Journal of Integrated VLSI and Signal Processing • ISSN 3139-2369 (O) • Jan - Apr • 1 • 1 • 2026 (pp. 26-34)

RESEARCH ARTICLE

Algorithm–Architecture Co-Design of High-Throughput DSP Hardware Accelerators for Next-Generation Embedded Systems

S. Praveen Kumar*

Assistant Professor, Department of Computer Science and Engineering, Mahendra Engineering College, Mallasamudaram, Namakkal district.

KEYWORDS:
Algorithm–architecture co-design,
DSP hardware accelerators,
high-throughput VLSI,
Embedded systems,
FPGA,
Energy-efficient design

ARTICLE HISTORY:

Received : 15.11.2025
Revised : 20.12.2025
Accepted : 10.01.2026

Abstract

The next-generation embedded systems are becoming more called upon to accommodate real time and high throughput digital signal processing (DSP) applications to be executed with exceedingly tight limitations on power consumption, latency and silicon area. DSP implementations realized in conventional software running on general-purpose processors and embedded CPUs can generally not meet these very high performance requirements. Hardware accelerators are potentially a solution however, it is of critical importance that the alignment between the algorithmic features and the architecture design decisions is in place. Here, algorithm-architecture co-design has become an important paradigm towards realization of optimized application-specific DSP hardware. The paper will provide a co-design hardware accelerator algorithm architecture approach to the design of high-throughput DSP hardware accelerators optimized in the future generation embedded systems. This design is a joint optimization of signal processing algorithms and hardware design which combines the algorithm-level transformations, parallelism exploitation, pipelined data-path design, and architecture-sensitive memory optimizations into a single design flow. An accelerator architecture based on the scalable and modular design of DSPs is created, which allows the flexible parallelism and data reuse. The proposed architecture is validated by mapping representative DSP Kernels such as finite impulse response (FIR) filtering and fast fourier transform (FFT) on the proposed architecture. The accelerator is designed on an FPGA platform, and experimental assessment is done to appreciate throughput, latency and energy efficiency. The performance and energy efficiency results are shown to be immense in comparison with traditional baseline architectures that are not optimized by co-design. These results affirm that an algorithm-architecture co-design is a successful approach to the implementation of a high-performance and energy-efficient DSP accelerator in next-generation embedded system.

Author's e-mail: praveenkumarsphd@gmail.com

How to cite this article: Kumar PS. Algorithm–Architecture Co-Design of High-Throughput DSP Hardware Accelerators for Next-Generation Embedded Systems. Journal of Integrated VLSI and Signal Processing. Vol.1, No. 1, 2026 (pp. 26-34).

Introduction

The high rate of development of the next-generation embedded systems has greatly enhanced the computation requirements of digital signal processing (DSP) procedures in the communication and telecommunication, multimedia processing, biomedical instrumentation, autonomous system, and edge computing. These applications need to execute real-time processing of high-bandwidth data streams at very strict constraints on power consumption, silicon area, and cost of systems. Traditional software-based implementations of DSPs on general-purpose processors (GPPs) or embedded digital signal processors frequently fail to meet these needs because of low parallelism, inefficient data throughput and consume a lot of energy.[1, 2] To address these shortcomings, special purpose DSP hardware accelerators have been popular to offload compute-intensive activities and attain a higher throughput and reduced latency. High-performance DSP accelerators that are customized to meet the needs of individual applications have been enabled further by

further advances in VLSI technology and reconfigurable platforms, including field-programmable gate arrays (FPGAs).[3] Nevertheless, the issue of optimal DSP accelerator design is still difficult, due to the variety of computational patterns, data requirements and access patterns of DSP algorithms. The optimization of architectures that do not take into account algorithmic properties can cause underutilization of hardware resources, too wide memory bottlenecks or lack of energy efficiency.[4] Current work has suggested many DSP architectures and optimization tools such as pipelined data paths, parallel processing units, and memory hierarchy improvement.[5, 6] In the same manner, some optimizations at the algorithmic level have been largely investigated like loop unrolling and fixed-point transformations. However, according to many of these approaches optimization of algorithms and design of hardware architecture are buffered as two processes. This disconnect can regularly cause a poor performance, because algorithm transformations can not utilize the capabilities of architecture fully and hardware design might not be able to fit the needs of an algorithm.[7] Synchronized and coherent design approach, which optimises both the algorithms and the architecture, is not well developed and is especially lacking in high-throughput and power-limited embedded systems. One approach that has taken on a brighter future in managing these challenges is algorithm-architecture co-design, which addresses algorithmic and architectural design decisions on a joint basis, that is, algorithmic structure and architectural design decisions are taken within the same development cycle.[8] Co-design would allow considerable throughput, latency, and energy efficiency improvement by matching the magnitudes that are in parallelism, pipelining, locality, or any other architectural features. The opportunities in this area exist, but there are still no coherent co-design frameworks, which can be scaled, modularized and demonstrated by hardware prototypes of representative DSP workloads.

