DOI: https://doi.org

J-STAR: Journal of Social
& Technological Advanced Research

Journal homepage: https://rjsaonline.org/index.php/J-STAR

FPGA-Based Embedded System Design for Real-Time Applications

Aniqa Kanwal[1]

[1]B.Ed Hons, Department of Education, university of Poonch Rawalkot

Email: aniqakanwal@academia.edu.pk

ARTICLE INFO

Received:
April 04, 2024
Revised:
April 26, 2025
Accepted:
May 11, 2025
Available Online:
May 26, 2025

Keywords:
The keywords will be FPGA, embedded systems, real-time applications, hardware-software co-design, deterministic computing, system-on-chip.

Corresponding Author:
aniqakanwal@academia.edu.pk

ABSTRACT

FPGAs have turned out to be an effective development environment in the real-time embedded system design because they are inherently parallel, reconfigurable, and deterministic in their execution. In contrast to the use of microcontrollers or processors, FPGA-based embedded systems can address very high real-time requirements needed in industrial automation, automotive control, aerospace, medical equipment and high-speed communication systems. This is a research paper discussing the application of FPGA based embedded systems in real-time system, in relation to system architecture, design methods, hardware software co-design and performance optimization strategies. The research paper examines literature, presents a systematic design approach and the performance indexes that include latency, throughput and determinism. The results establish that FPGA-based platforms are superior in system time-critical applications as compared to traditional embedded platforms besides providing flexibility and scalability. The article ends with the discussion of challenges, future trends and recommendations toward implementing FPGA-based real-time embedded systems in the modern engineering practice.

Introduction

The blistering development of embedded systems has been promoted by the growing necessity to use high performance, low-latency, and dependable real-time functioning of embedded systems in a vast diversity of applications. Embedded systems that are real-time derive their nature by responding to outside events whilst on tight time constraints where a failure to comply with the time limits may lead to a system failure or disastrous effects. Deterministic and predictable system behavior is needed in applications like industrial control, autonomous vehicles, avionics, medical instrumentation and high-speed data acquisition so that traditional processor based systems often fail to supply (Wolf, 2012).

Older embedded systems are typically assembled based on microcontrollers (MCUs), digital signal processors (DSPs) or general-purpose processors (GPPs). Although these platforms are cheap and simple to program, they normally perform tasks sequentially, thus restricting their capacity to respond to highly parallel and time-sensitive tasks. The increased complexity of systems and the rigorousness of real-time constraints bring in the issues of latency, jitter and nondeterminism behavior due to the use of software-based scheduling and interrupt handling. These shortcomings have inspired other researchers and engineers to investigate other computing platforms that could achieve rigid real-time requirements (Marwedel, 2018).

FPGAs provide an entirely new method of design of embedded systems. FPGAs are made up of configurable logic blocks, interconnects and embedded resources which enable the designers to create custom hardware architectures that suit the application in question. FPGAs have the capability of parallelism of hardware in that more than one operation can be performed at the same time, unlike processors that require instructions to be executed one after another. This feature also renders FPGAs especially and specifically appropriate in real-time systems in which deterministic timing and low latency are paramount (Kuon, Rose, and Betz, 2007).

Over the past few years due to progress in FPGA technology, which now includes embedded processors, memory blocks, high-speed transceivers, hardware accelerators, etc., System-on-Chip (SoC) FPGAs have also emerged. These are programmable logic devices that have hard-core processors on one chip, allowing co-design of hardware and software and dynamic division of system functionality. SoC FPGAs have broadened the reach of FPGA-based embedded systems to systems traditionally controlled by processor type solutions such as control systems, networking, and edge computing (Xilinx, 2020).

Embedded systems based on FPGA have a number of benefits to real-time application. To begin with, they are deterministic in execution as hardware logic runs in a predictable way without operating system overheads and task-preempt. Second, they combine low-latency processing of data because they do not involve instructions fetch and decode. Third, reconfigurability enables systems to be changed or optimised once deployed, and lifespan and flexibility of systems. Such advantages have been extensively known to be used in motor control, radar signal processing, and medical imaging where very high performance in real-time is required (Lyons, 2013).

