A Multimode SoC FPGA-Based Acoustic Camera for Wireless Sensor Networks

Bruno da Silva*†‡, Laurent Segers*, Yannick Rasschaert*, Quentin Quevy*, An Braeken* and Abdellah Touhafi*†
*Department of Industrial Sciences (INDI), Vrije Universiteit Brussel (VUB), Brussels, Belgium
†Department of Electronics and Informatics (ETRO), Vrije Universiteit Brussel (VUB), Brussels, Belgium
‡Department of Electronics and Information Systems (ELIS), Ghent University (UGent), Ghent, Belgium

***Abstract*—Acoustic cameras allow the visualization of sound sources using microphone arrays and beamforming techniques. The required computational power increases with the number of microphones in the array, the acoustic images resolution, and in particular, when targeting real-time. Such computational demand leads to a prohibitive power consumption for Wireless Sensor Networks (WSNs). In this paper, we present a SoC FPGA based architecture to perform a low-power and real-time accurate acoustic imaging for WSNs. The high computational demand is satisfied by performing the acoustic acquisition and the beamforming technique on the FPGA side. The hard-core processor enhances and compresses the acoustic images before transmitting to the WSN. As a result, the WSN manages the supported configuration modes of the acoustic camera. For instance, the resolution of the acoustic images can be adapted on-demand to satisfy the available network's BW while performing real-time acoustic imaging. Our performance measurements show that acoustic images are generated on the FPGA in real time with resolutions of 160x120 pixels operating at 32 frames-per-second. Nevertheless, higher resolutions are achievable thanks to the exploitation of the hard-core processor available in SoC FPGAs such as Zynq.**

I. Introduction

Acoustic cameras visualize the intensity of sound waves, which is used to be graphically represented as an acoustic heatmap, allowing the identification and localization of sound sources. Arrays of microphones are used to collect the acoustic information from certain beamed directions by applying beamforming techniques. The relatively low-cost of the Micro-Electromechanical Systems (MEMS) microphones together with recent advances in the MEMS technology facilitates the construction of large MEMS microphone arrays with reasonable quality in their acoustic response [1]. As a result, the development of acoustic cameras composed of tens of MEMS microphones have became popular in the latest years. Nevertheless, the computational demand increases with the number of microphones present in the array, becoming a challenge when targeting real-time. Due to the high I/O capability required to interface such microphone arrays, the high level of parallelism presented in such a systems and the relative low-power that Field-Programmable Gate Arrays (FPGAs) offer nowadays, most of the acoustic cameras use this technology to compute the needed operations for acoustic imaging. The use of microphone arrays for acoustic imaging, however, has barely been considered for Wireless Sensor Networks (WSN) applications due to the power constraints and the limited bandwidth (BW) that WSN present.

In this paper, we propose the use of a Xilinx Zynq architecture to enable the use of acoustic cameras for WSN applications. Such System-on-Chip (SoC) FPGA architecture not only provide relatively large reconfigurable logic resources in the Programmable Logic (PL), but also a hard-core general purpose processor in the Processing System (PS) in the same die, enabling a fast communication between both. Our solution exploits the heterogeneous nature of the Zynq architecture by generating real-time acoustic images on the PL while alleviating the WSN's BW limitations by performing acoustic imaging processing locally on the PS. As a result, our proposed architecture supports multiple configuration modes, which are managed by the WSN through the hardcore processor in order to adapt the response of the system to the network's context.

The main contributions of this work can be summarized as follows:

- A SoC FPGA-based architecture for real-time acoustic imaging.
- Multiple operational modes to satisfy the bandwidth WSN demands.
- The use of image enhancement techniques and the detection of Regions-of-Interest ($ROIs$) in a SoC FPGA architecture.

This paper is organized as follows. Section II presents related work. In Section III our approach is introduced. Section IV describes the microphone array and the generation of real-time acoustic heatmaps on the PL. The description of the operations computed on the PS as well as the multiple modes supported is done in Section V. In Section VI, our SoC FPGA architecture is evaluated. Finally, our conclusions are presented in Section VII.

II. Related Work

Similar related work and the main differences with the proposed architecture are discussed here.

The FPGA-based architecture proposed in [2] fully integrates all the operations needed to generate acoustic heatmaps in a Xilinx Spartan 3E FPGA. Despite their architecture achieves up to 10 frames-per-second (FPS) for acoustic image resolutions of 320x240 pixels, their architecture does not include any filter further than the inner filtering during the ADC conversion of the incoming data from their analogue

978-1-5386-3344-1/17/$31.00 2018 IEEE

electrec microphones. Furthermore, the acoustic images include ultrasound acoustic information since the frequency response reaches up to 42 kHz due to a missed high-pass filtering stage. Our architecture reaches the same performance while generating lower resolution images on the PL to reach real-time. Nevertheless, our proposed architecture is able to increase the image resolution on the PS based on the WSN demand.

The authors in [3] implemented a 3D impulsive sound-source localization method on combining one FPGA with a PC. Their system computes the delay-and-sum beamforming operation on the PC while the FPGA filters the acquired audio signals and displays through VGA the acoustic heatmap generated on the PC. Instead, our architecture is fully embedded on the Zynq device the filtering and the beamforming operations and the generation of the acoustic heatmap.

