CS-Audio: A 16 pJ/b 0.1–15 Mbps Compressive Sensing IC With DWT Sparsifier for Audio-AR

Gaurav Kumar K, *Graduate Student Member, IEEE*, Baibhab Chatterjee, *Graduate Student Member, IEEE*, and Shreyas Sen, *Senior Member, IEEE*

***Abstract*— With the expansion of sensor nodes to newer avenues of technologies, such as the Internet of things (IoT), internet of bodies (IoB), augmented reality (AR), and mixed reality, the demand to support high-speed operations, such as audio and video, with a minimal increase in power consumption is gaining much traction. In this work, we focus on these nodes operating in audio-based AR (AAR) and explore the opportunity of supporting audio at a low power budget. For sensor nodes, communicating one bit of data usually consumes significantly higher power than the power associated with sensing and processing/computing one data bit. Compressing the number of communication bits at the expense of a few computation cycles considerably reduces the overall power consumption of the nodes. Audio codecs such as AAC and LDAC that currently perform compression and decompression of audio streams burn significant power and create a floor to the minimum power possible in these applications. Compressive sensing (CS), a powerful mathematical tool for compression, is often used in physiological signal sensing, such as EEG and ECG, and it can offer a promising low-power alternative to audio codecs. We introduce a new paradigm of using the CS-based approach to realize audio compression that can function as a new independent technique or augment the existing codecs for a higher level of compression. This work, CS-Audio, fabricated in TSMC 65-nm CMOS technology, presents the first CS-based compression, equipped with an ON-chip DWT sparsifier for non-sparse audio signals. The CS design, realized in a pipelined architecture, achieves high data rates and enables a wake-up implementation to bypass computation for insignificant input samples, reducing the power consumption of the hardware. The measurement results demonstrate a 3X–15X reduction in transmitted audio data without a perceivable degradation of audio quality, as indicated by the perceptual evaluation of audio quality mean opinion score (PEAQ MOS) >1.5. The hardware consumes 238 μW power at 0.65 V and 15 Mbps, which is (∼20X–40X) lower than audio codecs.**

***Index Terms*— Audio augmented reality (AR), audio compression, compressive sensing (CS), MP3, SoC design.**

I. Introduction

EMERGING applications involving Internet of Bodies (IoB) [1], augmented reality (AR), virtual reality (VR), and mixed reality require audio data transfer with low power and low latency for enhanced battery life and better user experience. For example, audio-based AR (AAR) [2] is a developing technology that incorporates auditory content generated by cloud servers or handheld mobile devices into the human's real-world acoustic space. In addition to complementing vision in AR, audio also presents a promising alternative to vision as a display modality in AR for visually impaired or blind individuals or users involved in visually demanding tasks such as driving and walking. Fig. 1 shows a typical AAR system. Several prototypes have demonstrated AAR applications in the fields of tourism [3], location-based communication and information sharing [4], [5], gaming and entertainment [6], and even as a novel technique for machine–human interaction [7].

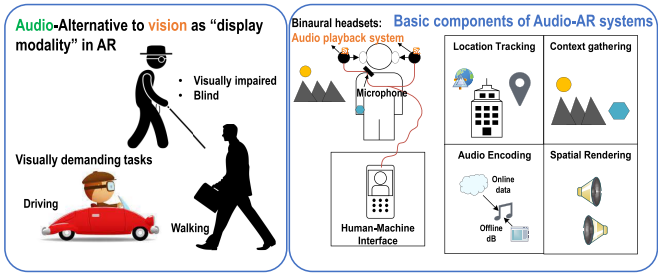


Fig. 1. Audio offers an alternative interactive modality to vision in AR systems targeting zero visual distractions, including blind individuals or for those involved in driving or walking that demand continuous visual attention. Building blocks of AAR system: i) audio playback system, ii) location tracking, iii) context gathering, iv) audio encoding, v) spatial rendering, and vi) human–machine interaction interface.

A binaural headset with a microphone acts as an audio playback setup for simultaneous perception of real and virtual sounds [8]. Sensors, present in the headset systems, capture and relay the information, regarding the environment, user posture, etc. in the form of speech signals, to the cloud servers or mobile handsets, which in turn send back interactive augmented auditory data to the headsets [9]. To function effectively, these sensor nodes must operate at higher data rates to support audio, both for capturing and receiving information, but with low power and latency to ensure a longer lifetime and facilitate real-time applications. This work primarily focuses on reducing the power requirement of these sensors, which could find utilization in some of the future AAR systems, especially the ones in which the sensor communicates with a machine RX/HUB and can tolerate lower SNR. Note that RX/HUB can use or itself be the server present on the cloud. Even the mobile handsets (when used as RX/HUB), equipped

Manuscript received July 28, 2021; revised November 5, 2021 and January 14, 2022; accepted February 15, 2022. This article was approved by Associate Editor Nick van Helleputte. This work was supported by the National Science Foundation (NSF) Career Award under Grant ECCS1944602. *(Corresponding author: Shreyas Sen.)*

The authors are with the Elmore Family School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907 USA (e-mail: gauravk@purdue.edu; bchatte@purdue.edu; shreyas@purdue.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/JSSC.2022.3155366.

Digital Object Identifier 10.1109/JSSC.2022.3155366

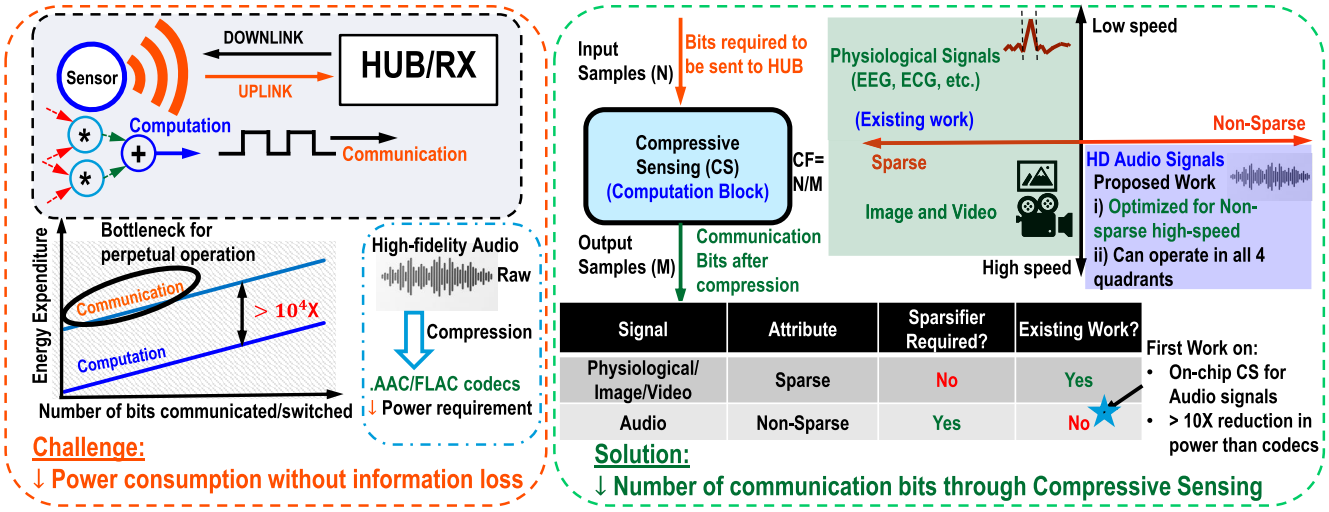


Fig. 2. Motivation and contributions of our work; the challenge of reducing power consumption of sensor nodes traces back to communication power. The codec-based compression techniques require a huge power budget and fail to work efficiently in energy-constrained nodes. Existing works show the applications of CS in sparse signals like the image or physiological signals, but an SoC exhibiting the compression of non-sparse audio signals remains unexplored. This work explores the applicability of CS to audio and proposes a low-power alternative to audio codecs for these nodes.

with the power of high storage batteries and wireless charging systems, can be assumed to be less energy-constrained, compared with sensor nodes.

Analyzing the power profile of a sensor node, it is observed that the communication power, associated with transmitting the data, is orders of magnitude higher than the sensing and computation power in a node [10]–[12], which is shown in Fig. 2. This motivates the reduction in the number of communicating bits, i.e., the audio compression at the node (headsets), without incurring any loss in information, to bring down the overall power consumption of the sensor node. As the RX/HUB accommodates a sufficient power budget, the downlink audio to the node, containing the interactive feedback, can be communicated in better quality audio to ensure enhanced user perception. As TX nodes operate in energy-constrained scenarios, the uplink audio streams (node to RX/HUB) should be compressed to reduce the communication burden. Note that RX/HUB, cloud servers, or mobile devices are usually well-equipped to work with low-quality compressed audio.

Established audio codecs, such as AAC and MP3, employed in mobile phones, broadcasting systems, perform high-fidelity compression of audio streams to reduce the number of communication bits while maintaining acceptable audio quality. Unfortunately, these codecs require high power, which is usually in the order of milliwatts [13], [14], which hinders their applications in auditory sensor nodes. Complex codecs such as Sony's LDAC [15] and AAC also suffer from significant latency in the range of 30-60 ms [16], thereby restricting their use in real-time applications that demand low latency.

Compressive sensing (CS) [17], [18], a well-known signal processing technique, used for processing physiological [19]–[23], such as EEG and ECG, image [24] and video [25], [26] signals, can present a promising low-power alternative to audio codecs. CS uses the sparseness of the input signal in a particular domain to reduce the number of transmitting samples. For reconstruction, the CS requires fewer samples than Nyquist sampling and uses mathematical tools such as L1-norm minimization to recover the original signal. Previous CS-based sensors have focused entirely on "sparse" physiological, image, or video signals. CS exploits the low information rate of bio-signals (signals of interest occurring infrequently) to acquire them at a rate proportional to information rate rather than the frequency content of the signal, reducing the number of bits transmitted, as shown in Fig. 2 (right).

For non-sparse signals with high information content such as audio, an additional pre-processing, i.e., sparsifier, is required for using CS. Although a few works have investigated the application of CS to audio signals, including heart sound acquisition [27], [28], speech [29], etc., and developed a theoretical framework [30], as far as our knowledge goes, a complete SoC demonstrating CS for audio is not explored in the literature yet. Preliminary version of this work is presented in IEEE CICC, 2021 [31]. In this version, we explored the relevance of the proposed compression scheme in real-time applications and benchmarked its performance with well-established conventional audio codecs, such as MP3. Furthermore, a comprehensive study is conducted to estimate the range of reconstructed audio quality across different audio types.

A. Contributions of Our Work

1) This work, CS-Audio, presents the first CS design, enabled with a discrete wavelet transform (DWT) sparsifier for catering to non-sparse signals such as high-definition audio.
2) CS-Audio achieves 3X–15X reduction in transmitted audio data with orders of magnitude lower power, when compared with traditional audio codecs [13].

3) This work incorporates a pipelined architecture implemented in a wake-up mode to achieve the highest operating speed (∼>200X) and lowest energy efficiency (∼>2X) than previously reported works that are based on the principle of CS. Note that these previous works target physiological applications that usually function at lower data rates.
4) This work benchmarks the performance of the proposed CS-Audio with conventional codecs, such as MP3 in terms of achievable compression, power requirements, and latency. The proposed compression scheme achieves (∼>40X) reduction in power with a comparable audio decoding time and quality.
5) The proposed architecture, though optimized for non-sparse high-speed operations, is generic enough to function with both non-sparse and sparse signals. Thus, we can, as shown in Fig. 2, operate across different speed and sparsity specifications.

B. Organization of Paper

The remainder of the paper is organized as follows. Section II explains the theory and mathematical background of the CS. In Section III, the different design choices are studied, including the metrics, sparsifier, sensing matrix generator, spare recovery algorithm, etc. for our architecture. Section IV presents the proposed architecture for CS-Audio, illustrating the need for pipelining and wake-up mode implementation of the design. Section V provides useful insights on the application of CS-based audio compression on different types of audio signals. Section VI shares the measurement results of ASIC and compares the proposed design with the related works reported in the literature. In Section VII, we benchmark the proposed CS-based compression with the existing audio codecs and discuss the role of CS-Audio in Audio AR systems. Section VIII concludes the paper.

