

A Fully Streaming Low-Latency FPGA Architecture for Range-Doppler SAR Processing Using Radix-2 SDF FFT

Abbas Fadavi

abbasfadavi@gmail.com

Islamic Azad University

Research Article

Keywords: Deterministic latency, FPGA, radix-2 SDF FFT, Range-Doppler algorithm (RDA), streaming architecture, synthetic aperture radar (SAR)

Posted Date: August 18th, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-10696474/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

Abstract

Conventional FPGA implementations of the Range-Doppler Algorithm (RDA) execute range and azimuth compression sequentially, requiring the complete range-compressed data frame to be stored in a centralized two-dimensional buffer before azimuth processing can begin. This frame-oriented organization introduces significant processing latency and reduces temporal efficiency by forcing range and azimuth processing to execute sequentially. This paper presents a fully streaming FPGA architecture that replaces the centralized frame buffer with distributed stage-wise delay memories embedded within radix-2 Single-Path Delay Feedback (SDF) FFT/IFFT pipelines. By exploiting complementary DIF/DIT processing, the proposed architecture enables azimuth processing to start immediately after the first N_s range-compressed samples become available, allowing range and azimuth compression to operate concurrently without explicit frame-level buffering. Unlike conventional approaches, the proposed architecture reorganizes rather than reduces the required memory, transforming it into an integral part of the streaming pipeline. FPGA implementation results demonstrate deterministic real-time operation with a sustained throughput of one complex sample per clock cycle after pipeline filling. For $N_r = 512$ and $N_s = 1024$, the first-output latency is reduced from 525,824 to 2,048 clock cycles, while the total frame processing time decreases from 1,050,112 to 526,336 clock cycles. These results demonstrate that substantial performance improvement can be achieved through streaming architectural scheduling without modifying the underlying RDA or increasing its computational complexity.

1. Introduction

Synthetic Aperture Radar (SAR) has become an indispensable remote sensing technology because of its ability to produce high-resolution images independent of daylight and weather conditions. Among the various SAR image formation algorithms, the Range-Doppler Algorithm (RDA) is one of the most widely adopted owing to its computational efficiency and compatibility with frequency-domain processing [1]. The algorithm performs range and azimuth compression using FFT-based matched filtering, making it well suited for hardware implementation on Field-Programmable Gate Arrays (FPGAs) [1], [2].

Conventional FPGA implementations of the RDA generally employ a frame-oriented processing model [3]. During range compression, all range-compressed samples are first stored in a two-dimensional frame buffer of size $N_r \times N_s$. Only after the entire frame has been generated does azimuth processing begin by reading the stored data from this buffer. Although this organization simplifies data management, it forces range and azimuth processing to execute sequentially, resulting in long end-to-end latency and leaving the azimuth processing pipeline idle while the frame buffer is being filled.

A straightforward approach to reducing this latency would be to start azimuth processing as soon as range-compressed samples become available. However, this is not possible in conventional FFT-based architectures because the input sequence required by the azimuth FFT is not available until the complete frame has been assembled and reordered. Consequently, existing FPGA implementations typically rely

on centralized frame buffers and intermediate data transposition, which delay the start of azimuth processing and prevent continuous streaming operation [3].

This paper presents a fully streaming FPGA architecture that removes this scheduling bottleneck. Instead of storing the entire range-compressed image in a centralized frame buffer, the proposed architecture distributes the required storage across the stage-wise delay memories of streaming FFT pipelines. By combining radix-2 Single-Path Delay Feedback (SDF) FFT/IFFT processors with complementary Decimation-in-Frequency (DIF) and Decimation-in-Time (DIT) processing, the proposed architecture naturally generates the data ordering required for continuous streaming computation without explicit frame-level buffering or intermediate transposition [4],[5],[6].

It should be emphasized that the proposed architecture does not reduce the total memory capacity required for SAR processing. Rather, it reorganizes memory from a centralized frame buffer into distributed delay memories embedded within the streaming FFT pipeline. Because the required synchronization memory is integrated into the processing pipeline, azimuth processing no longer waits for completion of the entire range-compressed frame. Instead, it starts immediately after the first N_s range-compressed samples become available, allowing range and azimuth compression to overlap in time. Consequently, the processor achieves deterministic low-latency streaming operation while sustaining a throughput of one complex sample per clock cycle after pipeline filling. Figure 1 illustrates the conceptual organization of the proposed streaming architecture and highlights the overlap between the range and azimuth processing stages.

The main contributions of this paper are summarized as follows:

- A fully streaming FPGA architecture for the Range-Doppler Algorithm that overlaps range and azimuth processing without centralized frame-level buffering.
- A distributed memory organization that replaces the conventional $N_r \times N_s$ frame buffer with stage-wise delay memories embedded in the streaming FFT pipeline.
- A radix-2 SDF FFT/IFFT architecture employing complementary DIF/DIT processing to eliminate explicit bit-reversal and intermediate transposition buffers.
- A deterministic streaming SAR processor that achieves one complex sample per clock cycle after pipeline filling while significantly reducing first-output latency and overall frame processing time.

