Design Exploration and Performance Strategies towards Power-Efficient FPGA-Based Architectures for Sound Source Localization

Bruno da Silva, Laurent Segers, An Braeken, Kris Steenhaut and Abdellah Touhafi

1INDI Department, Vrije Universiteit Brussel, Brussels, Belgium
2ETRO Department, Vrije Universiteit Brussel, Brussels, Belgium

Correspondence should be addressed to Bruno da Silva; bruno.da.silva@vub.be

Received 24 May 2019; Accepted 26 July 2019; Published 15 September 2019

Academic Editor: Tomasz Wandowski

Copyright © 2019 Bruno da Silva et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Many applications rely on MEMS microphone arrays for locating sound sources prior to their execution. Those applications not only are executed under real-time constraints but also are often embedded on low-power devices. These environments become challenging when increasing the number of microphones or requiring dynamic responses. Field-Programmable Gate Arrays (FPGAs) are usually chosen due to their flexibility and computational power. This work intends to guide the design of reconﬁgurable acoustic beamforming architectures, which are not only able to accurately determine the sound Direction-Of-Arrival (DoA) but also capable to satisfy the most demanding applications in terms of power efﬁciency. Design considerations of the required operations performing the sound location are discussed and analysed in order to facilitate the elaboration of reconﬁgurable acoustic beamforming architectures. Performance strategies are proposed and evaluated based on the characteristics of the presented architecture. This power-efﬁcient architecture is compared to a different architecture prioritizing performance in order to reveal the unavoidable design trade-offs.

1. Introduction

Audio streaming applications involve multiple signal processing operations performed in streams of audio signals and, often, on resource and power-limited embedded devices. Many applications demand the processing in parallel of multiple input streams of audio signals while requiring a real-time response. This is the case of microphone arrays, which are nowadays used in many acoustic applications such as hearing aids [1], biometrical systems [2–4], or speech enhancement [5, 6]. Many of these acoustic applications demand an accurate and fast localization of the sound source prior to any operation [7]. Arrays of microphones are used for acoustic sensing to increase the Signal-to-Noise Ratio (SNR) by combining the input signals from the microphones while steering the microphone’s response in a desired direction using acoustic beamforming techniques. Such beamforming techniques involve compute-intensive operations which must be optimized, especially when facing real-time constraints.

FPGAs present valuable features which make them interesting computational units to embedded acoustic beamformers. Firstly, a full customization of an architecture enables multisensor real-time systems. Such embedded systems demand low latency, which cannot be achieved on general-purpose CPUs (e.g., microprocessors). Secondly, FPGAs provide reprogrammable circuitries which can become very power efﬁcient thanks to a high-level architecture customization. Although the amount of the programmable logic resources of low-end FPGAs used in embedded systems is relatively low, streaming applications such as sound locators based on acoustic beamforming can largely beneﬁt from the FPGA’s features. Real-time behavior and power efﬁciency are priorities for sound locators based on acoustic beamforming since a low latency is demanded to estimate the sound Direction-of-Arrival (DoA) while consuming as low power as possible.

We propose several design considerations and performance strategies to fully exploit the current FPGA’s capabilities. On the one hand, design considerations are needed to
properly satisfy the main priority of the acoustic application. Power efficiency is a key feature of the proposed architecture. On the other hand, performance strategies are proposed to accelerate this power-efficient architecture. Each performance strategy firstly considers the architecture’s characteristics before exploiting the FPGA’s features. As a result, reconfigurable acoustic beamforming architectures are able to operate certain orders of magnitude faster.

The presented work extends the architecture proposed in [8]. Several improvements, such as performance strategies to accelerate the proposed power-efficient architecture, are now added. Moreover, the features of alternative architectures are discussed in detail when describing the required operations for sound source localization and their implementation. The main extensions and new results presented in this work are

(i) a complete design exploration of FPGA-based architectures using a time-domain Delay-and-Sum beamforming technique for sound source localization is used to identify the more power-efficient architectures

(ii) performance strategies are proposed to accelerate a power-efficient architecture

(iii) a detailed comparison between the proposed low-power architecture and the high-performance architectures described in [9, 10] helps to identify the trade-offs when targeting power efficiency

This paper is organized as follows. An overview of related literature is presented in Section 2. A detailed description of the required stages is done in Section 3 in order to properly understand the impact of the architecture’s parameters. In Section 4, the metrics used to evaluate acoustic beamforming architectures are described. A power-efficient reconfigurable acoustic beamforming architecture, which embeds not only a time-domain Delay-and-Sum beamformer but also all the operations to demodulate the Pulse Density Modulation (PDM) signals from the microphones, is proposed and analysed in Section 5. This section exemplifies how performance strategies can fully exploit the architecture’s characteristics to accelerate the sound localization. A final comparison with real-time architectures proposed in [9, 10] is done in Section 6 in order to emphasize the existing trade-offs when targeting power efficiency. Finally, some conclusions are drawn in Section 7.

2. Related Work

The interest in microphone arrays has increased in the last decade, partially thanks to recent advances in microelectromechanical systems (MEMS) which have facilitated the integration of microphone arrays in smartphones [11], tablets, or voice assistants, such as Amazon’s Alexa [12]. The digital output formats like PDM or Inter-IC Sound (I2S) currently offered by MEMS microphones facilitate their interface to FPGA-based systems. Figure 1 depicts the number of papers related to microphone arrays, MEMS microphones, and FPGA-based sensor arrays since 1997. The number of publications related to microphone arrays has significantly increased in the last decades (notice the log scale in the number of publications). There is a relation between the evolution of the MEMS technology and the replacement of Digital Signal Processors (DSPs) by FPGAs for audio processing. The majority of the publications related to MEMS microphones or microphone arrays mainly discuss microphone technologies, with around 4% of the overall number of publications describing FPGA-based systems using microphone arrays. We believe that FPGAs are currently underexploited in this area [13].
FPGA-based embedded systems provide enough resources to fully embed acoustic beamformers while presenting power efficiency [8–10]. Such features, however, might not be directly obtained when developing reconfigurable acoustic beamforming architectures. Related literature lacks in design exploration, with few publications discussing architecture’s parameters [14, 15] or exploring the full FPGA potential [2, 16].

Fully embedded architectures are, unexpectedly, rare. The available resources and the achievable performance that current FPGAs provide facilitate the signal processing operations required by the PDM demodulation and the beamforming techniques. An example of a fully embedded beamforming-based acoustic system for localization of the dominant sound source is presented in [17, 18]. The authors in [17] propose a FPGA-based system consisting of a microphone array composed of up to 33 MEMS microphones. Their architecture fully embeds the PDM demodulation detailed in [19] together with a Delay-and-Sum beamformer and a Root Mean Square (RMS) detector for the sound source localization.

The authors in [20] also fully embed a beamforming-based acoustic system composed of digital MEMS microphone arrays acting as a node of a Wi-Fi-based Wireless Sensor Network (WSN) for deforestation detection. The proposed architecture performs the beamforming operations before the PDM demodulation and filtering. Instead of implementing individual PDM demodulators for each microphone, the authors propose the execution of the Delay-and-Sum beamforming algorithm over the PDM signals. The output of the Delay-and-Sum, which is no longer a 1-bit PDM signal, is filtered by windowing and processed in the frequency domain. As power consumption is a critical parameter for WSN-related application, their architecture uses an extremely low-power Flash-based FPGA, which allows to only consume 21.8 mW per 8-element microphone array node. A larger version of this microphone array, composed of 16 microphones, is proposed by the author in [21]. Their architecture is migrated to a Xilinx Spartan-6 FPGA due to the additional computational operations, leading to 61.71 mW of power consumption. FPGA-based low-power architectures for WSN nodes to perform sound source localization are, however, not an exception. The authors in [8] propose a multimode architecture implemented on an extremely low-power Flash-based FPGA, achieving a power consumption as low as 34 mW for a 52-element microphone array proposed in [22]. In these architectures, the strategy of beamforming PDM signals has the benefit of saving area and power consumption due to the drastic reduction of the number of filters needed. The architecture’s trade-offs, like the real-time capabilities, are, however, not discussed.

Current low-end FPGAs provide enough resources to perform in real-time complex beamforming algorithms involving tens of microphones. Nevertheless, the choice of the architecture is strongly linked to the characteristics and constraints of the target application. Here, a power-efficient architecture is proposed to fully exploit the power-efficient but also resource-constrained FPGAs.

3. Stages of Reconfigurable Architectures for Time-Domain Acoustic Beamforming

Reconfigurable acoustic beamforming architectures share several common components to perform the signal processing operations required to locate sound sources using acoustic beamforming techniques with a MEMS microphone array. The mandatory operations can be grouped in several stages embedding the processing of the acquired data from the MEMS microphone array on the FPGA. Although the microphone array is an external sensing component from the FPGA perspective, its features directly determine some of the architecture’s characteristics. Nonetheless, the implementation of reconfigurable acoustic beamforming architectures demands a study and analysis of the impact of the application’s parameters. For instance, the sampling frequency ($F_s$) of the audio input determines the filters’ response at the PDM demodulation stage. $F_s$ also affects the beamforming operation, affecting the FPGA resource consumption, which might be critical when targeting small FPGA-based embedded systems.

