A 110nW Always-on Keyword Spotting Chip using Spiking CNN in 40nm CMOS

Chen Shen*, Junran Pu, Yisheng Chong, Zhongyi Zhang, Wangling Goh,
Bin Zhao, Anh Tuan Do and Yuan Gao††
School of Electrical and Electronic Engineering, Nanyang Technological University,Singapore, Email: *chen.shen@ntu.edu.sg,
IC-Design Department, Institute of Microelectronics (IME), A*STAR, Singapore, Email: ††gaoy@ime.a-star.edu.sg

***Abstract*—This paper presents an ultra-low power keyword spotting (KWS) chip for Artificial Intelligence of Things (AIoT) device's always-on ambient sensing function. The core KWS engine is based on a spiking convolutional neural network (SCNN) model for its attractive features of sparse activation and addition-only operations inside the spiking neurons. The proposed SCNN model improves the existing framewise incremental computation flow by adding a spike processing unit (SPU) to reduce the computing cycles. The power and latency of the whole system are reduced by 16.5% and 43.2% respectively. Extensive network quantization reduces the weight bit-length to 4-bit and only 1-bit activation is required. The chip also supports power gating by an energy-based voice activity detection (VAD) module to further reduce power consumption in random and sparse event (RSE) scenarios. Full chip simulation results show that the chip consumes only 110nW with 2.15% False alarm rate and 3.00% False reject rate in a 10% voice event stream test. It achieves state-of-art recognition accuracy of 99% and 96% for one and two keyword detection tasks.**

***Index Terms*—keyword spotting (KWS), spiking convolutional neural network (SCNN), Sparsity, Zero Skipping, Clock/Power gating**

I. Introduction

Keyword spotting (KWS) is becoming a major choice of human-machine interface for mobile and Artificial Intelligence of Things (AIoT) devices due to its non-contact nature and ease-of-use merit. KWS enables the AIoT device to stay in low-power ambient sensing mode. The power-consuming main sensor block will be activated only when the target keyword is detected by the KWS module. Power consumption, response time, and recognition accuracy are key specifications to evaluate the performance of KWS function.

Artificial Neural Network (ANN) based KWS has achieved the state-of-art recognition accuracy [1] and response time. However, it still requires a relatively large memory footprint and power expensive Multiply and Accumulate (MAC) operations. On the one hand, researchers proposed quantization [2] and pruning methods [3] to compress the ANN model so that it can fit into resource and power-constrained devices [4]. On the other hand, activation sparsity [5] is also explored to reduce MAC operations needed during the inference of ANN-based KWS. To fully avoid MAC operation, brain-inspired Spiking neural network (SNN) emerges. Compared to ANN, activation values can be rate coded into spikes in SNN within certain time frames. This way, the MAC operation is simplified into a series of addition operations and temporal sparsity among spikes can be also explored to reduce computing complexity. In addition, SNN can achieve comparable accuracy to ANN using the state-of-art SNN training algorithm [6], [7].

However, the optimized ANN or SNN model deployed on general-purpose hardware such as FPGA or microcontroller still consumes milliwatt-level power. Several ANN-based KWS chips using Convolutional Neural Network (CNN) topology [8]–[13] or Recurrent Neural Network (RNN) topology [14]–[16] were proposed with power consumption below 144μW. In [8], the framewise incremental computation flow and binarized quantization can even reduce the power of the chip to 510nW for 1 or 2 keyword spotting tasks. With sparsity and addition-only properties, the power consumption of SNN-based KWS chips [17], [18] is further reduced to 378nW and 75-220nW respectively.

In this work, we design a Spiking Convolutional Neural Network (SCNN) based KWS chip. The SCNN model is obtained through an ANN to SNN conversion algorithm [7]. Through co-optimization between the SCNN model and the KWS chip, the chip can avoid power expensive MAC operation, explore activation sparsity, and require minimal on-chip memory. The key contributions of this work are summarized as follows:

1) Compress SCNN model's weight bit-length and spike time steps to ensure the model is compact and addition-only without sacrificing the accuracy.
2) Improve the framewise incremental computation flow by adding SPU to explore the activation sparsity and perform SPU-based clock gating to reduce the latency and power consumption by 43.2% and 16.5%.
3) Add power gating support so that the KWS engine can maintain ultra-low power (110nW) at RSE scenarios.

