A Real-Time Speech Enhancement Processor for Hearing Aids in 28-nm CMOS

Sungjin Park, *Member, IEEE*, Sunwoo Lee, *Graduate Student Member, IEEE*, Jeongwoo Park, *Member, IEEE*, Hyeong-Seok Choi, Kyogu Lee, *Senior Member, IEEE*, and Dongsuk Jeon, *Member, IEEE*

***Abstract*— Speech enhancement (SE) plays a key role in many audio-related applications by removing noise and enhancing the quality of human voice. Recent deep learning-based approaches provide high-quality SE, but real-time processing of those algorithms is challenging in resource-constrained devices due to high computational complexity. In this article, we present an energy-efficient real-time SE processor aimed at hearing aids. To implement high-quality SE with a very limited power budget, various algorithm and hardware optimization techniques are proposed. Our SE algorithm adaptively allocates computational resources to each region in the input feature domain depending on their importance, reducing overall computations by 29.7%. Along with 4-bit channel-wise logarithmic quantization, the processor adopts a reconfigurable multiplier-less processing element (PE) that supports both pre-/post-processing and neural network layers, resulting in a 21.5% area reduction. In addition, the design employs efficient scheduling and input buffering schemes to reduce on-chip memory access by 70.8%. Fabricated in a 28-nm CMOS process, our design consumes only 740 μW at 2.5 MHz with a total latency of 39.96 ms, satisfying the real-time processing constraints. In addition, our approach demonstrated higher SE quality than prior art in both objective and subjective evaluations.**

***Index Terms*— Digital hearing aids, multiplier-less processing element (PE), neural network processor, reconfigurable architecture, speech enhancement (SE).**

I. Introduction

SPEECH enhancement (SE) is an algorithm to eliminate noise and obtain a clean voice from noisy audio inputs, improving perceptual quality and intelligibility [1]. This algorithm is widely used in real-world applications such as video conferences, telephones, and hearing aids, for the purpose of making communication easier. Furthermore, SE can be employed as a pre-processing scheme to improve reliability in other audio-related tasks, such as keyword spotting [2] and speech recognition [3].

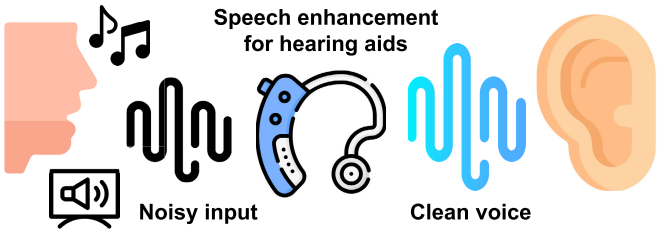


Fig. 1. SE for hearing aids.

Over the past few decades, many studies have focused on improving the performance of SE algorithms. Deep learning-based methods have recently demonstrated superior performance compared to traditional signal processing approaches [4], [5]. The prevalent approach involves supervised learning, which employs a mixture of clean speech and background noise as the training set and clean speech as the target. In most recent models, the raw input data in the time domain is converted into features in the time–frequency domain and then further processed by a neural network to achieve high SE quality [6], [7], [8], [9], [10], [11], [12].

Hearing aids remove noise from external noisy utterances to deliver a clean voice to the wearer (Fig. 1). Since they are worn inside or behind the ear, the device should support real-time processing while consuming very low power as it must last for days with a highly limited energy budget. For example, most hearing aids are powered by zinc-air batteries [13], which have a capacity of 230–285 mAh (for zinc-air A13). To support 12–16 h of use per day for 10 days, the IC must maintain a current consumption below 1.8 mA. Additionally, other rechargeable options, such as nickel metal hydride and silver–zinc batteries, which have small capacities of 30 and 32–35 mAh, respectively [14], also require similarly low current consumption for one day of use. The energy consumption is dictated by the size of the neural network model, and hence it is critical to minimize the model when it comes to adopting SE in hearing aids. In addition, the complexity of the operations required within the model should also be lowered for more efficient hardware implementation.

Recent hardware-oriented works mitigated these problems by co-optimizing algorithms and hardware. For example, the design in [15] employs a 327-kB on-chip SRAM and consumes 2.17 mW power while running an SE algorithm based on a convolution neural network (CNN) in real time.

Received 15 May 2024; revised 13 August 2024; accepted 5 September 2024. This article was approved by Associate Editor Ben Keller. This work was supported in part by the National Research Foundation of Korea under Grant NRF-2022R1C1C1006880 and Grant NRF-2021M3F3A2A01037633, and in part by the Institute of Information and Communications Technology Planning and Evaluation under Grant IITP-2021-0-01343 and Grant IITP-2023-RS-2023-00256081. *(Corresponding author: Dongsuk Jeon.)*

Sungjin Park, Sunwoo Lee, and Dongsuk Jeon are with the Department of Intelligence and Information, the Research Institute for Convergence Science, and the Inter-University Semiconductor Research Center, Seoul National University, Seoul 08826, South Korea (e-mail: djeon1@snu.ac.kr).

Jeongwoo Park is with the Department of Electrical and Computer Engineering, Sungkyunkwan University, Suwon 16419, South Korea.

Hyeong-Seok Choi is with the ElevenLabs, New York, NY 10016 USA.

Kyogu Lee is with the Department of Intelligence and Information and the Interdisciplinary Program in Artificial Intelligence, Seoul National University, Seoul 08826, South Korea.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/JSSC.2024.3460426.

Digital Object Identifier 10.1109/JSSC.2024.3460426

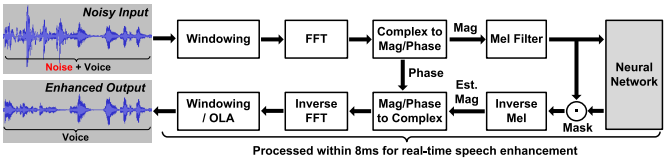


Fig. 2. Proposed SE pipeline.

Fedorov et al. [16] present an FPGA-based SE processor that implements an SE algorithm using a recurrent neural network (RNN). However, both designs exhibit noticeably lower SE performance compared to state-of-the-art models. In addition, their model sizes exceed 100 kB, which limits practical usage in hearing aid devices with severe energy constraints.

In this work, we present an end-to-end SE processor that meets real-time processing constraints with ultralow power consumption and a small footprint while delivering high-quality SE, rendering it an attractive solution for hearing aids. Our contributions can be summarized as follows.

1) An importance-aware neural network optimization method that significantly reduces computational costs while maintaining enhancement quality.
2) A reconfigurable multiplier-less processing element (PE) that efficiently supports pre-/post-processing and neural network layers.
3) Efficient scheduling and input buffering schemes to improve energy efficiency by reducing on-chip memory access.
4) An energy-efficient real-time SE processor with only 740 μW power consumption that was verified using both quantitative and qualitative evaluations, demonstrating superior SE quality suitable for real-world applications.

The remainder of this article provides further elaboration on our design and offers detailed explanations of the points discussed in our conference paper [17]. In Section II, we introduce the entire SE pipeline and the proposed algorithm optimization techniques, along with various ablation studies. Section III presents the reconfigurable PE and the overall architecture of the SE processor with on-chip memory access reduction techniques. Section IV presents qualitative and quantitative evaluations of the design, followed by conclusions in Section V.

II. Lightweight Importance-Aware SE Algorithm

A. Overall SE Pipeline

Recent SE algorithms commonly employ a mask to remove noise from the input audio [6], [7], [8], [9], [10], [11], [12]. In those algorithms, a neural network takes an input audio or its extracted features and generates a mask as an output. Then, the input is multiplied by the mask to eliminate the noise component. This approach demonstrated superior performance compared to training the model in a way that it generates a clean speech directly. Fig. 2 displays the SE pipeline we designed, based on a recent masking-based algorithm [6].

Our SE system utilizes a Mel-scale spectrogram generated using a Mel filter [18], contrasting with the baseline that directly employs the spectrogram as neural network input and in the masking stage. The Mel filter finely analyzes the low-frequency range containing crucial information for speech analysis, whereas it performs coarse analysis on the high-frequency region. By employing the Mel filter, we can significantly reduce the input dimension of the neural network, thereby decreasing the computational cost while maintaining good SE quality. The detailed operation of our SE pipeline, which processes input audio frames sequentially, is as follows. First, the input audio is segmented into 32-ms frames (512 points for 16 kHz sampling), and windowing is applied to mitigate spectral leakage. Subsequently, a fast Fourier transform (FFT) is applied to obtain the real and imaginary values of the frequency-axis components for the corresponding frame. With these real and imaginary values, magnitudes and phases are calculated. The magnitudes are transformed into Mel-magnitudes through the Mel filter, while the phases are temporarily saved until they are later used for reconstructing clean audio. The neural network takes the Mel-magnitudes as the input and generates an output mask, which is multiplied by the network input to generate the noise-suppressed Mel-magnitudes. These values are then transformed into the noise-suppressed magnitudes using the inverse Mel filter. The noise-suppressed magnitudes are multiplied by the cosine and sine values calculated from the pre-computed phases of the input frame to produce noise-suppressed real and imaginary components. Finally, performing windowing and overlap-and-add based on the hop length yields clean output audio.