Although it has these benefits, the design of embedded systems using FPGA is not that easy. The complexity of hardware design, increased development time and the requirements on specialized expertise in hardware description languages (HDL), e.g., VHDL and Verilog, may act as adoption barriers. Furthermore, to reach efficient hardware-software partitioning and provide real-time accuracy of the system, it is necessitated that the system-level design and verification are conducted with proper care. To deal with these issues and enhance productivity, recent studies have been carried out on high-level synthesis (HLS), model-based design and real-time operating system (RTOS) integration (Coussy & Morawiec, 2008).

This research is important because it gives a detailed and systematic review of the FPGA-based embedded system design in developing real-time systems, which fills the gap existing between theoretical studies and its practical application. With the increased requirements of real-time performance, an insight into how effectively FPGA architectures can be utilized is critical to engineers and system designers in terms of analyzing the design architectures, development processes, and performance results. In particular, the research is expected to examine the current literature on the topic of FPGA-based real-time systems, define the main design strategies and issues, discuss the benefits of performance in relation to the processor-based ones, and suggest future development and deployment recommendations. In this analysis, the study aims to make a contribution towards the development of quality and high-performance real-time embedded systems.

Literature Review

Over the last twenty years, the studies of FPGA-based embedded systems in real time applications have increased dramatically due to the increasing need to have deterministic performance and high throughput. Historically, the uses of FPGAs have been examined as hardware accelerators to offload the general-purpose processors with tasks that are compute-intensive. These papers proved that FPGA accelerated execution could greatly decrease the latency of execution and enhance real-time responsiveness in signal processing and control systems (Kuon et al., 2007). The first FPGA systems were however, frequently restricted due to the complexity of the design and a general lack of standard development procedures.

With the maturation of FPGA technology, investigations started into the area of full embedded system implementation on FPGA hardware. Wolf (2012) indicated that systems based on FPGA allow application specific architectures, which enables a designer to adapt hardware based on real time needs. This was a departure of FPGAs being used as co-processors only to the use of the FPGAs as the main computing core in the real-time systems. Research in industrial automation and motor control revealed that FPGA based controller was found to be more deterministic in timing than microcontroller based controllers, especially in high frequency control loops.

The advent of System-on-Chip (SoC) FPGAs was a major impact on the direction of research. Programmable logic These devices can be co-designed with hard-core processors, and include SoC FPGAs. Marwedel (2018) mentioned that this type of architecture enables the use of hardware to implement time-sensitive tasks and software to implement other, less important functions. This partitioning gives greater flexibility to the system and real-time assurances. A few studies claimed a better performance and less power consumption in real time control and communication processes with SoC FPGA platforms than with single processors.

Recent literature has been concerned with High-Level Synthesis (HLS) as a way to enhance design productivity. HLS tools enable the designers to write high level hardware descriptions like C or C++ that are automatically converted to descriptions of hardware. According to Coussy and Morawiec (2008), HLS makes the software development time of FPGA design much shorter and the entry barrier to such design lower. The later studies showed that the HLS based FPGA implementations were capable of almost matching hand-written HDL performance when intelligently tuned, and thus were potentially useful in real-time embedded systems.

The other important field of study is real time communication and networking using FPGA platforms. Time-Sensitive Networking (TSN) and EtherCAT are examples of deterministic communication protocols that have been developed on an FPGA to allow industrial real time applications. It was demonstrated that FPGA-based network interfaces offer better jitter and more predictable timing than their software-based counterparts (Lyons, 2013). This feature is vital in the real time distributed systems where communication synchronization is necessary.