The rest of this paper is arranged in the following manner. Section II presents the SCNN background theory and domain-specific optimization process. Section III presents the chip architecture and design considerations. The implemented chip performance is shown in section IV.

II. SCNN model for KWS

A. Spiking Neuron versus Artificial Neuron

In this paper, we use an integrate-and-fire (IF) neuron-based SCNN model. The operation of IF neuron is described by (1) and (2).

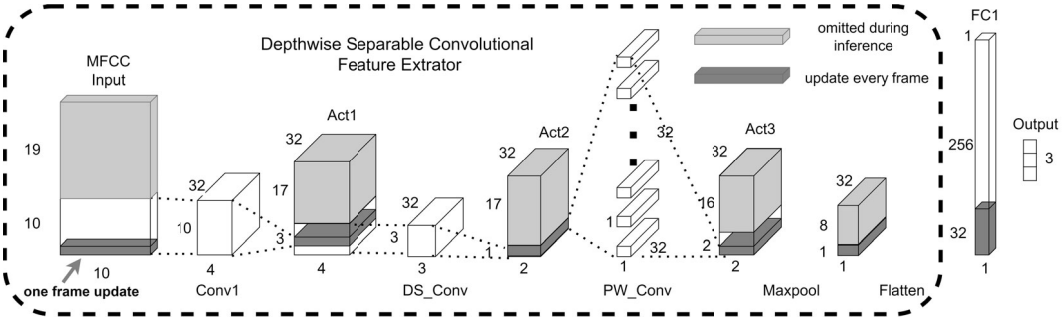

Fig. 1: Topology of neural network and framewise update of neural network during stream inference.

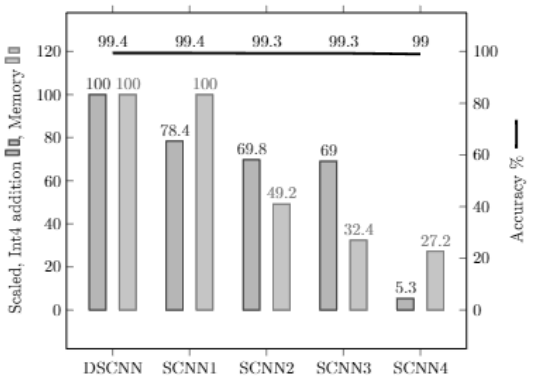

Fig. 2: Comparison of INT4 addition counts, memory footprint and recognition accuracy of different neural network models.

$$\frac{dU_i^l}{dt} = \sum_j w_{ij}^{l-1} s_j^{l-1}[t-1] + b_i^l \tag{1}$$

Equation (1) is the integration stage of IF neuron. U_i^l represents the membrane potential of i^{th} IF neuron in layer 'l'. U_i^l increase or decrease when spikes s_j^{l-1} from previous layer pass through excitatory or inhibitory synapses w_{ij}^{l-1}.

$$s_i^l(t) = \Theta\left(U_i^l(t) - \vartheta\right) \tag{2}$$

At the firing stage shown in (2), IF neuron emits a spike s_j^l if U_i^l is larger than threshold voltage ϑ and ϑ is subtracted from membrane potential. Θ is a Heaviside function.

$$c_i^l = \frac{1}{\vartheta}\rho\left(\sum_j w_{ij}^{l-1} c_j^{l-1} + b_i^l T\right) \tag{3}$$

Under the assumption of evenly distributed spike generation over T time steps [6], an analytical expression (3) of spike count, c_i^l , is deduced by substituting (1) into (2) and performing integration.

