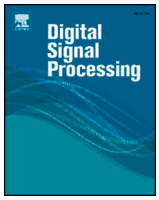
Digit. Signal Process. 155 (2024) 104732

Contents lists available at ScienceDirect

Digital Signal Processing

journal homepage: www.elsevier.com/locate/dsp



# Enhancing efficiency in spaceborne phased array systems: MVDR algorithm and FPGA integration

Eduardo Ortega *, Alejandro Vicente, Agustín Martínez, Óscar Rodríguez, Manuel Prieto, Pablo Parra, Antonio Da Silva, Sebastián Sánchez

*Space Research Group, Department of Automatics, University of Alcalá, Alcalá de Henares, 28805, Madrid, Spain*

ARTICLE INFO

*Keywords:*
FPGA
MVDR
Digital beamforming

ABSTRACT

Digital Beamforming (DBF) has garnered significant attention within the space community, driven by the heightened flexibility offered by satellites equipped with active antennas for diverse applications like spaceborne synthetic aperture radar (SAR) systems and communication services. This paper presents a comprehensive exploration encompassing analysis, design, and implementation of a Minimum Variance Distortionless Response (MVDR) algorithm tailored for application in a Digital Beamforming Receiver system. The described system leverages the capabilities of the high-speed Analog to Digital Converter (ADC) device EV12AQ600, space-qualified, and the high-performance FPGA XCKU085. These components are seamlessly integrated into the EV12AQ60X-ADX-EVM evaluation board, thoughtfully provided by Teledyne e2v Semiconductors. The design methodology outlined herein is driven by the overarching objective to minimize FPGA resource utilization and power consumption for spaceborne phased array systems. This objective is chiefly met through the implementation of a systolic array structure adept at processing both real and complex input samples. This innovative approach results in a substantial reduction in allocated resources compared to conventional architectures. Furthermore, the design incorporates the use of the CORDIC algorithm to compute the inverse of the correlation matrix, obviating the need for square root and division operations. This strategic utilization of algorithms enhances the system's efficiency in terms of resource utilization, computational load, and processing time. To validate the anticipated behavior of the adaptive beamforming system, various radio frequency (RF) input signals are applied to the system, which is physically instantiated on the EV12AQ60X-ADX-EVM evaluation board. The experimental results affirm the efficacy of the proposed design, underscoring its suitability for spaceborne applications.

## 1. Introduction

Digital beamforming (DBF) is a revolutionary technique that dynamically alters the radiation pattern of an antenna array using only digital signal processing (DSP). Unlike traditional methods that require analog hardware for signal adjustments, DBF uses digital processing, transforming a conventional antenna into a "smart" antenna. This flexibility allows for energy-efficient processing and easy reconfiguration through software, making DBF ideal for space applications where traditional analog hardware is limited. DBF enables communication payloads in satellites to adapt to changing mission requirements, overcoming the constraints of analog signal chains and enhancing operational capabilities [1], [2]. Recent advancements in Analog-to-Digital Converter (ADC) devices have significantly improved performance, with increased sample rates and power efficiency [3], [4], [5]. These improvements have transformed space-based array architectures by shifting towards digital processing. High-performance ADCs and radiation-tolerant digital signal processors, such as the Kintex UltraScale XQRKU060, now handle digitized RF signals, introducing a digital layer to traditionally analog systems. This shift enhances the adaptability and performance of space-based arrays, highlighting the transformative impact of digitalization on space-based communication systems [6].

Digital beamforming and adaptive techniques have found diverse applications within the space community, notably in the domain of spaceborne synthetic aperture radar (SAR) systems and satellite telecommunications systems. Particularly, for SAR systems operating with extended-duration pulses, sophisticated beamforming approaches, including the Minimum Variance Distortionless Response (MVDR), play a pivotal role.

* Corresponding author.
*E-mail address:* eduardo.ortega@uah.es (E. Ortega).

https://doi.org/10.1016/j.dsp.2024.104732

Available online 14 August 2024

These advanced techniques are instrumental in mitigating Signal-to-Noise Ratio (SNR) losses, thereby enhancing the overall performance of SAR systems. Complementary adaptive methods, such as Linearly Constrained Minimum Variance (LCMV) beamforming, contribute further to elevating SAR image quality, as discussed by Huber et al. [7]. In the realm of satellite telecommunication systems, the demand for high-gain satellite antennas is critical to ensuring adequate link budgets, particularly for small user terminals. The integration of multiple beam antennas (MBA) on satellites has proven successful in various applications, spanning personal communications, military communications, and Internet services. Adaptive beamforming techniques, a key component in such systems, play a crucial role in optimizing the Signal-to-Interference plus Noise Ratio (SINR) and improving radio resource utilization efficiency [8]. The significance of adaptive beamforming techniques, especially MVDR, becomes evident in achieving substantial levels of interference cancellation [9]. This capability is paramount for enhancing the robustness and reliability of spaceborne systems in the face of challenging interference scenarios. In summary, the widespread adoption of adaptive beamforming techniques, exemplified by MVDR and LCMV, underscores their pivotal role in elevating the performance and efficiency of spaceborne synthetic aperture radar systems and satellite telecommunication systems. These techniques prove indispensable in overcoming challenges related to interference, ensuring optimal SNR, and maximizing resource utilization efficiency in space-based applications.

This paper presents a comprehensive exploration involving the analysis, design, and practical implementation of a MVDR beamforming algorithm. The algorithm is strategically applied to a Digital Beamforming Receiver system, featuring cutting-edge components such as the high-speed Analog to Digital Converter (ADC) device EV12AQ600 and the high-performance FPGA XCKU085. These components are seamlessly integrated into the sophisticated EV12AQ60X-ADX-EVM evaluation board, a testament to Teledyne e2v Semiconductors' commitment to advanced semiconductor solutions. The Digital Beamforming Receiver system adopts a unique approach by applying the beamforming algorithm prior to the complex down-conversion process [10]. This arrangement ensures that incoming input samples, initially digitized by the high-speed ADC, undergo processing by the adaptive beamformer, resulting in real samples for subsequent stages. The digital signal processing system leverages Inverse QR Decomposition (IQRD) with Givens rotations and the Coordinate Rotation Digital Computer (CORDIC) algorithm for practical FPGA implementation with fixed-point precision. This enables multiplication-free vector rotations, contributing to enhanced computational efficiency. The design achieves a fully pipelined structure with minimal memory usage and maximal parallelism, facilitating real-time signal processing in the FPGA hardware implementation. Tailoring the systolic array structure implemented for IQRD to accommodate different types of input samples—real number values from the ADC and complex number samples representing steering vector values—addresses the specific requirements of spaceborne phased arrays. This innovative approach allows for the combination of various processing cells within the array structure, minimizing resource usage and resulting in a streamlined and novel implementation of an adaptive beamforming architecture. In essence, the presented work not only advances the understanding of MVDR beamforming but also showcases a practical and efficient FPGA-based implementation tailored to meet the stringent demands of spaceborne phased arrays. The proposed architecture stands as a testament to innovation in size, power, and performance optimization, contributing to the evolution of beamforming technologies for space applications.

In the subsequent sections, we present a detailed hardware design of a digital beamforming architecture employing the MVDR technique. As demonstrated in our work, this novel signal processing design markedly reduces resource usage, computational load, processing time and power consumption, aligning with the critical requirements for spaceborne phased arrays. A comparative analysis of resource utilization, computational load and processing time is conducted between our proposed architecture and a conventional approach, denoted as the "standard architecture". The standard architecture refers to one utilizing processing elements of the same type within the systolic array structure, specifically handling complex types of samples [11], [12]. To validate the implementation, synthetic RF signals are generated using MATLAB, and the results are compared between the ideal outcome with floating-point precision and the results obtained from our hardware implementation. The comparison illustrates the expected behavior and performance of the proposed system. It's important to note that during validation tests, we assume knowledge of the angle of arrival of the desired signal, eliminating any performance impact related to mismatches between the actual and assumed signals by the beamformer model. Before delving into the intricacies of our proposed design, the following sections provide a concise introduction to beamforming theory and offer a detailed description of the chosen hardware platform integral to our implementation.

## 2. Adaptive beamforming: state of the art

Conventional digital beamforming techniques rely on assumptions about the signal characteristics, such as known direction and delay per sensor, resulting in weight values that do not adapt to dynamic conditions [13]. These weights values are determined independently of the data to be processed [14]. To address these limitations, the conventional beamforming algorithm is modified to incorporate spatial correlation, enhancing adaptability to varying signal and noise conditions. This approach optimizes reception performance by considering the dynamic electromagnetic environment, particularly valuable in scenarios with time-varying signal and noise characteristics. Adaptive signal processing algorithms based on estimation theory offer powerful methods to dynamically adjust sensor outputs [13]. These algorithms consider signal frequency, spatial and temporal noise characteristics, and the number of observed signals. When sufficient observations are available, adaptive methods demonstrate superior performance compared to conventional techniques.

In adaptive beamforming, weights are calculated based on processed array data to optimize spatial response, enhancing desired signals while suppressing unwanted ones. Adaptive beamformers estimate matrix correlation from array snapshots, adjusting weights dynamically based on observed characteristics. This adaptability is crucial in scenarios with diverse signal frequencies, spatial intricacies, temporal noise variations, and multiple signals. Numerous examples in the literature illustrate the successful application of these adaptive algorithms in digital signal processing [15], [16], [17], [18], [19], [20], [8].

### 2.1. Array signal model

Considering a signal received by a uniform linear array (ULA) of $M$ elements with interelement spacing $d$, the output of a narrowband beamformer is expressed as [21]

$$\mathbf{y}(t) = \mathbf{w}^H \mathbf{x}(t) \tag{1}$$

where $t$ is the time index, $\mathbf{x}(t) = [x_1(t)\ x_1(t)\ \dots\ x_M(t)]^T$ is the $M \times 1$ vector of array observation, $\mathbf{w} = [w_1,\ w_2 \dots w_M]^T$ is the $M \times 1$ complex vector of beamformer weights. $(\cdot)^T$ and $(\cdot)^H$ stand for the transpose and Hermitian transpose, respectively. The observation vector may be written containing the individual sensor signals [14]

$$\mathbf{x}(t) = \mathbf{s}(t) + \mathbf{i}(t) + \mathbf{n}(t) = \mathbf{v}(\phi_s)s(t) + \mathbf{i}(t) + \mathbf{n}(t) \tag{2}$$

where $s(t)$ is the signal of interest (SOI) at time $t$ with deterministic amplitude and uniformly distributed random phase. $\mathbf{v}(\phi_s)$ is the $M \times 1$ steering vector at the $\phi_s$ direction of arrival (DOA) of the SOI. $\mathbf{i}(t)$ and $\mathbf{n}(t)$ represents the interference signal vector and thermal noise vector at time $t$.

The interference plus noise components of the array, $\mathbf{x}_{i+n}(t) = \mathbf{i}(t) + \mathbf{n}(t)$, are both modeled as zero-mean stochastic processes. The interference signal exhibits spatial correlation in accordance with the angles of arrival to the sensor array, while the thermal noise is considered to be spatially uncorrelated. It is assumed that these three components are mutually uncorrelated. Consequently, the array correlation matrix can be defined by the following expression [14]

$$\mathbf{R}_x = E\{\mathbf{x}\mathbf{x}^H\} = \sigma_s^2 \mathbf{v}(\phi_s)\mathbf{v}^H(\phi_s) + \mathbf{R}_i + \mathbf{R}_n \tag{3}$$

Being $\sigma_s^2$ the power of the signal of interest and $\mathbf{R}_i$ and $\mathbf{R}_n$ are the interference and noise correlation matrices, respectively. $E[\cdot]$ represents the mathematical expectation. The noise correlation matrix can be defined as $\mathbf{R}_n = \sigma_n^2\mathbf{I}$, where $\sigma_n^2$ is the power of the noise and $\mathbf{I}$ is the identity matrix, as the sensor thermal noise is spatially uncorrelated. The interference-plus-noise correlation matrix is defined as [14]

$$\mathbf{R}_{i+n} = \mathbf{R}_i + \sigma_n^2\mathbf{I} \tag{4}$$

The objective of adaptive beamforming is to reduce the interference signal to the level of thermal noise by combining sensor signals while preserving the desired signal [14]. In doing so, the aim is to maximize the ratio of the desired signal power to that of the interference plus noise, commonly referred to as the SINR. Maximizing SINR enhances the visibility of the desired signal amidst background interference.

In practical implementation, the interference-plus-noise correlation matrix is estimated through adaptive methods based on data collection. Two types of methods are commonly employed: block adaptive and sample-by-sample adaptive. The block adaptive method analyzes a "block" of data (snapshot) to estimate the adaptive beamforming weight vector, while the sample-by-sample adaptive method updates the statistics with each new sample read. In this work, we consider the block adaptive method for the MVDR beamforming.

### 2.2. Minimum variance distortionless response (MVDR)

Adaptive beamforming algorithms are instrumental in optimizing beamformer responses by estimating data array statistics, with the primary aim of minimizing the influence of noise and interfering signals. The challenge of insufficient knowledge about the desired signal within the sensor array is addressed through the application of linear constraints to the weight vector. This method allows for precise control over the beamformer's response, enabling signals from the direction of interest to pass with specific gain and phase characteristics. The linearly constrained minimum variance (LCMV) technique is employed, wherein the selected weight values undergo an optimization process that minimizes output variance or power, while adhering to the response constraints. This approach ensures the preservation of the desired signal while concurrently minimizing interference signals and noise originating from different directions of interest [21].

The beamformer response to a source arriving at angle $\theta$ is given by [21]

$$\mathbf{w}^H\mathbf{v}(\theta) \tag{5}$$

In this manner, by linearly constraining the weights to satisfy [21]

$$\mathbf{w}^H\mathbf{v}(\theta) = g \tag{6}$$

where $g$ is a complex constant, it is ensured that any signal from angle $\theta$ is passed to the output with response $g$.