2. Related Work

2.1. FPGA-Based SAR Processors

The Range-Doppler Algorithm (RDA) is one of the most widely adopted SAR image formation algorithms due to its computational efficiency and compatibility with frequency-domain processing [1]. Because the computational complexity of the RDA is dominated by one-dimensional FFT operations in both the range

and azimuth directions, FPGA implementations have received considerable attention for real-time SAR imaging [2], [7].

Most reported FPGA-based SAR processors employ a frame-oriented processing model. In these architectures, range compression is first completed for the entire SAR data frame. The resulting range-compressed samples are stored in a centralized $N_r \times N_s$ frame buffer, from which the azimuth processing stage subsequently reads its input [2], [3], [7]. This organization simplifies scheduling and memory access but forces range and azimuth processing to execute sequentially.

Consequently, the azimuth processor remains idle while the complete range-compressed frame is generated. In addition, intermediate frame buffering and data transposition introduce significant first-output latency and increase the complexity of memory management. Several studies have improved computational performance through parallel processing, optimized memory access, and efficient hardware scheduling [2], [3], [7]. Nevertheless, the fundamental frame-oriented execution model has remained largely unchanged.

2.2. Streaming FFT Architectures

Streaming FFT architectures are widely used in communication, radar, and signal processing systems because they provide deterministic timing, continuous dataflow, and high throughput [4],[5],[6],[8]. Among the various pipelined FFT architectures, the radix-2 Single-Path Delay Feedback (SDF) architecture is particularly attractive for FPGA implementation owing to its simple control logic, efficient hardware utilization, and ability to sustain one sample per clock cycle after pipeline filling [5],[6], [8].

Another important issue in streaming FFT processors is data ordering. Conventional FFT implementations generally require explicit bit-reversal or intermediate memory reordering, which introduces additional storage and latency [5], [6],[8]. Complementary Decimation-in-Frequency (DIF) and Decimation-in-Time (DIT) FFT/IFFT pairs provide an elegant solution by naturally compensating the ordering between consecutive transforms, thereby eliminating explicit bit-reversal buffers [4],[5],[6],[8].

Although streaming FFT architectures have been extensively investigated, their application has primarily focused on accelerating individual FFT operations rather than restructuring the execution of the complete Range-Doppler processing chain. Consequently, existing FPGA-based SAR processors still rely on centralized frame-level buffering between range and azimuth compression, preventing continuous streaming execution of the overall algorithm[2], [3], [7].

To the best of the author's knowledge, deterministic low-latency scheduling of the complete RDA using distributed stage-wise delay memories instead of a centralized frame buffer has received little attention in the existing literature. The architecture proposed in this work addresses this limitation by reorganizing the required memory into the intrinsic delay elements of streaming radix-2 SDF FFT pipelines, thereby enabling overlapped range and azimuth processing while maintaining continuous one-sample-per-clock streaming operation.

3. Proposed Streaming SAR Architecture

3.1. Overall Processing Architecture

Figure 2 illustrates the overall architecture of the proposed streaming SAR processor. Similar to the conventional Range-Doppler Algorithm (RDA), the processing chain consists of range compression followed by azimuth compression [1]. The range compression stage performs Range FFT, range matched filtering, and Range IFFT, whereas the azimuth compression stage consists of Azimuth FFT, azimuth matched filtering, and Azimuth IFFT.

The key difference from conventional implementations is not the computational sequence itself but the scheduling of these processing stages. Conventional FPGA implementations execute range and azimuth compression sequentially by inserting a centralized frame buffer between the two stages. Consequently, azimuth processing cannot begin until the entire range-compressed image has been generated.

In the proposed architecture, the centralized frame buffer is eliminated from the processing schedule. Instead, the memory required for streaming computation is distributed among the delay memories inherently present inside the radix-2 Single-Path Delay Feedback (SDF) FFT/IFFT processors. As a result, the complete processing chain operates as a single streaming pipeline rather than two independent processing stages.

Figure 2 shows that range compression and azimuth compression are directly connected through streaming interfaces. After the initial pipeline filling period, one complex sample is accepted and one processed sample is produced every clock cycle. This fully streaming organization enables deterministic timing while significantly reducing the delay before azimuth processing begins.

Unlike frame-based architectures, the proposed processor does not require completion of the entire SAR frame before initiating azimuth compression. Instead, processing proceeds continuously as valid streaming data become available, allowing both processing stages to operate concurrently.

3.2. Streaming Scheduling Strategy

The principal contribution of the proposed architecture lies in restructuring the execution schedule of the Range-Doppler Algorithm rather than modifying the algorithm itself. As illustrated in Fig. 1, the conventional and proposed processors perform exactly the same sequence of operations—Range FFT, matched filtering, Range IFFT, Azimuth FFT, matched filtering, and Azimuth IFFT. The difference lies entirely in how data are buffered and transferred between the range and azimuth processing stages.

In conventional FPGA implementations, range compression is completed for the entire SAR frame before azimuth processing begins [2], [3]. The range-compressed data are first stored in a centralized two-dimensional frame buffer of size $N_r \times N_s$. Since the azimuth FFT requires column-wise data access, the azimuth processor remains idle until the complete frame has been written into memory. Consequently,

range and azimuth compression are executed sequentially, introducing a first-output latency proportional to the frame size.