An architecture targeting a SoC FPGA is presented in [4]. The high parallelism offered by the FPGA part is used to perform the filter operations needed to retrieve the audio signals. The embedded processor, a dual core ARM processor, handles the user communication. Our architecture, instead, fully exploits the hard-core processor, enhancing and locally processing the acoustic images further than managing the communication. Moreover, the main goal of their architecture is to generate acoustic images from a broadband acoustic signals ranging from 20 kHz to 80 kHz. This range scopes out of the audible range of human hearing and belongs to ultrasound range. The proposed architecture uses only audible acoustic signals to construct the acoustic heatmap.

The authors in [5] propose an architecture based on alternative Cascaded Recursive-Running Sum (CRRS) filters as replacement of the commonly used Cascaded Integrator-Comb (CIC) filters for acoustic signal processing applications. These filters are evaluated on a real-time acoustic camera fully implemented on an FPGA. Nevertheless, the authors do not provide information about the target resolution of their acoustic camera neither specifications related to the overall performance.

The system proposed in [6] combines a PC, an FPGA, an embedded processor and a GPU to generate acoustic images using a planar MEMS microphone array. Despite the high-quality of the obtained acoustic images and the use of a larger microphone array, the system does not generate real-time acoustic images. Moreover, the distributed nature of their system to generate acoustic images hinders the fully embedding in a compatible system for WSN applications.

To our knowledge, our SoC FPGA based architecture is the first one to fully exploit the combination of the reconfigurable logics and the hard-core processors available in the current SoC FPGA while targeting WSNs.

III. Proposed SoC FPGA-Based Architecture

The proposed architecture intends to exploit the combination for the PS and the PL of the Xilinx Zynq architectures to extend the use of acoustic cameras to WSNs. While the reconfigurable logic on the PL satisfies the low-power demands of WSNs, it also provides enough computational power to produce acoustic images in real-time. On the other hand, the PS not only provides the necessary control to interface WSN but also the flexibility to support multiple configurations without the need to partially reconfigure the PL logic. The computational balance between both components presents, however, several trade-offs that must be analyzed before reaching the truly potential of SoC FPGAs for this particular application. Moreover, the presented architecture supports multiple modes, which are decided by the WSN and managed by the PS, to better respond to the WSN's conditions. Figure 1 depicts the proposed distribution of the computations between the PS and the PL. The main components of the proposed WSN node to produce acoustic images are the microphone array, the PL and the PS parts of the Zynq architecture, and the WSN mote. The microphone array and the PL compose the *front-end* while the PS and the WSN mote are the *back-end*.

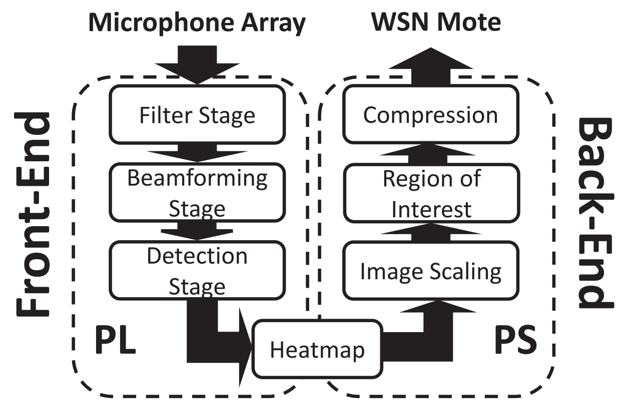

Fig. 1: *Distribution of the components into the Zynq Processor.*

At the *front-end*, the PL receives the acquired acoustic signal from the microphone array. The audio signal is retrieved from the microphone's acquired signal after a filtering process performed in the filter stage. The beamforming stage aligns the audio signals in order to focus into a particular orientation while discriminating the inputs from other orientations. The sound relative power (SRP) is calculated at the detection stage. The SRP values obtained for each orientation are propagated to the PS part to be represented as an acoustic heatmap. The Xillybus [7] is used for the communication between the PL and the PS part, achieving a BW of 103MB/s [8].

The *back-end* performs the local image processing and manages the WSN communication. Moreover, several image enhancement operations are supported on the PS. These operations involved the generation of the heatmap from the values generated on the PL, the scaling of the image, the identification of $ROIs$ and the image compression. The SRP values of the 3D beamforming are graphically represented in a heatmap format. The heatmap resolution determines the number of orientations (No) performed by the beamformer. While a low value of No leads to higher number of frames per second (FPS), low resolutions are supported to satisfy the

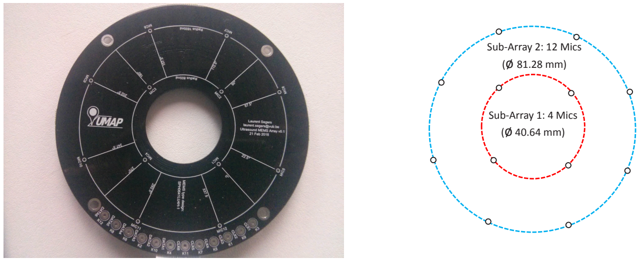

Fig. 2: *The microphone array consists of 12 digital MEMS microphones arrange in two concentric sub-arrays.*

Parameter	Definition	Value
F_s	Sampling Frequency	3.125 MHz
F_{min}	Minimum Frequency	1 kHz
F_{max}	Maximum Frequency	16.275 kHz
BW	Minimum BW to satisfy Nyquist	32.55 kHz
D_F	Decimation Factor	96
D_{CIC}	CIC Filter Decimation Factor	24
N_{CIC}	Order of the CIC Filter	4
D_{FIR}	FIR Filter Decimation Factor	4
N_{FIR}	Order of the FIR Filter	24