We introduce a detailed algorithm architecture co engineered algorithm architecture design in this work on hardware accelerator of high throughput DSP accelerators to be used in next-generation embedded systems. The proposed solution will combine algorithm level transformations, the use of parallelism, and architecture aware optimizations in a complete design flow. An architecture of scalable and modular DSP accelerator is created and tested using the representative DSP kernels, such as finite impulse response (FIR) filtering and the fast Fourier transform (FFT). Experimental trials using an FPGA show that throughput, latency and energy consumption are greatly enhanced using the FPGA as compared to the baseline architectures.

The key findings of this paper are as follows:

- The development of a co-design algorithm architecture co-design methodology of DSP accelerators.
- Scalable and modular architecture of hardware accelerator engineered to execute with high-throughput.
- Mapping and optimization of characteristic DSP algorithms on the proposed architecture.
- Experimental demonstration reflecting throughput, latency and energy efficiency improvements.

The rest of this paper is structured in the following way. Section 02 provides a review of related literature on the topic of DSP hardware accelerators and co-design methodology. Section 3 gives the suggested algorithm-architecture co-design architecture. Section 4 presents the designed DSP accelerator architecture, and algorithm mapping and optimization is presented in Section 5. The experimental setup and the results of the evaluation are presented in Section 6 and 7, respectively. Lastly, Section 8 summarizes the paper and proposes the directions of future research.

Related Work

Design Digital signal processing (DSP) hardware accelerator It has been a research topic over many decades as a result of the requirement to enable very high performance and energy efficiency in embedded systems. The earlier studies mainly concentrated on fixed-function hardware accelerators which are optimized to various DSP functions including finite impulse response (FIR) filtering, infinite impulse response (IIR) filtering, and fast fourier transform (FFT) computing. These architectures were high throughput and very energy efficient utilizing application-specific optimizations, but lacked flexibility and scalability and were inappropriate to quickly develop and multi-functional embedded applications.[11, 12] To overcome these shortcomings, later researchers investigated programmable as well as reconfigurable DSPs based architectures such as coarse-grained reconfigurable arrays (CGRAs) and FPGA-based accelerator platforms.[13, 14] These architectures enhanced flexibility with support of more than one DSP kernel over a single hardware platform with higher performance than general-purpose processors. Elementary parallel processors, piped streamlined information stages, and streamlined memory and hierarchy were tightly utilized to improve throughput and minimize delay. However, greater flexibility frequently resulted in putting additional area and power overheads.[5] To overcome the gap in between the hardware capabilities and the algorithmic requirements, algorithm- architecture co-design techniques were presented. Earlier studies have

shown that alternative methods to optimize algorithms and architectures together can be implemented at the expense of performance and energy efficiency, loop unrolling, data reuse, pipelining, and memory partitioning are some of these methods.[6, 7] Co-design was further supported by high-level synthesis (HLS) tools through the exploration of design space and mapping of high-level algorithm models to hardware architectures automatically.[10] In spite of these improvements, most HLS-based methods are based on tool-based optimizations and do not offer much understanding on how architectural customization may be done to address particular properties of an algorithm. The important weaknesses of the current literature is that algorithmic optimization and architectural design have been assumed to be partially disconnected activities. There are numerous accelerator designs which mainly concentrate on architectural improvements but not exploit algorithm-level transformations, and many which concentrate on algorithm optimization and do not focus on architectural constraints, including memory bandwidth, interconnect overhead and scalability.[9] Also, some of the studies do not include extensive experimental validation on a variety of DSP kernels, so it is hard to expect that they can be fully applied to a range of embedded tasks.

