Stochastic Computing Design and Implementation of a Sound Source Localization System

Peter Schober, Seyedeh Newsha Estiri, Sercan Aygun, *Member, IEEE*, Amir Hossein Jalilvand, M. Hassan Najafi, *Member, IEEE*, and Nima TaheriNejad, *Member, IEEE*

***Abstract*— Stochastic computing (SC) is an alternative computing paradigm that processes data in the form of uniform bit-streams. SC is fault-tolerant and can compute on small, efficient circuits. However, SC is primarily used in scientific research, and its practical implementations for end-users are rare. Digital sound source localization (SSL) is a useful signal processing technique that locates speakers using multiple microphones. SC has not been integrated into SSL in practice or theory. In this work, for the first time to the best of our knowledge, we implement an SSL algorithm in the stochastic domain and develop a functional SC-based sound source localizer. The practical part of this work shows that the proposed stochastic circuit does not depend on conventional analog-to-digital conversion and can process data in the form of pulse-width-modulated (PWM) signals. The proposed SC design consumes up to 39% less area than the conventional binary design. It can also consume less power depending on the computational accuracy, for example, 6% less power consumption for 3-bit inputs. We propose a new cross-correlation (CC) design based on the state-of-the-art Sobol bit-streams for further area and power saving. The proposed design utilizes a `MUX` unit for bit-stream generation. It saves the area footprint up to 64% and the power consumption up to 82% compared to the counter-based SC design of CC, which relies on a comparator for bit-stream generation. The presented stochastic circuits, are not limited to SSL and are readily applicable to other practical applications such as radar ranging, wireless location, sonar direction finding, beamforming, and sensor calibration. The project's source code is made available for public access.**

***Index Terms*— Audio processing, cross-correlation, digital design, sound source localization, stochastic computing.**

Manuscript received 22 October 2022; revised 3 January 2023; accepted 1 February 2023. Date of publication 9 February 2023; date of current version 20 March 2023. This work was supported in part by the National Science Foundation (NSF) under Grant 2019511, the Louisiana Board of Regents Support Fund under Grant LEQSF(2020-23)-RD-A-26, and the generous gifts from Cisco, Xilinx, and Nvidia. An earlier version of this paper was presented in part at the International Conference on Computer-Aided Design (ICCAD), San Diego, CA, USA, in 2022 [DOI: 10.1145/3508352.3549373]. This article was recommended by Guest Editor J. Han. *(Corresponding author: M. Hassan Najafi.)*

Peter Schober was with the Institute of Computer Technology, Technische Universitat Wien, 1040 Wien, Austria. He is now with ITK Engineering, 1110 Wien, Austria (e-mail: peter-schober@gmx.at).

Seyedeh Newsha Estiri, Sercan Aygun, Amir Hossein Jalilvand, and M. Hassan Najafi are with the School of Computing and Informatics, University of Louisiana at Lafayette, Lafayette, LA 70503 USA (e-mail: seyedeh-newsha.estiri1@louisiana.edu; sercan.aygun@louisiana.edu; amir.jalilvand@louisiana.edu; najafi@louisiana.edu).

Nima TaheriNejad was with the Institute of Computer Technology (ICT), Technische Universität Wien, 1040 Vienna, Austria. He is now with the Institute of Computer Engineering (ZITI), Heidelberg University, 69120 Heidelberg, Germany (e-mail: nima.taherinejad@tuwien.ac.at).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/JETCAS.2023.3243604.

Digital Object Identifier 10.1109/JETCAS.2023.3243604

I. Introduction

In RECENT years, recording and processing audio is gaining more and more attention as modern communication technology has become a part of our daily lives. Voice control as a human-machine interface is widespread. For example, it appears in smart homes and personal assistants (e.g., Amazon Alexa and Google Assistant). Current laptops and mobile phones are equipped with multiple microphones to record user voices and ambient sound. A set of microphones allows spatial filtering of the recorded sound, which increases the signal-to-noise ratio (SNR) by suppressing most of the noise outside the direction of interest. The process of digitally listening at the speaker's location is known as beamforming which requires the exact position of the speaker [2].

Sound source localization (SSL) is a real-time application, starting with capturing audio data and ending in location estimations. Reducing sound source localizer's power and area consumption is in high demand. Saving resources in computation systems is an active field of research and crucial for modern battery-powered devices. Mobile phones, laptops, and personal assistants require the speaker location to enhance their functionality. Stochastic computing (SC) [3], [4] is a promising computing paradigm that can provide a lower power consumption, a lower hardware area footprint, and a higher tolerance to soft errors compared to conventional binary designs. SC promises high power efficiency that can contribute to longer battery lives. Developing a sound source localizer using SC could save considerable hardware costs. Most prior works in the SC literature have been limited to scientific research and practical implementations for end-users are rare. This work develops the first functional SC-based SSL system by implementing a low-cost mixed-signal analog-digital design.

In SC, numbers are encoded into stochastic bit-streams of 0s and 1s. SC is possible in both time-continuous and time-discrete domains [5]. However, some operations such as delaying and storing stochastic sequences are simpler in the time-discrete domain. The concept of coding a value into a stochastic number (SN) (aka bit-stream) is fundamentally different from the conventional binary-radix encoding. Stochastic sequences have voltage level 1 (high) with probability ρ and level 0 (low) with probability $1-\rho$. In other words, a bit within a stochastic bit-stream is 1 with probability $\mathrm{P}(S_i = 1) = \rho$ and is 0 with probability $1-\rho$. For example, a stochastic sequence S with 20% 1s and 80% 0s is an SN with probability value

$\rho = 0.2$. Unsigned and signed numbers are supported in SC with unipolar and bipolar SN formats, respectively [3]. In the unipolar format, each bit of the bit-stream is 1 with probability ρ, and in bipolar format is 1 with probability $(\rho+1)/2$. We use the notation S_i for the ith bit in bit-stream S with length N $(0 \leq i \leq N-1)$. SC can perform arithmetic operations with simple standard logic gates. For example, the multiplication operation can be performed using a single standard AND gate, which provides a significantly lower hardware cost than the conventional binary multiplication.

This work targets the computationally intensive parts of SSL to replace their costly conventional binary computations with low-cost SC alternatives. The emphasis is on applying SC to a new application, namely SSL, designing a novel SC-based functional unit, and analyzing the limitations and advantages of SC for SSL. The work focuses on time-discrete SC, although a time-continuous variant of the sound source localizer is also feasible. To the best of our knowledge, this is the first work that applies SC to SSL. Our design takes advantage of the recently introduced deterministic approaches of SC [5], [20], [21] which provides lower latency and higher accuracy compared to the conventional random SC. We share the open source project files of the proposed designs in [22]. In summary, the main contributions of this paper are as follows:

- Developing the first SC-based design for SSL, a promising alternative to its costly conventional implementation.
- Near-sensor processing of sound signals using time-encoded pulse-width-modulated (PWM) data; Our signal processing system avoids using conventional analog-to-digital converters (ADCs).
- Designing an accurate SC-based multiply-and-accumulate (MAC) unit, which is used in the SSL design based on time-delay estimation (TDE).
- Practical hardware implementation of a new, efficient sound source localizer with SC in both digital and analog domains.
- Developing an efficient cross correlation (CC) design based on low-discrepancy (LD) Sobol bit-streams.

This work offers electronic design automation beyond application by directly processing sensor data in the form of PWM signals. This eliminates the need for conventional ADCs. We explore seven design approaches on the digital side to evaluate different design possibilities. We employ the state-of-the-art LD Sobol sequences in the hardware design, highlighting their advantages in a practical application.

II. Sound Source Localization (SSL)

SSL is a challenging task with different applications, from robotics to personal assistant devices. SSL identifies the spatial locations of the sound sources using aural information [23]. There have been some prior studies for SSL using conventional machine learning techniques [24], [25], [26], [27], [28]. However, the efficient hardware design of the whole SSL system is relatively complex as it requires an integration of analog and digital components. Field programmable gate array (FPGA) and processor co-operated solutions have been studied before [7], [14], [15], [19]. Liaquat et al. classify the prior works based on the keywords such as *direction of arrival*, *localization dependencies*, and *improving sound localization* [29]. According to a recent review study [29], several proposed methods for sound location determination are given as energy-based methods, beamforming, time difference of arrival, time of arrival, steered response power, direction of arrival, and inter-microphone intensity difference. Table I summarizes some important prior studies on SSL hardware design and implementation. To the best of our knowledge, this work is the first to employ SC for cost-efficient design of a complete SSL system with both analog and digital components. SSL starts with an array of multiple microphones capturing sound waves. The microphone array is a set of closely positioned microphones packed in a single device. A single microphone cannot deal with 3D-processes, such as reverberation and ambient noise, whereas multiple microphones can record sound sources spatially selective through beamforming [30], [31]. The sound processing system's accuracy, precision, and capabilities improve with the computational progress of signal processing and the number of microphones in the array. In what follows, we introduce technical terms related to sound source localizers.

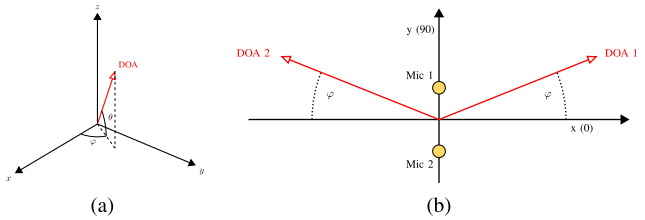


Fig. 1. (a) azimuth and elevation (φ,θ) that define the DOA vector. The DOA points from the center of the microphone array to the sound source. (b) the coordinate system and the ambiguity in locating sound sources with two microphones. DOA, with equal phase delay, form a *cone of uncertainty*.

A. Coordinate System and Direction of Arrival

The spherical coordinate system describes points and vectors in the 3D space. Microphones and sound sources are defined relative to the center of a microphone array with elevation θ, azimuth φ and radius r. We use the convention that elevation is zero in the x, y plane and increases in the z-axis. The elevation angle is often replaced by the polar angle measured from the z-axis so that the polar angle is $90 - \theta$. Equation (1) shows the conversion between the Spherical and Cartesian coordinate systems. *Small microphone arrays* estimate direction-of-arrival (DOA), a vector from the center of a microphone array to a sound source, instead of the absolute source locations. Equation (2) and Fig. 1(a) show the definition and an illustration of a DOA vector.

$$\vec{p} = \begin{bmatrix} r \cdot \cos\theta \cdot \cos\varphi \\ r \cdot \cos\theta \cdot \sin\varphi \\ r \cdot \sin\theta \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{1}$$

$$\vec{\varsigma} = \begin{bmatrix} \cos\theta \cdot \cos\varphi \\ \cos\theta \cdot \sin\varphi \\ \sin\theta \end{bmatrix} \tag{2}$$

TABLE I
PRIOR ARTS ON THE HARDWARE DESIGN AND IMPLEMENTATION OF AN SSL SYSTEM

Reference	Year	Implementation Details
[6]	2003	Field-programmable gate array (FPGA)-based design for real-time SSL implementation cross-correlation technique based on a modified version of the phase transform
[7]	2009	Far-field SSL design using FPGA
[8]	2010	Single-channel, FPGA-based sound source detection
[9]	2011	Direction-of-arrival (DOA) estimation using an FPGA-based micro-electro-mechanical systems (MEMS) platform
[10]	2013	MEMS microphone-based MEMS source signal processing in the far-field using digital signal processing (DSP) and microcontroller platform
[11]	2013	Sound-card-supported computer running direction of arrival method via Fourier analysis of sound intensities
[12]	2014	Sound compass design using FPGA and digital MEMS microphones.
[13]	2015	Voltage-controlled module design for spiking neural network-based SSL implementation
[14]	2017	System-on-Chip design for SSL using hardware-software co-design
[15]	2018	Delay-and-Sum beamforming algorithm implementation on FPGA for sound source localization
[16]	2020	System-on-Chip microcontroller-based (ESP32) design with electret microphones
[17]	2020	ARM microcontroller and circular MEMS microphone array architecture for fuzzy fusion and beamforming method
[18]	2022	Single board computer-based time difference of arrival implementation
[19]	2022	Audio DSP and microprocessor-based design for time difference of arrival method architecture
This work	2023	Analog & digital design for SSL using SC for the first time

B. Near- and Far-Field

Sound sources generate spherical sound waves that propagate towards observers. An observer close to the source (near-field) can measure the curvature of arriving sound waves. The curvature decreases for an observer far away (far-field), and the sound waves appear planar [32]. Microphone arrays in near-field conditions can estimate the curvature of arriving sound waves by calculating the frequency-dependent phase delay at each microphone. The curvature of sound waves is proportional to the distance between the source and the observer. Generally, SSL with small microphone arrays in far-field conditions is limited to 1D or 2D DOA estimations without estimating the distance. However, DOA estimations of multiple small microphone arrays can be used to compute the absolute position of a sound source with triangulation. Analyzing sound-pressure levels (SPLs) at each microphone is an alternative or complementary method to estimate distances. The SPL decreases with the distance to sources. This method works in near-field and far-field, but the accuracy decreases for sources in far-field. So, we assume far-field conditions and do not attempt to estimate the distance to sources.