A straightforward solution would be to start azimuth processing immediately after the first range-compressed samples become available. However, this is not feasible with conventional frame-oriented FFT implementations because the required input sequence for the azimuth FFT cannot be formed until the complete frame has been assembled and reordered. Therefore, conventional architectures inherently depend on centralized frame buffering and intermediate data transposition.

The proposed architecture removes this scheduling constraint by replacing the centralized frame buffer with the distributed delay memories that are inherently required by the radix-2 Single-Path Delay Feedback (SDF) FFT/IFFT processors. Instead of acting as an intermediate storage between two independent processing stages, the required memory becomes an integral part of the streaming processing pipeline. Although the total memory capacity remains approximately unchanged, its organization fundamentally changes the execution schedule of the RDA.

By employing complementary Decimation-in-Frequency (DIF) and Decimation-in-Time (DIT) FFT/IFFT pairs, the streaming pipeline naturally generates the data ordering required by the subsequent processing stage without explicit frame-level buffering or bit-reversal memory [4],[5],[6]. Consequently, once the first N_s range-compressed samples have propagated through the range processing pipeline, sufficient data are already available to initiate the first azimuth FFT operation. From this point onward, range and azimuth compression execute concurrently in a continuous streaming manner.

This overlap between the two processing stages substantially reduces the first-output latency while preserving deterministic pipeline behavior. After the initial pipeline filling period, the processor accepts one complex input sample and produces one processed output sample every clock cycle. The resulting architecture therefore transforms the conventional sequential RDA implementation into a fully streaming low-latency processing pipeline without altering the underlying SAR image formation algorithm.

3.3. Distributed Memory Organization

The proposed processor employs a radix-2 Single-Path Delay Feedback (SDF) FFT/IFFT architecture to enable continuous streaming operation while eliminating the need for frame-level buffering [5],[6]. As illustrated in Fig. 2, each FFT stage consists of a complex butterfly, a stage-specific delay memory, and a twiddle-factor multiplication unit. The delay memories are distributed throughout the processing pipeline rather than being centralized in a dedicated frame buffer.

For an FFT of length N , the architecture contains $\log_2 N$ pipeline stages. For example, an FFT length of $N = 512$ requires nine stages. After the pipeline is filled, the architecture accepts one complex input sample and produces one output sample every clock cycle.

Unlike conventional frame-based FFT processors, the stage-wise delay memories simultaneously perform two functions: they provide the data alignment required by the radix-2 SDF algorithm while also

acting as the distributed storage needed for continuous dataflow between the range and azimuth processing chains. Consequently, no intermediate two-dimensional frame buffer is required.

As shown in Fig. 2, the distributed delay memories naturally preserve the streaming data sequence required by the subsequent azimuth FFT. Therefore, azimuth processing can begin immediately after the first N_s range-compressed samples become available, allowing the range and azimuth processing pipelines to operate concurrently.

3.4. Streaming Scheduling Overview

The proposed architecture achieves low-latency operation by changing the execution schedule of the Range-Doppler processing chain rather than by modifying the underlying algorithm. In conventional frame-based implementations, range compression must be completed and stored in an intermediate frame buffer before azimuth processing can begin. In the proposed architecture, the distributed delay memories embedded within the streaming SDF FFT/IFFT pipelines provide the synchronization required for continuous data propagation. As a result, azimuth processing can start once sufficient range-compressed samples become available, allowing range and azimuth compression to overlap in time. The detailed scheduling model, latency formulation, and quantitative timing comparison are presented in Section 4.

4. Streaming Scheduling Analysis

4.1. Streaming Execution Mode

The primary contribution of the proposed architecture lies in its streaming scheduling strategy rather than in modifying the computational algorithm itself. The Range-Doppler Algorithm (RDA) remains unchanged; however, the execution order of its processing stages is fundamentally reorganized to enable continuous dataflow and low-latency operation.

In conventional FPGA implementations, range compression and azimuth compression are executed sequentially. During range compression, all N_r range lines, each containing N_s samples, are processed and stored in a centralized two-dimensional frame buffer. Only after the complete range-compressed frame has been generated does azimuth processing begin. Consequently, the azimuth processing pipeline remains idle while the entire frame is being produced.

The proposed architecture removes this frame-level scheduling dependency. Instead of waiting for the complete range-compressed image, azimuth processing starts as soon as sufficient streaming data become available. Specifically, after the first N_s range-compressed samples have been generated, the first azimuth FFT can be initiated. This allows range compression and azimuth compression to execute concurrently rather than sequentially.

This scheduling strategy is enabled by the streaming radix-2 Single-Path Delay Feedback (SDF) FFT/IFFT architecture employing complementary Decimation-in-Frequency (DIF) and Decimation-in-Time (DIT)

processing. The resulting data ordering naturally provides the samples required for the subsequent processing stages without requiring explicit frame-level buffering or intermediate data transposition.