$$a_i^l = \rho\left(\sum_j w_{ij}^{l-1} x_j^{l-1} + b_i^l\right) \tag{4}$$

Equation (4) describes the artificial neuron model where a_i^l and x_j^{l-1} are continuous voltage level input and output. ρ is the activation function for both spike neurons and artificial neurons. Comparing (3) and (4), we can see spiking neurons can be approximated as a discrete neural representation of artificial neurons. This means SCNN has the potential for the same performance as quantized ANN with proper training framework [6], [7]. Unlike an artificial neuron, in which activation sparsity comes from zero activation, a spiking neuron has temporal sparsity over T simulation time steps even though its spike count c_i^l is not zero. Besides, only addition is required at both integration stage and firing stage of every simulation time step.

B. SCNN Model Optimization

Due to the functional equivalence between spiking neurons and artificial neurons in Section II-A. we first train an 8-bit quantized, Depthwise Separable Convolutions Neural Network (DSCNN) in [4], [8] and then convert it to the corresponding SCNN model with the ANN to SNN conversion algorithm in [7]. The overall DSCNN and SCNN Network Topology is shown in Fig. 1.

Both models are trained on Google Speech Command Dataset (GSCD) [19]. The 1-second audio clips are further trimmed to 480ms by Montreal Forced Aligner [20] to remove the silent or noisy part. After trimming, 10 MFCC features are extracted every 32ms (16ms stride) with 8-bit precision (4-bit int + 4-bit decimal). For a 480ms audio clip, the size of the 2D MFCC feature map is 29×10. The dataset is split into train set, validation set, and test set with a ratio of 8:1:1. The test set (4000 words) is concatenated with background noise with 20dB SNR to form a 5.3 hours audio clip for stream test.

For one keyword detection, recognition accuracy, memory footprint, and the equivalent number of INT4 addition [21], [22] of DSCNN model and 4 SCNN fine-tuned models are shown in Fig. 2. The following 3 sub-sections will explain the optimization process step by step.

1) Simulation Time Reduction: SCNN1 is obtained from DSCNN model through ANN to SNN conversion. As shown in Fig. 2, DSCNN and SCNN1 have the same recognition accuracy and memory footprint. However, due to temporal sparsity among spiking neurons, SCNN1 can reduce the number of operations by 21.6%. To further reduce memory footprint and

number of operations, we reduce simulation time T from 8 to 1. With a negligible accuracy loss, the memory footprint and number of operations are reduced by 7.6% and 50.8% respectively in SCNN2.

2) Weight Bit-length Reduction: We further quantized the weight of SCNN3 from 8-bit to 4-bit with no accuracy drop. The memory footprint is reduced by 16.8%

3) MAC Removal: SCNN3 is already a compact model with high performance and low power. However, it is not an addition-only model. Convolving 8-bit MFCC features with 4-bit weight in Conv1 layer (Fig. 1) still requires MAC operations. Although we can rate encoding 8-bit MFCC inputs into spikes with T=8 neuron simulation time steps, it will increase the clock cycles of KWS hardware significantly. Therefore, we decide to quantize the weight of Conv1 layer to +1/-1 that the convolution of the first layer only needs addition and can finish computation in one time step. Then, we obtain an addition-only SCNN model with over 99% accuracy. Compared to DSCNN model, SCNN4 saves memory footprint and number of operations by 72.7% and 94.7%.

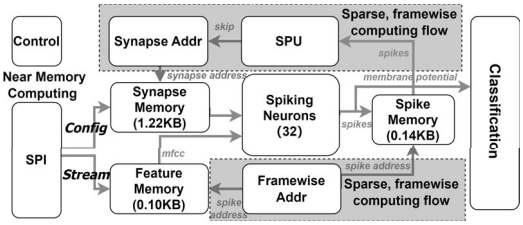

Fig. 3: Architecture of SCNN-based KWS engine

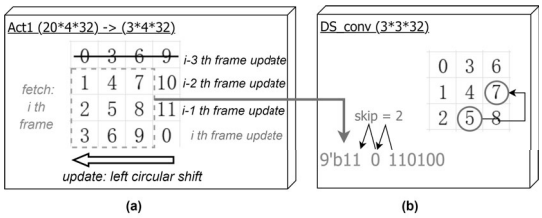

Fig. 4: An illustrative example of sparse, framewise computation flow for DS_Conv layer in i^{th} frame.