TABLE I: small *Configuration of the architecture under analysis.*

real-time constraints. The presented architecture offers a trade-off in terms of performance and image resolution. Despite a relatively low resolution acoustic heatmap is performed at the FPGA side to provide a real-time response, image scaling operations are supported on the PS to improve the image resolution. Moreover, multiple modes are supported in order to adapt the image operations on the PS to satisfy the WSN demands. For instance, sound sources can be identified in the heatmap, where $ROIs$ are marked to be lately profiled. The identified $ROIs$ and it's coordinates are compressed and sent to the network by the WSN mote. In this operational mode, the overall WSN BW consumption is reduced.

IV. Front-End

A. *Microphone Array*

The microphone array consists of 12 MEMS microphones SPH0641LU4H-1 [9] provided by Knowles placed in 2 sub-arrays (Figure 2). All microphones are bottom layer mounted on a printed circuit board (PCB) with the aperture hole facing upwards in the top layer. In order to reduce acoustic scattering, all other components are mounted on the bottom layer of the PCB. The output of the microphones is a PDM signal, which is internally obtained in each microphone by a sigma delta modulator typically running between 1 and 3 MHz. All microphones are paired such that 6 clock and 6 data lines are required to interface the FPGA, which is done through two PMOD connectors. Despite the SPH0641LU4H-1 microphones are also suitable for ultrasound applications, in this paper we only consider the audible acoustic frequencies. The shortest distance between the microphones is 23.20 mm and the longest distance equals 81.28 mm. These distances respectively correspond to acoustic frequencies ($\frac{\lambda}{2}$) of 7.392 kHz and 2.110 kHz.

Detecting the direction of arrival of sound waves with microphone arrays is applied by using a variation of the Delay-and-Sum beamforming technique, which relies on the principle of aligning the recorded sound samples in time before to sum them. In order to properly delay the incoming sound samples, a delay table with delays for each microphone in all desired beamforming directions is calculated. Our acoustic camera uses an adapted hypercube distribution [19] to the field-of-view of the camera, which is 51°. Here, only a portion corresponding to the 51° in the xy-plane laying at $z = 1$ is used. A rectangular grid is then taken in this section and all obtained points are then normalized to obtain unitary vectors which are used to calculate the required delays.

B. *A Filter-Delay-Decimate-and-Sum Architecture on the PL*

The proposed architecture running on the PL is based on the one presented in [11], and accelerated in [12]. These filter-delay-and-sum architectures offer a response fast enough to satisfy the performance demands of an acoustic camera. Unfortunately, the price to pay is a relative degradation in the accuracy of the beamforming, reflected in a relatively poor frequency response [13]. The proposed architecture achieves the same performance than in [12] while improving the frequency response. The architecture parameters are detailed in Table I.

Figure 3 depicts the inner components of the three stages of the architecture implemented on the PL. The complete architecture is processing in streaming and pipeline all the operation within each stage.

1) Filter Stage: The first stage is the filter stage, which is composed of multiple filter chains. The MEMS microphones of the array provide an oversampled PDM signal that needs to be processed to retrieve the original audio signal. Each microphone is associated to a filter chain, which is composed of a cascade of filters to reduce the signal BW and to remove the high frequency noise. The first filter is a 4th order low pass CIC decimator filter with a decimation factor of 24. This type of filter has a lower resource consumption since it only involves additions and subtractions [14]. The CIC filter is followed by a moving average filter to remove the DC offset introduced by the MEMS microphone. The last component of each filter chain is a 23th order low-pass FIR filter. The serial design of the FIR filter drastically reduces the resource consumption but forces the maximum order of the filter to be equal than the decimation factor of the CIC filter. The data representation used in the filter chain is a signed 32-bits fixed point representation with 16 bits as fractional part. Nevertheless, the bitwidth is increased inside the filters to minimize the quantization errors that the internal filter operations might be introduced. The data representation is set to signed 32-bits at the output of each filter by applying the proper adjustment. Finally, the FIR filter's coefficients are represented with 16 bits.

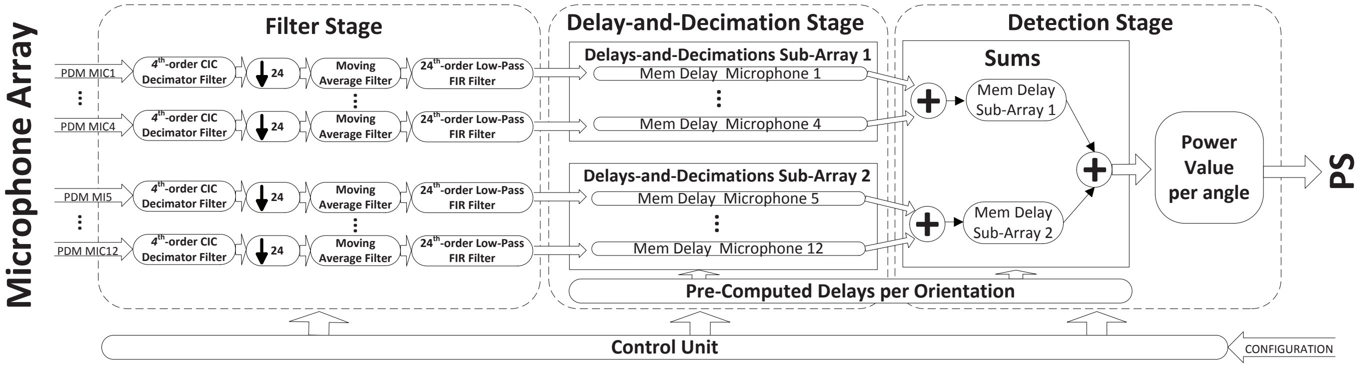

Fig. 3: *Overview of the FPGA's components. The PDM input signal is converted to audio in the cascade of filters. The Delay-and-Sum beamforming is composed of several memories, associated to each sub-array to disable those memories linked to deactivated microphones, to properly delay the input signal. The SRP is finally obtained per orientation.*