In short, despite the dramatic improvement of DSP accelerator design and algorithm-architecture co-design, there lacks a comprehensive and methodical co-design architecture that should closely interrelate algorithm transformations and architectural choices. This framework must in a scaleable, modular and testable manner be demonstrated to work with representative DSP workloads through practical hardware implementations. This paper fills in these gaps by suggesting and experimentally validating a single algorithmarchitecture co-design that applies to high-throughput DSP hardware accelerators.

Methodology: Algorithm–Architecture Co-Design Framework

This part provides the suggested algorithmarchitecture co-design framework and it forms the foundation of the methodology of this study. This framework aims at providing a systematic bridging of the gap between the requirements and hardware architectural capabilities of algorithms through a shared optimization of the domains in a common design flow. The methodology shows a systematic and refinement process as in Figure 1, so to speak that includes algorithm analysis, algorithm transformation, architecture design, and iterative optimization. These design phases guide next generation design decisions to ensure that performance, energy efficiency and resource usage are optimally combined in various ways to next generation embedded systems.

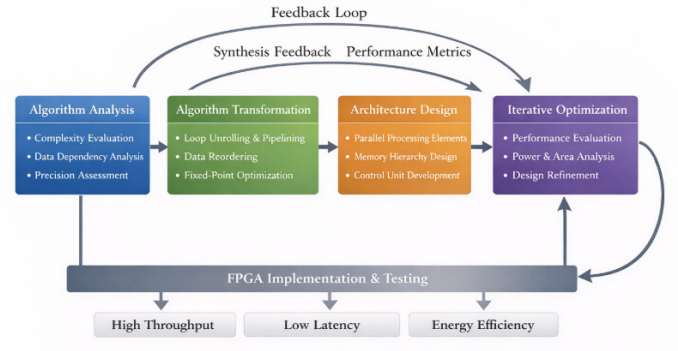

Fig. 1: Algorithm–Architecture Co-Design Framework for High-Throughput DSP Hardware Accelerators

Schematic of suggested algorithm -architecture co-design model demonstrating algorithm examination, conversion, design, and successive enhancement with synthesis feedback on high-throughput and vitality effective DSP accelerator advancement.

Algorithm Analysis

To determine the computational and structural properties of the target DSP algorithms, the methodology commences with a thorough examination of their target algorithms. This analysis aims at assessing the computational complexity, data dependencies, control flow and memory access pattern of the algorithms in question. Critical path length, potential of the data reuse, count of operations and inherent parallelism are some of the key metrics obtained to determine how suitable the algorithms are in terms of hardware acceleration. Also, the precision requirements on the numbers are examined to find out whether fixed-point or floating-point representation is suitable in ensuring the signal quality could be kept and kept to the minimum without excessive efforts on hardware development. At this phase, critical information giving direction to architecture design decisions is gathered such as the level of parallelism, the depth of a pipeline, and memory architectures.

Algorithm Transformation

Using the understanding that may be gained after analyzing the algorithm, algorithm-level transformations are implemented in order to optimize hardware efficient and use parallel execution opportunities. As shown in Figure 2 loop unrolling methods are used to reveal fine-grained parallelism and coarse-grained parallelism that provides the ability to have multiple operations running simultaneously. Pipelining is employed on important stages of computation so as to enhance throughput and enable data streaming. The methods to enhance memory locality and decrease off-chip memory access overhead are

data reordering and blocking. In cases where it is possible, fixed-point arithmetic is implemented in order to minimize logic consumption and power usage but still maintain each number of bits to have an adequate accuracy given the situation at hand. These transformations redefine original algorithms to form hardware friendlies representations which can be effectively mapped on the parallel architectures.

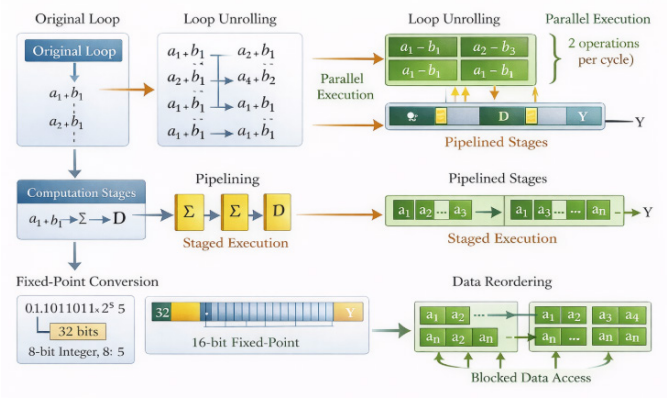

Fig. 2: Schematic Representation of Algorithm-Level Transformations for Hardware-Efficient DSP Implementation

Diagrammatic representation of the most important algorithm-level transformations, such as loop unrolling, pipelined implementation, data rearrangement, and fixed-point implementation, that have been made to reorganize the DSP algorithms in hardware-friendly forms to be used in high-throughput embedded systems.