The impact of the design parameters on the implementation is analysed in this section. Firstly, the required stages in the operations for the audio retrieval, beamforming operations, and the sound localization are detailed. Secondly, a design space of each stage is explored in order to identify the key design parameters and their impact. Such a Design-Space Exploration (DSE) is general enough to obtain power-efficient as well as high-performance reconfigurable architectures such as the one presented in [10]. We start with a short overview of the required stages enabling audio retrieval, beamforming, and sound localization.

**Microphone array:** digital MEMS microphones, especially PDM MEMS ones, have become popular when building microphone arrays [13]. Besides the multiple advantages of MEMS microphones, some of their features, such as low-power modes, make them interesting candidates for reconfigurable acoustic beamforming architectures. A generic microphone array is used to exemplify how the low-power mode of PDM MEMS microphones can be exploited to construct arrays supporting a variable number of active microphones. Such flexibility demands additional design considerations.

**Filter stage:** instead of integrating the PDM demodulator in the microphone package, PDM MEMS microphones output a single-bit PDM signal, which needs to be demodulated at the processing side. The PDM demodulation requires additional computations, rather undesirably seen in the relatively low amount of resources available on FPGAs in embedded systems, but it also presents an opportunity to build fully customized acoustic beamformer architectures targeting sound location. The PDM demodulation must also be flexible enough to support dynamic microphone arrays while being power- and resource-efficient. The parameters which determine the required filter response are here identified and used to evaluate multiple designs.

**Beamforming stage:** its relatively low complexity makes the Delay-and-Sum technique the most popular acoustic beamforming technique. The inherent parallelism in large
microphone arrays can be exploited when embedding this type of beamformer on FPGAs. Although the computation of the Delay-and-Sum in the frequency domain is mentioned in the literature, its execution in the time domain is preferred since it avoids the computation of the discrete Fourier transformation, which is a time- and resource-demanding computation. The complex data representation of the data in the frequency domain demands a high bit width or even the use of floating-point representation, leading to multiple Multiply-ACCumulate (MACC) operations to perform the phase shift corrections needed to compensate the difference in path lengths. Instead, the computation of the Delay-and-Sum beamforming technique in the time domain is reduced to the storage of audio samples to synchronize the acquired audio samples for a particular steered orientation. The consumption of the FPGA internal memory, which is used to properly delay the audio samples during the beamforming operation, can be optimized through a judicious choice of design parameters.

**Power stage:** the direction of the sound source is located by measuring the Sound Relative Power (SRP) per horizontal direction. The SRP obtained by a 360° sweep overview of the surrounding sound field is known as Polar Steered Response Power (P-SRP), which provides information about the power response of the array. The P-SRP is only obtained after the audio recovery and the beamforming operation. The accuracy of the DoA based on the P-SRP is determined by parameters such as the number of steered directions. The impact of this parameter is evaluated in the DSE.

The parameters leading to a dynamic response of the sound locator, which can be adapted at runtime thanks to the FPGA’s flexibility, are firstly presented together with their unavoidable trade-offs.

### 3.1. Parameters for a Runtime Dynamic Response

FPGAs present an opportunity to develop dynamic reconfigurable acoustic beamforming architectures, which self-adapt their configuration to satisfy certain criteria. The power consumption, for instance, can be dynamically adjusted at runtime. This dynamism is obtained by exploiting the following architecture parameters at the design stage.

**Active microphones:** the number of active microphones in the array ($N_{am}$) directly affects power consumption, frequency response, and performance. The architecture, however, must be designed to support a variable $N_{am}$. The architecture must be able to selectively deactivate the PDM MEMS microphones of the array, for instance, through the clock signal, and be able to deactivate the FPGA resources associated with disabled microphones. The following DSE demonstrate how this deactivation can be supported at runtime, without requiring a partial reconfiguration of the FPGA.

**Angular resolution:** one of the parameters which determine the capability to properly determine the DoA is the angular resolution. The number of steered orientations ($N_o$) defines the achievable angular resolution when calculating the P-SRP. Similar to $N_{am}$, this parameter affects the frequency response, the performance, and, indirectly, the power consumption. With these features in mind, the architecture can be designed to support a runtime variable angular resolution as presented in [9, 10].

**Sensing time:** the sensing time ($t_s$), a well-known parameter of radio frequency applications, represents the time the receiver is monitoring the surrounding sound field. This parameter is known to increase the robustness against noise [23] and directly influences the probability of proper detection of a sound source. The value of $t_s$ is determined by the number of processed acoustic samples at the power stage ($N_p$). A higher $N_p$ is needed to detect and to determine the direction of the sound sources under low SNR conditions. Reconfigurable acoustic beamforming architectures can certainly support a variable $N_p$ to adapt at runtime the sensing of the array based on a continuous SNR estimation. Although the proposed architecture must support a variable sensing time at runtime, the evaluation of this parameter is out of the scope of the presented work.

The three parameters are used to provide dynamism to reconfigurable acoustic beamforming architectures. Note that the selection at runtime of the values of $N_{am}$, $N_o$, and $N_p$ leads to multiple trade-offs, as already summarized in Table 1. The exact values used for the architecture’s analysis are detailed in Table 1.

### 3.2. Description of the Stages

#### 3.2.1. PDM MEMS Microphone Array

The position of the microphones into the array, known as the array geometry, does affect not only the system’s response but also the parameters described in Section 3.1. Moreover, the grouping of microphones in subarrays enables a variable $N_{am}$ and frequency response [9, 10]. This is a topic that has been largely explored ([24, 25] or [26]) and is out of the scope of this work. The microphone array used to evaluate the proposed reconfigurable architectures presents the following characteristics to achieve the desired dynamism:

(i) The array is composed of PDM MEMS microphones.

---

**Table 1: Architecture’s parameters used for the analysed reconfigurable acoustic beamforming architectures, the range under evaluation, and their trade-offs.**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Range</th>
<th>Trade-offs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of active microphones ($N_{am}$)</td>
<td>4, 12, 28, 52</td>
<td>Resources, frequency response, directivity, and power</td>
</tr>
<tr>
<td>Number of steered orientations ($N_o$)</td>
<td>4, 8, 16, 32, 64</td>
<td>Execution time, performance, frequency response, and directivity</td>
</tr>
<tr>
<td>Number of acoustic samples at the power stage ($N_p$)</td>
<td>64</td>
<td>Power, execution time, and performance</td>
</tr>
</tbody>
</table>

---

[9, 10] This is a topic that has been largely explored ([24, 25] or [26]) and is out of the scope of this work.
(ii) The PDM MEMS microphones support low-power modes (a.k.a. sleep modes).

(iii) The microphones are grouped in subarrays through a common clock signal.

The reference microphone array is composed of 52 digital PDM MEMS microphones like described in [22]. The array geometry consists of four concentric subarrays of 4, 8, 16, and 24 PDM MEMS microphones mounted on a 20 cm circular printed circuit board, depicted in Figure 2. Each concentric subarray has a different radius and number of microphones to facilitate the capture of spatial acoustic information using a beamforming technique. The selection of PDM MEMS microphones is also motivated to the multiple modes that such microphones support. Most of the PDM MEMS microphones offer a low-power mode and drastically reduce their power consumption when the microphones’ clock signal is deactivated. This feature allows the construction of microphone arrays composed of multiple subarrays. The response of these microphone arrays can be dynamically modified by individually activating or deactivating subarrays. This distributed geometry can also adapt the architecture’s response to different sound sources. For instance, not all subarrays need to be active to detect a particular sound source. The value of $N_{am}$ has a direct impact on the array’s output SNR since the SNR increases with $N_{am}$. In this regard, the computational requirements drastically decrease and the sensor array becomes more power efficient if only a few subarrays are active.

The features of the described microphone array, like the deactivation of the microphones or their group in subarrays, lead to microphone arrays with dynamic response, ideal for high-performance or power-efficient reconfigurable architectures.

3.2.2. Filter Stage. The audio retrieval from PDM MEMS microphones requires certain operations. The first operation to be performed on the FPGA is the PDM demultiplexing since every pair of microphones has its PDM output signal multiplexed in time. The PDM demultiplexing is a mandatory operation to retrieve the individual sampled audio data from each microphone. The incoming data from one of the microphones is sampled at every clock edge. A PDM splitter block, located on the FPGA, demultiplexes the PDM samples.

(1) PDM Demodulators. Figure 3 depicts the internal components of a PDM MEMS microphone. The MEMS transducer converts the input Sound Pressure Level (SPL) to a voltage. This transducer is followed by an impedance converter amplifier, which stabilizes the output voltage of the MEMS for the Sigma-Delta ($\Sigma\Delta$) modulator. The analog signal is digitalized at the ADC and converted into a single-bit PDM signal by a fourth-order $\Sigma\Delta$ modulator running at a high oversampling rate. PDM is a type of modulation used to represent analog signals in the digital domain, where the relative density of the pulses corresponds to the analog signal’s amplitude. The $\Sigma\Delta$ modulator reduces the added noise in the audio frequency spectrum by shifting it to higher frequency ranges. This undesirable high-frequency noise needs to be removed when recovering the original audio signal.

Digital MEMS microphones usually operate at a clock frequency ranging from 1 MHz to 3.072 MHz [27] or up to 3.6 MHz [28]. This range of $F_s$ is chosen to oversample the audio signal in order to have sufficient audio quality and to generate the PDM output signal in the $\Sigma\Delta$ modulator. The PDM signal needs not only to be filtered in order to remove the noise but also to be downsampled to convert the audio signal to a Pulse-Code Modulation (PCM) format.