2) Beamforming Stage: The beamforming techniques provide directionality to the microphone array. Such type of techniques allow to focus the array to a specific orientation while suppressing the acoustic data coming from other directions. The presented architecture uses the Delay-and-Sum beamforming to focus the array to pre-configured orientations, which are determined at pre-compile time based on the desired resolution of the acoustic image. The filtered audio from the filter stage is stored in banks of block memories (BRAM) in order to be delayed by a specific amount of time determined by the focus direction, the position vector of the microphone, and the speed of sound [10]. All possible delays are precomputed, grouped based on the supported beamed orientations, and stored in BRAM during the compilation time. In order to support a variable number of active microphones (N_a), the implementation of the beamforming operation groups in sub-arrays the incoming signal of microphones. Therefore, the beamforming operation is only executed on the active sub-arrays, disabling all the operations associated to the inactive microphones in order to reduce the power consumption.

3) Detection Stage: At the last stage, the delayed values from the beamforming stage are accumulated before the calculation of the SRP per orientation in the time domain. The computation of SRP for different beamed orientations is used at the PS to generate a heatmap. Thus, these orientations presenting a higher SRP correspond to the location of potential sound sources.

C. Trade-offs

The proposed architecture combines the beamforming operation with a downsampling operation. While the architectures in [11], [12] and [13] downsample the filtered signal just after the FIR filter in the filter chain, the presented architecture downsamples during the beamforming operation. The sampling frequency at the beamforming stage in [10] and [13] equals the clock frequency of the MEMS microphones (2 MHz). The architectures in [11] and in [12] present, however, a lower sampling frequency at the beamforming stage (31.25 kHz). The accuracy in the first architectures is higher than in the latest ones because the delay unit at the beamforming stage is inversely proportional to the sampling frequency at this stage. Therefore, the architectures in [10] and [13] offer higher accuracy than the architectures in [11] and in [12]. Nevertheless, the price to pay is the higher latency. Our architecture solves the latency drawback by increasing the memory consumption at the beamforming stage.

1) Performance: The proposed architecture is an intermediate solution where the highest performance is achieved while preserving the level of accuracy. One of the main differences of the presented architecture and the architectures in [11], [12] is the location of the FIR filter decimation of a factor of D_{FIR} after the beamforming to increase accuracy. Thus, the accuracy of the beamforming is increased but the strategies proposed in [12] cannot be applied, drastically increasing the overall latency. Moreover, the D_{FIR} values read from the BRAMs at the beamforming stage are discarded by the detection stage. Instead, our architecture decimates while beamforming. The read operation of the beamforming memories has increments of D_{FIR}, which is equivalent to decimation. On the one hand, this solution allows to perform at the same speed than the architectures in [12] while increasing by a factor of D_{FIR} the accuracy at the beamforming stage. On the other hand, the memory requirements at the beamforming stage are increased by a factor of D_{FIR} due to all the undecimated filtered values that must be stored in the beamforming memories.

2) Frequency Response: A higher accuracy at the beamforming stage directly affects to the overall frequency response of the architecture. Figure 4 depicts the comparison of the architectures in [12], [13] and the proposed one. Each architecture has been evaluated for one sound source from 100 Hz to 12 kHz, with the same design parameters (Fs, D_F, ...) and considering 64 orientations in 2D for the SoundCompass microphone array [10]. The quality of the frequency response of each architecture is measured based on the directivity (D_P), which reflects the ratio between the main lobe's surface and the total circle in a 2D polar map [10]. The average of all directivities along with the 95% confidence interval is calculated for 64 orientations. Moreover, the resulting directivities are based

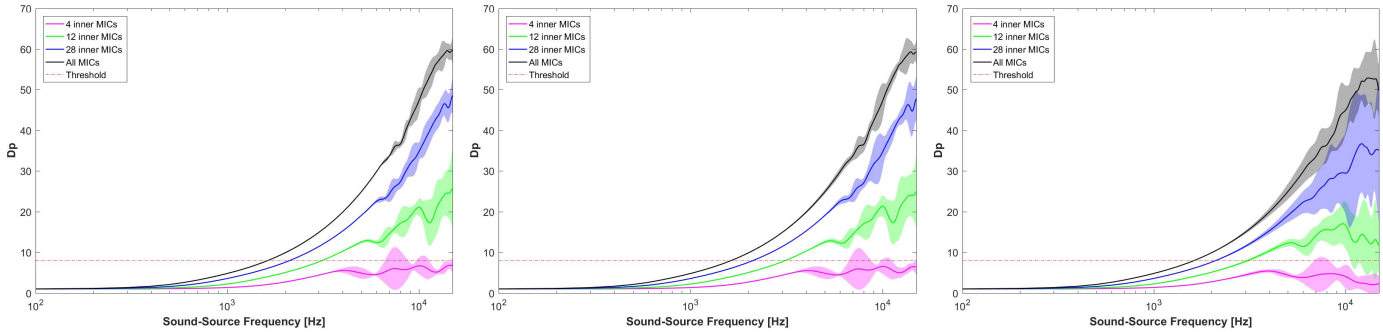

Fig. 4: *Comparison of the architecture in [13] (left), the proposed architecture (centre) and the architecture in [12] (right) using the 2D directivity [10] as metric.*