Minimization of contributions to the output from signals not arriving from angle $\theta$ is accomplished by choosing the weights to minimize the output power or variance [21]

$$E\{|y|^2\} = \mathbf{w}^H\mathbf{R}_x\mathbf{w} \tag{7}$$

The LCMV problem for choosing the weights is thus written [21]

$$min\ \mathbf{w}^H\mathbf{R}_x\mathbf{w}\ subject\ to\ \mathbf{v}^H(\theta)\mathbf{w} = g^* \tag{8}$$

The symbol $*$ denotes the complex conjugate. Utilizing the Lagrange multipliers method to solve the equation results in [21]

$$\mathbf{w} = g^*\frac{\mathbf{R}_x^{-1}\mathbf{v}(\theta)}{\mathbf{v}^H(\theta)\mathbf{R}_x^{-1}\mathbf{v}(\theta)} \tag{9}$$

When the specified gain response $g$ is fixed at 1, the resulting equation is commonly recognized as the minimum variance distortionless response (MVDR) beamformer. The invertibility of the correlation matrix $R_x$ is assured by the presence of uncorrelated noise. The determination of the optimized beamformer weight vector necessitates knowledge of the second-order statistics associated with the information received by the sensor array. Given that these statistics are often unknown, their estimation becomes imperative and is typically derived from the available captured data.

Adaptive algorithms are employed to ascertain these statistical parameters, recognizing the potential temporal variations in the statistics. In the block adaptation approach, a temporal block of array data is subjected to analysis and incorporated into the weight equation. This method estimates the correlation matrix formed from $K$ samples. The estimation of the sample covariance matrix is calculated using Eq. (10), [22].

$$\hat{\mathbf{R}}_x = \frac{1}{K}\sum_{k=1}^{K}\mathbf{x}(n_k)\mathbf{x}^H(n_k) \tag{10}$$

where the indices $n_k$ define the $K$ samples of $\mathbf{x}(n_k)$ for $1 \le n \le N$ that compose the *trainingset*. The symbol$\hat{}$in the formula represents the estimated correlation matrix. This estimation indicates that as $K \to \infty$, the estimated correlation matrix $\hat{\mathbf{R}}_x$ approaches the true correlation matrix $\mathbf{R}_x$ [14].

In this study, the correlation matrix is constructed by considering only the interference and noise data, excluding the desired signal, as further discussed in Section 5.1. Consequently, the sample covariance matrix is defined as [14]

$$\hat{\mathbf{R}}_{i+n} = \frac{1}{K}\sum_{k=1}^{K}\mathbf{x}_{i+n}(n_k)\mathbf{x}_{i+n}^H(n_k) \tag{11}$$

By substituting the sample correlation matrix from Eq. (11) into Eq. (9), we obtain the following MVDR beamformer, as described in [14].

$$\mathbf{w} = \frac{\hat{\mathbf{R}}_{i+n}^{-1}\mathbf{v}(\theta)}{\mathbf{v}^H(\theta)\hat{\mathbf{R}}_{i+n}^{-1}\mathbf{v}(\theta)} \tag{12}$$

## 3. Hardware implementation

The evaluation and validation of the MVDR beamforming were conducted utilizing the Teledyne e2V Semiconductors' evaluation board, EV12AQ60X-ADX-EVM, depicted in Fig. 1. This board incorporates the high-speed ADC EV12AQ600, a 12-bit quad-channel converter, and the high-performance FPGA Xilinx Kintex UltraScale XCKU085.

Featuring four edge-mounted SMA connectors corresponding to four analog input channels for capturing RF signals, the board employs the PLL device LMX2592 for clocking and initializing the ADC/FPGA interface, as well as the FPGA data path.

The EVM provides three modes of operation contingent on the configured interleaving functionality: one-channel mode, two-channel mode, and four-channel mode. Internal registers facilitate the modification of various settings, including ADC functionality, LMX2592 PLL settings, channel modes, clock configurations, and acquisition settings. Teledyne e2V Semiconductors further supports the EVM with the graphical application ADCaptureLab, streamlining control over configuration,

triggering data acquisition, and visualization of data in time and frequency domains. This application also enables the assessment of performance metrics such as SNR, SNDR, SFDR, and THD [23].

The high-speed ADC and the powerful FPGA enable the system to handle large volumes of data with high precision. This setup ensures that the DBF system can perform real-time processing of RF signals, achieving high SNR, SNDR, SFDR, and low THD.

For a comprehensive description of the hardware platform used in this work please refer to [10]. As part of the description of the Digital Beamforming Receiver architecture in [10], a detailed overview of the utilized hardware is provided. In this section, a brief description and analysis of each component is included to aid in the context of the article.

It is noteworthy that this study exclusively focuses on evaluating the digital processing tasks within the FPGA as part of the MVDR design and implementation. The ADC device and ESIstream high-speed serial link protocol, although integral components of the EVM, are not utilized in this specific context but will be described for comprehensive understanding.

### 3.1. EV12AQ600 analog to digital converter

The sampling process for input RF signals on the evaluation board EV12AQ60X-ADX-EVM utilizes the EV12AQ600 ADC device. This ADC is designed to meet the stringent standards of both the European Space Components Coordination (ESCC) and the QML-Y space requirements. With its quad channels and 12-bit resolution, the ADC operates at an impressive sampling rate of 1.6 GSps. Notably, the ADC supports multi-mode operation, enabling the interleaving of its four independent cores to achieve higher sampling rates. In the 1-channel operating mode, the ADC can reach sampling rates of up to 6.4 GSps [24].

The characterization of the ADC was conducted by measuring its response to input tones spaced at 0.5 GHz across a range from 1 to 20 GHz [10]. The obtained analog equivalent bandwidth shows a significant signal attenuation starting around 5.5 GHz. This indicates that the ADC effectively processes radio frequency signals in the S band and part of the C band up to 5.5 GHz, while signals beyond this frequency experience considerable attenuation.

The high-speed sampling rates of this ADC, along with its characterization study results, performance metrics [10], and space-qualified conditions, make it ideal for the objectives of this work. As discussed in a later section, the ADC device exhibits substantial power consumption. This factor should be carefully considered when scaling this design to accommodate additional RF input channels and integrating more ADC devices.

### 3.2. ESIstream high-speed serial link protocol

The EV12AQ600 generates digital data, which is efficiently streamed to the FPGA through a serial communication link utilizing the ESIstream protocol [25]. Developed by Teledyne e2v Semiconductor, this license-free protocol is designed to facilitate seamless serial communication between high-speed data converters and FPGAs. The ESIstream system involves the transmitter (TX) ADC, the receiver (RX) — in this implementation, the FPGA — and features 8 lanes for transmitting data frames from each ADC core (with data outputted on 2 serial links per ADC core). Additionally, a synchronization signal is employed to initialize communication.

The protocol employs a 14b/16b encoding scheme, providing an efficiency of 87.5%, surpassing the 80% efficiency of the 8b/10b encoding. ESIstream supports deterministic and low link latency, a lane rate of up to 12.8 Gbps, multi-serial lane synchronization, and multi-device synchronization for use with various transmission and receiver modules. To facilitate the setup and implementation of the TX and RX ESIstream modules, Teledyne provides an IP core for Xilinx Vivado [26]. It's crucial to note that this IP core is platform-dependent and exclusively functions on the targeted FPGA.

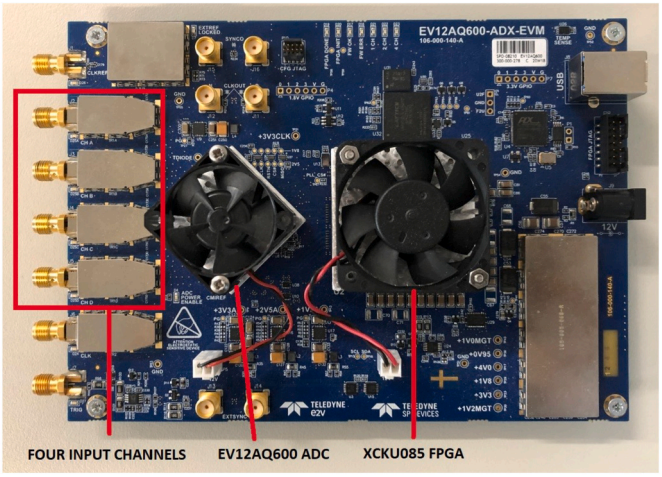



**Fig. 1.** EV12AQ60X-ADX-EVM evaluation board.

Currently, diverse ESIstream IP cores are available for some of the most widely adopted space-grade SerDes in the industry, encompassing the Xilinx XCKU60, Xilinx Virtex 5QV, Microsemi Polar Fire, and Microsemi RTG4.

Upon implementation within the FPGA, the ESIstream IP core's RX module constructs a data framework that delivers eight parallel samples for each channel, with each sample being 12 bits in length, operating at a clock frequency of 200 MHz. This configuration is meticulously aligned with the output specifications of the ADC, which samples each input channel at a rate of 1.6 GSps, producing 12-bit data words. This synchronicity ensures that the high-speed data transmission from the ADC to the FPGA is managed efficiently, with precision and minimal latency.

The license-free protocol specification, its straightforward implementation on the evaluation board, high-speed rates, multi-lane synchronization, and its compatibility with various platforms, including the space-qualified XCKU060, make it ideal for the Digital Beamforming Receiver system.

### 3.3. Xilinx Kintex UltraScale XCKU085 FPGA

The evaluation board EV12AQ60X-ADX-EVM incorporates the Xilinx Kintex UltraScale XCKU085 FPGA, a high-performance device that leverages both monolithic and next-generation stacked silicon interconnect technology. This enables the FPGA to meet the demanding requirements of high-performance digital signal processing and facilitates seamless data transmission and reception through high-speed serial communication links [27].

Equipped with 48 high-speed serial transceivers, known as GTH transceivers, the FPGA utilizes 8 of these for receiving data sampled via the ESIstream protocol. The GTH transceivers support line rates ranging from 500 Mb/s to 16.375 Gb/s, with a valid data lane capacity of up to 12.5 Gbps. Organized in groups of four, adhering to the CML interface standard, each transceiver is capable of handling the data rate generated by the ADC.

Implementation of the transceivers in the hardware design is facilitated through the UltraScale FPGAs Transceivers Wizard—an application provided by Xilinx. This tool allows for the automatic generation of the XDC constraint file, which can be further edited to customize operational parameters and placement information for the application.

Digital signal processing operations within the FPGA make use of dedicated resources known as DSP slices or DSP48E2, as designated by the manufacturer. Proper allocation of these resources during the hardware design development is crucial to enhance the performance of

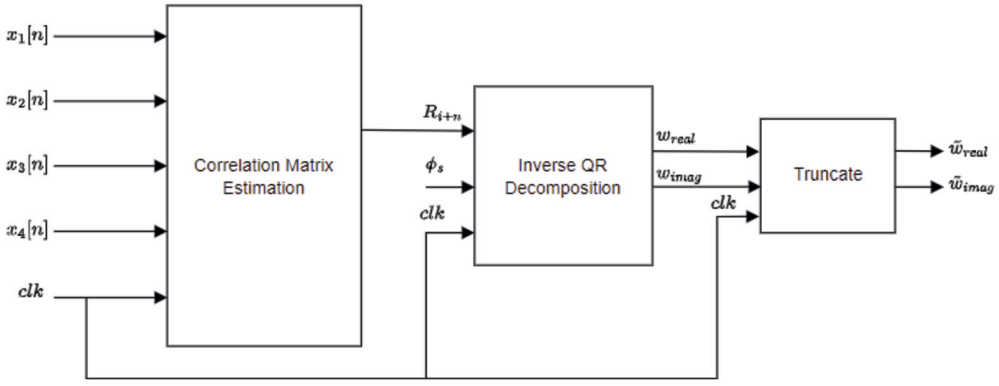



**Fig. 2.** Adaptive Beamforming elements.

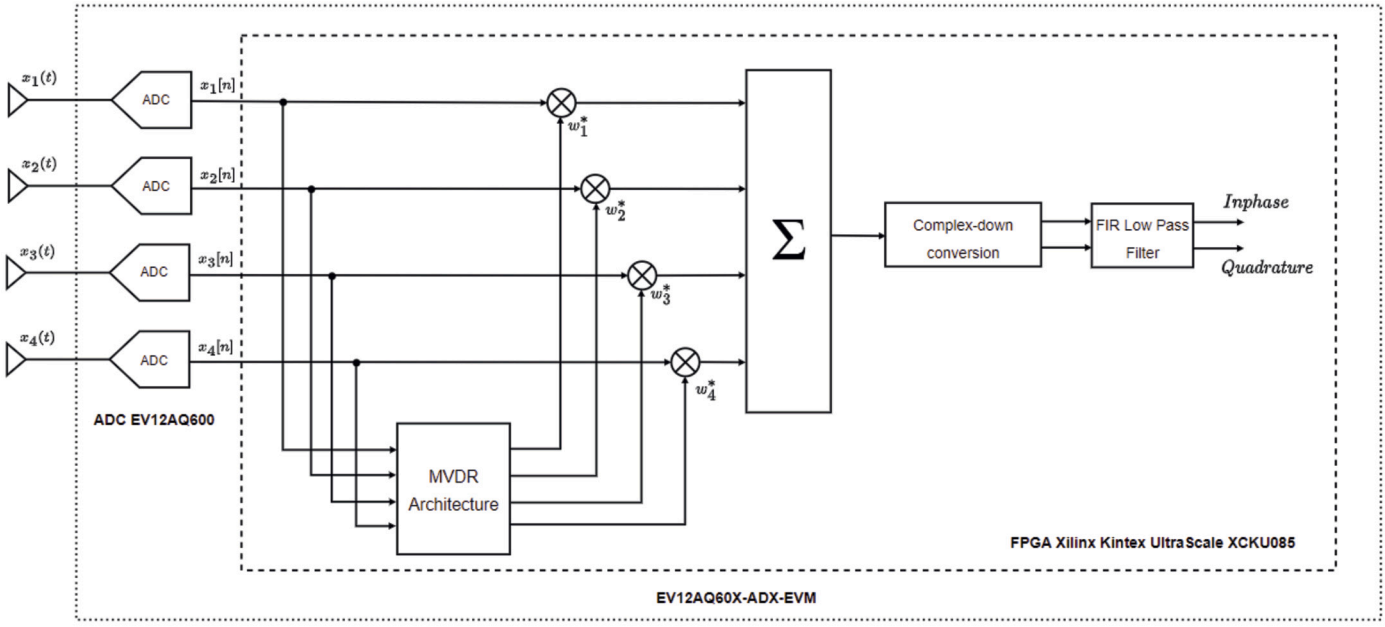



**Fig. 3.** General Adaptive Beamforming schema.

operations in terms of speed and power consumption [28]. The number of DSP slices within the FPGA is 4100, which is enough for the operations performed in the design.