Several examples of PDM demodulators proposed in the literature and incorporated in commercial MEMS...
microphones are depicted in Figure 4. For instance, the PDM demodulators in Figures 4(a) and 4(b) are the block diagrams of the \( \hat{f} \)S MEMS microphones [29, 30], respectively. The PDM demodulator in [29] incorporates a decimator to downsample the PDM signal by a factor of 64 and converts the signal to PCM. The remaining high-frequency components in the PCM signal are removed by a low-pass filter. The PDM demodulator in [30] is composed of two cascaded filters acting as a digital bandpass filter. The first one is a low-pass decimator filter which eliminates the high-frequency noise, followed by a high-pass filter, which removes the DC and the low-frequency components. Notice that the decimation factor and the filters’ response are fixed in both cases. This fact reduces the DSE since it limits the operational frequency range of the target acoustic application to be a fixed multiple of the microphones’ sampling frequency \( F_S \). For instance, if the PDM demodulator decimates by a fixed factor of 64 like in [29], the microphones must be clocked at \( F_S = 3.072 \) MHz for a desired output audio at 48 kHz. At that frequency, audio signals up to 24 kHz can be recovered without aliasing according to the Nyquist theorem.

PDM demodulators in Figures 4(c) and 4(d), which are proposed in [19] and [14], respectively, present a cascade of three different types of filters in a filter chain fashion, that is, a CIC decimation filter followed by two half-band filters with a decimator factor of 2 and a low-pass FIR filter in the final stage. The CIC filters are used to convert the PDM signals in PCM format. This class of linear phase FIR filters, developed by Hogenauer [31, 32], involves only additions and subtractions. It consists of 3 stages: the integrator stage, the decimator or interpolator stage, and the comb section.
PDM input samples are recursively added in the integrator stage while being recursively subtracted with a differential delay in the comb stage. The number of recursive operations in the integrator and comb section determines the order of the filter \(N_{\text{CIC}}\). This order should at least be equal to the order of the \(\Sigma\Delta\) converter from the DAC of the microphones. After the CIC filter, the signal growth \(G\) is proportional to the decimation factor \(D_{\text{CIC}}\) and the differential delay \(D\) and is exponential to the filter order [32]. CIC decimation filters decimate the signal by \(D_{\text{CIC}}\) and convert the PDM signal in PCM at the same time. A major drawback of this type of filter is the nonflat frequency response in the desired audio frequency range. To improve the flatness of the frequency response, a CIC filter with a lower decimation factor followed by compensation filters is usually a better choice, as proposed in [19, 32, 33]. The CIC filter is followed by a couple of half-band filters of order \(N_{\text{HB}}\) with a decimation factor of two. Half-band filters are widely used in multirate signal processing applications. These types of filters that let only half of the frequency band of the input signal present two important characteristics. Firstly, the passband and stopband ripple must be the same. Secondly, the passband-edge and stopband-edge frequencies are equidistant from the half-band frequency \(\pi/2\). As a result, the filter’s coefficients are symmetrical and every second coefficient is zero. Both characteristics can be exploited for resource savings. The last component is a low-pass compensation FIR filter of order \(N_{\text{FIR}}\) to remove the high-frequency noise introduced by the ADC conversion process in the microphone. This filter can also be designed to compensate the passband drop usually introduced by CIC filters [32]. Optionally, it can additionally perform a downsampling of the signal being further decimated by a factor of \(D_{\text{FIR}}\) like that proposed in Figure 4(d).

(2) Proposed PDM Demodulator. The analysed filter stage, originally proposed in [10] and in [8], is composed of single or multiple filter chains performing the PDM demodulation. Each filter chain corresponds to several cascaded filters performing a PDM demodulation of the microphone array output signals (Figure 5), simplifying the PDM demodulators in Figures 4(c) and 4(d) by reducing the number of the cascaded filters. Both half-band filters are replaced by a moving average filter, which removes the DC level of the CIC’s output signal, improving the dynamic range of the signal entering the low-pass compensation FIR filter. The FIR filter presents a cut-off frequency of \(F_{\text{max}}\) at a sampling rate of \(F_s/D_{\text{CIC}}\), which is the sampling rate obtained after the CIC decimator filter with a decimation factor of \(D_{\text{CIC}}\). The stream nature of such architecture enables the generation of an output value from the CIC filter every clock cycle. Due to the decimation factor, only one output value per \(D_{\text{CIC}}\) input value is propagated to the low-pass FIR filter. Consequently, the FIR filter has \(D_{\text{CIC}}\) clock cycles to compute each input value. This low-pass FIR filter needs to be designed in a serial fashion to reduce the resource consumption, and its maximum order is also determined by \(D_{\text{CIC}}\):

\[
N_{\text{FIR}} \leq D_{\text{CIC}} - 1.
\]

Hereby, \(N_{\text{FIR}}\) is assumed to be equal to its maximum order \((D_{\text{CIC}} - 1)\) since the order is directly related to the quality of the response of the filter. The overall \(D_s\) can be expressed based on the downsampling rate change of each filter

\[
D_s = \left[ \frac{F_s}{2 \cdot F_{\text{max}}} \right] = \left[ \frac{F_s}{\text{BW}} \right] = D_{\text{CIC}} \cdot D_{\text{FIR}},
\]

where \(D_{\text{FIR}}\) is the decimation factor needed for the FIR filter to obtain the minimum bandwidth \(\text{BW}\) to satisfy the Nyquist theorem for the target \(F_{\text{max}}\).

The filter chain depicted in Figure 5 enables dynamic architectures while performing the PDM demodulation. The range of parameters such as \(F_s\) and \(F_{\text{max}}\) depends on the PDM MEMS microphone specifications. For instance, the PDM MEMS microphone ADMP521 from Analog Devices used in [22] operates at a \(F_s\) in a range from 1.25 MHz to 3.072 MHz as specified in [27], and its frequency response ranges from 100 Hz to 16 kHz. The specifications of the acoustic application also determine \(F_{\text{max}}\), which must be in the range of the supported frequencies of the microphone. Both parameters, \(F_s\) and \(F_{\text{max}}\), determine the value of \(D_s\) and, therefore, the signal rate of each filter. However, not all possible configurations are supported when specifying the low-pass FIR filter’s characteristics. For instance, the passband and the stopband, the transition band, or the level of attenuation of the signal out of the passband limit the supported FIR filter’s configurations.

(3) DSE of the PDM Demodulator. Based on the previous description of the different filters of the filter chain, a DSE can be done to evaluate the supported configurations. The analysis considers an \(F_{\text{max}}\) ranging from 13 kHz to 16.5 kHz in steps of 125 Hz, and \(F_s\) ranges from 1.25 MHz to 3.072 MHz. All possible combinations of \(D_{\text{CIC}}\) and \(D_{\text{FIR}}\) are considered on equation (2), based on the \(D_s\) obtained for every possible value of \(F_s\) and \(F_{\text{max}}\). The low-pass FIR filter parameters are \(N_{\text{FIR}}\), which is determined by \(D_{\text{CIC}}\), and \(F_{\text{max}}\) as the cut-off frequency. Each possible low-pass FIR filter is generated considering a transition band of 3.5 kHz and

![Figure 5: The filtering stage consists of single or multiple filter chains performing the PDM demodulation.](image-url)
an attenuation of at least 60 dB at the stop band. If the minimum order or the filter is higher than $N_{\text{FIR}}$, the filter is discarded. Furthermore, a minimum order of 8 is defined as the threshold for $N_{\text{FIR}}$. Some values are discarded because $D_p$ is a prime number or $N_{\text{FIR}}$ is below 8. The filter’s parameters are realistic constraints for low-pass FIR filters.

Figure 6 depicts the values of $D_{\text{CIC}}$ for the supported configurations detailed in Table 2. Each low-pass FIR filter is generated and evaluated in MATLAB 2016b. The values of $D_{\text{CIC}}$ provide information of $D_p$ and $D_{\text{FIR}}$ due to equation (2). Higher values of $F_{\text{max}}$ allow higher values of $D_{\text{CIC}}$, which can greatly reduce computational complexity of narrowband low-pass filtering. However, too high values of $D_{\text{CIC}}$ lead to such low rates that, although a higher-order low-pass FIR filter is supported, it cannot satisfy the low-pass filtering specifications. Notice how the number of possible solutions decreases when increasing $F_{\text{max}}$. Due to $F_S$ and $F_{\text{max}}$ ranges, the values of $D_p$ vary between 38 and 154. Although, as previously explained, many values cannot be considered since they are either prime numbers or the decomposition in factors of $D_{\text{CIC}}$ that leads to values below 8. Because higher values of $F_{\text{max}}$ lead to low values of $D_{\text{CIC}}$ for low $F_S$, these $D_{\text{CIC}}$ values cannot satisfy the specifications of the low-pass FIR filter. High values of $D_{\text{CIC}}$ lead to high-order low-pass FIR filters and lower $D_{\text{FIR}}$.

The presented DSE of the filter chain performing the PDM demodulation is general enough to be applied to any of the PDM demodulators depicted in Figure 4. It can be applied to identify the most performing solutions as well as to reduce the resource consumption as discussed in the following section.