As illustrated in Fig. 1, once the initial pipeline filling period has elapsed, data continuously propagate through the complete processing chain. Every processing stage remains active simultaneously, sustaining a throughput of one complex sample per clock cycle while significantly reducing the end-to-end processing latency.

4.2. Latency Analysis

The primary advantage of the proposed architecture is the significant reduction in end-to-end processing latency achieved through overlapped execution of range and azimuth compression. Figure 1(c) illustrates the execution schedules of the conventional frame-based architecture and the proposed fully streaming processor, while the quantitative comparison is summarized in Fig. 1(d).

In a conventional implementation, azimuth processing cannot begin until all N_r range lines, each containing N_s samples, have been processed and stored in the intermediate frame buffer. Consequently, the azimuth processor remains idle during the entire range-compression stage, resulting in a large first-output latency.

In the proposed architecture, azimuth processing begins immediately after the first N_s range-compressed samples become available. Because the distributed delay memories continuously supply the required input data, range and azimuth compression proceed concurrently without waiting for completion of the entire image frame.

Based on the execution schedule illustrated in Fig. 1(c), the first-output latency of the conventional architecture can be expressed as

$$2N_r N_s + N_r + N_s$$

whereas the proposed streaming architecture reduces it to

$$N_s + 2N_r$$

Similarly, the total processing time required to generate one complete SAR image is

$$2N_r N_s + N_r + N_s$$

for the conventional implementation, while the proposed processor requires

$$N_r N_s + 2N_r + N_s$$

For the experimental configuration used in this work ($N_r = 512$ and $N_s = 1024$), the first-output latency is reduced from 525,824 clock cycles to only 2,048 clock cycles. Furthermore, the total frame processing time decreases from 1,050,112 to 526,336 clock cycles. After the initial pipeline filling period, both architectures maintain the same steady-state throughput of one complex sample per clock cycle. At the

system level, the conventional frame-based architecture achieves an effective throughput of about 0.5 sample per clock cycle, whereas the proposed streaming architecture approaches one sample per clock cycle by overlapping range and azimuth processing.

4.3. Complexity and Memory Discussion

The proposed architecture preserves the original computational flow of the Range-Doppler Algorithm (RDA) while fundamentally changing its execution schedule. No additional signal-processing operations are introduced, and the sequence of range compression, azimuth compression, and matched filtering remains identical to that of the conventional implementation. Consequently, the arithmetic complexity of the processor remains unchanged.

The proposed architecture is also not intended to reduce the total amount of on-chip memory required for SAR processing. Instead, it reorganizes the memory from a centralized frame buffer into distributed stage-wise delay memories embedded within the streaming radix-2 SDF FFT/IFFT pipelines. Therefore, the improvement does not arise from reducing memory capacity, but from changing how the required memory participates in the computation.

In conventional frame-based implementations, the intermediate frame buffer acts as passive storage between two sequential processing stages. In the proposed processor, the distributed delay memories become active components of the streaming pipeline, providing both data synchronization and algorithmic delay requirements. This allows data to propagate continuously through the complete processing chain, enabling range and azimuth processing to overlap after the initial pipeline filling period.

As a result, the first-output latency is no longer dominated by the complete $N_r \times N_s$ frame size. Instead, it is determined by the availability of the first N_s range-compressed samples and the depth of the streaming pipeline. The steady-state throughput remains one complex sample per clock cycle, meaning that the latency improvement is achieved through architectural scheduling and memory reorganization rather than by increasing the computational throughput of the FFT processors.

From a system perspective, the principal benefit of the proposed architecture is deterministic low-latency operation. This property makes the proposed processor particularly suitable for real-time embedded SAR systems, where predictable timing and reduced end-to-end latency are often more critical than peak computational throughput.

4.4. Conceptual Comparison of Processing Organizations

To clarify the architectural role of the proposed scheduling model, Table 1 provides a qualitative comparison of several representative SAR processing organizations. The comparison focuses on processing mode, memory organization, and latency behavior rather than implementation-specific FPGA resources or operating frequency.

Table 1
Conceptual Comparison of SAR Processing Organizations

Architecture	Processing Mode	Memory Organization	Latency
Frame-based SAR	Sequential	Full frame buffer	High
Line-based / Tiled	Partial streaming	Line buffers	Medium
GPU-based SAR	Batch processing	External memory	Variable
Conventional FPGA	Sequential	Intermediate frame buffer	High
Proposed	Fully streaming	Distributed SDF delay memories	Deterministic / Low

As shown in Table 1, conventional frame-based FPGA SAR processors separate range and azimuth compression using an intermediate frame buffer. This organization simplifies memory management but forces the two processing stages to operate sequentially, resulting in high first-output latency. Line-based or tiled organizations can reduce the amount of intermediate buffering, but they still rely on partial data accumulation and do not provide a fully continuous range-to-azimuth streaming pipeline.

GPU-based SAR processors are typically optimized for batch throughput using external memory and therefore do not provide deterministic sample-level latency. In contrast, the proposed architecture integrates the required synchronization storage into the distributed delay memories of the radix-2 SDF FFT/IFFT pipelines. This enables continuous data propagation between range and azimuth processing, allowing the two stages to overlap in time while maintaining deterministic low-latency streaming operation.