C. Ambiguity of Sound Source Localization

A microphone array with two microphones can estimate angles in the $[-\pi/2, \pi/2]$ interval. For simplicity, we use the convention that microphones and the source are in the xy-plane, and the microphones are on the y-axis. Using this spatial distribution, the one-dimensional (1D) DOA vector ς_{1D} forms a right angle with the z-axis:

$$\vec{\varsigma}_{1D} = \begin{bmatrix} \cos\varphi \\ \sin\varphi \\ 0 \end{bmatrix} \tag{3}$$

Fig. 1(b) shows a microphone array with two microphones. Sound waves reach `Mic 1` before they reach `Mic 2`. The signal delay is equal for DOA1 and DOA2. Any DOA with the same azimuth φ (but any elevation θ) will result in the same TDE. We can visualize the so-called cone of uncertainty by rotating the DOA vector in Fig. 1(b) around the y-axis.

Spatial aliasing also causes ambiguity in SSL results. A microphone array can measure the same signal delay for multiple different DOA, even if the θ angle is equal and we only consider sound sources in one half-plane. The concept of spatial aliasing is shown in Fig. 2. The frequency of arriving sound waves is high enough to allow spatial aliasing. The microphones measure the same delay for $\varphi = 0$ and $\varphi = \varphi_1 \neq 0$. We observe spatial aliasing for sources that emit a sine wave with frequency $f_1 = \frac{v}{\lambda_1}$, if Equation (4) is fulfilled [2]:

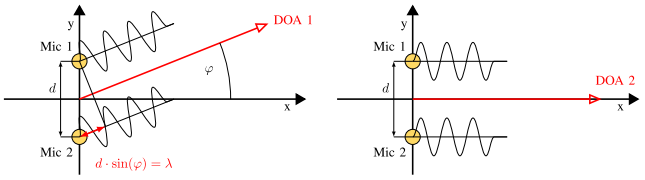


Fig. 2. Spatial aliasing for high frequencies. DOA 1 and DOA 2 result in the same phase delay.

$$\pi \frac{d}{\lambda_1} \sin(\varphi_1) = \pi \frac{d}{\lambda_1} \sin(\varphi_2) + n2\pi, \quad \varphi_1 \neq \varphi_2 \tag{4}$$

and the velocity of sound waves is $v = 343\frac{m}{s}$. Equation (4) is solvable when $d \leq \frac{\lambda_1}{2}$, as two different azimuth angles yield the same phase shift. We avoid spatial aliasing if the distance between the microphones is smaller than half of the maximum wavelength. SSL can tolerate some spatial aliasing if the most dominant frequencies are below that threshold. Digital or analog lowpass filters are used before SSL to dampen high-frequency components and reduce spatial aliasing. For instance, Amazon Echo Dot device has six microphones arranged in a circle around one centered microphone. The distance between two microphones is $d = 77.8\ mm$. Frequencies above $f = 2.2$ kHz will cause partial aliasing, where:

$$2d < \lambda = \frac{v}{f} \tag{5}$$

$$f < \frac{v}{2d} \tag{6}$$

$$f < \frac{343}{0.1556} = 2.2044. \tag{7}$$

SSL for speech is still possible because the most dominant frequencies of humans are below 2 kHz.

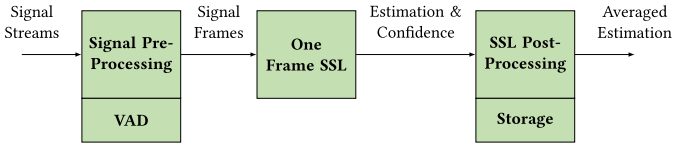


Fig. 3. Block diagram of a sound source localizer.

D. Components of Sound Source Localizers

SSL algorithms exist in both the frequency and time domains. Search-based SSL algorithms in the frequency domain are superior in locating multiple sources simultaneously but are more computationally intensive [32], [33]. This work uses a time-domain algorithm without Fourier transforms and assumes only one active source. An SSL system consists of three subsystems. The pre-processing subsystem usually segments, filters, and weights the audio stream. It splits the stream into short data blocks as the SSL algorithm calculates source locations based on short audio frames. Following that, an SSL algorithm processes a single frame when the voice activity detection (VAD) detects a valid signal. By averaging multiple one-frame-estimations or using knowledge about the room geometry and array position, an SSL post-processor can further enhance the accuracy of estimations [2]. Fig. 3 summarizes the overall flow of a sound source localizer.

The first subsystem of a sound source localizer prepares raw microphone signals for the SSL algorithm. The pre-processing usually includes low-pass or band-pass filtering, framing and VAD. The theory of SSL can be divided into algorithms that locate sources in either one or two processing steps. The one-step methods are based on the steered response power. Algorithms based on the steered response power require a Fourier transform in the pre-processing subsystem because they compute the source location in the frequency domain. This work focuses on the two-step methods that rely on TDEs for each microphone pair [2], [33]. The post-processor is the final part of a sound source localizer that receives the one-frame SSL estimations and optional confidence levels. The post-processor uses application-specific information and previous one-frame location estimations to enhance the accuracy of the sound source localizer. Application-specific information can, for example, include the location of the microphone array. If the microphone array is mounted on a wall, it can filter out false estimations that come from behind. The SSL post-processor has access to previous location estimations and can average over multiple frame estimations.

E. SSL Based on Time Delay Estimation (TDE)

Algorithms based on TDE use the propagation speed of sound waves to localize sources [34]. A sound wave reaches each microphone with a relative delay. The time delay is calculated pairwise for each microphone pair. Note that an M-microphone array has $N_{pairs} = (M(M-1))/2$ unique pairs. Selecting two microphones m_1, m_2 (that lie on the y-axis), a sound wave in the far-field reaches microphone m_2 with a delay that depends on the azimuth angle φ of the DOA:

$$\tau_d = \frac{d}{v} \sin \varphi, \tag{8}$$

where d is the distance between the microphones and v is the velocity of sound waves.

An intuitive computing approach for TDEs is the cross-correlation (CC) function, which indicates the similarity of signals. The function has a sharp peak at a position proportional to the time delay. For discrete inputs, CC is defined as:

$$c[n] = (\mathbf{x_1} \star \mathbf{x_2})[n] \sum_{i=-\infty}^{\infty} x_1^*[i] x_2[i+n] \tag{9}$$

where x_1^* is the complex conjugate of signal x_1. Since microphone signals have no imaginary component, we can set $x_1^* = x_1$. When $c[i_{max}]$ is the maximum value of the CC function and i_{max} is different from zero, then the audio waves reach `Mic 1` before or after `Mic 2`. When the time delay between the microphone signals increases, $|i_{max}|$ increases. The maximum time delay is reached when both microphones and the source are lined up. The TDE (τ_d) is used in Equation (8) to calculate the azimuth (φ) of the DOA with a sample rate of frequency (f_s):

$$\tau_d = \frac{i_{max}}{f_s} \tag{10}$$

$$\varphi = \arcsin \frac{\tau_d \cdot v}{d} = \arcsin \frac{i_{max} \cdot v}{f_s \cdot d} \tag{11}$$

Equation (11) results from a simple one-frame SSL algorithm that uses two microphones. Spatial aliasing causes a broader peak in the CC function. The CC function always has one clear peak if dominant frequencies are below the spatial aliasing threshold ($d \leq \frac{\lambda}{2}$) and only damped frequencies cause spatial aliasing. A larger microphone array can calculate source locations in a closed-form [35] or search-based. For search-based algorithms, the final location estimation is the position with the minimum root-mean-square error of the calculated and measured TDE. For a detailed analysis, the readers are referred to [2], [32], and [35]. The resolution of CC is determined by the distance between the microphones (d) and the sampling frequency (f_s). For example, a distance $d = 0.066$ m between two microphones and a sample rate of $f_s = 15.6$ kHz results in $2 \times N_{MaxLag} + 1 = 7$ different possible time delays:

$$\tau_{d,max} = \frac{d}{v} = 192 \mu sec \tag{12}$$

$$N_{MaxLag} = \tau_{d,max} * f_s = 3 \tag{13}$$

with $\tau_{d,max}$ being the maximal time delay. A maxima of the CC at $c[i_{max} = N_{MaxLag}]$ corresponds to a time delay of $\tau_d = i_{max}/f_s = 192$ μsec, whereas a maxima at $c[i_{max} = N_{MaxLag} - 1]$ means a time delay of 128 μsec. The resolution is limited to 64 μsec. Interpolating the CC function increases the resolution with additional computational effort. We can interpolate the *maximum* between two samples instead of using the closest sampled value $c[i_{max}]$.

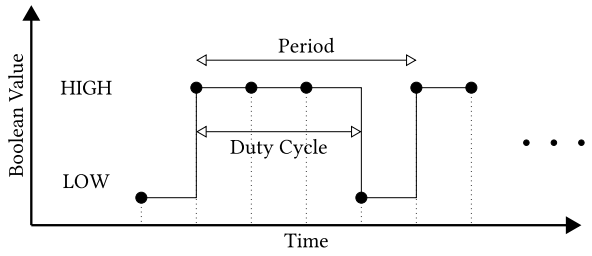


Fig. 4. Encoding in time-domain with PWM signals. Repeated unary bit-streams are equal to sampled PWM signals.

This section underscored that we need at least two microphones to locate sound sources. For two microphones, the SSL is synonymous with one-dimensional DOA estimation. The one-frame SSL processing block is the core of the sound source localizer and computes the CC function of both microphone signals. Next, the SSL algorithm searches the peak of the CC result. The peak index (i_{max}) is then forwarded to the SSL post-processing block.

III. Deterministic Stochastic Computing (SC) and an Overview of the SC-Based SSL Design

In this section, we first present the fundamentals of deterministic SC, and then show an overview of the proposed SC-based SSL implementation.

Truly random or pseudo-random numbers have been used for generating stochastic bit-streams since the introduction of SC [3], [36]. The target probability is compared with N random numbers in N clock cycles to generate a bit-stream of N bits. A 1 is produced in each comparison if the random number is less than the target value. Bit-streams generated with this method suffer from random fluctuations error. Deterministic methods of SC have been recently introduced to remove this source of error and generate bit-streams *accurately*. SC using these methods can produce completely accurate results [21], [37]. Besides quantization errors, processing bit-streams with *specific lengths* is the only constraint to producing completely accurate results with these deterministic methods. These methods operate on the same SC constructs but use special stochastic number generators (SNGs) to generate deterministic bit-streams with the desired distribution. LD and unary bit-streams are two common variants of deterministic bit-streams [5], [20], [21]. The so-called *unary bit-streams* have the 1s grouped at the beginning or end of the bit-stream. Unary bit-streams are generated by replacing the random number generator (RNG) with an up- (or down-) counter [38]. Usually, we generate more than one period of 1s followed by 0s, letting the counter overflow and repeat. The time-continuous equivalent of a periodic unary bit-stream is a PWM signal. As shown in Fig. 4, a PWM signal is defined by a duty cycle (D) and a period or frequency ($f = \frac{1}{period}$). The duty cycle is the fraction of time in which the signal is high and is equivalent to the probability of observing 1s in the time-discrete domain [39].

Correlation between inputs can change the functionality of logic circuits in SC. An AND gate works as a multiplier if it is connected with *uncorrelated* or *independent* inputs [40], [41]. However, when connected with *correlated* inputs, it performs a minimum value operation [39]. When *independent* inputs are needed, unary bit-streams (or PWM signals) with relatively prime lengths (periods) are generated [5]. In the example below, bit-stream A is built from four times repeating unary bit-stream 100 (period=3) and bit-stream B from three times repeating unary bit-stream 1110 (period=4). The periods of the unary bit-streams are relatively prime ($GCD(3,4) = 1$). The unary bit-streams are repeated to generate bit-streams A and B with a length equal to the least common multiple (LCM) of both periods ($LCM(4,3) = 12$). Bit-streams are bit-wise ANDed to perform multiplication operation. Selecting relatively prime periods satisfies the independence requirement of the multiplication operation, and producing bit-streams with the LCM of the periods guarantees the accuracy of the operation. We can see that an accurate output (here, 3/12) is produced from bit-wise ANDing input bit-streams.

$$\begin{aligned} A &= \frac{1}{3} = 100\,100\,100\,100 \\ B &= \frac{3}{4} = 1110\,1110\,1110 \\ A \times B &= \frac{1}{3} \times \frac{3}{4} = \frac{3}{12} = 1000\,0010\,0100 \end{aligned} \tag{14}$$

Now, we present an overview of the proposed sound source localizer based on the SSL architecture flow in Fig. 3. The design uses two microphones, analog signal processing, digital TDE with CC, an average filter for SSL post-processing, and light-emitting diodes (LEDs) for outputs. Fig. 5 shows the block diagram of the proposed sound source localizer. The implemented design initiates the signal processing by a custom analog processing that generates PWM signals from the two input microphones. Those signals are sampled by an FPGA and thus converted to periodic unary bit-streams. This is a unique step in our design, which in contrast to prior art in hardware-based SSL, encodes data to a uniform data representation. This allows the designs to use simple logic operators such as AND and OR gate for basic arithmetics such as multiplication and addition, respectively, based on SC. As a secondary alternative for comparison purposes, we use a serial peripheral interface (SPI)-controlled ADC for reference. We implement different designs of the CC function using periodic unary and Sobol-based bit-streams. Following the CC block, the subsequent computation block searches for the maximum $c[i_{max}]$ in the centered K values of the CC, as $[i_{max}]$ is proportional to the time delay. The last processing block computes the average of multiple TDEs. The output of the digital design is an averaged TDE. We linearly map the TDE to 15 individually addressable LEDs on a semicircle that point towards the source location.

