RUAH: A Reconfigurable Unified Acoustic Processor with Hierarchical Inference for Keyword Spotting and Speaker Verification

Donghwan So[1], Seojin Kim[1], Huijeong Woo[1], Jeongmin Lee[1], Yejin Hwang[2], Seoyoon Chae[2], John Kustin[3], Seungjong Lee[3] and Taewook Kang[1]

[1] Department of Semiconductor Convergence Engineering, Sungkyunkwan University, Suwon, South Korea,
[2] Department of Electronic and Electrical Engineering, Sungkyunkwan University, Suwon, South Korea,
[3]University of Michigan, Ann Arbor, MI, USA

***Abstract*—Always-on acoustic inference for edge devices requires an energy-efficient and scalable processor, yet prior designs remain limited by fixed-task operation and redundant computation. RUAH is a reconfigurable, unified, hierarchical AI processor for acoustic processing that supports VAD, KWS, SV, and more (Spoofing and Emotion) within a single hardware architecture. It employs a shared neural front-end and a modified BC-ResNet backbone to enable progressive inference by reusing intermediate features across task stages. RUAH reduces redundant computation through frame-wise incremental processing and a dynamic global buffer that maximizes feature reuse during streaming inference. It also supports reconfigurable execution of multiple models and task modes through a hierarchical control unit and a multi-mode PE. Fabricated in 65-nm CMOS, RUAH achieves 97.6% VAD, 95.2% KWS, and 96.8% SV accuracy while consuming 39.59µW, demonstrating a scalable and energy-efficient solution for always-on acoustic intelligence.**

***Index Terms*—Reconfigurable processor, Hierarchical inference, Voice activity detection (VAD), Keyword spotting (KWS), Speaker verification (SV), BC-ResNet**

I. Introduction

Always-on acoustic inference reduces power consumption in edge speech interfaces by continuously monitoring the acoustic stream and activating more complex processing only when required. However, prior acoustic processors have mainly focused on single-task or fixed-function architectures [1]–[9]. Practically, such designs provide limited support for task expansion or replacement across different users. They also duplicate feature extraction across tasks, leading to redundant computation and memory access [7], [10], and commonly rely on simple front-end and voice-activity detectors that are not robust to noisy conditions [1]–[4], [6]–[8]. As a result, they are not well-suited to hierarchical acoustic processing, where robust low-complexity monitoring and selective extension to deeper inference stages are both required.

To address these limitations, we present RUAH, a reconfigurable unified acoustic hierarchical processor for always-on inference, as shown in Fig. 1. The proposed processor has four key features: 1) a reconfigurable hardware architecture that supports task-dependent process and multi-mode execution on a single processor, while maximizing data stationary; 2) a modified BC-ResNet with frame-wise incremental processing to reduce redundant computation and memory access during streaming; 3) feature reuse across hierarchical stages to avoid repeated feature extraction; 4) NN-based VAD and front-end for robust operation under noisy conditions, replacing fixed feature extraction, and supporting task-aware optimization.