5. FPGA Implementation and Experimental Results

5.1. Experimental Setup

The proposed streaming SAR processor was developed in MATLAB/Simulink using Xilinx System Generator and automatically translated into synthesizable HDL. The generated RTL was synthesized and implemented using Xilinx Vivado 2023.2 targeting a Xilinx Kintex UltraScale XCKU060-FFVA1156-2-I FPGA at a target clock frequency of 200 MHz's.

All processing modules were implemented using a 16-bit signed fixed-point representation. The FFT/IFFT processors employ a radix-2 SDF architecture, while the matched filtering stages perform complex multiplication with the corresponding reference functions. The experimental configuration uses $N_r = 512$ azimuth lines and $N_s = 1024$ range samples per line, corresponding to the SAR imaging scenario considered throughout this work.

The implementation results presented in the following sections are obtained from the synthesized FPGA design and are used to evaluate functional correctness, hardware utilization, and processing latency.

5.2. FPGA Implementation

The proposed streaming architecture was implemented as a fully pipelined hardware design using Xilinx System Generator. Each processing stage operates independently and exchanges data through streaming interfaces synchronized by valid signals. After the initial pipeline filling, all modules process one complex sample per clock cycle.

The implementation consists of six processing modules: Range FFT, range matched filtering, Range IFFT, Azimuth FFT, azimuth matched filtering, and Azimuth IFFT. The FFT/IFFT processors are implemented using radix-2 SDF architectures, while the matched filtering stages perform complex multiplication with the corresponding reference functions.

Unlike conventional frame-based implementations, no centralized two-dimensional frame buffer is inserted between range and azimuth processing. Instead, data synchronization is achieved through the distributed delay memories within the streaming FFT pipeline, allowing continuous data propagation between consecutive processing stages.

The complete design was automatically translated into synthesizable HDL by System Generator and implemented on the target FPGA without manual RTL modification.

5.3. Resource Utilization and Timing Analysis

The hardware resource utilization of the proposed streaming SAR processor was obtained after synthesis and implementation in Xilinx Vivado. Table 2 summarizes the utilization of the primary FPGA resources, including lookup tables (LUTs), flip-flops (FFs), block RAMs (BRAMs), and DSP slices.

The implementation demonstrates that the proposed architecture can be realized on a mid-range FPGA while maintaining a fully streaming processing pipeline. As discussed in Section 3, the proposed architecture is not intended to reduce the total memory capacity. Consequently, the BRAM utilization is expected to be comparable to that of conventional frame-based implementations. The principal advantage of the proposed design is the reduction of end-to-end latency through streaming scheduling rather than a reduction in hardware resources.

Table 2
Hardware Modules of the Proposed Streaming SAR Processor

Name	FF	LUT	BRAM	DSP	Latency cycles
Range FFT	4k	2k	4	40	~ 1024
Range MF	163	4k	1	4	~ 0
Range IFFT	4k	2k	4	40	~ 1024
Azimuth FFT	5k	4k	488	24	~ 512
Azimuth MF	2k	2k	2	5	~ 0
Azimuth IFFT	4k	4k	488	24	~ 512

The synthesized design successfully met all timing constraints after implementation in Xilinx Vivado. The maximum operating frequency reached 262 MHz with a WNS of + 1.184 ns and a TNS of 0 ns, confirming stable timing closure and supporting continuous one complex sample per clock cycle streaming operation after pipeline filling.

The relatively high BRAM utilization in the azimuth FFT/IFFT stages is caused by the required long synchronization delays associated with azimuth-direction streaming. In a conventional frame-based implementation, this storage would appear as a centralized two-dimensional frame buffer. In the proposed architecture, the same storage requirement is redistributed across the stage-wise delay memories of the streaming SDF pipelines. Therefore, the BRAM usage should not be interpreted as additional memory overhead; rather, it represents the relocation of the buffering function from a passive frame memory to active streaming delay elements.

5.4. Functional Verification

The functional correctness of the proposed streaming SAR processor was verified by comparing the FPGA implementation with a floating-point MATLAB reference model. Identical input SAR data were applied to both implementations, and the generated range-compressed and focused SAR images were compared.

The comparison confirmed that the FPGA implementation produces results consistent with the MATLAB model. Any observed differences are attributed to the finite word-length effects of the 16-bit fixed-point arithmetic and remain within acceptable limits for SAR image formation.

The proposed processor successfully performs continuous streaming operation while preserving the numerical accuracy of the reference implementation. These results confirm that the streaming scheduling strategy and distributed memory organization do not alter the functionality of the original Range-Doppler Algorithm.

The fixed-point FPGA output was compared with the floating-point MATLAB reference model using the normalized mean square error (NMSE). The obtained NMSE was below 0.1%, indicating that the fixed-point quantization error is very small compared with the reference output energy and does not significantly affect the SAR image formation result.

5.5. Performance Evaluation

The performance of the proposed streaming architecture was evaluated in terms of first-output latency, total frame processing time, and steady-state throughput. The results were compared with those of a conventional frame-based implementation executing the same Range-Doppler Algorithm under identical processing parameters.