The neural network model adopted in the baseline [6] consists of 1-D depthwise separable convolution layers, gated recurrent units (GRUs), and 1-D transposed depthwise separable convolution layers. An encoder composed of 1-D depthwise separable convolution layers performs downsampling of input features along the frequency axis. It is followed by a frequency-axis GRU (FGRU) that enlarges the receptive field along the frequency axis. Then, a time-axis GRU (TGRU) propagates information from the current frame to the next frame, thereby increasing the receptive field along the time axis. Lastly, a decoder composed of 1-D transposed depthwise separable convolution layers upsamples the output of TGRU. This structure uses 1-D convolution layers in the encoder and decoder, while TGRU integrates information from previous frames into the current frame. This approach enhances model performance by incorporating time-axis information into the neural network output with minimal computational complexity of processing multiple input frames in the encoding path. Furthermore, in the decoding path where the mask is generated, a 1-D transposed convolution layer is used instead of fully connected (FC) layers adopted in previous SE processors [15], [16], thereby keeping the number of parameters low. However, this approach processes the entire frequency range equally, although each sub-band may have a varying amount of information. Based on this observation, we propose a modified neural network structure described in Fig. 3. In our model, the separable convolution layers in the encoder and decoder are split into two separate groups based on the frequency regions of the input spectrogram. In addition, TGRU is replaced with a band-separated TGRU (BS-TGRU) consisting of four separate GRUs. These modifications allow

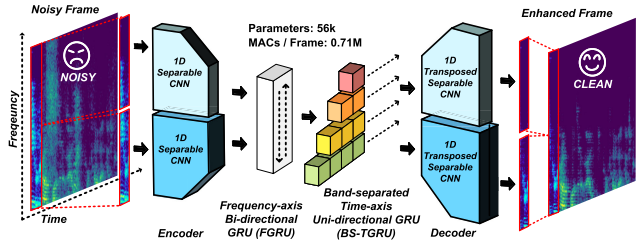


Fig. 3. Proposed neural network model for SE.

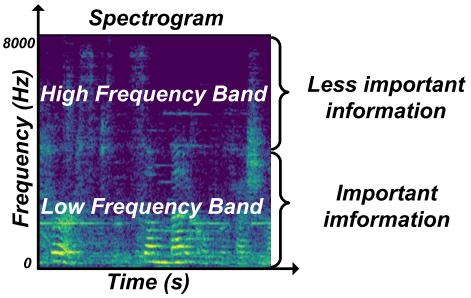


Fig. 4. Characteristics of each sub-band in the spectrogram.

for adopting the proposed importance-aware band optimization technique which reduces the model size and computational costs while maintaining high SE performance, which will be detailed in Section II-B.

B. Importance-Aware Band Optimization

The fundamental frequency of the human voice is typically found below 1 kHz. Furthermore, formant frequencies that provide important information for distinguishing pronunciation are found below 4 kHz [19]. In higher frequency sub-bands above 4 kHz, relatively less important components for intelligibility such as fricatives are distributed [20] (Fig. 4). Therefore, we optimized the neural network to analyze the lower frequency range more thoroughly, where the essential information of the speech is primarily contained. We allocate more resources to the low-frequency sub-bands and relatively fewer resources to the high-frequency sub-bands to reduce computational costs while maintaining SE performance.

Fig. 5 depicts the conventional TGRU [6] and our proposed BS-TGRU. In our model, BS-TGRU replaces the conventional TGRU, which uses a single GRU and accounts for 30.7% of the total network operations. While the conventional TGRU uses only one GRU for all bands, our BS-TGRU consists of four separate GRUs, each processing a different number of sub-bands to reflect the importance of the sub-bands. Fig. 6 illustrates the detailed operations of BS-TGRU. The lowest sub-band (32 channels) is composed of four frequency bins (green). This is processed by all four GRUs, and the outputs of each GRU are concatenated along the output channel axis, resulting in 88-channel GRU output for the lowest sub-band. Conversely, only one GRU is applied to the highest sub-band (red), producing an eight-channel output, representing the highest sub-band of the GRU output. In other words, BS-TGRU comprises four GRUs: one operating only on the most important sub-band (the lowest), and the remaining three

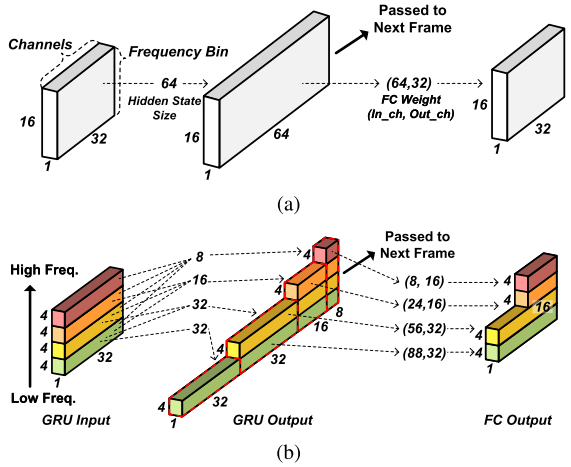


Fig. 5. (a) Conventional TGRU and (b) proposed BS-TGRU.

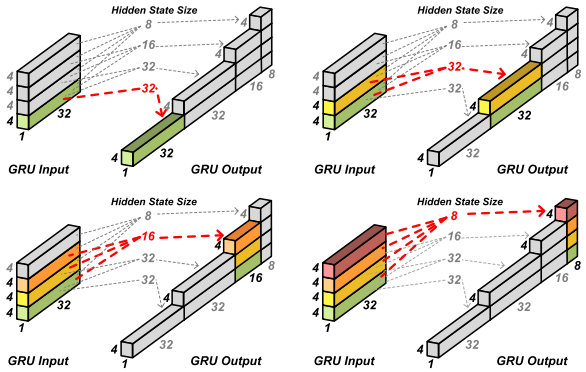


Fig. 6. Computations for each sub-band in BS-TGRU.

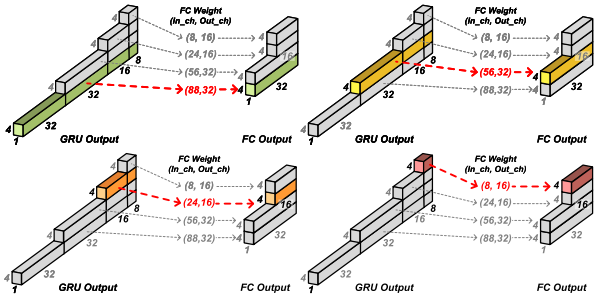


Fig. 7. FC layers to combine results of BS-TGRUs.

GRUs simultaneously processing multiple sub-bands while sharing parameters between sub-bands. The significance of lower sub-bands is reflected in the training process by having additional dedicated parameters not shared with higher sub-bands. The outputs of the individual GRUs for each sub-band are combined by four FC layers, grouping sub-bands of the same color as depicted in Fig. 7. Unlike BS-TGRU with shared parameters across sub-bands, the FC layers assign unique weight parameters to each sub-band to preserve its distinctive characteristics. Through these optimizations, our BS-TGRU requires 58.1% fewer operations than the conventional TGRU. In addition, we apply a similar importance-aware band optimization technique to the encoder and decoder; we reduce the number of channels for the higher sub-band in both the

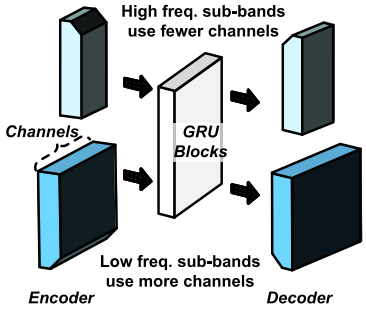


Fig. 8. Importance-aware optimization of encoder and decoder.

		Low Band			High Band		
		In_ch	Out_ch	Filter	In_ch	Out_ch	Filter
Encoder	DW	1	32	5x1	1	16	5x1
	PW	32	64	1x1	16	32	1x1
	DW	1	64	5x1	1	32	5x1
	PW	64	64	1x1	32	64	1x1
	DW	1	64	5x1	1	64	5x1
Decoder	DW	1	32	5x1	1	16	5x1
	PW	32	32	1x1	16	16	1x1
	DW	1	32	5x1	1	16	5x1
	PW	32	32	1x1	16	16	1x1
	DW	1	32	5x1	1	16	5x1
	PW	32	1	1x1	16	1	1x1

		FGRU		BS-TGRU							
				Band 1		Band 2		Band 3		Band 4	
		In	Out	In	Out	In	Out	In	Out	In	Out
GRU	W_i	64	96	32	24	32	48	32	96	32	96
	W_h	32	96	8	24	16	48	32	96	32	96
	Wr_i	64	96								
	Wr_h	32	96								
	FC	64	32	8	16	24	16	56	32	88	32

Fig. 9. Neural network model configuration.

encoder and decoder by half compared to the lower sub-band (Fig. 8), further reducing parameters and operations.

Fig. 9 shows the detailed configurations of our neural network model. The encoder and decoder consist of depthwise separable convolution layers, with a filter size of 5×1 for all depthwise convolution layers. BS-TGRU comprises four GRUs, each with output channels of 8, 16, 32, and 32, respectively. ReLU6 activation function is used between all layers. The final model only has 56k weight parameters and requires 0.71 M MACs to process an input frame. Fig. 10 shows the ablation studies for the proposed band optimization technique. Employing BS-TGRU reduced the overall MAC operations by 19.0% with a minimal drop in evaluation score signal-to-distortion ratio (SDR) of 0.03. In addition, allocating fewer channels to the higher sub-band in both the encoder and decoder resulted in a 12.4% reduction in overall neural network operations and a decrease of 8.6% in parameters, at the expense of a slight evaluation score degradation of 0.12. Combining all optimization techniques described above, the final SE model achieves an 8.6% reduction in parameters and a 29.7% reduction in operations with only a minimal evaluation score decrease of 0.15 compared to the baseline.