### 3.2.3. Beamforming Stage.

Microphone arrays can focus a specific orientation thanks to beamforming techniques. Such techniques amplify the sound coming from the targeted direction while suppressing the sound coming from other directions. The time-domain Delay-and-Sum beamforming is a beamforming technique that delays the output signal of each microphone by a specific amount of time before adding all the output signals together. The detection of sound sources is possible by continuously steering in loops of 360°. The number of steered orientations per 360° sweep, $N_{\text{s}}$, is the angular resolution of the microphone array. Higher angular resolutions demand not only a larger execution time per steering loop but also more FPGA memory resources to store the precomputed delays per orientation.

The beamforming stage performs the time-domain Delay-and-Sum beamforming operation and is composed of a bank of memories, a precomputed table of delays, and several cascaded additions. Although Delay-and-Sum beamforming assumes a fixed number of microphones ($N_{\text{Mics}}$) and a fixed geometry, our scalable solution satisfies those restrictions while offering a flexible geometry [9]. Figure 7 shows our proposed beamforming stage, which is basically composed of FPGA blocks of memory (BRAM) in ring-buffer fashion that properly delays the filtered microphone signal. The delay for a given microphone is determined by its position on the array and on the focus orientation. All possible delay values ($\Delta$) per microphone for each beamed orientation are precomputed, grouped per orientation, and stored in

<table>
<thead>
<tr>
<th>Definition</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F_S$</td>
<td>1.25 MHz to 3.072 MHz</td>
</tr>
<tr>
<td>$F_{\text{max}}$</td>
<td>13.75 kHz to 16.5 kHz</td>
</tr>
<tr>
<td>Steps for $F_{\text{max}}$</td>
<td>125 Hz</td>
</tr>
<tr>
<td>Parameters of the low-pass FIR filter</td>
<td></td>
</tr>
<tr>
<td>Minimum order</td>
<td>7</td>
</tr>
<tr>
<td>Transition band</td>
<td>4 kHz</td>
</tr>
<tr>
<td>Stop band attenuation</td>
<td>60 dB</td>
</tr>
</tbody>
</table>
ROMs during compilation time. During execution time, the delay values $\Delta_m(\theta)$ of each microphone $m$ when pointing to a certain orientation $\theta$ are obtained from this precomputed table.

The memory requirements of the beamforming stage are obtained for all the possible locations of the beamforming stage between the components of the filter stage. Figure 8 depicts the potential locations of the Delay-and-Sum-based beamformer. The memory requirements of the beamforming stage based on $F_S$ and on $F_{max}$ are shown in Figure 9. That figure depicts the memory requirements for the supported configurations of the filter chain explored when assuming the FIR filter’s characteristics summarized in Table 2. All the discussed characteristics of the filter stage depicted in Figure 6 are evaluated for each possible placement of the Delay-and-Sum-based beamformer. The first possible location is between the microphone array and the CIC filter. The beamforming memory demand linearly increases with $F_S$. The input signals to be stored are single-bit PDM signals, which in theory should reduce the need for memory. However, due to high values of $\Delta_m$, thousands of PDM signals need to be stored per microphone. The bit width of the output signals from the CIC filter grows [32], which increases the beamforming memory demands when placing the beamforming stage after the CIC filter. Nevertheless, the signal bit width after the moving average filter and before the low-pass FIR filter can be reduced to 32 bits. Although a lower bit width would not cause significant signal degradation, 32 bits are assumed enough to guarantee the good signal quality for all supported possible microphone array configurations. Due to the audio signal downscaling, low values of $\Delta_m$ are obtained when the beamforming stage is located after the low-pass FIR filter, leading to a significant reduction of the memory demands. Detailed analysis of the beamforming memory demands fuels the quest for the most memory efficient architecture.

The memory requirements depicted in Figure 9 have been calculated for a beamforming stage designed to support
a variable $N_{\text{Mics}}$. The input signals are grouped following their subarray structure. Every microphone $m$ is associated with a memory, which properly delays that particular audio stream with an amount $\Delta m$. Each delay memory belonging to a subarray has the same width and length to support all possible orientations. The length is defined by the maximum
delay \((\Delta_{mi})\) of that subarray \(i\), which is determined by the MEMS microphone planar distribution and \(F_S\). All memories associated with the same subarray can be disabled. Therefore, instead of implementing one simple Delay-and-Sum beamformer for a 52-element microphone array, there are four Delay-and-Sum beamforming operations in parallel for the subarrays composed of 4, 8, 16, and 24 microphones. Their sum operation is firstly done locally for each subarray and afterwards between subarrays. The only restriction of this modular beamforming is the synchronization of the outputs in order to have them properly delayed. Therefore, the easiest solution is to delay all the subarrays with the maximum delay \((\max(\Delta_{mi}))\) of all subarrays. Although the output of some subarrays is already properly delayed, additional delays, shown at the Sums part in Figure 7, are inserted to assure the proper delay for each subarray. This is achieved by using the valid output signals of each subarray beamforming, without additional resource cost. Consequently, only the Delay-and-Sum beamforming modulo linked to an active subarray is enabled. The nonactive beamformers are set to zero in order to avoid any negative impact of the beamforming operation.

A side benefit of this modular approach is a reduced memory consumption. Figure 10 shows the memory savings for the supported configurations of the filter chain explored in the previous section. Since each subarray has its ring-buffer memory properly dimensioned to its maximum sample delay, the portion of underused regions of the memories is significantly lower. For the filter chain parameters under evaluation, the memory savings range between 19% and 23%. The variation of the memory savings depends on the placement of the beamforming stage in the architecture. Thus, a mostly constant memory saving of around 21% is possible when the beamforming stage is located between the microphone array and the filter chains. The higher variation occurs when the beamforming stage is located at the end of the filter chains because the memory demands are more sensitive to small differences in the maximum delay values. For instance, whereas in the first case \(\max(\Delta)\) rounds to 1048, its value is reduced to 16 when the beamforming stage is located after the filter chain. The modular approach of the beamforming stage does not only increase the flexibility of the architecture by supporting a variable number of microphones at runtime but also represent a significant reduction of the memory requirements.

3.2.4. Power Stage. The Delay-and-Sum beamforming technique allows to obtain the relative sound power of the retrieved audio stream for each steering direction. The computation of the P-SRP in each steering direction provides information about the power response of the array. The power value per steering direction is obtained by accumulating all the individual power values measured for a certain time \(t_e\) needed to detect and locate sound sources under low SNR conditions. All the power signals in one steering loop conform the P-SRP. The peaks identified in the P-SRP point to the potential presence of sound sources.

Figure 11 shows the components of the power stage. Once the filtered data has been properly delayed and added, the SRP can be obtained for a particular
orientation \( \theta \). The P-SRP is obtained after a steering loop, allowing the determination of the sound sources. The sound source is estimated to be located in the direction shown by the peak of the maximum SRP.

### 3.2.5. Summary

The proposed design considerations and their impact on the architecture are summarized in Table 3. Notice, however, that such design considerations can be individually applied to each stage.

### 4. Evaluation of Reconfigurable Acoustic Beamforming Architectures

The selection of the design parameters determines the characteristics of the reconfigurable acoustic beamforming architecture. The speed of the architecture, the frequency response, and the accuracy of the sound localization are some of the features used to evaluate and compare designs. The following metrics are used to evaluate the reconfigurable architectures embedding time-domain beamformers for sound location:

1. **Acoustic response**
   - (a) Frequency response
   - (b) Directivity

2. **Architecture’s characteristics**
   - (a) Timing and performance
   - (b) Resource and power consumption

#### 4.1. Evaluation of the Acoustic Response

The first two metrics are used to determine the quality of the sound localization and use the array’s characteristics to profile the overall response of the selected architecture. The directional power output of a microphone array shows the directional response of the architecture to all sound sources present in a sound field. The lobes of this polar map can then be used to estimate the bearing of nearby sound sources in nondiffuse sound field conditions. The defined P-SRP allows the estimation of the DoA of multiple sound sources under different sound field conditions. The accuracy of its estimation can be determined by the following quality metrics.

**Frequency response:** the evaluation of the frequency response of the reconfigurable acoustic beamforming architecture for different sound source frequencies is needed. The beam pattern of the microphone array can be represented as a waterfall diagram. Such diagrams show the power output of the sound locator in all directions for all frequencies, which demonstrates how \( D_p \) varies with multiple orientations and frequencies. Waterfall diagrams allow the evaluation of the frequency response of different reconfigurable acoustic beamforming architectures for certain beam orientations. The resolution of the waterfall diagram can be increased by reducing the frequency steps or by increasing the number of steered angles.

**Directivity:** the P-SRP’s lobes are used to estimate the bearing of nearby sound sources in nondiffuse sound field conditions. The capacity of the main lobe to unambiguously point to a specific bearing when considering the scenario of a single sound source determines the architecture’s directivity (\( D_p \)). This definition of \( D_p \) is originally proposed in [34] for broadband signals, where \( D_p \) is used as a metric of the quality of the architecture as a sound locator since \( D_p \) depends on the main lobe shape and its capacity to unambiguously point to a specific bearing. \( D_p \) is a key metric when locating sound sources because it reflects how effectively the architecture discriminates the direction of a sound source. The definition of directivity presented in [34, 35] is adapted for 2D spatial coordinates [22] as follows:

\[
D_p(\theta, \omega) = \frac{\int_0^{2\pi} P(\theta, \omega)^2 d\theta}{(1/2\pi) \int_0^{2\pi} P(\theta, \omega)^2 d\theta},
\]

where \( P(\theta, \omega) \) represents the output power of the microphone array when pointing to the sound source’s direction \( \theta \) and \( (1/2\pi) \int_0^{2\pi} P(\theta, \omega)^2 d\theta \) is the average output power in all other directions. It can be expressed as the ratio between the area of a circle whose radius is the maximum power of the array and the total area of the power output. Therefore, \( D_p \) defines the quality of the sound locator and can be used to specify certain thresholds for the architecture. For instance, if \( D_p \) equals 8, the main lobe is eight times lower than the unit circle and offers a trustworthy estimation of a sound source within half a quadrant.