While the FPGA device used in this work is not space-qualified, there exists another FPGA in the same family—the Xilinx Kintex UltraScale XCKU060—that is space-qualified. It shares the same architecture and resource types as the XCKU085, albeit in slightly fewer numbers, which remains sufficient for the implementation of this work.

For the implementation of the MVDR beamforming module, digital signal processing tasks were designed and executed in the XCKU085 FPGA using VHDL as the Hardware Description Language, facilitated by the Xilinx Vivado 2020.2 (64-bit) software. The FPGA was programmed with the bitstream generated through the JTAG connection. Results, including the baseband filtered beam, are stored in an onboard RAM module within the FPGA. To visualize the recovered baseband signals, samples are transmitted from the RAM memory to a PC via the UART interface.

## 4. Adaptive beamforming MVDR design

The implemented hardware design in this study executes the MVDR algorithm to compute the optimal weights. This design, based mathematically on the Eq. (12), primarily consists of three sequentially executed stages: calculation of the correlation matrix, inverse QR Decomposition, and weights rescale process. Fig. 2 provides an illustration of the various elements constituting the MVDR algorithm executed in the FPGA. These calculated weights are subsequently transmitted to the Digital Beamforming (DBF) stage, where they are applied in the corresponding linear combination, complex down-conversion, and low-pass filtering operations. Fig. 3 illustrates the architecture of the Digital Beamforming Receiver in conjunction with the MVDR algorithm. [10] provides a comprehensive description of the DBF architecture used, including full details of the actual processing flow. Additionally, [29] proposes a similar DBF design approach focused on SAR. Notably, the beamformer stage precedes the complex down-conversion, resulting in input samples received at this point being exclusively real and not complex. This configuration provides the advantage that the adaptive algorithm processes only real samples, simplifying the computation of the correlation matrix and its inverse.

In this design, the system takes as inputs the digitized samples originating from the EV12AQ600 ADC in the four-antenna array receiver. These samples, each 12 bits long, are generated at a rate of 1.6 GSps. The resulting information is then streamed to the XCKU085 FPGA through the high-speed serial link (HSSL) ESIstream protocol. This data is transmitted to the FPGA in the form of 8 samples in parallel per channel, each being 12 bits long and operating at a frequency of 200 MHz. It is important to note that the samples from the ADC are real-type values, whereas the steering vector coefficients of the desired signal are

complex-type values. This distinction informs the design of the IQRD systolic array structure, featuring different processing elements for real and complex input values. This departure from the conventional design, where all processing elements are of the same data type [11], is a deliberate choice aimed at optimizing resource allocation in the implementation, albeit introducing complexity to the VHDL design.

Arithmetic operations within the MVDR algorithm on the FPGA are executed using DSP48E2 resources. These resources enhance digital signal processing tasks by reducing power consumption and expediting adders and multipliers, especially in multiply-accumulate (MAC) operations. Consequently, during the development of the implementation, we allocate the maximum number of DSP slices to accommodate the operations, thereby optimizing performance using this technology. The DSP48E2 slices are automatically inferred from the VHDL code for the arithmetic functions during the application of the synthesis tool. Alternatively, it is possible to explicitly instantiate the DSP48E primitive from Xilinx to gain direct access to the DSP [28].

It is essential to emphasize that the calculation of the Direction Of Arrival (DoA) of the desired signal is beyond the scope of this work, and therefore, it is assumed to be known. Various approaches, as outlined in [30], [31], and [32], may be employed to estimate the DoA. The implementation of Direction of Arrival (DoA) estimation for real-time systems on an FPGA has been extensively covered in the literature [33], [34]. These studies demonstrate significant resource utilization and computational load, which could impact the FPGA used in this work. Consequently, a larger FPGA could be considered, or the DoA estimation could be implemented on a separate FPGA working synchronously with the proposed one. Since some DoA estimation techniques involve QR Decomposition, it is expected that a similar approach could be used as the one performed in this work. Also, the potential for implementing pulse compression at the IF in SAR applications and integrating it with DoA estimation requires further exploration. Additionally, this work does not consider the performance implications of any mismatch between the actual desired signal and the assumed signal model used by the adaptive beamformer [35].

### 4.1. Correlation matrix estimation

This procedure executes the operation expressed in Eq. (11) to estimate the correlation matrix. Operating through an on-the-fly process, each clock cycle involves the multiplication of digitized input samples from each channel with their corresponding conjugate sample, followed by summation with the previous multiplication result. In the subsequent clock cycle, a new multiplication occurs with the incoming samples, and this process iterates for the $K$ samples considered in the snapshot. Real-time calculation of the matrix correlation is achieved as each new sample is processed until completion of the $K$ samples. This approach eliminates the need to store all $K$ samples from the input channels before initiating the calculation of the correlation matrix, leading to a more resource-efficient method.

Given the Hermitian property of the correlation matrix, only the upper triangle values need calculation. The lower triangle values are the complex conjugates of the upper triangle, necessitating a straightforward data reordering and sign change in the imaginary part. However, as the samples from the ADC and streamed by the ESIstream are real-type values with no imaginary part, the operation on the complex conjugate is unnecessary. Each component of the correlation matrix can be expressed as shown in Eq. (13).

$$R_{ij} = \sum_{s=0}^{K-1} x_i(n-s)x_j^*(n-s) \tag{13}$$

Fig. 4 provides a visualization of the data flow during the estimation of the correlation matrix for a 4-channel input system. Each cell value of the matrix is stored in its corresponding register.

The data flow from the ESIstream into the correlation matrix process consists of 8 samples in parallel per channel, each being 12 bits long,

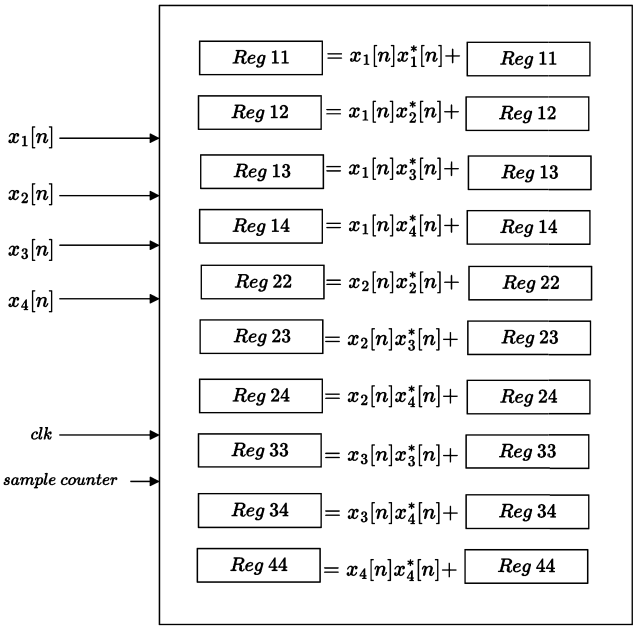



**Fig. 4.** Correlation Matrix data flow process.

**Table 1**
Correlation Matrix Calculation Parameters.

| Parameter | Value |
|---|---|
| Clock frequency | 200 MHz |
| Input sample length | 12 bits |
| Maximum snapshot size (K) | $2^{13}$ |
| Output sample length | 32 bits |
| Latency (clock cycles) | K/8 + 3 |
| Number of channels | 4 |

with a clock frequency of 200 MHz. Due to the parallel transfer of 8 samples per channel at 200 MHz, the multiplication process is replicated 8 times per channel and matrix element. A final adder operation is then applied to sum the results of the 8 multiplications. Being $K$ a power-of-2 value, the division operation $1/K$ is efficiently executed through a right bit-shift after the previously described multiplication and summation operations, obviating the need for an integer division operation. For a maximum snapshot number of $K = 2^{13}$ samples, the calculated values of the correlation matrix needs to be represented with 32 bits. This precision is essential to accurately represent the numbers generated after the multiplication and summation operations. A larger number of samples considered in the snapshot would necessitate more bits to faithfully represent the outcome of the correlation matrix. Also, the processing time of the correlation matrix calculation will depend on the number of samples considered in the snapshot. The number of clock cycles to perform this calculation will be the number of samples divided by 8 (as input data is transferred in sets of 8 samples in parallel), plus 3 extra cycles to perform the last multiplication and summation operations. Table 1 shows a summary of correlation matrix parameters.

Consequently, the output of the correlation matrix estimation process is an array of 4x4 (a square matrix of order equal to the number of input channels) samples, each being 32 bits long.

### 4.2. Inverse QR decomposition (IQRD-RLS) algorithm

As indicated in Eq. (12), obtaining optimal beamforming weights requires the inversion of the correlation matrix. The Gaussian elimination method, traditionally utilized for matrix inversion, presents computational challenges on an FPGA due to its $O(n^3)$ complexity. Moreover,

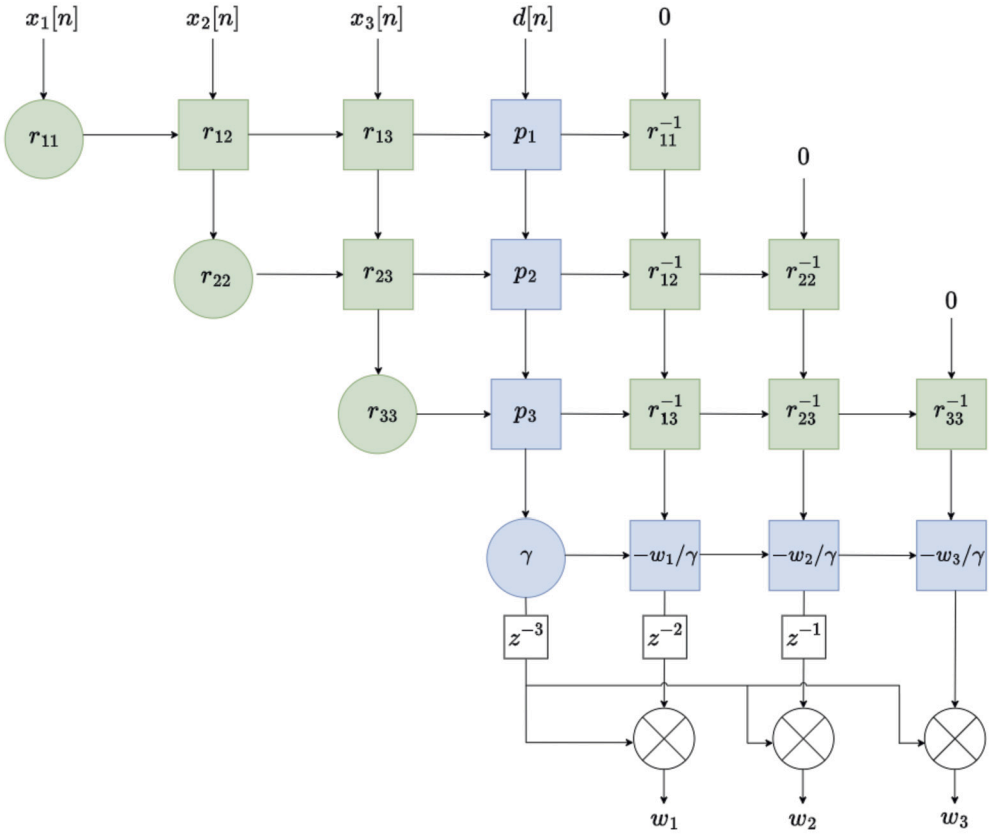



**Fig. 5.** Inverse QR Decomposition Systolic Array.

it exhibits poor numerical robustness and stability for fixed-point values [36]. In this study, we deploy the Inverse QR Decomposition algorithm (IQRD-RLS) to compute the optimal beamforming weights [37]. This algorithm is a variant of the Recursive Least Square algorithm (QRD-RLS) and leverages the QR Decomposition (QRD) technique for pseudo-inverting the correlation matrix. The IQRD-RLS algorithm directly extracts the values of the required weights from the correlation matrix and the steering vector values. It addresses the numerical stability issues associated with the direct matrix inversion method [38]. Additionally, it offers the flexibility to be implemented as a systolic array architecture using the Givens rotation method [39], leveraging the inherent parallelism in digital architectures.

From this point, the proposed design for the systolic array of the IQRD system will be explained in detail. The theory of QRD-RLS, IQRD-RLS and its systolic array structure can be found in detail in the suggested literature [40], [41], [38].