TDE is a part of any two-step SSL algorithm. In the literature, TDE is also called the time difference of arrival estimation, time of arrival estimation, or time of flight estimation. TDE in the time domain is a maximum search over the CC result of two signals. The CC function can be broken down into a set of MAC operations. Using SC for implementing these operations can lead to significant savings in the hardware resources, particularly in implementing the multiplication

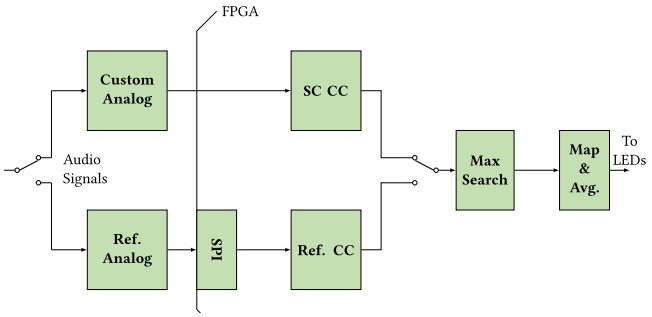


Fig. 5. The implementation overview. Block diagram of the signal processing chain.

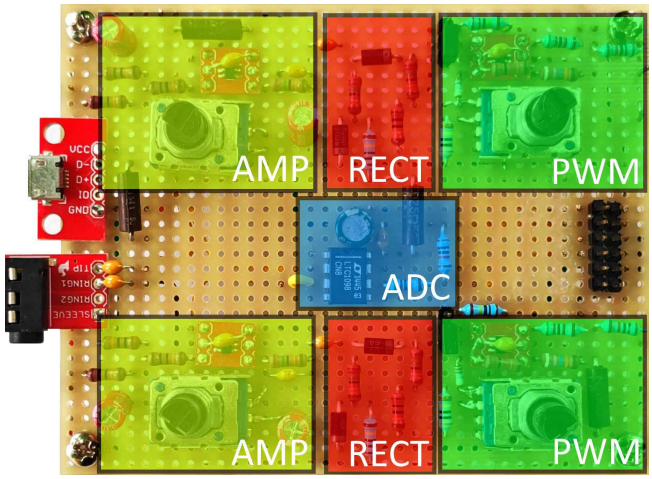


Fig. 6. *Analog board* designed for two audio signals. The amplifier (AMP), half-wave rectifier (RECT), PWM generation, and ADC circuits are marked yellow, red, green and blue, respectively.

operations [42]. In our SC design, we use *periodic unary bit-streams* for the following reasons:

1) Generating random bit-streams is costly. Often the bit-stream generation modules consume more resources than the stochastic computation circuit [4]. A unary bit-stream generator reduces the bit-stream generation cost.
2) Using analog to time converters (PWM signal generators) instead of analog voltage to digital binary converters followed by digital bit-stream generators could save further hardware resources. This is, in particular, appealing for near-sensor processing [43].

IV. Custom Analog Processing Set-up

In what follows, we describe the analog circuitry we built to handle the inputs and outputs of the sound source localizer. We use discrete components such as capacitors, resistors, and operational amplifiers (OPAMPs) for analog processing and a Digilent Zedboard with an FPGA for digital computations.

A. Analog Board

The *analog board* is shown in Fig. 6. The board requires a 5 V direct current (DC) power supply. It takes two audio signals as inputs and has two outputs. The first is a conventional SPI, and the second is a unidirectional interface with two channels that carry the PWM signals for the SC circuit. In the board shown in Fig. 6, the signals flow from left to right with the input audio jack on the left side and the Peripheremodule (PMOD) interface to the FPGA on the right side. The PMOD interface combines both outputs into a single cable. The signal processing chain for the left and the right audio channels is on the top and bottom of the *analog board*. (For the details of the designed analog board and its components, please see Appendix.)

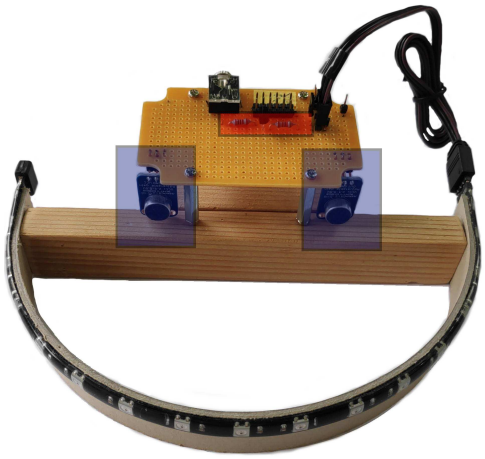


Fig. 7. *Input/output board* with two microphones (marked blue) as input sensors and 15 LEDs as outputs. The level shift circuit for interfacing with the LED strip is marked with red.

B. Input/Output Board

Fig. 7 shows the *input/output* board. It connects to the *analog board* and the Zedboard FPGA through the PMOD interface. The *input/output* board fulfills two tasks: First, it provides audio signals to the *analog board* via two electret microphones. The microphones are mounted to the *input/output* board at a 90° angle as shown and marked blue in Fig. 7. The outputs are single-ended microphone signals and forwarded to the *analog board* through an input jack located at the top of the board shown in Fig. 7. Second, the *input/output* board visualizes the SSL output through 15 individually controllable LEDs. Fig. 7 shows a LED strip mounted on a semicircle at the bottom. A speaker in front of the *input/output* board sees a lighted LED pointing in his direction because the spatial distribution of the LEDs matches the coordinate system of the microphone array. The origin of the coordinate system is between the microphones on the ±90° axis. The LED strip expects a 5 V control signal with a threshold greater than the output of the FPGA (HIGH≥3.4 V). Thus, the *input/output* board has a level shift circuit (marked with red in Fig. 7.) The unidirectional level-shift circuit consists of a single bipolar junction transistor and two resistors at the base and the collector. This circuit only works for high-impedance loads at the output pin and drivers that can sink enough current at the input pin.

V. Proposed Digital Implementation

This section explains the proposed digital processing blocks of the sound source localizer (right side of the vertical line,

labeled "FPGA", in Fig. 5). We first design several SC circuits for the MAC operation. We consider OR-based and counter-based accumulators after performing SC multiplication via bit-wise AND operation. We then design the CC function block. We utilize periodic unary and also state-of-the-art Sobol bit-streams in the CC design. We need to divide the continuous microphone signals into frames before computing the CC function. The window length is a trade-off between responsiveness and accuracy. Setting $N_{frame} = 128$ means that one computation of the CC function requires $\Delta t = \frac{1}{f_s} \times N_{frame} = 8.2$ ms. Subsequently, we assume that the position of a sound source is approximately constant within 8.2 ms, which is essential for accurate estimations. We can write the CC function for signals of length N_{frame} as:

$$c'[n] = \sum_{i=0}^{N_{frame}-1} x_1[i]x_2[(i+n)_{\bmod N_{frame}}]. \tag{15}$$

Furthermore, only the CC values close to $c'[0]$ are relevant for TDE. The distance between the microphones of the *input/output* board is $d = 0.066\ m$, and the sampling rate is $f_s = 15.6\ kHz$. Therefore, only $N_{MaxLag} = 3$ (see Equation (13)) values to the right and left of $c'[0]$ are of interest for TDE. We can efficiently calculate the seven ($K = 2 \times N_{MaxLag} + 1 = 7$) centered values of the CC function as a set of MAC operations. We denote $x[n]$ as the current audio sample and $x[n-1]$, $x[n-2]$, $x[n-3]$ as the three former samples. We calculate the K-centered values of the CC function with K MAC units as:

$$c'[i] = \begin{cases} c'[-3] \leftarrow c'[-3] + x_2[n]x_1[n-3] \\ c'[-2] \leftarrow c'[-2] + x_2[n-1]x_1[n-3] \\ c'[-1] \leftarrow c'[-1] + x_2[n-2]x_1[n-3] \\ c'[0] \quad \leftarrow c'[0] + x_2[j]x_1[j] \\ c'[1] \quad \leftarrow c'[1] + x_2[n-3]x_1[n-2] \\ c'[2] \quad \leftarrow c'[2] + x_2[n-3]x_1[n-1] \\ c'[3] \quad \leftarrow c'[3] + x_2[n-3]x_1[n] \\ 0 \qquad \text{if } i < -3 \text{ or } i > 3 \end{cases} \tag{16}$$

with $j \in \{n-3, n-2, n-1, n\}$. We can use any pair of audio samples $(x_1[j]x_2[j])$ to compute $c'[0]$. The accumulator is set to zero after N_{frame} samples just before the next frame begins. The two architectures corresponding to Equation (16) and Equation (15) for sequential and parallel MAC operation compute the centered value $c'[0]$ of the CC because there is no relative shift between microphone signals $\mathbf{x_1}$ and $\mathbf{x_2}$.

A. Stochastic Circuit for MAC Operation

In what follows, we present three SC designs for MAC operation. The first design is based on the MAC technique proposed by Schober et al. [42]. This design uses AND gates for multiplication and OR gates for addition. The AND gates compute accurate multiplication by processing bit-streams with relatively prime lengths of n and k ($k = n-1$).[1] We use $n = 16$ and $k = 15$ unless stated otherwise, which is comparable to conventional computations with 4-bit precision ($B = \log_2(n) = \log_2(16) = 4$). The input bit-streams and the results of AND gate have $N = LCM(nk) = 240$ bits, so the precision of the products is approximately $2B = 8$. [2] The OR gates require *negatively* correlated bit-streams to function as saturating adders. By introducing a relative delay between the products we shift the 1s of the summand to a section of the accumulator bit-stream with 0s only. This guarantees a negative correlation between input bit-streams and perform saturate addition.

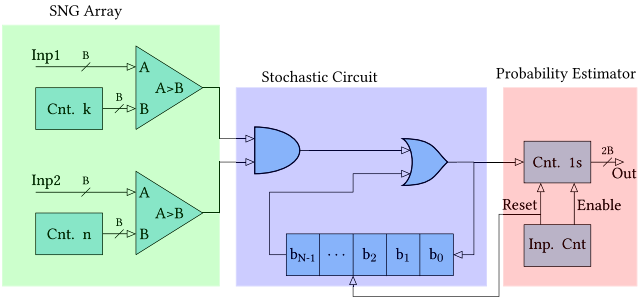


Fig. 8. OR-based SC Design for MAC operations.

Combining the stochastic MAC circuit with an SNG array and a probability estimator gives the architecture shown in Fig. 8. The two counters (marked with green and labeled with Cnt. n and Cnt. k) repeatedly count to n and $k = n-1$ to generate bit-streams with relatively prime periods. The bit-streams are multiplied using an AND gate. The AND gate is the first component of the stochastic circuit and forwards the product $\mathbf{x_1x_2}$ to the OR gate. The OR gate computes the bitwise disjunction of the most significant bit (MSB) of the shift register (accumulator) and the current bit of the product. The result is then stored back to the shift register's least significant bit (LSB). The OR gate's result has an error if both inputs are 1s, which we refer to as an *error due to overlap*. The authors in [42] show how to calculate the number of bits for the shift register to guarantee no overlap for small input values. But audio signals can have high amplitudes. We know that at least half of the input values are zero due to the half-wave rectifier. Hence, we use an approximate variant of the technique with a shift register of length:

$$N_{\text{SR}} = nk + \text{ceil}(\tfrac{nk}{N_{frame}-1}) \cdot (N_{frame} - 1) = 494. \tag{17}$$

The first summand, in the beginning, will be in the [0, 239] interval of the shift register. After the second summand, the first one is shifted to [240, 479], and the second summand is in the [0, 239] interval. After that, the OR gate computes the logical disjunction of the first and the third summand, with a relative shift between them. The final block in Fig. 8 is the *probability estimator* in the form of a simple up-counting counter. The *probability estimator* is enabled by a separate counter (*Inp. Cnt*) that sets the *Enable* signal N_{SR} clock cycles before the reset and occurs every $nk \times N_{frame}$ cycles. After each frame, the *Reset* signal resets the *probability estimator* and the accumulator bit-stream. For the rest of the paper, we refer to the MAC design in Fig. 8 as the *OR-based design*.

The second MAC design is shown in Fig. 9. It performs the multiplication operations in the stochastic domain but

[1] Note that $n = k - 1$ is relatively prime for $n \geq 3$.

[2] Bit-streams with 256 bits would exactly have 8-bit precision.

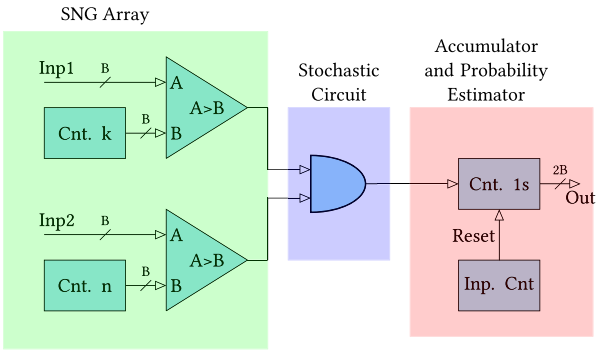


Fig. 9. Counter-based SC Design for MAC operations.