To further demonstrate the benefits of our BS-TGRU, we compared it with other possible variants of TGRU. For fair comparisons, we only replace the BS-TGRU with other variants in the proposed SE model without modifying the number of parameters and operations in the FGRU, encoder, and decoder. We first implemented a TGRU with a reversed structure (Fig. 11, Variant #1), where more parameters are allocated to the high-frequency sub-band in contrast to our

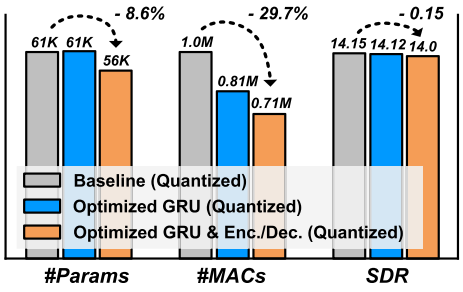


Fig. 10. Ablation studies for band optimization technique.

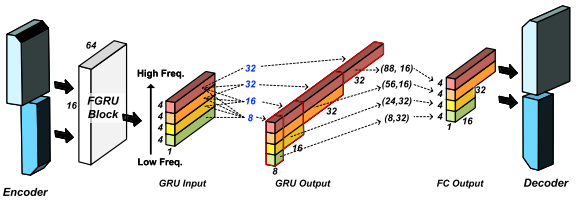


Fig. 11. Modified TGRU with reversed structure (Variant #1).

proposed BS-TGRU. We also designed two additional variants (Variants #2-1 and #2-2) by applying a similar sub-band optimization technique without parameter sharing. Variant #2-1 allocates more resources to the lower sub-band as we did in BS-TGRU, but without sharing parameters across multiple sub-bands; instead, each sub-band has its own unique GRU, with more parameters allocated to the lower sub-bands. In this variant, the number of GRU output channels is adjusted to match that of our BS-TGRU [Fig. 12(a)]. Variant #2-2 has the same structure, but we tuned the GRU hyper-parameters to achieve the same MACs as BS-TGRU [Fig. 12(b)]. Lastly, Variant #3 replaces BS-TGRU in our final model with conventional TGRU shown in Fig. 5(a). Table I compares weight parameters, MACs, and evaluation scores between models. First, our model outperforms the reversed structure with the same number of weight parameters and MACs (Variant #1) in all evaluation scores, confirming the effectiveness of our approach that recognizes the importance of the lower sub-band. Compared to Variant #2-1 with a simple band channel optimization scheme without parameter sharing, our method reduces weights by 63% and MACs by 40% while achieving similar evaluation scores. Compared to Variant #2-2, our method achieves slightly higher evaluation scores while reducing weights by 40%. Finally, compared to the model adopting conventional TGRUs (Variant #3), our method maintains similar evaluation scores but reduces MACs by 58%.

C. Log-Scale 4-Bit Quantization

Previous SE processors employed either INT16 [15] or INT8 [16] data formats to represent weight parameters. Lowering data precision will save storage space and reduce computational power, but we need at least 5 bits for the integer format to avoid significant degradation in SE performance as demonstrated in Fig. 13, which shows experiments with gradually reduced weight precision. Our design mitigates this

TABLE I
COMPARISONS WITH OTHER TGRU STRUCTURES

Structure	Evaluation scores			Weight parameters (k)		MACs (k)	
	PESQ	SDR (dB)	STOI (%)	TGRU	Entire network	TGRU	Entire network
Proposed BS-TGRU	2.73	14	91.44	21	56	137	711
Variant #1: Reversed architecture	2.65	13.41	90.74	21	56	137	711
Variant #2-1: Channel opt. (same output ch.)	2.73	14.13	91.4	58	93	226	800
Variant #2-2: Channel opt. (same MACs)	2.71	13.99	91.24	35	70	136	710
Variant #3: Conventional TGRU	2.72	13.96	91.39	22	57	319	894

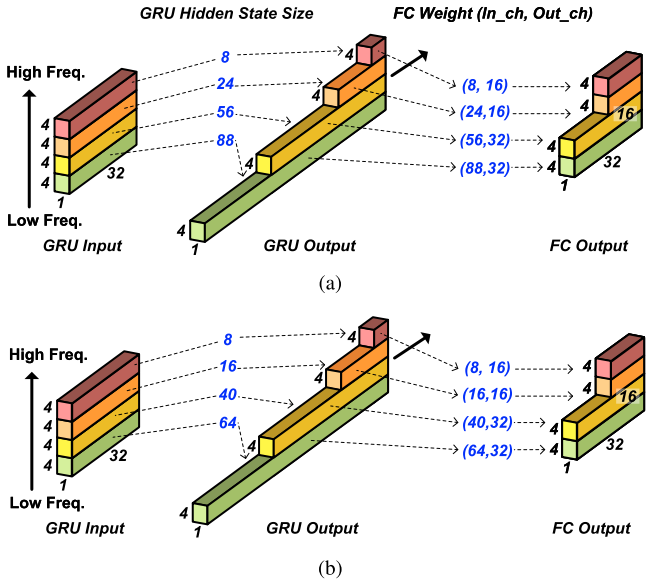


Fig. 12. Modified TGRUs based on a simple channel optimization without parameter sharing. (a) Variant with the same number of GRU output channels as BS-TGRU (Variant #2-1). (b) Variant with the same number of MACs as BS-TGRU (Variant #2-2).

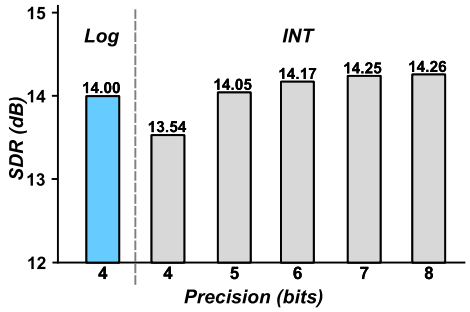


Fig. 13. Comparisons of quantization schemes.

constraint and further reduces the required precision to 4 bits by adopting a logarithmic data format [21] for the weight parameters. Our 4-bit logarithmic format supports a zero value and has an asymmetric dynamic range from -2^7 to 2^6, which is nearly equivalent to the INT8 format, from -2^7 to $2^7 - 1$. As a result, we can minimize the performance degradation of the neural network while significantly reducing the model size. Furthermore, this approach allows for multiplication operations between inputs and weights to be simplified into shift operations, thereby greatly lowering power and area overheads. Since all weight parameters are powers of 2, multiplication with a weight parameter can be simply implemented using

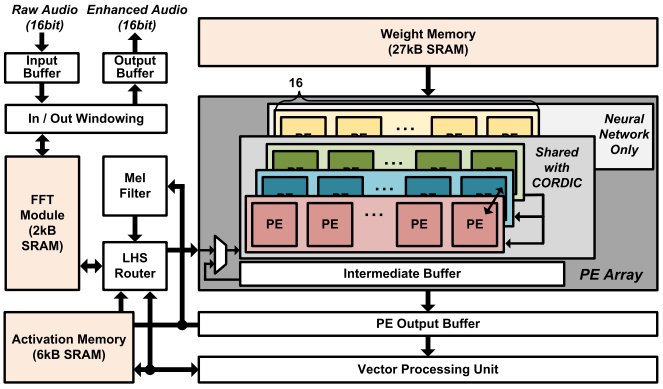


Fig. 14. Top-level architecture of the proposed SE processor.

only a single shift operation, contrary to conventional multipliers that require the accumulation of partial products. Details regarding the weight encoding scheme and hardware implementation of these operations will be discussed in Section III. The final model was obtained by applying a channel-wise quantization to a pre-trained full-precision (FP) model and fine-tuning the quantized model, with all intermediate activations in the neural network quantized to 8-bit integer format.

III. HARDWARE IMPLEMENTATION

A. Overall Architecture

Based on the lightweight accurate SE algorithm discussed in Section II, we designed an energy-efficient real-time SE processor, whose overall architecture is displayed in Fig. 14. In our SE pipeline (Fig. 2), we need to compute the magnitudes and phases from the FFT outputs, as well as to calculate the cosine and sine values of the phases. For these steps, we suggest using CORDIC [22], as it can efficiently realize those complex operations using only shift and add operations. In addition, our neural network model with logarithmic weights also only requires shift and add operations, and hence we designed a unified reconfigurable PE that can support both CORDIC and neural network layers. The PE array consists of 4×16 multiplier-less PEs and performs matrix multiplications for GRUs and convolution layers, while a 3×16 sub-array is reconfigurable for CORDIC operations. Each of the 64 PEs corresponds to one output channel in the neural network computation, enabling high utilization since most layers have output channels in multiples of 16. Also, each PE has a 34.5-B scratchpad memory as a local weight buffer to support row-stationary dataflow and reduce access to the weight memory. A total of 35 kB compiled SRAM macros

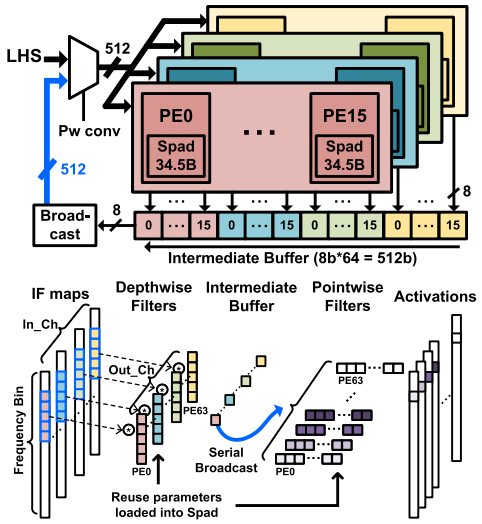


Fig. 15. Layer fusion technique for depthwise separable convolution.