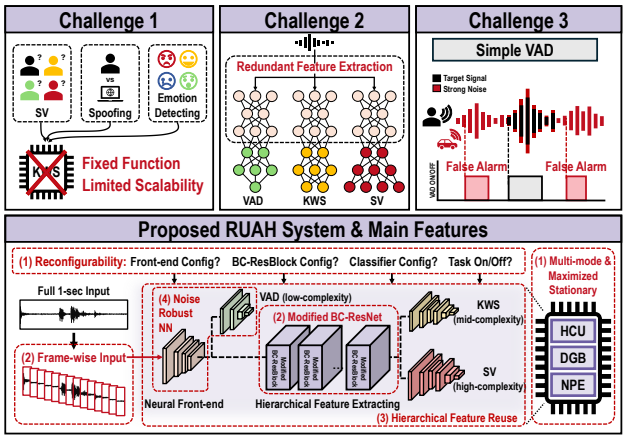

Fig. 1. Challenge and Proposed Idea

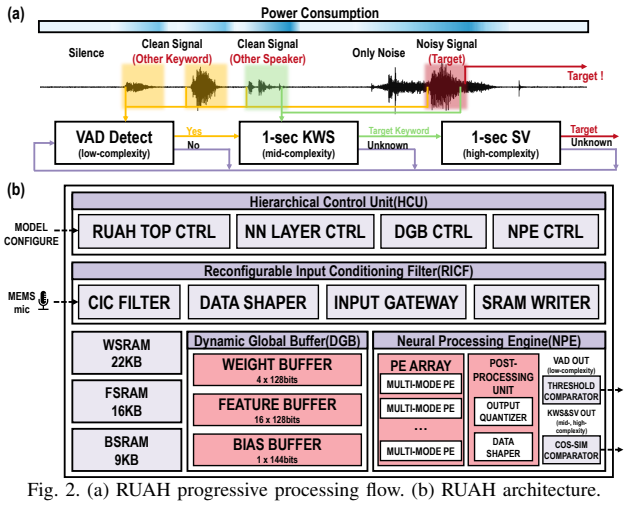

Fig. 2. (a) RUAH progressive processing flow. (b) RUAH architecture.

II. Overall RUAH System

Fig. 2(a) illustrates the execution flow of the proposed RUAH. In the default state, RUAH continuously runs low-

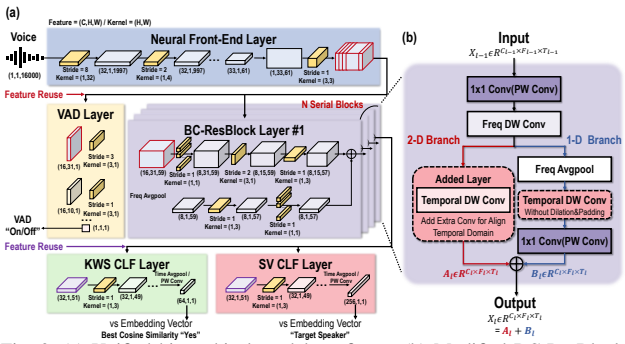

Fig. 3. (a) Unified hierarchical model configure. (b) Modified BC-ResBlock configure.

power VAD at the lowest-complexity stage. When a VAD is detected, the processor selectively extends the inference to deeper stages, such as KWS or SV, reusing shallow front-end features so that inference proceeds hierarchically along the configured task path. After the selected task is completed, RUAH returns to the default low-power state.

Fig. 2(b) shows the top-level architecture. The reconfigurable input conditioning filter (RICF) converts the output of the CIC filter into 8-bit or 16-bit precision for subsequent processing. Based on the configured task mode, the Hierarchical Control Unit (HCU) controls stage activation, precision mode selection, and execution scheduling across the hierarchical inference flow. Under this control, Neural Processing Engine (NPE) performs the neural-network computation for the activated stages, while a Dynamic Global Buffer (DGB) supplies features and weights to the PE array with reuse-aware buffering and data movement.

III. Unified AI Model Configuration

Fig. 3(a) shows the proposed unified AI model. To maximize feature reuse across tasks, VAD, KWS, and SV all share the same front-end (FE), and KWS and SV additionally share the same backbone, followed by their own classifiers (CLF).

A. Neural Front-End and Neural VAD

Fig. 3(a) adopts a shared FE and backbone for multiple tasks. Unlike conventional acoustic systems that rely on FFT, filterbanks, MFCC, and rule-based VAD with separate signal-processing hardware, the proposed system implements both the front-end and VAD as neural networks. This allows all operations to run on a unified neural processing engine without additional dedicated hardware.

In addition, the neural FE and neural VAD are trained for the target tasks and noise conditions, enabling task-aware feature extraction and reducing unnecessary activation of subsequent stages under noise.

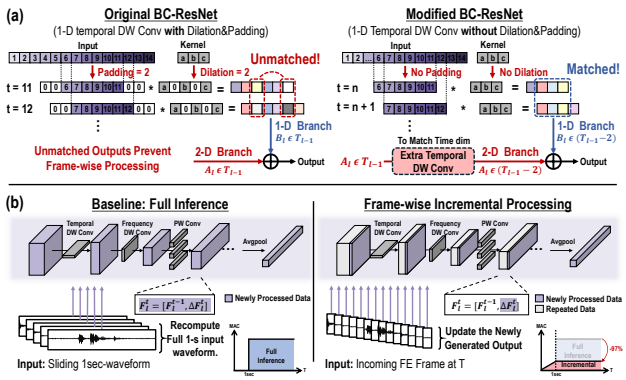

Fig. 4. (a) Limitation of original BC-ResNet and solution. (b) Advantage of frame-wise incremental processing compared with full inference.