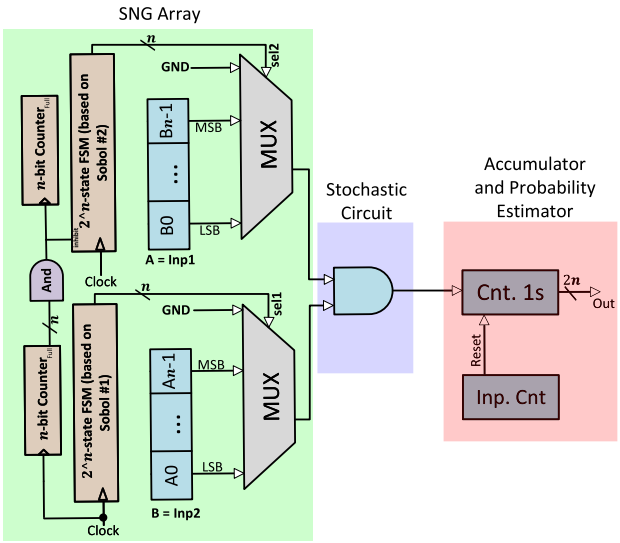


Fig. 10. Counter-based SC MAC Design with Sobol bit-streams.

the accumulation part in the binary domain. The probability estimator directly counts all 1s from the AND gate instead of first accumulating with OR. The results are accurate because both the multiplication and the accumulation are accurate. We will refer to this MAC design as the *counter-based design*.

We further implement a *counter-based* MAC design based on LD Sobol-based bit-streams. The proposed design is shown in Fig. 10. This design performs the multiplication operations in the stochastic domain by bit-wise ANDing Sobol bit-streams and the accumulation part in the conventional binary domain. In SC systems, the bit-stream generation units often consume more resources than the computation circuits. Conventionally, the SNGs for bit-stream generation use a comparator and a random number generator. This results in a significant area and power overhead, particularly when generating LD Sobol-based bit-streams. For low area and power consumption, the proposed Sobol-based design of this work uses a MUX and finite state machine (FSM)-based Sobol bit-stream generator. Compared to the architecture in Fig. 9, the comparator-based unary bit-stream generator is replaced with an FSM-based Sobol-based bit-stream generator. The two input bit-streams are generated based on two different Sobol sequences, ensuring accurate multiplication [44]. Binary inputs A and B are fed as the primary inputs of the MUX. Two FSMs are designed

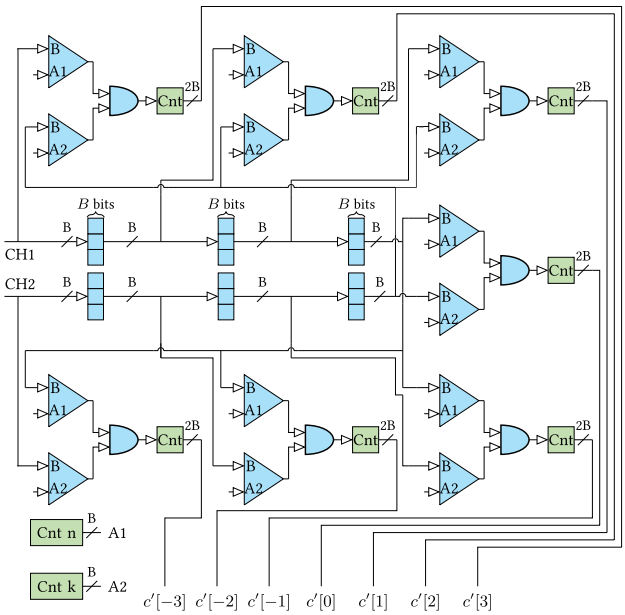


Fig. 11. Block diagram of the CC with digital bit-stream generation.

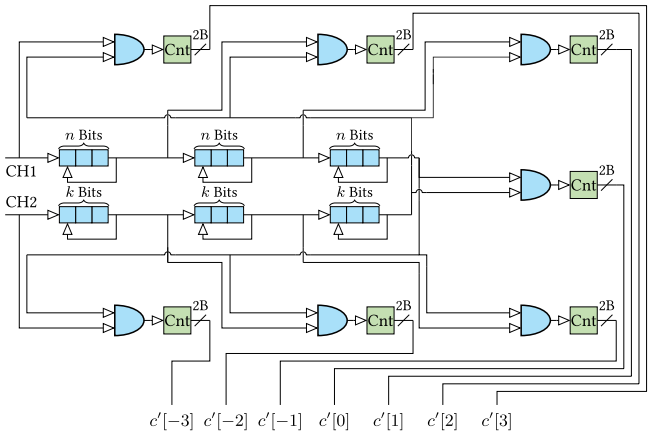


Fig. 12. Block diagram of the CC with PWM signals as inputs.

based on Sobol sequences #1 and #2 from MATLAB built-in Sobol sequence generator [45] as proposed in [44]. The FSMs are connected to the select inputs of the MUXs. We will refer to this MAC design as the *Sobol-based design*.

B. Stochastic Circuit for Cross-Correlation

Fig. 11 and Fig. 12 show the block diagram of the CC with the counter-based MAC for digital bit-streams and analog PWM signals, respectively. Channel One and Two (CH1 and CH2) in Fig. 11 are weighted binary numbers and in Fig. 12 are PWM signals. We draw the registers in a simplified manner as three horizontal squares when storing bit-streams and three vertical squares when storing weighted binary numbers. The output ($c[i]$) is the CC result in weighted binary format.

Next, we provide a simple and more complex method to store the PWM signals from LTC6992. For the simple approach, we continuously sample with f_{clk} and get $n = 16 (k = 15)$ bits per period of the PWM signal. During one audio sample ($\frac{1}{f_s}$), the PWM signal has $k(n)$ periods

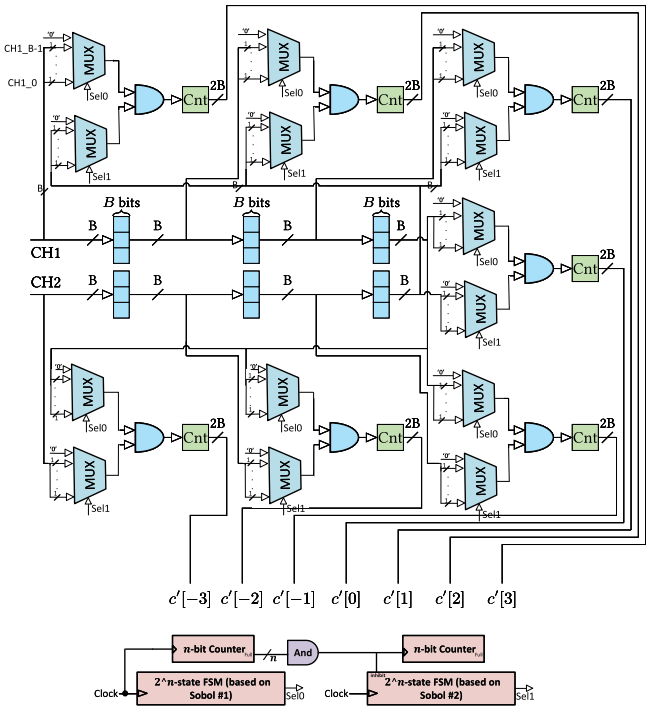


Fig. 13. Block diagram of the CC with digital Sobol-based bit-stream generation.

(see Equation (24)), which results in $nk = 240$ bits of data per microphone for each audio sample. The second approach takes advantage of the redundancy within the periodic PWM signals and stores only the first period. Following that, we perform roll operations until the first period of the next audio sample. In Fig. 12, we use arrows that point backward from the end to the start of the shift registers to indicate that we either do shift-through operations or roll operations. Both approaches provide the same bit-streams to the stochastic circuit if we assume the duty cycle of the PWM signals is constant within one audio sample. Fig. 13 depicts how we include Sobol-based bit-stream generators before AND gates for CC operation.

C. SPI, Maximum Search, Map and Average

The SPI block shown in Fig. 5 is a minimal SPI-master that polls data from the ADC. LTC1098 transmits one channel per query with 14 bits. The first six bits configure the ADC, and the remaining eight bits are either the left or right channel. To simplify the implementation, we increase the number of bits per query to $N_{query} = 16$. The ADC operates at its maximum clock frequency of 500 kHz ($f_{sclk} = f_s \times 2 \times N_{query} = 499.2$ kHz). The post-processing tasks are the maximum search, mapping and average, that we implement with conventional binary arithmetic. The maximum search finds the maxima of the MAC results $c[i_{max}]$ and forwards its index (i_{max}) to the Map & Average block if the maximum is above a threshold.

The Map & Average block maps $i_{max} \in \{-3, -2, \ldots, 2, 3\}$ to $i'max \in \{0, 2.5, 5.0, \ldots, 12.5, 15.0\}$ to create 15 equally sized intervals for the 15 LEDs of the *Input/Output board*. Averaging over multiple i'_{max} with an average filter results in

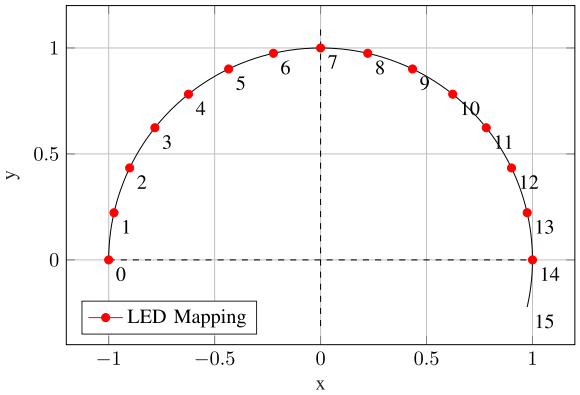


Fig. 14. The digital output of the FPGA in the $z_{avg} \in [0, 15]$ interval mapped to a LED strip with 15 LEDs mounted to the *Input/Output board* (Fig. 7).

z_{avg}. z_{avg} is mapped to the LED strip of the *analog board*, as shown in Fig. 14. The LEDs are marked red, and z_{avg} is labeled with black. The number of the LED that is turned on is the truncated filter output.

$$\text{LED}_{out} = \left\lfloor z_{avg} \right\rfloor \in \{0, 1, 2 \ldots, 13, 14\} \tag{18}$$

As z_{avg} is proportional to the TDE, we can use its value to turn on the LEDs of the *Input/Output board*. Mathematically, by mapping the filter output to a semicircle, we transform z_{avg} to an angle in the $[-90, 90]$ interval. When z_{avg} is close to zero, the first LED turns on. When z_{avg} is close to its maximum, the last LED turns on. The two examples indicate sound source locations at $\varphi = \pm 90$ with the LED positions shown in Fig. 14. We can generalize and calculate φ as a function of z_{avg} as follows:

$$\varphi = \sin^{-1}(\frac{v}{d}\tau_d) \cdot \frac{180^\circ}{\pi} \tag{19}$$

$$= \sin^{-1}(\frac{z_{avg}}{7.5} - 1) \cdot \frac{180^\circ}{\pi} \tag{20}$$

The average filter has an infinite impulse response (IIR) with the linear difference equation:

$$z_{avg}[n] = \alpha \cdot i'max[n] + (1 - \alpha) \cdot z_{avg}[n - 1]. \tag{21}$$

We choose $\alpha = 2^{-4}$ and approximate a moving average filter of length $N_{average} = 20$ with an average interval of 164 ms ($N_{average} \times 1/f_s \times N_{frame}$). Choosing the α coefficient is a trade-off between responsiveness and the accuracy of location estimations for stationary sound sources. Choosing an even smaller value for α (a larger averaging interval) hardly increases the accuracy of estimations for stationary sound sources in simulations. However, it drastically reduces the accuracy of estimations for moving sound sources.

D. Weighted Binary Implementation

We also implement the CC function with weighted binary arithmetic and $B = 3, 4, 5, 8$-bit precision signed inputs for reference. The weighted binary CC design has seven MAC units. The multiplier is exact, with an output bit-width twice as large as the input bit-width. We note that the precision of the products is comparable to that of the output bit-stream from

TABLE II
SUMMARY OF THE RELATIONS BETWEEN THE DIGITAL RESULTS AND ESTIMATED SOURCE LOCATIONS

i_{max}	i'_{max}	τ_d	φ
-3	0	$-192\ \mu s$	$-90°$
-2	2.5	$-128\ \mu s$	$-41.8°$
-1	5	$-64\ \mu s$	$-19.5°$
0	7.5	$0\ \mu s$	$0°$
1	10	$64\ \mu s$	$19.5°$
2	12.5	$128\ \mu s$	$41.8°$
3	15	$192\ \mu s$	$90°$

the AND gate with $N = nk$ bits. Choosing the accumulator bit-width is a trade-off between not adding too many integer bits (saving resources) and avoiding overflow for audio frames with high volume.

