MANTECH Publications

Journal of Embedded Systems & Its Applications

Volume 6 Issue 2, July-December 2021

FPGA-Based Embedded Systems for High-Performance Computing

Joginder Singh, Atul Dixit

Associate Professor, Students

Department of Electronics and Communication

Zenith Technical University, India

Jogindersingh08@gmail.com, dixit51atul@yahoo.com

Abstract

Field-Programmable Gate Arrays (FPGAs) have emerged as a pivotal technology for embedded systems in high-performance computing (HPC) applications. Unlike traditional processors, FPGAs provide hardware-level parallelism, configurability, and low-latency processing, making them suitable for computationally intensive tasks. This paper reviews the architecture, design methodologies, and application areas of FPGA-based embedded systems for HPC. It also explores performance comparisons with GPUs and CPUs, the challenges in integration, and emerging trends, such as FPGA-accelerated AI and reconfigurable HPC clusters. The paper concludes with insights into future research directions and design considerations for optimizing FPGA-based embedded systems.

Keywords: *FPGA, Embedded Systems, High-Performance Computing, Hardware Acceleration, Reconfigurable Computing, Low-Latency Processing*

1. Introduction

High-performance computing has traditionally relied on multi-core CPUs and GPUs for computational acceleration. However, as data-intensive applications, such as artificial intelligence (AI), cryptography, and scientific simulations, demand lower latency and higher energy efficiency, alternative architectures have gained attention. FPGAs, due to their reconfigurability and ability to implement highly parallel hardware pipelines, are increasingly integrated into embedded systems for HPC applications.

Embedded FPGA systems combine programmable logic with microprocessors, memory interfaces, and I/O peripherals in a single platform, enabling customization at the hardware level. This integration allows designers to optimize algorithms and data paths for specific applications, leading to performance gains unattainable with general-purpose processors alone.

Page 75-87 © MANTECH PUBLICATIONS 2025. All Rights Reserved

2. Overview of FPGA-Based Embedded Systems

FPGA-based embedded systems integrate programmable hardware with software components to create a flexible, high-performance platform. Unlike traditional microprocessors or microcontrollers, FPGAs allow designers to implement custom hardware pipelines, which can execute tasks in parallel, dramatically improving performance for compute-intensive workloads.

An FPGA-based embedded system typically includes:

- **An FPGA device** for configurable hardware acceleration
- **A processor core** (hard-core like ARM Cortex or soft-core like MicroBlaze)
- **Memory modules** (on-chip BRAM and off-chip DDR)
- **Peripheral interfaces** (Ethernet, USB, GPIO)
- **Power and clock management circuits**

These components together allow developers to offload computationally demanding tasks to the FPGA while using the processor for general control logic.

2.1 FPGA Architecture

FPGAs are composed of several specialized components that enable configurable hardware. These components work together to implement complex embedded systems:

2.1.1 Configurable Logic Blocks (CLBs)

CLBs are the fundamental building blocks of an FPGA. Each CLB contains **look-up tables (LUTs)**, **flip-flops**, and multiplexers.

- **Look-Up Tables (LUTs):** Implement combinational logic by mapping input combinations to output values.
- **Flip-Flops:** Store intermediate data and implement sequential logic.
- **Multiplexers:** Route signals within the CLB to enable flexible logic functions.

CLBs allow developers to create custom arithmetic units, control logic, or even entire processor cores in hardware. In HPC applications, CLBs can be configured to run multiple operations in parallel, improving throughput significantly.

2.1.2 Programmable Interconnects

Programmable interconnects are networks of wires and switches that connect CLBs, DSPs, memory blocks, and I/O.

Page 75-87 © MANTECH PUBLICATIONS 2025. All Rights Reserved

- They allow **flexible routing** of signals across the FPGA fabric.
- Interconnect design is critical for **timing optimization**, as long paths can increase signal delay.
- High-performance FPGAs include **hierarchical routing** with local, regional, and global interconnects to balance flexibility and speed.

Effective use of interconnects ensures that parallel operations in different CLBs can communicate efficiently, which is vital for HPC workloads requiring massive data movement.

2.1.3 Block RAM (BRAM)

BRAM is on-chip memory embedded within the FPGA fabric.

