This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

RS-LPU: A ReRAM-on-Logic Stacked Large Language Model Accelerator With Software–Hardware Co-Designed Weight Compression and Speculative Decoding

Pingcheng Dong, *Graduate Student Member, IEEE*, Yonghao Tan, *Graduate Student Member, IEEE*, Xuejiao Liu, Peng Luo, Yu Liu, Di Pang, Fengshi Tian, *Graduate Student Member, IEEE*, Songchen Ma, *Member, IEEE*, Xijie Huang, Shih-Yang Liu, Dong Zhang, Zhichao Lu, *Member, IEEE*, Luhong Liang, *Senior Member, IEEE*, Chi-Ying Tsui, *Senior Member, IEEE*, Fengbin Tu, *Senior Member, IEEE*, Liang Zhao, *Member, IEEE*, and Kwang-Ting Cheng, *Fellow, IEEE*

***Abstract*—Large language models (LLMs) have demonstrated remarkable performance in the field of artificial intelligence (AI). However, their autoregressive decoding (AD) paradigm incurs heavy latency overhead from massive weight external memory access (EMA) on edge accelerators. Speculative decoding (SD) has recently been proposed to alleviate this issue by offloading AD to a small draft LLM (DLM), with its multiple output tokens verified by the original target LLM (TLM) in parallel. However, SD is still bottlenecked on edge accelerators by weight EMA stemming from both the TLM and the DLM. First, activation outliers hinder low-bit TLM quantization for EMA reduction while existing rotation-based outlier-free methods incur heavy area overhead. Second, the DLM EMA is amplified by autoregressive drafting as the limited on-chip buffer cannot store the whole DLM, while a high rejection ratio at a long draft length wastes substantial latency that outweighs the TLM EMA savings. This article introduces RS-LPU, a resistive random access memory (ReRAM)-on-logic stacked LLM accelerator with several software–hardware co-design techniques to tackle these issues. RS-LPU reduces the TLM EMA by approximating the outlier-free global rotation with two overlapped local rotations, and supports reconfigurable rotation at minimal area by exploiting locality similarity and dataflow optimization. Furthermore, RS-LPU leverages the high bandwidth and density of ReRAM-on-logic stacking with blockwise weight clustering to compress the DLM into the stacked ReRAM dies, with multi-chip extension to store the DLM backbone on-chip. In addition, RS-LPU features a workload-decoupled out-of-order scheduler (WDOS) co-optimized with the compiler to support an adaptive parallel speculative decoding (APSD) scheme proposed in this work, which jointly hides drafting latency at a long draft length and delivers high accepted token yield without resource competition. With these features, on the representative LLaMA2-7B model, RS-LPU achieves 17.82 token/s decoding throughput at 123.41 mJ/token energy consumption on the MT-Bench dataset.**

***Index Terms*—Co-design, large language model (LLM) accelerator, resistive random access memory (ReRAM)-on-logic stacking, speculative decoding (SD), Transformer.**

I. Introduction

RECENTLY, Large language models (LLMs) [1] exhibit exceptional performance through massive parameters and large-scale pre-training, and have been widely adopted in various fields of artificial intelligence (AI), including Chatbot [1], AI Agent [2], and Embodied AI [3]. However, the autoregressive decoding (AD) paradigm adopted by LLMs introduces heavy latency overheads. As shown in Fig. 1(a), AD generates tokens one-by-one, and each token entails loading the same LLM weights and the KV cache from off-chip memory, resulting in heavy external memory access (EMA) [4], [5], [6], [7], [8]. Specifically, each token of LLaMA2-7B requires 6.28 GB of weight EMA, while the KV cache EMA reaches up to 0.5 GB under 2 K sequence length at decoding, where the weight EMA occupies over 80% of the latency obtained from our simulation with an LPDDR3 interface. Therefore, LLM with AD is fundamentally memory-bound, with weight EMA as the dominant bottleneck.

To address this, speculative decoding (SD) [9], [10] is proposed by transferring partial EMA from the large-scale target LLM (TLM) to the small-scale draft LLM (DLM) as shown in Fig. 1(b). In SD, the DLM first autoregressively generates γ draft tokens, where γ is a predefined draft length.

Received 30 April 2026; revised 1 July 2026; accepted 20 July 2026. This article was approved by Associate Editor Zhengya Zhang. This work was supported in part by I Chip Center for Emerging Smart Systems (ACCESS)–Artificial Intelligence (AI) Chip Center for Emerging Smart Systems, Sponsored by InnoHK, Hong Kong, SAR; in part by the National Key Research and Development Program of China under Grant 2024YFE0201100; and in part by HKSAR Research Grants Council (RGC) General Research Fund (GRF) under Grant 16208823. *(Corresponding authors: Liang Zhao; Kwang-Ting Cheng.)*

Pingcheng Dong, Yonghao Tan, Fengshi Tian, Songchen Ma, Dong Zhang, Chi-Ying Tsui, Fengbin Tu, and Kwang-Ting Cheng are with The Hong Kong University of Science and Technology, Hong Kong, China, and also with the AI Chip Center for Emerging Smart Systems (ACCESS), Hong Kong, China (e-mail: timcheng@ust.hk).

Xuejiao Liu, Peng Luo, Yu Liu, Di Pang, and Luhong Liang are with the AI Chip Center for Emerging Smart Systems (ACCESS), Hong Kong, China.

Xijie Huang and Shih-Yang Liu are with The Hong Kong University of Science and Technology, Hong Kong, China.

Zhichao Lu is with Hefei Reliance Memory Ltd., Hefei 230088, China.

Liang Zhao is with Zhejiang University, Hangzhou 310027, China (e-mail: lzhao2020@zju.edu.cn).

Digital Object Identifier 10.1109/JSSC.2026.3718555

0018-9200 © 2026 IEEE. All rights reserved, including rights for text and data mining, and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on November 03,2026 at 12:36:40 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

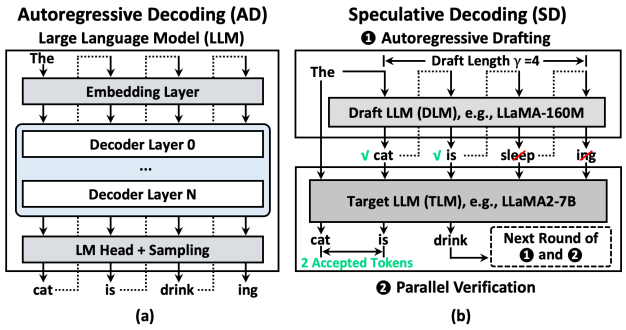


Fig. 1. Two decoding paradigms in LLM. (a) AD. The output token is decoded one-by-one with a full inference of the TLM per token. (b) SD. The DLM first decodes γ draft tokens autoregressively, and then the TLM verifies all of them in a single parallel inference.

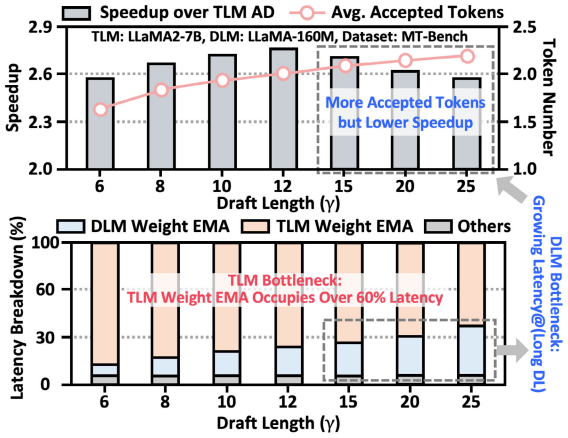


Fig. 2. SD profiling on an edge accelerator (TLM: LLaMA2-7B, DLM: LLaMA-160M, dataset: MT-Bench [11]). (Top) Speedup over TLM AD and average accepted tokens versus draft length γ. Although more tokens are accepted at longer γ, the speedup saturates and eventually decreases. (Bottom) Latency breakdown showing that TLM weight EMA occupies over 60% of the total latency, while DLM weight EMA grows with γ and eventually offsets the speedup gain from accepting more tokens.

Then, the TLM verifies all γ draft tokens in a single parallel inference rather than γ rounds of AD, and accepts the draft tokens whose distributions match those of the TLM. As proven by Leviathan et al. [9], speculative sampling guarantees an identical output distribution to the TLM's original AD for any choice of DLM, making SD a mathematically lossless acceleration technique that achieves 2×–3× wall-time speedup on GPUs [9], [10].

However, on resource-constrained edge accelerators, the LLM latency under SD is still dominated by weight EMA due to their limited external memory bandwidth. analyzed in Fig. 2, over 60% of the latency stems from the TLM, while the DLM weightEMA also becomes non-negligible due to limited on-chip SRAM and is further exacerbated atlong γ [7]. Therefore, the TLM and DLM weight EMA synergistically become the bottleneck of SD on edge accelerators, raising three key challenges. First, low-bit post-training quantization (PTQ) [12] is the most practical and mainstream approach to reduce TLM weight EMA, as quantization-aware training (QAT) is prohibitive for billion-parameter models. However, LLM activations exhibit a few high-magnitude outlier channels that stretch the quantization range [13], [14], [15]. During integer-only matrix multiplication, quantization errors are amplified by these outlier activations, and the amplification becomes more pronounced at lower weight bit-widths due to larger quantization step sizes. For example, W4A8 PTQ with GPTQ [12] on LLaMA2-7B degrades the Wikitext-2 perplexity (PPL) and zero-shot reasoning accuracy from 5.5 PPL/66.9% to 20.2 PPL/64.8%, whereas removing these outliers by rotation-based methods [14], [15] recovers them to 5.7 PPL/65.8% under the same W4A8 precision. Nevertheless, supporting reconfigurable global rotation on-chip incurs heavy area overhead due to numerous high-precision operators and registers [7]. Groupwise and microscaled quantization provide an alternative rotation-free path to suppress these outliers, yet they match the accuracy of rotation only at small group sizes and require additional co-design to mitigate the extra area and power overheads due to frequent high-precision dequantization [16]. Second, although the DLM is much smaller than the TLM, the limited on-chip SRAM capacity of edge accelerators (24 KB–3.34 MB [4], [8], [17], [18], [19], [20]) still cannot buffer whole DLM weights. As a result, the DLM inference still remains bounded by the external memory bandwidth. Moreover, SD requires γ rounds of autoregressive drafting on the DLM, each reloading its weights from off-chip memory and thereby amplifying the DLM weight EMA, especially at longer γ, as shown in Fig. 2. Third, despite the increase in average accepted tokens per SD iteration at longer γ, the overall SD speedup saturates and even decreases. The reason is that over 90% of the draft tokens are rejected, and the verification cost on them is entirely wasted, outweighing the TLM EMA savings. Hence, SD is generally restricted to a short draft length, limiting potential speedup.

This article proposes RS-LPU, a resistive random access memory (ReRAM)-on-logic stacked LLM accelerator [7] with several software–hardware co-design strategies, addressing the dual weight EMA bottleneck from both TLM and DLM for energy-efficient SD on edge accelerators. The main contributions are as follows.

1) A local rotation unit (LRU) removes activation outliers for accurate W4A8 TLM quantization. It is reconfigurable to support arbitrary LLM channel dimensions by decomposing the deep Fast Walsh-Hadamard Transform (FWHT) into overlapped upper and lower low-cost FWHTs, attaining 3.82×–3.93× speedup over vanilla SD with 92.7% rotation engine area savings.
2) A ReRAM-stacked process-near-memory (RS-PNM) architecture with blockwise vector quantization (BVQ) alleviates DLM weight EMA by clustering weights into block-level codebooks (CBs) and compact indices. A four-chip system further scales the on-chip capacity and bandwidth to 32 MB and 102.4 GB/s, respectively, achieving 1.1×–1.46× speedup over W4A8 SD with LRU.
3) An adaptive parallel SD (APSD) scheme dynamically switches between drafting-only and parallel draft-and-verify modes based on TLM verification feedback, and a workload-decoupled out-of-order scheduler (WDOS) further enables long γ for higher accepted token yield while hiding the long γ drafting latency by TLM

Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on November 03,2026 at 12:36:40 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

verification without resource competition. The APSD with WDOS attains 1.1×–1.29× speedup and 10%–14% rejected token ratio reduction.