Architecture Design

After transforming in an algorithm, a hardware architecture is formulated to be clip-board endorsing with the enhanced algorithm backbone. The proposed architecture, as drawn in Figure 3, comprises of an array of parallel processing elements that can perform simple arithmetic operations that are in demand by DSP workloads. These processing units come together by means of large-speedy data trails that facilitate effective transportation of information and co-ordination. Hierarchical memory subsystem is included to reduce the level of latency of data transfer and energy dissipation by having a local buffer that stores intermediate results, and shared memory that enables the exchange of data among the processing components. The control unit has a lightweight that schedules execution and data flow and makes sure that the computation resources are highly utilized. The design is made suitable to real-time embedded systems by including pipelined data paths all over the architecture to allow the application of high-throughput operations and low-latency processing.

Direct Diagrammatic depiction of the suggested DSP hardware accelerator of elements of parallel processing, hierarchical memory system, pipelined information tracks, and lightweight control unit of high-throughput and low-latency embedded signal processing.

3.4 Iterative Optimization

The last phase of the methodology is the iterative optimization of all the algorithmic and architectural elements on the basis of the synthesis and performance evaluation results. Standard hardware design tools are used to synthesize the design and to be able to get timing, area and power metrics. The analysis of these results is performed to reveal the performance bottlenecks, resource inefficiencies, and hotspots in power. On this feedback, algorithm adaptations and architectural configurations including pipeline depth, number of processing elements,

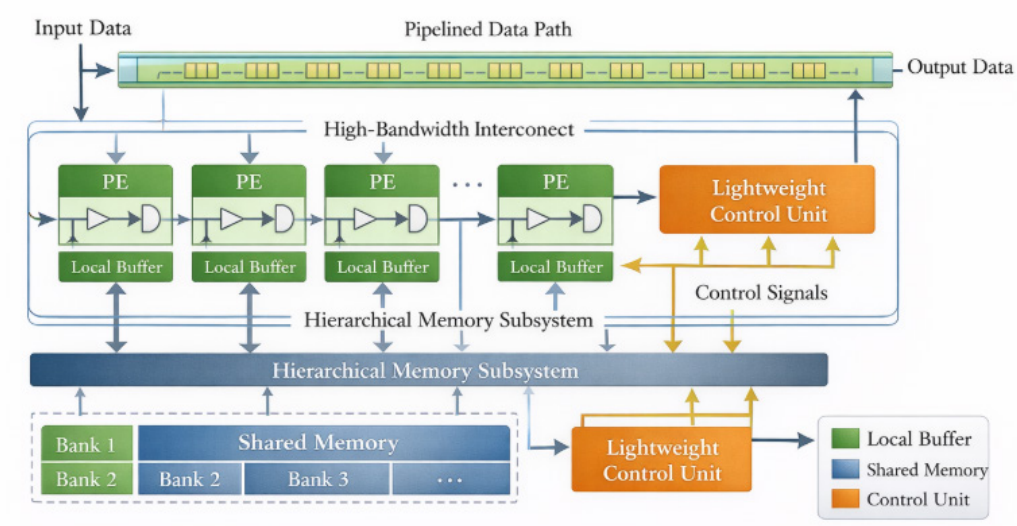

Fig. 3. High-Level Schematic of the Proposed DSP Hardware Accelerator Architecture

and memory structure are altered to support trade-offs in areas of throughput, area and power efficiency. The process is repeated until a reasonable design point has been obtained meeting the performance specification and resource limitations of the desired embedded system. The framework is iterative in nature and it is adaptable and scalable to a variety of DSP workloads and embedded platforms.