#### 4.2. Evaluation of the Architecture’s Characteristics

An evaluation of the reconfigurable acoustic beamforming architecture must cover not only the quality of the sound location but also other implementation-related parameters like the achievable performance and the power consumption.

**Time and performance analysis:** a proper timing analysis helps to identify performance bottlenecks and to tune the architecture towards lower latency. The time needed by the microphone array to compute P-SRP (\( t_{P-SRP} \)) can be determined by decomposing the execution time of the architectural stages. A proper implementation of the stages, and especially of their data flow, can significantly reduce this time. For instance, \( t_{P-SRP} \) decreases if the architecture is designed to pipeline the operations of each stage within a steered orientation, enabling the overlapping of the execution.
of the architecture’s components. A detailed analysis of the implementation of each component and its latency provides a good insight in the speed of the system. On the other hand, a performance analysis of a reconfigurable acoustic beamforming architecture gives an idea about what design parameters have a higher performance impact. The performance units can be defined at different levels. The processed audio samples per second reflect the reusability of the acquired data. During the beamforming operation, the same audio sample can be used to calculate the SRP for multiple different orientations. Another performance unit could be the number of beamformed orientations per second (Or/s). This type of units better reflects the achievable performance of reconfigurable acoustic beamforming architectures and facilitates the comparison of reconfigurable acoustic beamforming architectures in terms of performance.

Resource and power consumption: further analysis regarding the power or resource consumption of the architecture is needed to satisfy the architecture’s target priorities. For instance, the streaming nature of acoustic beamforming applications, with continuous flux of incoming data, needs a large amount of memory to store the intermediate results of the signal processing operations. As analysed in the previous section, a decomposition in subarrays of the beamforming stage reduces the consumption of internal memory. However, it also affects the power consumption and might finally determine the supported FPGA.

The metrics described above are used in the next section to evaluate a power-efficient architecture. Different performance strategies are proposed to increase the performance of this architecture.

Table 3: Summary of the design considerations per stage.

<table>
<thead>
<tr>
<th>Stage</th>
<th>Design consideration</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microphone array</td>
<td>PDM MEMS microphones</td>
<td>Deactivation of microphones. Power savings.</td>
</tr>
<tr>
<td></td>
<td>Subarray decomposition</td>
<td></td>
</tr>
<tr>
<td>Filter stage</td>
<td>Moving average filter</td>
<td>Resource savings. Improved dynamic range.</td>
</tr>
<tr>
<td></td>
<td>Serial FIR filter</td>
<td></td>
</tr>
<tr>
<td>Beamforming stage</td>
<td>Subarray decomposition</td>
<td>Power and resource savings.</td>
</tr>
</tbody>
</table>

Figure 12: Overview of the proposed power-efficient architecture. The Delay-and-Sum beamforming is composed of several memories to properly delay the input signal. Our implementation groups the memories associated to each subarray to disable those memories linked to deactivated microphones. The beamformed input signal is converted to audio in the cascade of filters. The DoA is finally obtained based on SRP obtained per orientation.

5. Power-Efficient Reconfigurable Architecture

Current low-end FPGAs offer enough resources to embed power-efficient reconfigurable acoustic beamforming architectures such as the one described and analysed in this section. The presented architecture, firstly presented in [8], drastically reduces resource consumption, making it suitable for low-end Flash-based FPGAs. This type of FPGAs presents a power consumption as low as few tens of mW but lacks available resources. Such resource restriction drastically reduces the achievable performance if the architecture’s characteristics are not properly exploited. Here, different performance strategies are applied in order to accelerate this architecture, becoming more attractive for time-sensitive applications.

Figure 12 depicts the main components of the power-efficient architecture. The input rate is determined by the microphone’s clock and corresponds to $F_s$. The architecture
is designed to operate in the streaming mode, which guarantees that each component is always computing after an initial latency.

The oversampled PDM signal coming from the microphones is multiplexed per microphone pair, requiring a PDM splitter block to demultiplex the input PDM signal into 2 PDM separate channels. Thus, the PDM streams from each microphone of the array are properly delayed at this stage to perform the Delay-and-Sum beamforming operation. The beamforming stage is followed by the filter stage, where the high-frequency noise is removed and the input signal is downsampled to retrieve the audio signal. Notice that a filter stage is only composed of one filter chain like the one described in Section 3.2.2 instead of \( N_{\text{Mic}} \) filter chains thanks to placing the beamforming stage before the filter stage. The SRP for the beamed orientation is calculated in the last stage. The lobes of the P-SRP are used to estimate the DoA for the localization of the sound sources.

5.1. Architecture Performance Exploration. The architecture is designed to satisfy power-constraint acoustic beamforming applications. Multiple performance strategies can be applied to increase performance while preserving the power efficiency. Such strategies minimize the timing impact of the signal processing operations of the architecture.

The execution time \( t_{\text{P-SRP}} \) is defined as the time needed to obtain the P-SRP. Each steered orientation involves multiple signal processing operations that can be executed concurrently in a pipelined way. Therefore, the times to filter \( t_{\text{Filtering}} \), to beamform \( t_{\text{Beamforming}} \), and to get the SRP \( t_{\text{Power}} \) are overlapping with the sensing time \( t_{s} \). Although most of the latency of each component of the design is hidden when pipelining operations, there are still some cycles, defined as Initiation Interval (II), dedicated to initialize the components. The proposed architecture also demands an additional time to reset the filters \( t_{r} \) at the end of the computation of each orientation. The relatively low value of \( t_{r} \) can be neglected because only a few clock cycles are needed to reset the filters. As detailed in Figure 13, \( t_{\text{P-SRP}} \) for a certain \( N_{o} \) can be determined by

\[
t_{\text{P-SRP}} = N_{o} \cdot (t_{\text{II}} + t_{r} + t_{s}) \approx N_{o} \cdot (t_{\text{II}} + t_{s}),
\]

where \( t_{\text{II}} \) corresponds to the sum of the II of the filter stage \( (t_{\text{II}}^{\text{Filtering}}) \), the II of the beamforming stage \( (t_{\text{II}}^{\text{Beamforming}}) \), and the II of the SRP stage \( (t_{\text{II}}^{\text{Power}}) \).

The power-efficient architecture presents several limitations when considering performance strategies. For instance, due to the architecture’s characteristics, the strategies proposed in [10] cannot be applied without a significant increment of the resource consumption. Some new performance strategies are here proposed to overcome these limitations.

5.1.1. Continuous Beamforming. The computation of P-SRP considers a reinitialization of the beamformer per beamed orientation. Such initialization can be drastically reduced if the architecture continuously beamforms the acquired data. Due to the fact that the beamforming stage is mainly composed of delay memories, the required data to start the computation of SRP for a new orientation has been already stored when computing the previous orientation. Therefore, a single initialization is needed at the very beginning, as detailed in Figure 14. The value of \( t_{o} \) becomes

\[
t_{o} = t_{\text{II}}^{\text{Filtering}} + t_{\text{II}}^{\text{Power}} + t_{r} + t_{s},
\]

since \( t_{\text{II}}^{\text{Beamforming}} \) must be only considered when the system starts. In this regard, \( t_{\text{P-SRP}} \) becomes

\[
t_{\text{P-SRP}} = N_{o} \cdot t_{o} \approx N_{o} \cdot (t_{\text{II}}^{\text{Filtering}} + t_{s}),
\]

where \( t_{\text{II}}^{\text{Filtering}} \) and \( t_{r} \) are neglected.

The performance for using this strategy is defined as follows:

\[
\text{Performance} = \frac{1}{t_{o}} = \frac{1}{t_{\text{II}}^{\text{Filtering}} + t_{s}}.
\]

5.1.2. Parallel Continuous Time Multiplexing. The single increment of the functional clock frequency beyond \( F_{S} \) is not enough to improve performance improvement. Unfortunately, the simple acceleration by increasing the operational frequency of the filter stage demands the
prestorage of all the samples acquired in $t_s$. Although such storage can take place at the beamforming stage, this component would largely increase its resource consumption to store $N_S \cdot (D_s/F_s)$ samples per microphone. Because of the nature of the filtering operation, the switching between different orientations demands the storage of the intermediate values stored in the multiple taps of the filter structure. Such storage should be applied for each intermediate register of each filter and for each orientation. The impact of this resource overhead would be similar to the cost of replicating each filter chain per orientation. This strategy causes a significant increment of resource consumption due to the fact that the filter stage is located after the beamforming stage. The solution is the replication of the filter stage while increasing its operational frequency to $F_P$. Figure 15 details how the architecture would perform when multiple filter chains, as many as $N_o$, are available. Two clock regions are defined since the filter chains must operate at a higher frequency in order to retrieve the beamformed data from the beamforming stage (Figure 16). The incoming data from the microphone array enters the beamforming stage at $F_S$ rate. To process $N_o$ orientations in parallel in one clock cycle at $F_S$, the beamforming stage needs to generate data at a desired $F_P$:

$$F_P \geq \frac{F_S \cdot N_{\text{FStages}}}{N_B}.$$  \hspace{1cm} (8)

where $N_{\text{FStages}}$ is the number of filter chains available in the filter stage and $N_B$ is the number of beamformed values out of the beamforming stage accessible per clock cycle. The value of $N_B$ is defined as

$$N_B = N_{\text{BStages}} \cdot M_{\text{ports}},$$  \hspace{1cm} (9)

with $N_{\text{BStages}}$ the number of beamforming stages if the available resources support more than one and $M_{\text{ports}}$ the memory ports of each beamforming memory. For instance, dual-port memories allow 2 readings per memory access, which results in 2 output beamformed values per clock cycle if the sums performed in the beamforming stage are duplicated. The use of dual-port memories is equivalent to duplicating the beamforming stage composed of single-port memories. In both cases, 2 beamformed values can be loaded from each microphone delay memory. This strategy, however, does not exploit the remaining resources to instantiate multiple beamforming stages, and therefore, $N_B$ is assumed to be 1 for this strategy since single-port memories are considered for the beamforming stage.