Table 3 summarizes the quantitative comparison between the two architectures. In the conventional implementation, azimuth processing starts only after the complete range-compressed frame has been generated and stored. Consequently, the first valid output is produced after 525,824 clock cycles, and one complete SAR frame requires 1,050,112 clock cycles to process.

In contrast, the proposed streaming architecture initiates azimuth processing immediately after the first $N_{s,range}$ -compressed samples become available. As a result, the first-output latency is reduced to only 2,048 clock cycles, while the total frame processing time decreases to 526,336 clock cycles.

The conventional frame-based architecture achieves only about 0.5 sample per clock cycle at the frame level, whereas the proposed streaming architecture reaches nearly one sample per clock cycle by overlapping range and azimuth processing. These results demonstrate that the proposed streaming organization significantly improves the temporal efficiency of FPGA-based SAR processing while preserving the computational algorithm and maintaining deterministic real-time operation.

Table 3
Performance Comparison between Conventional and Proposed Streaming Architectures

Method	Total Cycles	First-output latency (cycles)	Throughput
Classic (Frame-Based)	1050112	525824	0.4993
Proposed (Fully Streaming)	526336	2048	0.9961
Note: Throughput is reported in samples per clock cycle.			

5.6. Comparison with Previous Works

To further evaluate the effectiveness of the proposed architecture, Table 4 compares the proposed processor with several representative FPGA-based SAR implementations reported in the literature. The comparison includes the processing architecture, memory organization, streaming capability, and latency characteristics.

Most existing FPGA-based SAR processors adopt a frame-oriented processing strategy, in which range and azimuth compression are executed sequentially with an intermediate frame buffer. Although these implementations achieve high computational throughput, the frame-based scheduling introduces considerable end-to-end latency.

In contrast, the proposed architecture replaces the centralized frame buffer with distributed stage-wise delay memories embedded within the streaming FFT pipeline. This organization enables azimuth processing to begin immediately after the first N_s range-compressed samples become available, thereby overlapping the range and azimuth processing stages without modifying the underlying RDA.

Unlike approaches that primarily improve computational performance through parallelism or increased hardware resources, the proposed work focuses on reducing processing latency by reorganizing the execution schedule. The comparison demonstrates that the proposed architecture achieves deterministic low-latency streaming operation while preserving the computational complexity of the conventional algorithm.

Table 4
Comparison with Representative FPGA-Based SAR Processors

Work	Stream	Frame Buffer	Overlap	Latency	Throughput
[2]	No	Yes	No	High	1
[3]	Partial	Yes	No	High	1
[7]	Partial	Yes	No	High	1
Proposed	Yes	No	Yes	Low	1

6. Conclusion

This paper presented a fully streaming FPGA architecture for the Range-Doppler Algorithm (RDA) aimed at reducing the end-to-end latency of real-time SAR image formation. Unlike conventional frame-based implementations, the proposed architecture eliminates centralized frame-level buffering by distributing the required synchronization memory across the stage-wise delay memories of streaming radix-2 SDF FFT/IFFT processors. This organization enables continuous dataflow throughout the complete processing chain.

By allowing azimuth processing to begin immediately after the first N_s range-compressed samples become available, the proposed architecture overlaps range and azimuth compression while preserving the original computational flow of the RDA. As demonstrated by the timing analysis and FPGA implementation results, the proposed scheduling strategy significantly reduces both the first-output latency and the total frame processing time without increasing the computational complexity or reducing the steady-state throughput.

The FPGA implementation confirms that the proposed architecture provides deterministic low-latency operation while maintaining one complex sample per clock cycle after pipeline filling. Future work will investigate extending the proposed streaming scheduling strategy to more advanced SAR image formation algorithms and evaluating its performance on larger SAR datasets and next-generation FPGA platforms.

Declarations

Statements and Declarations

Funding

The author received no financial support for the research, authorship, and/or publication of this article.

Competing Interests

The author declares no competing interests.

Author Contributions

Abbas Fadavi was responsible for conceptualization, methodology, implementation, analysis, visualization, and writing of the manuscript.

Data Availability

The data generated and/or analyzed during the current study are available from the corresponding author upon reasonable request.

Use of Generative AI and AI-Assisted Technologies

During the preparation of this manuscript, the author used ChatGPT to assist with language refinement, readability improvement, manuscript organization, formatting guidance, and the preparation/editing of illustrative schematic figures. The AI-assisted figures were used only for visual representation of the proposed architecture and scheduling concept, not for generating experimental data or numerical results. The technical content, methodology, numerical results, interpretations, figures, and final version of the manuscript were reviewed, verified, and approved by the author. The author takes full responsibility for the content of the manuscript.

The data generated and analyzed during the current study are available from the corresponding author upon reasonable request.