B. Modified BC-ResNet for Frame-wise Processing

The original BC-ResNet [11], [12] merges 2-D and 1-D temporal branches in a residual block, but temporal mismatch caused by dilation and padding prevents frame-wise execution. To address these issues, the BC-ResBlock is modified in three aspects, as shown in Fig. 3(b). First, all blocks are unified into a transition-type structure to support consistent frame-wise processing. Second, the 1-D branch eliminates dilation and padding in the temporal convolution to preserve previously computed features, as shown in Fig. 4(a), thereby enabling frame-wise feature reuse. Third, an extra temporal convolution is added to the 2-D branch to match the temporal dimension and frame positions of the 1-D branch at the residual layer.

As illustrated in Fig. 4(b), the modified block supports frame-wise execution, allowing consecutive frames to reuse previously computed features. Moreover, only the minimum feature history required for temporally dependent operations is stored, thereby reducing memory usage.

IV. RUAH Hardware Architecture

A. Reconfigurable Execution Control with HCU

RUAH adopts a configuration-driven HCU, shown in Fig. 5(a), instead of a full instruction-based control architecture. The HCU maps a compact external configuration to cycle-accurate datapath control through four control levels.

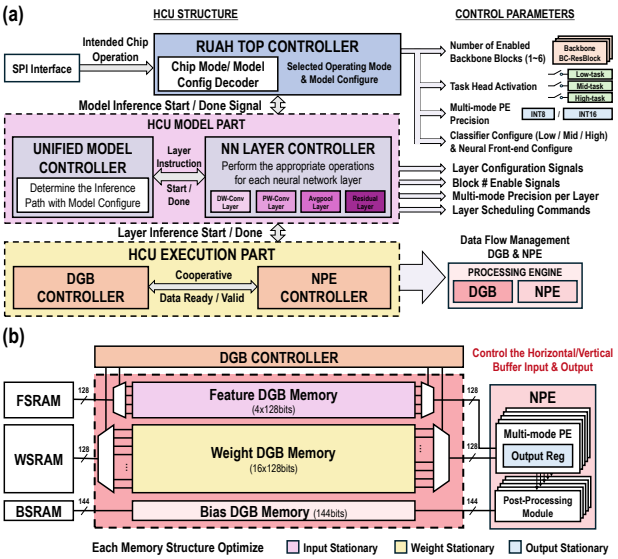

Fig. 5. (a) HCU structure and control parameters. (b) DGB architecture.

At the top level, the RUAH Top Controller receives the model configuration via SPI and determines the active operating mode by disabling unused task heads, controlling frame-wise inference timing, and selecting the execution scope. The configuration is then passed to the Model Controller, which generates the layer execution order, including depthwise convolution, pointwise convolution, average pooling, and residual operations, directly from the configuration without storing per-model schedules. The Neural-Layer Controller then interprets each layer descriptor, defined by the layer type and tensor shape (C, F, T), and computes the read and write base addresses of SRAM at runtime, avoiding hardwired address mappings across model variants.

At the execution level, the DGB and NPE Controllers follow a unified timing schedule generated by the Neural-Layer Controller. The DGB Controller manages data capture and buffer assignment for each layer to maximize data stationarity, while the NPE Controller governs accumulation timing and post-processing handoff. By following the same centralized schedule rather than relying on inter-controller negotiation, the HCU supports low-overhead reconfigurable execution with deterministic dataflow.

B. Data Reuse Optimization with DGB

To reduce memory traffic during hierarchical execution, RUAH employs a DGB that selects the reuse strategy for each layer, as shown in Fig. 5(b). For weights, the DGB leverages the high reuse of convolution by loading all parameters for each output channel and streaming them according to the computation timing. For features, it applies different reuse schemes to DW and PW convolutions: in DW Conv, input features are loaded output-channel-wise and streamed row by row, while overlapping data are retained and only newly required data are loaded by moving the pointer; in PW Conv, one input-channel feature is loaded and streamed in channel order, and all derived output values are computed before loading the next feature. Combined with the output-stationary PE structure, the DGB maximizes feature and weight reuse while minimizing redundant SRAM accesses. As a result, RUAH reduces the total SRAM row accesses to 73.3K, which corresponds to only 0.02×, 0.08×, and 0.44× of those of output, input, weight stationary.