II. Compressive Sensing

This section discusses the basic overview of the principles of CS, highlighting the key concepts of signal sparsity, incoherent sampling, and reconstruction and their relevance toward applying CS to audio signals.

A. Theory and Background of CS

CS leverages the signal structure, either to sample signals, such as video [25] at sub-Nyquist frequencies while countering the effects of aliasing and facilitates effective compression during sampling or reconstructs signals such as EEG and ECG [20], with fewer samples than Nyquist sampling. Mathematically, CS is a linear transformation of a set of vectors (signal representation) from one basis to another using sensing or measurement matrix ϕ. To achieve compression, the signal or vector must satisfy two fundamental criteria, namely, signal sparsity and incoherent sampling [32]. CS allows for the reconstruction of N samples of a sparse signal X by transmitting only M samples of compressed data, Y, where $M < N$, achieving a compression, quantified by compression factor (CF), defined as the ratio of N/M. This reduces the communication overload of the sensor and hence reduces the power consumption. To recover N samples from only M samples of data, L1-norm minimization is used to solve the convex optimization problem. The reconstruction step, being computation-intensive, requires high power and is usually carried out in RX/HUB, which is less energy-constrained. Previous works have explored the use of CS in biophysical applications such as EEG and ECG [20]–[22] and biomedical imaging applications such as magnetic resonance imaging (MRI) [33], [34], ultrasound [35], and computed tomography (CT) [36]. CS is used to compress analog images before digitization in image sensors [37] and also perform video compression [25].

CS-based compression can be broken down into two broad categories, which include signal compression comprising signal conditioning (sparsification) followed by CS encoding and signal reconstruction step that recovers back the original signal.

1) Signal Compression:

a) Signal acquisition/sparsification: The first step for CS involves a sparsifier that performs linear transformation of signal to another basis ψ, where the signal is sparse, such as FFT, DWT, DCT, or DST. Equation (1) depicts the matrix manipulations involved in the sparsifying operation to convert the signal from non-sparse (X) to sparse representations (f). ψ is a unit matrix for inherently sparse signals, which indicates that we can skip this step for these signals:

$$[f]_{N\times 1} = \psi_{N\times N}.[X]_{N\times 1}. \tag{1}$$

b) CS Encoding: The next step involves compressing the sparse signal (f) of length N into M number of output measurements (Y), thereby exhibiting a compression of N/M. CS encoding is a matrix multiplication operation of the sparse input signal (f) with a sensing or measurement matrix (ϕ) to obtain compressed output measurements (Y) as shown in (2). These compressed samples are transmitted to the RX, where they are reconstructed to get the original signal (X):

$$[Y]_{M\times 1} = \phi_{M\times N}.[f]_{N\times 1}. \tag{2}$$

2) Signal Reconstruction: Reconstructing the original signal (X) is a two-step process, the first being recovering the sparse signal (f), followed by desparsification. Sparse recovery is carried out by solving a convex optimization problem using L1-norm minimization. The sensing matrix, used during CS encoding, is provided as input to the optimizer. Desparsification is then done, using the sparse basis, to get the original signal (X). The following equations show the underlying mathematical expressions for sparse recovery and desparsification, respectively:

$$\min_{f\in\Re^N} = ||f||_{L1}, \quad \text{subject to } Y = \phi f \tag{3}$$

$$[X]_{N\times 1} = \psi^{-1}_{N\times N} \cdot [f]_{N\times 1}. \tag{4}$$

B. Prerequisites

1) Signal Sparsity: The signal of interest must exhibit sparsity in some basis or domain, i.e., when transformed to

that basis, the majority of the constituent elements in the signal are zero. For an instance, a sinusoidal signal requires an infinite number of samples to represent it in the time domain, while in the Fourier or frequency domain, a single frequency can represent the input sinusoidal. Thus, a sinusoidal signal is sparse in frequency or Fourier basis. Fortunately, many biophysical, image, and video signals are sparse in the time, spatial, or temporal domains, making them suitable for data compression using CS. For non-sparse signals, such as audio, a pre-processing unit in the form of a sparsifier, represented by sparse basis ψ, is required to perform sparsification to comply with the requirements of CS. Note that signal sparsity is directly related to compressibility in CS. Consequently, sparse signals such as ECG and APs undergo a higher degree of compression compared with non-sparse signals, such as audio.

2) Incoherent Sampling: Besides sparsity, CS also depends on the degree of coherence μ between the sensing matrix ϕ and sparse basis ψ. For reliable reconstruction at the receiver, the rows of the sensing matrix are required to be "uncorrelated" with columns of the sparse basis

$$\mu(\phi, \psi) = \max_{k,j} |\langle \phi_k, \psi_j \rangle|. \tag{5}$$

The lesser the coherence μ, given by (5) [38], the fewer samples are required to recover the original signal. Gaussian or Bernoulli distributions are commonly used to generate sensing matrices ϕ, with elements having least correlation with that of sparse basis ψ. The minimum number of measurements (M) required to recover the original sparse signal [22] is given by

$$M \geq C.\mu^2(\phi, \psi) \cdot S \cdot \log N \tag{6}$$

where C is the empirical constant in the range of 2–2.5 [39], N is the number of samples of the signal considered for compression, and S determines the sparsity of the signal, i.e., the number of non-zero elements out of N samples or the measure of information contained in the signal. Equation (6) shows that the higher the information content (S), the more measurement samples are required to recover the original signal.

III. Design Considerations

This section provides valuable insights into the different architectural explorations that have been performed to realize an optimized design. The performance metrics used for evaluation, the used sparsifier, recovery techniques, and sensing matrix generator are discussed while providing details on the optimal operating conditions for applying CS to audio signals.

The results demonstrated in this section are calculated by averaging the values of the evaluation metrics, obtained for a set of 50 randomly picked audio signals from [40] and [41], which comprises environmental, music, and speech audios, thereby generalizing to most of the conditions.

A. Metrics for Audio Performance Evaluation

To encapsulate all the aspects while comparing the audio signal, signal-to-noise distortion ratio (SNDR) and perceptual evaluation of audio quality mean opinion score (PEAQ MOS) are chosen to gauge the effectiveness of audio compression.

SNDR is defined as the ratio of the sum of the magnitude of amplitudes in the original audio signal ($s(n)$) to the sum of the absolute difference in amplitudes of the original and reconstructed audio signal ($r(n)$) for every sample, signifying the correlation of original and reconstructed signals in the time domain

$$\text{SNDR} = \frac{|s(k)|^2}{|s(k) - r(k)|^2}.$$

PEAQ MOS, developed in 1994–1998 by International Telecommunication Union's Radiocommunication Sector (ITU-R BS.1387), is a standardized algorithm that objectively measures the perceived audio quality [42]. It uses software [43], [44] that throws out a numerical value in the range of 1 (Bad) to 5 (Excellent) by integrating the multiple model output variables, obtained by simulating the perceptual properties of the human ear. Audio inferencing operated in intelligent assistants is more tolerant to input quality, functioning effectively when the PEAQ MOS for characterizing perceived audio quality exceeds 1.5, as stated in [45]. Also, in the targeted application, where the focus lies on compressing the uplink audio (from wireless headphones to cloud servers/mobile handsets), the quality of reconstructed audio (PEAQ MOS) is not a stringent requirement. With the development of robust neural-network-based systems, with inherent error tolerance, and software-based audio processing schemes, these servers can function effectively even with moderate quality audio. Thus, PEAQ MOS of 1.5, a fair representation of audio (according to ITU-R standards), is sufficient for successfully decoding the upstream audio signals communicated to the servers/mobile devices. As seen in Fig. 3, the PEAQ MOS correlates with the SNDR of the reconstructed audio, degrading with increased compression levels.

Due to its ability to incorporate the perceivable characteristics of the audio signal, MOS is assigned a higher priority than SNDR. SNDR, on the other hand, compares the time-domain signal amplitudes of the original and reconstructed signals, which is not a sufficient parameter to gauge the performance of signals, such as audio.

B. Choice of Sparsifier (DWT Versus DCT)

The choice of sparsification technique plays a key role in faithful signal reconstruction at the RX and also dictates the degree of compression at the encoder end. There are typically two types of compression techniques, based on their effect on the original signal, namely, lossy or lossless compression. Lossless compression focuses on the reconstructed signal quality while reducing the size of the audio signal to be transmitted. Lossy compression compromises the signal quality for the degree of compression, reducing perceptual redundancy, which includes the frequency that cannot be perceived by the human ear. Thus, lossy compression imposes irreversible loss to the original audio signal. Conventional audio codecs such as .MP3 and .AAC constitute lossy compression while .FLAC, .WAV, etc. are lossless. These audio codecs are fundamentally built on two of the compression algorithms, i.e., discrete cosine transform (DCT) and discrete wavelet transform (DWT). Although DCT offers better sparsity than

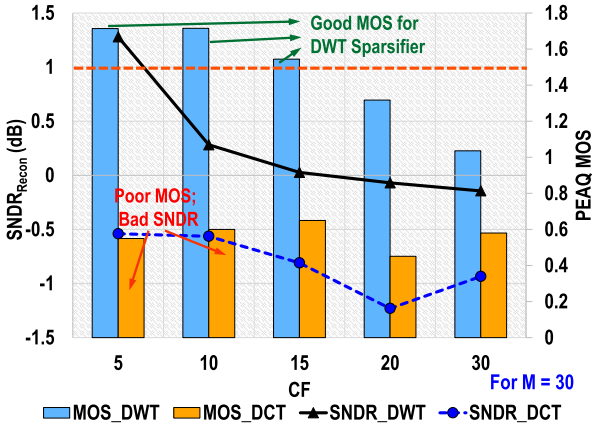


Fig. 3. Choice of sparsifier: Simulation results show the impact of DWT and DCT based sparsifier on the performance of CS. DWT outperforms DCT in terms of reconstructed audio quality at the RX.

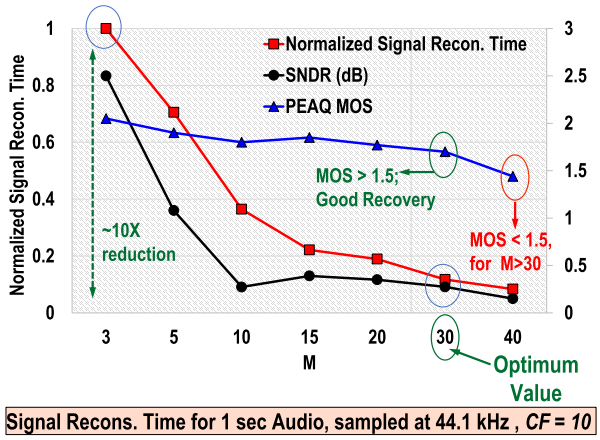


Fig. 4. Simulation results justify the choice of number of measurements (M). For $M = 30$, the design offers superior fidelity of reconstructed audio signal, while lowering the signal reconstruction time (latency).

DWT for an audio signal, yet DCT, being a lossy compression, fails to maintain the signal integrity, thereby exhibiting poor signal reconstruction. Fig. 3 presents the quantitative analysis comparing the performance of DCT- and DWT-based sparsification for audio compression. To determine the effectiveness of the used sparsifier, CS-based audio compression is performed with both DWT and DCT sparsifiers for different audio compression levels, quantified by compression factor (CF) and fixed number of output compressed samples, M(=30). CF denotes the number of input samples (N) that are compressed into M linear measurements at the output, thereby achieving a compression of N/M. The results show that the SNDR and PEAQ MOS for the reconstructed audio signal lie within acceptable limits for a DWT-based sparsification (up to CF of 15) when compared with DCT that fails to maintain audio quality post-reconstruction. Thus, DWT is chosen as an audio sparsifier over DCT. Note that the range of SNDR values is consistent with the previously reported works in audio [27], [29].