The rest of this article is organized as follows. Section II introduces the background and motivation. Section III presents the overall RS-LPU architecture. Sections IV–VI detail the LRU with decomposed FWHT, RS-PNM with BVQ, and APSD with WDOS, respectively. Section VII reports measurement results and comparison. Section IX concludes this article.

II. Background and Motivation

A. Quantitative Speedup Analysis of SD

As shown in Fig. 1(b), SD [9] accelerates AD by introducing a small-scale DLM alongside the TLM. The DLM first autoregressively generates γ draft tokens, and then the TLM verifies all of them in a single parallel inference through *speculative sampling* [9], which provably preserves the TLM's output distribution. Suppose t_{TLM} and t_{DLM} are the single-batch latency of the TLM and DLM, r is the expected token rejection ratio, and $n_{\text{a}} = (1 - r)\gamma$ is the expected number of accepted draft tokens per SD iteration. Along with one token that the TLM inherently samples during verification, each SD iteration produces $n_{\text{a}} + 1$ tokens in total. The expected speedup of SD over AD is therefore,

$$S = \frac{(n_{\text{a}} + 1) \cdot t_{\text{TLM}}}{\gamma \cdot t_{\text{DLM}} + t_{\text{TLM}}} \tag{1}$$

since both the TLM and DLM are bounded by weight EMA on edge accelerators, improving S requires jointly optimizing four factors in (1). This motivates us to optimize TLM and DLM weight EMA to reduce t_{TLM} and t_{DLM}, where the latter is relatively marginal but heavily amplified by γ, and to enhance the accepted token yield n_{a} while avoiding the diminishing returns at large γ as discussed in the third challenge.

B. Rotation-Based Quantization and Its Overheads

Recent PTQ methods [14], [15] leverage rotation to achieve outlier-free low-bit weight quantization. The rotation is built on Hadamard matrix $\mathbf{H}_n$, which is an $n \times n$ orthogonal matrix with entries drawn from $\{+1, -1\}$, satisfying $\mathbf{H}_n\mathbf{H}_n^\top = \mathbf{I}$. By incorporating $\mathbf{H}_n\mathbf{H}_n^\top$ into a general matrix multiplication (GEMM) $\mathbf{X}\mathbf{W}^\top$, the computation can be reformulated as

$$\mathbf{X}\mathbf{W}^\top = (\mathbf{X}\mathbf{H}_n)\left(\mathbf{H}_n^\top\mathbf{W}^\top\right). \tag{2}$$

This reformulation indicates that the orthogonality of $\mathbf{H}_n$ achieves computational invariance, where $\mathbf{H}_n^\top\mathbf{W}^\top$ can be offline merged into the weight matrix before deployment, and $\mathbf{X}\mathbf{H}_n$ is executed online to rotate the activation into an outlier-free distribution before quantization. For power-of-two (pot) dimensions $d = 2^k$, the normalized Hadamard matrix $\mathbf{H}_{2^k}$ is constructed recursively through the Kronecker product $\otimes$ as

$$\mathbf{H}_{2^k} = \mathbf{H}_2 \otimes \mathbf{H}_{2^{k-1}} = \frac{1}{\sqrt{2}}\begin{bmatrix} \mathbf{H}_{2^{k-1}} & \mathbf{H}_{2^{k-1}} \\ \mathbf{H}_{2^{k-1}} & -\mathbf{H}_{2^{k-1}} \end{bmatrix} \tag{3}$$

where $\mathbf{H}_2 = (1/(2)^{1/2})\left[\begin{smallmatrix} 1 & 1 \\ 1 & -1 \end{smallmatrix}\right]$. However, modern LLMs often contain non-power-of-two (npot) channel dimensions. For example, LLaMA3-8B adopts $d = 14\,336$ for its feed-forward network intermediate dimension. To handle such cases, d is factorized into $d = 2^k \times m$ (e.g., $14\,336 = 2^9 \times 28$), where m is the size of a pre-computed npot Hadamard matrix [21], and the npot Hadamard matrix is formed as

$$\mathbf{H}_d = \mathbf{H}_{2^k} \otimes \mathbf{H}_m. \tag{4}$$

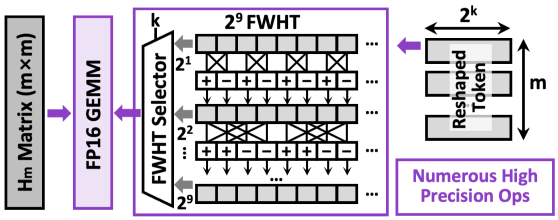


Fig. 3. Cascaded FWHT-GEMM for npot Hadamard rotation.

To minimize the additional computation and EMA overhead introduced by the online rotation $\mathbf{X}\mathbf{H}_n$, the FWHT is generally utilized to decompose $\mathbf{H}_d$ into k stages of butterfly operations and a low-cost GEMM with $\mathbf{H}_m$, as illustrated in Fig. 3, reducing the complexity from $\mathcal{O}(d^2)$ to $\mathcal{O}(d \log_2 d)$. However, to support the diverse channel dimensions across different TLMs, the FWHT-GEMM array must be equipped with a substantial depth, which introduces numerous high-precision operators and registers, occupying nearly 4.37× the area of a 4 K INT8 MAC array as estimated from our synthesis results [7].

C. Bumping-Based ReRAM-on-Logic Stacking

Another approach to overcoming the weight EMA bottleneck is 3-D stacking, which provides substantially higher memory-to-logic bandwidth than conventional off-chip interfaces. Among various 3-D integration technologies, DRAM hybrid bonding has been increasingly adopted for LLM acceleration [22], [23], [24], achieving 12.77 GB/s/mm^2 bandwidth density [22], over 16 TB/s per-chip bandwidth [23], and 0.88 pJ/bit memory access energy [24]. However, DRAM hybrid-bonding integration requires a large logic die area to match the DRAM bank parallelism, leading to high power consumption that reaches up to 76.8 W in [23].

In contrast, ReRAM is non-volatile and consumes lower power than DRAM without periodic refresh while achieving over 10× higher density than SRAM [25] for larger on-chip memory capacity. On the other hand, SD spreads weight EMA across the TLM and a much smaller DLM, where the DLM weights do not demand multi-gigabyte DRAM capacity, allowing ReRAM capacity to well match the DLM weight footprint. Through low-cost bumping-based face-to-face 3-D stacking, ReRAM-on-logic integration delivers high bandwidth and density. Co-designed with aggressive weight compression shown in Fig. 4 and multi-chip interconnection, the DLM backbone weights are available to be stored on-chip, substantially reducing t_{DLM} and unlocking the opportunity to adopt longer draft lengths for higher accepted token yield [7].

Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on November 03,2026 at 12:36:40 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

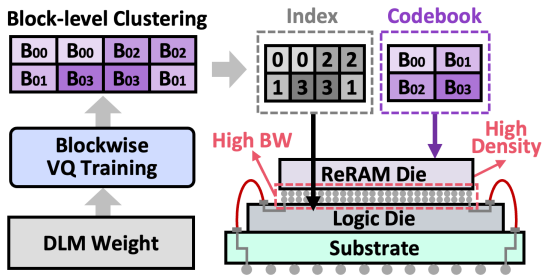


Fig. 4. RS-PNM architecture with BVQ.

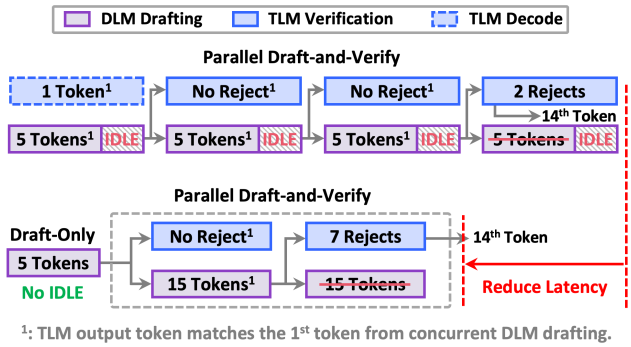


Fig. 5. (Upper) Parallel SD with a short draft length (γ = 5). The DLM finishes drafting before the TLM completes verification, resulting in DLM idle periods in each iteration. (Lower) Proposed APSD dynamically switches between draft-only and parallel draft-and-verify based on TLM verification feedback, combining the high acceptance of short draft length with the high token yield of long draft length while eliminating DLM idle periods.

D. Parallel SD

As illustrated in Fig. 1(b), standard SD executes the DLM drafting and TLM verification sequentially, where the TLM remains idle during γ rounds of DLM drafting, and the DLM stays idle during TLM verification. This mutual waiting problem [26] severely underutilizes the hardware resources. Parallel SD [26] addresses this by overlapping the drafting and verification phases shown in the upper part of Fig. 5, achieving up to 4.43× and 1.50× speedup over AD and vanilla SD on cloud GPUs, respectively. In practice, parallel SD typically adopts a short γ, since a smaller γ yields a higher probability of all-acceptance in TLM verification [27]. However, such a short drafting window forces the DLM to finish well before the TLM completes verification, leaving it idle for the remainder of each iteration. To address this, we propose APSD [7] as shown in the lower part of Fig. 5, which dynamically switches between draft-only with a short γ and parallel draft-and-verify with a long γ based on TLM verification feedback, alleviating the DLM idle issue while improving the accepted token yield. Moreover, since cloud GPUs offer abundant memory bandwidth and computation resources, existing parallel SD methods, such as PEARL [26], avoid resource competition through inter-chip parallelism that isolates the DLM and TLM on separate chips. However, on edge accelerators with severely constrained bandwidth and computation resources, this multi-chip isolation wastes both. Therefore, we adopt intra-chip parallelism that serves the DLM and TLM concurrently at each chip, and resolve resource competition with an out-of-order scheduler, WDOS. The detailed APSD scheme with WDOS is presented in Section VI.

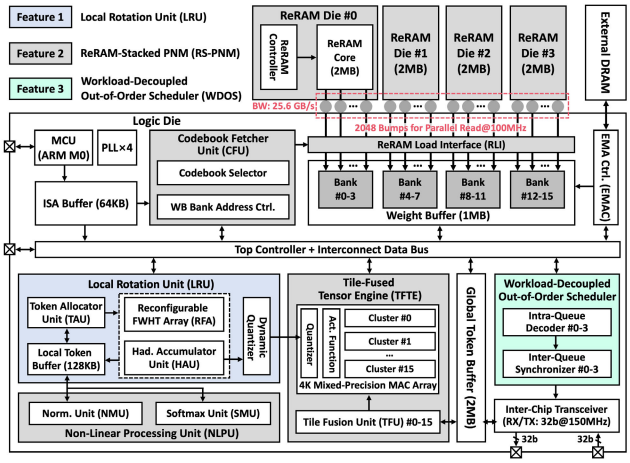


Fig. 6. Overall architecture of the proposed RS-LPU with three main features.

III. Overall Architecture

Fig. 6 shows the overall architecture of the proposed 55 nm RS-LPU, which adopts bumping-based ReRAM-on-logic stacking technology. The ReRAM dies are first diced from a full wafer and then bonded face-to-face onto the logic wafer via bumping, which is subsequently diced into individual chips. In RS-LPU, four ReRAM dies are stacked on each logic die via 2912 face-to-face bumps, including 2048 data and 864 control bumps with 40 μm diameter and 80 μm pitch, delivering 8 MB ReRAM capacity and 25.6 GB/s bandwidth at 100 MHz. A ReRAM load interface (RLI) bridges the logic die and the stacked ReRAM dies, leveraging double clock-rate sampling to ensure reliable clock domain crossing. The CB fetcher unit retrieves blockwise CBs from the ReRAM dies through the RLI based on the indices stored in the instruction buffer (ISA buffer).