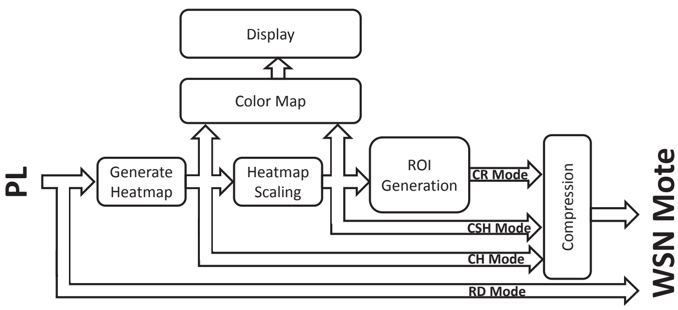

Fig. 5: *Overview of the image processing steps executed on the PS. Multiple modes are supported to satisfy the most constrained WSN demands.*

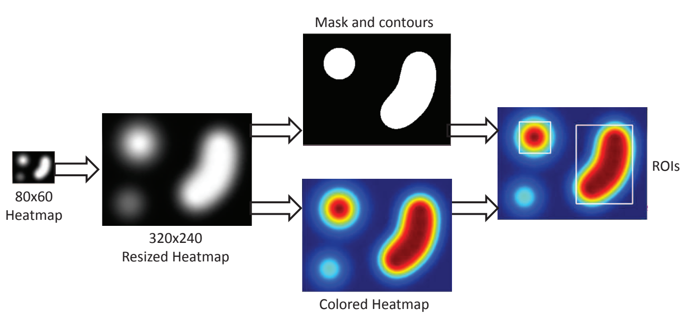

Fig. 6: *Operations needed to identify $ROIs$.*

on the active sub-arrays of the original SoundCompass for the proposed architecture.

The proposed architecture provides a slightly worst D_P than the architecture in [13] while performing as fast as the architecture in [12]. Nevertheless, the cost is a higher internal memory consumption in order to store D_{FIR} more delayed values per microphone.

V. Back-End

A. Acoustic Image Enhancements on the PS

Our prototype runs Xillinux 2.0 [7], a Linux OS (Ubuntu 16.04) on the PS to enable a graphical use of the C++ OpenCV library (ver. 2.4.13.6) [15], which contains optimized functions for computer vision applications. Figure 5 depicts our C++ OpenCV-based applications used by the PS to construct an acoustic heatmap from the FPGA data, to generate the $ROIs$ and to compress the results. The complete dataflow starts with the generation of a relatively low resolution heatmap. The values from the FPGA are placed in an $H \times W$ matrix, where H and W are the height and the weight of the target heatmap resolution respectively. The communication with the PL is via Xillybus [7], which is basically composed of FIFO buffers. The application can then read the data generated on the PL from this buffer. Once all data is received to construct one acoustic heatmap frame. Depending on the selected operational mode, the enhancement of the frame begins with the rescaling of the heatmap. The rescaled heatmap is optionally displayed by using the *imshow* function from OpenCV at this stage, after applying a colormap to the grayscale heatmap by using the *applycolormap* function from OpenCV. Based on the selected mode, acoustic events are detected using the grayscale heatmap to identify the $ROIs$ (Figure 6). To detect acoustic events, a threshold is applied to the grayscale heatmap to form a mask. All values above the threshold become 1, the other values 0. Thanks to this mask, it is possible to find the contours of the sound sources louder than the selected threshold. From these contours, the corner coordinates are found using the function *boundingrect* from OpenCV and $ROIs$ are extracted from the heatmap. Each ROI is then compressed in JPEG format and all the $ROIs$ in the same frame are further compressed before to be propagated to the WSN. Such approach allows a more efficient consumption of the BW of the network.

The acoustic image enhancement is not the only task performed on the PS. The support of multiple modes (Figure 5) is managed on the PS side based on the received WSN commands. The supported modes are the following:

1) Raw Data (RD): The raw data from the PL is not processed on the PS before being transmitted by the WSN mote.
2) Compressed Heatmap (CH): The data from the PL is compressed and transmitted by the WSN mote.
3) Compressed Scaled Heatmap (CSH): A grayscale scaled heatmap is transmitted by the WSN mote.

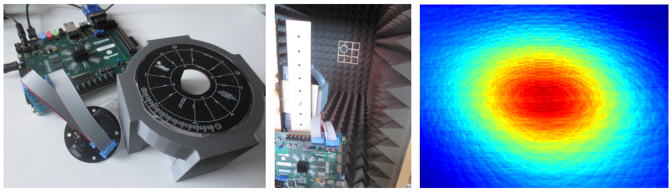

Fig. 7: *(Left) Zedboard and the microphone array used for our measurements. (Center) Special anechoic boxes are used to evaluate our acoustic camera. (Right) Typical acoustic heatmap of* 160×120 *pixels without scaling.*

4) Compressed ROI (CR): $ROIs$ are identified and compressed together with their location in the image to be transmitted by the WSN mote.