References

1. Cumming, I. G., & Wong, F. H. (2005). *Digital processing of synthetic aperture radar data: Algorithms and implementation*. Artech House.
2. Choi, Y., Jeong, D., Lee, M., Lee, W., & Jung, Y. (2021). FPGA implementation of the Range-Doppler Algorithm for real-time synthetic aperture radar imaging. *Electronics*, 10(17), 2133. <https://doi.org/10.3390/electronics10172133>
3. Wang, G., Chen, H., & Xie, Y. (2021). An efficient dual-channel data storage and access method for spaceborne synthetic aperture radar real-time processing. *Electronics*, 10(6), 662. <https://doi.org/10.3390/electronics10060662>
4. He, S., & Torkelson, M. (1998). Designing pipeline FFT processor for OFDM (de)modulation. In Proceedings of the 1998 URSI International Symposium on Signals, Systems, and Electronics. <https://doi.org/10.1109/ISSSE.1998.738077>
5. Garrido, M., Grajal, J., Sánchez, M. A., & Gustafsson, O. (2013). Pipelined radix-2^k feedforward FFT architectures. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 21(1), 23–32. <https://doi.org/10.1109/TVLSI.2011.2178275>
6. Garrido, M. (2022). A survey on pipelined FFT hardware architectures. *Journal of Signal Processing Systems*, 94(11), 1345–1364. <https://doi.org/10.1007/s11265-021-01655-1>
7. Lin, J. Z., Chen, P. T., Chin, H. Y., Tsai, P. Y., & Lee, S. Y. (2024). Design and implementation of a real-time imaging processor for spaceborne synthetic aperture radar with configurability. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 32(4), 669–681. <https://doi.org/10.1109/TVLSI.2023.3338476>
8. Li, Y., Chen, H., & Xie, Y. (2021). An FPGA-based four-channel 128k-point FFT processor suitable for spaceborne SAR. *Electronics*, 10(7), 816. <https://doi.org/10.3390/electronics10070816>