- Provides **fast, low-latency storage** for temporary data and intermediate results.
- Configurable in size and width, often in blocks of 18–36 Kb.
- Can be used as **single-port or dual-port memory**, enabling simultaneous read/write operations.

In HPC applications, BRAM is often used to store frequently accessed data, such as input matrices for scientific simulations or weights for neural network inference, reducing dependency on slower off-chip memory.

2.1.4 Digital Signal Processing (DSP) Blocks

DSP blocks are specialized hardware units optimized for arithmetic operations.

- Include **multipliers, adders, and accumulators**.
- Ideal for operations such as **matrix multiplication, convolution, and filtering**, common in AI and scientific computing.
- Offloading arithmetic-heavy tasks to DSP blocks frees CLBs for control logic, improving overall FPGA efficiency.

DSPs allow FPGA-based embedded systems to handle **high-throughput numerical computations** with minimal latency, a key advantage over general-purpose processors in HPC tasks.

2.1.5 I/O Blocks

I/O blocks interface the FPGA with external devices, memory modules, and sensors.

- Support a wide range of communication protocols: **UART, SPI, I2C, Ethernet, PCIe**.

Page 75-87 © MANTECH PUBLICATIONS 2025. All Rights Reserved

- Include **signal conditioning and voltage level conversion** for compatibility with external devices.
- Critical for **embedded HPC systems** that must communicate with accelerators, storage, and sensor networks.

By providing high-speed I/O channels, FPGA-based systems can **stream large datasets directly into the FPGA fabric**, enabling real-time processing for applications like financial analytics or autonomous systems.