III. Hardware Implementation of SCNN-KWS

A. Hardware Architecture

The overall architecture of the SCNN-KWS chip is shown in Fig. 3. In configuration mode, SPI loads the SCNN model into synapse memory. In stream mode, SPI transfer 10, 8-bit, MFCC features to Feature Memory every 16ms. After receiving the MFCC feature, Spiking Neurons fetch data from Synapse Memory or Feature Memory to perform integration and firing operations. The emitted spikes of Spiking Neurons are stored in Spike Memory. At the end of the FC1 layer calculation in Fig. 1, the Classification module directly reads and compares the membrane potential of spiking neurons and outputs the classification result. Compared to the in-memory computing and one-to-one mapping architecture in [18], we adopt the near-memory computing architecture, which reuses spiking neurons and stores emitted spikes in on-chip memory. By adding re-configuration logic, this architecture can support various Convolutional Neural Network topologies without redesigning the whole chip. Furthermore, advanced computation flow can be easily adapted to reduce memory footprint and the number of operations.

B. Sparse, Framewise Computation Flow

When using SCNN in Fig. 1 to classify the 2D MFCC feature map (29×10), only 1×10 data are updated every frame (16ms) during streaming inference. In addition, the intermediate feature maps, Act1, Act2, and Act3 in Depthwise Separable Convolutions Feature Extractor are also partially updated. Most of data in MFCC feature map and Act1-3 are unchanged and redundant. For inference, only FC1 layer is needed to be full size to perform classification. Therefore, the memory height of MFCC input, Act1-3 can be compressed to the kernel height of Conv1, DS_conv, PW_conv, and Maxpool respectively. This way, the memory size is just enough for one frame data update in the feature extractor during stream inference. Every frame's newest update overwrites the oldest frame's data.

Due to this compression, the computation flow requires a customized address generation module, Framewse Addr, to control the fetch and update process of Spike Memory and Feature Memory in Fig. 3. An illustrative example of updating and fetching Act1 in the i^{th} frame is shown Fig. 4 (a). The number represents the row address of spikes in Spike Memory. In i^{th} frame, the generated spikes overwrite the spikes generated in $i-3^{th}$ frame. The update address of i^{th} frame spikes in Spike Memory is the address of $i-3^{th}$ frame update address left circular shifted. After the update, the spikes are fetched sequentially starting from 1. If the row address exceeds the upper bound, 11, the row address jump back to 0.

However, this computation flow does not explore the sparsity of the SCNN model. Therefore, in Fig. 3, we propose to add an SPU module [23] to detect the spatial difference between two spiked neurons. An illustrative example is shown in Fig. 4 (b). Based on the fetch address generated by Framewise Addr, a 9-bit spiking vector from Spike Memory is passed to SPU to decide which synapse weight should be fetched in Synapse Memory and passed to Spiking Neurons. As shown in Fig. 4 (b), there is a non-spiking neuron between 5^{th} bit and 7^{th} bit. SPU detects the spatial difference and generates the control signal "skip = 2". Synapse Addr adds "skip" to the previous weight address, 5, and generates the next weight address, 7, for Synapse Memory to fetch the weights of spiked neurons in the SCNN model.

C. Clock gating & Power gating

The SCNN-KWS chip is designed to work at 40KHz, equals to 640 clock cycles every frame (16ms). The proposed design requires at most 563 clock cycles for inference every frame.

