performance Metrics

To evaluate the performance of FPGA-based systems, several key metrics are typically used:

- **Throughput:** Measured as the amount of data processed per second, throughput is a critical metric for applications like signal processing, where high data volumes must be handled efficiently.
- **Latency:** The time delay between input and output is an essential consideration in real-time applications, such as automotive ADAS or live video streaming, where even slight delays can have significant consequences.



- **Energy Efficiency:** This metric evaluates performance relative to power consumption and is typically measured as performance per watt. It is especially critical for battery-operated and energy-constrained systems.

## Challenges and Limitations

While FPGA-based embedded systems offer numerous advantages, they also come with several challenges and limitations that can impact their adoption and implementation. These issues stem from the inherent complexity of FPGA technology, cost considerations, and the reliance on advanced tools and methodologies. A detailed exploration of these challenges is outlined below:

### Design Complexity

One of the primary challenges in FPGA-based system development is the steep learning curve associated with designing and debugging using Hardware Description Languages (HDLs) such as VHDL or Verilog. These languages require a deep understanding of digital design principles, timing constraints, and resource optimization, which can be daunting for developers, especially those with software-focused backgrounds. Moreover, debugging FPGA designs is inherently more complex than debugging software, as it often involves analyzing low-level hardware behaviors and timing issues that are difficult to detect and resolve. While tools like High-Level Synthesis (HLS) have simplified the process by enabling hardware design through high-level programming languages, the underlying complexity of FPGA design remains a significant barrier for many developers.<sup>8</sup>

### Higher Initial Costs

FPGAs typically require a higher initial investment compared to traditional microcontrollers or other general-purpose processors. This cost includes the price of the FPGA hardware itself, which can be significant for high-performance or large-scale devices, as well as the expenses associated with development tools and licensing. Additionally, the need for specialized expertise to design, optimize, and debug FPGA-based systems further adds to the overall cost. While these upfront investments may be offset by the flexibility and performance benefits of FPGAs, they can be a deterrent for low-budget or small-scale projects.

### Thermal Management

FPGAs, particularly those used in high-performance applications, can generate substantial amounts of heat due to their dense logic elements and high-speed operations. Effective thermal management is critical to ensure the reliable operation and longevity of the device. However, designing efficient cooling solutions, such as heatsinks, fans, or advanced liquid cooling systems, can increase the system's complexity and cost. In compact or embedded

applications where space and power are limited, managing heat dissipation without compromising performance becomes an even greater challenge.

### Toolchain Dependence

The quality and efficiency of FPGA-based designs are heavily influenced by the capabilities of the development tools provided by FPGA vendors. While tools like Vivado, Quartus Prime, and Libero SoC offer extensive features for design, simulation, synthesis, and debugging, they can also be a limiting factor. Designers often face challenges such as tool bugs, steep learning curves, and limited cross-platform compatibility. Furthermore, the proprietary nature of many FPGA toolchains can lead to vendor lock-in, reducing flexibility in choosing hardware or migrating designs to different FPGA platforms.

### Resource Utilization and Scalability

While FPGAs provide flexibility through reconfigurability, optimizing resource utilization within the constraints of the FPGA fabric can be challenging, especially for complex designs. The finite number of logic blocks, DSP slices, and memory resources can limit the scalability of applications, requiring designers to carefully balance performance and resource allocation. Overcoming these constraints often demands significant design expertise and iterative optimization, which can be time-intensive.

### Power Consumption in High-Performance Designs

Although FPGAs are generally more energy-efficient than CPUs for specific workloads, high-performance designs or those operating at high clock frequencies can result in increased power consumption. This is particularly true when implementing resource-intensive applications like AI inference or high-resolution video processing. Managing power consumption while maintaining performance and avoiding thermal issues requires careful design and often necessitates advanced power management techniques.

### Market and Ecosystem Maturity

The ecosystem surrounding FPGA development, including third-party libraries, IP cores, and community support, is less mature compared to more established platforms like CPUs and GPUs. This relative immaturity can lead to longer development times and fewer resources for troubleshooting and innovation. Furthermore, the rapidly evolving nature of FPGA technology requires continuous learning and adaptation, which can strain development teams.