The logic die communicates with the off-chip DDR on a Xilinx Zynq XCZU5EV FPGA host through parallel GPIOs, where an EMA controller (EMAC) manages the DDR access, while an ARM Cortex-M0 MCU configures all modules over APB buses with four PLLs supplying independent clocks. It also integrates a 64 KB ISA buffer and 3 MB SRAM comprising a 1 MB weight buffer and a 2 MB global token buffer, together with several specialized units. The tile-fused tensor engine (TFTE) packs a 4 K mixed-precision MAC array for both W4A8 and W8A8 with 16 tile fusion units, the non-linear processing unit holds separate LayerNorm/RMSNorm and Softmax engines, the LRU rotates activations through the reconfigurable FWHT array (RFA) and Hadamard accumulator unit (HAU) before dynamic quantization, and the WDOS coordinates TLM verification and DLM drafting through out-of-order scheduling. To store the DLM weights beyond the single-chip 8 MB ReRAM, the RS-LPU scales to a four-chip system with unidirectional ring topology, where an inter-chip transceiver on each chip links its neighbor over source-synchronous parallel GPIOs at 150 MHz with 32-bit width.

Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on November 03,2026 at 12:36:40 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

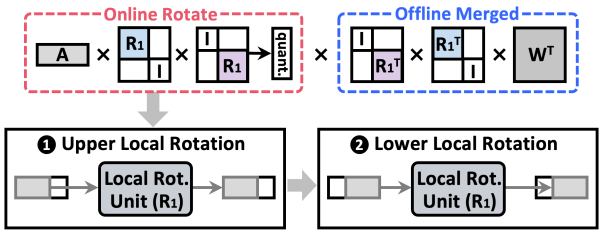


Fig. 7. Two-staged overlapped local rotation based on decomposed FWHT.

IV. LRU With Decomposed FWHT

As discussed in Section II-B, activation outliers amplify quantization errors at low bit widths, whereas Hadamard rotation enables outlier-free W4A8 quantization. However, supporting Hadamard rotation for diverse TLM channel lengths demands a reconfigurable global FWHT array with heavy area overheads. To address this, we propose a decomposed FWHT algorithm that approximates the global rotation with two overlapped local rotations shown in Fig. 7, together with an area-efficient on-chip LRU. As shown in Fig. 8, the LRU conducts the local rotation over two consecutive stages, during which the upper and lower tiles are dispatched sequentially through the RFA and the HAU. The rotated tokens are dynamically quantized to INT8 in the end for the next-layer W4A8 GEMM. Recent works such as LightRot [28] and Revolver [8] leverage a similar strategy that couples a pot local rotation with fine-grained groupwise quantization, whereas the proposed decomposed FWHT supports an arbitrary channel shape with two overlapped npot local rotations without groupwise quantization. Architecturally, LightRot optimizes its pot rotation with a transposable register file, while Revolver utilizes a paired FWHT to reuse adders, and they both perform input permutation to narrow the per-group quantization range. In contrast, the LRU leverages locality-aware folding for the deep FWHT while optimizing the Hadamard GEMM with an outer-product dataflow. Besides, the local rotation is applied to the feed-forward network layer of the TLM and not to the Q and K projection layers, since we do not quantize the KV cache to a low bit width.

A. Decomposed FWHT

To achieve reconfigurable FWHT across diverse TLM channel dimensions n, we leverage the npot Hadamard construction $\mathbf{H}_d = \mathbf{H}_{2^k} \otimes \mathbf{H}_m$ described in Section II-B to factorize the global FWHT into two overlapped local FWHTs as shown in Fig. 7, each covering a $2^k \times m$ channel of the activation. Unlike the pot $\mathbf{H}_{2^k}$ that is recursive in k, the npot $\mathbf{H}_m$ only admits a finite set of orders from the precomputed library [21], and not every (m, k) pair is realizable. Consequently, the supported (m, k) design space is fixed at compile time, and the FWHT range is restricted to an area-efficient 2^6 within which the enumerable $\mathbf{H}_m$ orders are pre-tabulated. Given this restricted range, we search the (m, k) pair whose combined coverage spans the original dimension n while keeping the overlapped region as small as possible. The search is formulated as a lexicographic minimization of $(2 \cdot 2^k \cdot m - n, m)$ subject to $2 \cdot 2^k \cdot m \geq n$, where the first term bounds the overlap region and the second breaks ties toward a smaller $\mathbf{H}_m$ GEMM. The searched (m, k) is first validated through offline accuracy evaluation, and the validated pair is then encoded by the compiler into the LRU instruction. At runtime, the TAU decodes the (m, k) from the instruction and schedules the RFA and HAU accordingly.

B. Reconfigurable FWHT Array

As shown in Fig. 3, FWHT is generally realized as a multi-stage butterfly network whose direct implementation relies on FP16 adders and registers, which incurs heavy area overhead that scales rapidly with the butterfly depth. To avoid utilizing high-precision FP16 operators, we first conduct dynamic INT16 input activation quantization at the array input, after which the RFA is built entirely out of INT16/32 adders and registers. Notably, the two operands of every adder reside in closed register positions at the early butterfly stages, and consecutive stages share highly overlapping input windows. By exploiting this locality similarity, we merge two neighboring stages into a shared adder tree with output reuse. At its inputs, a multi-input router selects the operand pair, and a negation controller selectively flips operand signs according to the active FWHT stage. In our design, the 2^1–2^2 and 2^3–2^4 stages are each merged into a single stage, and a lightweight FWHT mode selector drives the multiplexers to pick which of the two logical stages is active. However, the 2^5 and 2^6 stages are kept separate, as paired operands at later stages are routed far apart in physical and no longer exhibit the locality similarity required for cross-stage merging. Fig. 9 depicts the overall RFA architecture. The FWHT mode selector receives the (m, k) decoded from the instruction and activates the specific 2^k butterfly path. At runtime, the TAU then dispatches the rotation tiles into the array under this selection. In this way, the RFA is reconfigurable to support 2^1–2^6 FWHT with minimal area overhead.

Unlike conventional folded butterfly FFT hardware [29] that time-multiplexes complex multipliers through long pipeline FIFOs and commutators, the RFA first quantizes the floating-point activations into fixed-point and exploits the ± 1 Walsh–Hadamard factors to replace each complex multiplier with a single-cycle INT adder and a negation controller. Then, it leverages the operand locality to split and merge the early stages while keeping the later stages unfolded.

C. Hadamard Accumulator Unit

In the Hadamard GEMM $\mathbf{H}_m\mathbf{X}$, the $m \times m$ matrix $\mathbf{H}_m$ serves as the left operand, while the output matrix from the RFA serves as the right operand, which inherently matches the layout requirement of the outer-product dataflow. To reduce area overhead, the HAU and the RFA share a local token buffer implemented on multi-bank dual-port SRAM, whose row width is aligned to the maximum pot FWHT range 2^6 with INT32 data precision, and four HAU processing elements (PEs) execute the Hadamard GEMM as depicted in Fig. 10. Following the outer-product dataflow layout, we map the token rows of $\mathbf{X}$ and the $\mathbf{H}_m$ columns onto the dual-port SRAM, and the detailed timing chart of the HAU is shown in Fig. 11. In this schedule, the two read ports fetch both operands

Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on November 03,2026 at 12:36:40 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

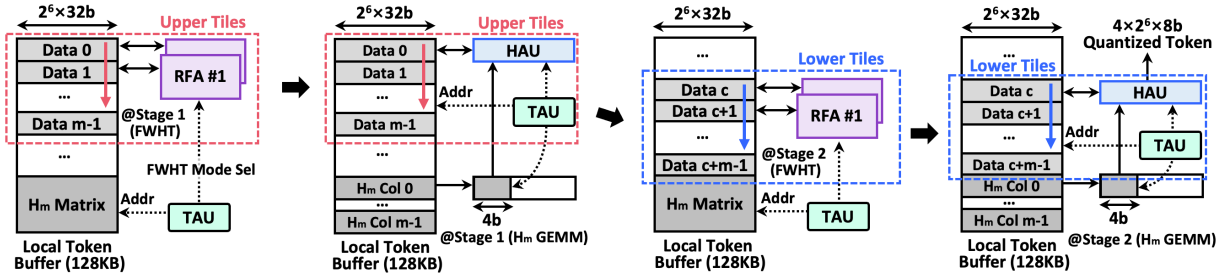


Fig. 8. LRU workflow based on proposed decomposed FWHT algorithm. c, starting token row of the second-stage rotation. $4 \times 2^6 \times 8b$, quantized token output of four rows with up to 2^6 elements each. $4b$, the 4-bit H_m column tile feeding the four PEs. Dashed arrows, control flow. Black arrows, local token buffer accesses. Red and blue, first-stage and second-stage dataflow.

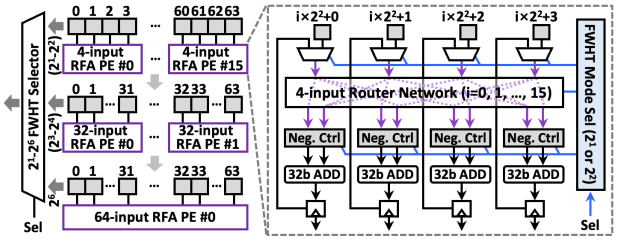


Fig. 9. RFA architecture.

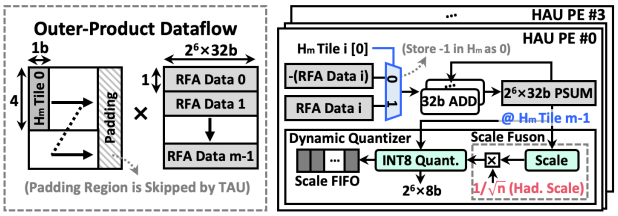


Fig. 10. HAU architecture.

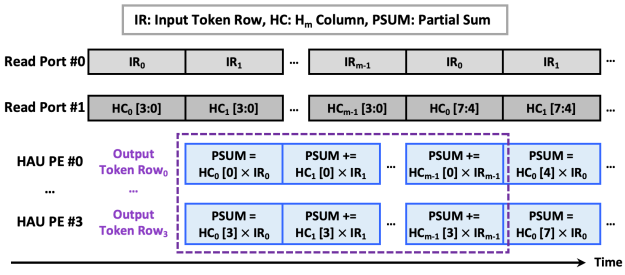


Fig. 11. HAU outer-product dataflow timing chart.

concurrently so that the four PEs jointly produce four output token rows after m cycles, whereas a single-port buffer would serialize the two reads and halve the HAU throughput. In each cycle, the TAU simultaneously fetches a $4 \times 1 \times 1b$ $\mathbf{H}_m$ column tile and a $1 \times 2^k \times 32b$ token row tile from the 128 KB local token buffer, and the four PEs accumulate a $4 \times 2^k \times 32b$ partial sum. After all $\mathbf{H}_m$ columns are traversed, the $4 \times 2^k \times 32b$ output slice is streamed to the dynamic quantizer for round-to-nearest (RTN) [15] quantization, and the quantized $4 \times 2^k \times 8b$ output tile is stored in the global token buffer, with the scaling factors bypassed to the TFTE. For $k < 6$ or $m < 2^6$, the unused lanes are zero-padded in the local token buffer and dynamically skipped in the accumulators. Moreover, we fuse the Hadamard scale $1/(n)^{1/2} = 1/(m)^{1/2} \cdot 1/(2^k)^{1/2}$ and the scale in the dynamic quantizer to avoid FP operations. Besides, the PE conditionally negates the token row tile before partial sum accumulation whenever the corresponding $\mathbf{H}_m$ bit is 0, since the ± 1 entries are stored as single bits, with -1 encoded as 0. As a result, the FP scale fusion and the outer-product dataflow together replace the MAC in each HAU PE with a single adder. The LRU based on the RFA and HAU enables accurate W4A8 TLM quantization, delivering a 3.82×–3.93× speedup over BF16 SD and saving 92.7% area compared with global rotation.