C. Choice of No. Of Measurements (M)

M denotes the number of linear measurements/combinations into which the input audio samples are compressed. Equations (5) and (6) show that bigger M should yield better signal reconstruction at the RX/HUB even with fewer samples. However, several other factors, such as incoherence between the sensing matrix and sparse basis, and sparsity of the signals, influence the quality of reconstruction. The CS encoder accumulates N number of samples into M number of measurements, achieving compression of CF (=N/M) compressing in windows of size of $M \times$ CF samples. For a constant CF, a bigger M increases the compression window, which thereby increases the signal information that we are accumulating, as shown in Fig. 4. Consequently, the performance of the decoder degrades to reconstruct the signal effectively for large compression windows, reducing the quality of output audio. On the other hand, we see that having a smaller

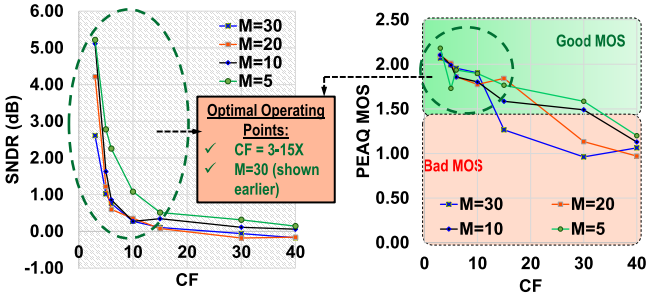


Fig. 5. Simulation results show the variation in SNDR and PEAQ MOS for different CF, indicating the optimal operating CF for audio signals. PEAQ MOS ≥1.5 and SNDR ≥0.2 are required conditions for acceptable quality of reconstructed audio.

compression window recovers the original signal faithfully. Moreover, increasing the value of M also increases the power and area overhead of the design, making it a crucial design consideration. Fig. 4 show that for higher M, the quality of the reconstructed signal degrades, impacting the SNDR and PEAQ MOS of the recovered signal. On the contrary, reducing M increases the time needed for reconstructing the signal, which is detrimental to real-time applications such as speech processing due to the latency involved with signal recovery. In this work, the value of M is chosen to be 30, considering the trade-off between latency and output audio quality. Fig. 4 depicts that for $M = 30$, the perceptual quality of output audio signal is higher than the acceptable level of MOS, i.e., ≥1.5, improving the normalized signal reconstruction time by ∼10X when compared with that for $M = 3$.

D. Choice of Optimal Compression Factor

The degree of compression or operating CF is dependent on the sparsity of the input signal. As described earlier, CF is the ratio of the number of input samples (N) that are being compressed to the number of linear measurements (M) into which they are accumulated. i.e., CF = N/M. Simulations are performed to find the optimal CF for performing CS-Audio, as depicted in Fig. 5. For non-sparse signal-like audio, increasing

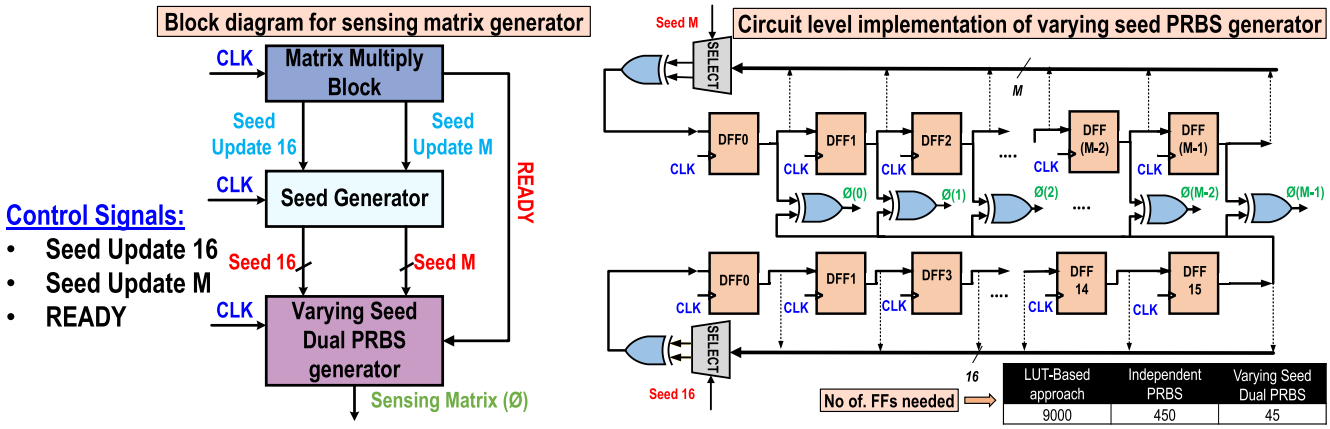


Fig. 6. Control flow and circuit implementation of varying seed dual PRBS sensing matrix generator. The control signals synchronize the operation of matrix multiply block, seed generator, and sensing matrix generator.

the CF implies integrating higher signal content in the compression window, thereby degrading the output audio quality, which is observed in SNDR and PEAQ MOS. For sparser signals, such as physiological signals, as the signal content is low for a given compression window, the achievable degree of compression is higher. Fig. 5 throws light on the optimal CF for the audio signal. It is found that for the reconstructed signal with PEAQ MOS of higher than 1.5, which is considered acceptable audio quality, the CF that can be applied for audio signals, without perceivable quality degradation, lies is in the range of 3–15X for chosen $M = 30$.

E. Choice of Sensing Matrix Generator

The random matrix generator plays a limiting factor in the power and area overhead of the CS encoder. One straightforward technique to realize a random matrix is to implement a memory or lookup table (LUT) to hold the predefined entries that can be referenced at the time of computation. This method occupies a huge area with high power consumption, as these memory elements (flops) need to be operated at Nyquist frequency. To obtain a compression of 10X with an M of 30 requires accumulation of 300 input samples (N), thereby demanding 9000 entries in the LUT/memory, occupying considerably higher area than other constituent elements of CS encoder. Another approach proposed by [46] uses an independent PRBS generator for each of M output measurements. This method provides a significant improvement over the previous memory implementation, requiring only 450 flip-flops (15×30) for a 2^{15} PRBS sequence for each measurement. Although the area of the matrix generator is reduced in [46], the PRBS generators and clocking network contribute to the majority of power consumption in the CS encoder. In this work, a varying seed dual PRBS-based random matrix generator [22] is used that produces the required random Bernoulli matrix by XOR-ing the output of one PRBS generator with every state of the second PRBS generator, as shown in Fig. 6. For every new input sample, a new seed is chosen for the two PRBS, generating Bernoulli matrix entries that are needed for CS encoding. Without considering additional overhead for seed update and generation, this implementation requires only 45 flip-flops for M of 30 when compared with 450 in the previous technique, thereby saving at least 10X in terms of power and area. Fig. 6 shows the circuit implementation of the sensing matrix generator, along with the control signals that synchronize the operation of the CS encoder, seed generator, and matrix generator.

F. Choice of Sparse Recovery Algorithm

Minimization of L1-norm is preferred over L0-norm and L-2 norm for several reasons [17]. Although L0-norm provides the exact solution of the original sparse signal but being a non-convex NP-hard optimization problem, it fails to produce a numerical solution [32]. The L2-norm minimization is less computation-intensive than L0-norm, but fails to perform sparse recovery, with the energy distributed over all the signal elements as opposed to the L1 norm, where they are consolidated into fewer samples [32].

Basis pursuit (BP) [47] effectively solves L1-norm optimization and returns an accurate reconstructed signal. Several greedy algorithms [48], such as orthogonal matching pursuit (OMP) [39], [49] and compressive sampling matching pursuit (CoSaMP) [50], [51], provide quicker outputs than the L1-norm BP method, by making locally optimal decisions, but fail to generate a globally optimum solution. Fig. 7 shows the comparison of different sparse recovery algorithms, i.e., BP [52], OMP [53], and CoSaMP [54], in terms of PEAQ MOS, SNDR of reconstructed signal, and normalized reconstruction time (correlated with latency of operation). Although BP works slowest as far as the latency is concerned, for non-sparse signals such as audio, BP works best in terms of the quality of the reconstructed signal (MOS and SNDR). For real-time applications such as voice assistants or audio inferencing, the latency involved in decoding the audio plays a decisive role. Thus, CoSaMP provides a potential alternative to BP, as it has low latency decoding with acceptable audio quality (SNDR and MOS). *In this work, we select CoSaMP as the decoding algorithm in RX, keeping in mind the Audio AR applications and possible machine inferencing.*

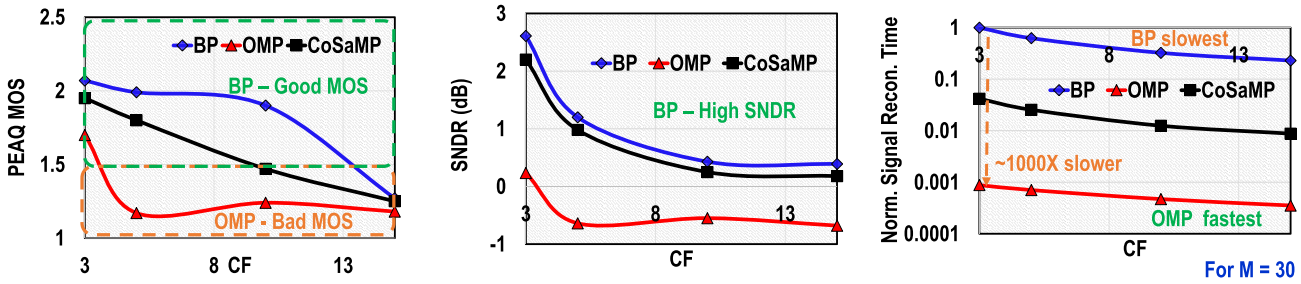


Fig. 7. Performance evaluation of different sparse recovery algorithms in terms of PEAQ MOS, SNDR, and normalized signal reconstruction time. BP outperforms OMP and CoSaMP with regard to reconstructed signal quality, while consuming longer time for signal reconstruction.

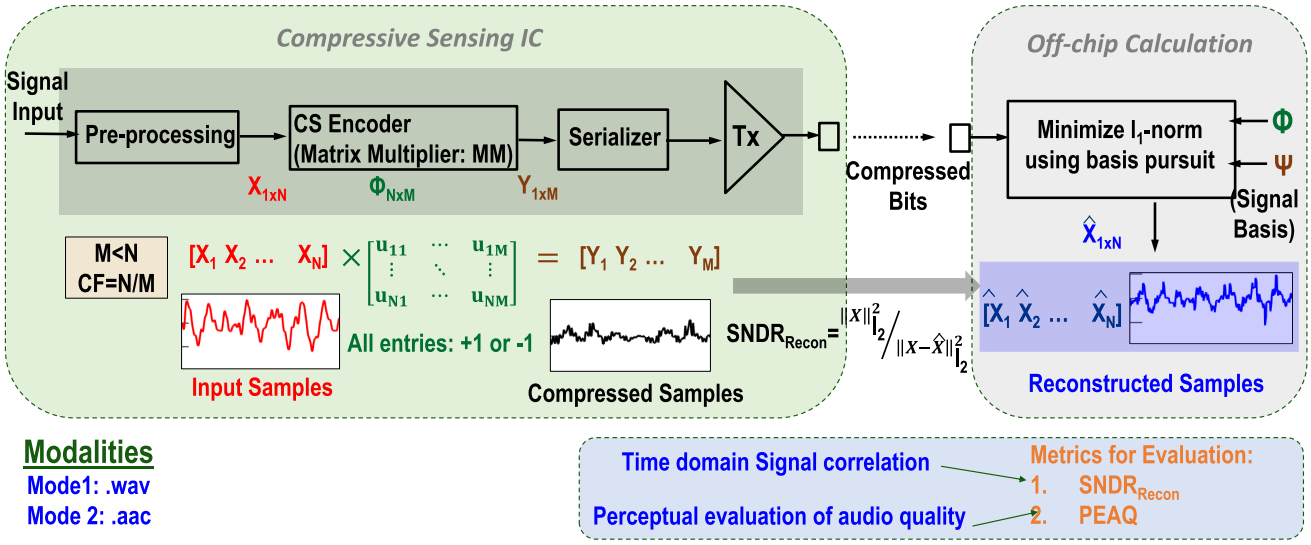


Fig. 8. Block level overview of the proposed architecture, showing the implemented CS-Audio with modes of operation, along with OFF-chip computations and performance metrics.

G. Operating Modalities

The design can operate in two modalities based on the type of audio that is provided to it. The default modality Mode 1 acts on raw audio data in the form of.WAV, etc., while Mode 2 uses the pre-processed compressed high-quality audio data in.AAC, etc. formats. Mode 1 provides a new alternative solution for low-power audio compression when compared with conventional codec-based audio compression. Mode 2 offers additional compression to the codec-based compressed audio signal, further reducing the number of transmitted bits and the power associated with their communication, but it suffers from higher computation power and degraded reconstructed signal quality, resulting from multiple compression techniques that operate on the source audio signal.