#### 4.2.1. Systolic array structure

In general, the systolic array process incorporates two distinct types of processing elements: Boundary Cells (BC) and Internal Cells (IC) [42], [43], [44]. These cells execute Givens Rotations on each element of the input correlation matrix and the steering vector. The Boundary Cells, depicted by circles in the systolic array architecture shown in Fig. 5, take the input formed by the current and previous input data, performing the vectoring operation and annihilating the lower element. On the other hand, the Internal Cells, represented by squares in the systolic array architecture, execute the rotation operation on the input values using rotation angles calculated from the corresponding Boundary Cell row. Data within the systolic array structure moves between cells from top to down and left to right. Communication between cells is facilitated through a handshaking protocol using ready/valid signals.

The input samples $x[n]$ and $d[n]$ from the correlation matrix $R_{xx}$ and steering vector, respectively, are fed into the systolic array structure in a staggered manner. Cells with the symbol $r$ store the elements of the upper triangular matrix from the QR Decomposition, while cells with the symbol $p$ store the result of the multiplication of the orthogonal $Q$ matrix by the steering vector $d$. The cell with the $\gamma$ symbol stores the likelihood factor. A crucial addition to this group of cells is the inverse QR update for weight extraction. Therefore, cells with the symbol $r^{-1}$ store the inverse of the upper triangular matrix $r$. The bottom line of the lower triangular matrix $r^{-1}$ stores the scaled weight vector $-\mathbf{w}/\gamma$. The optimum beamforming weights are extracted by multiplying them by the scaling factor $\gamma$. Each cell in the systolic array structure stores values represented by 32 bits. This 32 bits precision is needed to maintain the values inside their range through the systolic array structure. The final weight values calculated are 64 bits long, which are then truncated to be 12 bits long in a subsequent stage. Table 2 shows a summary of the main parameters from the systolic array structure defined, where $N$ is the number of channels and $PE$ is the latency in clock cycles of the processing element.

**Table 2**
Systolic Array Structure Parameters.

| Parameter | Value |
|---|---|
| Clock frequency | 200 MHz |
| Cell precision | 32 bits |
| Output weight precision | 64 bits |
| Latency (clock cycles) | $(4 \cdot N + 2) \cdot PE$ |
| Number of channels | 4 |

One advantage of the systolic array is its regularity and locality [45]. This reduction in signal path length and delay results in an increase in possible execution speed. A consequence is that inputs at the top of the array must be delayed by different amounts.

In this work, as the input values from the correlation matrix and the steering vector from the desired signal are real and complex, respectively, different types of boundary cells and internal cells are also

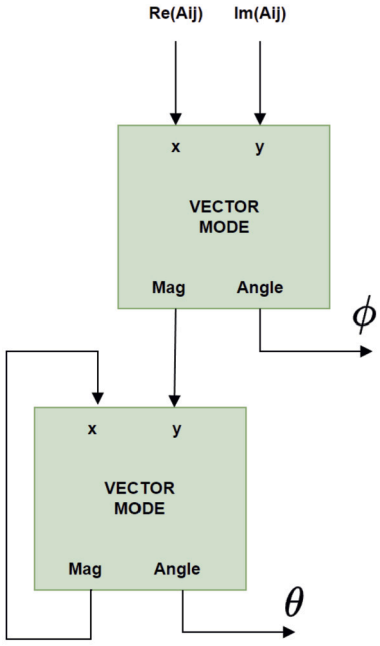



**Fig. 6.** Complex Boundary Cell schema.

considered, mixing different types of processing cells depending on the type of values they read. This approach helps to reduce the resources allocated to the systolic array, saving area and power consumption, in exchange for increasing the complexity of the VHDL design. Therefore, two types of boundary cells and two types of internal cells are considered in this implementation. Although the boundary and internal cells from the upper triangular matrix and the lower triangular matrix are not exactly the same (down-dating cells use $1/\lambda$ as a forgetting factor in the internal operation), for our purposes moving forward, they will be treated as identical.

From the systolic array architecture in Fig. 5, the cells with symbols $p$, $\gamma$, and $-\mathbf{w}/\gamma$ process complex data. Hence, the boundary and internal cells are of a complex type (cells in blue). The rest of the cells in the systolic array architecture process only real data, so they are boundary and internal cells of a real type (cells in green).

#### 4.2.2. Complex boundary and internal cells

The Boundary cell, responsible for executing the vectoring operation on complex input values, adheres to the functionality described in Algorithm 1, [46].

**Algorithm 1** Boundary cell: vectoring operation.

**if** $xin = 0$ ($includes\ the\ case\ r = xin = 0$) **then**
  $c = 1$
  $s = 0$
  $r_n = r$
**else if** $r = 0$ ($xin\ must\ be\ nonzero$) **then**
  $c = 0$
  $s = sign(xin^*)$
  $r_n = |xin|$
**else**($r\ and\ xin\ both\ nonzero$)
  $c = |r|/\sqrt{|r|^2 + |xin|^2}$
  $s = sign(r)xin^*/\sqrt{|r|^2 + |xin|^2}$
  $r_n = sign(r)/\sqrt{|r|^2 + |xin|^2}$
**end if**

Considering $xin$ as the complex input value, where $r$ is the previously calculated value, $r_n$ is the next calculated value, $c$ is $cos\theta$, and $s$ is $sin\theta$, with $\theta$ representing the angle used to eliminate the element from the input vector by aligning it with the axis.

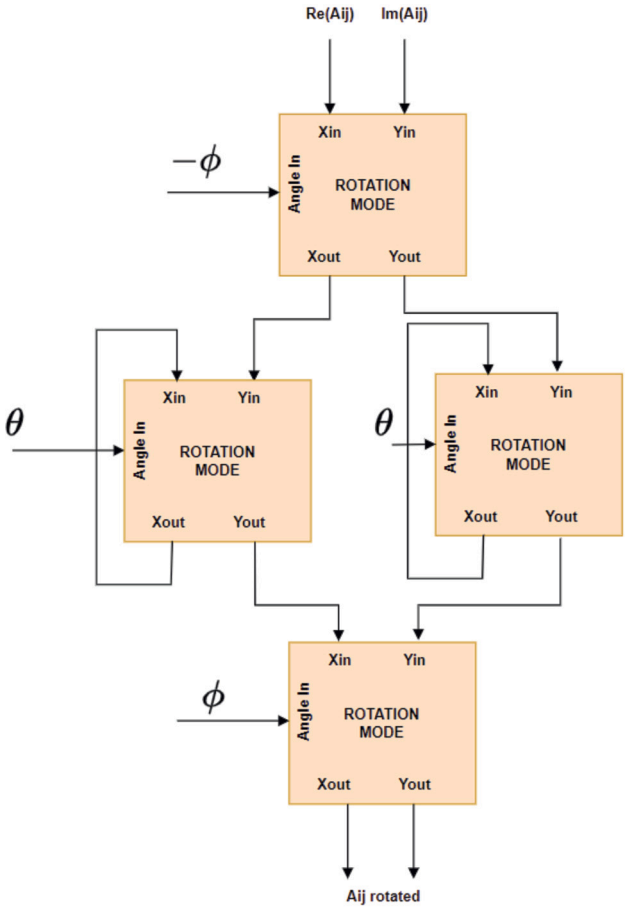



**Fig. 7.** Complex Internal Cell schema.

The Internal cell, responsible for executing the rotation operation on complex input values, adheres to the expression shown in Eq. (14).

$$\begin{aligned} xout &= c\ xin - s^*\ \lambda^{1/2} r \\ r_n &= s\ xin + c\ \lambda^{1/2} r \end{aligned} \tag{14}$$

Where $\lambda$ represents the forgetting factor, set to a value close to 1 for our MVDR application [12]. $x_{\text{out}}$ denotes the new vector rotated by the angle $\theta$.

To execute the processing tasks required by the boundary and internal cells, the CORDIC algorithm is employed. The calculation of the angle formed by the vector in the boundary cell is conducted by the CORDIC in vectoring mode, eliminating an element from the input vector by aligning it with the axis. The calculated angle value is then passed to the internal cell, which subsequently rotates its input vector. The internal cell operation is performed by the CORDIC in rotation mode. This algorithm rotates the angle passed along by the boundary cell. The utilization of the CORDIC algorithm replaces the square root and multiplication operations needed for vector rotation and angle calculation with shift and add arithmetic operations. This, in turn, simplifies the resources, complexity, and power consumption in the FPGA. Figs. 6 and 7 schematically represent the functionalities of the Complex Boundary and Internal Cells, which process complex values. The Complex Boundary Cell comprises two CORDIC vectoring modules. One calculates the angle $\phi$ formed by the input complex value, and the other calculates the angle $\theta$ formed by the current input value and the previous value. Both angle values, $\phi$ and $\theta$, are then passed along to the internal cell. The Complex Internal Cell consists of four CORDIC rotation modules, which output the rotated incoming input complex values. The same $\phi$ and $\theta$ angles are passed along to the internal cell.

**Table 3**
Complex boundary cell resource utilization.

| Resource | Utilization | Available | Utilization % |
|---|---|---|---|
| CLB LUTs | 1495 | 497520 | 0.30 |
| CLB Registers | 780 | 995040 | 0.08 |
| CARRY8 | 98 | 75060 | 0.13 |
| F7 Muxes | 0 | 300240 | 0 |
| F8 Muxes | 0 | 150120 | 0 |
| DSPs | 8 | 4100 | 0.2 |

**Table 4**
Complex internal cell resource utilization.

| Resource | Utilization | Available | Utilization % |
|---|---|---|---|
| CLB LUTs | 3151 | 497520 | 0.63 |
| CLB Registers | 1393 | 995040 | 0.14 |
| CARRY8 | 168 | 75060 | 0.22 |
| F7 Muxes | 0 | 300240 | 0 |
| F8 Muxes | 0 | 150120 | 0 |
| DSPs | 32 | 4100 | 0.78 |

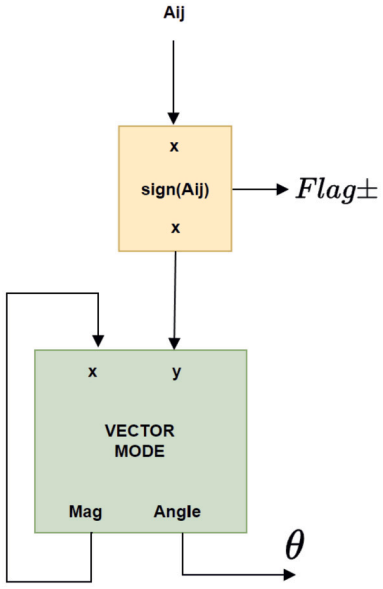



**Fig. 8.** Real Boundary Cell schema.

**Table 5**
Real boundary cell resource utilization.

| Resource | Utilization | Available | Utilization % |
|---|---|---|---|
| CLB LUTs | 722 | 497520 | 0.15 |
| CLB Registers | 439 | 995040 | 0.04 |
| CARRY8 | 59 | 75060 | 0.08 |
| F7 Muxes | 0 | 300240 | 0 |
| F8 Muxes | 0 | 150120 | 0 |
| DSPs | 4 | 4100 | 0.1 |

The FPGA resource utilization for implementing the Complex Boundary Cell and Complex Internal Cell is presented in Table 3 and Table 4, respectively.

### 4.2.3. Real boundary and internal cells

The Boundary and Internal Cells, designed for processing real data, are simpler as the $\phi$ angle is no longer required. Figs. 8 and 9 schematically illustrate the functionalities of the Boundary and Internal Cells for processing real values. The Real Boundary Cell incorporates one CORDIC vectoring module and a process that detects the sign of the input value, indicating whether it is positive or negative. This process

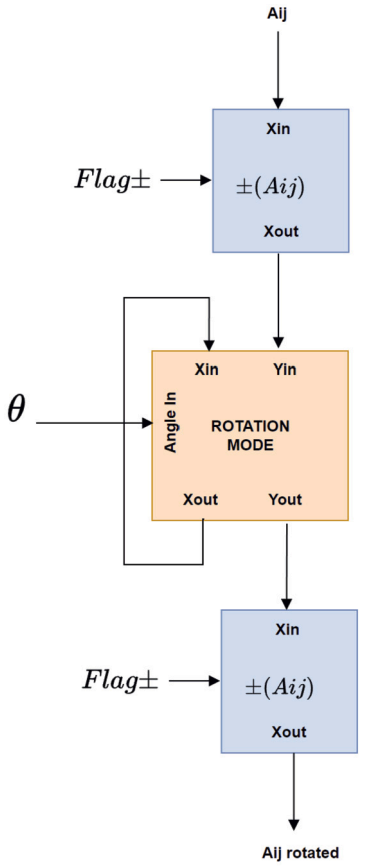



**Fig. 9.** Real Internal Cell schema.

**Table 6**
Real internal cell resource utilization.

| Resource | Utilization | Available | Utilization % |
|---|---|---|---|
| CLB LUTs | 857 | 497520 | 0.17 |
| CLB Registers | 513 | 995040 | 0.05 |
| CARRY8 | 52 | 75060 | 0.07 |
| F7 Muxes | 0 | 300240 | 0 |
| F8 Muxes | 0 | 150120 | 0 |
| DSPs | 8 | 4100 | 0.19 |

replaces the first CORDIC vectoring module in the complex cell scenario. Similar to the Complex Boundary Cell, the CORDIC vectoring module calculates the angle $\theta$ formed by the current input value and the previous value. The values of both the flag and the $\theta$ angle are then passed along to the internal cell. The Real Internal Cell comprises one CORDIC rotation module and two processes to set the sign of the input values. In this case, the processes that set the sign of the input values replace the corresponding CORDIC rotation modules that acted over the angle $\phi$ in the Complex Internal Cell. Additionally, only one CORDIC rotation module is needed to act over the angle $\theta$ instead of two in the Complex counterpart, representing a considerable reduction in resources. The same flag and $\theta$ angle values are passed along to the internal cells.