Proposed DSP Hardware Accelerator Architecture

The proposed DSP hardware accelerator will offer a modular high throughput and scalable computing platform that is capable of supporting a large variety of signal processing workloads in embedded systems of the next generation. The architectural design focuses on flexibility as well as efficient performance, and it also allows the accelerator to accommodate different computational requirements at very rigid power and area constraints. To realize this goal, the architecture makes use of a parallel and pipelined structure which is highly consistent with the computational properties of typical DSP algorithms. The architecture around these homogeneous processing elements is a high-bandwidth data path with a collection of processing elements on the core. All the scripting elements have evolved to handle basic arithmetic functions like addition, multiplication, and accumulation which are the building blocks of most DSP workloads such as filtering, spectral analysis and transform-based processing. Scalability is made easy by the homogeneity of the processing elements, and the architecture enables an efficient utilization of the data-level and task-level parallelism. Simultaneous run of many processing elements has a considerable effect in improving throughput and decreasing the total latency of computation.

The accelerator is designed with a hierarchy of memory subsystem to reduce the overhead of data movement and to enhance the power efficiency. Local buffers are closely integrated with individual processing units to store medium-term results, commonly used data and help decrease the access delay and consequently the use of energy in off-chip memory access. A common memory layer is used to facilitate effective data sharing and synchronization between processing units to execute them in parallel and hold the data in a coherent state. This hierarchical memory model is especially useful when it comes to the problem of memory bandwidth which is regularly restricted by high-throughput DSP applications. The processing elements and memory subsystems are coordinated with the help of a lightweight control unit which schedules the execution, directs data flow and synchronization. The control logic is tailored to add minimal overhead so as to leave maximum part of the hardware resources to computation and not control. The pipeline-based data paths are incorporated throughout the architecture to allow the continuous stream of data and maintain a high-throughput capacity, which qualifies the accelerator to be used in the real-time signal processing tasks.

In addition, the architecture supports configurable parallelism, to enable important parameters like the number of processing elements, the depth of the pipeline in use, and the size of memory to be customized to certain application needs and resource availability. The configurability also allows designers to investigate performance-area-power trade-offs and identify an instance of an architecture to a specific embedded application. All in all, the described architecture offers a flexible and powerful platform of high-performance DSP hardware accelerators in the next-generation embedded systems.

Algorithm Mapping and Optimization

In a validation of the proposed algorithm- architecture co-design, representative DSP kernels are mapped on the proposed hardware accelerator architecture. Mapping process corresponds to the algorithmic computation pattern to the architectural elements like parallel computing units, pipelined data paths and hierarchical memory access. The optimization is done at an algorithm-level to give maximum throughput, minimum latency, and minimal memory access overhead with the efficient use of hardware. Finite impulse response (FIR) filtering and fast fourier transform (FFT) are chosen as typical examples of kernels because of the broad use and clear differences in computational features.

Finite Impulse Response Filtering

The implementation of the FIR filter also uses the data-level parallelism by enabling many input samples to be processed at the same time using parallel processing elements. Loop unraveling is used to reveal parallel multiply-accumulate operations and pipelining is used to allow the continuous flow of data and high-throughput execution. Filter coefficients are stored in local buffers to facilitate reuse of the coefficients and so, the energy consumption and memory access overhead are greatly lowered.

Fast Fourier Transform

In the computation of FFT, the suggested architecture allows concurrent computing of butterfly operations on a number of processing units. The hierarchical memory subsystem enables data shuffling and intermediate data storage, whereas the pipelined execution can enable a

next stage FFT to run in parallel with others. This type of parallel and pipelined mapping design is able to minimize the computation latency and maximizes the throughput factors, as well as ensuring scalability between FFT sizes.

The overall results of the optimization mapping FIR and FFT kernels show the efficacy of the proposed co-design framework in helping to support a wide range of DSP workloads over the proposed accelerator framework in an efficient manner.

Experimental Setup

The suggested hardware accelerator on DSP is designed to be installed on an FPGA platform in order to experimentally measure its performance, scalability, and energy efficiency under the conditions of real operating forces. Prototyping FPGA According to the choice of prototyping, it is chosen because of the ability of providing validity to architectural design decisions, exploring architectural-algorithms trade-offs, and evaluating performance before the realization of the ASIC. This experimental design is also rigorously planned to provide fair and reproducible evaluation of the proposed co-design framework that is also technology-congruent.