IV. Proposed Architecture for CS-Audio

To benefit from the lower power consumption of digital CS implementation over analog CS [22] and reap the advantages of process portability [55], we have considered a fully digital architecture in this work. The CS-Audio comprises the pre-processing unit, the CS encoder, and a serializer, followed by a transmit buffer. Fig. 8 shows the block-level overview of the proposed architecture. The CS-Audio takes in N input samples and compresses them into M linear combinations, using on-chip DWT sparsifier and the sensing matrix ϕ, realizing a matrix multiplication, i.e., CS Encoding. Thus, instead of N input samples, M compressed samples are transmitted, thereby reducing the communication power of the sensor by approximately N/M, which consequently lowers the overall sensor power. The hardware is designed to operate with audio in both raw formats (such as.WAV—in Mode 1) and compressed format (such as.AAC—in Mode 2). Finally, to ensure operation at high speeds, the entire hardware is realized as a pipelined architecture. Convex optimization problem, given by (3), is solved OFF-chip in MATLAB to recover the signal from compressed audio samples.

A. Sensor/TX

1) Pre-Processing Units: The input signal is digitized to 8 bits and is fed serially to the CS IC. The pre-processing units consist of a deserializer, the signal conditioner that uses a DWT-based sparsifier to sparse the audio signal, and the bit padding block that pads zeros to the input signal in case of a sparse input signal (when the sparsifier is bypassed).

The deserializer unit deserializes the serial input audio bits into parallel data samples that are then sent to the subsequent

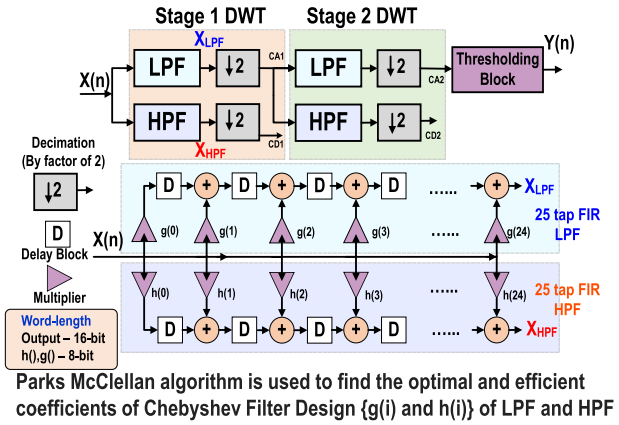


Fig. 9. Circuit implementation of a two-stage DWT.

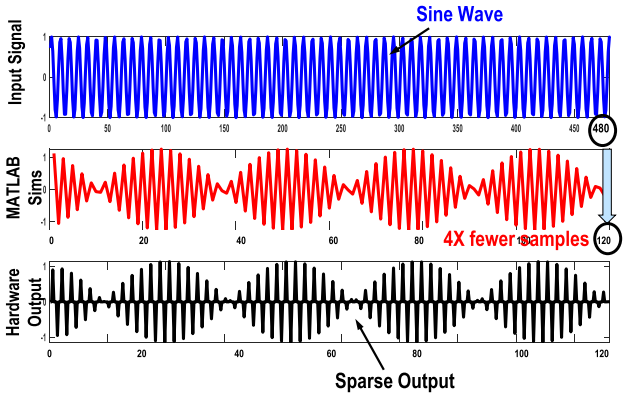


Fig. 10. Comparison of the post-synthesis sparsifier hardware results with MATLAB simulations for an input sinusoidal signal.

signal conditioner block. The signal conditioner performs sparsification of audio samples using two-stage DWT sparsifier, before feeding to the CS encoder for compression. Unlike Fourier transform, wavelet transform provides the time and frequency information of a given signal, which makes it a powerful tool to extract small changes or disturbances in the audio data. DWT is realized by passing the signal through low-pass filter (LPF) and high-pass filter (HPF), followed by decimation by a factor of two to generate approximation coefficients (cA) and detail coefficients (cD) that capture the low- and high-frequency information respectively, as shown in Fig. 9. To prevent bit overflow, the word-length for the two-stage 25-tap DWT is chosen to be 16. Thresholding block is used to omit all the insignificant samples below a specific threshold, without impacting the final signal recovery. A 25-stage FIR-implementation of Chebyshev LP and HP filters are realized with the filter coefficients calculated from the Parks McClellan algorithm. Figs. 9 and 10 show the circuit-level implementation and the post-synthesized results of the two-stage DWT sparsifier, respectively. It is seen that the post-synthesis results match with the MATLAB simulations, validating the efficacy of our design. In Fig. 10, the results obtained from MATLAB simulations are plotted as a continuous-time signal, while the results from the post-synthesis hardware are plotted as

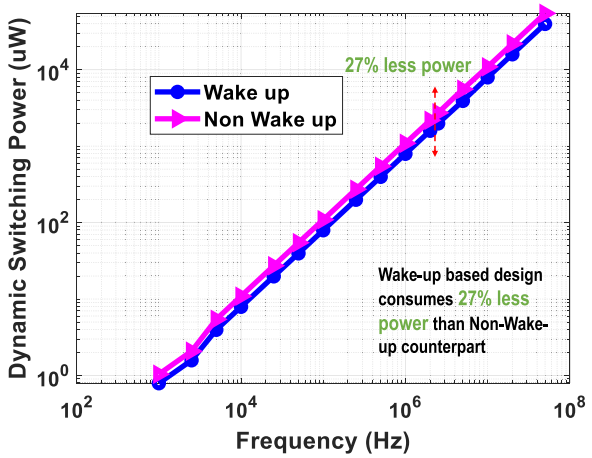


Fig. 11. Comparison of dynamic power for wake-up and nonwake-up CS encoder implementations.

a discrete time signal. The corresponding values of y-axis for both the cases are same. This is because a digitized sinusoidal signal is fed to the DWT accelerator, whereas a continuous time sinusoidal is used for MATLAB simulations. Also, due to sparsification, the number of non-zero samples required to represent the original signal reduces by four times.

There is also a provision to bypass the sparsifier using an external scan to operate sparse inputs like physiological signals, e.g., ECG. As sparsification produces a 16-bit output of 8-bit audio input, a bit padding block is incorporated to pad zeros to the sparse inputs for proper functioning of the CS encoder, which is designed to cater to 16-bit input.

2) CS Encoder: The output of the sparsifier, supplied to the CS encoder, includes the majority of elements with zero value. Thus, to reduce power consumption, the CS encoder, including the sensing matrix generator and matrix multiplier, is implemented in a wake-up mode, such that the block is power-gated when the input is zero, preserving the previously accumulated values in the registers. The BEGIN signal from the control and sequencing block, as shown in Fig. 12, indicates the state of the input sample. Fig. 11 provides the benefit in the dynamic switching power for wake-up- and nonwake-up-based implementations of CS encoder for different input frequencies. The power numbers show that wake-up-based implementation consumes at least 27% less power than the nonwake-up counterpart. Thus, wake-up implementation uses signal sparsity to skip few computation cycles to enable low-power operation.

Fig. 12 shows the implementation of the proposed wake-up CS encoder. Multiply and accumulate (MAC)-based CS encoder performs matrix multiplication and accumulation operations for N input samples to achieve compression, by a factor of N/M. For every non-zero input sample, indicated by BEGIN, the matrix multiplier accumulates the product of the input sample and the row elements of the sensing matrix ϕ in an M number of 16-bit accumulators. As shown in Fig. 4, the value of M is chosen to be 30, considering the trade-off between signal reconstruction time (latency) and output audio quality, to obtain area and power optimal design. The degree of compression can be set using a 3-bit external scan

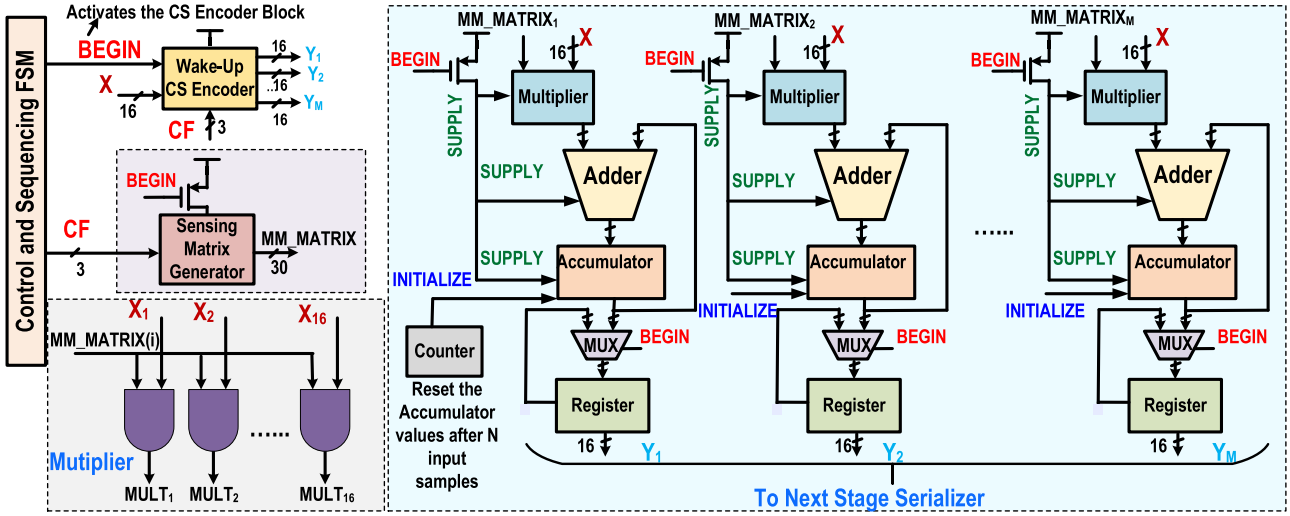


Fig. 12. MAC-based CS encoder implemented in wake-up configuration. BEGIN signal controls the operation of the block. INITIALIZE resets the accumulators to encode next N input samples. The multiplier block, realized using AND logic gates, reduces the power and area.

and can vary in the range of 3X–33.3X for both the modes (Mode 1 and 2). The accumulator size is chosen to be 16, trading off the area and power with buffer overflow. After N samples, the counter generates the INITIALIZE signal, which resets the accumulator for the next N samples, before serializing the accumulated values for transmitting the compressed audio samples.

For reliable reconstruction at the receiver, the rows of the sensing matrix are required to be "uncorrelated" with each other. A varying seed dual PRBS-based sensing matrix [22], consisting of two independent PRBS generators, produces the required random Bernoulli matrix. A seed update signal is sent to the seed generator to update the seed of the sensing matrix for every incoming input sample. Fig. 6 shows the circuit implementation of the matrix generator, along with the control signals that sequence the events. The seeds of the PRBS generators in the sensing matrix are re-seeded after N samples. As the elements of the sensing matrix are either 0 or 1, the matrix multipliers used in the CS encoder block are realized using logic AND gates (Fig. 12) to further reduce the power and area of the design.

3) Serializer: The serializer after the CS encoder block performs the serialization of accumulated values, which are then transmitted using transmit buffer. To ensure proper synchronization of output data, each measurement is preceded by a header that marks the beginning of the data. The serializer takes in 30 output measurements, each of 16 bits, and serializes them into constituent bits.

B. RX/HUB

RX operations including the signal reconstruction and performance evaluation are performed OFF-chip in MATLAB. As explained in the previous section, for signal reconstruction, L1-norm minimization is done by compressive sampling matching pursuit (CoSaMP) [50], [51] algorithm using the received compressed bits, sensing matrix (ϕ) used for CS encoding, and signal basis (ψ). SNDR and PEAQ MOS are used for performance evaluation. The values of these metrics are used to tune the operating knobs of the CS-Audio, such as CF and operating speeds, to ensure reconstructed audio lies within acceptable quality limits. Fig. 13 shows the OFF-chip computation steps that are performed for signal recovery.