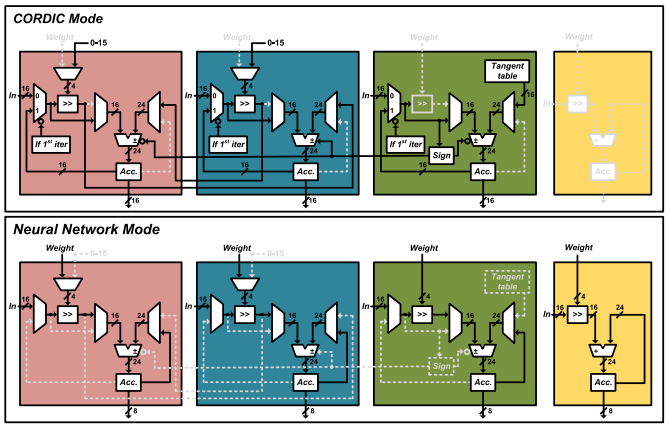


Fig. 16. Reconfigurable PEs.

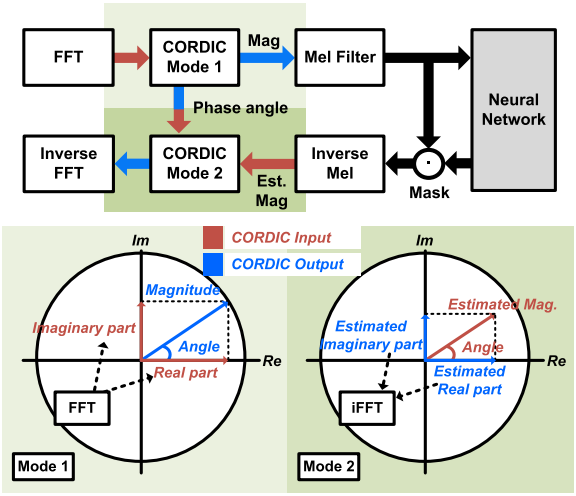


Fig. 17. Two operation modes of CORDIC.

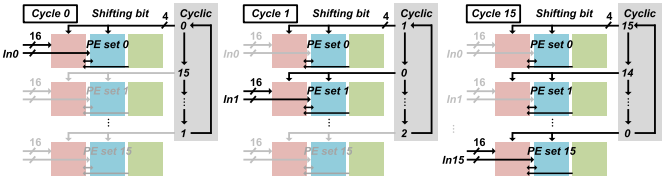


Fig. 18. Example dataflow of CORDIC operation.

are integrated, where 27 kB is used to store the weights of the neural network. A 6-kB activation memory temporarily stores all intermediate activations in each layer. A 2-kB memory within the FFT module is also shared with the PE array to store the results of CORDIC operations. The vector processing unit handles vector operations for GRUs, including LUT-based activation functions and arithmetic operations. The audio input is streamed in at a fixed rate of 16 kHz and temporarily stored in the on-chip input buffer until a frame is collected for subsequent processing. The final output audio is also streamed out at an identical rate of 16 kHz.

Our SE processor fuses depthwise and pointwise layers to alleviate frequent access to the 6-kB activation memory by using a 512-bit intermediate buffer near the PE array, as illustrated in Fig. 15. The intermediate buffer temporarily stores one row of the accumulated results from the depthwise convolution layer. Subsequently, the values stored in the intermediate buffer are broadcasted to the PEs for performing pointwise convolution. This approach eliminates the need to store the activations of the depthwise convolution layer in the activation memory, reducing access energy by 5.34× in simulation.

B. Reconfigurable PEs

To process neural network layers and CORDIC operations with minimal hardware overheads, our SE processor employs multiplier-less reconfigurable PEs. The PE array consists of four types of PEs, which correspond to the PEs with different colors in Fig. 16. All PEs essentially perform add and shift operations, but their internal routings are reconfigured depending on the desired operations.

For CORDIC operations, three types of PEs (red, blue, and green PEs in Fig. 16) are interconnected together. Our SE pipeline requires two separate CORDIC modes, as depicted in Fig. 17. Mode 1 uses the real and imaginary parts of the FFT output as inputs to compute magnitudes and phases. Mode 2 utilizes pre-computed phases to determine the cosine and sine values of the phase angle. These cosine and sine values are then multiplied by the noise-suppressed magnitudes that have passed through an inverse Mel filter, resulting in the real and imaginary parts of the final clean audio output. Each colored PE in a PE set accumulates values for the magnitude (red) and phase angle (green) in Mode 1, and accumulates the cosine (red) and sine values (blue) in Mode 2. When computing the magnitude and phase, the red and blue PEs use the real and imaginary parts as an initial input. In contrast, the green PE uses the phase as an initial input when computing the cosine and sine values of the phase angle. The CORDIC algorithm iteratively performs shift and add operations to approximate the desired result, and simulations show that performing 16 iterations results in no changes in the evaluation scores up to the third decimal place in our SE pipeline. Fig. 18 illustrates an example dataflow of the CORDIC in Mode 1 (calculating magnitude and phase).

Weight	-2^7	-2^6	-2^5	-2^4	-2^3	-2^2	-2^1	-2^0
Encoded Weight	1000	1001	1010	1011	1100	1101	1110	1111
Right Shift	>>0	>>1	>>2	>>3	>>4	>>5	>>6	>>7

Weight	0	2^0	2^1	2^2	2^3	2^4	2^5	2^6
Encoded Weight	0000	0111	0110	0101	0100	0011	0010	0001
Right Shift	Skip	>>7	>>6	>>5	>>4	>>3	>>2	>>1

Fig. 19. Weight encoding scheme.

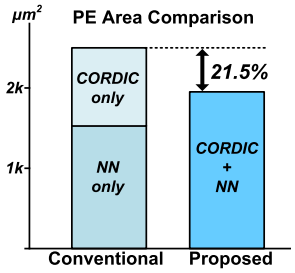


Fig. 20. Comparison of PE implementations.

The first set of PEs (PE set 0) takes the first input (In0) and starts processing it, which will take 16 cycles. Simultaneously, in the next cycle, the next PE set (PE set 1) also takes the next input (In1) and performs CORDIC operations on it. The process continues onto the remaining PEs and returns to PE set 0 after 16 cycles, when it becomes ready to take the next input. As a result, the PE sets are pipelined and fully utilized.

Since our SE model uses 4-bit logarithmic weights, the neural network operations can be replaced with shift and add operations utilizing all four types of PEs. To achieve this, we apply a customized weight encoding scheme as shown in Fig. 19. In this scheme, MSB represents a sign bit, while the lower 3 bits determine the amount of right shift. Since CORDIC uses right-shift operations, we configured the input router to perform a 7-bit left preshift for multiplication with weights, allowing the use of a right shifter during PE operations. All intermediate activations in the neural network are quantized into an 8-bit integer format, and the 16-bit input in Fig. 16 represents $\{A[7], A, 7'b0000000\}$ including a 7-bit preshift, where A is the 8-bit activation from the previous layer. The right-shifted data after multiplication in Fig. 16 is accumulated as a 24-bit value. After completing the computation, the accumulated value is aligned by shifting according to the channel-wise scale value, which is a power of 2. Negative values are set to 0, while positive values are clipped to specific values by the ReLU6 function, and rounding is performed using the round-to-even method. Zero skipping is applied only when the encoded weight is "0000." In all neural network operations, each PE accumulates the output of one output channel, and the weights of each layer are loaded into the scratchpad of each PE before the operation of the layer begins.

CORDIC operations require fewer cycles compared to neural network computations, and they are not concurrently processed during SE tasks. Therefore, using reconfigurable PEs to process both neural network layers and CORDIC operations reduces area overhead without a performance penalty. When synthesized, reconfigurable PE occupies 21.5% less area compared to separate implementations of CORDIC and neural network modules (Fig. 20).

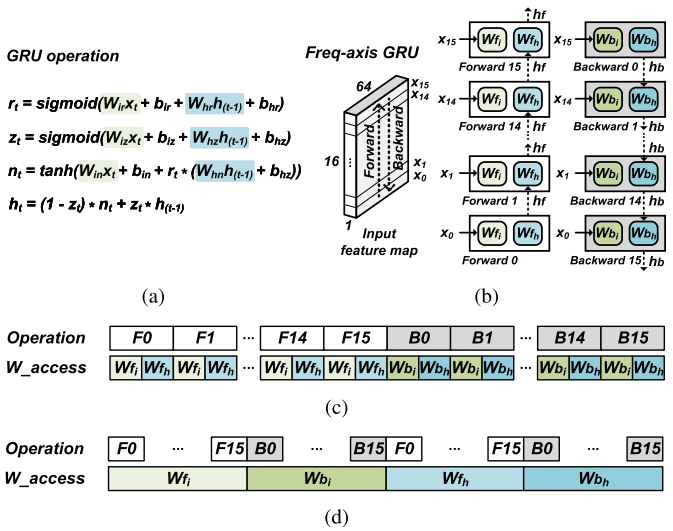
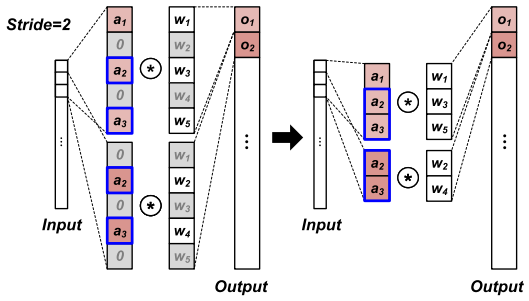


Fig. 21. (a) Typical GRU cell, (b) bi-directional FGRU operation, (c) naïve scheduling, and (d) proposed scheduling.