Addressing these challenges requires advancements in design tools, training programs to upskill developers, and innovations in hardware architectures to simplify development and enhance usability. Despite these limitations, the unique benefits of FPGAs, including their

reconfigurability, parallelism, and adaptability, ensure their continued relevance in industries where high performance and customization are essential.

### Future Directions

The field of FPGA-based embedded systems is evolving rapidly, driven by advancements in technology and the increasing demand for high-performance, energy-efficient, and flexible solutions. As industries continue to explore new ways to leverage FPGAs, several promising directions are shaping the future of this technology.

### Integration with AI and ML

FPGAs are playing an increasingly pivotal role in accelerating Artificial Intelligence (AI) and Machine Learning (ML) workloads. Their ability to perform massive parallel computations and process data in real time makes them ideal for AI inference tasks. Integration with popular AI frameworks like TensorFlow, PyTorch, and ONNX is streamlining the deployment of AI models on FPGAs, making them accessible to a broader developer community. For instance, FPGAs are being used to accelerate convolutional neural networks (CNNs) and recurrent neural networks (RNNs), enabling applications such as image recognition, natural language processing, and autonomous systems. Looking ahead, advancements in FPGA architectures, such as AI-specific hard IP blocks, will further enhance their efficiency in executing AI workloads.

### Heterogeneous Computing

Heterogeneous computing, which combines different types of processors to optimize system performance, is an emerging trend in FPGA deployment. In heterogeneous systems, FPGAs work alongside CPUs and GPUs to balance workloads based on the strengths of each processor type. For example, GPUs can handle dense matrix computations, while FPGAs manage low-latency tasks such as data preprocessing or control logic. This synergy is particularly valuable in applications like high-performance computing (HPC), data analytics, and real-time streaming, where diverse computational demands must be met simultaneously. Future developments may focus on creating seamless interfaces and communication protocols between FPGAs and other processors, enhancing interoperability and system efficiency.

### IoT and Edge Computing

As the Internet of Things (IoT) and edge computing continue to grow, FPGAs are emerging as key enablers for these technologies. Edge devices require a balance of power efficiency and performance to process data locally, reducing latency and minimizing reliance on cloud computing. FPGAs meet these demands by providing real-time processing capabilities, low power consumption, and adaptability to

evolving application needs. For example, FPGAs are being used in smart cameras for real-time video analytics, in industrial IoT for predictive maintenance, and in healthcare for portable diagnostic devices. Future directions include the development of ultra-low-power FPGA architectures tailored for edge applications and tighter integration with IoT ecosystems.

### Smaller Form Factors

Advances in semiconductor process technology are enabling the design of smaller and more compact FPGAs without compromising their performance or capabilities. These miniaturized FPGAs are particularly advantageous for applications with stringent size constraints, such as wearable devices, portable medical equipment, and aerospace systems.<sup>9-11</sup> Smaller form factors also open up new opportunities for integrating FPGAs into consumer devices, enhancing functionalities like voice recognition, gesture control, and augmented reality. As process nodes shrink further, innovations such as 3D stacking and advanced packaging technologies will contribute to even more compact and efficient FPGA solutions.

### Enhanced Design Automation and Toolchains

The future of FPGA-based systems will be significantly influenced by advancements in design automation tools. Tools that simplify hardware design, such as High-Level Synthesis (HLS) and AI-driven optimization frameworks, are making FPGAs more accessible to developers with limited hardware expertise. Additionally, open-source FPGA toolchains are gaining traction, reducing dependence on proprietary ecosystems and fostering community-driven innovation. The continued evolution of these tools will empower developers to create complex systems more quickly and efficiently, driving wider adoption across industries.

### Security and Cryptography Applications

With growing concerns over data security, FPGAs are becoming increasingly valuable for implementing secure cryptographic systems. Their ability to perform high-speed encryption and decryption operations while being resistant to tampering makes them ideal for secure communications, blockchain applications, and cybersecurity solutions. Future research may focus on developing FPGA architectures specifically optimized for post-quantum cryptography, ensuring data security in the era of quantum computing.