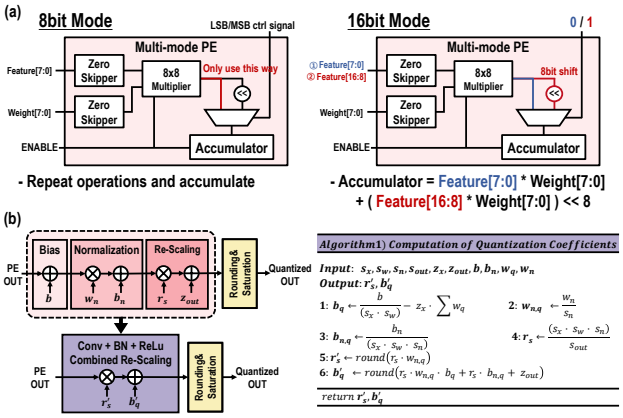

Fig. 6. (a) Multi-mode PE structure & processing Flow. (b) RUAH quantization datapath & Hardware-aware quantization algorithm.

C. Multi-Mode Processing with NPE

The Neural Processing Engine (NPE) implements the core neural-network computation and supports both 8-bit and 16-bit modes, as shown in Fig. 6(a). The PE is built around an 8×8 multiply-accumulate unit with an internal 8-bit shifter. In 8-bit mode, it performs standard MAC operations, whereas in 16-bit mode, the controller decomposes the computation into MSB/LSB-based partial operations and accumulates the shifted results using the same datapath, avoiding a dedicated 16×8 multiplier.

To further reduce unnecessary switching activity, the NPE employs two gating mechanisms: PE deactivation for layers or task stages not selected by the HCU, and zero skipping when either the weight or feature input is zero.

RUAH merges normalization, activation, and rescaling into a single affine transform, eliminating SRAM accesses and computations, as shown in Fig. 6(b). PW layers adopt per-tensor quantization to minimize coefficient storage and SRAM access, while DW layers use per-channel quantization to preserve accuracy.

V. Measurement

The measurement setup consists of an external MEMS microphone (Adafruit evaluation board), a 65nm fabricated chip mounted on a custom PCB, and a Zybo Z7-20 FPGA for system control, data interfacing, and real-time streaming.

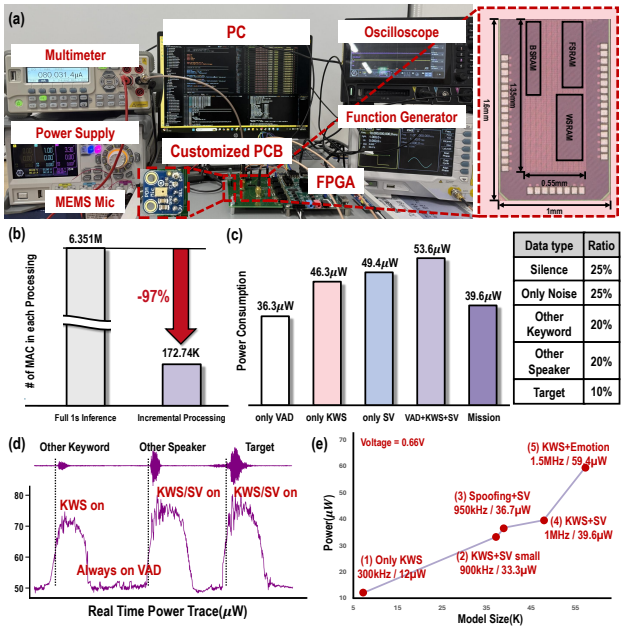

Fig. 7. (a) Measurement setup, (b) Comparative analysis of MAC: full 1-sec inference vs. frame-wise processing, (c) Power efficiency gain through unified hardware integration for VAD, KWS, and SV, with the Input sound-type distribution used for measurement, (d) Real time power trace in RUAH, (e) Power scalability under varying model sizes.

TABLE I. Accuracy of Multi-Task Configurations

Index	Multi-Task	# of Parameter	Test Accuracy					
			Low-complexity	Accuracy	Mid-complexity	Accuracy	High-complexity	Accuracy
1	VAD+KWS	7.2K	VAD	96.5%	KWS(1 class)	92.0%	-	-
2	VAD+KWS+SV(small)	37.3K	VAD	97.2%	KWS	93.2%	SV	95.9%
3	VAD+Spoofing+SV	39K	VAD	96.6%	Spoofing	94.8%	SV	94.0%
4	VAD+KWS+SV(large)	47.8K	VAD	97.6%	KWS	95.2%	SV	96.8%
5	VAD+KWS+Emotion	57.1K	VAD	98.1%	KWS	95.3%	Emotion(1 class)	94.1%