The FPGA resources required to implement the Real Boundary cell and Real Internal cell are detailed in Table 5 and 6, respectively. As deduced from the values in the tables, utilizing different types of cells and taking advantage of the Digital Beamforming Receiver architecture significantly reduces the number of resources and power consumption in the final design. In a later section, the total number of resources allocated in the proposed design in this work will be detailed, along with a comparison with the more standard version of the design with all cells of the same type.

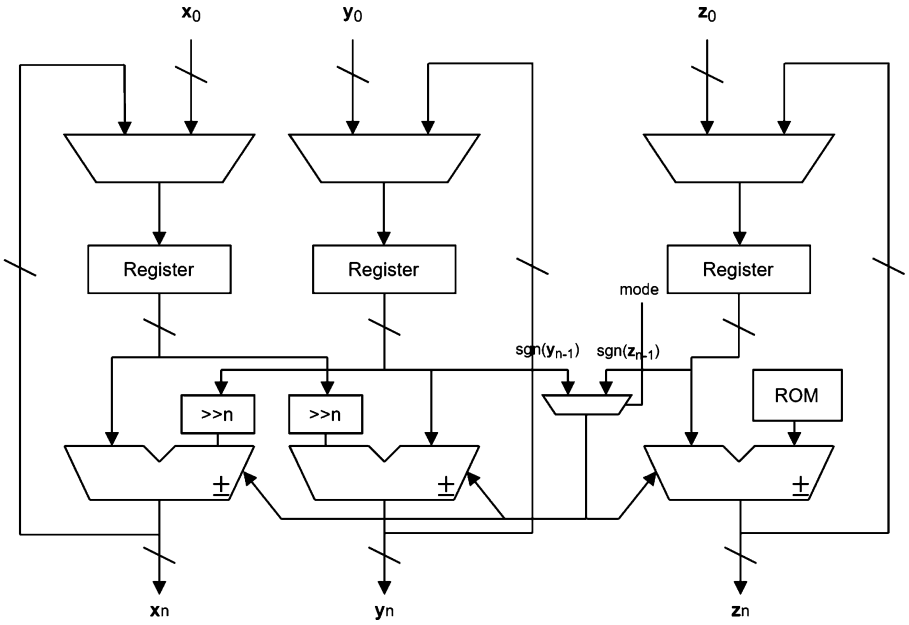



**Fig. 10.** CORDIC iterative architecture.

**Table 7**
CORDIC Algorithm Parameters.

| Parameter | Value |
|---|---|
| Clock frequency | 200 MHz |
| Input sample length | 32 bits |
| Number of iterations | 30 |
| Latency (clock cycles) | 33 |
| Output sample length | 32 bits |

#### 4.2.4. CORDIC architecture

In this study, the iterative architecture type of the CORDIC algorithm is implemented [47]. Fig. 10 provides a schematic representation of this architecture. In this design, the output and input of the vector components and angle accumulator ($X$, $Y$, and $Z$ components in the schema) are connected, enabling the reuse of multiplexers, registers, shifter units, and adders components on every iteration. This configuration results in an efficient utilization of resources. However, it requires a control process through a state machine to manage the multiplexers, the number of bits to shift, and the address of the precalculated constants. Despite the increased complexity of the control process, this architecture configuration saves approximately 65% of the FPGA resources compared to the unrolled architecture type [47]. The unrolled architecture, while advantageous for its combinational nature and reduced complexity, consumes a significantly larger area in its implementation. Given the goal of conserving resources in the FPGA and considering that the iterative architecture meets timing conditions, it is the chosen architecture in this work.

As discussed previously, the length of the input vector components and angle are 32 bits. The number of iterations performed both in rotation and vectoring mode are 30, which ensures enough precision at the output. For this number of iterations, the latency to resolve the rotation and vectorization are 33 clock cycles. The Table 7 shows the CORDIC parameters applied.

### 4.3. Weights rescale process

The weight values generated from the Inverse QR Decomposition are 64 bits long due to the multiplication process operating over the 32-bit long samples generated by the processing elements. However, the beamforming weights need to be 12 bits long as per the requirement. Therefore, a dynamic truncation process is executed before the weights become valid for the Digital Beamforming process. This truncation process identifies the most significant set bit position from the four groups of coefficients (real and imaginary components) and collects the following significant 12 bits from the calculated position. Fig. 11 illustrates a schematic representation of this process. Consequently, a dynamic truncation process is performed based on the calculated values of the weights each time.

## 5. Validation and implementation analysis

In this section, the proposed MVDR design implemented on the FPGA is validated by comparing the beampatterns obtained from the optimal weights calculated, with those obtained from the floating-point MATLAB simulation. Then, resource utilization, computational load, processing time and energy consumption are analyzed to demonstrate the advantages of the proposed design.

### 5.1. Validation

To test the hardware design and implementation of the MVDR algorithm on the FPGA, specific synthetic RF input signals are applied to the system. The fixed-point output data from the implementation is then compared against a floating-point MVDR beamformer running under MATLAB. This comparison ensures that the processed information achieves the expected SINR level improvement and maintains signal quality. This validation process has been applied using 4, 8 and 16 receiving elements. Although the number of channels available in the evaluation board are 4, the design has been extrapolated to 8 and 16 channels inside the FPGA, being the input signals stored in the FPGA's internal RAM memory. For debugging purposes and to retrieve the processed data from the DBF receiver, additional hardware modules have been developed and integrated into the system. This enhancement facilitates the debugging process and enables the examination of the output data generated.

To simulate various angles of arrival for the desired and interference signal waveforms into the array system, a MATLAB script generates digital signals (one per channel) with the corresponding time delays for each input channel. Subsequently, these digital signals are stored in the FPGA's RAM memory blocks, emulating the input digital signals sam-

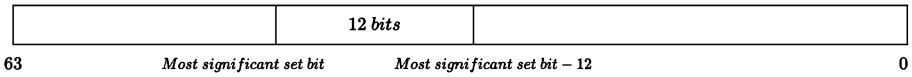



**Fig. 11.** Truncation process.

pled by the ADC cores. These tests validate the anticipated behavior of the MVDR algorithm.

The communication between the EVM and the PC, responsible for triggering the input signal waveforms and storing and retrieving the outcomes, is facilitated through Python scripts. These scripts also handle the activation and configuration of the EV12AQ600 ADC and the LMX2592 PLL, ensuring the appropriate channel mode and sampling frequency are set.

In the first test scenario, for a 4-channel system, it is considered a desired signal arriving at 35º perpendicular to the array with a power of 20 dB and an interference signal arriving at 20º with a power of 30 dB. In a second test scenario with 8 channels, the desired signal arrives at 15º with 20 dB, and three interference signals impinging on the array from angles of 30º, 50º, and -30º, respectively, all with a power of 30 dB. In a third test scenario, 16 channels are considered in the system, with the same desired and interference signals than the second scenario. A thermal noise level of 0 dB is added also to all the test cases. Fig. 12 illustrates the spectral information from the desired signal, which is a linear FM modulated signal transmitted at 3.6 GHz. The signal is translated to the 1st Nyquist Zone (400 MHz) with a base signal tone of 1 MHz and a deviation frequency of ±100 MHz. Fig. 13 illustrates the spectrum of the interference input signal, while Fig. 14 depicts the spectral information from the resulting input signal. This is the summation of the desired signal, interference signal, and thermal noise. This incoming interference is expected to be stationary during the process of reading input samples. The processing time to calculate the updated weights determine changing conditions of the input signals. In a later subsection is analyzed the processing time.

For this validation, simplicity dictates that the correlation matrix considers only interference plus noise data, omitting the desired signal. This is intended to prevent a substantial degradation of the SINR in the event of mismatches between the actual desired signal and the signal assumed by the beamformer model. In certain applications, such as some communication systems and radar setups, the presence of the desired signal is controlled, typically initiated by the system itself. In scenarios involving jamming, a common occurrence in both communication systems and radar applications, it is feasible to abstain from transmitting for a specific duration to gather data for estimating the interference-plus-noise covariance matrix $\mathbf{R}_{i+n}$. This form of training is often referred to as 'listen-only' [14]. Alternatively, other applications may employ specific algorithms to extract the desired signal information from the correlation matrix data [48], [49].

Fig. 15, 16 and 17 displays the resultant beampatterns for the different $N$ channels system. The solid and dashed vertical lines represent angles of the desired and interference signals, respectively. The dashed blue line represents the beampattern calculated using MATLAB with floating-point precision, and the solid red line represent the beampattern calculated from the FPGA implementation. It can be seen for all cases there is no distinguishable difference between the FPGA implementation with the design proposed and the MATLAB implementation. The beampatterns attenuates the interference signals and maintain the desired signals.

Applying the beamforming process with the generated optimized weights, Fig. 18 illustrates the resulting spectrum signal (prior to the demodulation and low-pass filtering process). The optimized weights calculated in the implementation effectively eliminate the interference signal presented in the input RF signal.

Figs. 19 and 20 illustrate the corresponding in-phase and quadrature signals generated at the output of the 4-channel Digital Beamforming Receiver system, including the beamformer, complex down-conversion and filtering process.

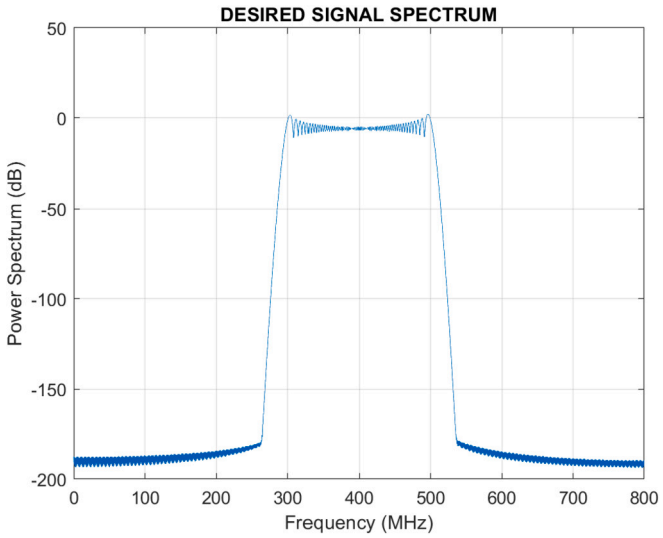



**Fig. 12.** Desired Signal Spectrum.

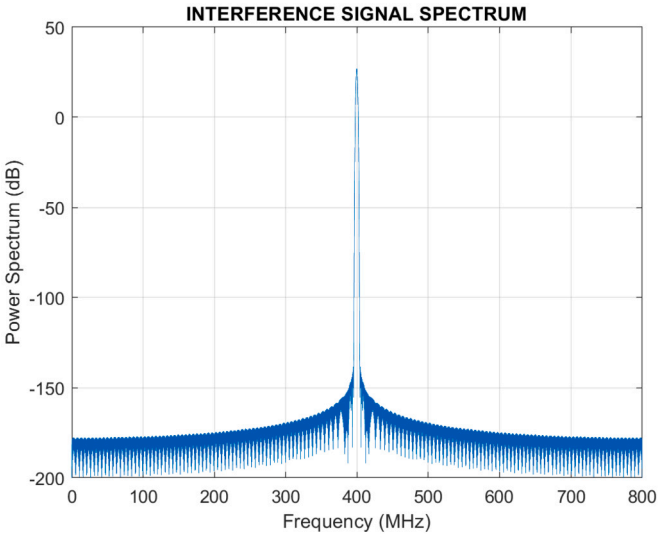



**Fig. 13.** Interference Signal Spectrum.

### 5.2. Resource utilization

The resource allocation for implementing the correlation matrix and systolic array structure described in this work, for a 4 array element, is presented in Table 8 and 9. The resources allocated by the weights rescale process described in the previous section are not analyzed as its impact is minor in the FPGA. The overall resource allocation for the adaptive algorithm design in the FPGA, including matrix correlation calculation, systolic array structure, and weights rescaling process, is outlined in Table 10.