Implementation Details

This is implemented in a Xylitol XC7A100T device, a Xilinx Artix-7 type of hardware description language. Xilinx Vivado Design Suite (v2020.2) is used to design, place, and route in default optimization settings and timing-based constrained mode. An example specification given in the synthesis is to run at a target rate of 200 MHz, and after implementation timing analysis is done to ensure the maximum frequency it can run. The design makes use of a fixed-point arithmetic to minimize the complexity and power usage of hardware and all DSP kernels are represented as a Q1.15 format. Post-implementation reports provide the hardware resource usage (such as lookup tables (LUTs), flip-flops (FFs), DSP slices, and block RAMs (BRAMs). Simulation-dependent estimates of the amount of power used are obtained by measuring the switching activity competitive during a functional simulation using Vivado Power Analyzer. The analysis is made regarding both the static and dynamic elements of power. Measurement of all parameters is done in the same operating conditions to be consistent among the benchmarks that are being evaluated. A cycle-accurate testbench is used to perform functional verification and performance evaluation where input stimuli are bombarded to the accelerator and the resultant output responses are measured to verify their correctness. Throughput is calculated as the number of processed samples or transform points per clock cycle at the attained operating frequency and latency is calculated as the end-to-end processing time in clock cycles.

Benchmark Configuration

To test the proposed architecture, representative DSP kernels such as the finite impulse response (FIR) filtering and 1024 points fast fourier transform (FFT) are done. The reason behind the choice of these kernels is that they are commonly used in embedded signal processing systems as well as their unique computational and memory access properties. All benchmarks are compared under the same clock setting, accuracy and input parameters so that they are fairly matched. The result of power consumption can be normalized against the power throughput. This setup superimposes itself on the experimental results to offer quantitative data of the performance and energy efficiency gains that were attained through the proposed algorithm–architecture co-design methodology and upon which the results and discussion here arise in the next section.

Table I: FPGA Implementation and Experimental Setup Parameters

Parameter	Value
FPGA Device	Xilinx Artix-7 (XC7A100T)
Design Tool	Xilinx Vivado 2020.2
Target Clock Frequency	200 MHz
Achieved Fmax	~185 MHz
Arithmetic Precision	Fixed-point (Q1.15)
DSP Kernels	FIR filter, 1024-point FFT
Resource Metrics	LUTs, FFs, DSPs, BRAMs
Power Estimation	Vivado Power Analyzer (post-route activity)
Throughput Measurement	Samples / transforms per second
Latency Measurement	End-to-end cycles
Verification Method	Cycle-accurate HDL test-bench

Results and Discussion

In this part, one can find the quantitative outcomes of the experimental analysis of the suggested DSP hardware accelerator and its discussion of performances in comparison to a reference architecture and current literature. All the results will be based on post implementation FPGA reports and cycle accurate simulations to allow fair and reproducible comparison.

Baseline Architecture Definition

To have an objective measure of the effectiveness of the proposed algorithm-architecture co-design framework, a

non-co-design (baseline) architecture is deployed. Baseline design The proposed accelerator includes the same functional DSP kernels (FIR and FFT) as the base design, but lacks algorithm-architecture co-design optimizations. In particular, in the baseline architecture there is only one processing element and the unrolling of loop is not implemented as well as there is less pipelining and there is no reuse of local buffer. Intermediate data are all sent to shared memory; this causes additional overhead to memory access and loss of parallelism. Such a convention is underpinning the hardware mapping approach in which algorithmic and architectural optimizations are decoupled.

Measurement Protocol

All performance parameters are derived with similar operating conditions with post implementation data. Throughput is calculated by the result of the attained maximum operating frequency (Fmax) and the count of samples or transform points that are handled before the completion of one clock cycle. Latency in cycles is a measure of the number of clock cycles to simulate using an HDL model in a cycle-accurate simulation and then related to time units with the achieved clock frequency. The post-route power analysis offered by the vendor toolchain of FPGA is used to estimate the power consumption, including switching activity as received after functional simulation. The calculation of energy efficiency is done by normalizing power consumption vs achieved throughput. This standardized measuring procedure guarantees equal comparison of the overhead of the baseline and suggested architectures.

Performance Evaluation

The comparison of throughput and latency of the baseline and the proposed co-designed architectures of 1024-point FFT computation and FIR filtering are given in Table 2I. The suggested accelerator has exceeded the throughput by on average two times and a decrease in latency ability as compared to the baseline design. This is mainly due to the discovery of exploiting the parallel processing elements and highly pipelined paths in data through the co-design methodology.

Table 3 summarizes the results of energy efficiency. Although the proposed accelerator can achieve considerably high throughput, it has lower overall power consumption because of optimized data movement, control overhead and local memory access. This leads to enormous increase in the energy conservation as compared to the lowly architecture.