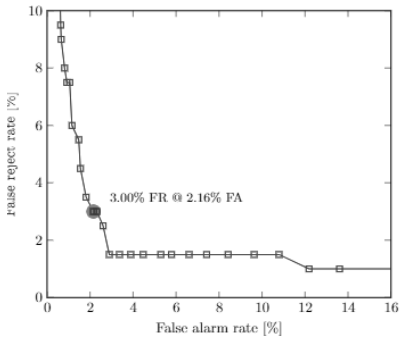

Fig. 5: ROC curve of SCNN-KWS for one keyword spotting.

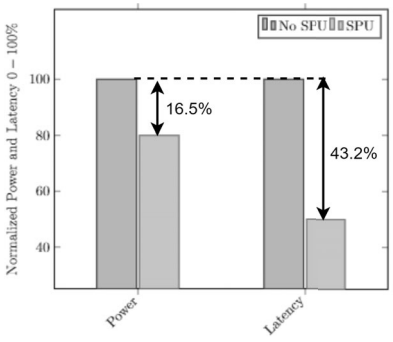

Fig. 6: Compare latency and power between design with and without SPU.

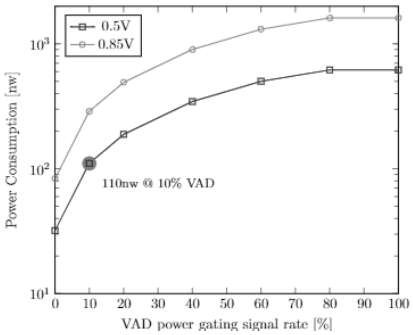

Fig. 7: Power consumption of SCNN-KWS @40KHz.

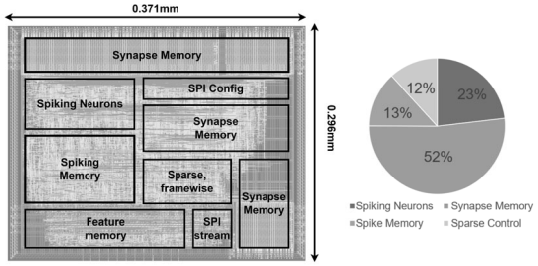

Fig. 8: Layout photo and power breakown of SCNN_KWS.

TABLE I: Performance comparison with the state-of-the-arts

	[24]	[8]	[18]	This Work
Tech (nm)	65	28	65	40
Voltage (V)	0.575	0.41	0.5	0.5
Freq (KHz)	250	40	70	40
Size (mm2)	1.035	0.23	1.99	0.11
Mem (KB)	32	0.65	8.375	1.56
latency (ms)	16ms	16ms	NA	8ms
Power (nW)	5000	173.4	75-220*	110–617
Model	LSTM	DSCNN	SCNN	SCNN
Dataset	TIMIT	GSCD	Heysnips	GSCD
Keyword	1	1 or 2	1	1 or 2
Accuracy	92%	98% or 94.6%	95.8%	99% or 96%

However, with the help of SPU to explore spiking neuron sparsity. The average clock cycle is around 320 every frame. The latency of the SCNN module is hence reduced by 43.2% in Fig. 6. Based on this finding, we propose to clock gate the whole SCNN chip except the Control module to reduce power consumption. This SPU-enabled clock gating reduces the power by 16.5% in Fig. 6.

The design also supports power gating by an energy-based Voice Activity Detection (VAD) module. When no voice activity happens, the power supply of the KWS engine is switched off to save power. In order to quickly recover from shutdown to normal stream mode, re-load Feature Memory by SPI module in configuration mode must be avoided. Therefore, Synapse Memory has separate power supply nets and goes into retention mode when the KWS engine is power gated. As shown in Fig. 7, the power consumption of the design ranges from 32nW to 617nW with 0-100% VAD power gating rate.

IV. IMPELMENTATION RESULTS

The chip is implemented in UMC 40nm CMOS technology with a core size of 0.371mm $\times$ 0.296mm. The operating frequency and voltage are 40KHz and 0.5V.

For power estimation, we use Cadence Joules to measure the average power consumption of the chip in stream mode and scale this power with actual VAD signal rate generated.