The resource allocation for implementing the correlation matrix and systolic array structure in the FPGA using the standard design, for a 4 array element, are presented in Table 11 and 12. The overall resource allocation for the adaptive algorithm using the standard architecture is shown in Table 13. A comparison between the proposed MVDR design and the standard one is depicted in Fig. 21. Remarkably, the

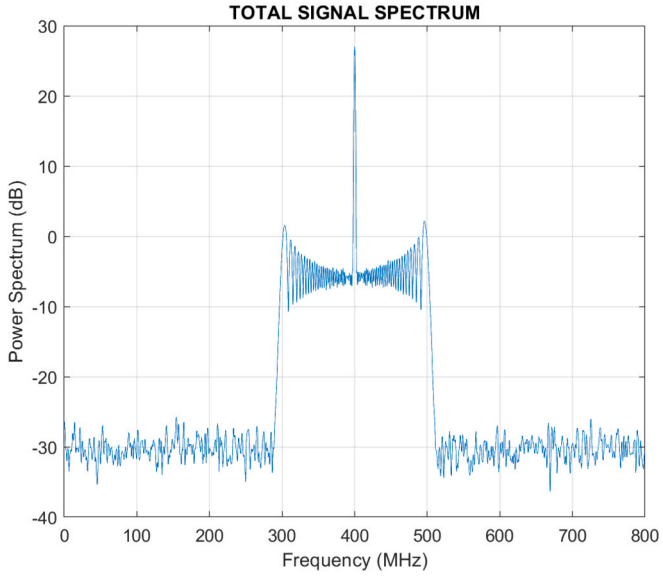



**Fig. 14.** RF Input Signal.

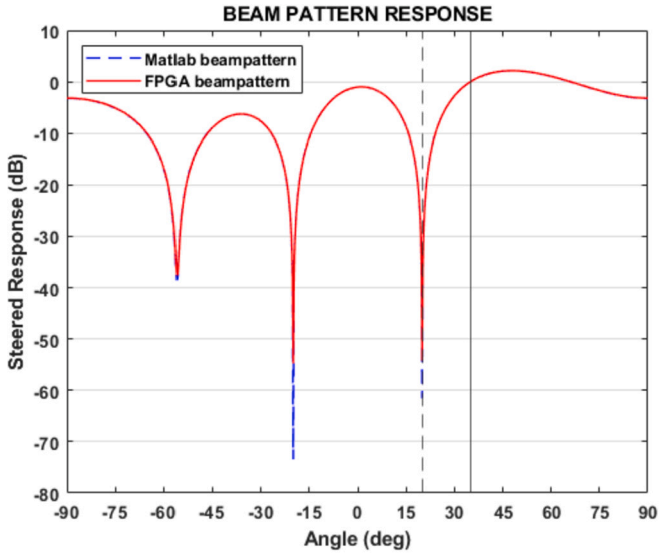



**Fig. 15.** Beampattern Response N = 4 channels.

**Table 8**
Correlation matrix resource utilization.

| Resource | Utilization | Available | Utilization % |
|---|---|---|---|
| CLB LUTs | 5510 | 497520 | 1.11 |
| CLB Registers | 3597 | 995040 | 0.36 |
| CARRY8 | 721 | 75060 | 0.96 |
| F7 Muxes | 0 | 300240 | 0 |
| F8 Muxes | 0 | 150120 | 0 |
| DSPs | 80 | 4100 | 1.95 |

**Table 9**
Systolic array architecture resource utilization.

| Resource | Utilization | Available | Utilization % |
|---|---|---|---|
| CLB LUTs | 36097 | 497520 | 7.26 |
| CLB Registers | 21638 | 995040 | 2.17 |
| CARRY8 | 2402 | 75060 | 3.2 |
| F7 Muxes | 0 | 300240 | 0 |
| F8 Muxes | 0 | 150120 | 0 |
| DSPs | 344 | 4100 | 8.39 |

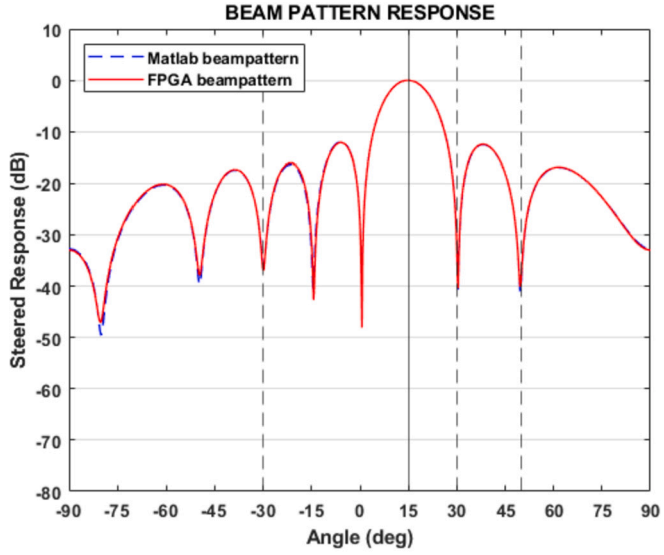



**Fig. 16.** Beampattern response N = 8 channels.

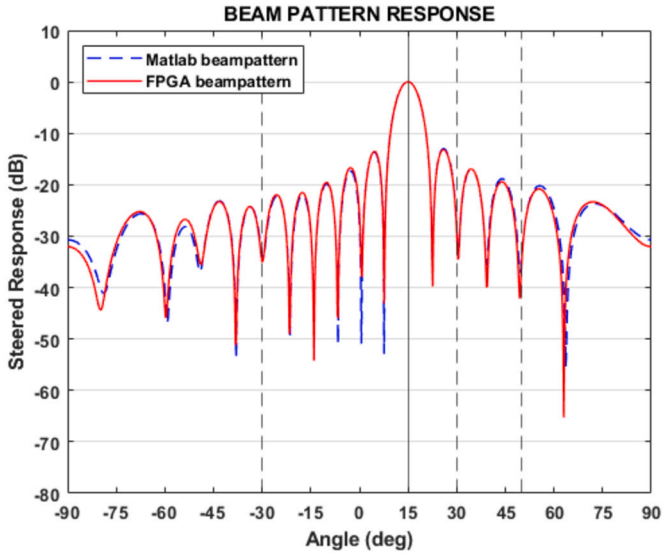



**Fig. 17.** Beampattern response N = 16 channels.

**Table 10**
Adaptive algorithm architecture resource utilization.

| Resource | Utilization | Available | Utilization % |
|---|---|---|---|
| CLB LUTs | 42894 | 497520 | 8.62 |
| CLB Registers | 27131 | 995040 | 2.73 |
| CARRY8 | 3200 | 75060 | 4.26 |
| F7 Muxes | 65 | 300240 | 0.02 |
| F8 Muxes | 0 | 150120 | 0 |
| DSPs | 424 | 4100 | 10.34 |

proposed implemented architecture demonstrates substantial resource savings compared to the standard one.

For the FPGA implementation of the proposed design with 8 and 16 channels, Table 14 shows the resource utilization.

To evaluate the conserved resources achieved by the proposed architecture in contrast to the standard one across varying channel numbers, approximations for the number of DSPs, LUTs, and Registers are depicted in Figs. 22, 23, and 24, respectively. These calculations are derived from the count of boundary and internal cells for each architecture. The blue line denotes the resources allocated by the standard

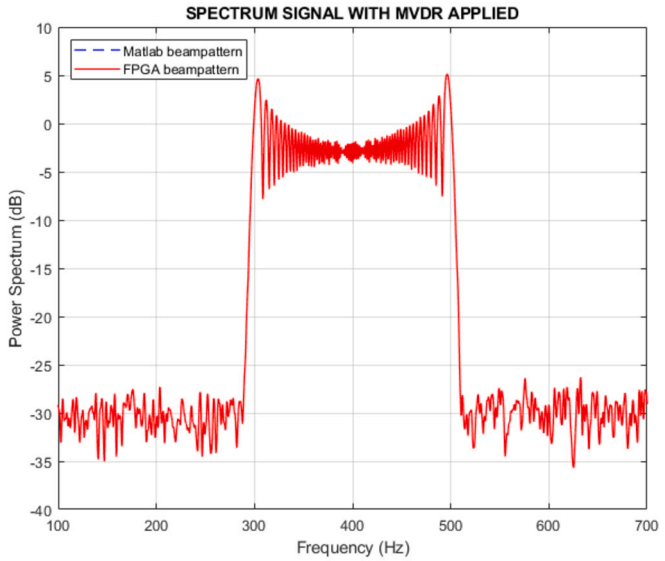



**Fig. 18.** Beamformer Output Signal.

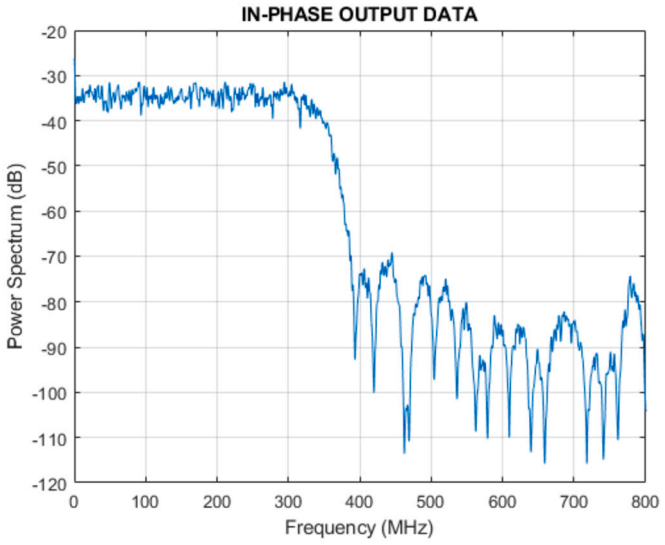



**Fig. 19.** In-phase output processed data.

**Table 11**
Standard version correlation matrix resource utilization.

| Resource | Utilization | Available | Utilization % |
|---|---|---|---|
| CLB LUTs | 13811 | 497520 | 2.77 |
| CLB Registers | 7397 | 995040 | 0.74 |
| CARRY8 | 1681 | 75060 | 2.23 |
| F7 Muxes | 0 | 300240 | 0 |
| F8 Muxes | 0 | 150120 | 0 |
| DSPs | 320 | 4100 | 7.8 |

**Table 12**
Standard version systolic array architecture resource utilization.

| Resource | Utilization | Available | Utilization % |
|---|---|---|---|
| CLB LUTs | 78249 | 497520 | 15.72 |
| CLB Registers | 43639 | 995040 | 4.38 |
| CARRY8 | 4677 | 75060 | 6.23 |
| F7 Muxes | 0 | 300240 | 0 |
| F8 Muxes | 0 | 150120 | 0 |
| DSPs | 808 | 4100 | 19.7 |

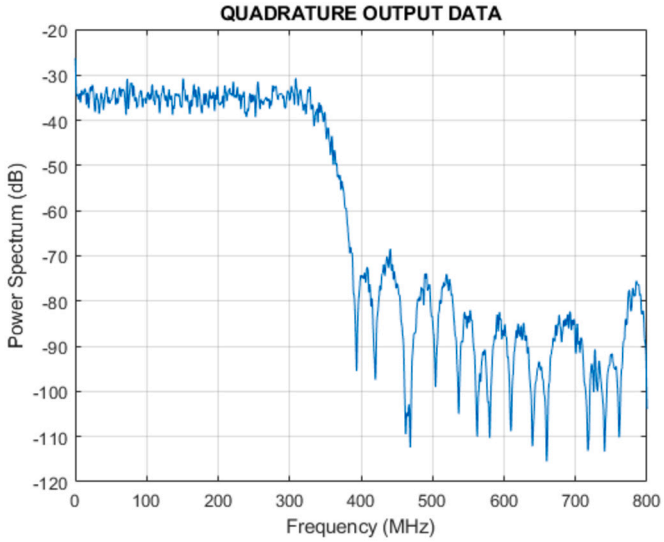



**Fig. 20.** Quadrature output processed data.

**Table 13**
Standard version adaptive algorithm architecture resource utilization.

| Resource | Utilization | Available | Utilization % |
|---|---|---|---|
| CLB LUTs | 93279 | 497520 | 18.74 |
| CLB Registers | 52812 | 995040 | 5.31 |
| CARRY8 | 6426 | 75060 | 8.56 |
| F7 Muxes | 61 | 300240 | 0.02 |
| F8 Muxes | 0 | 150120 | 0 |
| DSPs | 1128 | 4100 | 27.51 |

**Table 14**
Adaptive algorithm resource utilization for 8 and 16 channels.

| Channels | LUTs | Registers | CARRY8 | DSPs |
|---|---|---|---|---|
| 8 | 123448 | 78727 | 9332 | 1224 |
| 16 | 429520 | 265921 | 37854 | 3780 |

architecture, while the red line represents the resources for the proposed design. Remarkably, the disparity in resources grows significantly with an increasing number of channels.

### 5.3. Analysis of computational load

In order to analyze the computational load of the proposed design and compare it with the standard one, the number of real multiplications is derived in this section. For a $N$ number of channels, the number of multiplications performed in the correlation matrix module in the standard design can be expressed as

$$MC1 = 16 \cdot N(N+1) \tag{15}$$

In the new design proposed, the number of multipliers to perform the correlation matrix is determined by

$$MC2 = 4 \cdot N(N+1) \tag{16}$$

In the systolic array structure, the number of multiplications is determined by the number of complex and real type processing elements allocated. Every BC real type performs 1 multiplication, the IC real type performs 2 multiplications, the BC complex type performs 2 multiplications, and the IC complex type performs 8 multiplication operations. In addition, every weight multiplier from the last row in the systolic array performs 4 multiplications, corresponding to 1 complex multiplication.