The value of $N_{\text{FStages}}$ not only is determined by the available resources but also depends on $N_o$ and the maximum operational frequency $F_{\text{op}}$. The number of orientations that can be computed in parallel when increasing the operational frequency of the filter stage to $F_{\text{op}}$ defines the number of the supported filter stages $N_F$ as

$$N_F = \left \lfloor \frac{F_{\text{op}} \cdot N_B}{F_S} \right \rfloor.$$  \hspace{1cm} (10)

\hspace{1cm}
Therefore, \( N_{\text{FStages}} \) becomes

\[
N_{\text{FStages}} = \min \left( N_o, N_R, N_F \right),
\]

(11)

where \( N_R \) is the supported number of filter chains determined by the available resources. Notice that \( N_{\text{FStages}} \) can be limited to \( N_o \) when there is no additional benefit of processing a higher value of \( N_o \), which is determined by the target acoustic application.

With this strategy, the time to compute one orientation is reduced with a factor of \( N_{\text{FStages}} \):

\[
t_0 = t_{\text{II} + \text{Filtering}} + t_e + t_s,
\]

(12)

and the value of \( t_{\text{P-SRP}} \) becomes

\[
t_{\text{P-SRP}} = N_o \cdot t_o \approx \frac{N_o}{N_{\text{FStages}}} \left( t_{\text{II} + \text{Filtering}} + t_s \right),
\]

(13)

where \( t_{\text{Power}} \) and \( t_e \) are neglected. Regarding the achievable performance,

\[
\text{Performance} = \frac{N_{\text{FStages}}}{\left( t_{\text{II} + \text{Filtering}} + T_s \right)}. \tag{14}
\]

The cost, however, is an increment of the resources and power consumption. The extra resources are dedicated to the \( N_{\text{FStages}} \) filter chains or even to additional beamforming stages if \( N_{\text{FStages}} \) is limited by \( N_F \), in order to fully compute in parallel.

The strategies to improve performance are summarized in Table 4. Notice that their impact is not as significant since the main goal of the power-efficient architecture is to reduce the resource consumption and, as a result, the overall power consumption.

5.2. Experimental Results. The design parameters of the architecture under evaluation are summarized in Table 5. The variation of the target \( F_{\text{max}} \) and the \( F_S \) directly affects the beamforming stage by determining the length of the memories and the filter stage, by determining the decimation factor and the FIR filter order. Moreover, the impact of \( N_{\text{am}} \), which changes at runtime thanks to the subarray distribution, is analysed. Like the evaluation of the previous architecture, P-SRP is obtained from a steering loop composed of 64 orientations. The power-efficient architecture has been evaluated for a Microsemi’s SmartFusion2 M2S025 FPGA.

5.2.1. Frequency Response. The frequency response of the microphone array is determined by \( N_{\text{am}} \). The experiments cover four configurations with 4, 12, 28, or 52 microphones determined by the number of active subarrays. The waterfall diagram of each configuration is generated in order to analyse the frequency response while locating sound sources. The waterfall diagrams show the power output of the combined subarrays in all directions for all frequencies. The results are calculated with a single sound source placed at 180°. The frequency of the sound source varies between 100 Hz and 15 kHz in steps of 100 Hz. All results are normalized per frequency.

Figure 17 depicts the waterfall diagrams when combining a different number of subarrays. Every waterfall shows a clear distinctive main lobe. However, this lobe dominates the most in case when subarrays 3 and 4 are also capturing sound waves. When only subarray 1 is active, the side lobes affect the capacity of finding the main lobe. The frequency response of the subarrays improves when they are combined since their frequency responses are superposed. Consequently, the combination of the subarrays 1 and 2 reaches a minimum

### Table 4: Summary of the performance strategies for the described power-efficient architecture.

<table>
<thead>
<tr>
<th>Strategy</th>
<th>Performance</th>
<th>Time execution of one P-SRP ( (t_{\text{P-SRP}}) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>( \frac{1}{t_{\text{II}} + t_s} )</td>
<td>( N_o \cdot (t_{\text{II}} + t_s) )</td>
</tr>
<tr>
<td>Continuous beamforming</td>
<td>( \frac{1}{t_{\text{II}} + t_s} )</td>
<td>( N_o \cdot (t_{\text{II}} + t_s) )</td>
</tr>
<tr>
<td>Parallel continuous time multiplexing</td>
<td>( \frac{N_{\text{FStages}}}{t_{\text{II}} + t_s} ) with ( N_{\text{FStages}} )</td>
<td>( \frac{N_o \cdot t_{\text{Filtering}} + t_s}{N_{\text{FStages}}} ) with ( N_{\text{FStages}} = \min (N_o, N_R, N_F) )</td>
</tr>
</tbody>
</table>

### Table 5: Configuration of the architecture under analysis.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Definition</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( F_S )</td>
<td>Sampling frequency</td>
<td>2.08 MHz</td>
</tr>
<tr>
<td>( F_{\text{min}} )</td>
<td>Minimum frequency</td>
<td>1 kHz</td>
</tr>
<tr>
<td>( F_{\text{max}} )</td>
<td>Maximum frequency</td>
<td>16.250 kHz</td>
</tr>
<tr>
<td>BW</td>
<td>Minimum bandwidth to satisfy Nyquist</td>
<td>32.5 kHz</td>
</tr>
<tr>
<td>( D_{\text{F}} )</td>
<td>Decimation factor</td>
<td>64</td>
</tr>
<tr>
<td>( D_{\text{CIC}} )</td>
<td>CIC filter decimation factor</td>
<td>32</td>
</tr>
<tr>
<td>( N_{\text{CIC}} )</td>
<td>Order of the CIC filter</td>
<td>4</td>
</tr>
<tr>
<td>( D )</td>
<td>CIC differential delay</td>
<td>32</td>
</tr>
<tr>
<td>( D_{\text{FIR}} )</td>
<td>FIR filter decimation factor</td>
<td>2</td>
</tr>
<tr>
<td>( N_{\text{FIR}} )</td>
<td>Order of the FIR filter</td>
<td>31</td>
</tr>
</tbody>
</table>
detectable frequency of 2.4 kHz, whereas the combination of the subarrays 1, 2, and 3 and the combination of all subarrays reach 2.2 kHz and 1.8 kHz, respectively.

5.2.2. Directivity. The standalone waterfall diagrams only provide information about the frequency response but cannot be considered a metric of the quality of the sound source location. Alongside with the waterfalls, $D_p$ is calculated to properly evaluate the quality of the array’s response. The evaluation covers a variable $N_o$ and $N_{am}$. A low angular resolution leads to a lower resolution of the waterfall diagrams, but only the metrics can show the impact. The frequency response of a subarray has a strong variation at the main lobe and, therefore, in $D_p$. A threshold of 8 for $D_p$ indicates that the main lobe’s surface corresponds to maximum half of a quadrant. Figure 18 depicts the evolution of $D_p$ for our frequency range when increasing the angular resolution and when combining subarrays. The angular resolution determines the upper bound $D_p$ converges to, which is defined in equation (3), and coincides with the number of orientations. Notice that a higher angular resolution does not improve $D_p$ when only the inner subarray is active. The value of $N_{am}$, on the other hand, determines how fast $D_p$ converges to its upper limit, based on the frequency of the sound source. Thus, a higher value of $N_{am}$ increases $D_p$ for lower sound source frequencies. For instance, in the case that only subarray 1 is used, $D_p$ shows better results only at frequencies beyond 6.9 kHz. This frequency decreases to approximately 1.7 kHz when all microphones are used in the beamforming.

The evaluation of the frequency response of this architecture concludes that the angular resolution determines the quality of the array’s response. This is reflected in the $D_p$, which is clearly limited when reducing the number of orientations. $N_{am}$ determines if at what sound source frequency a certain threshold is achieved. Clearly, a higher value of $N_{am}$ allows the achievement of a better $D_p$ at a lower frequency. A higher number of orientations and active microphones
lead to other trade-offs. Whereas the angular resolution will also affect the performance, the $N_{am}$ will determine the power consumption.