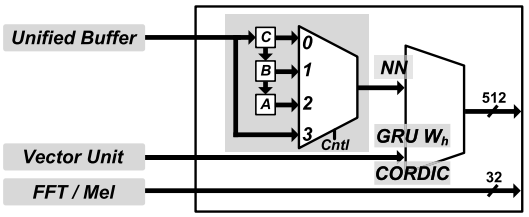
C. On-Chip Memory Access Reduction Techniques

Our SE processor employs two techniques to reduce access to on-chip memory blocks for better energy efficiency. The first technique is optimizing the operation sequence of bi-directional FGRU, as described in Fig. 21. This reduces the memory access that occurs when loading weight matrices into the local weight buffer of each PE. A typical GRU cell requires an input x_t and the hidden state vector of the previous GRU cell h_{t-1}, along with learnable parameters W and b. The reset gate r_t, update gate z_t, and candidate hidden state vector n_t contribute to creating the final output hidden state vector h_t. The weight matrices $\{W_{ir}, W_{iz}, W_{in}\}$ and $\{W_{hr}, W_{hz}, W_{hn}\}$ undergo matrix multiplication with x_t and h_{t-1} in the PE array, respectively [Fig. 21(a)]. In our model, FGRU expands the receptive field along the frequency axis by propagating sub-band features in both directions [Fig. 21(b)]. Thus, there are two separate GRUs for forward and backward directions, each with weight matrices $\{Wf_i, Wf_h\}$ and $\{Wb_i, Wb_h\}$. A straightforward approach would be sequentially processing those GRUs as shown in Fig. 21(c), but this approach suffers from redundant access to the weight memory. Our processor mitigates this issue by employing a modified processing sequence shown in Fig. 21(d). In our scheme, the processor first performs Wf_i and Wb_i matrix operations, followed by Wf_h and Wb_h matrix operations, allowing each weight matrix to be accessed only once. This optimization is based on the observation that Wf_i and Wb_i matrices operate independently on the input x_t, regardless of the computation result h_{t-1} from the previous time step. This scheduling requires temporarily storing the computation results of Wf_i and Wb_i in the activation memory and then fetching them for Wf_h and Wb_h operations. In simulation, this optimization reduces access to the weight memory by 138.2 kB/frame, at the expense of only a 7.2-kB/frame increase in access to the activation memory.

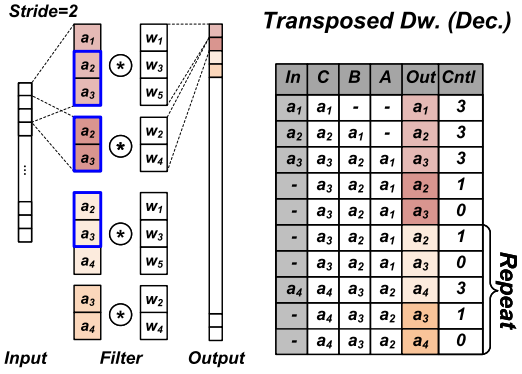
The second technique reduces activation memory access by optimizing the input order of the depthwise convolution layer through input buffering. The decoder of the neural network



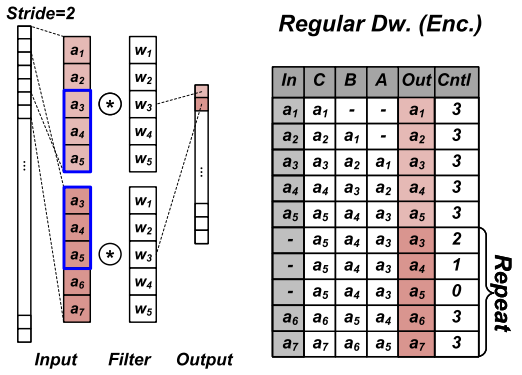
(a)



(b)



(c)



(d)

Fig. 22. (a) Decomposition of transposed depthwise convolution layer, (b) input router for PE array, (c) example of transposed depthwise convolution layer and its control signal pattern, and (d) example of regular depthwise convolution layer and its control signal pattern.

uses a 1-D transposed depthwise convolution layer consisting of 5 × 1 filters. Since the transposed depthwise convolution layer has a stride of 2, the input feature map is expanded by inserting a zero between activations, which incurs unnecessary zero operations (Fig. 22(a), left). To mitigate those redundant zero operations that occupy half of the operations in the transposed depthwise convolution layer, we replace the

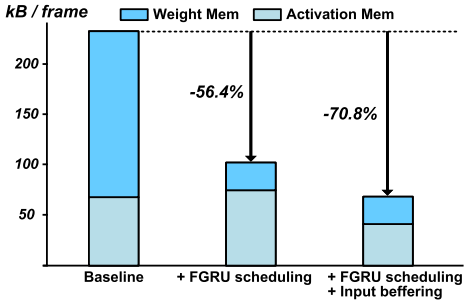


Fig. 23. On-chip memory access reductions.

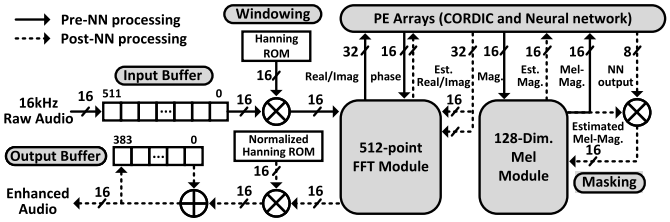


Fig. 24. Hardware for pre- and post-processing.

5 × 1 filters with smaller 3 × 1 and 2 × 1 filters (Fig. 22(a), right), effectively eliminating the redundant zero operations. However, it still suffers from duplicate activation memory access since the stride is smaller than the kernel size. More specifically, since the kernel slides over the input activations, if the stride is smaller than the kernel size, an input activation should be repeatedly accessed to generate multiple output activations. Fig. 22(a) shows an example of this overhead, where a_2 and a_3 are used for generating both o_1 and o_2. We avoid this overhead through locally buffering inputs using the input router circuit depicted in Fig. 22(b). The three buffers A, B, and C in the input router temporarily store activations and feed them into the PE in the correct order. This ensures that the activation values stored in the activation memory are accessed only once when processing transposed depthwise convolution layers. Fig. 22(c) details the operations of the transposed depthwise convolution layer in the decoder and the control signals for the multiplexer. Activations a_2 and a_3 (marked by blue boxes) are required not only for the first two outputs but also for the third output. The control signals in the table appropriately route the buffered inputs to provide the required data to the PE. Fig. 22(d) shows that similar redundant memory access patterns are found in regular 1-D depthwise convolution layers in the encoder, and our input router removes this overhead using the control signals in the table. In both cases, the last five patterns in the table continue to repeat. Fig. 23 shows the overall memory access reductions. The optimal scheduling in FGRU reduces on-chip memory access by 56.4%, and the input buffering scheme lowers it further by 33.1%, achieving a total reduction of 70.8%

D. Pre- and Post-NN Processing Hardware

Fig. 24 illustrates the hardware module for pre- and post-processing in our SE pipeline. The solid lines represent feature extraction before the neural network, while the dashed lines

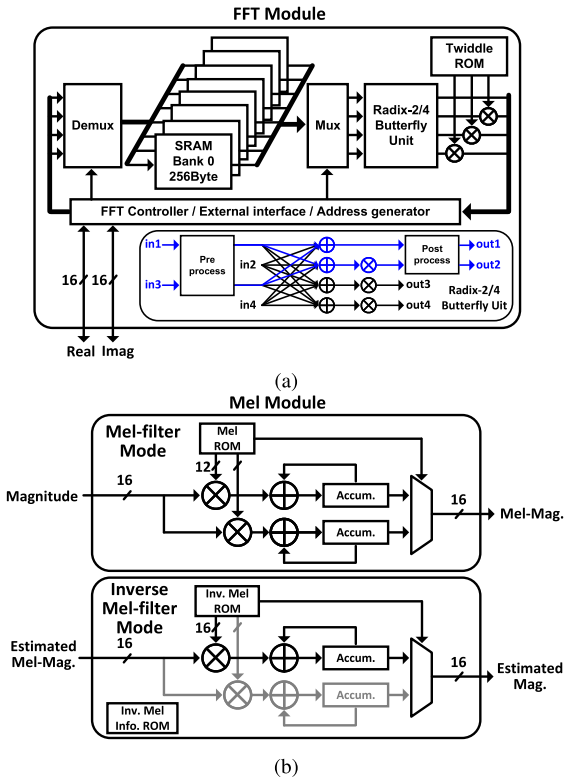


Fig. 25. (a) FFT module. (b) Mel module implementation.

indicate the audio reconstruction process after the neural network. The bitwidths of the internal paths and coefficients in the feature extractor were determined using quantize-aware fine-tuning to avoid degrading application performance. Our SE processor primarily supports 50% hopping with a 32-ms window size and 16-ms hop size, but it also supports 25% hopping with an 8-ms hop size for better SE quality. To support variable hop size, additional ROM, input, and output buffer control logic are integrated into the processor. Initially, 16-kHz 16-bit raw audio input samples are stored in a 1-kB input buffer, and once the input buffer is full, the computation for the current frame begins. The input samples are multiplied by the windowing coefficients stored in a 1-kB ROM before entering the FFT module detailed in Fig. 25(a). The FFT module performs a mixed-radix 512-point real-valued FFT using memory-based architecture [23]. To reduce latency and improve energy efficiency, it employs radix-4 for all stages except the post-processing that reflects the symmetry of real-valued inputs, which is implemented using radix-2, completing the computation in 392 cycles. The butterfly unit supports both radix-2 and radix-4 operations, and only the blue part in the figure is activated during radix-2 operations. A 1.28-kB twiddle factor ROM contains the coefficients for both radix-2 and radix-4 operations. The 32-bit complex-valued outputs from FFT are transferred to the PE array to compute magnitudes and phases using CORDIC. The 16-bit phase values are temporarily stored in the FFT memory, while the 16-bit magnitude values pass through a 128-D Mel filter displayed in Fig. 25(b). The 832-B Mel ROM used for Mel filtering has a bit width of 26 bits, consisting of two 12-bit coefficients required for two multipliers and two 1-bit signals for exporting and initializing the values of the two accumulators. The 16-bit Mel-magnitudes resulting from Mel filtering are used as the input features for the neural network, and they are temporarily stored in the FFT memory until the output of the neural network is obtained.