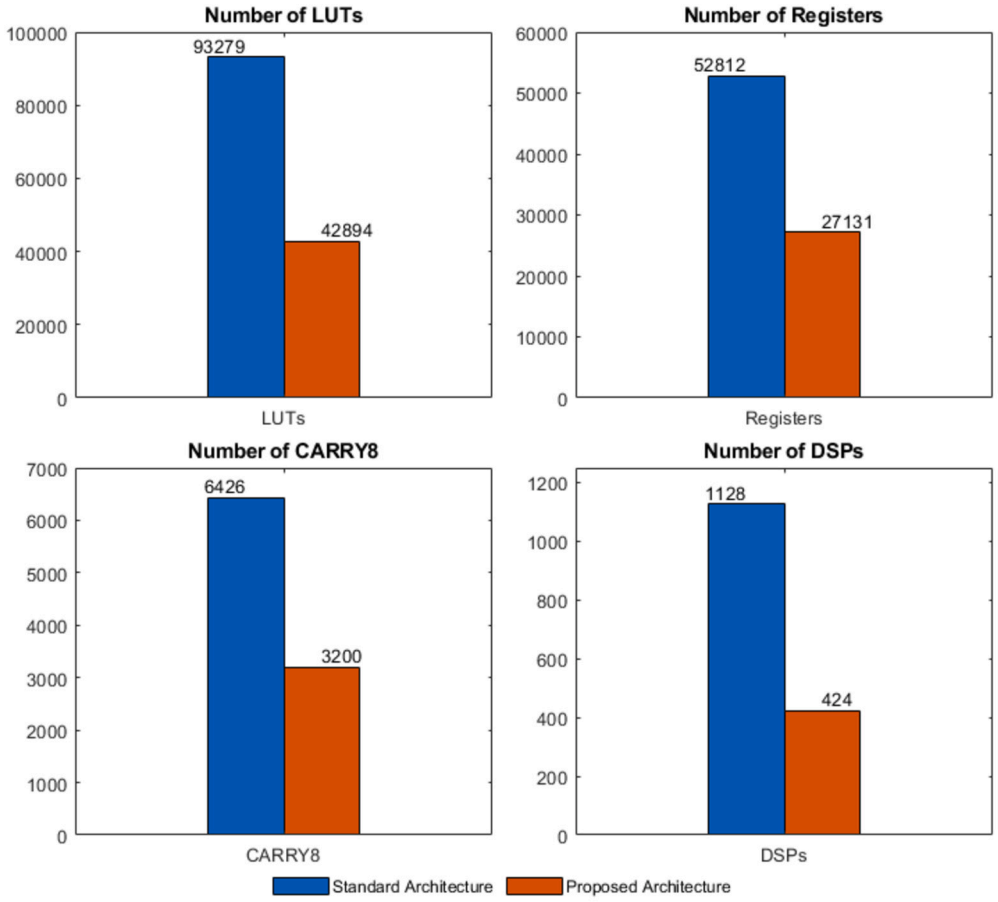



**Fig. 21.** Comparison architectures.

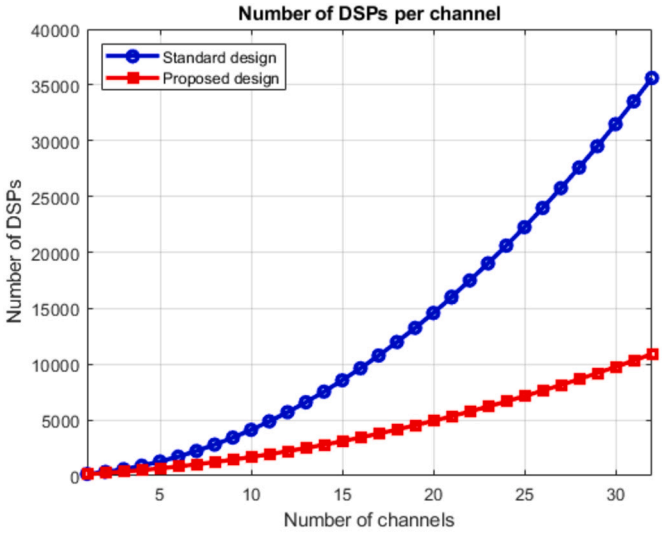



**Fig. 22.** Number of DSPs used per number of channels.

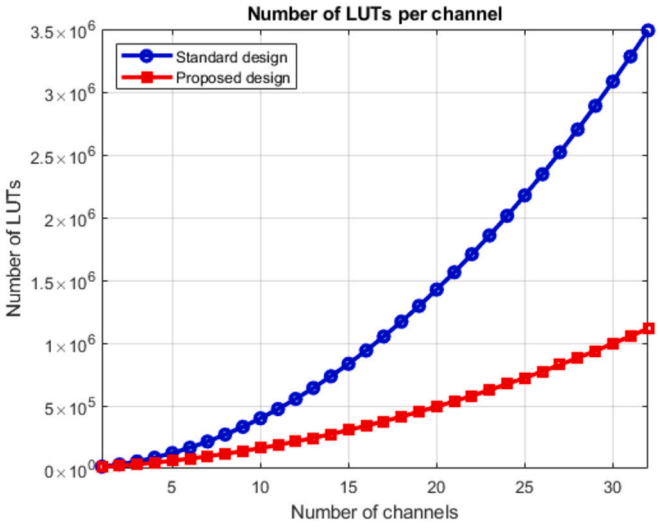



**Fig. 23.** Number of LUTs used per number of channels.

Knowing that in the standard version all the processing elements are complex type, there is a number of $N+1$ BCs and $N(N+2)$ ICs, so the number of multiplications can be expressed as

$$SA1 = (N+1)\cdot 2 + N(N+2)\cdot 8 + N\cdot 4 \tag{17}$$

In the proposed design there is a number of $N$ BC real type, 1 BC complex type, $N^2$ IC real type, and $2N$ IC complex type. Therefore, the total number of multiplications can be determined as

$$SA2 = N + 2 + 2\cdot N^2 + 2N\cdot 8 + N\cdot 4 \tag{18}$$

In (17) and (18) the last term represent the number of weight multipliers cells ($N$), which perform 4 multiplications each. The total number of multiplications performed in the standard design is the sum of $MC1$ and $SA1$, and for the proposed design is the sum of $MC2$ and $SA2$. For a 4 channel system, as the one implemented in the evaluation board, the number of multiplications performed in the standard evaluation is 538,

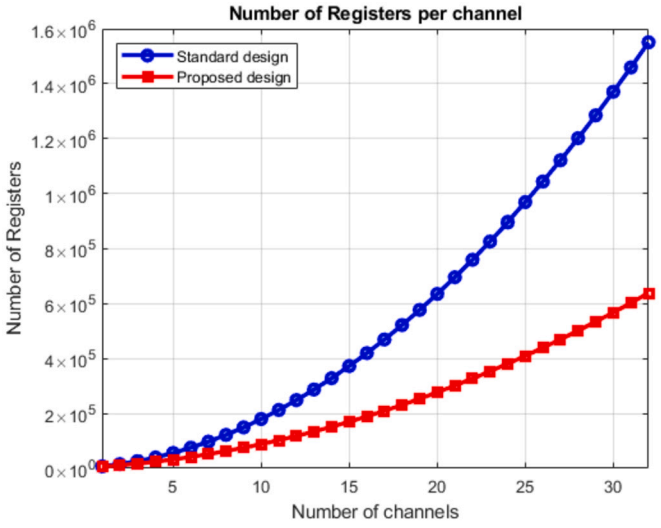



**Fig. 24.** Number of Registers used per number of channels.

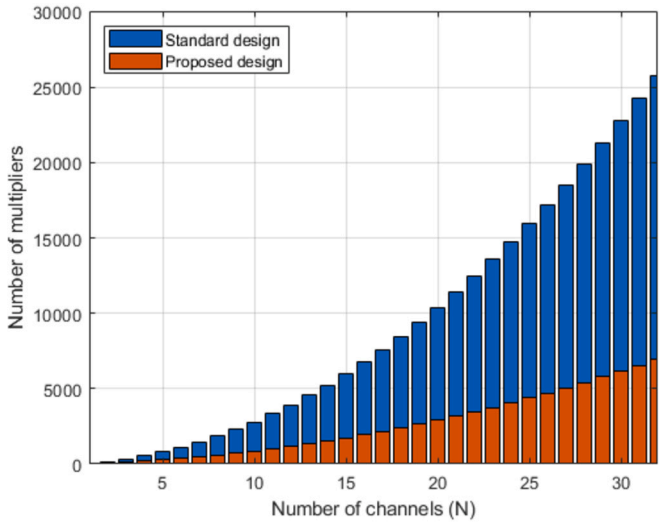



**Fig. 25.** Number of multiplications of the standard design and proposed design.

and in the proposed design is 198. Note that the weight rescale process does not contribute with any multiplication operation.

The comparison between the standard design and the proposed design for different number of channels is illustrated in Fig. 25. From Fig. 25, it can be seen that the proposed design significantly reduces the computational load.

### 5.4. Processing time

The processing time to perform the MVDR algorithm is related with the number of channels and the number of input samples that conform the snapshot. This duration is the sum of the correlation matrix calculation, the inverse QR decomposition, and the weight rescale process, plus a minor time dedicated to control the data flow between the stages. For a 4-channel system and a block of $2^{13}$ samples, the processing time to calculate the optimal weights for the proposed design is 10595 *ns*.

As mentioned before, the number of clock cycles to perform the correlation matrix is dependent on the number of samples read. Every clock cycle, a set of 8 samples in parallel are processed. Considering a block of $2^{13}$ samples, the number of clock cycles needed are $2^{10}$, plus 3 additional cycles as the latency to perform the last multiplication process and control. For a FPGA clock frequency of 200 MHz, the time to resolve the correlation matrix is 5135 *ns*. During this time, the input signals into the

**Table 15**
Processing time per stage.

| Stage | Standard design | Proposed design |
|---|---|---|
| Correlation Matrix | 5135 ns | 5135 ns |
| IQRD | 8015 ns | 5375 ns |
| Weight Rescale | 65 ns | 65 ns |
| Total Time | 13235 ns | 10595 ns |

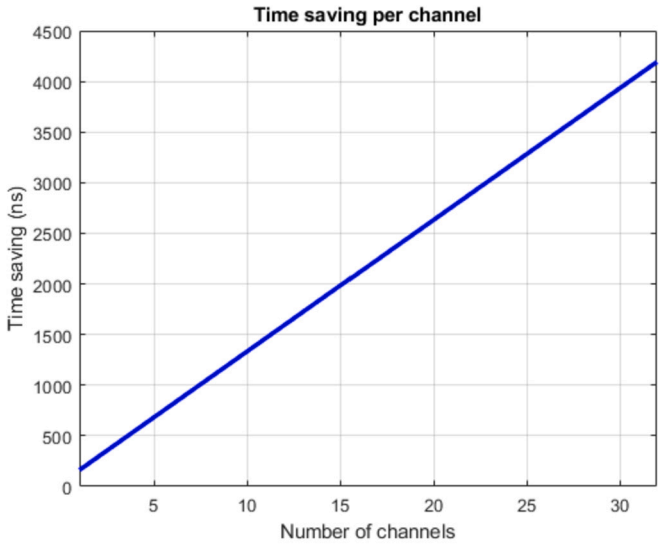



**Fig. 26.** Processing time difference per channel between proposed design and standard design.

array are expected to be stationary [14]. This processing time, hence, determine the variability in the interference signal and the adaptability of the weights generated. Although increasing the number of samples can yield more accurate weight calculations, it also results in longer processing times. Conversely, decreasing the number of samples can reduce processing time, enabling faster adaptation to changing interference conditions, but may compromise the accuracy of interference attenuation.

The time to process the Inverse QR Decomposition is 5375 *ns*. The weights rescale process takes 65 *ns*.

For the standard design implementation, considering the same input data of $2^{13}$ samples and 4 channels, the processing time to perform the MVDR algorithm is 13235 *ns*. From this time, 5135 *ns* were dedicated to the correlation matrix calculation, 8015 *ns* to the Inverse QR Decomposition, and 65 *ns* to the weights rescale process. The Table 15 collects the processing times values for both designs per stage.

The time to process the systolic array structure is related with the number of channels and the type of processing elements. Knowing that the BC and IC real type cells need 34 clock cycles to process their information, and the BC and IC complex type need 66 and 99 clock cycle respectively, and estimation of the processing time difference between the proposed design and the standard one per channel is depicted in Fig. 26.

The proposed design needs less processing time to perform the MVDR algorithm mainly due to less number of operations in the processing elements within the systolic array structure.

The static timing analysis for the proposed architecture, with regard to the worst negative slack for Setup, Hold, and Pulse Width parameters, yields values of 0.055 ns, 0.030 ns, and 0.295 ns, respectively.

### 5.5. Power consumption

The power consumed by the MVDR algorithm implemented on the evaluation board has been measured using the Vivado power analysis

**Table 16**
Power consumption.

| Stage | Standard design | Proposed design |
|---|---|---|
| Correlation Matrix | 0.655 W | 0.285 W |
| IQRD | 3.185 W | 1.405 W |
| Weight Rescale | 0.031 W | 0.031 W |
| Total Power | 3.905 W | 1.753 W |

tool. Table 16 presents the power ratings for each stage of the implemented MVDR algorithm and its total value, comparing both the proposed design and the standard design. As observed in Table 16, the proposed design significantly reduces the power consumption due to the reduced resource utilization in the correlation matrix calculation and the inverse QR decomposition.

In addition, the EV12AQ60X-ADX-EVM evaluation board was connected to a configurable power supply through which the voltage and current measurements could be observed. In this scenario, it is possible to measure separately the power voltage of the EV12AQ600 ADC, LMX2592 PLL and XCKU085 FPGA running the complete Digital Beamforming Receiver system, including the adaptive beamforming and demodulation process. The measured power consumption on the EV12AQ60X-ADX-EVM evaluation board, while running the complete hardware implementation as described, is 1817 mA (21.8 W). This encompasses the ADC, LMX2592 PLL, and ESIstream protocol systems, in addition to the Digital Beamforming Receiver system. When these systems run individually, the power consumption is as follows: 925 mA (11.1 W) for the XCKU085 FPGA with the proposed design implemented, 111 mA (1.33 W) for the LMX2592 PLL activated and configured at 6.4 GHz, and 675 mA (8.1 W) for the EV12AQ600 ADC.

It is noteworthy that the ADC device exhibits significant power consumption, a factor that should be taken into account when scaling this design for new RF input channels and integrating additional ADC devices. The power consumption of the FPGA in the implementation of the standard architecture is approximately 1001 mA, indicating a slight increase compared to the proposed design in this work.

## 6. Conclusion

An innovative architecture proposal has been thoroughly analyzed and implemented for the design of an Adaptive Digital Beamforming Receiver, tailored for space applications. The implementation took place on the Evaluation Board EV12AQ60X-ADX-EVM, featuring the high-speed EV12AQ600 12-bit quad-channel ADC and the high-performance Xilinx Kintex UltraScale XCKU085 FPGA. This design aims to address critical needs in spaceborne phased arrays, specifically focusing on reducing digital signal processing resources, power consumption, and area utilization.