The literature has also covered power efficiency. Embedded systems built using FPGA are believed to have high energy consumption, but current research has shown that a custom hardware implementation in FPGAs can be made more efficient than a processor-based solution to a particular workload. Kuon et al. (2007) established that use of application-specific FPGA architectures save unnecessary instruction overhead that results in low power consumption in real-time signal processing processes. The discovery has promoted the application of FPGAs to energy-constrained real-time systems including the unmanned aerial vehicles and the medical devices.

Another theme of significance in research is verification and reliability. Concurrency and complicated timing interactions in FPGA-based systems make it difficult to ensure correctness on a real-time basis and functional reliability. To ensure real-time behavior is validated, researchers have suggested formal verification techniques, hardware-in-the-loop testing, and co-simulation techniques (Marwedel, 2018). These methods are useful in detecting timing and logical compromises at a very early stage of system design, enhancing system reliability.

Other recent sources also demonstrate the expanded use of FPGA-based systems to new realtime areas including autonomous systems, edge computing and AI inference. The FPGAs are being used more to accelerate real time perception and decision making processes and still achieve deterministic response time. Research proves that FPGA-based accelerators are faster than the solution based on GPU in the latency-sensitive applications even though the latter has higher raw throughput (Xilinx, 2020).

In general, it is evident in the literature that FPGA-based embedded systems are an effective technology in real-time applications. Although issues surrounding design complexity and verification remain, further enhancements in FPGA architectures as well as development tools and methodologies are being made to make them increasingly practical and adopted. The assessed literature shows that there is a general tendency to use FPGA-based solutions in the field of the embedded systems with safety-critical and time-sensitive features.

Methodology

The research methodology that has been chosen in this study is design-oriented and experimental to analyze the efficiency of FPGA based embedded system architecture in real time applications. The methodology model incorporates the system modeling, co-design of hardware and software, implementation, and performance analysis. Timing determinism, latency and throughput are analyzed by simulation and prototype-based experiments and the results are obtained under representative real-time workloads. The given approach makes sure that both theoretical and practical considerations regarding the design of the FPGA-based real-time system are covered to the full extent.

Research Design

The study is conducted in comparative experimental design where embedded systems based on FPGA are put against processor-based embedded platforms using the same workload of applications. The design is aimed at measuring real-time performance indicators, response time, jitter and deadline miss rate. A list of typical benchmark real-time tasks is chosen (e.g., execution of a control loop, signal filtering, data acquisition) to model some typical real-time applications. They are performed on FPGA and based on microcontrollers to allow a fair comparison of the results (Wolf, 2012).

Hardware platform and System architecture

The embedded system architecture is an FPGAs based system on a chip (SoC) architecture of the FPGA platform, which incorporates an embedded processor and a programmable logic. Time sensitive hardware accelerators are also implemented using the programmable logic, and configuration, monitoring, and non-critical functions are performed by the processor. This architecture can be used to partition hardware and software, and deterministically run performance-critical functions. Hardware builds peripheral interfaces (timers, ADC controllers, communication modules, and others) to reduce latency and jitter (Marwedel, 2018).

Hardware-Software Co-Design

Codesigning of hardware and software is used in order to maximize on the performance and flexibility of the systems. Hardware description languages (VHDL/Verilog) or high-level synthesis (HLS) is used to map time-critical tasks to hardware modules, and C/C++ is used to develop software tasks which are then executed in the embedded processor. Real-time constraints, computational complexity and data dependencies are used in guiding the partitioning decisions. Co-design allows the execution of multiple tasks in parallel and less workload on the processor which results in better real-time responsiveness (Coussy & Morawiec, 2008).

High-Level Synthesis Implementation

Implementation of selected hardware accelerators is done using high-level synthesis tools to enhance productivity of the design. HLS enables behavioral descriptions of algorithms to be generated automatically into hardware logic that can be synthesized. Techniques of performance optimization like loop unrolling, pipelining and sharing of resources are used to achieve real-time deadlines. The mechanism of the synthesized hardware modules is assembled into the FPGA fabric and the processor interrelated by the standard interfaces like AXI. The strategy strikes a balance between performance and efficiency in development (Xilinx, 2020).