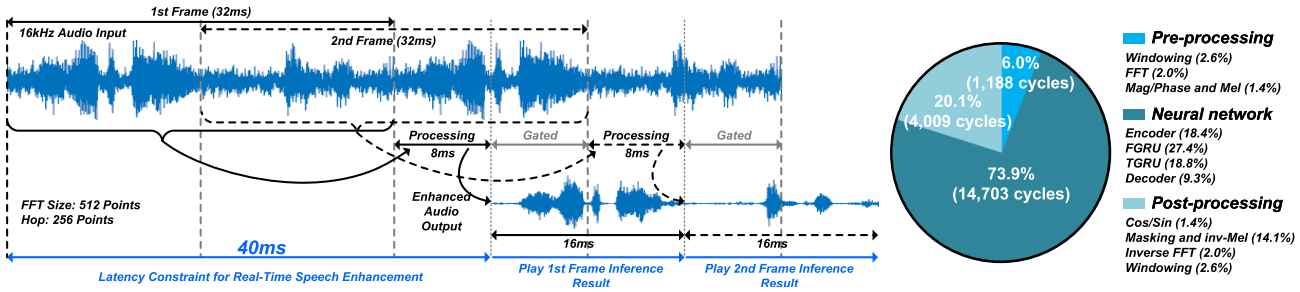
The audio reconstruction process starts with masking, which performs element-wise multiplication between the 8-bit neural network output and the Mel-magnitudes. The masked values undergo pseudo-inverse Mel filtering, sharing hardware with regular Mel filtering [Fig. 25(b)]. Unlike typical triangular Mel filters that require only two operations per cycle [24], [25], the inverse Mel filter overlaps multiple filters within a single cycle. Therefore, an additional 350-B ROM containing the start address (7 bits) and accumulation count (4 bits) is used in addition to the 1.98-kB inverse Mel coefficient ROM with 16-bit bitwidth. The noise-suppressed magnitudes obtained by the inverse Mel filter are passed to the PE array along with the pre-generated phase during the feature extraction process. Subsequently, noise-suppressed real and imaginary parts are generated through CORDIC operations. These values are then transferred to the FFT module, where it runs a mixed-radix complex-to-real 512-point inverse FFT for 394 cycles. The output of the inverse FFT is multiplied by a normalized Hanning window coefficient for the reconstruction of the audio signal. The normalized window coefficients are stored in two separate 1-kB ROMs, each for the 50% and 25% hopping modes, respectively. Finally, to recover the overlapped regions caused by hopping, the processor adds a portion of the final output from the previous frame stored in the 768-B output buffer to the windowed output of the current frame, generating a reconstructed audio as the final output.

IV. Experimental Results

Since our processor is designed to perform real-time SE for hearing aids, it must meet real-time constraints. We target a total latency of 40 ms, following the requirements for real-time denoising in the well-known deep noise suppression challenge [26]. The total latency refers to the time from when the raw input enters the processor to when the enhanced output is produced. With a window size of 32 ms, the processor meets the 40-ms requirement only if each frame can be processed within 8 ms as illustrated in Fig. 26(a). Fig. 26(b) shows the latency breakdown of each processing stage. Since our processor requires 19 900 cycles to process a single frame, the minimum operating frequency is 2.5 MHz to complete the entire processing within 8 ms.

A. Test Chip Measurements and Comparisons

The proposed SE processor was fabricated in a 28-nm CMOS process, and Fig. 27 shows the die photograph. The core area is 0.81 mm^2 including a total of 35-kB on-chip SRAM. Fig. 28 displays the measurement setup. The FPGA board only acts as an interface between the processor and the host PC. For measurements, an audio file is first loaded into the internal memory of FPGA. Then, the audio samples are sequentially streamed into the chip at a fixed rate of 16 kHz.



(a) (b)

Fig. 26. (a) Overview of real-time SE in our design and (b) latency breakdown of each processing stage.

TABLE II
COMPARISON TABLE

	This Work	JSSC'20 [15]	INTERSPEECH'20 [16]		TCAS-II'21 [27]
Implementation	**ASIC**	ASIC	FPGA	FPGA	ASIC (Sythesized)
Technology (nm)	**28**	40	-	-	90
Neural Network	**Separable Conv, GRU, FC**	Conv, FC	LSTM, FC	LSTM, FC	FC
FFT Window / Hop	**512 / 256**	256 / 128	512 / 256	400 / 100	512 / -
Weight Precision (bits)	**4**	16 (stored as 4-bit)	8	8	8
Parameters (k)	**56**	495	330	460	64
Model Size (kB)	**28**	248[1]	310	430	62
Total Latency[2] (ms)	**39.96**	41.88	36.26[3]	27.39[3]	32[3]
Frequency (MHz)	**2.5 - 20**	5 - 20	-	-	500
On-Chip SRAM (kB)	**35**	327	313.7	434.67	-
Core Area (mm^2)	**0.81**	4.2	-	-	11.4
Power (mW)	**0.74 (0.8V, 2.5MHz)**	2.17 (0.6V, 5MHz)	144.9	206.4	636 (1.2V, 500MHz)
Frames / sec	**63**	125	63	160	-
Efficiency (μJ/frame)	**11.75**	17.36	2300	1290	10095.24[4]
Dataset	**CHiME2**	CHiME2	CHiME2	CHiME2	TIMIT
SDR (dB) / PESQ / STOI (%)	**14.00 / 2.73 / 91.44**	12.21[5] / 2.38[5] / 89.08[5]	13.18 / - / -	12.96 / - / -	- / 1.52 / -

[1]Estimated assuming 4-bit weights [2]Delay from raw audio input to enhanced audio output
[3]Neural network only (pre- and post-processing excluded) [4]Assuming hop size = 50% of FFT window size [5]Our implementation in FP32

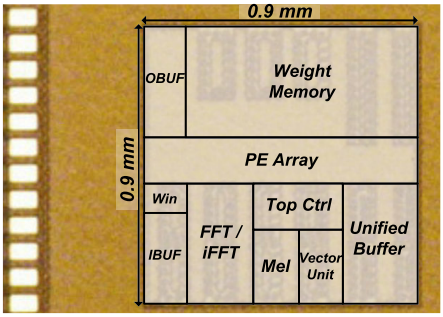


Fig. 27. Die photograph.

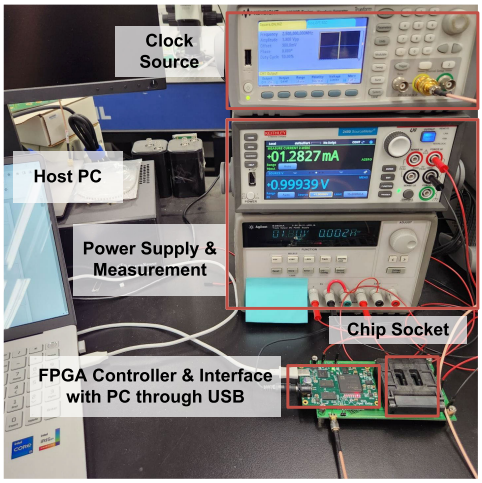


Fig. 28. Measurement setup.

The final output is also sent back to FPGA at an identical data rate of 16 kHz. As demonstrated in Fig. 29, the test setup allows for comparing the input audio and enhanced audio in real time. Fig. 30 shows the measured power consumption. The processor consumes 740 μW with 50% hopping and 790 μW with 25% hopping at 2.5 MHz and 0.8 V. Fig. 31 displays the energy breakdown estimated using a final place-and-routed netlist at 1.0 V. Although our design does not include analog front-end circuits such as amplifier and ADC, considering that the analog chain in prior fully integrated SE SoCs [28], [29] consumes between 0.34 and 0.39 mA at 1.0 V, our design is expected to meet the power budget when integrated with the analog front-end.

Table II shows comparisons with state-of-the-art designs. Our design consumes 2.05× lower power and occupies 2.54× less area compared to the prior ASIC design in [15] considering technology scaling (linear for power and quadratic for area), while achieving significantly higher evaluation scores. Furthermore, compared to the FPGA design in [16], our processor exhibits higher evaluation scores despite employing a significantly smaller neural network model. While our

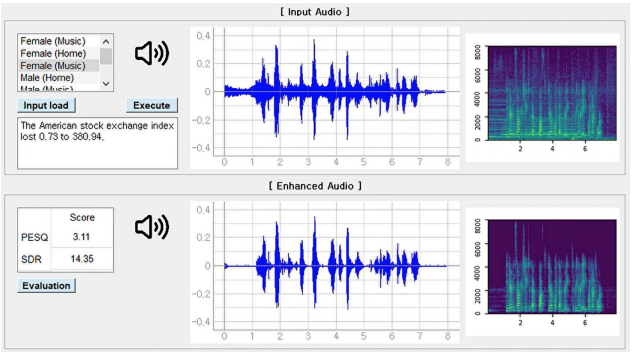


Fig. 29. Demo GUI.

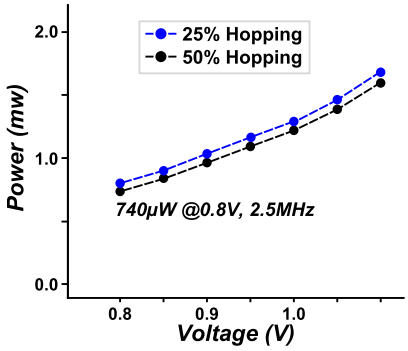


Fig. 30. Measured power consumption.