The optional visualization mode is not included in the list above since it is not a WSN feature. This visualization mode allows the user not only to visualize the acoustic images but also to record video in the local SDCard.

B. WSN Communication

The acoustic image enhancements performed on the PS compensate the lack of BW that WSN provides while enriching the acoustic information computed by this acoustic WSN node. Moreover, the chosen WSN technology determines the processing speed needed in the SoC FPGA. Current WSN communication systems can be classified by range, data rate, network topology, network size, power consumption, etc. Our selection, however, targets WSNs with high data transmission rates, usually with limited ranges, and relatively low power consumption

VI. Experimental Results

Our experiments evaluate the response of the microphone array, the resource consumption and the performance of the architecture and the overall performance of the system for WSN. The supported modes have been evaluated in a stand-alone node without the WSN mote.

A. Experimental Setup

Figure 7 (left) shows the experimental platform, which is composed of a Digilent Zedboard interconnected to the microphone array through two PMOD connectors. An Anechoic Box (Abox) [16] is used to evaluate our acoustic camera. The Aboxes are designed to facilitate the evaluation of acoustic WSN nodes, allowing the early identification of possible incongruities in the beginning of the WSN development cycle [16]. A wooden and a 3D printed structure are designed to hold the sound source(s) (speaker(s)) against the wall, without damaging the foam. Figure 7 (centre) shows our setup, where a single speaker is placed at multiple positions inside the box and connected via Bluetooth to a PC where the acoustic signals under evaluation are generated. Figure 4 (right) depicts an acoustic heatmap of 160×120 pixels without scaling obtained using one loudspeaker generating 4 KHz and placed around half-meter in front of the microphone array.

Resources	Available	Utilization	Percentage
Registers	106400	29447	27.67 %
LUTs	53200	19538	36.72 %
BRAM18k	140	33	23.57 %
DSP48	220	28	12.72 %

TABLE II: *Zynq 7020 resource consumption after placement and routing of the proposed architecture.*

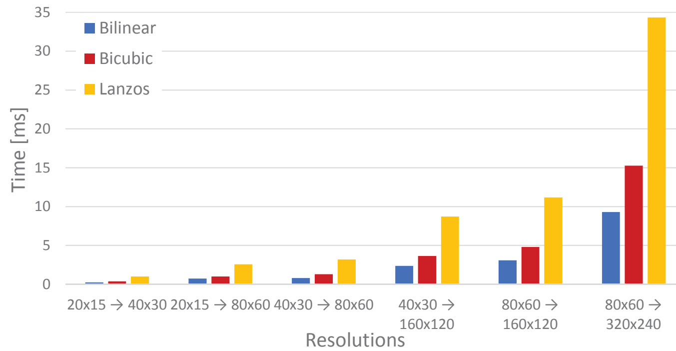

Fig. 8: *Average timing of the different scaling methods. The values have been obtained after 1000 executions with scaling factors ranging from 2 to 4.*

B. Resource Consumption

Table II summarizes the resource consumption on the PL. The storage of the pre-computed orientations dominates the consumption of LUTs, while the streaming and pipelined implementation of the architecture increases the consumption of registers. The consumption of DSPs, on the other hand, is mainly produced in the filter stage. Despite the relatively large resource consumption of our architecture, we believe that further optimizations can significantly reduce the overall resource consumption. For instance, the pre-computed delays necessary to support thousands of orientations during the beamforming operation can be generated on-the-fly on the PL or stored in the external memory.

The relative low resource consumption of the architecture enables the migration of the system to a more power efficient SoC FPGA device like the Flash-based SoC FPGA considered in [13]. Unfortunately, despite such low-power Flash-based SoC FPGAs like the Microsemi's SmartFusion2 promise a low power consumption as low as few tens of mW, such devices embed an ARM Cortex-M3 microcontroller, which is not powerful enough to support a OS compatible with C++ OpenCV libraries.

C. Performance Analysis

1) Evaluation of the Scaling Methods: The OpenCV function *resize* scales the image to a desired resolution. This can be done by multiple methods [17], [18]:

1) Nearest-neighbour
2) Bilinear
3) Bicubic
4) Lanczos

Resolution	No	t_{frame} [ms]	FPS
40×30	1200	1.92	520.8
80×60	4800	7.68	130.2
160×120	19200	30.72	32.5
320×240	76800	122.88	8.1

TABLE III: *Performance of the supported resolutions on the PL.*

The timing measurements of these methods running on the PS are detailed in Figure 8. The average values are obtained after 1000 executions and with scaling factors ranging from 2 to 4. Thus, an image of 20×15 is scaled to 80×60, an image of 40×30 is scaled to 160×120, and so on. The *Nearest-neighbour* method has been discarded since it only selects the value of the nearest pixel without performing interpolation. Despite being the fastest method, its output images are highly pixelated. The *Bilinear* method is the fastest of the other three. This method calculates a new pixel value by taking a weighted average of the four nearest neighbouring original pixel values. A smoother result than the *Nearest-neighbour* is obtained at the cost of undesired lines. The *Bicubic* interpolation provides the best visual result, but also is the more time demanding algorithm. Each new pixel is calculated by the bicubic function using the 16 pixels in the nearest 4x4 neighborhood. The result is a smooth heatmap image. Lastly, the *Lanczos* method is also supported. This interpolation method is based on the sinc function but it demands roughly the double of time to resize an image than the *Bicubic* method. Despite the result is closer to the *Bicubic* method, some artefacts might appear in the rescaled image. Due to the performance/quality trade-off, *Bicubic* interpolation is the selected method from here on.