V. ReRAM-Stacked PNM With BVQ

As analyzed in Section II-C, the DLM weight EMA is amplified by γ in each SD iteration and becomes a non-negligible bottleneck of SD at long γ. To alleviate this issue, we leverage the ReRAM-on-logic stacking technology together with the co-designed BVQ algorithm to cluster and store the DLM weights in high-density ReRAM. Four ReRAM dies are stacked above the logic die, each providing 2 MB of capacity through a 512-bit read port. On the logic die, the CB fetcher unit decodes the CB index field of the instruction into ReRAM physical addresses, after which the ReRAM controller inside each die fetches the requested CBs into the RLI at 100 MHz. Since the arrival time of the ReRAM read data varies due to net delays, the RLI first samples the data through a double-rate stabilizer in the 200 MHz domain and then forward the sampled read data into the asynchronous FIFO groups for reliable clock domain crossing. In parallel, the CB fetcher unit decodes the bank-level addresses of the 1 MB weight buffer, under which the RLI writes the stabilized data into the corresponding weight buffer banks. Finally, the TFTE fuses the token tiles sharing the same CB entry at the MAC array inputs to avoid redundant ReRAM reads.

A. Blockwise Vector Quantization

Vector quantization is a weight compression technique that clusters weight tiles into vectorwise indices and a shared CB at extremely low bit widths with high accuracy [30], [31]. However, reconstructing the weights before the MAC array requires an on-chip CB buffer and multi-port register files, which introduces considerable area overhead [32]. To tackle this, we exploit the DLM's small parameter count that admits affordable retraining to co-design a BVQ algorithm BVQ with

Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on November 03,2026 at 12:36:40 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

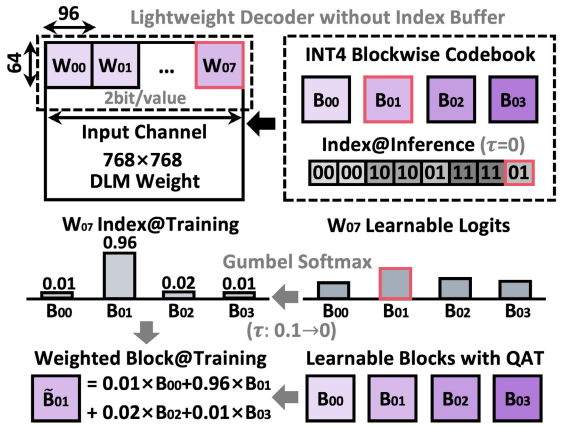


Fig. 12. Proposed BVQ algorithm that jointly optimizes CB selection and quantization via Gumbel–Softmax and INT4 QAT.

the RS-PNM architecture. As shown in Fig. 12, BVQ clusters weights into coarse-grain blockwise CBs with compact indices, rather than a tiny CB paired with enormous indices. During training, we jointly learn the CB selection and the INT4 CB entries through the combination of Gumbel–Softmax [33] and QAT inspired by [18] and [34]. A direct way to learn the CB selection is to assign each weight block a learnable categorical distribution $\{\pi_i\}_{i=1}^{K}$ over the K CB entries and to sample the one-hot CB index $\mathbf{y}$ via a Gumbel-noised argmax

$$\mathbf{y} = \text{onehot}\big(\arg\max_i [\log \pi_i + g_i]\big) \tag{5}$$

where $g_i = -\log(-\log u_i)$ with $u_i \sim \text{Uniform}(0, 1)$ is the Gumbel noise that carries the sampling randomness. The argmax operator, however, is non-differentiable and leaves the distribution $\{\pi_i\}$ without gradient updates during backpropagation. We therefore leverage the Gumbel–Softmax to approximate the one-hot index with a soft assignment $\tilde{\mathbf{y}} = [\tilde{y}_1, \ldots, \tilde{y}_K]$, formulated as

$$\tilde{y}_i = \frac{\exp\big((\log \pi_i + g_i)/\tau\big)}{\sum_{j=1}^{K} \exp\big((\log \pi_j + g_j)/\tau\big)} \tag{6}$$

where the temperature τ controls the hardness of the soft index. As τ tends to 0, $\tilde{\mathbf{y}}$ approaches the one-hot vector $\mathbf{y}$. Suppose the CB is $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_K\}$, each $\mathbf{b}_i$ is updated with a learnable scaling factor s through learned step-size quantization [35], following:

$$\hat{\mathbf{b}}_i = \big\lfloor \text{clip}\left(\mathbf{b}_i/s,\ -Q_n,\ Q_p\right) \big\rceil \cdot s \tag{7}$$

where $Q_n = 2^{b-1}$ and $Q_p = 2^{b-1} - 1$ for a b-bit signed integer. The reconstructed weight block at training is then represented as a differentiable weighted sum over the quantized CB entries

$$\tilde{\mathbf{W}}_b = \sum_{i=1}^{K} \tilde{y}_i \cdot \hat{\mathbf{b}}_i. \tag{8}$$

The backward pass then jointly optimizes the block distribution $\{\pi_i\}$, the INT4 CB entries, and the scaling factor associated with each CB. During BVQ training, τ is annealed from 0.1 toward 0, under which the soft assignment converges to the one-hot vector $\mathbf{y}$ and its argmax index is frozen as the CB index. In the example of Fig. 12, a 768 × 768 DLM weight matrix is partitioned into 64 × 768 sub-weights along the output channel, and each sub-weight is split into 8 blocks of 64 × 96 (W_{00} to W_{07}) that BVQ clusters into 4 blockwise CB entries (B_{00} to B_{03}) with 8 two-bit indices. Each entry is a 64 × 96 INT4 tensor, and each block weight is reconstructed from the CB entry selected by its 2-bit index, as W_{07} maps to B_{01}. Since each sub-weight then stores $4 \times (64 \times 96 \times 4) + 8 \times 2$ bits over 64 × 768 weights, BVQ compresses the DLM weights to ~2-bit/value.

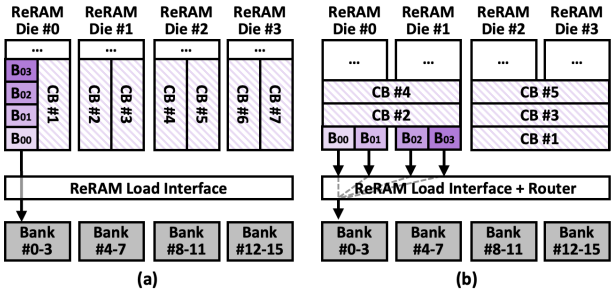


Fig. 13. Two CB mapping schemes for RS-PNM. (a) Vertical CB mapping. (b) Horizontal CB mapping.

B. Tile-Fused Tensor Engine

During BVQ, the CB size is first co-designed with the per-die ReRAM bank width to improve ReRAM utilization and maximize the bandwidth. We then design the CB mapping scheme to store the entries across the banks of the four stacked ReRAM dies. As shown in Fig. 13, the entries can be placed along two directions, where horizontal and vertical denote the placement direction of the CB entries. The horizontal CB mapping spreads the entries of one CB across multiple ReRAM dies, which requires a router in the RLI to gather them into the same weight buffer bank. The router not only incurs extra area but also blocks the loading of the next CB until the gathering completes, which lowers the effective ReRAM bandwidth. Therefore, we adopt the vertical CB mapping that stacks the entries of each CB vertically and loads each CB into the same weight buffer bank without routing. Since each PE line reads the CB entry in order, the complete weight tile is first reconstructed by storing its blocks into the weight buffer bank sequentially according to their indices, for example W_{00} to W_{07} in Fig. 12. However, the repeated selection of the same CB in the reconstruction will result in redundant CB EMA from the ReRAM, as shown in Fig. 14, leading to extra latency overhead.

To mitigate this redundant CB access, we design a tile fusion unit that reuses the indices of each CB entry encoded in the instruction to merge the token tiles sharing the same CB entry, after which the merged tiles are further quantized to INT8 and forwarded to the MAC array. Consequently, each CB entry is read from ReRAM only once, removing the redundancy caused by the repeated selection of the same CB entry. In addition, each TFTE is reconfigurable to support independent tile fusion across diverse CBs. The merged token tiles of different CBs are dispatched to the scratchpad of each vertical TFTE cluster, and each cluster then executes the output-stationary dataflow in parallel. For larger CBs spanning multiple ReRAM

Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on November 03,2026 at 12:36:40 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

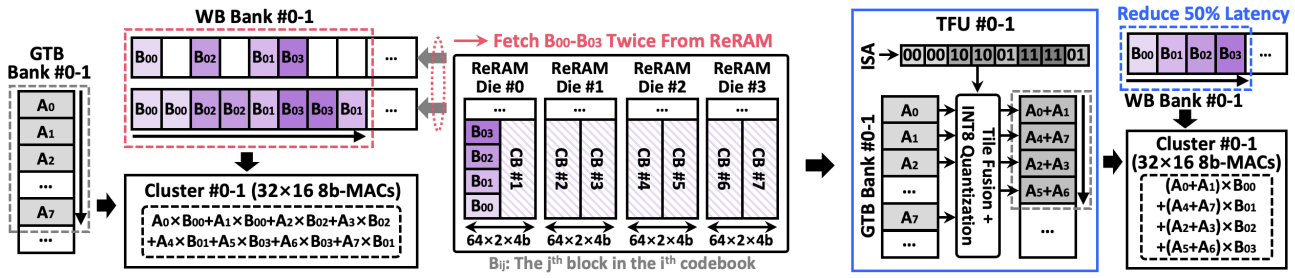


Fig. 14. Redundant CB access issue in vertical CB mapping across four ReRAM dies and TFTE with a tile fusion unit that merges token tiles sharing the same CB entry to resolve this issue.

banks, the TFTE clusters are grouped accordingly to match the bank-level parallelism. As shown on the right of Fig. 14, where two clusters are grouped to serve a two-bank CB parallelism. The RS-PNM with BVQ achieves a 1.1×–1.46× speedup over the W4A8 SD baseline with LRU.

VI. APSD With Out-of-Order Scheduling

As analyzed in Section II-D, parallel SD methods [26] overlap the DLM drafting and TLM verification to eliminate the mutual waiting latency, improving the SD speedup over vanilla SD. To avoid the resource competition between the concurrent TLM and DLM, parallel SD is typically deployed on multi-chip systems that isolate the two workloads through inter-chip parallelism [7]. Nevertheless, this isolation wastes both memory bandwidth and compute resources as they are partially allocated to either DLM or TLM, which is not fully utilized and is prohibitive for edge AI accelerators under stringent bandwidth budgets. Moreover, parallel SD typically adopts a short γ as it yields a higher all-acceptance probability to trigger TLM verification. A short γ frequently triggers the parallel draft-and-verify mode due to its high all-acceptance probability, yet it suffers from a limited accepted token yield and a low overlap efficiency, where the DLM finishes its short drafting window well before the TLM verification completes. Conversely, a long γ improves the per-round accepted token yield, yet its low all-acceptance probability prevents the parallel draft-and-verify mode from being sustained across consecutive iterations, wasting the DLM drafting work and increasing the overall decoding latency.

A. Adaptive Parallel SD

To trigger the parallel draft-and-verify mode through the high all-acceptance probability of a short γ and harness the high accepted token yield of a long γ, we propose an APSD algorithm that adaptively switches between two modes based on TLM verification feedback, illustrated in the lower part of Fig. 5. APSD begins in the drafting-only mode with a short γ, followed by the parallel draft-and-verify mode with a long γ. Once the verification feedback indicates that the previous-round draft tokens are all accepted and the newest decoded token from the TLM matches the first draft token of the current round, APSD continues in the parallel draft-and-verify mode with a long γ, forwarding the current-round draft tokens directly into the next TLM verification. Otherwise, APSD falls back to the drafting-only mode with a short γ, where the DLM proceeds to the next drafting round serially without waiting for the TLM single-token decode.

In APSD, the aggressive long-γ parallel drafting remains affordable because the DLM weights are loaded from the high-bandwidth stacked ReRAM, while the TLM weights are loaded from the bandwidth-limited off-chip DRAM on separate paths, so it consumes the ReRAM bandwidth without contending for the TLM DRAM bandwidth that bounds the decode latency. APSD is therefore a hardware-specific utilization strategy of the RS-LPU rather than a general SD algorithm, since on architectures where the DLM and TLM share the DRAM bandwidth, the long-γ parallel drafting policy would result in severe and ineffective drafting latency overhead. Furthermore, to store the complete DLM backbone that exceeds the single-chip RS-PNM capacity, we scale the RS-LPU into a 4-chip system organized in a unidirectional ring topology. Since the RS-LPU has isolated the storage of the TLM and DLM, we choose intra-chip parallelism in which each chip handles the DLM and TLM workloads in parallel. This intra-chip parallelism hinges on two design factors. One is the allocation of DLM weights across chips, as the effective capacity may be low if the mapping introduces excessive bubbles. The other is the per-chip scheduling of compute and memory resources, as resource contention between the DLM and the TLM exists, which may lead to low resource utilization.