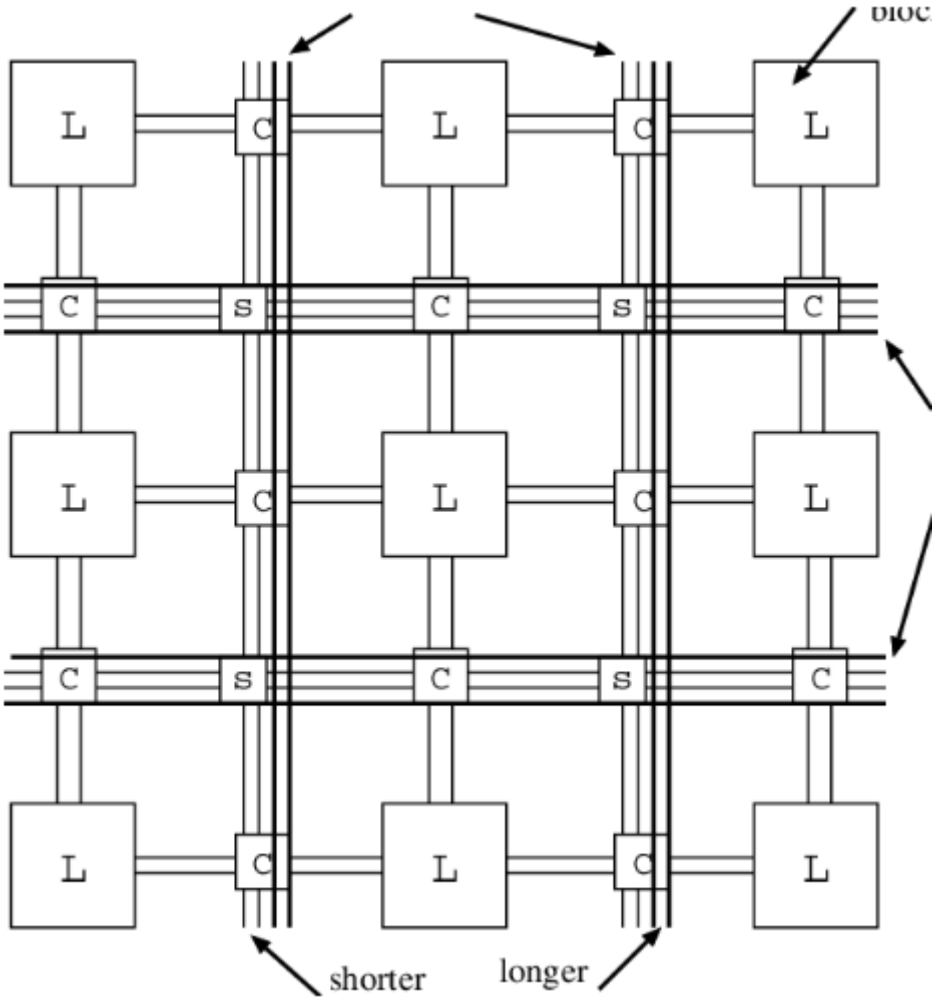

Figure 1: Basic FPGA Architecture

2.2 Embedded System Integration

FPGA-based embedded systems achieve their high performance by **combining programmable hardware (FPGA fabric) with embedded processors** in a single platform.

Page 75-87 © MANTECH PUBLICATIONS 2025. All Rights Reserved

This integration often takes the form of a **System-on-Chip (SoC)**, where the processor and FPGA fabric share memory, peripherals, and interconnects to work seamlessly.

2.2.1 Architecture of FPGA-Based SoC Systems

A typical FPGA-based SoC integrates:

1. **Embedded Processor:**
    - Examples include ARM Cortex-A or Cortex-R series, RISC-V cores, or soft processors like Xilinx MicroBlaze.
    - Handles general-purpose tasks, system control, and operating system management.
    - Executes tasks that are **not computationally intensive** or do not require parallelism.
2. **FPGA Fabric:**
    - Composed of CLBs, BRAM, DSPs, and programmable interconnects (as explained in Section 2.1).
    - Implements **compute-intensive operations in hardware** such as matrix multiplications, encryption, signal processing, or AI inference.
    - Tasks executed in FPGA fabric run in **highly parallel pipelines**, often achieving much higher throughput than software running on a CPU.
3. **Memory Hierarchy:**
    - Shared memory is typically divided between **on-chip BRAM**, **cache memory**, and **off-chip DDR memory**.
    - High-speed interfaces (e.g., AXI, PCIe) allow the processor and FPGA fabric to access data efficiently.
4. **Peripheral Interfaces:**
    - I/O controllers connect sensors, storage, networks, and other peripherals.
    - The processor usually manages I/O control, while FPGA can handle **high-speed streaming data** directly.

2.2.2 Hardware-Software Partitioning

In FPGA-based embedded systems, tasks are split between the **processor (software)** and **FPGA fabric (hardware)** according to performance requirements:

Page 75-87 © MANTECH PUBLICATIONS 2025. All Rights Reserved

Component	Tasks	Example in HPC Applications
Embedded Processor	Control logic, OS, low-compute tasks	Task scheduling, data pre-processing
FPGA Fabric	High-performance pipelines, parallel ops	AI inference, matrix operations, real-time simulations

This **hardware-software co-design** ensures:

- **Optimized throughput:** Compute-intensive tasks run in dedicated hardware.
- **Reduced latency:** Data pipelines in FPGA fabric avoid processor bottlenecks.
- **Energy efficiency:** Hardware execution consumes less power than software loops.

2.2.3 Advantages of FPGA-Based Embedded System Integration

1. **High Parallelism:**
    - FPGA fabric can implement thousands of operations simultaneously, unlike CPUs, which execute instructions sequentially or in limited parallel cores.
    - This is especially beneficial for **AI inference, large matrix multiplications, and scientific simulations**.
2. **Low-Latency Processing:**
    - Dedicated hardware pipelines in FPGAs eliminate the overhead of instruction fetch and decode in CPUs.
    - Real-time applications, such as **financial trading, autonomous vehicles, and video processing**, gain significant latency reduction.
3. **Hardware Flexibility Post-Deployment:**
    - FPGAs are reconfigurable, allowing updates to **hardware logic without changing the physical device**.
    - For embedded HPC systems, this means **new algorithms or optimizations can be deployed on existing hardware**, extending the system's lifespan.
4. **Customizable I/O and Memory Access:**
    - Embedded SoCs allow **custom memory interfaces** or **high-speed streaming I/O**, which are difficult to achieve in standard CPUs or GPUs.