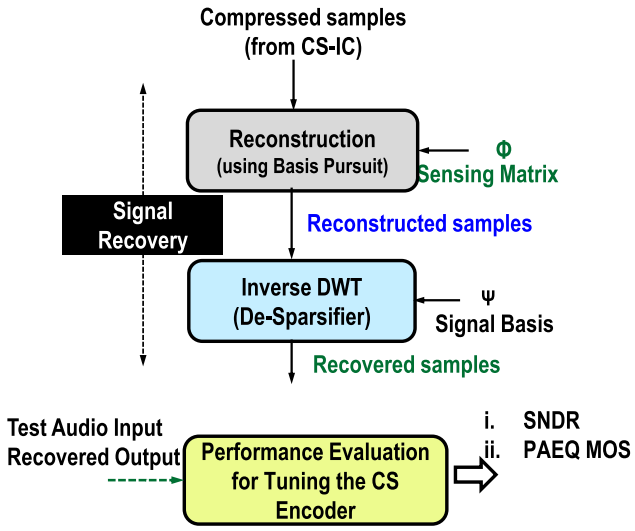


Fig. 13. OFF-chip computation flow for signal reconstruction.

V. Study on Different Types of Audio Signals

To quantify the impact of audio compression on different types of audio signals, we have performed a comprehensive study on CS-Audio with different audio datasets, ranging from speech, music, and environmental sounds, available from popular research databases. The study explores the max–avg–min values of SNDR for varying compressions (CF) and different input audio signals, bigger datasets than those

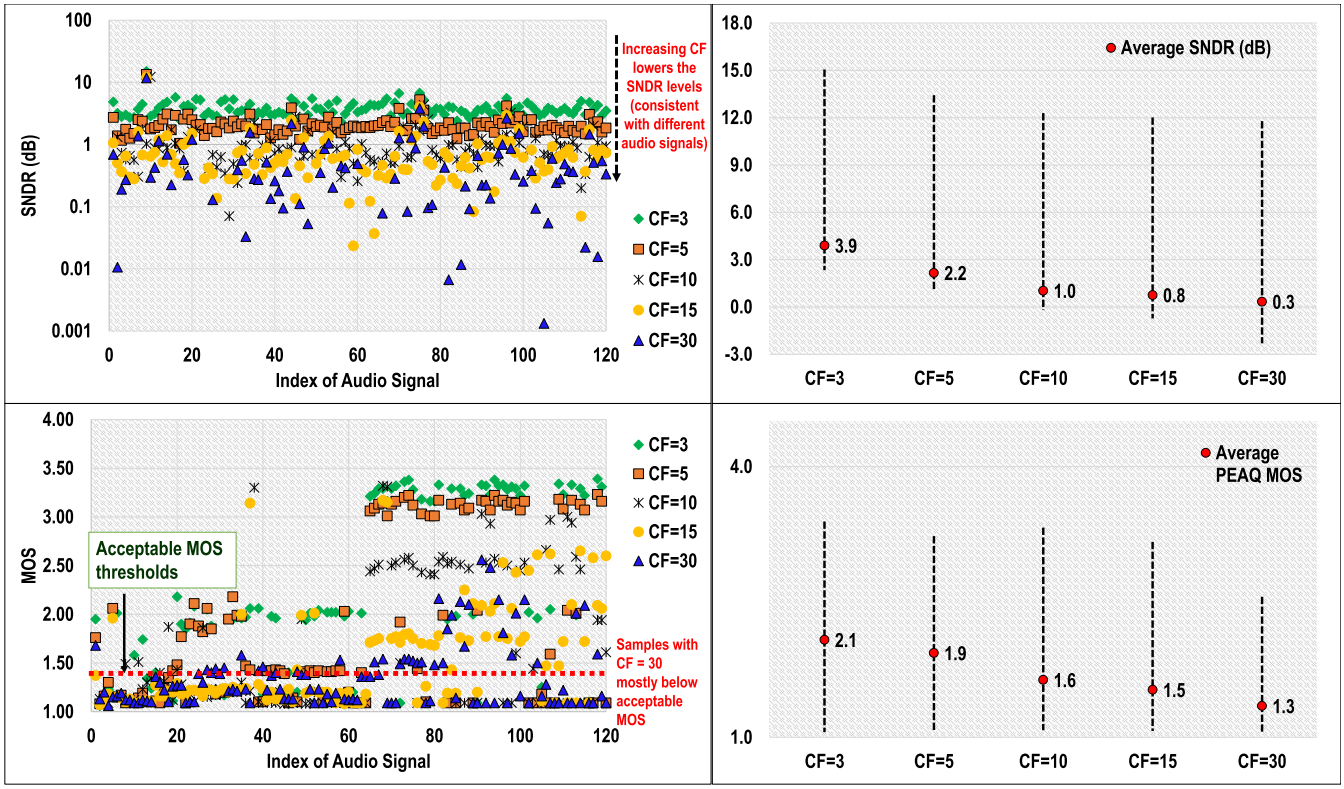


Fig. 14. SNDR and MOS values for different audio inputs for varying CF, along with max–avg–min curves to estimate/predict the range of achievable SNDR and MOS for a given CF. Results are taken for CoSaMP-based decoding scheme.

used for obtaining plots in Fig. 7, thereby providing richer insights on estimating the expected SNDR ranges based on the applied input and compression levels. Fig. 14 shows the SNDR and PEAQ MOS data points for different input audio signals (indexed from 1 to 100) for varying CF. As expected, with the increasing CF, the SNDR of the reconstructed audio degrades due to the accumulating more signal content in a fixed number of output measurements (M). Some input audio samples in Fig. 14 have SNDR values below 0.001 dB, implying a failed audio reconstruction at the receiver end. Note that these samples correspond to cases that are subjected to higher compression levels (CF $=$ 30), reiterating the fact that the optimal compression levels for audio signals lie in the range of 3X–15X, as discussed earlier in Section III-D. Fig. 14 also shows the max–avg–min values of achievable SNDR and PEAQ MOS for 100 different input audio samples taken from [56]. The plot helps in finding the maximum applicable CF for achieving a particular degree of fidelity at the receiver. The trade-off between energy savings due to compression and assimilated error in the output (lower SNDR) determines the applied CF. *Note that we can tune the parameters of the decoder for samples with MOS degrading below 1.5 at the cost of time needed for decoding the audio at RX.* Fig. 15 dives deeper into the SNDR and PEAQ MOS ranges for different types of audio inputs, such as speech and music. The plot shows the max–avg–min ranges while providing information on the percentage of audio inputs for specific SNDR and MOS conditions. We observe that the proposed CS-Audio works best for speech signals with $>$80% of audio samples exhibiting SNDR $>$0.5 for CF of 15, while that of music signals having $<$40% input audio samples with SNDR $>$0.5. The plots using PEAQ MOS provide similar inferences with around 60% audio samples $>$1.5 for both speech and environmental signals. Thus, depending on the input audio of interest, the CF can also be tuned in the CS encoder to ensure the SNDR of reconstructed audio remains within acceptable limits. Although we see around 50%–60% audio samples satisfy the acceptable range for PEAQ MOS in Fig. 15, the CoSaMP decoder parameters, such as maximum iterations and sparsity thresholds, can be increased to achieve better quality. Finally, advanced software-based audio post-processing techniques can be used in the RX to further bolster the audio quality. The environmental sounds, music, and speech audio datasets are reused from [40] and [41], respectively.

VI. Measurement Results

To validate the functionality of the CS encoder and to demonstrate the operation of audio compression, CS-Audio is fabricated in TSMC 65-nm CMOS technology. Fig. 16 shows the die micrograph and chip specifications of the fabricated IC. The design occupies 400 μm $*$ 320 μm area.

A. Measurement Setup and IO Waveforms

Fig. 17 shows the measurement setup for testing CS-Audio. The audio samples are digitized in MATLAB and fed to the chip through arbitrary waveform generator (AWG), along with

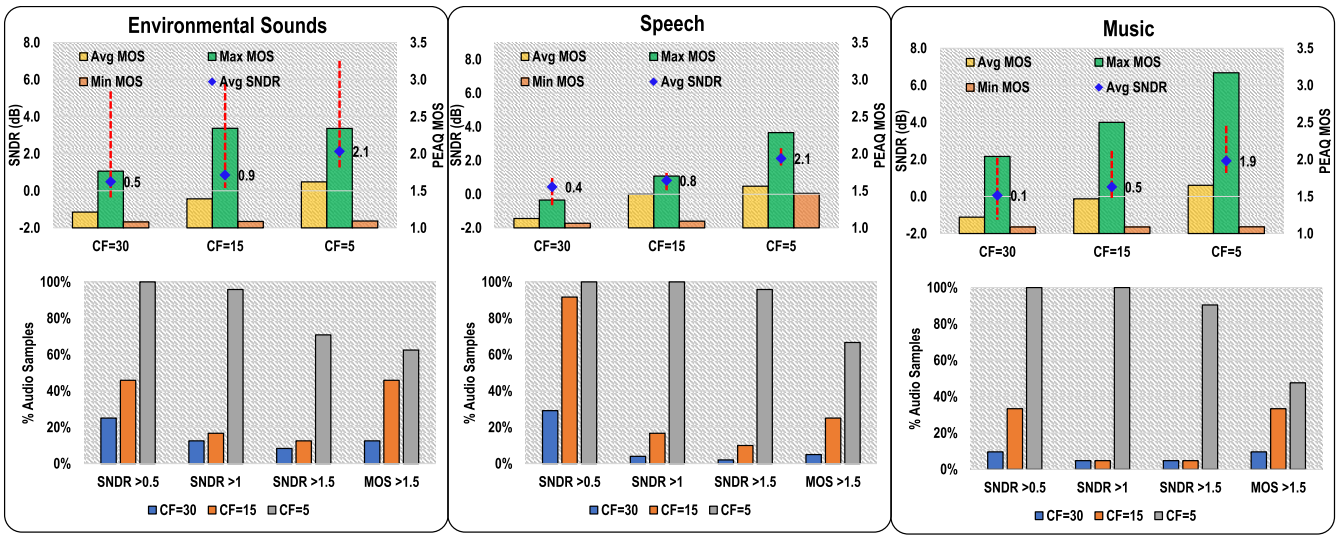
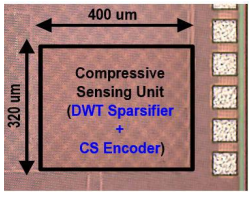


Fig. 15. Effect of audio compression on different audio signals, namely, environmental sounds, speech, and music. The analysis shows the max–avg–min curves for achievable SNDR and PEAQ MOS. This can be useful in predicting the range of reconstructed signal quality for different audio across varying CF.



Die Micrograph

Chip Specifications

Process	65nm CMOS LP
Supply	0.55V-1V
Active Area	0.13mm^2 (400 um X 320 um)
Performance	3X – 33.33X compression for 0.1-15 Mbps input
IO used	Four (4): CS VDD, GND, SERIAL_IN, SERIAL_OUT (Additional 4 IOs for Scan/Tests)
Programmable CF (3-bit Scan)	3X, 5X, 10X, 15X, 20X, 25X, 33.3X

Fig. 16. Die micrograph and chip specifications.

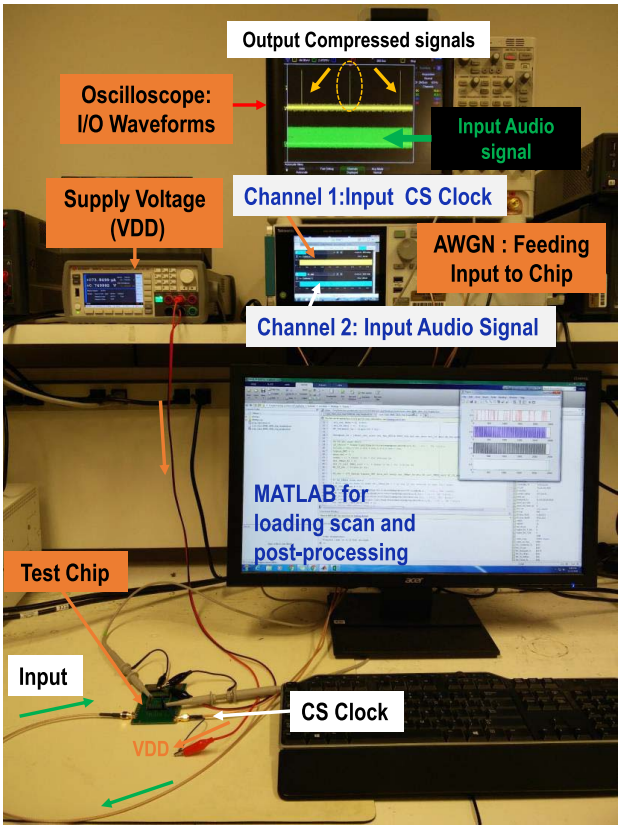


Fig. 17. Measurement setup.

the clock signal. The compressed output samples, observed in the oscilloscope, are captured and processed in MATLAB for signal reconstruction.