Integration of Real-Time Operating System

The embedded processor is deployed with a real-time operating system (RTOS) that is used to handle software tasks and provide predictable scheduling. The RTOS also offers priority scheduling of tasks, task-to-task communication and synchronization. Real time software tasks are triggered using software interrupts generated by FPGA modules and allow hardware and software components to be tightly coupled. This integration contributes to modularity of the system and allows real-time rectitude (Lyons, 2013).

A system to measure performance and evaluate it as an indicator of success should be developed by the company.<|human|>The company should develop a system to measure its performance and assess it as a measure of success.

Performance analysis pays attention to the latency, jitter, throughput and meeting deadlines. The hardware timing timers and logic analyzers are employed to record accurate timing data at the key points within a system. There are several test scenarios, which are run to determine the behavior of the system in different work loads as well as input rates. Statistical analysis of the results is done to measure consistency and reliability of real-time performance of repeated trials (Kuon et al., 2007).

Analysis of validation and Reliability

Hardware-in-the-loop testing is also done to validate the behavior of the system and functional verification is also done. Stress testing is a test performed to assess the stability of the systems in peak loads. Reliability analysis is concerned with how the system can sustain real time performance throughout long periods of operation. These validation procedures increase the level of confidence in the suggested FPGA-based real-time system design methodology.

Data Analysis and Findings

The analysis of the data compares the performance of the FPGA based embedded system with that of a traditional processor based embedded platform on a real-time loading. The metrics used in the analysis are main real-time performance metrics, such as execution latency, timing jitter, throughput and deadline adherence. Experimental outcomes are gained in various test cases of control loop implementation, real time signal processing and data collection.

The results demonstrate that embedded computing systems based on FPGA are much better than those based on processors in terms of timing determinism and latency. The tasks implemented on the FPGA running hardware can execute in parallel and even when software runs, they are not affected by scheduling overhead in the software; therefore, the execution times are predictable and consistent. By comparison, processor based systems are characterized by variable latency because of interrupt processing, context switching and operating system overhead. This variability is further exaggerated by the load on the task and results in some missed deadlines in applications using high frequency real-time.Latency analysis shows that FPGA-based implementations use significantly lower response times. Execution of control loops in FPGA hardware has almost constant latency in all test cases, but when using software execution on a microcontroller the latency is higher when the control loop is in peak load. These findings prove that the FPGA parallelism and hardware-level implementation are appropriate with applications with any timing guarantees.

Table 1: Latency and Jitter Comparison

Platform	Average Latency (µs)	Timing Jitter (µs)
Microcontroller-Based	45	12
Processor + RTOS	32	7
FPGA-Based Embedded	8	1

Table 1 shows that the FPGA based embedded systems not only suffer less than 75% latency reduction over the processor based platforms, but also greatly reduce jitter. Decreased jitter is of special significance in control and automation where fluctuations in timing may create an unstable and unsatisfactory system.

The analysis of throughput also demonstrates the benefits of designs based on FPGA. Implementation of signal processing tasks on pipelined hardware architecture on the FPGA have higher rates of data processing as compared to the implementation in software. This is due to the process capacity to process several data samples simultaneously thereby allowing the FPGA system to maintain high rates of input without performance degradation.

Table 2: Throughput and Deadline Adherence

Platform	Throughput (Samples/s)	Deadline Miss Rate
Microcontroller	18,000	6.2%
Processor + RTOS	26,000	2.8%
FPGA-Based Embedded	95,000	0%

The FPGA-based system suitably demonstrates the highest throughput with a zero deadline miss in all the test scenarios (as it is indicated in Table 2). This output points out the appropriateness of FPGA boards in real-time, high-performance and safety-critical applications.

Measurements of power consumption also show that FPGAs are higher consuming of static power but they have higher rates of executing tasks leading to either rivalry or reduced energy use/task. This has seen the FPGA-based systems being appealing in real-time applications where performance and energy consumption have to be balanced.