The receiver operating characteristic (ROC) curve of the chip is shown in Fig. 5. The model can achieve a 3.00% False reject rate at 2.16% False alarm rate for a 5.3 hours audio stream test. The result considered the keywords or the filler words falsely rejected by the energy-based VAD.

Table I compares our design with the state-of-art KWS engine for 1 or 2 keyword spotting tasks. The average power consumption in a 10% voice activity stream test is 110nW in Fig. 7, which is lower than 173.4nW in [24], 5μW in [24]. Although the maximum power consumption of 617nW is higher than 220nW in [18], the one keyword spotting accuracy of our work is significantly higher than the accuracy of [18]. Meanwhile, with the help of SPU, the latency of our design is 43.2% less than [8] and [24].

V. CONCLUSION

In this paper, we present a sub-700nW, SCNN-based KWS engine for 1 or 2 keyword spotting. The customized SCNN model achieves the state-of-art recognition accuracy (99% or 96%). The chip support VAD-based power gating and SPU-based clock gating to achieve an ultra-low power consumption (110nW) at a Random and Sparse voice event working environment. The near-memory computation architecture with sparse, framewise computation flow minimizes the memory footprint (1.56KB) and latency (8ms) of the KWS engine.

VI. ACKNOWLEDGEMENT

This work was supported by the Agency for Science, Technology and Research (A*STAR), Singapore under the Nanosystems at the Edge Programme, grant no. A18A1b0055.

REFERENCES

[1] G. Chen, C. Parada, and G. Heigold, "Small-footprint keyword spotting using deep neural networks," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4087–4091, 2014.

[2] M. Nagel, M. Fournarakis, R. A. Amjad, Y. Bondarenko, M. van Baalen, and T. Blankevoort, "A white paper on neural network quantization," *arXiv preprint arXiv:2106.08295*, 2021.

[3] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *arXiv preprint arXiv:1510.00149*, 2015.

[4] Y. Zhang, N. Suda, L. Lai, and V. Chandra, "Hello edge: Keyword spotting on microcontrollers," *arXiv preprint arXiv:1711.07128*, 2017.

[5] S. Han, X. Liu, H. Mao, J. Pu, A. Pedram, M. A. Horowitz, and W. J. Dally, "Eie: Efficient inference engine on compressed deep neural network," 2 2016.

[6] J. Wu, Y. Chua, M. Zhang, G. Li, H. Li, and K. C. Tan, "A tandem learning rule for effective training and rapid inference of deep spiking neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[7] C. Stöckl and W. Maass, "Optimized spiking neurons can classify images with high accuracy through temporal coding with two spikes," *Nature Machine Intelligence*, vol. 3, no. 3, pp. 230–238, 2021.

[8] W. Shan, M. Yang, T. Wang, Y. Lu, H. Cai, L. Zhu, J. Xu, C. Wu, L. Shi, and J. Yang, "A 510-nw wake-up keyword-spotting chip using serial-fft-based mfcc and binarized depthwise separable cnn in 28-nm cmos," *IEEE Journal of Solid-State Circuits*, vol. 56, no. 1, pp. 151–164, 2020.

[9] B. Liu, H. Cai, Z. Wang, Y. Sun, Z. Shen, W. Zhu, Y. Li, Y. Gong, W. Ge, J. Yang, and L. Shi, "A 22nm, 10.8 w/15.1 w dual computing modes high power-performance-area efficiency domained background noise aware keyword- spotting processor," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 67, no. 12, pp. 4733–4746, 2020.

[10] B. Liu, Z. Wang, H. Fan, J. Yang, B. Liu, W. Zhu, L. Huang, Y. Gong, W. Ge, and L. Shi, "Eera-kws: A 163 tops/w always-on keyword spotting accelerator in 28nm cmos using binary weight network and precision self-adaptive approximate computing," *IEEE Access*, vol. 7, pp. 82453–82465, 2019.