B. CB-Interleaved Intra-Layer Mapping

Since the RS-LPU targets single-batch DLM inference bottlenecked by weight EMA, we partition each projection layer of the DLM across the four chips with tensor parallelism, which aligns naturally with the horizontal CB mapping in RS-PNM shown in Fig. 13. As for the mapping in each chip, we propose a codebook-interleaved intra-layer mapping (CILM) scheme that interleaves the CBs of different projection layers to avoid storage bubbles, as shown in Fig. 15. The CILM allows CBs from a subsequent layer to occupy the tail space that would otherwise be padded. Beyond interleaving CBs across different layers, the CILM also supports headwise interleaving such as headwise Q/K/V CBs. Moreover, the TFTE natively supports the CILM by recombining its clusters to match the parallelism required by each case, delivering over 90% capacity and bandwidth utilization without degrading the gain from the multi-chip scaling.

Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on November 03,2026 at 12:36:40 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

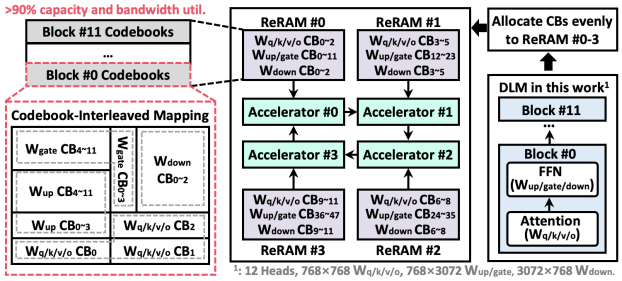


Fig. 15. Intra-chip parallelism of the multi-chip RS-LPU and a CILM example eliminating storage bubbles.

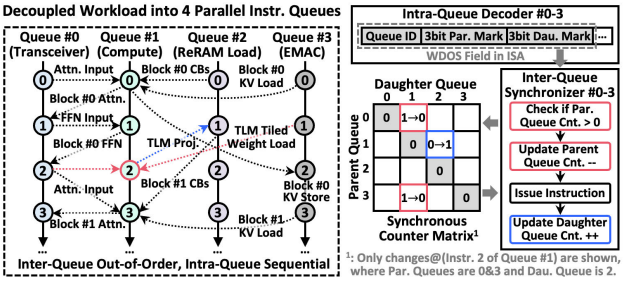


Fig. 16. WDOS that orchestrates four parallel instruction queues with intra-queue sequential and inter-queue out-of-order issue.

C. Workload-Decoupled Out-of-Order Scheduler

In addition to the DLM weight allocation addressed in Section VI-B, the per-chip scheduling must coordinate the concurrent DLM and TLM workloads within each chip. They compete for the on-chip compute engines, SRAM buffers, and inter-chip transceiver, while their weights reside in heterogeneous memories of the RS-LPU. Furthermore, these two workloads dynamically switch across SD iterations, which is difficult to manage through pure static compilation and motivates a hardware-compiler co-optimization. To this end, we first decouple the APSD workloads into four major operation types, namely the inter-chip transceiver for intermediate feature map exchange, the compute cores covering the TFTE, non-linear processing unit, and LRU, the ReRAM loading of DLM weights, and the EMAC for the EMA of TLM weight and TLM/DLM KV-cache. Based on this decoupling, we further design an out-of-order scheduler WDOS, shown in Fig. 16 that orchestrates four per-type instruction queues in parallel with minimal hardware overhead and offloads coarse-grain dependency resolution to the compiler.

In the WDOS, instructions within each independent queue are issued sequentially, whereas instructions from different queues are scheduled out-of-order, as illustrated in Fig. 16. With each queue bound to its dedicated engines, intra-queue sequential issue simplifies dependency tracking for lightweight scheduling, while inter-queue out-of-order issue enables coarse-grained latency overlap across queues. To support this scheme, the top controller of the RS-LPU first leverages four intra-queue decoders to extract the field of queue id and dependency markers in each instruction. Then, the markers are routed to the four inter-queue synchronizers by queue id. These synchronizers jointly maintain a synchronous counter matrix based on register groups that tracks the status of each instruction, where one denotes completion. An instruction issues only when all its parent registers in the counter matrix are one, indicating that its parent instructions in the other three queues have all completed on the corresponding engines. Once the instruction completes, its synchronizer flips its daughter register from from to one to notify its daughter instructions. This dependency-aware scheduling in the WDOS effectively handles the adaptive mode switching of APSD and achieves a balanced hardware-compiler co-optimization. It attains a 1.1×–1.29× speedup over the RS-PNM baseline and reduces the rejected DLM latency by 10%–14%.

VII. Measurement Results and Comparisons

A. Chip Specification and Verification Setup

Fig. 17 presents the specification summary and photographs of the logic die and ReRAM die in RS-LPU, which are fabricated in 55 nm with areas of 55.980 and 5.535 mm^2, respectively. Under a 0.89–1.40 V supply voltage, the RS-LPU operates from 62.5 to 285 MHz with 265 mW to 3.06 W power consumption, reaching a peak performance of 2.33 TOPS@INT8 at its maximum 285 MHz frequency. Each ReRAM die runs at a fixed 1.1 V voltage supply and 100 MHz operating frequency with 49.54 mW power consumption at a 100% toggle rate. In the logic die, the RS-PNM occupies only 7%, including the RLI and CFU, since the stacked ReRAM resides on separate dies. The LRU and the WDOS occupy 12% and less than 2%, respectively, while the TFTE MAC array and the on-chip SRAM dominate the remaining area.

Fig. 18 shows the RS-LPU chip photograph, cross-sectional view of the stacking interface, TEM image of ReRAM device, and the measured ReRAM resistance distribution curve. The logic and ReRAM dies are first fabricated at the full-wafer level, after which the ReRAM wafer is diced, and each ReRAM die is individually bonded onto the logic wafer via face-to-face bumping with 40 μm bump diameter and 80 μm pitch. The stacked logic wafer is then diced into individual chips, each comprising one logic die with four ReRAM dies on top. Each chip is packaged in a wire-bond BGA as previously illustrated in Fig. 4, where ReRAM power and data signals are delivered by the bottom logic die via the face-to-face bumps. Fig. 18(b) presents the cross-sectional SEM view of the stacking interface, showing that the bumps are bonded without misalignment between the ReRAM and logic dies. Fig. 18(d) plots the measured ReRAM current distribution, where the distributions of high- and low-resistance states remain fully separated down to a 0.1 ppm level.

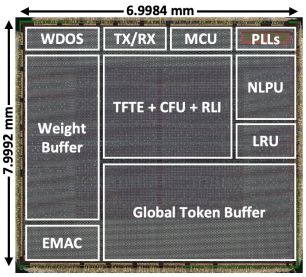
Figs. 19 and 20 show the experimental setups for ReRAM power and RS-LPU performance testing, respectively. In the wafer-level probing system of Fig. 19, an FPGA tester generates the test signals and delivers them to the ReRAM wafer through a probe card, a multimeter captures the current responses, and the probe-to-bump contact is aligned and monitored under a microscope. In the four-chip verification system of Fig. 20, the Xilinx Zynq ZU5EV FPGA board serves as the host. A DC power supply delivers a 0.9–1.4 V core voltage to the RS-LPU chips through an on-board voltage tuning

Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on November 03,2026 at 12:36:40 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

Specification (Logic Die)	
Technology	55nm
Voltage	0.89-1.40 V
Frequency	62.5-285 MHz
Die Area	55.98 mm²
Precision	W: INT4/8, A: INT8/16/32
SRAM	3.43 MB
Performance*1	0.51*2-2.33*3 TOPS
Power	0.265*2-3.060*3 W

*1: 1 OP = 1 Multiplication or Addition in the PE Array.
*2: At 0.89V and 62.5MH. *3: At 1.40V and 285MHz.



Specification (4 ReRAM Dies)	
Technology	55nm
Voltage	1.1 V
Frequency	100 MHz
Die Area	5.553×4 mm²
Capacity	2×4 MB
Bump #	2912 (Data: 2048, Ctrl.: 864)
Bump Pitch	80μm
Bump Diameter	40μm
Read Power	49.54×4 mW

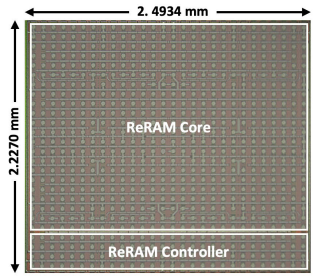


Fig. 17. Specification and photograph of logic die and ReRAM die.

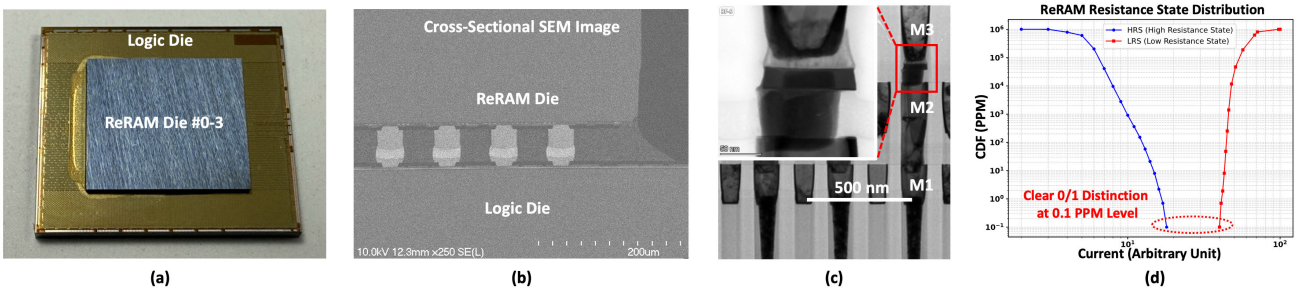


Fig. 18. (a) Chip photograph. (b) Cross-sectional SEM of the stacking interface. (c) TEM image of ReRAM device. (d) Measured ReRAM resistance distribution.

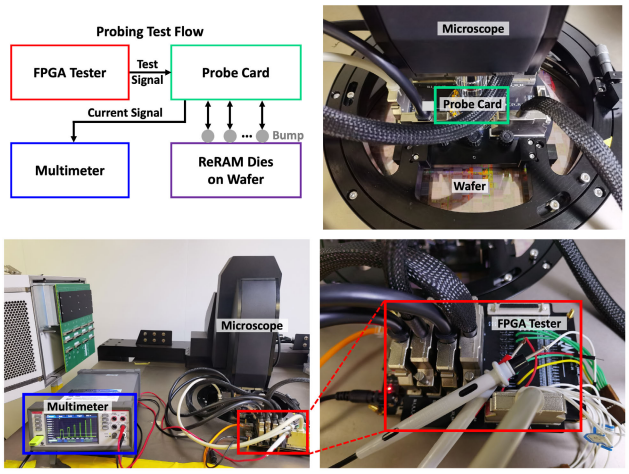


Fig. 19. Wafer-level probing setup of the ReRAM dies.

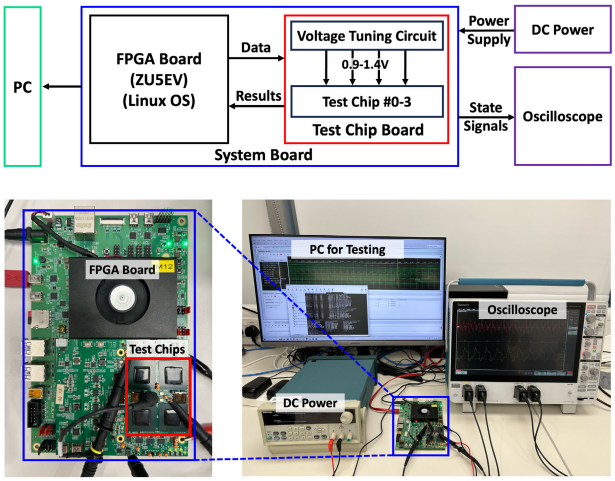


Fig. 20. Four-chip test system setup of the RS-LPU.

circuit. A custom compiler based on Multi-Level Intermediate Representation (MLIR) generates the data, instruction, and configuration packages offline. The host then delivers these packages to the RS-LPU, and the verification results are then forwarded to a PC for monitoring while working state and clock signals are observed on an oscilloscope.