TABLE II. Comparison Table

		JSSC'23 [5]	ISSCC'23 [6]	VLSI'19 [7]	JSSC'25 [8]	TCAS-I'26 [9]	ISSCC'24 [10]	This Work
Technology		28nm	28nm	65nm	28nm	28nm	55nm	**65nm**
Voltage(V)		0.4	-	0.6	0.35/0.9	0.42-0.9	0.84	**0.66-1**
Freq.(Hz)		8k/200k	1M/4M	250k	1M	8k/64k	2.5M	**300k~1.5M**[a]
Data Width (weight/activation)		1bit/1bit	8bit/12bit	4bit or 8bit/8bit	TC1 8bit, others 4bit, 6bit ofmap	4bit/8bit	NN 8b, FE 16b	**8bit/ 8bit or 16bit**
Area(mm²)		0.24	0.8	2.56	0.121	0.37	2.07	**0.74**
Memory(KB)		3.6	18	105	16	9	25.25	**47**
Normalized Power[b]		5.06	-	12.83	8.75	12.61	-	**39.59**
Power(μW)		0.8	1.48	10.6	1.06	2.2	-	**39.59**[c,d]
Energy/Inference(J)		-	-	-	1.73n	-	1.86μ[d]	**0.63μ**[c]
Model		DSCNN	skip-RNN	LSTM/GMM /RBF-SVM	Transfer CNN	BC-ResNet	KWS: TS-DSCNN SV: Lite-X-Vector DoA: EP-DSCNN	**Modified BC-ResNet**
Function	VAD	✓	✗	✓	✓	✗	✓	✓
	KWS	✓	✓	✓	✓	✓	✓	✓
	SV	✗	✗	✓	✓	✓	✓	✓
	Other	✗	✗	✗	✗	✗	✗	✓
Neural Front-end		✗	✗	✗	✓	✓	✗	✓
Neural VAD		✓	✗	✗	✗	✗	✓	✓
Reconfigurability		✓	✗	✓	✗	✗	✓	✓
Dataset		GSCD	GSCD	KWS : GSCD/TIDIGITS SV : TIMIT	KWS, SV: GSCD	GSCD	KWS: GSCD SV: TIMIT / GSCD VAD: TIMIT+NOISEX92	**KWS: GSCD SV: GSCD/ASVspoof Spoofing: ASVspoof Emotion: RAVDESS/CREMA-D**
Accuracy	VAD	0 miss rate (AAD)	-	-	-	-	94%	**97.6%**[c]
	KWS	97.8% (2 keywords)	92.8% (5 keywords)	90.87% (10 keywords)	91.8% (10 keywords)	93.5% (10 keywords)	96.8% (10 keywords)	**95.2%**[c] **(10 keywords)**
	SV	-	-	99.5% (TIMIT)	FAR 1.8%	99.4% - clean(TIMIT)	95.2%	**96.8%**[c]

[a] Frequency range covered by all models in Fig. 7(e).
[b] Normalized power = Power × (0.66 / Voltage)^2 × (65 / Technology)
[c] Result obtained with the configuration in Table I (Index 4).
[d] Evaluated using the sound-type distribution in Fig. 7(c) and [13].

Accuracy was measured with FPGA inputs, while real-time inference was demonstrated using acoustic signals played through a speaker, captured by the MEMS microphone, and delivered to the PCB. For training, the model is optimized in an end-to-end multi-task manner using binary cross-entropy (BCE) for the low-complexity branch and AAM-Softmax for the mid- and high-complexity branches.

Fig. 7 summarizes the measured performance of RUAH. VAD, KWS, and SV were evaluated on GSCD with MUSAN-based noise augmentation at SNRs ranging from 0 to 20 dB, while spoofing and emotion classification were evaluated on ASVspoof and RAVDESS/CREMA-D, respectively. Fig. 7(b) compares computational complexity between full-window processing and the proposed frame-wise processing, showing a 97% reduction in operations. Fig. 7(c) shows the measured power under different execution modes. With the proposed progressive wake-up, the effective average power is reduced to 39.59μW. Fig. 7(d) shows the measured real-time power trace, where the processor stays in always-on VAD mode by default and activates KWS or KWS/SV only when needed, increasing power progressively with task complexity. Fig. 7(e) shows that frequency and power scale with the model configuration, highlighting the flexible performance–power trade-offs of the proposed reconfigurable architecture.