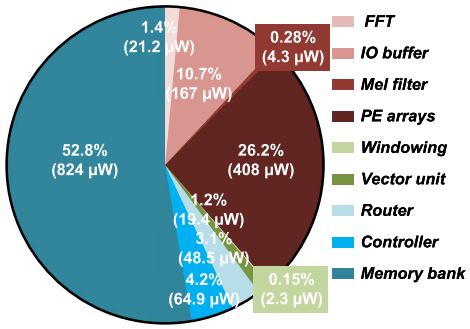


Fig. 31. Power breakdown.

design is optimized for the proposed SE model and hence supports lower flexibility than general-purpose processors, its superior energy efficiency allows for the hearing aids to utilize high-quality SE algorithm within the power budget. For instance, the authors in [16] presented an SE system using an STM32F746VE microcontroller [30] and reported a power consumption of 0.54 W at 155 MOps/s. If the processor runs our SE model, it is expected to consume 312.1 mW power, significantly exceeding the power budget discussed in Section I.

It could be also useful to compare our design with other SE processors employing conventional SE algorithms [28], [29], [31], [32]. Most of these processors [28], [29], [31] are larger in size and have higher power consumption compared to our design, with sizes ranging from 3.6 to 9.3 mm^2 and power consumption of over 1 mW. Another design in [32] reports

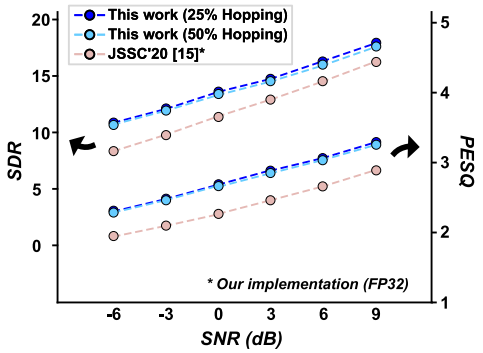


Fig. 32. Objective evaluation results.

a small area of 0.3 mm^2 through post-layout estimation but demonstrates a higher power consumption of 1.5 mW. Quantitative and qualitative evaluations of their actual application performance are not reported, except for the design in [32] that exhibits significantly lower short-time objective intelligibility (STOI) [33] scores ranging from 58.98 to 77.07.

B. Objective Evaluation

In experiments, the CHiME2 WSJ0 dataset [34] was used for model training and evaluation. The dataset consists of 166 h of English speech recorded in noisy living room environments with various background noises such as a vacuum cleaner, television, and child speech. The dataset is divided into training, validation, and test sets, comprising 7138, 2460, and 1980 utterances, respectively. In addition, the dataset is categorized into various signal-to-noise ratio (SNR) levels, ranging from −6 to 9 dB, with a 3-dB interval.

For objective evaluation of the SE processor, SDR [35], perceptual evaluation of speech quality (PESQ) [36], and STOI are adopted. Multiple metrics are used for evaluating enhanced voice quality to maximize the reliability of objective performance evaluations. In experiments, our design demonstrates high SE quality on the CHiME2 dataset across the entire SNR range, outperforming prior work (Fig. 32). It should be noted that the authors in [15] did not report these evaluation metrics, and hence we used our own FP software implementation of their algorithm for comparison. In the 50% hopping mode, the average values of SDR, PESQ, and STOI are 14.00, 2.73, and 91.44, respectively. The 25% hopping mode shows slightly higher performance, achieving 14.31, 2.80, and 91.61, respectively, whereas the average values from the software model of [15] are 12.21, 2.38, and 89.08, respectively.

C. Subjective Evaluation

While objective evaluation is a useful tool to assess the quality of enhanced audio, relying solely on objective metrics for performance evaluation may yield different results than those perceived by actual listeners. Hence, we conducted a human opinion test utilizing the ITU-T P.808 subjective evaluation of speech quality with a crowdsourcing approach [37] using Amazon Mechanical Turk (MTurk).

For subjective evaluations, human participants are presented with the prompt “Which audio sample has a clearer voice

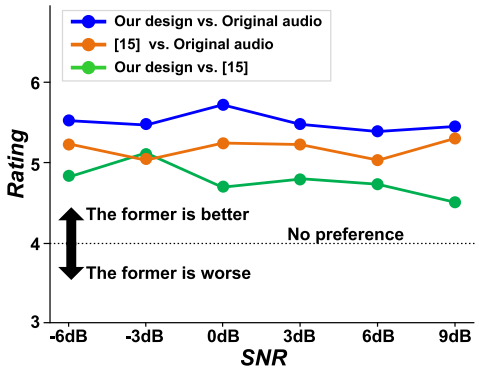


Fig. 33. Subjective evaluation results.

and better quality with less interference with the background noise?" The participants then choose a score on a seven-point Likert scale [38], where the score ranges from one (Sample B is strongly better) to seven (Sample A is strongly better) with four indicating no preference. Each participant is assigned one of three types of comparisons: 1) enhanced audio from our algorithm versus original noisy audio; 2) enhanced audio from our algorithm versus enhanced audio from the algorithm in [15]; and 3) enhanced audio from the algorithm in [15] versus original noisy audio. A high score suggests that the first sample is subjectively perceived to have better quality than the second sample. For evaluation, 50 samples are randomly selected from the CHiME2 test set at each SNR level, which produces five sample sets since each sample set consists of 10 samples. Consequently, we have 30 sample sets in total across the entire SNR range. Each sample set is assigned to 30 participants, and they conduct one of the comparisons above on the assigned sample set. Therefore, the total number of participants was 2700. The participants were presented with the experiment details, duration, and compensation before participating in the evaluations, and their understanding was verified.

The evaluation results are shown in Fig. 33. The blue line indicates that the audio output from our design received better qualitative ratings compared to the original noisy audio. While the audio generated by the algorithm in [15] also received favorable ratings compared to the noisy audio (orange line), its score is lower than ours over the entire SNR range. A direct comparison between our design and [15] (green line) also confirms that our approach achieves higher SE quality, which is consistent with objective evaluations.

It should be noted that we employed two filtering questions based on [37] to rule out questionable evaluations.

1) *Environmental Filter:* We inquired about the current listening environment and the level of surrounding noise. Then, we filter out the results when the participant is not wearing headphones or the surrounding noise level falls into the categories of "somewhat intrusive" or "very intrusive" on the five-category intrusiveness scale (not noticeable, slightly noticeable, noticeable but not intrusive, somewhat intrusive, and very intrusive).
2) *Trap Question Filter:* We included a trap question in the evaluation set where the first sample has apparently superior quality and filtered out the results when respondents rated it below "No preference."

V. Conclusion

In this work, we proposed a real-time SE processor aimed at hearing aids. We first presented a lightweight yet accurate SE algorithm. Our SE pipeline utilizes the Mel-filtered input feature for a more detailed analysis of the low-frequency range containing crucial information, also significantly reducing the input dimension. Importance-aware band optimization technique allocates more resources to the essential parts in the input feature domain, thereby reducing computational costs by 29.7% while maintaining SE quality. Along with the 4-bit logarithmic quantization of the neural network, the processor employs multiplier-less PEs, which are reconfigurable for CORDIC and neural network layers. This reconfigurable multiplier-less PE occupies 21.5% less area compared to the separate implementation of those two operations. In addition, optimal scheduling for FGRU operations and input buffering reduces on-chip memory access by 70.8%. Fabricated in 28 nm, the proposed SE processor consumes only 740 μW at 0.8 V while meeting the real-time processing constraints. Our design achieves 2.05$\times$ lower power consumption and 2.54$\times$ smaller area than prior art, exhibiting better SE quality in both objective and subjective evaluations.

Acknowledgment

The chip fabrication and EDA tools support was provided by the IC Design Education Center (IDEC).

References

[1] J. Benesty, S. Makino, and J. Chen, *Speech Enhancement*. Cham, Switzerland: Springer, 2006.
[2] C. Yang, Y. M. Saidutta, R. S. Srinivasa, C.-H. Lee, Y. Shen, and H. Jin, "Robust keyword spotting for noisy environments by leveraging speech enhancement and speech presence probability," in *Proc. INTERSPEECH*, Aug. 2023, pp. 1638–1642.
[3] J. Du, Q. Wang, T. Gao, Y. Xu, L.-R. Dai, and C.-H. Lee, "Robust speech recognition with speech enhanced deep neural networks," in *Proc. Interspeech*, Sep. 2014, pp. 616–620.
[4] Y. Xu, J. Du, L.-R. Dai, and C.-H. Lee, "An experimental study on speech enhancement based on deep neural networks," *IEEE Signal Process. Lett.*, vol. 21, no. 1, pp. 65–68, Jan. 2014.
[5] M. Kolbæk, Z.-H. Tan, and J. Jensen, "Speech intelligibility potential of general and specialized deep neural network based speech enhancement systems," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 25, no. 1, pp. 153–167, Jan. 2017.
[6] H.-S. Choi, S. Park, J. H. Lee, H. Heo, D. Jeon, and K. Lee, "Real-time denoising and dereverberation wtih tiny recurrent U-Net," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2021, pp. 5789–5793.
[7] Y. Hu et al., "DCCRN: Deep complex convolution recurrent network for phase-aware speech enhancement," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2020, pp. 2472–2476.
[8] S. Zhao, T. H. Nguyen, and B. Ma, "Monaural speech enhancement with complex convolutional block attention module and joint time frequency losses," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2021, pp. 6648–6652.
[9] S. Sun et al., "A lightweight Fourier convolutional attention encoder for multi-channel speech enhancement," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2023, pp. 1–5.
[10] J. Yu and Y. Luo, "Efficient monaural speech enhancement with universal sample rate band-split RNN," in *Proc.IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2023, pp. 1–5.

[11] S. E. Eskimez, T. Yoshioka, A. Ju, M. Tang, T. Pärnamaa, and H. Wang, "Real-time joint personalized speech enhancement and acoustic echo cancellation," in *Proc. INTERSPEECH*, Aug. 2023, pp. 1050–1054.