5.2.3. Resource Consumption. The relatively low resource requirements of the power-efficient architecture allow the use of small and low-power Flash-based FPGAs. Table 6 summarizes the resource consumption of the evaluated power-efficient architecture. The target SmartFusion2 M2S025 FPGA provides enough resources to allocate one instantiation of the architecture when fully using all the 52 MEMS microphones of the array. A higher number of subarrays mainly increase the resource consumption of the beamforming stage. Moreover, the most demanding resources are the dedicated memory blocks, achieving occupancy rates of 76.5% and 90.3% for the two types of memory resources uSRAM (RAM64x18) and LSRAM (RAM1K18), respectively, available in this Microsemi FPGA family [36]. The use of all subarrays demands the use of VHDL attributes to distribute the allocation of the different delay memories of

![Figure 18: Directivities of the power-efficient architecture when considering a variable number of orientations and active microphones. (a–d) The $D_p$ with only 8 orientations up to 64 orientations.](image)
Table 6: Resource consumption after placement and routing when combining the two innermost subarrays and when all microphones are active. Each subarray combination details the resource consumption of the filter and the beamforming stage.

<table>
<thead>
<tr>
<th>Resources</th>
<th>Available resources</th>
<th>Inner 4 mics</th>
<th>Inner 12 mics</th>
<th>Inner 28 mics</th>
<th>All 52 mics</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Beamformer</td>
<td>Filters</td>
<td>Beamformer</td>
<td>Filters</td>
</tr>
<tr>
<td>4LUT</td>
<td>27696</td>
<td>377</td>
<td>839</td>
<td>1626</td>
<td>955</td>
</tr>
<tr>
<td>DFF</td>
<td>27696</td>
<td>319</td>
<td>3858</td>
<td>4649</td>
<td>750</td>
</tr>
<tr>
<td>Interface logic</td>
<td>27696</td>
<td>144</td>
<td>144</td>
<td>432</td>
<td>144</td>
</tr>
<tr>
<td>RAM64x18</td>
<td>34</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>RAM1K18</td>
<td>31</td>
<td>4</td>
<td>0</td>
<td>4</td>
<td>12</td>
</tr>
<tr>
<td>MACC</td>
<td>34</td>
<td>0</td>
<td>2</td>
<td>6</td>
<td>0</td>
</tr>
</tbody>
</table>
the beamforming stage between these memory resources, since the Libero 11.8 tool does not automatically allocate the beamforming memories into these memory resources. In fact, the delay values linked to the outer subarray composed of 16 MEMS microphones have to be entirely allocated in uSRAM blocks. The consumption of the logic resources achieves a maximum consumption of 26% of the available D-type Flip-Flops (DFFs). In this regard, some of the performance strategies detailed in Section 5.1 can benefit from the use of the remaining logic resources.

5.2.4. Power Analysis. Flash-based FPGAs like Microsemi’s IGLOO2, PolarFire, or SmartFusion2 not only offer the lowest static power consumption, demanding only few tens of mW, but also support an interesting sleep mode called Flash-Freeze. The Flash-Freeze mode is a low-power static mode that preserves the FPGA configuration while reducing the FPGA’s power draw to just 1.92 mW for IGLOO2 and SmartFusion2 FPGAs [37].

Table 7 summarizes the reported power consumption. The power consumption of the FPGA design has been obtained by using the Libero SoC 11.8 Power tool, obtaining the dynamic and the static power consumption. Whereas the static power consumption is the power consumed based on the used resources, the dynamic power consumption is determined by the computational usage of the resources and the dynamism of the input data. The static power remains mainly constant since there is no significant increment of the consumption when increasing the number of microphones. The dynamic power consumption, on the contrary, increases since a large number of data must be stored and processed. The overall power consumption of the reconfigurable architecture rounds from 17.8 mW to 23.7 mW, which represents a significant reduction compared to architectures like the high-performance architecture in [9, 10], whose power consumption ranges from 122 mW to 138 mW. Furthermore, the low-power consumption of the FPGA is partially possible thanks to operating at a relatively low frequency (2.08 MHz).

Table 7: Power consumption expressed in mW when combining microphone subarrays. The values are obtained from the Libero SoC v.11.8 power report for the FPGA operating at $F_s = 2.08$ MHz and considering the standard mode of the PDM MEMS microphones [38].

<table>
<thead>
<tr>
<th>Active subarrays</th>
<th>MEMS microphones</th>
<th>Reported on-chip power</th>
<th>Total power</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Active</td>
<td>Inactive</td>
<td>Total</td>
</tr>
<tr>
<td>Inner 12 mics</td>
<td>9.288</td>
<td>0.864</td>
<td>10.152</td>
</tr>
<tr>
<td>Inner 28 mics</td>
<td>21.672</td>
<td>0.518</td>
<td>22.190</td>
</tr>
<tr>
<td>All 52 mics</td>
<td>40.248</td>
<td>0</td>
<td>40.248</td>
</tr>
</tbody>
</table>

Further acceleration is only possible by increasing the resource consumption while operating the filter stage at a higher frequency. Based on the remaining resources of the SmartFusion2 M2S025, up to 6 filter chains can be allocated in parallel due to the resource consumption of the filter chains detailed in Table 6. The maximum operational frequency of this architecture ranges from 93.11 MHz to 86.92 MHz if only the inner subarray or all subarrays are active, respectively. By operating at 86.92 MHz, and considering single-port memories, up to 43 filter stages can be fetched. The supported number of filter chains, $N_{F_{\text{Stages}}}$, is obtained from equation (11):

$$N_{F_{\text{Stages}}} = \min \left( N_o, N_p, N_f \right) = \min \left( 64, 6, 43 \right) = 6,$$  \hspace{1cm} (16)

where $N_{F_{\text{Stages}}}$ is limited by the available resources. Therefore,

$$t_{F_{\text{SRP}}} = \frac{N_o}{\min \left( N_o, N_{F_{\text{Stages}}} \right)} \cdot \left( t_{\text{filtering}} + t_s \right) = 23.7 \text{ ms},$$  \hspace{1cm} (17)

and the filter stage needs to operate at least at

$$F_p = F_s \cdot \frac{N_{F_{\text{Stages}}}}{N_B} = 12.48 \text{ MHz},$$  \hspace{1cm} (18)

in case all the microphones are active.
Table 8: Definition of the architecture’s parameter involved in the time analysis. $N_s$ is the number of output samples and $N_{am}$ is the number of active microphones.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Definition</th>
<th>Equation</th>
<th>Value (cc/MHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_{CIC}$</td>
<td>II of the CIC filter</td>
<td>$2 \cdot N_{CIC} + 1$</td>
<td>$\frac{9}{F_S} = 4.33 \mu s$</td>
</tr>
<tr>
<td>$t_{DC}$</td>
<td>II of the remove DC</td>
<td>$D_{CIC} + 2$</td>
<td>$\frac{34}{F_S} = 16.35 \mu s$</td>
</tr>
<tr>
<td>$t_{FIR}$</td>
<td>II of the FIR filter</td>
<td>$D_{CIC} \cdot \left( \frac{N_{FIR} + 1}{2} + 1 \right)$</td>
<td>$\frac{544}{F_S} = 261.5 \mu s$</td>
</tr>
<tr>
<td>$t_{Delay}$</td>
<td>II of the delay memories at $F_S$</td>
<td>$\max (\Delta)$</td>
<td>$\frac{1023}{F_S} = 491.18 \mu s$</td>
</tr>
<tr>
<td>$t_{Sum}$</td>
<td>II of the cascaded sums</td>
<td>$2 \cdot \lceil \log_2 (N_{am}) \rceil$</td>
<td>$\frac{12}{F_S} = 5.77 \mu s$</td>
</tr>
<tr>
<td>$t_s$</td>
<td>Sensing time</td>
<td>$D_{CIC} \cdot D_{FIR} \cdot (N_s - 1)$</td>
<td>$\frac{4032}{F_S} = 1.94 \mu s$</td>
</tr>
<tr>
<td>$t_{Filtering}$</td>
<td>II of the filter stage</td>
<td>$t_{CIC} + t_{DC} + t_{FIR}$</td>
<td>$\frac{587}{F_S} = 282.1 \mu s$</td>
</tr>
<tr>
<td>$t_{Beamforming}$</td>
<td>II of the beamforming stage</td>
<td>$t_{Delay} + t_{Sum}$</td>
<td>$\frac{1035}{F_S} = 497.59 \mu s$</td>
</tr>
<tr>
<td>$t_{Power}$</td>
<td>II of the SRP</td>
<td>$2$</td>
<td>$\frac{2}{F_S} = 0.96 \mu s$</td>
</tr>
<tr>
<td>$t_{BEAMF} + t_{FILTER} + t_{POWER}$</td>
<td></td>
<td>$\frac{1199}{F_S} = 599.5 \mu s$</td>
<td></td>
</tr>
<tr>
<td>$t_{P-SRP}$</td>
<td>Time to obtain a complete P-SRP</td>
<td>$N_s \cdot (t_s + t_{Delay})$</td>
<td>$\frac{361984}{F_S} = 174.03 \mu s$</td>
</tr>
</tbody>
</table>

Table 9: Performance analysis of the optimized designs when applying and combining the performance strategies. The values are expressed in ms.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Initial</th>
<th>Continuous</th>
<th>Parallel continuous time multiplexing</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_{P-SRP}$</td>
<td>174 ms</td>
<td>142.1 ms</td>
<td>23.7 ms</td>
</tr>
</tbody>
</table>

5.2.6. Performance Analysis. Table 10 summarizes the achievable performance based on the performance strategy. Notice that the power-efficient architecture analysed here achieves a higher performance than the high-performance architecture in [9, 10] when no strategy is applied. The performance difference is because the configuration of the power-efficient architecture under evaluation targets a different $F_{max}$. The difference between both $F_{max}$ values leads to different $F_S$ and $D_s$, which directly affects $t_{P-SRP}$.