Table I summarizes the measured accuracy across task and model configurations, showing scalable accuracy under a single reconfigurable architecture. Table II compares RUAH with prior acoustic AI and edge inference processors. For a fair comparison, the evaluation uses the typical ratio of different sound types reported in [13], as summarized in the table on the right side of Fig. 7(c). Overall, RUAH achieves a favorable trade-off between functionality, accuracy, and efficiency by supporting multi-task operation with a unified architecture, while maintaining competitive accuracy under a single reconfigurable design.

VI. Conclusion

In this paper, we presented RUAH, a reconfigurable unified acoustic hierarchical AI processor for always-on edge inference. By combining a shared neural front-end, a modified BC-ResNet, and reuse-aware hardware, the proposed system supports multi-task acoustic inference within a single framework. The proposed frame-wise processing reduces computation by 97%, while the unified model enables multi-task execution with minimal power overhead. With progressive wake-up, the effective average power is reduced to 39.59μW. Fabricated in 65-nm CMOS, RUAH achieves 97.6% VAD, 95.2% KWS, and 96.8% SV accuracy, demonstrating an efficient and scalable solution for always-on acoustic processing.

Acknowledgment

This work was supported by NRF funded by MSIT (Nos. RS-2025-07402970, RS-2026-25481418), by IITP funded by MSIT (No. RS-2025-10692981), and by KIAT funded by MOTIE (No. P0023704). The chip fabrication and EDA Tool were supported by the IC Design Education Center.

References

[1] J. Qian et al., "An On-Chip-Training Keyword-Spotting Chip Using Interleaved Pipeline and Computation-in-Memory Cluster in 28-nm CMOS," *IEEE TVLSI*, 2025.
[2] C. Li et al., "A 608nW Near-Microphone Keyword-Spotting Chip Using Real-Point Serial FFT-Based MFCC and Temporal Depthwise Separable CNN in 28nm CMOS," in *IEEE CICC*, 2023.
[3] W. Shan et al., "A 510-nW Wake-Up Keyword-Spotting Chip Using Serial-FFT-Based MFCC and Binarized Depthwise Separable CNN in 28-nm CMOS," *IEEE JSSC*, 2021.
[4] S. Park et al., "A 5.6μW 10-Keyword End-to-End Keyword Spotting System Using Passive-Averaging SAR ADC and Sign-Exponent-Only Layer Fusion with 92.7% Accuracy," in *IEEE Symposium on VLSI Technology and Circuits*, 2024.
[5] W. Shan et al., "AAD-KWS: A Sub-μW Keyword Spotting Chip With an Acoustic Activity Detector Embedded in MFCC and a Tunable Detection Window in 28-nm CMOS," *IEEE JSSC*, 2023.
[6] 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 ISSCC*, 2023.
[7] J. S. P. Giraldo et al., "18μW SoC for Near-Microphone Keyword Spotting and Speaker Verification," in *IEEE Symposium on VLSI Circuits*, 2019.
[8] F. Tan et al., "A 1.8% FAR, 2 ms Decision Latency, 1.73 nJ/Decision Keyword-Spotting (KWS) Chip Incorporating Transfer-Computing Speaker Verification, Hybrid-IF-Domain Computing and Scalable 5T-SRAM," *IEEE JSSC*, 2025.
[9] C. Li et al., "A 2.2-μW Keyword Spotting System With Filterbank Learning and Attention-Aware Soft-Threshold Denoising in 28-nm CMOS," *IEEE TCAS-I*, 2026.
[10] J. Xiao et al., "KASP: A 96.8% 10-Keyword Accuracy and 1.68μJ/Classification Keyword Spotting and Speaker Verification Processor Using Adaptive Beamforming and Progressive Wake-Up," in *IEEE ISSCC*, 2024.
[11] B. Kim, S. Chang, J. Lee, and D. Sung, "Broadcasted Residual Learning for Efficient Keyword Spotting," in *Interspeech*, 2021.
[12] S. Yang, B. Kim, I. Chung, and S. Chang, "Personalized Keyword Spotting through Multi-task Learning," in *Interspeech*, 2022.
[13] J. S. P. Giraldo et al., "Vocell: A 65-nm Speech-Triggered Wake-Up SoC for 10-μW Keyword Spotting and Speaker Verification," *IEEE JSSC*, 2020.