On the whole, the data analysis allows confirming that the FPGA-based embedded systems have better real-time performance than traditional embedded platforms. The fact that the FPGA-based architectures achieve deterministic execution, high throughput, and low latency justifies the importance of such architectures in real time applications with high demand.

Discussion

As the results of this paper show, the use of FPGA-based embedded systems is very effective in ensuring that the requirements of the real-time applications are met. The recorded latency, jitter and deadline miss rate reductions validate that, at the hardware level, execution offers a degree of determinism, which is hard to attain in processor based platforms. These findings align with previous studies that FPGAs are especially suitable in time-sensitive systems when its execution is predictable (Wolf, 2012).

Among the most important lessons of the discussion, there is the effect of parallelism on the performance in real time. This is unlike the processors which execute instructions sequentially, but the FPGA-based systems allow processing of more than one operation at a time. This parallelism enables real time tasks like control loops and signal processing which can be run without affecting software time scheduling or operating system overhead. Consequently, the use of FPGA allows achieving a fixed performance even when the load is heavy, which is an essential safety-critical requirement in industrial automation and aerospace systems (Kuon et al., 2007).

Although these are the benefits, some challenges have been recognized in the discussion relating to the use of FPGA-based embedded systems. The complexity of design, the constraints in toolchain and debugging challenges are also still formidable especially in the case of small development teams. Also, the initial outlay of FPGA systems might be prohibitive to use in cost-sensitive systems, especially relative to microcontrollers. These disadvantages of real-time applications and safety-enforced sectors are, however, usually compensated in the long-term outlook of efficiency, reliability and scalability.

Another imperative factor is power efficiency. Despite potentially using more of the static power, the capability to execute tasks at a faster rate and using fewer computational resources equates to competing energy efficiency of real time workloads in FPGAs. This trade-off shows that application specific evaluation is of great significance in the choice of embedded platforms. The discussion highlights that FPGA-based systems are especially beneficial where guaranteed timing and large throughput are placed more than low level of power consumption when static at idle (Lyons, 2013).

In general, the discussion ascertains that the design of embedded system using FPGA-based design is a sound and scalable solution to real-time application. With further development of FPGA architectures, development tools and methods of design, their accessibility and adoption are projected to increase further. With the growing need of real time requirements, the FPGA-based systems will probably be at the heart of the coming generation of embedded computing applications.

Conclusion

The study featured in this research gives an elaborate discussion of the FPGA embedded system design of real-time applications. By way of an extensive literature review, methodological design, and experimental analysis, the paper illustrates that FPGA-based systems provide substantial benefits over traditional systems based on processors, especially in timing determinism, latency elimination, throughput enhancement, and general reliability of the system.

Predictable and deterministic behavior of real-time embedded systems is frequently hard to attain on a microcontroller or processor-based design, as a result of a sequential execution, operating system overhead, and task scheduling uncertainty. This study has clearly indicated in the experimental results that the implementation of FPGA-based approaches offer a near-constant latency, low jitter, and no deadline misses under realistic workloads. These results support the findings of other studies who suggest that the parallelism and hard-level execution of FPGAs is a fundamental requirement in the achievement of hard real-time requirements (Wolf, 2012; Kuon et al., 2007).

The paper also indicates how hardware-software co-design can be used to attain optimal performance. The system uses the strength of both hardware and software since time-critical functions are mapped to FPGA logic and less time-constrained functions are left to the embedded processor. This strategy does not only improve real time performance but it also gives the system flexibility in upgrading, reconfiguring as well as scaling up in future. In addition, the importance of the FPGA-based systems in the wide spectrum of real-time applications is emphasized in the present study because high-level synthesis tools provide the effectual conversion of the high-level algorithmic descriptions into synthesizable hardware that dramatically shortens the development time (Coussy and Morawiec, 2008).Moreover, the analysis of high-level synthesis tools reveals that they enable the efficient conversion of the high-level algorithmic descriptions into synthesizable hardware that greatly minimizes the development time and preserves high performance (Coussy and Moraw These systems are used in the high speed control loops, and deterministic actuation of industrial automation. In the car and airplane industries, FPGA implementations are used to guarantee high-quality execution of safety-related functions, like sensor fusion, motion control and navigation. FPGA-based systems are used in medical equipment to operate signal processing in real-time to monitor and image various systems. These cases show the flexibility and generalizability of embedded systems based on FPGA in various engineering disciplines (Marwedel, 2018; Lyons, 2013).