### Sustainable and Green Computing

As industries prioritize sustainability, FPGAs are positioned as an eco-friendly alternative for energy-efficient computing. Their ability to execute specialized tasks with minimal power consumption aligns with global efforts to reduce carbon footprints. Future advancements may include the design

of FPGAs optimized for renewable energy applications, such as solar energy management and smart grid systems, furthering their role in creating sustainable technologies.<sup>12</sup>

## Conclusion

FPGA-based embedded systems are revolutionizing the landscape of modern technology, offering unparalleled performance, flexibility, and energy efficiency across a broad spectrum of applications. From consumer electronics to critical industries like automotive, healthcare, and aerospace, FPGAs have demonstrated their ability to meet the demands of complex, high-speed, and mission-critical tasks. Their reconfigurable nature, combined with advancements in parallel processing capabilities, allows for the development of customized solutions tailored to specific requirements, a key advantage over traditional processing technologies such as CPUs and ASICs.

Despite these benefits, challenges such as design complexity, higher initial costs, and thermal management must be addressed to fully unlock the potential of FPGA-based systems. However, ongoing advancements in design methodologies, including the adoption of High-Level Synthesis (HLS) and AI-driven optimization tools, are significantly lowering the barriers to entry. The introduction of innovative development toolchains and open-source ecosystems further simplifies the design process and fosters greater accessibility.

Emerging trends such as the integration of FPGAs with AI and ML frameworks, their role in heterogeneous computing systems, and their expanding use in IoT and edge computing applications underscore their growing relevance in a rapidly evolving technological landscape. Additionally, the development of smaller form factors and power-optimized architectures ensures their adaptability to new and constrained environments, making them indispensable for next-generation technologies.

Looking ahead, the role of FPGAs in shaping the future of embedded systems will become even more pronounced. As industries continue to demand higher levels of performance, security, and sustainability, FPGAs are uniquely positioned to deliver solutions that align with these objectives. Their versatility and evolving capabilities promise to drive innovation in areas such as green computing, post-quantum cryptography, and advanced AI applications.

## References

1. Kuon I, Rose J. Measuring the gap between FPGAs and ASICs. In Proceedings of the 2006 ACM/SIGDA 14th international symposium on Field programmable gate arrays 2006 Feb 22 (pp. 21-30).
2. Hauck S, DeHon A. Reconfigurable computing: the theory and practice of FPGA-based computation. Elsevier; 2010 Jul 26.
3. Amara A, Amiel F, Ea T. FPGA vs. ASIC for low power applications. Microelectronics journal. 2006 Aug 1;37(8):669-77.
4. Schlessman J, Chen CY, Wolf W, Ozer B, Fujino K, Itoh K. Hardware/software co-design of an FPGA-based embedded tracking system. In 2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'06) 2006 Jun 17 (pp. 123-123). IEEE.
5. Martin G, Smith G. High-level synthesis: Past, present, and future. IEEE Design & Test of Computers. 2009 Aug 21;26(4):18-25.
6. Youssef E, Elsemariy HA, El-Moursy MA, Khattab A, Mostafa H. Energy adaptive convolution neural network using dynamic partial reconfiguration. In 2020 IEEE 63rd International Midwest Symposium on Circuits and Systems (MWSCAS) 2020 Aug 9 (pp. 325-328). IEEE.
7. Sadeghi S. Classifying FPGA Technology in Digital Signal Processing: A review.
8. Wolf W. FPGA-based system design. Pearson education; 2004 Jun 15.
9. LaMeres BJ, Gauer C. Dynamic reconfigurable computing architecture for aerospace applications. In 2009 IEEE Aerospace conference 2009 Mar 7 (pp. 1-6). IEEE.
10. Dandekar O, Plishker W, Bhattacharyya S, Shekhar R. Multiobjective optimization of FPGA-based medical image registration. In 2008 16th International Symposium on Field-Programmable Custom Computing Machines 2008 Apr 14 (pp. 183-192). IEEE.
11. Belabed T, Coutinho MG, Fernandes MA, Sakuyama CV, Souani C. User driven FPGA-based design automated framework of deep neural networks for low-power low-cost edge computing. IEEE Access. 2021 Jun 17;9:89162-80.
12. Monmasson E, Idkhajine L, Cirstea MN, Bahri I, Tisan A, Naouar MW. FPGAs in industrial control applications. IEEE Transactions on Industrial informatics. 2011 Mar 24;7(2):224-43.