VI. EVALUATION

In this section, we compare and analyze the accuracy and resource consumption of the following SSL designs:

1) The conventional design that uses weighted binary arithmetic with 4-bit precision signed inputs (Design (1)).
2) The counter-based CC shown in Fig. 11 with 4-bit unipolar (unsigned) inputs (Design (2)).
3) The counter-based CC architecture shown in Fig. 12 with LTC6992 for analog SN generation and 4-bit precision unipolar inputs (Design (3)).
4) The architecture of Fig. 12 with OR-based MAC units (Fig. 8) and LTC6992 for analog SN generation of the 4-bit precision unipolar inputs (Design (4)).
5) The architecture of Fig. 11 with OR-based MAC units and digital 4-bit precision SNGs (Design (5)).
6) The stochastic circuits using signed instead of unsigned inputs (Design (6)).
7) The architecture of Fig. 13 with Sobol-based bit-streams (Design (7)).

For all designs, the CC output has a bit-width twice that of the input due to the multiplication stage of the MAC unit. The fraction bits of the accumulation stage are truncated to represent the result with 2B bits without overflow. Unless otherwise stated, we round the CC inputs of all designs to the nearest representable value. We want to emphasize that the resource and accuracy differences are due to different CC implementations and SNGs. The post-processing tasks (maximum search, mapping, and average filter) are the same for all designs and are implemented with conventional binary arithmetic.

A. Accuracy

We use the Grid corpus audio library [46], which consists of 50 files with male and female speakers for accuracy evaluations. All 50 files sum up to 90.08 s of audio data with 76.03 s audible speech (CC results are above the VAD threshold) and 14.05 s breaks. The distance between the sound source and the center of the two microphones is 1 m, and the speaker radiates with a 60 dB sound pressure level (SPL). The microphone SNR is 75 dB, and reverberations are disabled.[3] We sweep the sound source from $-90°$ to $90°$ in a $5°$ grid.

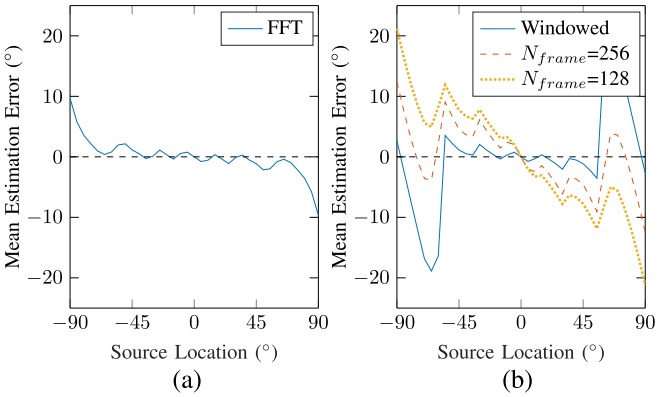


Fig. 15. The systematic error of the sound source localizer architecture. (a) Frame length of 256, Hanning window, FFT-based TDE, (b) variable frame length with and without Hanning window.

The accuracy results show the estimated source location ($\hat{\varphi}$) over the actual source location (φ). We show the results for $\hat{\varphi}$ rather than z_{avg} because $\hat{\varphi}$ is more intuitive and the LEDs of the *Input/Output board* also indicate $\hat{\varphi}$. We list the relation between i_{max}, i'_{max}, τ_d and $\hat{\varphi}$ in Table II. The index i_{max} is proportional to the TDE with $\tau_d = i_{max}/f_s$ (9). Both $i'_{max} = (i_{max}+3)\times 2.5$ and $z_{avg} \approx \text{mean}(i'_{max})$ are related to $\hat{\varphi}$ through the inverse sinus.

1) Systematic Error: Fig. 15 shows the mean estimation error of four sound source localizers with double-precision arithmetic and TDEs in the time and frequency domain. We can see a location-dependent bias, even with high computational effort and no external disturbances, such as reverberation and noise. Fig. 15(b) shows the mean estimation error for the sound source localizer with $N_{frame} = 128$ and $N_{frame} = 256$. The dataset labeled with Windowed comes from a sound source localizer that has an additional processing block preceding the CC, where frames are multiplied with the Hanning window. Fig. 15(b) shows that applying an appropriate window function after framing and increasing the window length reduces the systematic error. Fig. 15(a) also uses a Hanning window and a frame length of $N_{frame} = 256$ but computes the time delay through multiplication in the frequency domain instead of CC in the time domain. The sound source localizer with FFT-based TDEs has a lower systematic error for $|\phi| > 45°$ than the one labeled with Windowed. Fig. 15(a) shows the lowest possible error with two microphones and the averaging filter as post-processing. The sound source localizers in both plots have a systematic bias towards positive mean errors for source locations at negative angles and a tendency towards negative mean errors for source locations with positive angles.

Mathematically, by cutting the microphone signals into frames, we multiply the *infinite* microphone signals with the rectangular function of value 1 during the current frame and 0s otherwise. In the frequency domain, the framing is equal to a convolution with the *sinc* function.[4] The *sinc* function shows a high peak at $t = 0$, so we are more likely to see the

[3]Decreasing the SNR and increasing reverberation leads to higher SSL errors for all designs.

[4]The Fourier transform of the rectangular function is the sinc function.

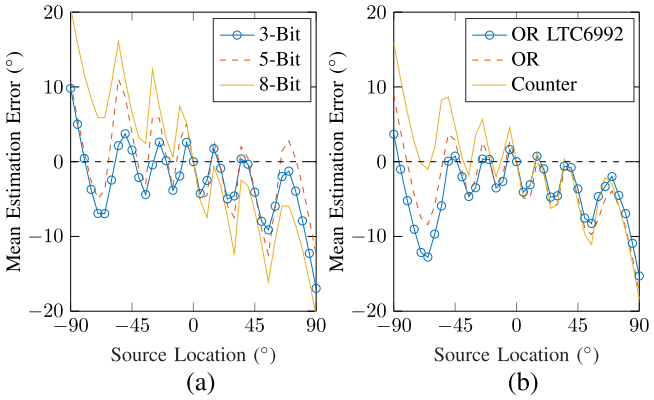


Fig. 16. Mean estimation error of (a) Design (6) (exact and bipolar) and (b) Design (2), (4) and (5) (4-bit unipolar inputs).

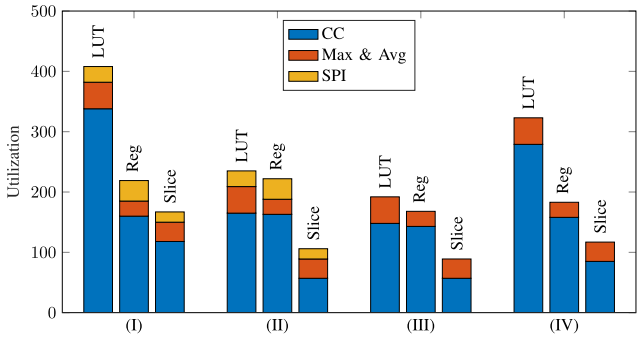


Fig. 17. LUT, registers and slice utilization of the digital processing blocks. (I) Weighted binary, (II) Counter, (III) Counter with LTC6992, (IV) OR with LTC6992.

maximum of the CC result at $i_{max} = 0$. As $i_{max} = 0$ occurs more frequently, we see a bias towards 0° and a positive or negative mean error depending on the source location. As a countermeasure, we can either increase the window length, for example, double N_{frame} in Fig. 15(b), to widen the peak of the *sinc* function or use a proper window function such as the Hanning function to avoid the *sinc* altogether.

We note that for source locations close to $\varphi = \pm 90°$, the average IIR filter cannot compensate errors with an opposite sign. For example, for a sound source at $\varphi = -90°$, 84 % of all CC evaluations have the maximum at $c[-3]$. However, 16 % of all CC evaluations are either $c[-2]$ and $c[-1]$ and cause a positive bias.

2) Low Precision and Approximate Computing: Fig. 16 shows the systematic bias of the sound source localizers that use fixed-point arithmetic, time domain CC, no Hanning window function, and a frame length of $N_{frame} = 128$. Fig. 16(a) shows the mean SSL error for Design (6). We observe the following three trends:

1) The error due to not using a window function and a short window length is higher for designs with 8-bit precision than those with 3 or 5-bit precision. For example, the mean estimation error at $\varphi = 55°$ is $-8.8°$ for 2-bit and $-14.9°$ for 8-bit precision.
2) For 5-bit and 8-bit precision, the results are point symmetrical with $\varphi = 0°$. For example, for 5 and 8-bit, the mean estimation error for $\varphi = -30°$ is the negation of the mean estimation error for $\varphi = +30°$. But this does not apply to 3-bit precision. The probability of having multiple equal maxima in the CC result increases with low precision. The Max Search block forwards the index of the first maxima (with the lowest index) to the Map & Average block, which causes a bias towards $\hat{\varphi} = -90°$.
3) The mean estimation error plotted over the source location follows a triangle function. For source locations at $\varphi \in \{-41.8°, -19.5°, 0°, 19.5°$ and $41.8°\}$, the mean error is approximately zero because these angles correspond to $i_{max} \in \{-2, -1, 0, 1, 2\}$ (see Table III). Between these angles, the averaging filter keeps the error below 20°, but for $i_{max} = \pm 3$ ($\pm 90°$), the error is higher because of the systematic error.

Fig. 16(b) shows the mean estimation error for the unipolar designs (Designs (2), (4) and (5)). The error difference between the unipolar and bipolar format is low, but the difference between the OR-based and the counter-based summation is high due to more bias towards $-90°$. Using OR gates for summation limits the output range of the CC and further increases the probability of having multiple equal maxima in the CC result. For the designs in Fig. 16, we also list the mean absolute error (MAE) in Table III. The right-most column in Table III shows the mean of MAEs, which is below 8.6° for all designs. The MAE decreases with lower input precision because low precision designs are less susceptible to the systematic error. In our simulations, we observed that the rounding method during the quantization of the inputs affects the estimation error, and truncation rather than rounding gives a lower MAE. For example, the average MAE (last column of Table III) changes to 4.7°; 4.9°; 5.4°; 8.2° for 3, 4, 5, and 8-bit precision, respectively (an improvement of 3% to 9%.). The results are similar for the unipolar designs. Truncation leads to sharper peaks in the CC result and better TDE.

B. Resource Consumption

Saving hardware resources is one of the main goals of implementing SC systems. This section shows our synthesis results produced by the Synopsys Design Compiler v2018.06 with the 45nm FreePDK CMOS library [47] for the ASIC design flow and Vivado Design Suite for synthesis on the FPGA. The HDL synthesis tools are used to analyze the digital processing blocks on the right-side of the vertical line in Fig 5.

The *analog board*, shown in Fig. 6, has four black current sense resistors for measuring power consumption. The low current consumption of the *analog board* in conjunction with the low resistance of the current sense resistor ($R = 0.025\ \Omega$), causes a small voltage drop over the current sense resistor that is difficult to measure. For example, the voltage drop over the current sense resistor of LTC1098 is approximately $0.1\ mA \times 0.025\ \Omega = 2.5\ \mu V$.

Fig. 17 and Fig. 18 show the resource consumption for the Max Search and Map & Average (referred to as Max & Average), SPI, and CC processing blocks for FPGA and ASIC implementations. The resource consumption of the 4-bit bipolar design (Design 6) is equal to that of the counter-based design (Design 2). The figures visualize the area reports of Vivado and Synopsys. However, we only show the power

TABLE III
MAE OF THE ANGLE ESTIMATION ($\hat{\varphi}$) IN DEGREE

		φ (°)													
		-90	-75	-60	-45	-30	-15	0	15	30	45	60	75	90	**Mean**
Design (6)	3-Bit Inputs	8.0	6.7	5.7	2.0	2.7	3.8	0.0	2.2	4.6	4.1	6.4	5.9	15.5	**4.9**
	4-Bit Inputs	6.2	8.5	5.1	3.2	2.9	4.0	0.0	3.0	5.3	3.8	6.4	6.4	11.0	**5.2**
	5-Bit Inputs	7.8	7.6	5.5	3.8	6.0	3.5	0.0	3.0	7.4	3.9	6.6	6.8	10.0	**5.6**
	8-Bit Inputs	18.5	9.0	10.6	6.5	12.3	2.7	0.0	2.7	12.3	6.5	10.6	9.0	18.6	**8.6**
Counter (2)		13.8	6.0	4.7	4.7	3.6	2.3	0.0	1.9	5.8	5.3	6.2	7.5	16.5	**5.8**
OR (3)		6.2	7.9	7.0	2.9	3.0	3.4	0.1	2.0	5.7	4.8	7.6	6.7	15.6	**5.3**
OR LTC6992 (4)		2.0	11.3	10.3	2.3	3.5	3.5	0.0	1.5	4.5	3.6	4.6	5.0	13.5	**5.1**

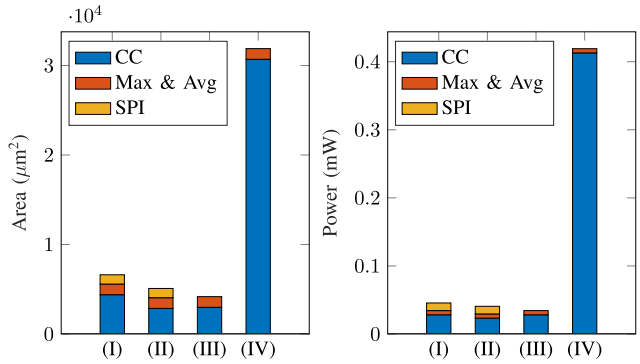


Fig. 18. Area and power consumption of SC and weighted binary designs for 4-bit precision and f_s = 15.6 kHz. (I) Weighted binary, (II) Counter, (III) Counter with LTC6992, (IV) OR with LTC6992.

consumption for the ASIC design. Power estimations from Vivado are not precise enough as the static power consumption of the FPGA is significantly higher than the power consumption of the individual processing blocks. Synopsys reports that the designs with SC components have higher dynamic power consumption but lower static power.[5] Vivado and Synopsys estimate that the CC function consumes more power than the Max & Average and SPI processing blocks. Both HDL synthesis tools agree that counter-based Design (3) with LTC6992 consumes the least resources, followed by counter-based Design (2) and conventional Design (1). The synthesis tools estimate resource consumption for OR-based Design (3) with LTC6992. Synopsys reports about ten times higher power and area consumption, whereas the utilization report of Vivado shows *only* a doubling in the look-up table (LUT) utilization compared to other CC designs. The OR-based design with LTC6992 (Design (4)) stores a bit-stream accumulator with $nk = 240$ bits (for 4-bit precision arithmetic). FPGAs are optimized for storing data, but the ASIC design synthesizes a separate flip-flop for each bit. When comparing Design (1) and Design (2), Design (2) saves 11 % power and 23 % area by using the SC-based CC design. The savings of the digital implementation increase to 25 % power and 37 % area when comparing the conventional design (Design (1)) with Design (3). However, Design (3) uses the PWM modules on the *analog board*, which consume more power than the ADC and outweigh the savings from the digital design.