B. Evaluation on Key Features

Table I reports the Wikitext-2 PPL of three LLMs from 1B to 8B parameters under multiple quantization settings, where the baseline quantizes weights with GPTQ and activations with RTN. We further incorporate hardware wrappers that align with the approximate computing of the Softmax, LayerNorm, and RMSNorm in the non-linear processing unit. Specifically, pushing to W4A8 without rotation degrades the PPL by 1.16–3.10 compared with the BF16 baseline across the three

TABLE I
WIKITEXT-2 PPL ON VARIOUS LLMS AND QUANTIZATION METHODS

Precision	Vicuna-1B	LLaMA2-7B	LLaMA3-8B
BF16	9.18	5.47	6.14
W8A8KV8	9.27 (+0.09)	5.92 (+0.45)	6.46 (+0.32)
W4A8KV8	10.34 (+1.16)	8.57 (+3.10)	7.54 (+1.40)
W4A8KV8 w/ g128	9.52 (+0.34)	6.71 (+1.24)	7.61 (+1.47)
W4A8KV8 w/ GR	9.43 (+0.25)	5.68 (+0.21)	6.70 (+0.56)
W4A8KV8 w/ LRU	9.41 (+0.23)	5.68 (+0.21)	6.71 (+0.57)

LLMs. The rotation-free groupwise quantization at a group size of 128, denoted as g128, narrows the W4A8 gap to 0.34 to 1.47, yet it stays inferior to rotation and still incurs frequent high-precision dequantization. In contrast, global Hadamard rotation (GR) reduces the W4A8 gap to 0.21–0.56. Notably,

Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on November 03,2026 at 12:36:40 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

TABLE II
ADJUSTED WIN-RATE OVER FP BASELINE ON MT-BENCH

Precision	Vicuna-1B & LLaMA-160M	LLaMA2-7B & LLaMA-160M	LLaMA3-8B & LLaMA-296M
W8A8KV8	44.1%	52.2%	50.6%
W4A8KV8	32.5%	36.9%	23.8%
W4A8KV8 w/ g128	41.6%	41.9%	36.6%
W4A8KV8 w/ GR	41.6%	45.9%	40.9%
W4A8KV8 w/ LRU	41.8%	45.9%	40.7%

TABLE III
BENCHMARK EVALUATION ON THREE TLM-DLM PAIRS

TLM & DLM	Vicuna-1B & LLaMA-160M	LLaMA2-7B & LLaMA-160M	LLaMA3-8B & LLaMA-296M
Data Precision	TLM: W4A8, DLM: W4A8 (2bit/value with BVQ)*1		
TLM Perplexity (↓)*2	9.41 (+0.23)	5.68 (+0.21)	6.71 (+0.57)
Avg. ReRAM Access*3*4	36.75 MB/Token	0.0432 GB/Token	0.0399 GB/Token
Avg. DRAM Access*3*4	97.68 MB/Token	0.6637 GB/Token	0.8674 GB/Token
Avg. EMA Saving*5	6.67×	4.79×	4.99×
Decoding Throughput*3*4	135.69 Token/s	17.82 Token/s	14.08 Token/s
Speedup*5	7.17×	4.46×	5.33×
Energy Consumption*3*4	18.26 mJ/Token	123.41 mJ/Token	151.59 mJ/Token
Energy Saving*5	4.85×	3.74×	3.95×

*1: Each DLM is trained on ShareGPT dataset. *2: Evaluated on Wikitext-2 dataset with 2048 prefill length. *3: EMA is simulated assuming 4-chip parallelism with an LPDDR3 interface while the on-chip performance is measured at 1.40V and 285MHz. *4: Evaluated on MT-Bench dataset. *5: The baseline is BF16 SD.

the proposed LRU with decomposed FWHT attains nearly the same PPLs as global rotation. Table II further reports the adjusted win-rate on MT-Bench [11] against the full-precision baseline judged by DeepSeek-V4-Flash, where the rotation-based methods are much more stable, and the LRU stays almost the same as the global rotation across the three pairs.

Table III presents the end-to-end evaluation on three TLM-DLM pairs. Among the three pairs, Table III shows that the benefit of the ReRAM-on-logic stacking is most pronounced on the Vicuna-1B and LLaMA-160M pair, which attains the highest EMA saving of 6.67× and energy saving of 4.85×. The reason is that the 160 M DLM shares a large portion of the EMA of the 1B TLM, while the four-chip system holds most of the DLM parameters in the stacked ReRAM, fully exploiting its high bandwidth to reach a favorable sweet spot. This also guides a co-design strategy that jointly explores the optimal DLM and architecture configuration for a specific TLM. The joint design space includes the DLM size, the number of chips, and the per-chip ReRAM capacity.

Table IV reports the draft length, the average accepted tokens τ, and the speedup of the LRU, RS-PNM, and APSD across the three pairs. In the evaluation, the short and long γ are set to 8 and 20 for the Vicuna-1B and LLaMA2-7B pairs, and 6 and 25 for the LLaMA3-8B pair. To determine them, we first run vanilla SD once at an extremely long γ such as 50 to record a single per-round accepted-token trace. We then approximately reconstruct the APSD traces under various combinations of short and long γ by splitting and recombining the recorded accepted tokens. After simulating each combination on our simulator, we select the one with the lowest end-to-end latency. Finally, we run APSD on the dataset with the searched short and long γ to obtain the actual per-round accepted-token trace for the reported results.

TABLE IV
AVERAGE ACCEPTED TOKENS τ AND SPEEDUP ANALYSIS

TLM & DLM	Method	Draft Length	τ	Speedup
Vicuna-1B & LLaMA-160M	+LRU	6	3.59	3.83×
	+RS-PNM	8	4.07	5.58×
	+APSD	8+20	4.48	7.17×
LLaMA2-7B & LLaMA-160M	+LRU	12	2.95	3.82×
	+RS-PNM	12	2.88	4.11×
	+APSD	8+20	2.87	4.46×
LLaMA3-8B & LLaMA-296M	+LRU	6	2.21	3.93×
	+RS-PNM	10	2.68	4.86×
	+APSD	8+25	2.63	5.33×

Table IV also indicates that the BVQ-based RS-PNM preserves or even improves the average accepted token yield τ, since a larger γ becomes feasible in the RS-PNM. APSD with WDOS further hides the long-γ drafting by the TLM verification, raising the speedup from 5.58× to 7.17×, from 4.04× to 4.45×, and from 4.86× to 5.34× on the Vicuna-1B, LLaMA2-7B, and LLaMA3-8B pairs, respectively. Within the total runtime, the weight loading of the DLM from ReRAM and the TLM from DRAM accounts for more than 82%, while the computation and the C2C communication account for around 12% and less than 6%, respectively, as WDOS overlaps them with the concurrent weight loading.

C. Thermal Analysis

Thermal analysis is an indispensable part of the 3-D-stacked architecture design, as the vertical integration of the ReRAM and logic dies raises the power density and makes heat dissipation a critical concern. Before sign-off, we evaluated the worst case power of the stacked chip with a margin of 6 W, and then characterized the thermal behavior of the LFBGA package using a commercial thermal estimation tool following the JEDEC standard. The results indicate that the maximum junction temperature of the RS-LPU package reaches 85.2 °C at a 25 °C ambient temperature and 1 m/s air flow. The actual power consumption of the RS-LPU is 3.3 W, which is far below the 6 W, thus indicating that the ReRAM-on-logic stacking introduces no thermal issue. Besides, the stacked ReRAM is qualified for a data retention of at least one year at 85 °C, and the actual junction temperature stays well below this bound, leading to an even longer retention in actual operation.

D. Comparison With State-of-the-Art LPU

Table V compares the proposed RS-LPU with several state-of-the-art LPUs. In this comparison, INT8 is for the TFTE MAC array, INT16 for the LRU activation input, and INT32 for the non-linear processing unit such as Softmax and RMSNorm. The RS-LPU integrates 3.43 MB SRAM and 8 MB stacked ReRAM per chip with 25.6 GB/s bandwidth, and scales to the four-chip system shown in Fig. 20 with 32 MB ReRAM capacity and 102.4 GB/s bandwidth. Combining the aggressive compression of BVQ with the high ReRAM capacity utilization of CILM, the RS-LPU stores the 108 M

Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on November 03,2026 at 12:36:40 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

TABLE V
MEASUREMENT RESULTS AND COMPARISONS WITH STATE-OF-THE-ART LPUS

	VLSI'25 [19]	VLSI'25 [20]	ISSCC'25 [5]	ISSCC'25 [6]	ISSCC'24 [4]	This Work
Technology	28nm	28nm	28nm	28nm	28nm	**55nm with ReRAM Die on Logic Wafer Stacking**
Memory Capacity	24KB SRAM	272KB SRAM	384KB SRAM	500KB SRAM	500KB SRAM	**3.43MB SRAM + 8MB ReRAM**
Memory BW	NA	NA	NA	1.6 GB/s (DRAM)	1.6 GB/s (DRAM)	**25.6 GB/s (ReRAM)**
Supply Voltage (V)	0.6-1.0	0.6-1.0	0.63-1.0	0.58-1.0	0.7-1.1	**0.89-1.40 (logic)**
Frequency (MHz)	85-410	110-500	50-460	25-200	50-200	**63.5-285 (Logic), 100 (ReRAM)**
Die Area (mm^2)	1.56	12.65	3.52	20.25	20.25	**55.98 (Logic), 22.21 (ReRAM)**
Precision	W: INT4 A: INT4/INT8	W: INT8/NUQ1~4 A: FP16	W: INT8 A: BF16	W: INT1-16/1.58b A: INT4/8/16	INT8	**W: INT4/INT8 A: INT8/16/32**
Peak Performance[*1] (TOPS or TFLOPS)	0.63-2.52 (Hybrid A4/A8)	0.12-0.96 (NUQ1~4@Decoding)	3.58 (Dense)	13.1 (87.5% Sparsity)	3.41 (Dense)	**2.33[*2] (Dense W4A8/W8A8)**
Benchmark Model and Perplexity (↓)	LLaMA2-7B (6.16)	OPT-1.3B (15.1) LLaMA2-7B (6.26)	GPT2-0.1B (30.85) OPT-1.3B (15.50) LLaMA2-7B (5.66)	LLaMA-3B (1.58bit: 10.3)	GPT2-0.7B (14.96)	**Vicuna-1B (9.41) LLaMA2-7B (5.68) LLaMA3-8B (6.71)**
System-Level Decoding[*3*4] Throughput (Token/s)	LLaMA2-7B (5.46)	OPT-1.3B (32.75) LLaMA2-7B (6.76)	GPT2-0.1B (286.69) OPT-1.3B (28.13) LLaMA2-7B (5.63)	LLaMA-3B (1.58bit: 74.92)	GPT2-0.7B (69.9)	**Vicuna-1B (135.69) LLaMA2-7B (17.82) LLaMA3-8B (14.08)**
System-Level Energy[*3*4] Consumption (mJ/Token)	LLaMA2-7B (223.69)	OPT-1.3B (47.09) LLaMA2-7B (181.48)	GPT2-0.1B (5.36) OPT-1.3B (55.09) LLaMA2-7B (218.25)	LLaMA-3B (1.58bit: 16.03)	GPT2-0.7B (21.90)	**Vicuna-1B (18.26) LLaMA2-7B (123.41) LLaMA3-8B (151.59)**

[*1]: EMA is excluded. [*2]: @1.40V, 285MHz. [*3]: EMA is simulated assuming 4-chip parallelism following the prototype setup in this work, and each is equipped with an LPDDR3 interface (6.4GB/s bandwidth and a 120pJ/Byte access energy), whereas on-chip performance is measured with ReRAM access included. [*4]: Evaluated on the MT-Bench dataset with average prefill/decode lengths of 454/512 tokens.

backbone parameters at a 2-bit/value compression ratio while the remaining ReRAM capacity holds the scaling factors, the channelwise parameters such as RoPE, and the instructions. The word embedding (23 M for LLaMA-160M and 94 M for LLaMA-296M) is accessed through a retrieval-based lookup on the FPGA host with tolerable latency, and the quantization-sensitive language-model head (lm_head) is quantized to 4-bit and stored in DRAM.

