PRIMAL：面向大语言模型推理加速器的存内处理低秩适配

Yue Jiet Chong[1†]、Yimin Wang[2†]、Zhen Wu[3],Xuanyao Fong[4]新加坡国立大学电气与计算机工程系，新加坡 电子邮箱： {jason.yj.chong[1], kelvin.xy.fong[4]}@nus.edu.sg， {yimin.wang[2],e0323083[3]}@u.nus.edu

arXiv:2601.13628v1 [cs.AR] 20 Jan 2026

摘要—本文介绍了 PRIMAL，这是一种基于存内处理（PIM）的低秩适配（LoRA）大语言模型推理加速器。PRIMAL 集成了异构存内处理单元（PE），并通过二维网格处理单元间计算网络（IPCN）将其互连。一种新颖的静态随机存取存储器重编程与电源门控（SRPG）方案，通过将重配置与计算重叠并关闭空闲资源，实现流水线化低秩适配更新和次线性功耗扩展。PRIMAL 采用优化的空间映射和数据流编排，以最大限度地降低通信开销；在 Llama-13B 上使用秩 8 的 LoRA（Q、V）时，与 NVIDIA H100 相比，实现了 1.5× 吞吐量和 25×能效。

索引词—软硬件协同设计、低秩适配（LoRA）、存内处理（PIM）、大语言模型推理加速

I. 引言

近年来，大语言模型取得的进展显著增强了其执行复杂任务的能力，从而推动其在客户服务、软件工程和内容生成等各个领域得到广泛应用 [1]。尽管具备通用能力，但 GPT 和 LLaMA 等模型往往难以满足下游任务的特定要求，因此需要进行特定任务适配，以提升性能和相关性 [2]。模型适配能够增强任务一致性，并通过支持本地部署或受控微调，为缓解数据隐私问题提供了一种途径。然而，对完整模型进行微调需要大量计算和资源，使其难以应用于许多场景。为解决这一局限，LoRA [3]应运而生，成为一种高效的替代方案。

LoRA 在 Transformer架构的特定层中引入可训练的低秩矩阵，而不修改原始模型权重，如图 1 所示。对于特定下游任务，仅替换低秩矩阵，同时保留原始模型权重。LoRA 在显著降低计算和存储成本的同时，实现了与完整微调相当的性能 [3]。

为充分发挥 LoRA 的优势并提升大语言模型推理的能效和吞吐量，我们研究了并针对集成 LoRA 的大语言模型推理加速器实现了软硬件协同设计，该加速器由异构存内计算处理单元和 IPCN 组成。

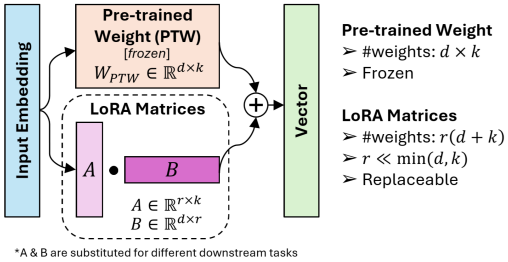

图 1. LoRA 的架构。

II. PRIMAL 硬件架构

PRIMAL 硬件架构采用基于芯粒的设计，每个芯粒称为计算瓦片（CT）。如图 2 所示，每个 CT 集成了一个二维网格 IPCN，用于互连多个异构处理单元。IPCN 负责数据流控制，并执行动态数据乘加（DMAC）操作，以表示注意力分数的$\mathbf{Q}\cdot\mathbf{K}^{\mathbf{T}}$。处理单元执行静态权重乘加（SMAC）操作，对应于 Transformer [4]架构中的矩阵-向量乘法。

A. 处理单元（*PE*）

处理单元由一个非易失性阻变随机存取存储器模拟存内计算（RRAM-ACIM）宏和一个易失性静态随机存取存储器数字存内计算（SRAM-DCIM）宏组成，分别用于对预训练权重矩阵和 LoRA 矩阵进行 SMAC。

1）*RRAM-ACIM*：RRAM-ACIM宏 [5] 具有高密度和非易失性，特别适合存储大型、冻结的预训练权重矩阵。其非易失性使基础模型的权重只需编程一次，从而大幅降低重新配置开销。初始化后，RRAM-ACIM可直接在模拟域中执行SMAC运算，利用模拟存内计算固有的并行性和能效。