Fig. 18 shows the timing diagram of the input audio signal and output reconstructed signal for CF of 3. One second of an audio signal is sampled at 44.1 kHz and fed to the chip using AWG. The CS encoder performs the ON-chip sparsing and encoding to generate compressed samples, shown in red. The outputs of OFF-chip processing, i.e., BP algorithm for sparse recovery, shown in blue, and the final inverse sparsification using inverse DWT, shown in green, are also uploaded in the GitHub repository [57].

B. Variation of Power Consumed With Data Rate

This subsection provides insights on power consumption of CS-Audio for different supplies with varying CF.

Fig. 19(a) shows the variation in the consumed power with input data rate for different supplies. It is observed that consumed power scales with the input data rate and supply voltages, with the leakage power dominating the low-speed regime. Fig. 19(b) shows the variation in power consumption with different CF. It is found that the power remains almost independent of the selected CF because varying the CF changes only the number of input samples that are

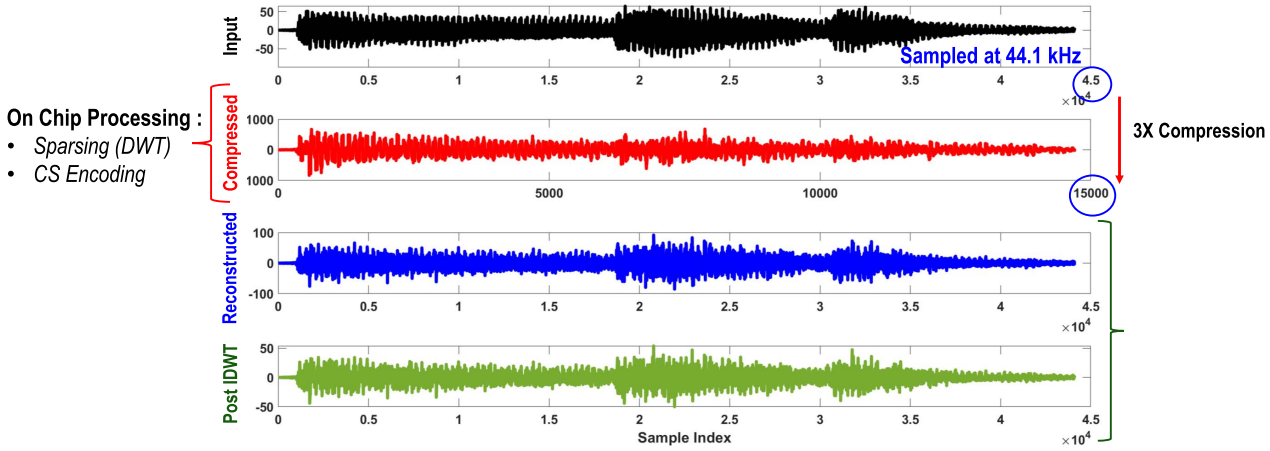


Fig. 18. Input–output waveforms of CS-Audio with 1-s audio signal, sampled at 44.1 kHz for compression of 3. The SparcLab GitHub repository [57] contains the test audio inputs and processed audio signals for varying degrees of audio compression.

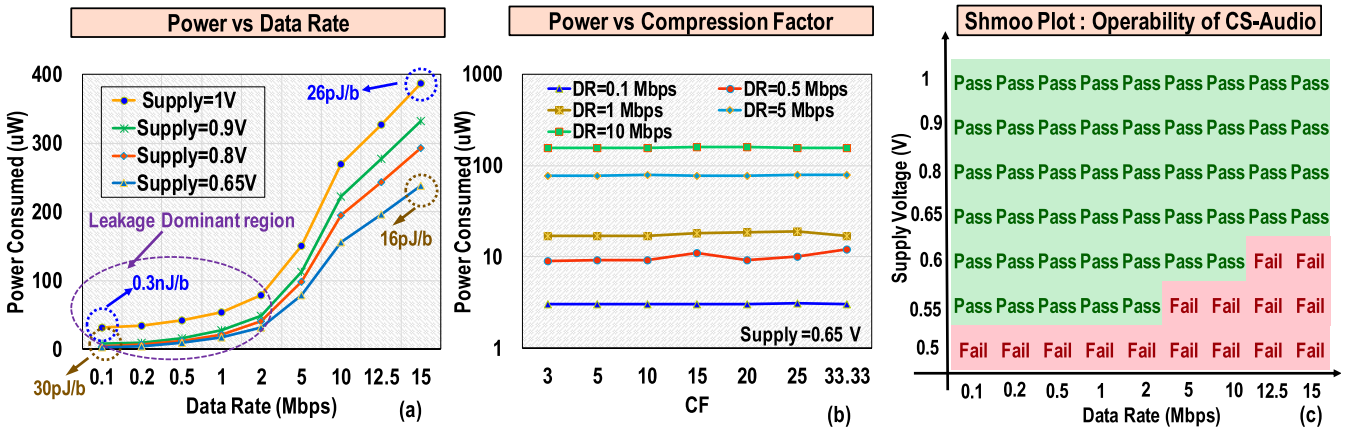


Fig. 19. Measurement Results. (a) Consumed power versus input data rate for fixed CF (= 5X) and supply voltage, (b) consumed power versus CF for different input data rates and fixed supply, and (c) shmoo plot showing the operability of CS-Audio for different data rates and supplies. It helps in resolving the optimal operating knobs for power-efficient operation of IC.

accumulated, the number of measurements(M) being constant. Thus, the number of bits switching during the computation remains unaffected for different CF, the consumed power being almost constant. Note that the results shown in Fig. 19(a) are taken for CF of 5X. The same trend is followed for any setting of CF as seen from Fig. 19(b).

C. Operability at Different Supply Voltages

The shmoo plot shown in Fig. 19(c) presents the operability of CS-Audio for different input data speeds. It indicates the functionality of IC at different supplies for a given input data rate. It is inferred that for an operating input of 15 Mbps, a supply voltage of at least 0.65 V is required. The same can function at lower supplies in case of slow inputs.

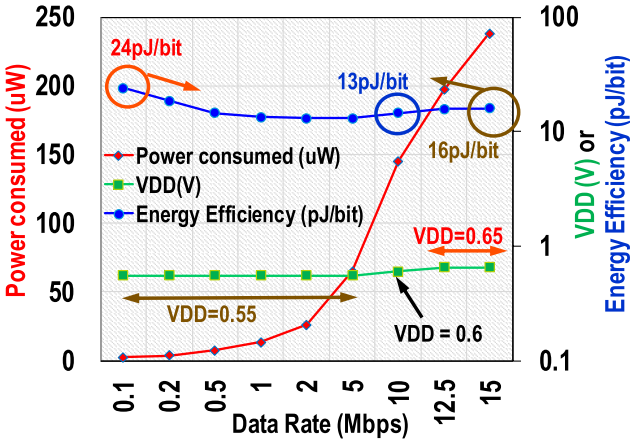


Fig. 20. Optimal operating points of CS-Audio.

D. Optimal Operating Points for CS-Audio

Fig. 20 gives intuitions on the optimal operating conditions for facilitating efficient functioning of CS-Audio. The operating points for achieving the most optimum energy efficiency, ranging from 24 to 16 pJ/bit for inputs varying from 0.1 to 15 Mbps, are shown. These points are generated by operating the CS-Audio at optimal supplies, obtained from Fig. 19(c).

	This Work	TBioCAS'16 [19]	CICC'15 [20]	JSSC'14 [21]	JSSC'12 [22]	Audio Codecs [13] [14]
Target Application	Audio	Physiological	Physiological	Physiological	Physiological	Audio
Sparsifier	2-stage DWT	No	No	No	No	-
Data Reduction	5-33.3X,16b	8X-16X	10X,8b	1-16,8b	10-40X, 8b	4X-22X
Speed	0.1-15Mbps	20kbps	48kbps	64 kbps (1kbps: 64 ch)	8-32kbps	-
Power	2.4-238µW	104µW	11.4µW	1.8 µW	1-2µW	~5mW : message tones ~7mW: Voice ~10 mW: HD Music
Energy/bit	24-16 pJ/b	520pJ/b	238pJ/b	28pJ/b	125-62 pJ/b	
Encoding Latency (ms)	15 @ 10Mbps for 1 sec Audio	-	-	-	-	60: AAC 30: LDAC
Normalized Decoding Latency	0.63 (CoSaMP)					1 (MP3 320 kbps)
Sensing Matrix Type	Dual PRBS with seed randomizer	Single PRBS	Dual PRBS	Single PRBS with seed randomizer	Dual PRBS with seed randomizer	
Clock Frequency	30 MHz	4 MHz	96 KHz	2 KHz	100 KHz	-
Supply Voltage	0.65V	-	-	0.9-1.2V	0.6V	-
Technology	65nm CMOS	180nm CMOS	65nm CMOS	130nm CMOS	90 nm CMOS	-

Fig. 21. Comparison with state-of-the-art CS works.

E. Comparison With Related Works

Fig. 21 tabulates the comparison of presented work with other existing state-of-the-art CS-based sensors. Due to the non-availability of CS works that target audio signals, we compare our proposed CS-Audio with previous works that focus on physiological signal sensing and also with audio codecs that are currently used to perform compression of audio. With the implementation of wake-up modes and pipelined architecture, the present work achieves the highest speeds of up to 15 Mbps with energy efficiency in the range of 24–16 pJ/bit. The power consumed by the design is >20X lower than that required by the conventional audio codecs. The algorithmic latency of the CS-based compression is around 15 ms, which is lower than that of AAC or LDAC audio codecs. Note that the algorithmic latency of the proposed technique is calculated from the time required (derived from the number of clock cycles) for processing 1 s of audio (44 100 samples) at a 10-Mbps input data rate.

F. GitHub Repository: Test Audio I/O Signals

The test input audio signals to the CS-Audio along with recovered audio signals are uploaded in the GitHub repository [57]. Due to the digitization of analog signals and consequent sparsing, encoding, and transmission of bits, the recovered signals exhibit some high-frequency noise. LPF-based denoising ensures removal of this high-frequency noising to provide final recovered audio. Note that additional software-based audio processing can be used to improve signal quality post-reconstruction, which can be part of future work. The SparcLab GitHub repository [57] includes the test audio inputs and processed audio signals for varying degrees of compression.

VII. Discussions

A. Benchmarking With Audio Codecs

1) Audio Compression Levels and Power Requirements: Conventional audio codecs, such as MP3, can offer a moderate-quality representation of raw format music, WAV (44.1 kHz, 16 bit, two channels), using 128 kbps, achieving compression of 11X and a high-definition audio at 320 kbps, compressing by 4.4X. AAC codec can produce MP3-equivalent quality with only 3/4 of the bits, so a 96-kbps AAC provides 14.7X compression with similar quality as 128 kbps MP3. For voice recordings or speech signals preserved in WAV format, bit rates of 64 kbps MP3 are sufficient for effective operation, providing an overall compression of 22X. Although these codecs achieve compression levels ranging from 4.4X to 22X, their encoding demands significant power, which is not desirable for sensor nodes, especially the ones applied in audio AR applications. This calls for a low-power alternative to audio codecs that can work efficiently in these nodes.

The measurement results show that standalone CS-Audio (Mode 1) achieved acceptable audio quality for compression of 3X–15X at orders of magnitude lower power than conventional audio codecs-based compression techniques and reduced latency involved in compression, as shown in Fig. 21. As discussed earlier in Section III-G, the proposed technique can also be combined with MP3 or other audio codecs (Mode 2) to enable additional compression of audio streams at the cost of reduced quality of reconstructed audio than audio codecs. This adds an additional operating modality, where Mode 1 can be used in cases where the energy of the sensor turns out to be a bottleneck while Mode 2 can find applications in scenarios where the nodes are not usually energy-limited.