Page 75-87 © MANTECH PUBLICATIONS 2025. All Rights Reserved

Table 1: Comparison of FPGA-Based Embedded Systems with CPU/GPU

Feature	FPGA	CPU	GPU
Parallelism	Fine-grained hardware	Limited multi-thread	Massive SIMD cores
Latency	Low	Moderate	Moderate
Power Efficiency	High	Low	Medium
Reconfigurability	High	Low	Low
Suitability for HPC	Excellent	Moderate	High

3. Design Methodologies

3.1 High-Level Synthesis (HLS)

High-Level Synthesis allows designers to write hardware descriptions in C/C++ or OpenCL, which are then converted to RTL (Register Transfer Level) for FPGA implementation. HLS accelerates development and reduces errors compared to traditional HDL coding.

3.2 Hardware-Software Co-Design

Optimal FPGA-based embedded systems require partitioning tasks between hardware and software. Critical operations are offloaded to FPGA fabric, while less intensive tasks remain in software. Profiling tools are used to identify performance bottlenecks.

3.3 Reconfigurable Pipelines

FPGAs can implement multiple pipelines for parallel execution of tasks. Reconfigurable pipelines allow runtime adaptation of hardware for varying workloads, which is particularly useful in AI inference and data processing tasks.

4. Applications in High-Performance Computing

4.1 Scientific Simulations

Simulations in physics, chemistry, and climate modeling require massive parallel computation. FPGAs accelerate matrix operations, Fourier transforms, and other numerically intensive tasks.

4.2 AI and Machine Learning

AI workloads, including convolutional neural networks (CNNs) and recurrent neural networks (RNNs), benefit from FPGA acceleration due to their ability to execute custom arithmetic pipelines with low precision formats (e.g., INT8, FP16).

Page 75-87 © MANTECH PUBLICATIONS 2025. All Rights Reserved

4.3 Cryptography and Blockchain

FPGAs provide hardware-level acceleration for cryptographic algorithms like AES, RSA, and SHA. This reduces processing time and power consumption for blockchain validation and secure communications.

4.4 Financial Computing

Low-latency trading systems and real-time risk analysis leverage FPGA-based embedded systems to achieve microsecond-level response times, outperforming CPU-only architectures.

5. Performance Evaluation

Evaluating the performance of FPGA-based embedded systems in high-performance computing (HPC) applications is crucial for understanding their advantages and limitations compared to traditional processors such as CPUs and GPUs. Performance evaluation helps designers choose the right platform, optimize system design, and justify hardware acceleration for specific workloads.

5.1 Metrics

Several key metrics are used to evaluate the performance of FPGA-based HPC systems:

5.1.1 Throughput

- **Definition:** Throughput measures the number of tasks or operations completed per unit of time, typically expressed in tasks per second or operations per second.
- **Relevance to HPC:**
    - High-throughput systems are essential for applications that process large volumes of data in parallel, such as matrix multiplications, neural network inference, or scientific simulations.
    - FPGAs achieve high throughput by implementing **massively parallel pipelines**, allowing multiple operations to be executed simultaneously in hardware.
- **Example:** A convolutional neural network (CNN) inference pipeline on an FPGA can process thousands of image pixels concurrently, achieving higher throughput than a CPU running the same network sequentially.

5.1.2 Latency

- **Definition:** Latency is the time taken to complete a single task from start to finish, typically measured in microseconds or milliseconds.

Page 75-87 © MANTECH PUBLICATIONS 2025. All Rights Reserved

- **Relevance to HPC:**
  - Low-latency processing is critical for real-time applications, such as autonomous vehicle sensor fusion, financial trading, and live video analytics.
  - FPGAs minimize latency by **eliminating instruction fetch/decode overhead** and executing tasks in dedicated hardware pipelines.
- **Example:** In a real-time trading system, FPGA-based processing of market data can reduce latency to a few microseconds, compared to tens or hundreds of microseconds on CPUs or GPUs.

5.1.3 Power Efficiency

- **Definition:** Power efficiency is the amount of computation achieved per unit of energy, typically expressed as operations per watt.
- **Relevance to HPC:**
  - Energy efficiency is especially important for embedded HPC systems and edge devices with limited power budgets.
  - FPGAs are more energy-efficient than GPUs for tasks with custom hardware implementations because they avoid unnecessary general-purpose execution overhead.
- **Example:** AI inference on edge devices like drones or robots can run longer and consume less energy when accelerated by FPGA fabric compared to GPU-based solutions.