[11] P. P. Bernardo, C. Gerum, A. Frischknecht, K. Lübeck, and O. Bringmann, "Ultratrail: A configurable ultralow-power tc-resnet ai accelerator for efficient keyword spotting," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 11, pp. 4240–4251, 2020.

[12] M. Wang and A. P. Chandrakasan, "Flexible low power cnn accelerator for edge computing with weight tuning," in *2019 IEEE Asian Solid-State Circuits Conference (A-SSCC)*, pp. 209–212, 2019.

[13] S. Yin, P. Ouyang, S. Zheng, D. Song, X. Li, L. Liu, and S. Wei, "A 141 uw, 2.46 pj/neuron binarized convolutional neural network based self-learning speech recognition processor in 28nm cmos," in *2018 IEEE Symposium on VLSI Circuits*, pp. 139–140, IEEE, 2018.

[14] Q. Li, S. Lin, C. Liu, Y. Liu, F. Qiao, Y. Wang, and H. Yang, "Ns-kws: Joint optimization of near-sensor processing architecture and low-precision gru for always-on keyword spotting," in *Proceedings of the ACM/IEEE International Symposium on Low Power Electronics and Design*, pp. 97–102, 2020.

[15] J. S. P. Giraldo, S. Lauwereins, K. Badami, and M. Verhelst, "Vocell: A 65-nm speech-triggered wake-up soc for 10- μ w keyword spotting and speaker verification," *IEEE Journal of Solid-State Circuits*, vol. 55, no. 4, pp. 868–878, 2020.

[16] Y. S. Chong, W. L. Goh, V. P. Nambiar, and A. T. Do, "A 2.5 w kws engine with pruned lstm and embedded mfcc for iot applications," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 69, no. 3, pp. 1662–1666, 2022.

[17] Z. Wang, L. Ye, Y. Liu, P. Zhou, Z. Tan, H. Fan, Y. Zhang, J. Ru, Y. Wang, and R. Huang, "12.1 a 148nw general-purpose event-driven intelligent wake-up chip for aiot devices using asynchronous spike-based feature extractor and convolutional neural network," in *2021 IEEE International Solid- State Circuits Conference (ISSCC)*, vol. 64, pp. 436–438, 2021.

[18] D. Wang, P. K. Chundi, S. J. Kim, M. Yang, J. P. Cerqueira, J. Kang, S. Jung, S. Kim, and M. Seok, "Always-on, sub-300-nw, event-driven spiking neural network based on spike-driven clock-generation and clock-and power-gating for an ultra-low-power intelligent device," in *2020 IEEE Asian Solid-State Circuits Conference (A-SSCC)*, pp. 1–4, IEEE, 2020.

[19] P. Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," *arXiv preprint arXiv:1804.03209*, 2018.

[20] M. McAuliffe, M. Socolof, S. Mihuc, M. Wagner, and M. Sonderegger, "Montreal forced aligner: Trainable text-speech alignment using kaldi.," in *Interspeech*, vol. 2017, pp. 498–502, 2017.

[21] M. Horowitz, "1.1 computing's energy problem (and what we can do about it)," in *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, pp. 10–14, IEEE, 2014.

[22] M. Ghadiry, M. Nadi, and A. K. A'Ain, "Dlpa: Discrepant low pdp 8-bit adder," *Circuits, Systems, and Signal Processing*, vol. 32, no. 1, pp. 1–14, 2013.

[23] J. Pu, W. L. Goh, V. P. Nambiar, M. M. Wong, and A. T. Do, "A 5.28-mm² 4.5-pj/sop energy-efficient spiking neural network hardware with reconfigurable high processing speed neuron core and congestion-aware router," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 68, pp. 5081–5094, 12 2021.

[24] J. S. P. Giraldo and M. Verhelst, "Laika: A 5uw programmable lstm accelerator for always-on keyword spotting in 65nm cmos," in *ESSCIRC 2018-IEEE 44th European Solid State Circuits Conference (ESSCIRC)*, pp. 166–169, IEEE, 2018.