Fig. 22(a) shows the comparison of size of bitrate file generated for 60 s of audio for different codecs and proposed CS for varying compression levels. It is observed that the proposed standalone CS-Audio offers comparable degrees of compression w.r.t. established codecs. Fig. 22(b) shows the comparison of different modalities of CS with MP3 (320 kbps). It is observed that Mode 1 offers low-power solution *(∼40X less power than MP3)* for energy-constrained scenarios, while Mode 2 provides a low overhead additional compression *(∼3X higher)* to conventional MP3.

2) Audio Decoding: The proposed CS-based scheme, the CS-Audio, uses a mathematical tool (CS) that exploits the sparseness of signals to compress signals. During decoding, an optimizer minimizes the L1-norm to recover the input signal. L1-norm is a convex optimization problem, which is computation-intensive and thus time-consuming. Several greedy algorithms, such as OMP ans CoSaMP, aim to reduce the decoding time by taking locally optimal solutions. On the other hand, MP3 decoders, which are focused and optimized for audio signals, are fast and usually applied in real-time applications.

A thorough analysis is done to measure the performance of CS decoders and conventional MP3 decoders. To estimate the time required for decoding MP3- and CS-based compressed audio, we have simulated MP3 [58] and CS [54] decoders in MATLAB in the same simulation environment for the same audio length. Multiple MP3 audios, with varying bitrates, are considered to emulate the different levels of compression. The results show that the CoSaMP decoder requires almost similar time for decoding when compared with the MP3 decoder across different compression levels, which is promising for

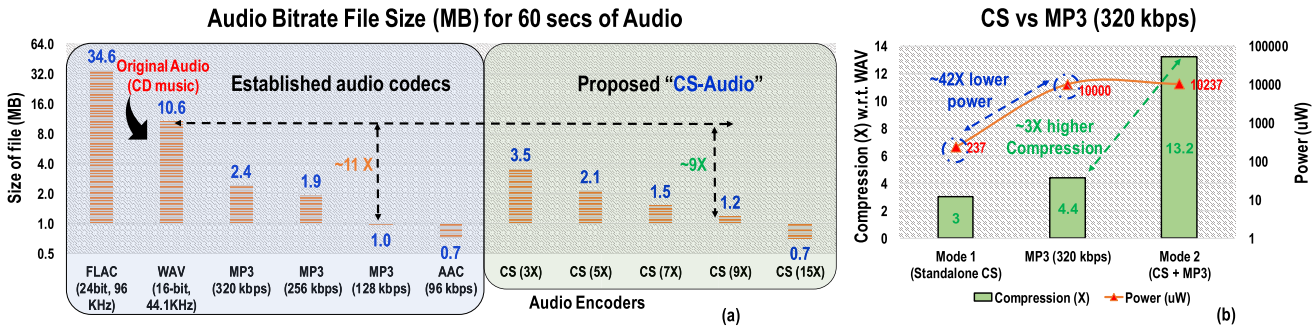


Fig. 22. Performance comparison of the proposed CS with established techniques, like MP3. (a) Audio bit rate file size (MB) for 60 s of audio with different audio encoders. (b) CS versus MP3 (320 kbps) across different modalities.

TABLE I
BP VERSUS COSAMP FOR TIME COMPLEXITY

Reconstruction Algorithm	Time Complexity O(n)
BP	O(n^3)
CoSaMP	O($m \times n$)

TABLE II
COSAMP VERSUS MP3 FOR TIME REQUIRED FOR DECODING

Normalized CPU Decoding Time			
CF	CoSaMP	MP3	MP3 Bitrate (CF)
3X	0.63	1	320 kbps (4.4X)
5X	0.38	0.72	256 kbps (5.5X)
7X	0.31	0.41	192 kbps (7.4X)
10X	0.19	0.16	128 kbps (11X)

applying CS to real-time applications. On the contrary, the BP-based decoding scheme demands exceedingly high decoding time, which can be observed from the time complexity comparison of BP and CoSaMP, as shown in Table I and Fig. 7, hindering its applicability to low latency AR. Table II provides an estimate of the normalized CPU decoding time needed for CoSaMP and MP3 decoders. Note that the CS decoders are generic in nature, whereas MP3 decoders are optimized to work with audio signals. Although MP3 decoding algorithms are deterministic in nature, few individual steps, such as Huffman decoding and re-quantizer, take a significant number of iterations due to multiple nested for-loops and bit manipulations [59]. Also, the MP3 encoders pack multiple overhead bits for CRC, and the exact time for decoding MP3 is pessimistic, given that no such overhead bits are introduced in this version of CS-Audio.

B. Role of CS-Audio in Audio AR

Audio AR uses sensor nodes that sense, compute, and communicate data in audio to the RX/HUB, which processes the received audio to provide appropriate feedback to augment the user's real-world acoustic space. These nodes, operating in an energy-starved regime, demand a low-power operation for longer lifetimes. As discussed earlier, the power expended in communicating the data is orders of magnitude higher than sensing or computation power. With the established audio codecs failing to provide low-power compression, the proposed CS-Audio presents a promising alternative for compressing the audio signals in these nodes before sending to RX/HUB, reducing the number of communication bits and lowering the power budget, which extends the lifetime of these nodes. Note that only the uplink audio compression (to RX) is required as RX/HUB has a sufficient power budget to send better quality audio feedback (downlink) to the nodes.

Although we have not provided any system-level demonstration for Audio AR, one can extend the developed concepts and techniques for these sensors that operate with audio, as follows. For a use-case, in a typical AAR system, the sensor node present in the headset systems captures the audio data from the user/environment (sensing), compresses the same by the proposed CS-based compression technique, the CS-Audio (computation), and transmits the compressed audio samples (communication) to the receiver, usually present on the cloud servers or mobile handsets. The RX, being less energy-constrained, can use advanced software-based audio shaping techniques, in addition to conventional L1-norm-based signal reconstruction, to ensure proper decoding of compressed audio. After gathering the relevant information, the RX sends back interactive audio that enhances the user's perception. Note that the feedback to the sensor nodes can work in high-definition audio without degrading the user perception, due to availability of sufficient power budget at RX.

VIII. CONCLUSION

In conclusion, a new CS-based audio compression technique, CS-Audio, has been demonstrated in 65-nm CMOS technology that offers a low-power alternative to conventional audio codecs. The CS encoder is realized in a pipelined fashion, implemented in a wake-up mode, and equipped with an on-chip DWT sparsifier, for the first time, to facilitate the compression of non-sparse audio signals. It achieves the highest input data rate for CS-based sensors (up to 15 Mbps) in the literature, while the wake-up operation enables low-power design (2.4–238 μW). The power overhead resulting from CS encoding (computation) is orders of magnitude lower than the power saved from communicating

the bits. Furthermore, benchmarking with established audio codecs shows that the proposed compression scheme achieves ($\sim$>40X) reduction in power with a comparable audio decoding time and quality. Finally, the presented architecture is a generic CS-based compression technique and can be extended to both sparse and non-sparse signals (all four quadrants in Fig. 2).

REFERENCES

[1] S. Sen, S. Maity, and D. Das, "The body is the network: To safeguard sensitive data, turn flesh and tissue into a secure wireless channel," *IEEE Spectr.*, vol. 57, no. 12, pp. 44–49, Dec. 2020.

[2] P. Tiefenbacher, N. H. Lehment, and G. Rigoll, "Augmented reality evaluation: A concept utilizing virtual reality," in *Virtual, Augmented and Mixed Reality. Designing and Developing Virtual and Augmented Environments* (Lecture Notes in Computer Science), R. Shumaker and S. Lackey, Eds. Cham, Switzerland: Springer, 2014, pp. 226–236.

[3] B. B. Bederson, "Audio augmented reality: A prototype automated tour guide," in *Proc. Conf. Companion Hum. Factors Comput. Syst. (CHI)*, Denver, CO, USA, 1995, pp. 210–211.

[4] K. Lyons, M. Gandy, and T. Starner, "Guided by voices: An audio augmented reality system," in *Proc. 6th Int. Conf. Auditory Display (ICAD)*, Apr. 2000.

[5] F. Z. Kaghat, A. Azough, M. Fakhour, and M. Meknassi, "A new audio augmented reality interaction and adaptation model for museum visits," *Comput. Electr. Eng.*, vol. 84, Jun. 2020, Art. no. 106606.

[6] T. Nilsen, S. Linton, and J. Looser, "Motivations for augmented reality gaming," in *Proc. NZGDC, Fuse*, vol. 4, Jun. 2004, pp. 86–93.

[7] A. N. Nagele *et al.*, "Interactive audio augmented reality in participatory performance," *Frontiers Virtual Reality*, vol. 1, pp. 1–14, 2021.

[8] H. Gamper, *Enabling Technologies for Audio Augmented Reality Systems*. Espoo, Finland: Aalto Univ., 2014.

[9] M. Mekni and A. Lemieux, "Augmented reality: Applications, challenges and future trends," *Appl. Comput. Appl. Comput. Sci.*, vol. 20, pp. 205–214, Apr. 2014.

[10] N. Cao, B. Chatterjee, M. Gong, M. Chang, S. Sen, and A. Raychowdhury, "A 65 nm image processing SoC supporting multiple DNN models and real-time computation-communication trade-off via actor-critical neuro-controller," in *Proc. IEEE Symp. VLSI Circuits*, Jun. 2020, pp. 1–2.

[11] B. Chatterjee, D. Das, S. Maity, and S. Sen, "RF-PUF: Enhancing IoT security through authentication of wireless nodes using *in-situ* machine learning," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 388–398, Jun. 2019.

[12] B. Chatterjee, N. Cao, A. Raychowdhury, and S. Sen, "Context-aware intelligence in resource-constrained IoT nodes: Opportunities and challenges," *IEEE Des. Test*, vol. 36, no. 2, pp. 7–40, Apr. 2019.

[13] X. Jiang *et al.*, "A low-power, high-fidelity stereo audio codec in 0.13 μm CMOS," *IEEE J. Solid-State Circuits*, vol. 47, no. 5, pp. 1221–1231, May 2012.

[14] L. Cacioli and D. Zaucha, "EETimes—Improving battery life with ultra low-power codecs," in *EETimes*, Nov. 2009. [Online]. Available: https://www.eetimes.com/improving-battery-life-with-ultra-low-power-codecs/

[15] (2021). *What is—LDAC?*. [Online]. Available: https://www.sony.com/electronics/support/articles/00146691

[16] (Jun. 2019). ValdikSS. *Audio Over Bluetooth: Most Detailed Information about Profiles, Codecs, and Devices*. [Online]. Available: https://habr.com/en/post/456182/

[17] E. J. Candes and M. B. Wakin, "An introduction to compressive sampling," *IEEE Signal Process. Mag.*, vol. 25, no. 2, pp. 21–30, Mar. 2008.

[18] E. J. Candès, "Compressive sampling," in *Proc. Int. Congr. Mathematicians Madrid*, May 2007, pp. 1433–1452.

[19] X. Liu *et al.*, "A fully integrated wireless compressed sensing neural signal acquisition system for chronic recording and brain machine interface," *IEEE Trans. Biomed. Circuits Syst.*, vol. 10, no. 4, pp. 874–883, Aug. 2016.

[20] V. Behravan *et al.*, "A compressed-sensing sensor-on-chip incorporating statistics collection to improve reconstruction performance," in *Proc. IEEE Custom Integr. Circuits Conf. (CICC)*, Sep. 2015, pp. 1–4.

[21] D. Gangopadhyay, E. G. Allstot, A. M. R. Dixon, K. Natarajan, S. Gupta, and D. J. Allstot, "Compressed sensing analog front-end for bio-sensor applications," *IEEE J. Solid-State Circuits*, vol. 49, no. 2, pp. 426–438, Feb. 2014.

[22] F. Chen, A. P. Chandrakasan, and V. M. Stojanovic, "Design and analysis of a hardware-efficient compressed sensing architecture for data compression in wireless sensors," *IEEE J. Solid-State Circuits*, vol. 47, no. 3, pp. 744–756, Mar. 2012.

[23] B. Chatterjee *et al.*, "A 1.15 μW 5.54 mm^3 implant with a bidirectional neural sensor and stimulator SoC utilizing bi-phasic quasi-static brain communication achieving 6 kbps-10 Mbps uplink with compressive sensing and RO-PUF based collision avoidance," in *Proc. Symp. VLSI Circuits*, Jun. 2021, pp. 1–2.