5.1.4 Area Utilization

- **Definition:** Area utilization measures the fraction of the FPGA's logic, DSP blocks, and memory that is consumed by a particular design.
- **Relevance to HPC:**
  - Efficient use of FPGA resources allows more tasks or pipelines to run in parallel without exceeding hardware capacity.
  - Poor area utilization can lead to routing congestion and timing violations, reducing overall system performance.
- **Example:** A design using 70% of CLBs, 60% of DSP blocks, and 50% of BRAM demonstrates efficient but not over-constrained utilization, leaving room for future expansion.

Page 75-87 © MANTECH PUBLICATIONS 2025. All Rights Reserved

5.2 Comparative Studies

Recent research and benchmarking studies highlight the **strengths and weaknesses of FPGA-based HPC systems** compared to CPUs and GPUs. Key observations include:

5.2.1 Real-Time Video Processing

- FPGA-based pipelines excel at **low-latency streaming operations**, such as video encoding, decoding, and object detection.
- Example: An FPGA video pipeline can process multiple HD video streams simultaneously with latency under 5 milliseconds, outperforming GPUs for real-time requirements.

5.2.2 AI Inference on Edge Devices

- FPGAs can implement **custom precision arithmetic (INT8, FP16)** optimized for AI models, significantly reducing power consumption and latency.
- Studies show that for small to medium CNNs deployed on embedded devices, FPGAs can achieve similar or better throughput than GPUs while consuming **30–50% less energy**.

5.2.3 Cryptographic Workloads

- FPGAs are widely used to accelerate encryption, decryption, and hashing algorithms due to **dedicated DSP blocks** and the ability to pipeline operations efficiently.
- Example: FPGA implementations of AES or SHA-256 can achieve throughput of several gigabits per second with microsecond-level latency, outperforming CPU and sometimes GPU solutions in low-latency scenarios.

5.2.4 Mixed Workloads and Heterogeneous Systems

- In many HPC applications, FPGAs are combined with CPUs or GPUs to form **heterogeneous platforms**.
- Tasks that require massive parallel computation or low-latency pipelines are offloaded to FPGA, while CPUs handle control logic and system management.
- Comparative studies indicate that **hybrid FPGA-CPU systems** can outperform GPU-only systems in applications requiring both throughput and deterministic low-latency response.