[12] X. Xu, W. Tu, and Y. Yang, "PCNN: A lightweight parallel conformer neural network for efficient monaural speech enhancement," in *Proc. INTERSPEECH*, Aug. 2023, pp. 2438–2442.

[13] M. H. de Sá, A. M. F. R. Pinto, and V. B. Oliveira, "Passive direct methanol fuel cells as a sustainable alternative to batteries in hearing aid devices—An overview," *Int. J. Hydrogen Energy*, vol. 47, no. 37, pp. 16552–16567, Apr. 2022.

[14] J. Ellison and M. MacRae. *The Emergence of Silver-Zinc Rechargeable Technology Within Hearing Aids*. Accessed: Sep. 20, 2024. [Online]. Available: https://www.starkeypro.com/continue-learning/research-and-publications

[15] Y.-C. Lee, T.-S. Chi, and C.-H. Yang, "A 2.17-mW acoustic DSP processor with CNN-FFT accelerators for intelligent hearing assistive devices," *IEEE J. Solid-State Circuits*, vol. 55, no. 8, pp. 2247–2258, Aug. 2020.

[16] I. Fedorov et al., "TinyLSTMs: Efficient neural speech enhancement for hearing aids," in *Proc. Interspeech*, Oct. 2020, pp. 4054–4058.

[17] S. Park, S. Lee, J. Park, H.-S. Choi, and D. Jeon, "22.8 A0.81 mm^2 740μW real-time speech enhancement processor using multiplier-less PE arrays for hearing aids in 28nm CMOS," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2023, pp. 21–23.

[18] S. S. Stevens, J. Volkmann, and E. B. Newman, "A scale for the measurement of the psychological magnitude pitch," *J. Acoust. Soc. Amer.*, vol. 8, no. 3, pp. 185–190, Jan. 1937.

[19] M. P. Gelfer and Q. E. Bennett, "Speaking fundamental frequency and vowel formant frequencies: Effects on perception of gender," *J. Voice*, vol. 27, no. 5, pp. 556–566, Sep. 2013.

[20] A. Jongman, R. Wayland, and S. Wong, "Acoustic characteristics of English fricatives," *J. Acoust. Soc. Amer.*, vol. 108, no. 3, pp. 1252–1263, Sep. 2000.

[21] D. Miyashita, E. H. Lee, and B. Murmann, "Convolutional neural networks using logarithmic data representation," 2016, *arXiv:1603.01025*.

[22] J. E. Volder, "The CORDIC trigonometric computing technique," *IRE Trans. Electron. Comput.*, vol. 8, no. 3, pp. 330–334, Sep. 1959.

[23] B. M. Baas, "A low-power, high-performance, 1024-point FFT processor," *IEEE J. Solid-State Circuits*, vol. 34, no. 3, pp. 380–387, Mar. 1999.

[24] J.-H. SeoI et al., "A 1.5μW end-to-end keyword spotting SoC with content-adaptive frame sub-sampling and fast-settling analog frontend," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2023, pp. 1–3.

[25] C. Li et al., "A 0.61-μW fully integrated keyword-spotting ASIC with real-point serial FFT-based MFCC and temporal depthwise separable CNN," *IEEE J. Solid-State Circuits*, vol. 59, no. 3, pp. 867–877, Mar. 2024.

[26] H. Dubey et al., "ICASSP 2022 deep noise suppression challenge," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2022, pp. 9271–9275.

[27] S. R. Chiluveru, G. Singh, S. Chunarkar, M. Tripathy, and B. K. Kaushik, "Efficient hardware implementation of DNN-based speech enhancement algorithm with precise sigmoid activation function," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 68, no. 11, pp. 3461–3465, Nov. 2021.

[28] L.-M. Chen et al., "A 1-V, 1.2-mA fully integrated SoC for digital hearing aids," *Microelectron. J.*, vol. 46, no. 1, pp. 12–19, Jan. 2015.

[29] C. Chen et al., "A 1 V, 1.1 mW mixed-signal hearing aid SoC in 0.13μm CMOS process," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2016, pp. 225–228.

[30] *ST Microelectronics STM32F746VE*. Accessed: Sep. 20, 2024. [Online]. Available: https://www.st.com/content/stcom/en/products/microcontrollers-microprocessors/stm32-32-bit-arm-cortex-mcus/stm32-high-performance-mcus/stm32f7-series/stm32f7x6/stm32f746ve.html

[31] L. Gerlach, G. Payá-Vayá, and H. Blume, "KAVUAKA: A low power application specific hearing aid processor," in *Proc. IFIP/IEEE 27th Int. Conf. Very Large Scale Integr. (VLSI-SoC)*, Oct. 2019, pp. 99–104.

[32] Y.-J. Lin, Y.-C. Lee, H.-M. Liu, H. Chiueh, T.-S. Chi, and C.-H. Yang, "A 1.5 mW programmable acoustic signal processor for hearing assistive devices with speech intelligibility enhancement," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 67, no. 12, pp. 4984–4993, Dec. 2020.

[33] C. H. Taal, R. C. Hendriks, R. Heusdens, and J. Jensen, "A short-time objective intelligibility measure for time-frequency weighted noisy speech," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, Mar. 2010, pp. 4214–4217.

[34] E. Vincent, J. Barker, S. Watanabe, J. Le Roux, F. Nesta, and M. Matassoni, "The second 'CHiME'speech separation and recognition challenge: Datasets, tasks and baselines," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2013, pp. 126–130.

[35] E. Vincent, R. Gribonval, and C. Fevotte, "Performance measurement in blind audio source separation," *IEEE Trans. Audio, Speech Language Process.*, vol. 14, no. 4, pp. 1462–1469, Jul. 2006.

[36] A. W. Rix, J. G. Beerends, M. P. Hollier, and A. P. Hekstra, "Perceptual evaluation of speech quality (PESQ)-a new method for speech quality assessment of telephone networks and codecs," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, vol. 2, Jun. 2001, pp. 749–752.

[37] *Subjective Evaluation of Speech Quality With a Crowdsourcing Approach*, document ITU-T, Recommendation P.808, 2018.

[38] R. Likert, "A technique for the measurement of attitudes," *Arch. Psychol.*, vol. 22, p. 140, Jan. 1932.

Sungjin Park (Member, IEEE) received the B.S. degree in computer science and electrical engineering from Handong Global University (HGU), Pohang, South Korea, in 2017, and the Ph.D. degree in intelligence and information engineering from Seoul National University (SNU), Seoul, South Korea, in 2024.

He is currently a Post-Doctoral Fellow with the SNU Center for AI Education and Research, Seoul. His research interests include audio signal processing algorithm, auditory application-oriented hardware and system, and low-power application-specific integrated circuit (ASIC) design for deep learning processors.

Sunwoo Lee (Graduate Student Member, IEEE) received the B.S. degree from the Department of Electrical and Computer Engineering, Seoul National University, Seoul, South Korea, in 2020, where he is currently pursuing the Ph.D. degree.

His research interests include quantized neural networks, low-precision neural network training algorithms and systems, and low-power application-specific integrated circuit (ASIC) design for deep learning inference/training.

Jeongwoo Park (Member, IEEE) received the B.S. degree from the Department of Electrical and Computer Engineering, Seoul National University, Seoul, South Korea, in 2017, and the Ph.D. degree from the Department of Intelligent Systems Engineering, Seoul National University, in 2022.

His research interests include neuromorphic algorithm and systems, low-precision floating point arithmetics for deep learning, and neural processing units.

Hyeong-Seok Choi received the Ph.D. degree in electrical and computer engineering from Seoul National University (SNU), Seoul, South Korea, in 2022.

He is currently a Machine Learning Researcher at ElevenLabs, working on various speech-related research and applications. He co-founded Supertone, an audio technology start-up, and contributed to the development of real-time voice conversion technology, which was recognized as a CES 2022 Innovation Awards Honoree in the Software and Mobile Apps category. His research interests include speech enhancement, representation learning, controllable synthesis of speech and singing voices, and speech-related technologies.

Kyogu Lee (Senior Member, IEEE) received the B.S. degree in electrical engineering from Seoul National University, Seoul, South Korea, in 1996, and the Ph.D. degree in computer-based music theory and acoustics from Stanford University, Stanford, CA, USA, in 2008.

He is currently a Professor with the Department of Intelligence and Information, Seoul National University, and leads the Music and Audio Research Group. His research interests include signal processing and machine learning techniques applied to music, audio, and hearing.

Dongsuk Jeon (Member, IEEE) received the B.S. degree in electrical engineering from Seoul National University, Seoul, South Korea, in 2009, and the Ph.D. degree in electrical engineering from the University of Michigan, Ann Arbor, MI, USA, in 2014.

From 2014 to 2015, he was a Post-Doctoral Associate with Massachusetts Institute of Technology, Cambridge, MA, USA. He is currently an Associate Professor with the Graduate School of Convergence Science and Technology, Seoul National University. His current research interests include hardware-oriented machine learning algorithms, hardware accelerators, and low-power circuits.

Dr. Jeon was a recipient of the Samsung Scholarship for Graduate Studies in 2009, the Samsung Humantech Thesis Contest Gold Award in 2021, and the Best Design Award at International Symposium on Low Power Electronics and Design (ISLPED) in 2021. He has served on the Technical Program Committee of the ACM/IEEE Design Automation Conference (DAC), IEEE/ACM Asia and South Pacific Design Automation Conference (ASP-DAC), and IEEE Asian Conference on Solid-State Circuits (ASSCC). He is currently serving as a Distinguished Lecturer for the IEEE Solid-State Circuits Society and an Associate Editor for IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS.