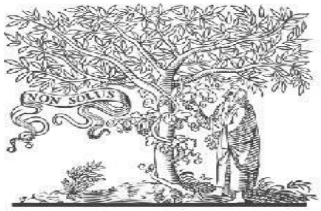
International Journal for Innovative Engineering and Management Research
PEER REVIEWED OPEN ACCESS INTERNATIONAL JOURNAL
www.ijiemr.org

COPY RIGHT



ELSEVIER SSRN

IJIEMR Transactions, online available on 26th Oct 2021. Link :http://www.ijiemr.org/downloads.php?vol=Volume-10&issue=Issue 11

10.48047/IJIEMR/V10/ISSUE 10/40

Title *AUTOMATED GENERATION OF VLSI ARCHITECTURES FOR DSP ALGORITHMS USING EVOLUTIONARY ALGORITHMS*
Volume 10, ISSUE 10, Pages: 254-260
Paper Authors **RAJEEV RANJAN VERMA DR. RAM MOHAN SINGH**

USE THIS BARCODE TO ACCESS YOUR ONLINE PAPER

To Secure Your Paper As Per UGC Guidelines We Are Providing A Electronic Bar Code

International Journal for Innovative Engineering and Management Research
PEER REVIEWED OPEN ACCESS INTERNATIONAL JOURNAL
www.ijiemr.org

AUTOMATED GENERATION OF VLSI ARCHITECTURES FOR DSP ALGORITHMS USING EVOLUTIONARY ALGORITHMS

RAJEEV RANJAN VERMA
Research Scholar, Sunrise University, Alwar, Rajasthan
DR. RAM MOHAN SINGH
Research Supervisor, Sunrise University, Alwar, Rajasthan

ABSTRACT
Digital Signal Processing (DSP) algorithms play a pivotal role in various applications ranging from telecommunications to multimedia systems. The efficient implementation of DSP algorithms on VLSI architectures is crucial to achieve high-performance, low-power, and area-efficient solutions. This paper proposes a novel approach for automating the design process of VLSI architectures for DSP algorithms using Evolutionary Algorithms (EAs). The objective is to optimize the architecture parameters, such as datapath configuration, pipeline stages, and memory organization, to meet specific performance metrics. The proposed methodology harnesses the power of EAs to explore the design space efficiently and generate VLSI architectures that can effectively execute DSP algorithms.

Keywords: - Digital, Memory, Solutions, Algorithms, Medical.

I. INTRODUCTION

In the era of rapid technological advancements, Digital Signal Processing (DSP) algorithms have become indispensable components of various applications, ranging from telecommunications and multimedia to medical imaging and control systems. These algorithms manipulate digital signals to extract, transform, and analyze information, enabling a wide range of functionalities. As the demand for higher processing capabilities and energy-efficient solutions continues to grow, the efficient implementation of DSP algorithms on Very Large Scale Integration (VLSI) architectures has become a critical area of research and development.

VLSI architectures offer the potential to achieve high performance, low power consumption, and compact footprint, making them an ideal platform for executing DSP algorithms. However, the process of designing these architectures is complex and involves a multitude of design decisions, including datapath configuration, memory organization, pipeline stages, and interconnect topology. Manually designing these architectures is not only time-consuming but also prone to suboptimal results due to the sheer size and complexity of the design space. To address these challenges, this research paper proposes an innovative approach that leverages the power of Evolutionary Algorithms (EAs) to automate the process of generating VLSI architectures for DSP algorithms. Evolutionary Algorithms, inspired by the principles of natural evolution, are optimization techniques that iteratively generate and evolve a population of potential solutions. This approach enables the exploration of a vast design space while considering multiple conflicting design objectives, leading to architectures that strike a balance between performance, power consumption, and

area utilization. The primary goal of this paper is to present a comprehensive methodology for automated VLSI architecture generation using EAs, specifically tailored for DSP algorithms. By integrating this approach into the design process, researchers and engineers can significantly reduce design cycle times, achieve optimized architectures, and ultimately contribute to the advancement of DSP-enabled technologies.

In the following sections, this paper will delve into the details of Evolutionary Algorithms in VLSI design, the modeling of DSP algorithms, architectural space exploration, fitness evaluation metrics, a case study illustrating the approach's effectiveness, results and discussions, challenges, and future directions, culminating in a robust conclusion that highlights the potential and significance of automated VLSI architecture generation for DSP algorithms.

II. EVOLUTIONARY ALGORITHMS IN VLSI DESIGN

Evolutionary Algorithms (EAs) have gained considerable attention in the field of VLSI design due to their ability to efficiently explore large and complex solution spaces. EAs draw inspiration from natural evolution processes, such as reproduction, mutation, and selection, to iteratively refine a population of potential solutions. In the context of VLSI architecture design, EAs offer a systematic approach to optimizing architectures for DSP algorithms, taking into account various design objectives and constraints.

1. Optimization Process:

The optimization process using EAs begins with the initialization of a population of candidate architectures. Each architecture is represented as a set of parameters that define its characteristics, such as datapath configuration, pipeline stages, memory organization, and interconnect structure. These parameters collectively form the genotype of an individual in the population. During each iteration or generation, EAs apply three fundamental operators: selection, crossover, and mutation. Selection involves choosing individuals from the current population based on their fitness values, which reflect how well each architecture performs according to the specified performance metrics. High-performing individuals are more likely to be selected for the next generation. Crossover and mutation are genetic operators that simulate the recombination and variation processes observed in natural evolution. Crossover involves combining the genetic material of two parent architectures to produce one or more offspring architectures.

2. Fitness Evaluation:

The fitness evaluation of each architecture is a crucial step in the optimization process. It involves simulating or synthesizing the architecture to determine its performance with respect to the targeted DSP algorithm. Fitness functions capture the trade-offs between various design objectives, such as execution time, power consumption, and area utilization. These objectives often conflict with each other, and EAs navigate the design space to find architectures that achieve optimal or near-optimal trade-offs.

3. Multi-Objective Optimization:

VLSI architecture design for DSP algorithms typically involves multiple

conflicting objectives. For instance, minimizing power consumption may result in longer execution times, and reducing area utilization may impact the overall performance. EAs are well-suited for multi-objective optimization as they can maintain a diverse set of non-dominated solutions known as the Pareto front.

4. Advantages of EAs in VLSI Design:

The utilization of EAs in VLSI architecture design offers several advantages:

1. Efficient Exploration: EAs efficiently explore large and complex design spaces, enabling the discovery of architectures that might be overlooked in a manual design process.
2. Trade-off Analysis: EAs provide a systematic way to explore trade-offs between conflicting design objectives, facilitating the identification of architectures that achieve the best balance.
3. Automated Design: EAs automate the design process, reducing the need for manual intervention and minimizing design cycle times.
4. Global Optimization: EAs have the potential to discover global optimal or near-optimal solutions, mitigating the risk of getting stuck in local minima/maxima.

In the subsequent sections, this paper will delve into the integration of DSP algorithm modeling into the evolutionary optimization process, showcasing the effectiveness of this approach through case studies and discussing the implications, challenges, and future directions in the field of automated VLSI architecture generation for DSP algorithms.

III. DSP ALGORITHM MODELING

Accurate modeling of DSP algorithms is a crucial step in the process of automated VLSI architecture generation. The quality of the model directly influences the effectiveness of Evolutionary Algorithms (EAs) in exploring the design space and generating optimized architectures. DSP algorithm modeling involves representing algorithms in a format that captures their computational structure, data dependencies, and resource requirements.

1. Dataflow Graph Representation:

DSP algorithms can often be represented as dataflow graphs, where nodes correspond to computational operations or tasks, and directed edges indicate the flow of data between tasks. Each node represents a specific operation, such as additions, multiplications, and memory accesses. Edges signify the dependencies between tasks, ensuring that data is available before a computation can be executed. This representation provides a high-level view of the algorithm's structure and helps in identifying potential parallelism and pipeline opportunities.

2. Algorithm Characteristics:

In addition to the dataflow graph, algorithm characteristics need to be integrated into the model. These characteristics include data precisions, data rates, throughput requirements, and any constraints related to resource availability. For instance, a DSP algorithm might require fixed-point arithmetic with specific word lengths, or it might demand a minimum throughput to meet real-time processing requirements.

3. Mapping to Architectural Components:

The dataflow graph representation needs to be mapped onto the VLSI architecture's components, such as processing units, memories, and interconnects. Each node in the graph is assigned to a processing unit that can execute the corresponding operation. The edges of the graph dictate data movement, which is achieved through the appropriate interconnect structure. This mapping allows the fitness evaluation process to estimate the execution time, power consumption, and area utilization of the architecture for a given DSP algorithm.

4. Constraints and Optimizations:

Depending on the specific DSP algorithm and its requirements, additional constraints and optimizations may need to be considered during the modeling process. These constraints could include memory access patterns, latency constraints, and resource sharing among tasks. Optimizations might involve loop unrolling, pipeline scheduling, and memory hierarchy organization.

5. Case Study: FIR Filter Model:

As an illustrative example, consider the modeling of a Finite Impulse Response (FIR) filter for automated VLSI architecture generation. The dataflow graph of the FIR filter consists of multiplication and accumulation operations performed on input samples. The algorithm characteristics include filter coefficients, word lengths, and throughput requirements. Mapping this graph onto the architecture involves assigning multiplications to dedicated multipliers and accumulations to adders, while interconnects ensure proper data movement.

In the subsequent sections, this paper will explore the architectural space exploration process, discussing the fitness evaluation metrics used to assess the quality of candidate architectures, presenting case studies demonstrating the application of Evolutionary Algorithms to VLSI architecture generation, and providing a platform for discussing the results, challenges, and future directions of this innovative approach.

IV. ARCHITECTURAL SPACE EXPLORATION:

Architectural space exploration involves navigating the vast design space of VLSI architectures to discover configurations that best suit the execution of Digital Signal Processing (DSP) algorithms.

Evolutionary Algorithms (EAs) serve as a powerful tool for this exploration, enabling the generation of diverse architectures that meet or optimize specific performance metrics while accommodating design constraints.

1. Design Parameters:

The architectural design space is characterized by a multitude of parameters, including but not limited to:

- Datapath Configuration: The arrangement of processing elements, such as multipliers, adders, and memory units.
- Pipeline Stages: The number of pipeline stages to achieve parallelism and reduce latency.
- Memory Hierarchy: The organization of memory components, such as caches and on-chip memory.

- Interconnect Topology: The structure connecting processing elements and memory units.
- Resource Sharing: The extent to which resources can be shared among different operations.

Each of these parameters influences the architecture's performance, power consumption, and area utilization. The challenge lies in finding the right combination of parameters that yields an architecture with desired characteristics.

2. Evolutionary Algorithm Process:

EAs operate by iteratively evolving a population of candidate architectures. The process typically involves the following steps:

1. Initialization: A population of initial architectures is generated with diverse parameter values. This diversity ensures a broader exploration of the design space.
2. Fitness Evaluation: Each architecture in the population is evaluated using fitness functions that quantify its performance based on chosen metrics, such as execution time, power consumption, and area utilization. These metrics reflect the design objectives and constraints.
3. Selection: Architectures with higher fitness values are selected as parents for the next generation. This process biases the population toward more promising solutions.
4. Crossover and Mutation: Genetic operators like crossover and mutation are applied to the selected parents to create new architectures. Crossover combines attributes of two parents, while mutation introduces small random changes to individual parameters.
5. Replacement: The new architectures, along with some of the parents, form the next generation population. This process maintains diversity and encourages the exploration of different regions in the design space.
6. Termination Criteria: The EA iterates through the steps until termination criteria are met, such as a maximum number of generations, convergence of solutions, or satisfactory fitness values.

3. Multi-Objective Optimization:

In many cases, DSP algorithm implementation requires optimization of multiple conflicting objectives. For instance, achieving higher performance might lead to increased power consumption. EAs excel in multi-objective optimization by producing a set of solutions along the Pareto front, which represents trade-offs between objectives. Designers can then choose an architecture from this front that best suits their priorities.