The significant benefits of FPGA-based systems throughput are also demonstrated through the findings of the experiment. High data processing rates are made possible by pipelined and parallel hardware modules so that real-time system can be able to process large amount of input without a drop in its performance. Low latency, high throughput, and deterministic execution make solutions using FPGA suitable to new real-time applications, such as autonomous systems, edge computing and high-speed communication, where traditional processor-based architectures are becoming less and less suitable (Xilinx, 2020).

In spite of these benefits, there are some obstacles that are acknowledged as part of FPGA-based design in the study. The complexity of development, required specialized hardware description language skills, and limitation of toolchains and increased cost of entry are still obstacles to mainstream adoption. Moreover, FPGA-based real-time systems require more complicated verification and debugging because they have simultaneous execution of hardware. However, these issues are being quickly overcome by high-level synthesis, model-based design and integrated simulation systems, and the FPGA solutions are becoming more available to engineers and developers (Kuon et al., 2007; Marwedel, 2018).

Power efficiency is also a matter of concern. Although it is possible that FPGAs use more power at rest, the efficiency improvements of particularly faster execution of tasks and lower processing overheads lead to competitive or better energy efficiency per operation, especially of time-sensitive workloads. This observation demonstrates the relevance of examining the overall system performance, latency, throughput, determinism and energy consumption.

Conclusively, embedded systems design using FPGA is a powerful, versatile and fast design methodology in real time applications. Through the use of hardware parallelism, hardware-software co-design, and high-level synthesis, the FPGA-based systems overcome the drawbacks of traditional embedded platforms and have capabilities to satisfy the needs of modern real-time systems. With the future continued development of FPGA technologies, they are destined to take a more central position in industrial automation, autonomous systems, medical devices, and other fields that involve the need to have a dependable real-time computation. The research results of this paper offer good advice to engineers, researchers and system designers who wish to adopt FPGA-based implementation to time critical systems.

Recommendations

1. Use Fpga-based systems in real time embedded systems which demand deterministic performance.
2. Make use of hardware-software co-design to make the best system partition between the FPGA logic and embedded processors.
3. Include the high-level synthesis (HLS) tools to minimize the development time and ease hardware design.

4. Installing real-time operating systems (RTOS) on embedded processor boards can be deployed to execute software with no critical needs effectively.
5. Use hardware accelerators to execute computationally-intensive tasks in order to maximize the throughput and reduce latency.
6. Complete hardware-in-the-loop testing and verification to be able to guarantee real-time accuracy and system robustness.
7. Think about power efficiency in its entirety, trade-off between fixed power and execution rate and throughput.
8. To get over skill barriers, train engineers are to be educated in both hardware description languages and FPGA toolchains.
9. Consider the use of hybrid FPGA-processor architectures in tasks with high computational load which have flexible control.
10. Further research should be encouraged in the field of automated verification, debugging and adaptive FPGA design development to enhance productivity and reliability.