2) PL Performance Analysis: The filtering and beamforming operations at the PL can be adjusted to generate acoustic heatmaps with different resolutions. The latency to process a single beamforming orientation is determined by design parameters like the sensing time (t_s), the sampling frequency (Fs) and the decimation factor (D). The value of t_s is the time the microphone array is monitoring a particular orientation [12] and determines the probability of detection of sound sources under low Signal-to-Noise (SNR) conditions. Therefore, higher values of t_s improve the profiling of the acoustic environment by increasing the overall execution time per frame (t_{frame}). The proposed architecture calculates the SRP with 64 samples, which represents 6144 input PDM samples per orientation. Thus, for the Fs described in I, $t_s \approx 1.96ms$. The latency to calculate the SRP with 64 samples per orientation is 80 clock cycles, independently of the operational frequency.

The beamforming operation is performed at a higher clock frequency than Fs as proposed in [12]. The operational frequency at the beamforming and detection stage is 50 MHz, which corresponds to the Xillibus' clock frequency. Therefore, the time to calculate the SRP per orientation (t_o) is approximately 1.6 μs. Table III details some of the possible heatmaps resolutions and the expected FPS when operating at 50 MHz. In order to reach real time, the time per frame t_{frame} must be between 33.3 ms and 50 ms to reach 30 FPS or 25 FPS respectively. This requirement drastically reduces the maximum heatmap resolution to 160×120. On the other hand, in order to guarantee the independency of each acoustic heatmap, each acoustic image must be generated from the acquired acoustic information in a period higher than $t_s/2$. Therefore, at least 32 out of the 64 samples used to calculate SRP have not been already used to generate one acoustic image. The number of orientations (N_o), which represents the acoustic heatmap resolution, must be high enough to satisfy the independency condition. Therefore, $t_{frame} \geq t_s/2$. This condition limits the minimum supported resolutions because it is only satisfied when $No > 612$ based on the design parameter in Table I and by operating at 50 MHz.

3) PS Performance Analysis: The acoustic image enhancements at the PS side must be computed fast enough to process the acoustic images generated on the PL and to locally compute the acoustic information in order to adapt the acoustic WSN note to the WSN conditions. The detection and compression of $ROIs$ is proposed in order to support higher resolutions while detecting in real time a particular type of acoustic events.

Table IV details the timing needed for the supported modes and the throughput to the WSN mote. The time values depicted in the table are experimentally obtained by running the required image enhancement operations on the PS and include the PL-PS communication overhead. The *CR mode*, which only transmits to the WSN the detected acoustic events, is the most time demanding mode due to the scaling of the acoustic image and the multiple ROI detection. The low time differences between the *CH mode* and the *CSH mode* reflect the low time cost of the bicubic scaling operation. Nevertheless, the image scaling increments the total amount of data to be sent to the WSN, which represents a significantly higher throughput. The throughput values also consider the PL's latency (Table III). As expected, the modes demanding lower amount of computations on the PS reach the higher throughputs. These modes, however, can only be activated for short periods of time due to the extremely limited BW that most of the WSN standards provide.

4) WSN motes: Despite the required BW it is possible, thanks to the techniques mentioned before, to acquire a real time acoustic image. Table V contains motes that we propose to use. The effective implementation of these motes is out of the scope of this paper and considered as future work.

Depending on the power consumption, the communication interface and the distance between motes, the most suitable mote can be selected. BLE112 [20] is a valid solution if images with a high resolution are needed, while the ReMote [21] is a more viable solution for low-power demands.

VII. CONCLUSION

The proposed architecture demonstrates one of the potential uses of SoC FPGA for WSN applications. In particular, the task distribution between the PL and PS of the Zynq

	Resolution	Modes without scaling		Modes with scaling x2					Mode with scaling x4				
		RD	CH	CSH	CR				CSH	CR			
					1	2	4	8		1	2	4	8
Timings [ms]	40 x 30	2.32	2.80	3.82	4.00	4.95	7.14	11.99	7.12	5.51	6.93	9.89	15.67
	80 x 60	9.33	10.48	14.28	12.60	14.11	16.96	22.33	26.88	17.64	20.10	24.94	34.81
	160 x 120	37.09	40.92	55.28	46.03	48.52	53.46	63.35	105.92	65.29	71.01	82.66	105.82
Throughput [kb/s]	40 x 30	16569.22	571.86	628.43	173.86	280.97	389.45	463.84	857.21	235.81	375.36	525.86	663.63
	80 x 60	16464.05	524.78	1890.43	103.14	184.28	306.68	465.68	3229.11	107.72	189.09	304.74	436.69
	160 x 120	16563.82	481.40	1888.72	41.27	78.31	142.16	239.92	3257.15	61.27	112.66	193.56	302.40