The proposed architecture incorporates the inverse QR decomposition with Givens rotations and the CORDIC algorithm, facilitating a practical FPGA implementation with multiplication-free vector rotations. The design achieves a fully pipelined structure, optimizing memory usage and maximizing parallelism for real-time signal processing within the FPGA hardware. In scenarios where the Digital Beamforming Receiver precedes the demodulation process, the systolic array structure's processing elements are adapted to the type of data they handle. This approach includes mixing different processing elements for real and complex data, resulting in reduced overall resource usage and power consumption compared to a more standard systolic array structure.

The resource and computational load savings achieved by this proposed architecture, in comparison to the standard one, escalate rapidly with the number of antennas. The system's scalability is emphasized by a simple modification of a parameter in the VHDL code to determine the required number of channels. While the FPGA model used in this work is not space-graded, it's important to note that the Xilinx Kintex UltraScale XCKU060 FPGA device, belonging to the same model family and architecture, is space-graded and compatible for use with the proposed implementation.

Applications for this Adaptive Digital Beamforming Receiver, employing MVDR techniques, extend to satellite communication services and spaceborne synthetic aperture radar (SAR) systems. The versatility and scalability of the proposed architecture make it well-suited for various space-based applications, offering improved efficiency and resource utilization.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Data availability

Data will be made available on request.

## References

[1] J.M. Loomis, Digital beamforming - a retrospective, in: 2019 IEEE International Symposium on Phased Array System & Technology (PAST), 2019, pp. 1–4.
[2] D. Sikri, R.M. Jayasuriya, Multi-beam phased array with full digital beamforming for satcom and 5g, in: 5G Phased Array Technologies, 2019, p. 30.
[3] B. Murmann, A/d converter trends: power dissipation, scaling and digitally assisted architectures, in: 2008 IEEE Custom Integrated Circuits Conference, 2008, pp. 105–112.
[4] B.E. Jonsson, A survey of a/d-converter performance evolution, in: 2010 17th IEEE International Conference on Electronics, Circuits and Systems, 2010, pp. 766–769.
[5] K. Merkel, A. Wilson, A survey of high performance analog-to-digital converters for defense space applications, in: 2003 IEEE Aerospace Conference Proceedings (Cat. No. 03TH8652), vol. 5, 2003, pp. 2415–2427.
[6] Xilinx, Rt Kintex UltraScale FPGAs for Ultra High Throughput and High Bandwidth Applications, 2020.
[7] S. Huber, M. Younis, A. Patyuchenko, G. Krieger, A. Moreira, Spaceborne reflector sar systems with digital beamforming, IEEE Trans. Aerosp. Electron. Syst. 48 (4) (2012) 3473–3493.
[8] J. Montesinos, O. Besson, C.L. De Tournemine, Adaptive beamforming for large arrays in satellite communications systems with dispersed coverage, IET Commun. 5 (3) (2011) 350–361.
[9] S. Haykin, J. Litva, T.J. Shepherd, et al., Radar Array Processing, Springer, 1993.
[10] E. Ortega, A. Martínez, A. Oliva, F. Sanz, O. Rodríguez, M. Prieto, P. Parra, A. Da Silva, S. Sánchez, A digital beamforming receiver architecture implemented on a fpga for space applications, arXiv preprint, https://doi.org/10.48550/arXiv.2405.18992, 2024.
[11] W.M. Gentleman, H. Kung, Matrix triangularization by systolic arrays, in: Real-Time Signal Processing IV, vol. 298, SPIE, 1982, pp. 19–26.
[12] S. Haykin, Adaptive Filter Theory, Prentice-Hall Google Schola, vol. 2, 2002, pp. 19–27.
[13] D.H. Johnson, D.E. Dudgeon, Array Signal Processing: Concepts and Techniques, Prentice Hall, 1993.
[14] V. Ingle, S. Kogon, D. Manolakis, Statisical and Adaptive Signal Processing, Artech, 2005.
[15] C. Fernandez-Prades, P. Closas, J. Arribas, Implementation of digital beamforming in gnss receivers, in: Proceedings of the European Workshop on GNSS Signals and Signal Processing, 2009.
[16] V. Behar, C. Kabakchiev, H. Rohling, Mvdr radar signal processing approach for jamming suppression in satellite navigation receivers, in: 11-th International Radar Symposium, IEEE, 2010, pp. 1–4.
[17] D. Li, Q. Yin, P. Mu, W. Guo, Robust mvdr beamforming using the doa matrix decomposition, in: 2011 1st International Symposium on Access Spaces (ISAS), IEEE, 2011, pp. 105–110.
[18] D. Wang, J. Li, W. Gong, S. Wu, Attitude aided space-time multi-beamformer anti-jamming approach for satellite navigation receiver, in: 2014 12th International Conference on Signal Processing (ICSP), IEEE, 2014, pp. 368–372.
[19] J. Liu, B. Weaver, Y. Zakharov, G. White, An fpga-based mvdr beamformer using dichotomous coordinate descent iterations, in: 2007 IEEE International Conference on Communications, IEEE, 2007, pp. 2551–2556.
[20] G. Sow, O. Besson, M.-L. Boucheret, C. Guiraud, Beamforming for satellite communications in emergency situations, Eur. Trans. Telecommun. 19 (2) (2008) 161–171.
[21] B. Van Veen, K. Buckley, Beamforming techniques for spatial filtering, in: Digital Signal Processing Handbook, 1997, 61-1.
[22] N.R. Goodman, Statistical analysis based on a certain multivariate complex Gaussian distribution (an introduction), Ann. Math. Stat. 34 (1) (1963) 152–177.
[23] Teledyne e2v Semiconductors, EV12AQ60x-ADX-EVM User Guide (May 2019).

[24] Teledyne e2v Semiconductors, EV12AQ600 Datasheet (May 2022).
[25] ESIstream, ESIstream the Efficient Serial Interface Protocol Specification (2020).
[26] ESIstream, ESIstream the Efficient Serial Interface User Guide (2020).
[27] AMD Xilinx, UltraScale Architecture and Product Data Sheet: Overview (May 2022).
[28] Xilinx, UltraScale Architecture DSP Slice User Guide, Aug. 2021.
[29] W. Wang, R. Wang, Y. Deng, T. Balz, F. Hong, W. Xu, An improved processing scheme of digital beam-forming in elevation for reducing resource occupation, IEEE Geosci. Remote Sens. Lett. 13 (3) (2016) 309–313.
[30] S. Komeylian, C. Paolini, Implementation of the digital qs-svm-based beamformer on an fpga platform, Sensors 23 (3) (2023) 1742.
[31] L.C. Godara, Application of antenna arrays to mobile communications. II. Beam-forming and direction-of-arrival considerations, Proc. IEEE 85 (8) (1997) 1195–1245.
[32] B.C. Simon, M. Supriya, V. Jayanthi, Deep neural based beamforming techniques for direction of arrival (doa) estimation, in: 2021 International Symposium on Ocean Technology (SYMPOL), IEEE, 2021, pp. 1–9.
[33] A. Alhamed, N. Tayem, T. Alshawi, S. Alshebeili, A. Alsuwailem, A. Hussain, FPGA-based real-time implementation for direction-of-arrival estimation, J. Eng. 2017 (6) (2017) 260–265.
[34] X.-W. Zhang, D. Yan, L. Zuo, M. Li, J.-X. Guo, High-performance of eigenvalue decomposition on fpga for the doa estimation, IEEE Trans. Veh. Technol. (2022).
[35] H.L. Van Trees, Optimum Array Processing: Part IV of Detection, Estimation, and Modulation Theory, John Wiley & Sons, 2002.
[36] J.C. Gentile, et al., Fpga-based adaptive digital beamforming using machine learning for mimo systems, Ph.D. thesis, Johns Hopkins University, 2021.
[37] A.L. Ghirnikar, S. Alexander, R. Plemmons, A parallel implementation of the inverse qr adaptive filter, Comput. Electr. Eng. 18 (3–4) (1992) 291–300.
[38] H. Leung, S. Haykin, Stability of recursive qrd-ls algorithms using finite-precision systolic array implementation, IEEE Trans. Acoust. Speech Signal Process. 37 (5) (1989) 760–763.
[39] W. Givens, Computation of plain unitary rotations transforming a general matrix to triangular form, J. Soc. Ind. Appl. Math. 6 (1) (1958) 26–50.
[40] H.-T. Kung, Why systolic architectures?, Computer 15 (1) (1982) 37–46.
[41] J.A. Apolinário, J.A. Apolinário, R. Rautmann, QRD-RLS Adaptive Filtering, Springer, 2009.
[42] M. Karkooti, J.R. Cavallaro, C. Dick, Fpga implementation of matrix inversion using qrd-rls algorithm, in: Asilomar Conference on Signals, Systems, and Computers, 2005.
[43] B. Haller, J. Gotze, J.R. Cavallaro, Efficient implementation of rotation operations for high performance qrd-rls filtering, in: Proceedings IEEE International Conference on Application-Specific Systems, Architectures and Processors, IEEE, 1997, pp. 162–174.
[44] C. Dick, F. Harris, M. Pajic, D. Vuletic, Real-time qrd-based beamforming on an fpga platform, in: 2006 Fortieth Asilomar Conference on Signals, Systems and Computers, IEEE, 2006, pp. 1200–1204.
[45] L. Wanhammar, DSP Integrated Circuits, Elsevier, 1999.
[46] D. Bindel, J. Demmel, W. Kahan, O. Marques, On computing givens rotations reliably and efficiently, ACM Trans. Math. Softw. 28 (2) (2002) 206–238.
[47] R. Andraka, A survey of cordic algorithms for fpga based computers, in: Proceedings of the 1998 ACM/SIGDA Sixth International Symposium on Field Programmable Gate Arrays, 1998, pp. 191–200.
[48] M. Pu, L. Li, H. Jiang, New adaptive beamforming for coherent interference based on covariance matrix reconstruction, J. Phys. Conf. Ser. 1971 (2021) 012011, IOP Publishing.
[49] S. Mohammadzadeh, O. Kukrer, Adaptive beamforming based on theoretical interference-plus-noise covariance and direction-of-arrival estimation, IET Signal Process. 12 (7) (2018) 819–825.

**Eduardo Ortega** received his M.S. degree in Space Science and Technology in 2020 and his B.S. degree in Telecommunication Engineering from the University of Alcalá in 2004. Since 2019, he has been participating with the Computer Engineering department and the Space Research Group of Universidad de Alcalá working on different projects focused on the Digital beamforming processing electronics for space application, researching new ways to enhance the efficiency for space systems. He is currently pursuing a Ph.D. in Space Research and Astrobiology.

**Alejandro Vicente** received his B.S. degree in Physics from the University of Salamanca in 2021 and his M.S. in New Electronics and Photonics Technologies at University Complutense of Madrid in 2022. He has worked in the University of Alcala in the Space Research group since 2022 focusing in digital beamforming techniques and implementations in software/hardware platforms.

**Agustín Martínez** received the M.S. degree from the Universidad Politécnica de Madrid (UPM) in 1986 and the Ph.D. degree from the Universidad de Alcalá in 2001. Professor of Computer Engineering Department at the University of Alcalá in Spain, Head of Department from 2010 to 2016. His research and teaching activities are in the areas of DSP, Computer Architecture, Embedded Systems and Space Systems. He has been in charge of technical and management issues at Telettra, Alcatel and Alcatel-Lucent over more than 20 years, accounting a deep experience over technical and management leadership activities in international projects.

**Óscar R. Polo** received his M.S. degree in physics from the Universidad del Pais Vasco in 1994 and his Ph.D. in physics from the Universidad Complutense de Madrid, Spain, in 2003. Since 2004, he has been with the Computing Engineering Department at the University de Alcalá. He is currently an assistant professor of embedded and real-time systems. His research interests include computer architecture, satellite on-board software, model driven engineering and embedded real-time systems. He has actively participated in several research projects in the area of computer engineering of satellite platforms in NANOSAT-01, NANOSAT-1B, Solar Orbiter and Euclid missions.

Born in Germany, **Antonio da Silva** has received the MSc and Ph.D. degrees in computer engineering from Universidad de Alcalá (UAH), in 2001 and 2015, respectively. He has worked as embedded software developer for primary radar systems and since 1991 he lectures in the field of Computing Science. He has worked with the Space Research Group of Universidad de Alcalá on projects related to the development of the energetic particle detector (EPD) on-board Solar Orbiter. His research interests include Serious Games for Education, Fault Tolerant systems design and critical software early development and verification.

**Manuel Prieto** is an associate professor in the Computer Engineering Department of the University of Alcalá, Spain. He obtained his first degree in Telecommunications Engineering in 1995. He obtained his degree in Electronics Engineering in 1999 and his Ph.D. in 2005. His research interests are FPGA-based digital electronics and radio-communications. He has taken part in multiple research projects in the field of hardware and software development for space. He has been the project manager of the EPD instrument on board ESA/NASA Solar Orbiter. Dr. Prieto is also involved in research projects in Antarctica for the automation of measurements.

**Pablo Parra** received his Ph.D. in Information and Communication Technologies from the University of Alcalá in 2012. Since 2006, he has been working with the Computer Engineering Department and the Space Research Group (SRG) of the University of Alcalá. His research interests include component-based software engineering and model-driven engineering applied to the field of real-time embedded systems. He has taken part in numerous research projects in the field of on-board satellite software development, such as the NANOSAT programme, Solar Orbiter and Euclid.

**Sebastián Sánchez** received his M.S. degree from the Universidad Politécnica de Madrid in 1994 and his Ph.D. at the Universidad de Alcalá, in 1998. Since 1990, he has been with the Computing Engineering Department at the University de Alcalá. He is full professor of operating systems and computer architecture. His research interests include space instrumentation, embedded real-time systems, and mobile robots. He has actively participated in several national and international research projects in the area of hardware and software on board satellites such as SOHO, PHOTON, FUEGO 2, NANOSAT, Exomars, MICROSAT, Solar Orbiter and Euclid.