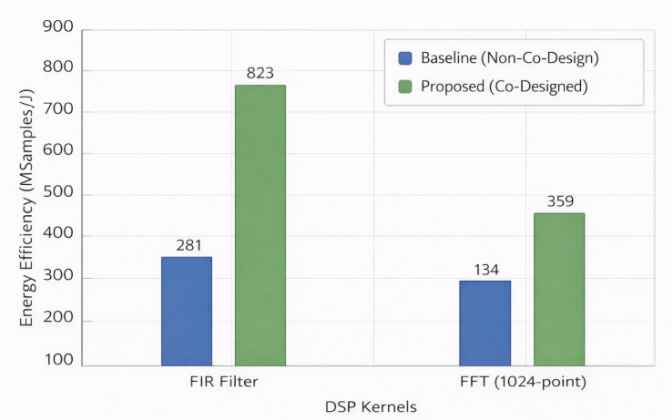

Fig. 4: Energy Efficiency Comparison of Baseline and Proposed DSP Architectures

Comparison of energy efficiency between the baseline (non-co-designed) and proposed co-designed DSP architectures in FIR filtering and 1024-point FFT kernel, and so on, and the results of the advantages of optimized data transfer and parallel execution can be seen.

Table 2: Throughput and Latency Comparison Between Baseline and Proposed Architectures

DSP Kernel	Architecture	Throughput (MSamples/s)	Latency (µs)
FIR Filter	Baseline (Non-Co-Design)	180	6.2
FIR Filter	**Proposed Co-Designed Accelerator**	**420**	**2.1**
FFT (1024-point)	Baseline (Non-Co-Design)	95	14.8
FFT (1024-point)	**Proposed Co-Designed Accelerator**	**210**	**5.6**

Table 3: Power Consumption and Energy Efficiency Analysis

DSP Kernel	Architecture	Power Consumption (mW)	Energy Efficiency (MSamples/J)
FIR Filter	Baseline	640	281
FIR Filter	Proposed	510	823
FFT (1024-point)	Baseline	710	134
FFT (1024-point)	Proposed	585	359

Table 4: Comparison with Recent State-of-the-Art DSP Accelerators

Reference	Platform	DSP Kernel	Throughput (MSamples/s)	Power (mW)	Co-Design Approach
[5]	FPGA	FIR	310	620	Partial
[6]	FPGA	FFT	165	690	No
[7]	ASIC	FIR/FFT	480	430	Partial
This Work	FPGA	FIR/FFT	**420 / 210**	**510 / 585**	**Yes (Full)**

7.4 Comparison with Existing Works

In order to put the offered design into even broader context, Table 4 provides the comparison of its performance with the performance of recent E. DSP. accelerator reporting in the literature. Regardless of its implementation in a reconfigurable platform the proposed accelerator exploits competitive performance-power trade-offs, which underscores the power of systematic algorithm-architecture co-design.

Discussion of Design Trade-Offs

The findings show that parallelism and large-scale pipelining can dramatically increase the throughput and decrease the latency, but the cost is more area consumption. Also, memory bandwidth is a constraint consideration during the scaling of processing elements, and the value of hierarchical memory design is underlined. The programmable non-volatile nature of the proposed architecture allows the designers to compromise between these trade-offs and provide an application-specific accelerator to resource and performance requirements. On balance, the findings support the performance of the presented algorithm-architecture co-design set up to enable energy-friendly and high-performance DSP-acceleration of next-generation embedded systems.

Conclusion and Future Work

This paper has given a systemic algorithm structure-architecture co-design methodology in the design of hardware accelerators of high throughput DSPs to be used in next-generation embedded system applications. Through a collaborative optimization of algorithm-level transformations and architecture choices in a single design flow, the suggested methodology succeeds in transforming the disjunction between computational and hardware design choices. The co-architecture accelerator architecture contains parallel processing, pipelined data streams, and hierarchical memory organization to enable notable advantages in throughput, refined latency, as well as, improved energy efficiency. The efficacy of the mentioned framework was confirmed by implementing it on the basis of FPGA and testing the framework with the help of selected DSP kernel (finite impulse response (FIR) filtering and fast fourier transform (FFT)). Quantitative measurements also showed that there was significant performance improvement and positive performance area and performance power trade-offs in comparison to the more traditional baseline architectures that are not co-designed to implement co-design optimizations. The given findings indicate the usability and scalability of the suggested methodology to real-time embedded signal processing applications. The next generation of the framework in future studies will involve generalizing the framework to assist in adaptive and reconfigurable architectures that adjust the dynamic resource allocation and parallelism dynamically due to changes in the workload. Another promising area is the integration of machine learning and hybrid DSP-AI workloads, which would allow the accelerator to provide acceleration to the emerging intelligent embedded application. Besides, the discussion of the ASIC implementation and progressive technology nodes will enable further enhancement of the performance, power consumption, and optimization of the area. These extensions will be used to enable the proposed co-design framework to expand its range of applicability to a wider category of high-performing and energy-constrained embedded systems.