Figures

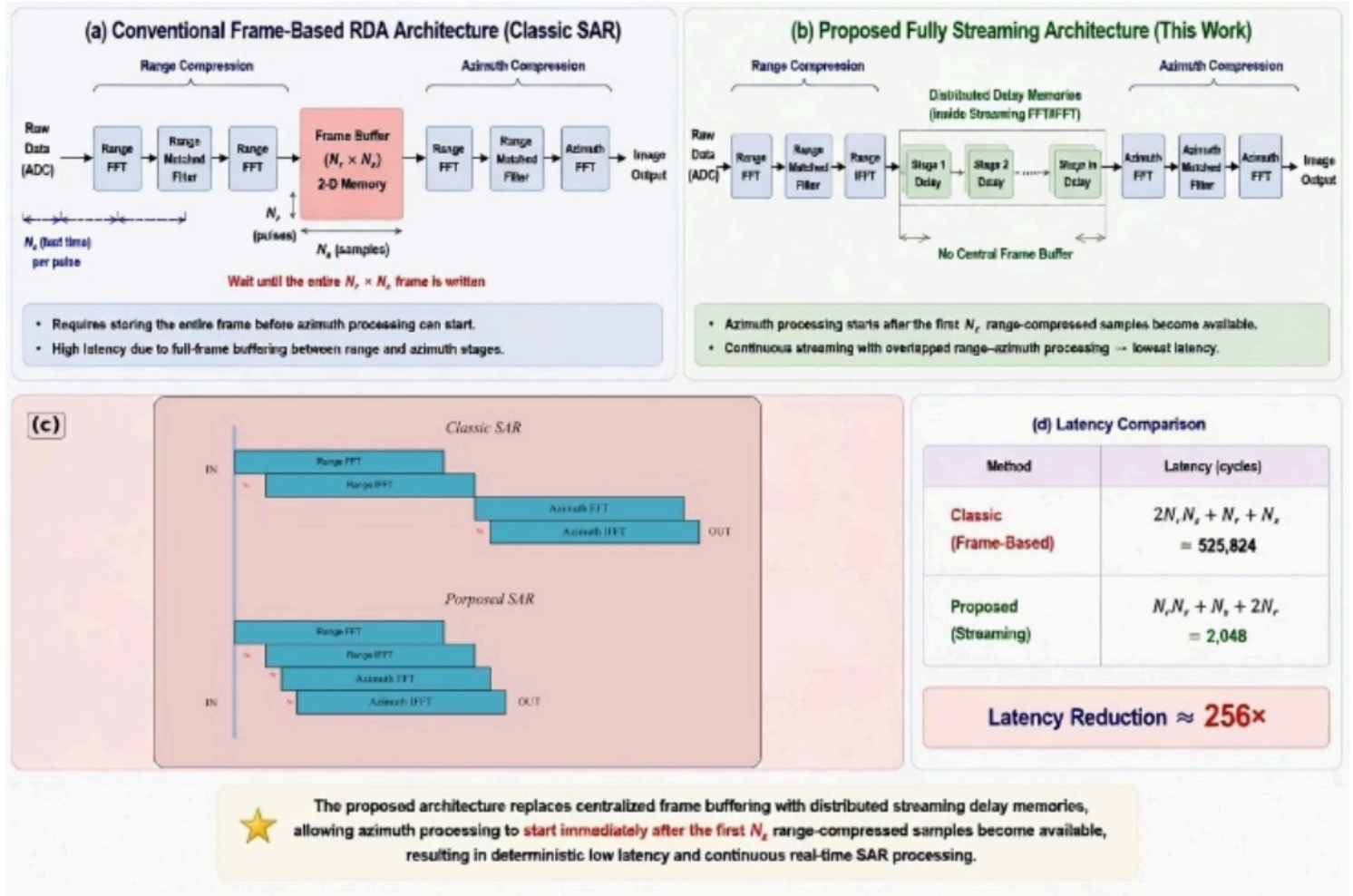


Figure 1

Conceptual comparison between the conventional frame-based RDA processor and the proposed fully streaming architecture. (a) Conventional sequential processing flow, where azimuth compression starts only after completion of the full range-compressed frame. (b) Proposed streaming processing flow, where range and azimuth compression are overlapped. (c) Temporal scheduling behavior of the two architectures. (d) Latency and total processing-time comparison for $N_r = 512$ and $N_s = 1024$.

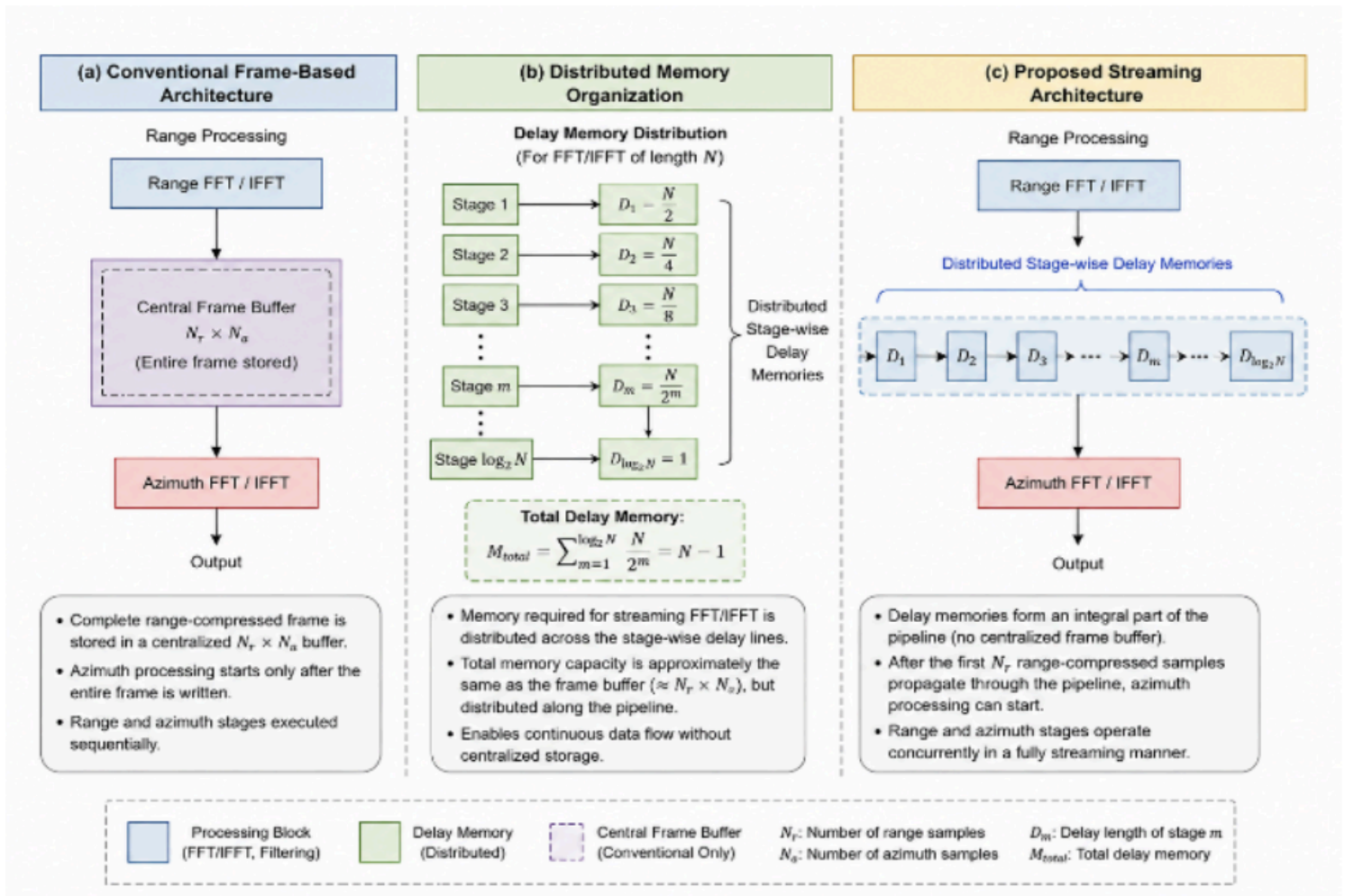


Figure 2

Distributed memory organization of the proposed fully streaming SAR processor. The figure compares the conventional centralized frame-buffer organization with the proposed distributed stage-wise delay memories embedded in radix-2 SDF FFT/IFFT pipelines. This memory reorganization enables continuous data propagation between range and azimuth processing without centralized frame-level buffering.