[5]Vivado also reports higher dynamic power for designs with SC if we increase clock speeds and fill up the FPGA with duplicates of the HDL modules.

TABLE IV
SYNTHESIS RESULTS OF THE CC FUNCTION FOR DIFFERENT IMPLEMENTATIONS IN $f_s = 15.60$ KHZ

$M =$		**3**	**4**	**5**	**8**
Synopsys- Area (μm^2)	Conventional	3,956	4,788	6,066	9,720
	Counter Based	3,136	3,393	3,699	4,232
	Sobol Based	**866**	**1,210**	**1,353**	**1,494**
Synopsys- Power @ 15.6 kHz (μW)	Conventional	19.6	23.4	29.9	54.8
	Counter Based	18.4	27.9	65.1	92.3
	Sobol Based	**5.6**	**9.0**	**11.2**	**13.4**
Vivado- LUT	Conventional	139	257	303	596
	Counter Based	72	83	88	174
	Sobol Based	**19**	**23**	**37**	**70**
Vivado-REG	Conventional	145	160	182	352
	Counter Based	151	163	174	350
	Sobol Based	**40**	**54**	**66**	**74**
Synopsys- Critical Path (ns)	Conventional	1.32	1.65	2.25	3.28
	Counter Based	0.70	0.99	1.15	1.43
	Sobol Based	**0.56**	**0.75**	**0.85**	**0.93**
Energy (fJ)	Conventional	25.9	38.6	67.3	179.7
	Counter Based	12.9	27.6	74.9	132.0
	Sobol Based	**3.1**	**6.7**	**9.5**	**12.4**
Area Delay Product	Conventional	5,222	7,900	13,649	31,880
	Counter Based	2,195	3,359	4,254	6,052
	Sobol Based	**485**	**908**	**1,150**	**1,389**

Table IV reports the resource consumption of the counter-based CC with digital unary SN generation (used in Design 2), the conventional CC (used in Design 1), and the Sobol-based CC (Design 7). The table compares the utilization, power, area, critical path latency, energy consumption, and area-delay product for $M = 3, 4, 5$ and 8-bit precision inputs. As it can be seen, the area consumption increases with the input precision, but the differences are higher for the conventional implementation. For higher precision, the weighted binary design has larger multipliers and adders. In contrast, the area of the counter-based CC implementation hardly increases because only the counter size for SN generation, the comparator size and the accumulation change. Going from 3-bit to 8-bit inputs increases the area consumption both in the conventional and in the SC design. The utilization increase on the FPGA is in the same range. As can be seen, the Sobol-based design consumes the minimum area and power compared to all designs. Comparing the Sobol and the counter-based designs for 8-bit inputs shows 65% and 85% reduction in area and power consumption, respectively. Also, the FPGA results show considerable savings with the Sobol-based CC design. The total resource utilization in terms of LUTs and registers decreases going from conventional to the Sobol-based design.

For total latency and energy consumption of the counter-based and Sobol-based design, we need to multiply the critical path latency and energy per cycle by the bit-stream length. The accuracy of the Sobol-based design for different bit-stream lengths is reported in Table V. As can be seen, 16-

TABLE V

ACCURACY (MAE) EVALUATION OF THE SOBOL-BASED IMPLEMENTATION OF THE CC FUNCTION

Bit-Stream Size (N)	**8**	**16**	**32**	**64**	**128**	**256**	**512**	**1024**
MAE	2.72	1.36	0.54	0.38	0.37	0.08	0.04	0.02

TABLE VI

SYNTHESIS RESULTS OF THE PARALLEL SOBOL IMPLEMENTATIONS OF CC FUNCTION ($1\times, 2\times, 4\times$, AND $8\times$) FOR $M = 8$ IN $f_s = 15.60$ KHZ

Parallel Design	**1×**	**2×**	**4×**	**8×**
Synopsys- Area (μm^2)	1,494	1,693	1,975	2,491
Synopsys- Power @ 15.6 kHz (μW)	13.37	15.31	17.65	20.30
Synopsys- Critical Path (ns)	0.93	0.98	1.04	1.08
Energy (fJ)	12.43	15.00	18.36	21.92
Area Delay Product	1,389	1,659	2,054	2,690

TABLE VII

POWER CONSUMPTION OF THE *analog Board*

Total Idle		LTC6992		LTC1098		Total
17.65mW	+	2 × 0.95mW	+	0.5mW	=	20.05mW

bit Sobol bit-streams can achieve an MAE of about 1.3%. This corresponds to a total latency of 14.8 ns, energy consumption of 198.4 fJ, and area-delay product of 22,224 for the 16-bit Sobol-based design. For an MAE of 2.7% with 8-bit Sobol bit-streams, the latency, energy, and area-delay product reduce to 7.4 ns, 99.2 fJ, and 11,112, respectively. Hence, Sobol-based design with 16-bit bit-streams provides a lower area-delay product and with 8-bit bit-streams provides a lower energy and area-delay product compared to the conventional binary design. The total latency, however, is still higher than the conventional design in both cases. The hardware efficiency of the Sobol-based design can be improved by parallelization, as suggested in [44]. We implemented a $2\times$, $4\times$, and $8\times$ parallel design of the Sobol-based design to improve energy efficiency. Table VI reports the synthesis results. As can be seen, the $8\times$ parallel design that generates 8 bits in a single clock cycle reduces the latency and energy consumption to 1.08 ns and 21.9 fJ, which are significantly lower than the conventional design.

The clock speed to finish one MAC operation is constant for the weighted binary and increases exponentially in the SC-based implementations. The conventional CC always computes with 15.6 kHz. A close look at the resource consumption of the CC designs indicates that the conventional CC consumes slightly more power than the counter-based CC. However, Table IV shows it the other way for 4-bit inputs. We use a faster clock in full system simulations because all designs need one high-frequency clock to control the SPI and generate the SCLK. When simulating Design 1 and 2, we use the same 3.744 MHz clock and a clock-enable signal to enable the weighted binary CC every $f_s = 15.6\ kHz$.[6] For the designs in Table IV, we use a 15.6 kHz clock and connect the clock-enable signal to a constant high (Logic-1). Using a higher base clock and a clock-enable signal increases the power consumption of the conventional CC from 23.4 μW to 28.2 μW. Finally, Table VII reports the total power consumed by the analog board equal to 20.05 mW.

[6]We use switching annotations to increase the accuracy of the power estimations.

We also compare the proposed SSL design with some prior studies regarding the number of microphones, microphone distance, the method used, and accuracy measures in Table VIII. A remarkable difference between this work and the previous studies is *data representation*. We utilize bit-stream representation for simple hardware design.

As can be seen, FPGA is the dominant platform in most prior works. The number and distance of microphones vary depending on the application (e.g., wolf [54], elephant [55], gunshot [56], underwater vehicle localization [57], etc.). Some studies aim for a wide-range source localization by placing many microphone arrays on different nodes [58]. Almost every work has its unique success rate. Generally, angle estimation (AE) or distance error calculation are used for accuracy evaluation. Using AE error, we measured a mean range of $[4.9°, 8.6°]$ for our design. Wang et al. [7] reported >90% AE accuracy with an error margin of $\pm 10°$. In that respect, our comparable system accuracy was 100%. Lai-Hui [48] calculated the mean values of the estimated angles and reported a maximum error of 21.26°, which shows a larger error margin than our work. Sha-Li [49] reported an AE of 80% by utilizing generalized CC and the average magnitude difference function (AMDF). Mean distance error (MDE) was also reported in prior works in two forms: 1) exact error distance (e.g., 0.54 m in [50] using fast and precise localization (FPL) algorithm) and 2) accuracy or error in percentage (e.g., 98% accuracy in [17]). A correct location detection (CLD) percentage of >99% was also reported in [53]. They reported an FPGA power consummation of between 27 to 61 mW. Ye et al. [52] employed SSL in a speech recognition application and defined word error rate (WER) for performance monitoring. They utilized a beamforming approach for the SSL system, and the total resource utilization in terms of LUTs in a Virtex-4 SX FPGA was 26%. Due to using SC, the resource usage of our design was much lower than in prior works. For example, the resource occupation of the overall system in [17] was about 95%. Furthermore, the total power consumption of our analog board (20.05 mW) and the worst-case power usage of the digital CC design (92.3 μW) are promising compared to prior designs. For example, a power consumption of 108 mW and gate utilization of around $1M$ was reported for the design implemented in [6].

VII. LIMITATIONS

We reported the power consumption estimations based on the synthesis results from the Synopsys Design Compiler with a 45nm CMOS gate library. The results will be different for other tools and technology. Any changes to the base clock or computational speed can benefit either the conventional or the SC designs. Simulating the designs for different clock speeds (with clock-enable signals), different sampling frequencies, and synthesis tools would give a complete view of the power consumption.

Our analysis considers SSL as the only processing task in the digital design. However, the system usually includes other signal processing tasks. A parallel MAC architecture could be more efficient than the stream-based MAC if the audio samples

TABLE VIII
COMPARISON OF THIS WORK AND PREVIOUS SSL STUDIES

	This work	[7]	[48]	[49]	[6]	[50]	[17]	[51]	[52]	[53]
Frequency	15.6 kHz	96 kHz	48 kHz	32 kHz	20 kHz	48 kHz	n/d	16 kHz	16 kHz	44.1 kHz
Method	TDE w/ CC	AMDF	Gen. CC	Gen. CC& AMDF	Gen. CC	FPL	Beamform.	TDE w/ CC	Beamform.	Beamform.
# of Microphones	2	2	2	2	2	4	8	3	2	4, 8, 16
Platform	Analog & FPGA	FPGA	DSP	Computer	FPGA	FPGA	μController	FPGA	FPGA	FPGA
Microphone Distance (d)	0.066 m	0.112 m	0.130 m	0.200 m	0.400 m	n/d	0.061 m	0.130 m	n/d	0.141 m, 0.076 m, 0.039 m
Accuracy	AE (Mean Error) 4.9°∼8.6°	AE (±10°) >90%	AE (Mean Error) 21.26°	AE (Accuracy) 80%	n/d	MDE 0.54 m	MDE 98%	n/d	WER 41.8%	CLD >99%
Misc.	LUT: 19∼70 (3∼8-bit) Power: 20.05 mW (analog) 13.4μW (Sobol-8)	Gate utilization: 188,000	n/d	n/d	Gate utilization: 1,192,793 Power: 108 mW	LUT: 94,032 (ZynQ UltraScale)	μCont. Occupation: 45∼95%	LUT: 16% Virtex-4 XC	LUT: 26% Virtex-4 SX	Power: 27.14∼ 61.71 mW

are stored (because they are also needed elsewhere) and do not add costs to SSL. The counter-based MAC is not suitable for parallel MAC architecture. It can be replaced with accumulative parallel counters (APCs). Even OR-based summation could be a competitive alternative because it does not need a bit-stream accumulator. We need $N_{frame} = 128$ multipliers and adders in the parallel architecture (instead of only one), changing the resource comparison of SC and weighted binary designs.

The resource consumption savings with SC are maximized if the cost of other sub-systems is similarly low. The block diagram in Fig. 5 could additionally contain a processing block to apply a window function and an adaptive whitening filter which both increase the accuracy of estimations but lower the overall gain through using SC. Further, SC designs cannot benefit from analog SN generation if the first digital processing block uses conventional digital binary arithmetic.