Page 75-87 © MANTECH PUBLICATIONS 2025. All Rights Reserved

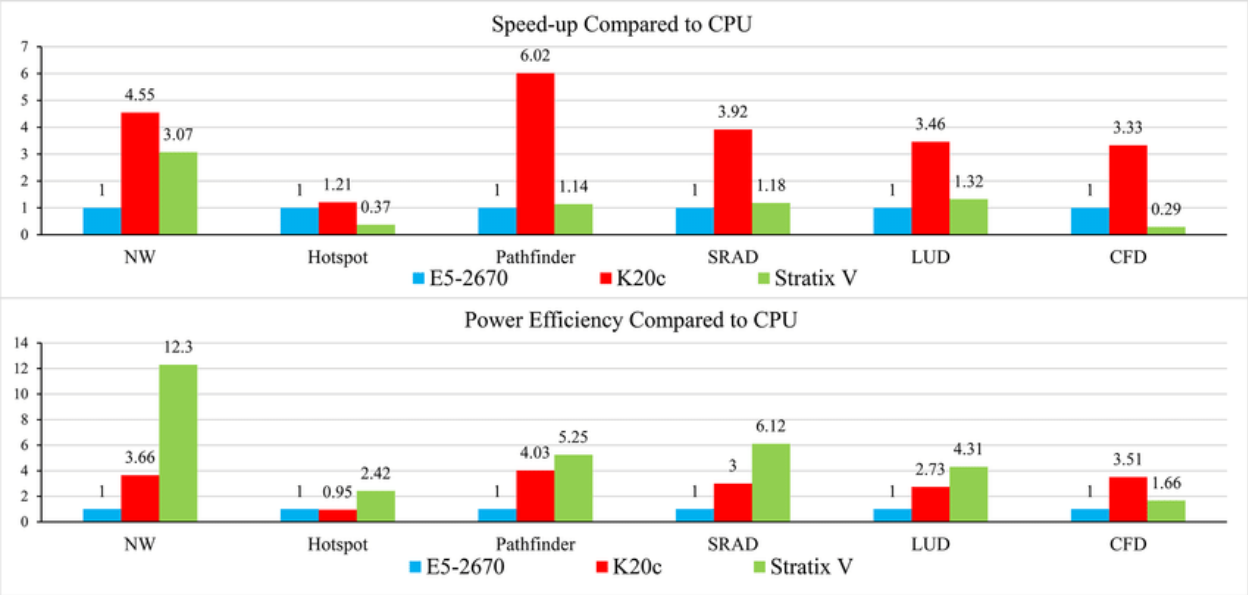

Figure 2: Performance Comparison of FPGA vs GPU vs CPU

6. Challenges and Limitations

6.1 Programming Complexity

While HLS simplifies FPGA programming, designing optimized hardware still requires understanding of hardware timing, parallelism, and memory hierarchy.

6.2 Resource Constraints

FPGAs have limited logic and memory resources. Complex HPC workloads may exceed available capacity, necessitating careful design partitioning.

6.3 Integration with Existing Systems

Embedding FPGAs into legacy HPC clusters requires compatible interfaces, drivers, and middleware, adding complexity to deployment.

6.4 Cost Considerations

High-end FPGAs are expensive compared to commodity GPUs, which may limit widespread adoption in certain HPC applications.

7. Emerging Trends

7.1 FPGA-Accelerated AI

Hybrid systems combining FPGAs with CPUs and GPUs for AI inference are gaining traction. FPGAs handle low-latency operations while GPUs perform bulk training.

Page 75-87 © MANTECH PUBLICATIONS 2025. All Rights Reserved

7.2 Reconfigurable HPC Clusters

Cloud providers are integrating FPGA nodes into HPC clusters, allowing users to dynamically allocate reconfigurable hardware for specific tasks.

7.3 Energy-Efficient Computing

FPGAs offer superior energy efficiency, making them suitable for sustainable HPC solutions and edge AI applications.

8. Conclusion

FPGA-based embedded systems present a promising solution for high-performance computing tasks, combining flexibility, parallelism, and energy efficiency. Despite challenges in programming complexity and resource limitations, advancements in HLS, AI acceleration, and reconfigurable clusters enhance their viability. Future research should focus on improving FPGA programmability, optimizing heterogeneous HPC architectures, and exploring novel applications in AI, scientific computing, and secure financial systems. Properly designed FPGA-based embedded systems have the potential to redefine performance benchmarks in high-performance computing.

References

1. Xilinx Inc., *"Zynq UltraScale+ MPSoC Technical Reference Manual"*, Xilinx, 2022.
2. Chen, Y., et al., *"High-Level Synthesis for FPGAs: From C to Hardware,"* IEEE Trans. CAD, 2020.
3. Putnam, A., et al., *"A Reconfigurable Fabric for Accelerating Large-Scale Datacenter Services,"* IEEE Micro, 2014.
4. Mittal, S., *"A Survey of FPGA-Based Accelerator Architectures for Deep Learning,"* Journal of Systems Architecture, 2019.
5. Li, P., et al., *"FPGA-Accelerated Scientific Computing: Applications and Case Studies,"* ACM Computing Surveys, 2021.
6. Reagen, B., et al., *"Minerva: Enabling Low-Power, Highly-Accurate Deep Neural Network Accelerators,"* ISCA, 2016.
7. Anderson, J., et al., *"Accelerating Financial Applications Using FPGAs,"* IEEE High-Performance Extreme Computing Conf., 2018.
8. Huang, R., et al., *"Energy-Efficient FPGA-Based AI Acceleration,"* ACM Transactions on Embedded Computing Systems, 2020.

Page 75-87 © MANTECH PUBLICATIONS 2025. All Rights Reserved

9. Altera Corporation, *"Stratix 10 Device Handbook,"* Altera, 2019.
10. Nurvitadhi, E., et al., *"Can FPGAs Beat GPUs in Accelerating Next-Generation Deep Neural Networks?"* FPGA, 2017.

Page 75-87 © MANTECH PUBLICATIONS 2025. All Rights Reserved