TABLE IV: *Experimental average time and throughput of the supported modes.*

	BLE112-A-V1[20]	**ReMote**[21]	**RF266PC1**[22]
Company	Silicon Labs / Bluegiga	Zolertia	Synapse Wireless
Comm. system	Bluetooth 4.0 BLE	Zigbee/ 6LoWPAN	RF (IEEE 802.15.4)
Interface	SPI, UART, USB	SPI, UART, I2C, USB	I2C, SPI
Rate	3 Mb/s	250 Kb/s	2 Mb/s
Power	$<$ 119 mW	$<$ 66 mW	$<$ 429 mW

TABLE V: *Comparison of considered wireless Motes.*

architecture satisfies the computational needs that real-time acoustic imaging applications demand. On the one hand, the PL allows the acceleration of signal processing operations with high parallelism. On the other hand, the PS not only manages the WSN communication through a WSN mote, but also allows the processing of the acoustic information in the SoC FPGA. As a result, the SoC FPGA-based acoustic WSN node supports multiple modes in order to satisfy the most demanding WSN's constraints.

ACKNOWLEDGMENT

This is 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 ICITYRDI.BRU. This work was also partially supported by the CORNET project "DynamIA: Dynamic Hardware Reconfiguration in Industrial Applications" [23] which was funded by IWT Flanders with reference number 140389. Finally, the authors would like to thank Xilinx for the provided software and hardware under the University Program donation.

REFERENCES

[1] Tiete, J., et al. *"MEMS microphones for wireless applications."* Wireless MEMS Networks and Applications. 2017. 177-195.
[2] Zimmermann, B., et al. *"FPGA-based real-time acoustic camera prototype."* Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on. IEEE, 2010.
[3] Seo, S., et al. *"3D Impulsive Sound-Source Localization Method through a 2D MEMS Microphone Array using Delay-and-Sum Beamforming."* Proceedings of the 9th International Conference on Signal Processing Systems. ACM, 2017.
[4] Kerstens, R., et al. *"Low-cost one-bit MEMS microphone arrays for in-air acoustic imaging using FPGA's."* 2017 IEEE SENSORS, October 29-November 1, 2017, Glasgow, Scotland, United Kingdom. 2017.
[5] Sanchez-Hevia, H. A., et al. *"FPGA-based real-time acoustic camera using PDM MEMS microphones with a custom demodulation filter."* Sensor Array and Multichannel Signal Processing Workshop (SAM), 2014 IEEE 8th. IEEE, 2014.
[6] Izquierdo, A., et al. *"Design and evaluation of a scalable and reconfigurable multi-platform system for acoustic imaging."* Sensors 16.10 (2016): 1671.
[7] Xillybus [Online], Available: http://xillybus.com
[8] Lin, Zhongduo, et al. *"Zcluster: A zynq-based hadoop cluster."* Field-Programmable Technology (FPT), 2013 International Conference on. IEEE, 2013.
[9] Datasheet [Online], Available: http://www.knowles.com/eng/content/download/6318/115469/version/1/file/SPH0644HM4H-1+RevB.PDF
[10] Tiete, J.,et al. *"SoundCompass: a distributed MEMS microphone array-based sensor for sound source localization"*. Sensors, 14(2), 1918-1949. 2014.
[11] da Silva, B., et al. *"Runtime reconfigurable beamforming architecture for real-time sound-source localization."* Field Programmable Logic and Applications (FPL), 2016 26th International Conference on. EPFL, 2016.
[12] da Silva, B., et al. *"Design Considerations When Accelerating an FPGA-Based Digital Microphone Array for Sound-Source Localization.* Journal of Sensors 2017 (2017).
[13] da Silva, B., et al. *"A Low-Power FPGA-Based Architecture for Microphone Arrays in Wireless Sensor Networks"*. International Symposium on Applied Reconfigurable Computing. Springer, Cham, 2018.
[14] Hogenauer, E. *"An economical class of digital filters for decimation and interpolation."* Acoustics, Speech and Signal Processing, IEEE Transactions on 29(2): 155-162. 1981.
[15] OpenCV Library. [Online]. Available: https://opencv.org/
[16] Carvalho, F. R., et al. *"ABox: New method for evaluating wireless acoustic-sensor networks."* Applied Acoustics 79 (2014): 81-91.
[17] Ye, Zhen, et al. *"Four image interpolation techniques for ultrasound breast phantom data acquired using Fischer's full field digital mammography and ultrasound system (FFDMUS): a comparative approach."* Image Processing, 2005. ICIP 2005. IEEE International Conference on. Vol. 2. IEEE, 2005.
[18] Sharma, H., et al. *Analyzing impact of image scaling algorithms on viola-jones face detection framework.* Advances in Computing, Communications and Informatics (ICACCI), 2015 International Conference on. IEEE, 2015.
[19] Saff E.B. et al. *"Distributing Many Points on a Sphere."* The Mathematical Intelligencer on 19 (1): 5-11, 1997.
[20] Datasheet [Online], Available: https://www.silabs.com/documents/login/data-sheets/BLE112-DataSheet.pdf
[21] Datasheet, [Online] Available: *http://wiki.zolertia.com/wiki/images/e/e8/Z1_RevC_Datasheet.pdf*
[22] Datasheet [Online], Available: https://static.sparkfun.com/datasheets/Wireless/General/Synapse-RF-Engine-RF266PC1-Data-Sheet.pdf
[23] Mentens, N., et al. *"DynamIA: Dynamic hardware reconfiguration in industrial applications"*, International Symposium on Applied Reconfigurable Computing. Springer, Cham, 2015.