[24] X. Yuan, Y. Liu, J. Suo, and Q. Dai, "Plug- and-play algorithms for large-scale snapshot compressive imaging," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 1447–1457.

[25] A. C. Sankaranarayanan, C. Studer, and R. G. Baraniuk, "CS-MUVI: Video compressive sensing for spatial-multiplexing cameras," in *Proc. IEEE Int. Conf. Comput. Photography (ICCP)*, Apr. 2012, pp. 1–10.

[26] J. Holloway, A. C. Sankaranarayanan, A. Veeraraghavan, and S. Tambe, "Flutter shutter video camera for compressive sensing of videos," in *Proc. IEEE Int. Conf. Comput. Photogr. (ICCP)*, Apr. 2012, pp. 1–9.

[27] S. Sun, J. Xing, Z. Zhou, W. Wang, and J. Chen, "Comparative study of compressed sensing for heart sound acquisition in wireless body sensor networks," *IEEE Access*, vol. 8, pp. 22483–22492, 2020.

[28] I. D. Irawati and E. M. Dewi, "Comparing on sparse heart sound recovery algorithms," in *Proc. Int. Seminar Intell. Technol. Appl. (ISITIA)*, Jul. 2016, pp. 111–116.

[29] A. Griffin and P. Tsakalides, "Compressed sensing of audio signals using multiple sensors," in *Proc. 16th Eur. Signal Process. Conf.*, Aug. 2008, pp. 1–5.

[30] S. Yu, R. Wang, W. Wan, L. Du, and X. Yu, "Compressed sensing in audio signals and it's reconstruction algorithm," in *Proc. Int. Conf. Audio, Lang. Image Process.*, Jul. 2012, pp. 947–952.

[31] K. Gaurav Kumar, B. Chatterjee, and S. Sen, "A 16 pJ/bit 0.1-15 Mbps compressive sensing IC with on-chip DWT sparsifier for audio signals," in *Proc. IEEE Custom Integr. Circuits Conf. (CICC)*, Apr. 2021, pp. 1–2.

[32] D. Craven, B. McGinley, L. Kilmartin, M. Glavin, and E. Jones, "Compressed sensing for bioelectric signals: A review," *IEEE J. Biomed. Health Inform.*, vol. 19, no. 2, pp. 529–540, Mar. 2015.

[33] J. C. Ye, "Compressed sensing MRI: A review from signal processing perspective," *BMC Biomed. Eng.*, vol. 1, no. 1, p. 8, Mar. 2019.

[34] S. G. Lingala and M. Jacob, "Blind compressive sensing dynamic MRI," *IEEE Trans. Med. Imag.*, vol. 32, no. 6, pp. 1132–1145, Jun. 2013.

[35] C. Quinsac, A. Basarab, J.-M. Girault, and D. Kouamé, "Compressed sensing of ultrasound images: Sampling of spatial and frequency domains," in *Proc. IEEE Workshop Signal Process. Syst.*, Oct. 2010, pp. 231–236.

[36] G.-H. Chen *et al.*, "Time-resolved interventional cardiac C-arm cone-beam CT: An application of the PICCS algorithm," *IEEE Trans. Med. Imag.*, vol. 31, no. 4, pp. 907–923, Apr. 2012.

[37] C. Park, W. Zhao, I. Park, N. Sun, and Y. Chae, "A 51-pJ/pixel 33.7-dB PSNR 4× compressive CMOS image sensor with column-parallel single-shot compressive sensing," *IEEE J. Solid-State Circuits*, vol. 56, no. 8, pp. 2503–2515, Aug. 2021.

[38] E. Candès and J. Romberg, "Sparsity and incoherence in compressive sampling," *Inverse Problems*, vol. 23, no. 3, pp. 969–985, Apr. 2007.

[39] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Trans. Inf. Theory*, vol. 53, no. 12, pp. 4655–4666, Dec. 2007.

[40] K. J. Piczak, "ESC: Dataset for environmental sound classification," in *Proc. 23rd ACM Int. Conf. Multimedia*, Oct. 2015, pp. 1015–1018.

[41] G. Tzanetakis. *Music Speech Database*. Accessed: Jun. 18, 2021. [Online]. Available: https://marsyas.info

[42] T. Thiede *et al.*, "PEAQ—The ITU standard for objective measurement of perceived audio quality," *J. Audio Eng. Soc.*, vol. 48, nos. 1–2, pp. 3–29, Feb. 2000.

[43] *P.862: Revised Annex A—Reference Implementations and Conformance Testing for ITU-T Recs P.862, P.862.1 and P.862.2*. Accessed: Jun. 6, 2021. [Online]. Available: https://www.itu.int/rec/T-REC-P.862-200511-I!Amd2/en

[44] A Prodeus. (Jun. 2021). *PESQ MATLAB Driver*. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/47333-pesq-MATLAB-% driver

[45] P. S. Rao and V. Sreelatha, "Implementation and evaluation of spectral subtraction with minimum statistics using WOLA and FFT modulated filter banks," M.S. thesis, Blekinge Inst. Technol., Karlskrona, Sweden.

[46] X. Chen, Z. Yu, S. Hoyos, B. M. Sadler, and J. Silva-Martinez, "A sub-nyquist rate sampling receiver exploiting compressive sensing," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 58, no. 3, pp. 507–520, Mar. 2011.

[47] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM J. Sci. Comput.*, vol. 20, no. 1, pp. 33–61, Dec. 1998.
[48] L. Du, R. Wang, W. Wan, X. Yu, and S. Yu, "Analysis on greedy reconstruction algorithms based on compressed sensing," in *Proc. Int. Conf. Audio, Lang. Image Process.*, Jul. 2012, pp. 783–789.
[49] S. G. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Trans. Signal Process.*, vol. 41, no. 12, pp. 3397–3415, Dec. 1993.
[50] D. Needell and J. Tropp, "CoSaMP: Iterative signal recovery from incomplete and inaccurate samples," *Appl. Comput. Harmon. Anal.*, vol. 26, no. 3, pp. 301–321, May 2009.
[51] W. Dai and O. Milenkovic, "Subspace pursuit for compressive sensing signal reconstruction," 2008, *arXiv:0803.0811*.
[52] R. J. Mohammad. (Jun. 2021). *Basis Pursuit (BP)*. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/73846-basis-pursui% t
[53] S. Mohamed. (Jun. 2021). *Orthogonal Matching Pursuit Algorithm (OMP)*, [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/50584-orthogonal-m% atching-pursuit-algorithm-omp
[54] (Apr. 2021). Chenhaodev. *Compressive Sensing Matching Pursuit (CoSAMP)*. [Online]. Available: https://github.com/chenhaodev/MATLAB-simul/blob/7eaa965d3025a3f369602d0% 6f484d8166ca1eb06/uwb140617/cosamp.m
[55] G. K. K, B. Chatterjee, and S. Sen, "OpenSerDes: An open source process-portable all-digital serial link," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Feb. 2021, pp. 386–391.
[56] V. Panayotov. *LibriSpeech ASR Corpus*. Accessed: Jun. 6, 2021. [Online]. Available: https://openslr.org
[57] (Jun. 2021). SparcLab. *SparcLab/CS-Audio*. [Online]. Available: https://github.com/SparcLab/CS-Audio
[58] A. Hassan. (Nov. 2021). *Complete Implementation of a MP3 Decoder*. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/27303-complete-imp% lementation-of-a-mp3-decode.
[59] J. Thiagarajan and A. Spanias, *Analysis of the MPEG-1 Layer III (MP3) Algorithm Using MATLAB*. San Rafael, CA, USA: Morgan & Claypool, 2011.

Gaurav Kumar K (Graduate Student Member, IEEE) received the bachelor's degree in electronics and telecommunication engineering from Jadavpur University, Kolkata, India, in 2017. He is currently pursuing the Ph.D. degree in electrical and computer engineering with Purdue University, West Lafayette, IN, USA, working with Prof. S. Sen.

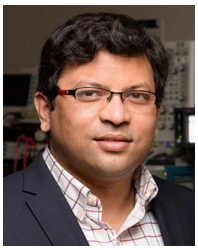
He worked as a Network Experience manager with the Telecom Industry, Bharti Airtel, India. He has interned with the Advanced R&D Discovery Team, Texas Instruments (TI) Kilby Labs, USA, during the summer of 2021, where he worked on designing RX front-end circuits for high-speed electrical links. His research interests include mixed-signal IC design for low-power and high-speed sensors and wireline links.

Mr. Kumar has served as a Primary and Secondary Reviewer for multiple reputed conferences, including IEEE International Conference on VLSI Design (VLSID) and IEEE Custom Integrated Circuit Conference (CICC).

Baibhab Chatterjee (Graduate Student Member, IEEE) received the B.Tech. degree in electronics and communication engineering from the National Institute of Technology (NIT), Durgapur, India, in 2011, and the M.Tech. degree in electrical engineering from IIT Bombay, Mumbai, India, in 2015. He is currently pursuing the Ph.D. degree with the School of Electrical Engineering, Purdue University, West Lafayette, IN, USA.

His industry experience includes 2 years as a Digital Design Engineer/a Senior Digital Design Engineer with Intel, Bengaluru, India, and 1 year as a Research and Development Engineer with Tejas Networks, Bengaluru. His research interests include low-power analog, RF, and mixed-signal circuit design for secure biomedical applications.

Mr. Chatterjee received the University Gold Medal from NIT, Durgapur, India, in 2011, the Institute Silver Medal from IIT Bombay in 2015, the Andrews Fellowship at Purdue University from 2017 to 2019, the HOST 2018 Best Student Poster Award (3rd), the CICC 2019 Best Paper Award (overall), the RFIC/IMS 2020 3MT Award (audience choice), and the Bilsland Fellowship at Purdue University from 2021 to 2022.



Shreyas Sen (Senior Member, IEEE) received the Ph.D. degree in ECE from Georgia Tech, Atlanta, GA, USA, in 2011.

He has over 5 years of industry research experience in Intel Labs, Hillsboro, OR, USA, Qualcomm, Austin, TX, USA, and Rambus, Los Altos, CA, USA. He is currently an Elmore Associate Professor of ECE & BME, Purdue University, West Lafayette, IN, USA. He is the Inventor of the Electro-Quasistatic Human Body Communication (EQS-HBC), or Body as a Wire Technology, for which he was a recipient of the MIT Technology Review top-ten Indian Inventor Worldwide under 35 (MIT TR35 India) Award. His work has been covered by 250+ news releases worldwide, invited appearance on TEDx Indianapolis, Indian National Television CNBC TV18 Young Turks Program, NPR subsidiary Lakeshore Public Radio, and the CyberWire podcast. He serves as the Director for the Center for Internet of Bodies (C-IoB). He has authored/coauthored three book chapters, over 175 journal and conference papers, and has 15 patents granted/pending. His current research interests span mixed-signal circuits/systems and electromagnetics for the Internet of Things (IoT), biomedical, and security.

Dr. Sen is a recipient of the NSF CAREER Award in 2020, the AFOSR Young Investigator Award in 2016, the NSF CISE CRII Award in 2017, the Intel Outstanding Researcher Award in 2020, the Google Faculty Research Award in 2017, the Purdue CoE Early Career Research Award in 2021, the Intel Labs Quality Award in 2012 for industry-wide impact on USB-C type, the Intel Ph.D. Fellowship in 2010, the IEEE Microwave Fellowship in 2008, the GSRC Margarida Jacome Best Research Award in 2007, and nine best paper awards, including IEEE CICC in 2019, 2021, and in IEEE HOST 2017–2020, for four consecutive years. His work was chosen as one of the top-ten papers in the hardware security field (TopPicks 2019). He serves/has served as an Associate Editor for IEEE SOLID STATE CIRCUITS LETTERS (SSC-L), *Frontiers in Electronics*, IEEE Design & Test, an Executive Committee Member of the IEEE Central Indiana Section and a Technical Program Committee Member of the ACM Design Automation Conference (DAC), the IEEE Custom Integrated Circuit Conference (CICC), Design, Automation and Test in Europe (DATE), the ACM/IEEE International Symposium on Low Power Electronics and Design (ISLPED), the International Conference on Computer-Aided Design (ICCAD), the International Test Conference (ITC), VLSI Design, among others.