For the large-scale TLM, the LRU enables reliable W4A8 quantization, achieving comparable PPL to the weight-only W8 [5] and the sub-4b LPU [20], and more stable PPL than the W4A4/W4A8 LPU [19]. While most prior LPUs [4], [6], [8] focus on compute-bound prefill optimization, the critical bottleneck of LLM lies in memory-bound decoding. For a fair system-level evaluation covering both prefill and decode stages, we evaluate each LPU at its peak performance on the MT-Bench dataset with an average prefill length of 454 and decode length of 512. Furthermore, we scale each LPU to a four-chip system matching the configuration of the prototype in this work, where each chip is equipped with an LPDDR3 interface delivering 6.4 GB/s bandwidth at 75% utilization and 120 pJ/Byte energy cost [36], [37]. Specifically, on the LLaMA2-7B model, the RS-LPU achieves the highest decoding throughput of 17.82 token/s and the lowest energy consumption of 123.41 mJ/token among all compared LPUs.

VIII. LIMITATIONS AND FUTURE WORK

Compared with the ReRAM-on-logic 3-D stacking, a 2-D single-die alternative with larger on-chip memory results in heavy area and power overhead, as the density of ReRAM is more than 10× that of SRAM. Furthermore, the vertical data transaction energy of the RS-LPU is 7.74 pJ/Byte, which is far lower than the off-chip DRAM access energy [36]. Therefore, the bumping-based ReRAM-on-logic 3-D stacking provides a low-power and cost-effective method for on-chip memory capacity and bandwidth extension. Nevertheless, the current prototype integrates the ReRAM and logic dies on an identical and mature 55 nm node for this proof-of-concept tapeout due to the cost considerations, where the capacity and bandwidth are still limited. However, the bumping-based die-on-wafer stacking does not require the two dies to share the same node and can move to a more advanced node, so a product-level design can fabricate both dies on more advanced and different nodes to enlarge the stacked capacity beyond the current 8 MB and deliver higher bandwidth. To quantify the benefit of moving to an advanced node, our previously fabricated 28 nm ReRAM product, with a 0.03 μm^2 1T1R cell, delivers 32 MB within 9.98 mm^2, whereas the current 55 nm prototype delivers only 2 MB within a 5.535 mm^2 ReRAM die, so the same silicon area provides about 8.9× the on-chip capacity. Meanwhile, the 28 nm ReRAM reads at 125 MHz under 1.05 V with a per-byte read energy of merely 0.34 pJ/Byte, lower than the 7.74 pJ/Byte of the 55 nm stacked ReRAM and far below the 120 pJ/Byte of the off-chip LPDDR3. Moreover, the RS-LPU could scale to a larger multi-chip system and be co-optimized with the dataflow architecture, or stack more ReRAM dies vertically in one chip through hybrid bonding. We leave these to our future work.

IX. CONCLUSION

This article presents RS-LPU, a ReRAM-on-logic stacked LLM accelerator that delivers practical accuracy with superior throughput and energy efficiency over prior LPUs. RS-LPU resolves the dual weight EMA bottleneck of SD

Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on November 03,2026 at 12:36:40 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

from DLM and TLM through several software–hardware co-design techniques. First, the decomposed FWHT approximates the global rotation by two overlapped local rotations and is realized by the on-chip LRU with reuse exploiting locality similarity and MAC-free dataflow, enabling reconfigurable outlier-free TLM quantization across arbitrary LLM channel dimensions at minimal area overhead. Second, the RS-PNM leverages the high bandwidth and density of ReRAM-on-logic stacking, with a tailored BVQ algorithm performing blockwise clustering on the DLM to reduce its weight, EMA amplified by autoregressive drafting. Third, the APSD scheme combines short-γ drafting with long-γ parallel SD to harness the high accepted token yield without heavy drafting latency overhead, while the WDOS further avoids the resource competition through hardware-compiler co-optimization. Benefiting from the proposed co-designs, RS-LPU achieves 17.82 token/s decoding throughput at 123.41 mJ/token energy consumption on the representative LLaMA2-7B model. Through ReRAM-on-logic stacking, multi-chip scaling, and software–hardware co-design, RS-LPU delivers low-power and cost-effective memory-capacity extension, paving the way for high-throughput and energy-efficient LLM deployment for edge applications.

REFERENCES

[1] J. Bai et al., "Qwen technical report," 2023, *arXiv:2309.16609*.

[2] M. A. Ferrag, N. Tihanyi, and M. Debbah, "From LLM reasoning to autonomous AI agents: A comprehensive review," 2025, *arXiv:2504.19678*.

[3] J. Duan, S. Yu, H. L. Tan, H. Zhu, and C. Tan, "A survey of embodied AI: From simulators to research tasks," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 6, no. 2, pp. 230–244, Apr. 2022.

[4] S. Kim et al., "C-transformer: An energy-efficient homogeneous DNN-transformer/SNN-transformer processor for large language models," *IEEE J. Solid-State Circuits*, vol. 60, no. 10, pp. 3802–3815, Oct. 2025.

[5] Y. Qin, "23.8 An 88.36 Tops/w bit-level-weight-compressed large-language-model accelerator with cluster-aligned INT-FP-GEMM and bi-dimensional workflow reformulation," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, vol. 68, Sep. 2025, pp. 420–422.

[6] S. Kim, J. Lee, and H.-J. Yoo, "Slim-llama: A 4.69 mW large-language-model processor with binary/ternary weights for billion-parameter llama model," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2025, pp. 421–423.

[7] P. Dong et al., "31.1 A 14.08-to-135.69 Token/s ReRAM-on-logic stacked outlier-free large-language-model accelerator with block-clustered weight-compression and adaptive parallel-speculative-decoding," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2026, pp. 532–534.

[8] S. Kim, J. Oh, B. Kim, Y. Choi, G. Park, and H.-J. Yoo, "31.2 revolver: Low-bit GenAI accelerator for distilled-model and CoT with phase-aware-quantization and rotation-based integer-scaled group quantization," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2026, pp. 534–536.

[9] Y. Leviathan, M. Kalman, and Y. Matias, "Fast inference from transformers via speculative decoding," in *Proc. Int. Conf. Mach. Learn.*, 2023, pp. 19274–19286.

[10] Y. Li, F. Wei, C. Zhang, and H. Zhang, "Eagle: Speculative sampling requires rethinking feature uncertainty," in *Proc. Int. Conf. Mach. Learn.*, 2024, pp. 28935–28948.

[11] L. Zheng et al., "Judging LLM-as-a-judge with MT-bench and chatbot arena," 2023, *arXiv:2306.05685*.

[12] E. Frantar, S. Ashkboos, T. Hoefler, and D. Alistarh, "GPTQ: Accurate post-training quantization for generative pre-trained transformers," 2022, *arXiv:2210.17323*.

[13] M. Sun, X. Chen, J. Z. Kolter, and Z. Liu, "Massive activations in large language models," 2024, *arXiv:2402.17762*.

[14] S. Ashkboos et al., "QuaRot: Outlier-free 4-bit inference in rotated LLMs," in *Proc. Adv. Neural Inf. Process. Syst.* 37, 2024, pp. 100213–100240.

[15] Z. Liu et al., "SpinQuant: LLM quantization with learned rotations," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2025, pp. 92009–92032.

[16] S. Kim, Y. Choi, B. Kim, J. Oh, and H.-J. Yoo, "GyRot: Leveraging hidden synergy between rotation and fine-grained group quantization for low-bit LLM inference," in *Proc. IEEE Int. Symp. High-Perform. Comput. Archit. (HPCA)*, Sydney, Australia, Jun. 2026, pp. 1–15, doi: 10.1109/HPCA68181.2026.11408453.

[17] S. Kim et al., "EdgeDiff: Energy-efficient multi-modal few-step diffusion model accelerator using mixed-precision and reordered group quantization," *IEEE J. Solid-State Circuits*, vol. 61, no. 1, pp. 168–182, Jan. 2026.

[18] P. Dong et al., "A 28 nm 0.22μJ/token memory-compute-intensity-aware CNN-transformer accelerator with hybrid-attention-based layer-fusion and cascaded pruning for semantic-segmentation," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2025, pp. 1–3.

[19] Y. Wang et al., "LLM-CIM: A 28 nm 126.7 TOPS/W input-LUT-based digital CIM macro with reconfigurable matrix multiplication and nonlinear operation modes for LLMs," in *Proc. Symp. VLSI Technol. Circuits (VLSI Technol. Circuits)*, Jun. 2025, pp. 1–3.

[20] Z. Wu et al., "CELLA: A 28 nm compute-memory co-optimized real-time digital CIM-based edge LLM accelerator with 1.78 ms-response in prefill and 31.32 Token/s in decoding," in *Proc. Symp. VLSI Technol. Circuits (VLSI Technol. Circuits)*, Jun. 2025, pp. 1–3.

[21] N. J. A. Sloane. (2024). *A Library of Hadamard Matrices*. [Online]. Available: http://neilsloane.com/hadamard/

[22] Y. Cao et al., "A 1.2 GHz 12.77 GB/s/mm^2 3D two-DRAM-one-logic process-near-memory chip for edge LLM applications," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2026, pp. 524–526.

[23] Z. Yue, Y. Wang, C. Li, S. Wei, Y. Hu, and S. Yin, "3D-PATH: A hierarchy LUT processing-in-memory accelerator with thermal-aware hybrid bonding integration," in *Proc. 58th IEEE/ACM Int. Symp. Microarchitecture*, Oct. 2025, pp. 78–93.

[24] Z. Yue et al., "Exploiting similarity opportunities of emerging vision AI models on hybrid bonding architecture," in *Proc. ACM/IEEE 51st Annu. Int. Symp. Comput. Archit. (ISCA)*, Jun. 2024, pp. 396–409.

[25] S. Yu, *Semiconductor Memory Devices and Circuits*. Boca Raton, FL, USA: CRC Press, 2022.

[26] T. Liu et al., "PEARL: Parallel speculative decoding with adaptive draft length," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2025, pp. 1085–1104.

[27] Y. Shen, J. Shen, Q. Kong, T. Liu, Y. Lu, and C. Wang, "SpecBranch: Speculative decoding via hybrid drafting and rollback-aware branch parallelism," 2025, *arXiv:2506.01979*.

[28] S. Kim, Y. Choi, J. Oh, B. Kim, and H.-J. Yoo, "LightRot: A light-weighted rotation scheme and architecture for accurate low-bit large language model inference," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 15, no. 2, pp. 231–243, Jun. 2025.

[29] M. Ayinala, M. Brown, and K. K. Parhi, "Pipelined parallel FFT architectures via folding transformation," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 20, no. 6, pp. 1068–1081, Jun. 2012.

[30] M. van Baalen et al., "GPTVQ: The blessing of dimensionality for LLM quantization," 2024, *arXiv:2402.15319*.

[31] Y. Liu et al., "VPTQ: Extreme low-bit vector post-training quantization for large language models," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2024, pp. 8181–8196.

[32] S. Li et al., "MVQ: Towards efficient DNN compression and acceleration with masked vector quantization," in *Proc. 30th ACM Int. Conf. Architectural Support Program. Lang. Operating Syst.*, Volume 1, Mar. 2025, pp. 731–745.

[33] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," 2016, *arXiv:1611.01144*.

[34] G. Fang et al., "MaskLLM: Learnable semi-structured sparsity for large language models," in *Proc. Adv. Neural Inf. Process. Syst.* 37, 2024, pp. 7736–7758.