The performance strategies are less effective for the power-efficient architecture. Although the initial time is reduced when continuously acquiring data, it is still higher than that in the high-performance architecture described in [9, 10] because of the $t_{Filtering}$ overhead.

The last strategy proposed in Section 5.1 fully exploits the limited resources. The achievable performance is several orders of magnitude lower than the one of the high-performance architecture described in [9, 10]. This is mainly because of two factors. Firstly, the evaluation of the power-efficient architecture is done on a Flash-based FPGA with a lower amount of resources than the one used to evaluate the one in [9, 10]. Secondly, the power-efficient architecture presents certain performance limitations which cannot be fully solved through the performance strategies.

The evaluation of the power-efficient architecture and the analysis done in [10], targeting different application requirements and with different types of FPGAs, demonstrate how the architectures and their performance strategies can be applied to any FPGA-based architecture.

5.3. Summary. The power-efficient architecture represents an alternative when power consumption is a priority. Due to its low demand in resources, this architecture is able to be implemented on Flash-based FPGAs. These FPGAs are very power efficient, but they lack in the number of available resources. Reduced performance of the sound locator is the price to pay for the power efficiency. Although several strategies can certainly accelerate the architecture’s response, the limited FPGA resources and the architecture characteristics bound further acceleration.

6. Comparison of the Reconfigurable Architectures

This section presents a comparison of the high-performance architecture described in [10] and the proposed power-efficient architecture detailed in Section 5. Both architectures are evaluated on a Zynq-based platform...
of the array activities along with the 95% confidence interval at all the 64 supported orientations. The average of all directivities for the supported orientations. Figure 20(a) depicts a comparison of the resource consumption of both architectures when targeting a Zynq 7020 FPGA. Although the low resource consumption of the power-efficient architecture allows the use of a smaller and lower demanding power FPGA, the Zynq 7020 FPGA is used in order to fairly compare both architectures. The amount of different types of resources demanded by the proposed architecture is significantly lower than the architecture presented in [9, 10]. The low resource consumption is possible thanks to the reduction of the number of filter chains, leading to a more efficient beamforming operation in terms of resources. Whereas each microphone in the high-performance architecture has an individual filter chain, the power-efficient architecture only needs one.

This percentage decreases to 14.7% and 32.8% of the consumed registers and Look-Up Tables (LUTs), respectively, in the power-efficient architecture. An efficient

6.2. Resource Consumption. The power-efficient architecture presents a significantly lower resource consumption compared to the high-performance architecture. Figure 21 depicts a comparison of the resource consumption of both architectures when targeting a Zynq 7020 FPGA. Although the low resource consumption of the power-efficient architecture allows the use of a smaller and lower demanding power FPGA, the Zynq 7020 FPGA is used in order to fairly compare both architectures. The amount of different types of resources demanded by the proposed architecture is significantly lower than the architecture presented in [9, 10]. The low resource consumption is possible thanks to the reduction of the number of filter chains, leading to a more efficient beamforming operation in terms of resources. Whereas each microphone in the high-performance architecture has an individual filter chain, the power-efficient architecture only needs one.

This percentage decreases to 14.7% and 32.8% of the consumed registers and Look-Up Tables (LUTs), respectively, in the power-efficient architecture. An efficient

---

**Table 10: Summary of the achievable performance and the related parameters when using the performance strategies proposed in Section 5.1.**

<table>
<thead>
<tr>
<th>Strategy</th>
<th>Parameter</th>
<th>Inner 4 mics</th>
<th>Inner 12 mics</th>
<th>Inner 28 mics</th>
<th>All 52 mics</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>$F_p$ (MHz)</td>
<td>$F_p$ (MHz)</td>
<td>$F_p$ (MHz)</td>
<td>$F_p$ (MHz)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Performance (Or/s)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$F_s$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Continuous processing</td>
<td>$F_p$ (MHz)</td>
<td>$F_p$ (MHz)</td>
<td>$F_p$ (MHz)</td>
<td>$F_p$ (MHz)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Performance (Or/s)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$F_s$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Parallel continuous time multiplexing</td>
<td>$F_{op}$ (MHz)</td>
<td>93.11</td>
<td>88.02</td>
<td>91.08</td>
<td>86.92</td>
</tr>
<tr>
<td></td>
<td>$N_{Bstages}$</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Performance (Or/s)</td>
<td>2.70E+03</td>
<td>2.70E+03</td>
<td>2.70E+03</td>
<td>2.25E+03</td>
</tr>
</tbody>
</table>
memory partition is possible thanks to the storage of PDM signals and the use of LUTs as internal memory. LUTs are the constraint resource in the high-performance architecture, justifying the increment of the consumption of dedicated internal memories (BRAMs). In this regard, BRAMs are the limiting resource for the power-efficient architecture. Due to the relatively low consumption of LUTs, the consumption of BRAMs can be certainly reduced if LUTs are used as internal memory in the beamforming stage. In fact, BRAMs are only required to store the incoming samples from the 24 microphones of the outer subarray. The available resources in the Zynq 7020 support up to 11 instantiations of the power-efficient architecture, which represents the capacity to compute the incoming signal from more than 500 microphones simultaneously.

6.3. Timing and Performance. Figure 22 depicts a comparison of $t_{P-SRP}$ for both architectures defined by the supported configurations detailed in Section 3.2.2. No performance strategy has been applied. Notice how the values of $t_{P-SRP}$ differ with approximately 33 ms for each supported configuration. Although $t_{P-SRP}$ is equally defined for both architectures, the II values of the beamforming stage vary due to $\max(\Delta)$, which is also reflected in the memory requirements (Figure 9).

The placement of the beamforming stage into the architecture does not only affect the frequency response but also directly determines the achievable performance. The performance strategies, proposed in [10] and in Section 5.1, accentuate the architectural differences. Figure 14 exemplifies the relevance of the beamformer placement from the performance perspective. Although both architectures support the same performance strategy of the continuous processing, the position of the beamforming stage determines what II of the components affects the increment of $t_{P-SRP}$. Whereas in the high-performance architecture only the II of the power stage increases $t_{P-SRP}$, the II of the filter stage and additional clock cycles to reset the filters also increase $t_{P-SRP}$ in the power-efficient architecture.

Figure 23 shows the values of $t_{P-SRP}$ for the same design space when applying the parallel continuous time
multiplexing performance strategy. The variation of $t_{P-SRP}$ based on $F_{\text{max}}$ reflects the dependency of $t_{P-SRP}$ on $F_S$ for the power-efficient architecture. This dependency, however, disappears for the high-performance architecture, where $t_{P-SRP}$ only depends on $F_{\text{max}}$. Such a characteristic reduces the dependency on $F_S$, which corresponds to the microphone’s clock, during the design stage of a high-performance architecture.

Figure 22: Evolution of $t_{P-SRP}$ for each architecture when no performance strategy is applied. The explored design space corresponds to the one depicted in Figure 6.

Figure 23: Evolution of $t_{P-SRP}$ for each architecture when the best performance strategy is applied. The explored design space corresponds to the one depicted in Figure 6.
Table 11 summarizes the parameters involved in the calculation of this performance strategy for both architectures. The achievable performance expressed in Or/s decreases when increasing \( N_{\text{am}} \) on both architectures. Notice that, although the power-efficient architecture is able to operate at a higher frequency and to allocate more instances, the achievable performance of the high-performance architecture outperforms the power-efficient architecture despite both target the same FPGA.

6.4. Summary. Table 12 summarizes the comparison of the proposed power-efficient architecture and the related works from a timing and power consumption point of view. As a consequence of the lower resource consumption, not only larger microphone arrays can be processed in parallel but also more power-efficient FPGAs can be used to minimize the power consumption. The main difference in power consumption between the analysed architecture and the one described in [8] is in the microphones’ power consumption due to operating at a different operational mode. The power-efficient architecture presents a major reduction of the power consumption when compared to the high-performance architecture described in [10], achieving the lowest power per microphone ratio when all the subarrays are active. Although the power-efficient architecture is slower than the high-performance architecture, the time-per-microphone ratio is better than other related solutions.

7. Conclusions

FPGA-based embedded systems offer sufficient flexibility to support dynamic acoustic beamformers able to achieve real-time performance or power efficiency. Nevertheless, these desirable features are only achievable through design considerations and performance strategies to fully exploit the FPGA’s characteristics. On the one hand, design considerations lead to compromises on the selection of the architecture’s components, grouped in stages, to obtain the desired response. The position of the beamforming stage, for instance, does affect not only memory requirements but also performance, frequency response, and resource consumption of the architecture. The specifications of the PDM demodulation present a limited impact on the performance and resource consumption. On the other hand, performance strategies enable a higher performance by exploiting the FPGA’s resources. Although such performance strategies are strongly dependent on the reconfigurable architecture, they are capable to significantly increase the performance of reconfigurable acoustic beamforming architectures.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.
Conflicts of Interest

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work was supported by the European Regional Development Fund (ERDF) and the Brussels-Capital Region-Innoviris within the framework of the Operational Programme 2014-2020 through the ERDF-2020 project ICITY-RDI.BRU.

References