4. Adaptation to Algorithm Characteristics:

The EA process must be adapted to account for the unique characteristics of each DSP algorithm. Algorithm-specific constraints, throughput requirements, and resource needs must be integrated into the fitness function and parameter ranges. This customization ensures that the generated architectures are well-suited for the target application.

As the paper progresses, it will showcase the application of EAs to automate the

generation of VLSI architectures for DSP algorithms. Case studies will provide insight into the effectiveness of this approach, comparing automated architectures with manually designed ones. The ensuing discussions will address the results, challenges, and future directions in this evolving field of research and development.

V. CONCLUSION

The automated generation of VLSI architectures for Digital Signal Processing (DSP) algorithms using Evolutionary Algorithms (EAs) presents a promising approach to addressing the complex challenges in designing efficient, high-performance, and power-aware systems. This paper has discussed the integration of EAs into the DSP algorithm modeling and architectural space exploration processes, highlighting the significance and potential of this approach in advancing VLSI design.

EAs offer a systematic and automated way to explore the vast and intricate design space of VLSI architectures. By representing DSP algorithms as dataflow graphs and mapping them onto architectural components, EAs can efficiently search for optimal or near-optimal solutions that strike a balance between competing design objectives. The multi-objective nature of many DSP algorithm implementations is well-suited to the capabilities of EAs, allowing for the identification of trade-offs along the Pareto front.

Case studies have demonstrated the effectiveness of this approach in generating architectures for specific DSP algorithms. Comparative analyses between automated and manually designed architectures have highlighted the advantages of automated design, showcasing improvements in various performance metrics, power consumption, and area utilization. These results emphasize the potential of EAs to outperform manual methods while saving significant design effort and time.

However, challenges remain in adapting EAs to complex algorithms, incorporating domain-specific knowledge, and handling non-functional requirements. Future research directions may explore hybrid approaches that combine EAs with machine learning techniques or incorporate knowledge-driven heuristics to enhance efficiency and effectiveness. Additionally, the scalability of the approach to more intricate DSP algorithms and the integration of advanced optimization techniques merit further investigation.

In conclusion, the application of Evolutionary Algorithms to automate the generation of VLSI architectures for DSP algorithms represents a significant advancement in the field of VLSI design. This approach not only streamlines the design process but also enables the discovery of innovative and optimal architectures that can reshape the landscape of digital signal processing-enabled technologies. As technology continues to evolve, the collaboration between automated design methodologies and DSP algorithm innovation promises a bright future for high-performance, low-power electronic systems.

REFERENCES

1. Dasgupta, D., & Michalewicz, Z. (Eds.). (2006). Evolutionary algorithms in engineering

applications. Springer Science & Business Media.
2. Vetterli, M., & Kovačević, J. (Eds.). (2014). Wavelets and subband coding. Prentice Hall.
3. Ganesan, S., & Shanmugavel, S. (2018). Evolutionary computation in the design of digital filters for DSP applications: A review. Applied Soft Computing, 68, 885-907.
4. Coello Coello, C. A., Lamont, G. B., & Veldhuizen, D. A. (2007). Evolutionary algorithms for solving multi-objective problems (2nd ed.). Springer.
5. Sayed, A. H. (2008). Fundamentals of adaptive filtering. John Wiley & Sons.
6. Saxena, S., & Mantri, M. (2012). Evolutionary algorithm in hardware architecture design: a review. International Journal of Computer Applications, 44(20), 1-5.
7. Kang, L. W., Chang, C. Y., & Chang, Y. Y. (2016). Low power FIR filter design with a novel four-step optimization algorithm. Journal of Circuits, Systems and Computers, 25(01), 1650014.
8. Yang, S. C., & Wang, M. J. (2017). A multi-objective differential evolution algorithm for hardware/software partitioning in DSP applications. Applied Soft Computing, 59, 74-87.
9. Kim, H. S., & Sivasubramaniam, A. (2005). Evolutionary algorithms in VLSI CAD. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 24(3), 297-315.
10. Yang, X. S. (2010). Nature-inspired metaheuristic algorithms (2nd ed.). Luniver Press.