[35] S. K. Esser, J. L. McKinstry, D. Bablani, R. Appuswamy, and D. S. Modha, "Learned step size quantization," in *Proc. Int. Conf. Learn. Represent.*, 2020.

[36] M. Gao, J. Pu, X. Yang, M. Horowitz, and C. Kozyrakis, "TETRIS: Scalable and efficient neural network acceleration with 3D memory," in *Proc. 22nd Int. Conf. Archit. Support Program. Lang. Oper. Syst.*, 2017, pp. 751–764.

[37] K. Chandrasekar et al., "DRAMPower: Open-source DRAM power and energy estimation tool," Delft Univ. Technol., Delft, The Netherlands, Tech. Rep., 2012. [Online]. Available: http://www.drampower.info

Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on November 03,2026 at 12:36:40 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.



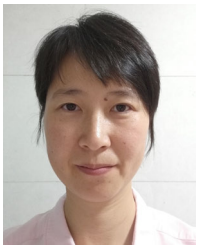
Pingcheng Dong (Graduate Student Member, IEEE) received the B.E. degree in microelectronics from the Southern University of Science and Technology, Shenzhen, China, in 2022. He is currently pursuing the Ph.D. degree with The Hong Kong University of Science and Technology, Hong Kong, China.

He has published more than ten papers in ISSCC, ESSERC, ASSCC, DAC, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—I: REGULAR PAPERS, and IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS II: EXPRESS BRIEFS. His research interests include model compression, AI chips, and hardware–software co-design.



Yonghao Tan (Graduate Student Member, IEEE) received the B.E. degree in microelectronic science and engineering from the Southern University of Science and Technology, Shenzhen, China, in 2023. He is currently pursuing the Ph.D. degree with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Hong Kong, China.

His research interests include hardware–software co-design for AI chips, computer vision, natural language processing, and 3-D processing.



Xuejiao Liu received the master's degree in microelectronics and solid-state electronics from Peking University, Beijing, China, in 2009.

She is currently a Chief Researcher with the AI Chip Center for Emerging Smart Systems (ACCESS), Hong Kong, China. Her research interests include application-specific algorithm-hardware co-design and high-power-efficiency AI accelerator chips for multimodal large model inference.



Peng Luo received the B.S. and M.S. degrees in microelectronics from Peking University, Beijing, China, in 2006 and 2009, respectively.

He is currently a Principal Researcher with the AI Chip Center for Emerging Smart Systems (ACCESS), Hong Kong, China. His research interests include AI accelerator chip design and energy-efficient technology for edge devices.



Yu Liu received the B.Sc. and M.Sc. degrees in electrical engineering and nuclear technology from Tsinghua University, Beijing, China, in 2012 and 2014, respectively, and the Ph.D. degree in system engineering from the City University of Hong Kong, Hong Kong, China, in 2017.

He is currently a Principal Researcher with the AI Chip Center for Emerging Smart Systems (ACCESS), Hong Kong. His research interests include hardware-friendly AI algorithms and efficient AI accelerator toolchain optimization.



Di Pang received the Ph.D. degree in computer architecture from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2009.

He is currently a Principal Engineer with the AI Chip Center for Emerging Smart Systems (ACCESS), Hong Kong, China. His research interests include software–hardware co-design for energy-efficient AI chips, automatic design space exploration (DSE), and agile architecture modeling and profiling for LLM inference accelerators.



Fengshi Tian (Graduate Student Member, IEEE) received the B.Eng. degree from Fudan University, Shanghai, China, in 2021. He is currently pursuing the Ph.D. degree with the ECE Department and the AI Chip Center for Emerging Smart System (ACCESS), The Hong Kong University of Science and Technology (HKUST), Hong Kong, supervised by Prof. Kwang-Ting Tim Cheng and Prof. Chi-Ying Tsui.

He was with CenBRAIN Neurotech, Westlake University, Hangzhou, China, as a Visiting Student, from 2020 to 2021, supervised by Prof. Mohamad Sawan. His current research interests lie in neuromorphic computing, neural interfaces, and computing-in-memory systems.

Mr. Tian was a recipient of several awards, including the Best Paper Award at IEEE AICAS 2023.



Songchen Ma (Member, IEEE) received the B.Eng. degree from Jilin University, Changchun, China, in 2019 and the Ph.D. degree from Tsinghua University, Beijing, China, in 2024.

He is currently a Post-Doctoral Fellow with ACCESS, The Hong Kong University of Science and Technology (HKUST), Hong Kong. He has published over ten papers in top-tier journals and conferences, including two papers in *Science Robotics* (one as a cover article, the other reported by AAAS News), as well as publications in *Nature Electronics*, IEEE TRANSACTIONS ON COMPUTERS, ISCA, CICC, and ICCAD. His research realms focus on neuromorphic computing, scalable AI chip architecture, and embodied intelligence systems.



Xijie Huang received the B.Eng. degree from Shanghai Jiao Tong University, Shanghai, China, in 2020. He is currently pursuing the Ph.D. degree with The Hong Kong University of Science and Technology, Hong Kong, China.

His current research interests include building efficient large-scale models (LLMs and diffusion models) and human-centric computer vision.



Shih-Yang Liu received the B.B.A. degree in finance from The Hong Kong University of Science and Technology, Hong Kong, China, in 2020, where he is currently pursuing the Ph.D. degree in computer science.

His research interests include model compression, training acceleration, and post-training methods for large language models (LLMs). His work has appeared in top-tier venues, such as ICML, ICLR, EMNLP, ISSCC, and DAC.



Dong Zhang received the Ph.D. degree from NJUST, Nanjing, China, in 2021.

He is currently a Research Assistant Professor with the Department of Electronic and Computer Engineering, HKUST. Prior to this, he was a Post-Doctoral Researcher with the Department of CSE from January 2022 to November 2023. He has published more than 35 technical papers in top journals and conferences, including IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON IMAGE PROCESSING, IEEE TRANSACTIONS ON MEDICAL IMAGING, IEEE TRANSACTIONS ON MULTIMEDIA, IEEE TRANSACTIONS ON CYBERNETICS, MedIA, PR, IJCV, AAAI, ACM MM, IJCAI, CVPR, ECCV, ICCV, and NeurIPS. His research focuses on machine learning, computer vision, and medical image analysis, with a particular interest in image classification, semantic segmentation, object detection, and their potential applications.

Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on November 03,2026 at 12:36:40 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.



Zhichao Lu (Member, IEEE) received the M.S. degree from Tsinghua University, Beijing, China, and the Ph.D. degree in electrical engineering from the University of Florida, Gainesville, FL, USA.

He is currently the Founder and the CEO of Hefei Reliance Memory Ltd. He was previously a Visiting Researcher with IMEC, Leuven, Belgium, and worked with GlobalFoundries, Malta, NY, USA, on 28-nm and 20-nm low-power CMOS process integration. He also joined Rambus Inc., San Jose, CA, USA, where he focused on emerging memory devices, chips, and systems, and low-power DRAM architectures for advanced data-center platforms. His current research interests include RRAM technology, emerging memory devices, and in-memory/near-memory computing systems.



Luhong Liang (Senior Member, IEEE) received the B.S. and Ph.D. degrees in computer science from Tsinghua University, Beijing, China, in 1997 and 2001, respectively.

He worked as a Senior Researcher and a Staff Researcher with the Intel China Research Center, Beijing, from 2001 to 2007, an Associate Professor with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, from 2008 to 2011, the Director of the System Research Group, National Engineering Laboratory for Digital Video Technology, Beijing, from 2009 to 2011, a Principal Engineer and the Director of the Machine Learning Platform Group, Hong Kong Applied Science and Technology Research Institute, Hong Kong, from 2011 to 2020. He joined AI Chip Center for Emerging Smart Systems (ACCESS), Hong Kong, as the Executive Director in November 2020. He authored and co-authored more than 40 peer-reviewed papers and ten U.S. patents in video coding, computer vision, pattern recognition, image processing, and AI accelerator areas.



Chi-Ying Tsui (Senior Member, IEEE) received the B.S. degree in electrical engineering from The University of Hong Kong, Hong Kong, and the Ph.D. degree in computer engineering from the University of Southern California, Los Angeles, CA,USA.

He joined the Department of Electrical and Electronic Engineering, The Hong Kong University of Science and Technology (HKUST), Hong Kong, in 1994, and is a Full Professor with the Department. He is the Founding Head and a Professor with the Division of Integrative Systems and Design, the Director of Entrepreneurship Education, HKUST, HKUST, the Director of Executive Education at the School of Engineering, HKUST, and the Associate Director of the AI Chip Center for Emerging Smart Systems. He has authored more than 300 referred publications and holds 16 U.S. patents in power management, VLSI architecture, and systems. His research focuses on VLSI design for energy-efficient embedded machine learning, low-power multimedia, wireless and AI applications, and power management circuits.

Dr. Tsui's work has earned six best paper awards from the IEEE journals and conferences, along with numerous international VLSI design prizes and teaching awards at HKUST.



Fengbin Tu (Senior Member, IEEE) received the B.S. degree from the School of Electronic Engineering, Beijing University of Posts and Telecommunications, Beijing, China, in 2013, and the Ph.D. degree from the Institute of Microelectronics, Tsinghua University, Beijing, in 2019.

He was a Post-Doctoral Fellow with the AI Chip Center for Emerging Smart Systems (ACCESS), Hong Kong, China, from 2022 to 2023, and a Post-Doctoral Scholar at the Scalable Energy-efficient Architecture Laboratory (SEAL), ECE Department, University of California, Santa Barbara, Santa Barbara, CA, USA, from 2019 to 2022. He is currently an Assistant Professor with the ECE Department, the Associate Director of the Institute of Integrated Circuits and Systems, and the Associate Director of HKUST and Intel Joint Laboratory, HKUST. His research interests include AI chips, computing-in-memory, computer architecture, and reconfigurable computing.

Dr. Tu's AI chips Thinker and ReDCIM have been recognized by the ISLPED Design Contest Award (2017) and Top-10 Research Advances in China Semiconductors (2023). He has received the MIT Technology Review "Innovators Under 35" (TR35) Asia–Pacific (2025), Gold Medal with Jury Congratulations at the Inventions Geneva (2025), and NSFC Excellent Young Scientists Fund (2024). He currently serves as an Associate Editor for IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS and the Vice Chair for IEEE CEDA Hong Kong Chapter. He served as the Design Contest Chair of ASP-DAC 2026 and a PC Member of ISCA, A-SSCC, and DAC.



Liang Zhao (Member, IEEE) received the B.S. degree from Tsinghua University, Beijing, China, in 2010, and the M.S. and Ph.D. degrees from Stanford University, Stanford, CA, USA, in 2013 and 2015, respectively.

He is currently a Tenure-Track Professor with the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, China. He has been engaged in the research and development of novel memory technologies for more than 16 years, specializing in RRAM. His current research interests include novel memory devices and their neuromorphic computing applications.



Kwang-Ting Cheng (Fellow, IEEE) received the Ph.D. degree in electrical engineering and computer science from the University of California, Berkeley, Berkeley, CA, USA, in 1988.

He is currently the Vice President for Research and Development and the Chair Professor jointly appointed with the Department of Electronic and Computer Engineering and Department the Computer Science and Engineering, The Hong Kong University of Science and Technology (HKUST), where he is also the Director of the AI Chip Center for Emerging Smart Systems (ACCESS). Before joining HKUST, he was a Professor of electrical and computer engineering with the University of California, Santa Barbara. Prior to this position, he spent five years with the AT&T Bell Laboratories. He has published more than 600 technical articles, co-authored five books, and holds 20 U.S. patents, with several of his inventions transferred into successful commercial products. His research interests include VLSI design, testing and verification, design automation, computer vision, and medical image analysis.

Dr. Cheng is also a fellow of ACM and Hong Kong Academy of Engineering Sciences. He received the 2020 Pan Wen Yuan Outstanding Research Award from Pan Wen Yuan Foundation, the 2024 CCF Overseas Outstanding Contributions Award from China Computer Federation, and the 2025 ACM SIGDA Pioneering Achievement Award.

Authorized licensed use limited to: Hong Kong University of Science and Technology. Downloaded on November 03,2026 at 12:36:40 UTC from IEEE Xplore. Restrictions apply.