VIII. DISCUSSION AND CONCLUSION

In this work, for the first time to the best of our knowledge, we proposed an accurate SC-based SSL system. To prove the functionality of the proposed sound source localizer, we implemented the full system and signal processing chain end to end. This resulted in two prototype circuit boards and multiple HDL designs. Implementing a separate analog processing chain for SC did not provide power savings because generating voltage-controlled PWM signals with LTC6992 consumes about two times more power than the conventional analog-to-digital conversion with LTC1098. This could stem from fundamental difference in the CMOS technology used for fabricating the two chips. Nevertheless, the analog implementation served as the proof of concept that SC does not rely on conventional ADCs. The digital design showed the weaknesses and strengths of SC. On the one hand, the approximate OR-based design has lower flexibility and more design constraints than the counter-based and conventional designs because the OR-based adder is sensitive to data value changes. On the other hand, the counter-based CC is accurate and easy to use. Our synthesis results showed that the area consumption of the ASIC design decreases by 21% for 3-bit inputs and by 39% for 5-bit inputs. The power consumption increases with computational precision for weighted binary but exponentially faster for SC. For 3-bit input precision, the SC design consumes 18.4 μW with an 874 kHz clock, and the conventional design consumes 19.6 μW with a sampling clock frequency of 15.6 kHz. When we increase the computational precision to 4-bit or 5-bit, we can keep the clock speed of the conventional design but must increase the clock frequency of the stochastic circuit to finish the computations in the same time. For 4-bit, for example, the power consumption increases to 27.9 μW for the SC design but only to 23.4 μW for the conventional design. To keep the power consumption in favor of the SC-based design, we can decrease the sampling rate to loosen the time constraints, or use technologies with lower dynamic power. We also proposed a new CC design based on LD Sobol bit-streams. The Sobol-based design takes advantage of utilizing a low-cost MUX-based bit-stream generator, and thus consumes less area and power compared to the counter-based CC design. The SC-based CC design can be reused in other applications, which use low-precision CC functions, and can provide the same resource savings as for SSL. The readers can view a video demonstration of the implemented design on [59] and access the source code on [22].

IX. APPENDIX: ANALOG BOARD DETAILS

In this appendix, we provide further details on different components of our designed analog board shown in Fig. 6.

1) The audio input to the *analog board* connects with a 3.5 mm stereo audio jack on the left side of the board. We use two microphone signals and a stereo sound card.

2) The stereo signal from the audio jack is fed into two (AC)-coupled, non-inverting amplifiers, marked with yellow rectangles in Fig. 6. The circuit uses a Texas Instrument OPA322 OPAMP and is shown in Fig. 19. The two 100 $k\Omega$ resistors at the input of the circuit in Fig. 19 bias the input AC signal, and the capacitor decouples the biasing network from the supply. A 10 pF capacitor in the feedback path decouples the AC amplification from the DC operating point at the output stage. The amplifier circuit fulfills three tasks: first, the circuit brings the OPAMPs operating point to 2.5 V (mid-supply) for a maximum voltage output swing of 0 to 5 V. Second, the circuit has an adjustable gain in the $[1.15, 151]$ interval using a potentiometer with 1 $k\Omega$ to 1000 $k\Omega$. Third, resistors and capacitors work as low and high-pass filters that attenuate

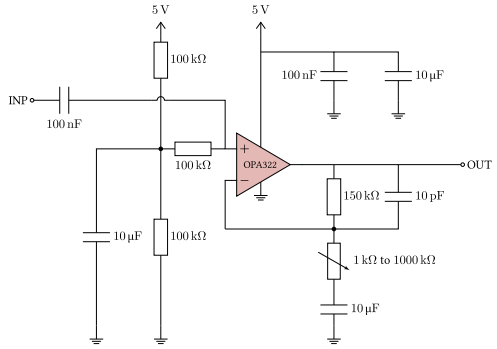


Fig. 19. Adjustable, non-inverting, analog amplifier using passive components and a Texas Instrument OPA322. The circuit is marked with yellow in Fig. 6.

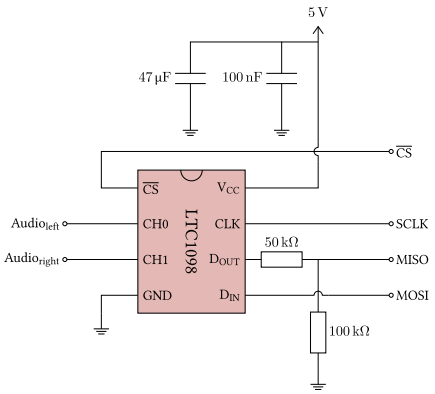


Fig. 20. Schematic of the ADC (LTC1098) circuitry with two input channels and an SPI. The circuit is marked with blue in the center of Fig. 6.

noise and pass-through human speech. The amplified signal is forwarded to an ADC and two half-wave rectifier circuits.

3) LTC1098 is an 8-bit ADC with two input channels. The circuit is shown in Fig. 20 and is marked with blue in Fig. 6. LTC1098 offers a channel selection through software, is used for both (amplified) microphone signals, and connects to the FPGA over an SPI. The pins driven by the FPGA (master out slave in (MOSI), chip select (CS), serial clock (SCLK)) are in the $[0\ V, 3.3\ V]$ interval and directly connected to the ADC. The master in slave out (MISO) requires a 5 V to 3.3 V voltage divider to protect the input pin of the FPGA. Besides the ADC, the amplifier outputs connect to two half-wave rectifiers (Fig. 21). The 100 nF AC coupling capacitor at the input of the circuit in Fig. 21 separates the 2.5 V DC offset from the AC audio signal. The AC component of the input signal does not have to overcome the threshold voltage of Diode D2 because of the bias created by Diode D1 in combination with a 150 $k\Omega$ and 1 $M\Omega$ resistor. A small positive signal at the input passes through D2, and negative voltages are blocked. The voltage divider at the output limits the positive swing to 1 V for the subsequent PWM module.

4) The PWM circuits, shown in Fig. 22, receive the inputs from the half-wave rectifiers and are built with Analog Devices LTC6992 microchips. They are located on the right side of the *analog board* and marked green in Fig. 6. The voltage level at the input (INP) pin of LTC6992 controls the duty cycle of the generated PWM signal with a non-linear transfer function.

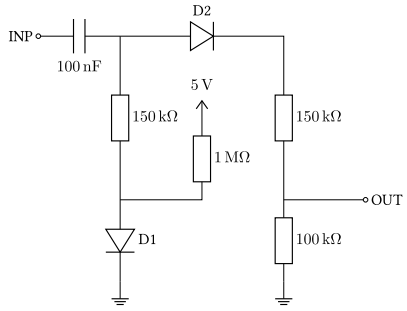


Fig. 21. Schematic of the half-wave rectifier circuit, marked with red in Fig. 6.

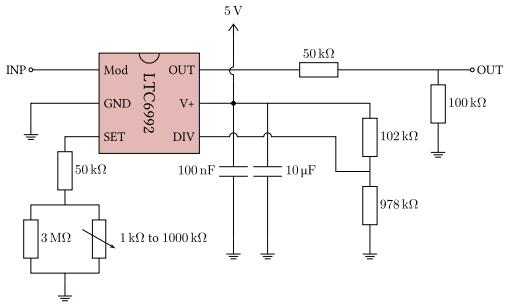


Fig. 22. LTC6992 is a voltage-controlled pulse-width-modulator from Analog Devices. The circuit converts voltages to PWM signals for unary SC and is marked green in Fig. 6.

The duty cycle is $D = 0\%$ for input voltages in the 0 V to 0.1 V interval and is $D = 100\%$ for input voltages in the 0.9 V to 1 V interval. The transfer function is $D_{out} = (V_{Mod} - 0.1) \times 1.25$ for V_{mod} in the 0.1 V to 0.9 V interval. Fig. 4 shows a PWM signal with $D = 75\%$, representing a voltage of $V_{mod} = \frac{0.75}{1.25} + 0.1 = 0.7\ V$, to give an example for the non-linear conversion. The passive components at the SET and DIV inputs determine the frequency of the PWM signal. The potentiometer at the SET input allows adjusting the frequency in the 15 kHz to 250 kHz interval. If not stated otherwise, one of the two circuits is set to $f_{PWM,1} = 234$ kHz, and the second is set to $f_{PWM,2} = 249.6$ kHz. The two PWM signals are sampled with a clock frequency of 3.744 MHz at the input register of the FPGA, which results in bit-streams with relatively prime periods of $n = 16$ and $k = 15$

$$n = \frac{f_{clk}}{f_{PWM,1}} = \frac{3.744 \cdot 10^6}{234 \cdot 10^3} = 16 \tag{22}$$

$$k = \frac{f_{clk}}{f_{PWM,2}} = \frac{3.744 \cdot 10^6}{249.6 \cdot 10^3} = 15. \tag{23}$$

For accurate stochastic multiplication, we need 15 periods of $f_{PWM,1}$ and 16 periods of the $f_{PWM,2}$ which brings us to the equivalent audio sampling frequency:

$$f_s = \frac{f_{PWM,1}}{15} = \frac{f_{PWM,2}}{16} = 15.6\ kHz \tag{24}$$

also used by the ADC. A voltage divider after LTC6992 reduces the signal voltage (5 V to 3.3 V) because the FPGA pins are limited to 3.3 V.

5) The outputs of the *analog board* are connected to the FPGA through a PMOD cable. The PMOD connection has a total of 12 pins. Eight are reserved for signals, two connected to the ground and two for the power. The board uses both ground pins and six signal pins. Four pins are used for the SPI protocol, and two lanes carry the outputs of the PWM circuits.

REFERENCES

[1] P. Schober, S. N. Estiri, S. Aygun, N. TaheriNejad, and M. H. Najafi, "Sound source localization using stochastic computing," in *Proc. 41st IEEE/ACM Int. Conf. Comput.-Aided Design*, Oct. 2022, pp. 1–9.

[2] I. Tashev, *Sound Capture and Processing: Practical Approaches*. Hoboken, NJ, USA: Wiley, 2009.

[3] A. Alaghi, W. Qian, and J. P. Hayes, "The promise and challenge of stochastic computing," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 37, no. 8, pp. 1515–1531, Aug. 2018.

[4] W. Qian, X. Li, M. D. Riedel, K. Bazargan, and D. J. Lilja, "An architecture for fault-tolerant computation with stochastic logic," *IEEE Trans. Comput.*, vol. 60, no. 1, pp. 93–105, Jan. 2011.

[5] M. H. Najafi, S. Jamali-Zavareh, D. J. Lilja, M. D. Riedel, K. Bazargan, and R. Harjani, "Time-encoded values for highly efficient stochastic circuits," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 25, no. 5, pp. 1644–1657, May 2017.

[6] D. Nguyen, P. Aarabi, and A. Sheikholeslami, "Real-time sound localization using field-programmable gate arrays," in *Proc. Int. Conf. Multimedia Expo. (ICME)*, Jul. 2003, p. 829.

[7] J.-F. Wang, Y.-C. Jiang, and Z.-W. Sun, "FPGA implementation of a novel far-field sound localization system," in *Proc. TENCON IEEE Region Conf.*, Nov. 2009, pp. 1–4.

[8] I. Aleksi, Ž. Hocenski, and P. Horvat, "Acoustic localization based on FPGA," in *Proc. Int. Conv. MIPRO*, Opatija, Croatia, May 2010, pp. 656–658.

[9] V. Kunin, M. Turqueti, J. Saniie, and E. Oruklu, "Direction of arrival estimation and localization using acoustic sensor arrays," *J. Sensor Technol.*, vol. 1, no. 3, pp. 71–80, 2011.

[10] A. Abdeen and L. Ray, "Design and performance of a real-time acoustic beamforming system," in *Proc. IEEE Sensors*, Nov. 2013, pp. 1–4.

[11] J. Kotus, "Multiple sound sources localization in free field using acoustic vector sensor," *Multimedia Tools Appl.*, vol. 74, no. 12, pp. 4235–4251, Jun. 2015.

[12] J. Tiete, F. Domínguez, B. Silva, L. Segers, K. Steenhaut, and A. Touhafi, "SoundCompass: A distributed MEMS microphone array-based sensor for sound source localization," *Sensors*, vol. 14, no. 2, pp. 1918–1949, Jan. 2014.

[13] M. M. Faraji, S. B. Shouraki, and E. Iranmehr, "Spiking neural network for sound localization using microphone array," in *Proc. 23rd Iranian Conf. Electr. Eng.*, May 2015, pp. 1260–1265.

[14] T. Sledevic and R. Laptik, "An evaluation of hardware-software design for sound source localization based on SoC," in *Proc. Open Conf. Electr., Electron. Inf. Sci. (eStream)*, Apr. 2017, pp. 1–4.

[15] B. da Silva, L. Segers, A. Braeken, K. Steenhaut, and A. Touhafi, "A low-power FPGA-based architecture for microphone arrays in wireless sensor networks," in *Applied Reconfigurable Computing. Architectures, Tools, and Applications*, N. Voros, M. Huebner, G. Keramidas, D. Goehringer, C. Antonopoulos, and P. C. Diniz, Eds. Cham, Switzerland: Springer, 2018, pp. 281–293.

[16] G. Fabregat, J. A. Belloch, J. M. Badia, and M. Cobos, "Design and implementation of acoustic source localization on a low-cost IoT edge platform," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 67, no. 12, pp. 3547–3551, Dec. 2020.

[17] M. M. Faraji, S. B. Shouraki, E. Iranmehr, and B. Linares-Barranco, "Sound source localization in wide-range outdoor environment using distributed sensor network," *IEEE Sensors J.*, vol. 20, no. 4, pp. 2234–2246, Feb. 2020.

[18] A. Khanal et al., "Search disaster victims using sound source localization," 2021, *arXiv:2103.06049*.