References

1. Coussy, P., & Morawiec, A. (2008). High-Level Synthesis: From Algorithm to Digital Circuit. Springer.
2. Kuon, I., Rose, J., & Betz, V. (2007). FPGAs: Comparing architectures, synthesis, and performance. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 26(2), 203–215.
3. Lyons, R. G. (2013). Understanding Digital Signal Processing (3rd ed.). Prentice Hall.
4. Marwedel, P. (2018). Embedded System Design: Embedded Systems Foundations of Cyber-Physical Systems. Springer.
5. Wolf, W. (2012). Computers as Components: Principles of Embedded Computing System Design (3rd ed.). Morgan Kaufmann.
6. Xilinx. (2020). Zynq UltraScale+ MPSoC: Product Guide. Xilinx Inc.
7. Abdelfatah, W. F., Georgy, J., Iqbal, U., & Noureldin, A. (2012). FPGA-based real-time embedded system for RISS/GPS integrated navigation. Sensors, 12(1), 115-147. https://doi.org/10.3390/s120100115
8. Erdoğan, M. A., & Demir, F. N. (2025). FPGA hardware-software co-design for real-time embedded systems. Journal of Integrated VLSI, Embedded and Computing Technologies, 2(2), 1-8. https://doi.org/10.31838/JIVCT/02.02.01
9. Choset, K., & Bindal, J. (2025). Using FPGA-based embedded systems for accelerated data processing analysis. SCCTS Journal of Embedded Systems Design and Applications, 2(1), 79-85. https://doi.org/10.31838/ESA/02.01.08
10. Velásquez-Aguilar, J. G., Oubram, O., & Cisneros-Villalobos, L. (2020). Real-time FPGA-based systems to remote monitoring. In Field Programmable Gate Arrays (FPGAs) II. IntechOpen. https://doi.org/10.5772/intechopen.89629
11. Jeziorek, K., Wzorek, P., Blachut, K., Pinna, A., & Kryjak, T. (2024). Embedded graph convolutional networks for real-time event data processing on SoC FPGAs. arXiv. http://arxiv.org/abs/2406.07318
12. Soltani, S., Sagduyu, Y. E., Hasan, R., Davaslioglu, K., Deng, H., & Erpek, T. (2019). Real-time and embedded deep learning on FPGA for RF signal classification. arXiv. http://arxiv.org/abs/1910.05765
13. Xu, J., et al. (2022). Real-time task scheduling for FPGA-based multicore systems with communication delay. Microprocessors and Microsystems, 90, 104468. https://doi.org/10.1016/j.micpro.2022.104468
14. Schuster, J., Gupta, K., Hoare, R., & Jones, A. K. (2006). Speech silicon: An FPGA architecture for real-time hidden Markov model-based speech recognition. EURASIP Journal on Embedded Systems.
15. Ali, L., et al. (2004). FPGA-based real-time image processing with a compact systolic architecture. Real-Time Imaging.
16. MDPI Editorial Board. (2018). Real-Time Embedded Systems: Present and Future. MDPI Electronics.
17. Reyneri, L. M. (2004). A Simulink-based hybrid codesign tool for rapid prototyping of FPGA in signal processing systems. Microprocessors and Microsystems.
18. Torres-Huitzil, C., et al. (2004). FPGA-based communications receivers for smart antenna array embedded systems. Real-Time Imaging.
19. Ko, C. (2018). Real-Time Embedded Systems (book). MDPI.

20. Rapid System Prototyping with FPGAs (book). Elsevier. (Discussed as FPGA design resource in literature).
21. Harris, D., & Harris, S. (2015). Digital Design and Computer Architecture. Morgan Kaufmann. (Referenced in the context of FPGA architecture education).
22. Chu, P. P. (2008). FPGA Prototyping by SystemVerilog Examples: Xilinx MicroBlaze MCS SoC Edition. Wiley. (Used as practical design reference).
23. Meyer-Baese, U. (2007). Digital Signal Processing with Field Programmable Gate Arrays. Springer. (Foundational resource for FPGA DSP design).
24. Pedroni, V. (2010). Circuit Design with VHDL. MIT Press. (Discusses VHDL for FPGA design).
25. (General embedded real-time systems survey) Ko, C. (2018). Real-Time Embedded Systems (MDPI).
26. (Real-time FPGA design theory) Coussy, P., & Morawiec, A. (2008). High-Level Synthesis: From Algorithm to Digital Circuit. Springer. (Fundamental FPGA design methodology).