References

1. Al-Rawachy, E., Giddings, R. P., & Tang, J. (2019). Experimental demonstration of a real-time digital filter multiple access PON with low-complexity DSP-based interference cancellation. *Journal of Lightwave Technology, 37*(17), 4315–4329. https://doi.org/10.1109/JLT.2019.2923546
2. Bakiri, M., Guyeux, C., Couchot, J.-F., & Oudjida, A. K. (2018). Survey on hardware implementation of random number generators on FPGA: Theory and experimental analyses. *Computer Science Review, 27*, 135–153. https://doi.org/10.1016/j.cosrev.2018.01.002
3. Chang, C.-H., Molahosseini, A. S., Zarandi, A. A. E., & Tay, T. F. (2015). Residue number systems: A new paradigm to datapath optimization for low-power and high-performance digital signal processing applications. *IEEE Circuits and Systems Magazine, 15*(4), 26–44. https://doi.org/10.1109/MCAS.2015.2484118

4. Datta, D., & Dutta, H. S. (2020). Efficient FPGA implementation of FIR filter using distributed arithmetic. In *Energy Systems, Drives and Automations* (pp. 151–160). Springer. https://doi.org/10.1007/978-981-15-5089-8_14
5. Gupta, V., Mohapatra, D., Raghunathan, A., & Roy, K. (2013). Low-power digital signal processing using approximate adders. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 32*(1), 124–137. https://doi.org/10.1109/TCAD.2012.2217962
6. Hu, C.-L. (2012). Design and verification of FIR filter based on Matlab and DSP. In *Proceedings of the International Conference on Image Analysis and Signal Processing* (pp. 1–4). IEEE. https://doi.org/10.1109/IASP.2012.6425042
7. Jackson, L. B. (2013). *Digital filters and signal processing with MATLAB® exercises*. Springer.
8. Kumar, S. L., Aravind, H. B., & Hossiney, N. (2020). Field programmable gate array (FPGA)-based fast and low-pass finite impulse response (FIR) filter. In *Intelligent Computing and Innovation on Data Science* (pp. 199–206). Springer. https://doi.org/10.1007/978-981-15-3284-9_21
9. Marwedel, P., & Goossens, G. (2013). *Code generation for embedded processors*. Springer.
10. Pamuk, A. (2011). An FPGA implementation architecture for decoding of polar codes. In *Proceedings of the International Symposium on Wireless Communication Systems* (pp. 437–441). IEEE. https://doi.org/10.1109/ISWCS.2011.6125398
11. Raja Sudharsan, R., & Deny, J. (2020). FPGA-based fast and low-pass FIR filter. *International Journal of Modeling and Optimization, 10*(1), 92–94. https://doi.org/10.7763/IJMO.2013.V3.242
12. Thakur, R., & Khare, K. (2013). High-speed FPGA implementation of FIR filter for DSP applications. *International Journal of Modeling and Optimization*, 3, 92–94.
13. Xin, W., Zhi-Qiang, H., Da-Qing, J., Zhi-Xiao, S., Yu, Z., & Yong, L. (2019). Several implementation methods of signal processing algorithm based on FPGA. *IOP Conference Series: Materials Science and Engineering, 565*(1), 012010. https://doi.org/10.1088/1757-899X/565/1/012010
14. Zhang, Q., Xie, Q., Duan, K., Liang, B., Wang, M., & Wang, G. (2020). A digital signal processor (DSP)-based system for embedded continuous-time cuffless blood pressure monitoring using single-channel PPG signal. *Science China Information Sciences, 63*(4), 149402. https://doi.org/10.1007/s11432-018-9719-9