[19] M.-A. Chung, H.-C. Chou, and C.-W. Lin, "Sound localization based on acoustic source using multiple microphone array in an indoor environment," *Electronics*, vol. 11, no. 6, p. 890, Mar. 2022.

[20] D. Jenson and M. Riedel, "A deterministic approach to stochastic computation," in *Proc. 35th Int. Conf. Comput.-Aided Design*, Nov. 2016, pp. 1–8.

[21] M. H. Najafi, D. J. Lilja, and M. Riedel, "Deterministic methods for stochastic computing using low-discrepancy sequences," in *Proc. Int. Conf. Comput.-Aided Design*, Nov. 2018, pp. 1–8.

[22] P. Schober. (2022). *Sound Source Localization Using Stochastic Computing*. Accessed: Dec. 29, 2022. [Online]. Available: https://github.com/serco425/stochastic_computing_sound_source_localization/

[23] T. Oya, S. Iwase, R. Natsume, T. Itazuri, S. Yamaguchi, and S. Morishima, "Do we need sound for sound source localization?" in *Computer Vision ACCV 2020*, H. Ishikawa, C.-L. Liu, T. Pajdla, and J. Shi, Eds. Cham, Switzerland: Springer, 2021, pp. 119–136.

[24] J. Ko, H. Kim, and J. Kim, "Real-time sound source localization for low-power IoT devices based on multi-stream CNN," *Sensors*, vol. 22, no. 12, p. 4650, Jun. 2022.

[25] P.-A. Grumiaux, S. Kitić, L. Girin, and A. Guérin, "A survey of sound source localization with deep learning methods," *J. Acoust. Soc. Amer.*, vol. 152, no. 1, pp. 107–151, Jul. 2022.

[26] T.-H. Tan, Y.-T. Lin, Y.-L. Chang, and M. Alkhaleefah, "Sound source localization using a convolutional neural network and regression model," *Sensors*, vol. 21, no. 23, p. 8031, Dec. 2021.

[27] B. da Silva, A. Braeken, and A. Touhafi, "FPGA-based architectures for acoustic beamforming with microphone arrays: Trends, challenges and research opportunities," *Computers*, vol. 7, no. 3, p. 41, Aug. 2018.

[28] L. Chen, G. Chen, L. Huang, Y.-S. Choy, and W. Sun, "Multiple sound source localization, separation, and reconstruction by microphone array: A DNN-based approach," *Appl. Sci.*, vol. 12, no. 7, p. 3428, Mar. 2022.

[29] M. U. Liaquat, H. S. Munawar, A. Rahman, Z. Qadir, A. Z. Kouzani, and M. A. P. Mahmud, "Localization of sound sources: A systematic review," *Energies*, vol. 14, no. 13, p. 3910, Jun. 2021.

[30] M. S. Brandstein, J. E. Adcock, and H. F. Silverman, "A closed-form location estimator for use with room environment microphone arrays," *IEEE Trans. Speech Audio Process.*, vol. 5, no. 1, pp. 45–50, Jan. 1997.

[31] P. Svaizer, M. Matassoni, and M. Omologo, "Acoustic source location in a three-dimensional space using crosspower spectrum phase," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Apr. 1997, pp. 231–234.

[32] J. H. DiBiase, "A high-accuracy, low-latency technique for talker localization in reverberant environments using microphone arrays," Ph.D. dissertation, Division Eng., Brown Univ., Providence, RI, USA, 2000.

[33] C. H. Knapp and G. C. Carter, "The generalized correlation method for estimation of time delay," *IEEE Trans. Acoust., Speech Signal Process.*, vol. ASSP-24, no. 4, pp. 320–327, Aug. 1976.

[34] S. T. Birchfield and D. K. Gillmor, "Acoustic source direction by hemisphere sampling," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, May 2001, pp. 3053–3056.

[35] J. Delosme, M. Morf, and B. Friedlander, "Source location from time differences of arrival: Identifiability and estimation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Apr. 1980, pp. 818–824.

[36] B. R. Gaines, *Stochastic Computing Systems*. Boston, MA, USA: Springer, 1969, pp. 37–172, doi: 10.1007/978-1-4899-5841-9_2.

[37] S. Liu and J. Han, "Energy efficient stochastic computing with Sobol sequences," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2017, pp. 650–653.

[38] C. F. Chuah and T. N. Kumar, "Fast and exact multiple-input unary-to-binary multiplier with variable precision for stochastic computing," *Electron. Lett.*, vol. 56, no. 10, pp. 482–485, May 2020.

[39] M. H. Najafi, S. Jamali-Zavareh, D. J. Lilja, M. D. Riedel, K. Bazargan, and R. Harjani, "An overview of time-based computing with stochastic constructs," *IEEE Micro*, vol. 37, no. 6, pp. 62–71, Nov. 2017.

[40] A. Alaghi and J. P. Hayes, "Exploiting correlation in stochastic circuit design," in *Proc. IEEE 31st Int. Conf. Comput. Design (ICCD)*, Oct. 2013, pp. 39–46.

[41] V. T. Lee, A. Alaghi, and L. Ceze, "Correlation manipulating circuits for stochastic computing," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2018, pp. 1417–1422.

[42] P. Schober, M. H. Najafi, and N. TaheriNejad, "High-accuracy multiply-accumulate (MAC) technique for unary stochastic computing," *IEEE Trans. Comput.*, vol. 71, no. 6, pp. 1425–1439, Jun. 2022.

[43] V. T. Lee, A. Alaghi, J. P. Hayes, V. Sathe, and L. Ceze, "Energy-efficient hybrid stochastic-binary neural networks for near-sensor computing," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2017, pp. 13–18.

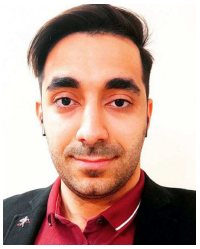
[44] S. Asadi, M. H. Najafi, and M. Imani, "A low-cost FSM-based bit-stream generator for low-discrepancy stochastic computing," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Feb. 2021, pp. 908–913.

[45] *MATLAB Sobolset*. Accessed: Dec. 29, 2022. [Online]. Available: https://www.mathworks.com/help/stats/sobolset.html#brx24a7-6
[46] M. Cooke, J. Barker, S. Cunningham, and X. Shao, "An audio-visual corpus for speech perception and automatic speech recognition," *J. Acoust. Soc. Amer.*, vol. 120, no. 5, pp. 2421–2424, Nov. 2006.
[47] (2008). *NCSU FreePDK 45nm Library*. Accessed: Dec. 29, 2022. [Online]. Available: https://eda.ncsu.edu/freepdk/freepdk45/
[48] S. Lai-Hui, "Implementation of DOA for speech using OMAP5912 on a wheeled robot," M.S. thesis, Inst. Elect. Control Eng., Nat. Chiao Tung Univ., Taipei City, Taiwan, 2005.
[49] T. Sha-Li, "Two-dimensional source localization: Implementation and discussions of time domain methodologies," Master's thesis, Dept. Comput. Sci., Nat. Tsing Hua Univ., Taipei City, Taiwan, 2005.
[50] X. Ding et al., "FRL: Fast and reconfigurable accelerator for distributed sound source localization," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 41, no. 11, pp. 3922–3933, Nov. 2022.
[51] S. Jin, D. Kim, H. S. Kim, C. H. Lee, J. S. Choi, and J. W. Jeon, "Real-time sound source localization system based on FPGA," in *Proc. 6th IEEE Int. Conf. Ind. Informat.*, Jul. 2008, pp. 673–677.
[52] H. Ye et al., "FPGA implementation of dual-microphone delay-and-sum beamforming for in-car speech enhancement and recognition," in *Proc. AutoCRC Conf.*, Melbourne, VIC, Australia, 2009, pp. 1–12.
[53] L. Petrica, "An evaluation of low-power microphone array sound source localization for deforestation detection," *Appl. Acoust.*, vol. 113, pp. 162–169, Dec. 2016.
[54] M. Papin, J. Pichenot, F. Guérold, and E. Germain, "Acoustic localization at large scales: A promising method for grey wolf monitoring," *Frontiers Zoology*, vol. 15, no. 1, pp. 1–10, Dec. 2018.
[55] C. M. Dissanayake, R. Kotagiri, M. N. Halgamuge, and B. Moran, "Improving accuracy of elephant localization using sound probes," *Appl. Acoust.*, vol. 129, pp. 92–103, Jan. 2018.
[56] G. Valenzise, L. Gerosa, M. Tagliasacchi, F. Antonacci, and A. Sarti, "Scream and gunshot detection and localization for audio-surveillance systems," in *Proc. IEEE Conf. Adv. Video Signal Based Surveill.*, Sep. 2007, pp. 21–26.
[57] P. Felisberto, O. Rodriguez, P. Santos, E. Ey, and S. M. Jesus, "Experimental results of underwater cooperative source localization using a single acoustic vector sensor," *Sensors*, vol. 13, no. 7, pp. 8856–8878, 2013.
[58] M. U. Liaquat, H. S. Munawar, A. Rahman, Z. Qadir, A. Z. Kouzani, and M. A. P. Mahmud, "Sound localization for ad-hoc microphone arrays," *Energies*, vol. 14, no. 12, p. 3446, Jun. 2021.
[59] P. Schober. (2022). *Sound Source Localization using Stochastic Computing*. Accessed: Dec. 29, 2022. [Online]. Available: https://youtu.be/ETfOnW–0bU

Peter Schober received the B.Sc. degree in electronics and information technology from Johannes Keppler University, Linz, Austria, in 2017, and the M.Sc. degree in embedded systems from TU Wien (formerly known as the Vienna University of Technology), Vienna, Austria, in 2022. From September to December 2019, he was a Visiting Research Scholar at the School of Computing and Informatics, University of Louisiana, LA, USA. Currently, he is employed as a FPGA Engineer at ITK Engineering in Vienna.

Seyedeh Newsha Estiri received the B.Sc. degree in computer engineering–hardware system design from the Iran University of Science and Technology, Tehran, Iran, in 2020. She is currently pursuing the Ph.D. degree in computer engineering with the University of Louisiana at Lafayette. Her research interests include stochastic computing, machine learning applications and hardware, and computer architecture. She was selected as a DAC Young Fellow in DAC 2022.

Sercan Aygun (Member, IEEE) received the B.Sc. degree in electrical and electronics engineering and a double major in computer engineering from Eskisehir Osmangazi University, Turkey, in 2013, the first M.Sc. degree in electronics engineering from Istanbul Technical University, in 2015, the second M.Sc. degree in computer engineering from Anadolu University, in 2016, and the Ph.D. degree in electronics engineering from Istanbul Technical University, in 2022. He is currently a Post-Doctoral Researcher at the University of Louisiana at Lafayette, USA. He works on emerging computing technologies, including stochastic computing in computer vision and machine learning. His Ph.D. work has appeared in several Ph.D. Forums of top-tier conferences, such as DAC and ESWEEK. He received the Best Scientific Research Award of the ACM SIGBED Student Research Competition (SRC) ESWEEK 2022.



Amir Hossein Jalilvand received the B.Sc. degree in computer engineering from Bu-Ali Sina University, Hamadan, Iran, and the M.Sc. degree in computer engineering–computer architecture from the Iran University of Science and Technology, Tehran, Iran. He is currently pursuing the Ph.D. degree in computer engineering. His research interests include stochastic and unary computing, computer architecture, fuzzy logic, and machine learning.

M. Hassan Najafi (Member, IEEE) received the B.Sc. degree in computer engineering from the University of Isfahan, Iran, in 2011, the M.Sc. degree in computer architecture from the University of Tehran, Iran, in 2014, and the Ph.D. degree in electrical engineering from the University of Minnesota, Twin Cities, USA, in 2018. He is currently an Assistant Professor with the School of Computing and Informatics, University of Louisiana, LA, USA. He has authored/coauthored more than 60 peer-reviewed papers and has been granted five U.S. patents with more pending. His research interests include stochastic and approximate computing, unary processing, in-memory computing, and hyperdimensional computing. In recognition of his research, he received the 2018 EDAA Outstanding Dissertation Award, the Doctoral Dissertation Fellowship from the University of Minnesota, and the Best Paper Award at the ICCD'17. He was an Editor of the IEEE JOURNAL ON EMERGING AND SELECTED TOPICS IN CIRCUITS AND SYSTEMS.

Nima TaheriNejad (Member, IEEE) received the Ph.D. degree in electrical and computer engineering from The University of British Columbia (UBC), Vancouver, Canada, in 2015. He is currently a Full Professor at Heidelberg University, Heidelberg, Germany, and affiliated with TU Wien (formerly known also as Vienna University of Technology), Vienna, Austria, as the PI of two projects. He has published three books, three patents, and more than 80 articles. His research interests include in-memory computing, cyber-physical and embedded systems, systems on chip, memristor-based circuit and systems, self-* systems, and health-care. He has received several awards and scholarships from universities, conferences, and competitions he has attended. This includes the Best University Booth Award at DATE 2021, the First Prize in the 15th Digilent Design Contest (2019) and in the Open-Source Hardware Competition at Eurolab4HPC (2019), and the Best Teacher and Best Course Awards at TU Wien (2020). He was an organizer and the chair of various conferences and workshops. He has served as a reviewer and an editor for many journals and conferences.