本研究部分由新加坡国立大学通过微电子种子基金（2024财年）资助；另一部分由新加坡国家研究基金会（NRF）通过竞争性研究计划资助（项目编号 NRF-CRP24-2020-0002 和 NRF-CRP24-2020-0003）。

† 两位作者对本研究的贡献相同。

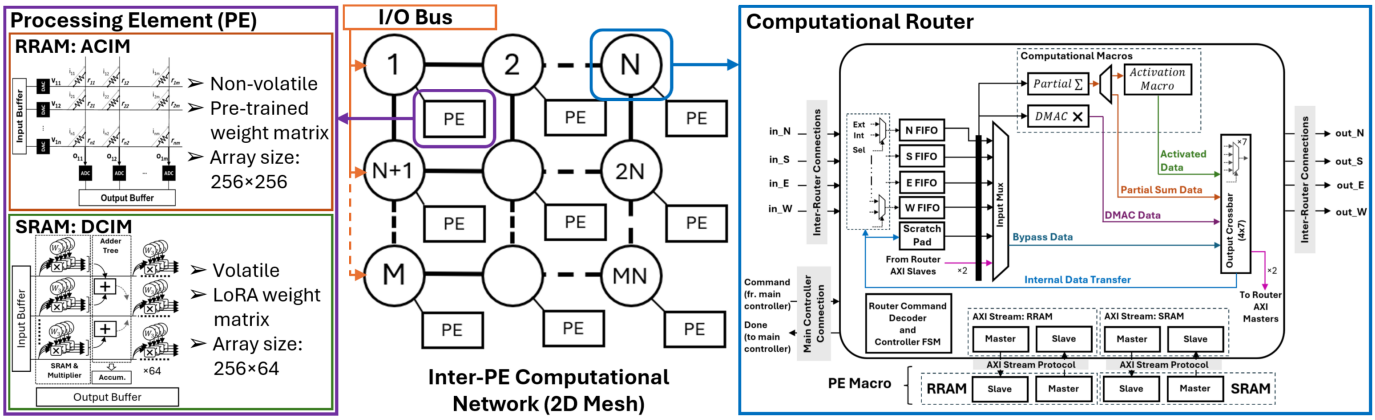

图 2. PRIMAL 硬件设计（每个计算瓦片）：异构处理单元、二维网格 IPCN、计算路由器 ter.

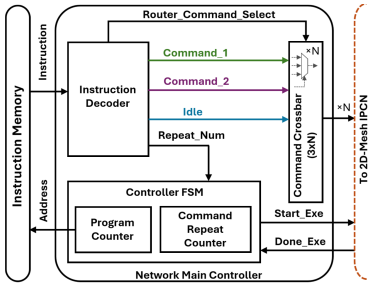

图 3. 网络主控制器（NMC）。

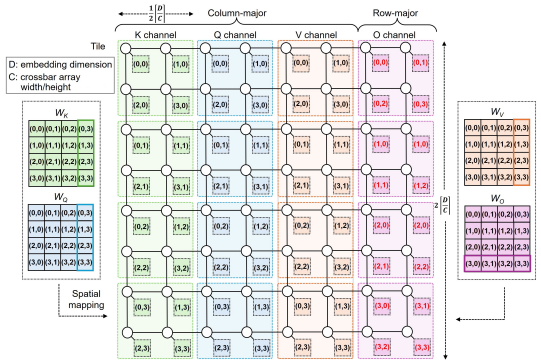

图 4. 注意力层权重矩阵的空间映射。

2）*SRAM*-存内计算：SRAM-存内计算宏 [6] 支持快速写入操作和高精度数字MAC运算，非常适合存放LoRA矩阵。尽管其密度低于RRAM-ACIM，但考虑到LoRA的参数量大幅减少，这一局限可以接受。此外，LoRA具有模块化特性，矩阵常需替换以适应不同的下游任务，因此必须能够快速重新配置。SRAM的快速编程能力可高效更新LoRA权重，从而最大限度地减少运行停顿，并提高系统吞吐量和响应能力。

B. 处理单元间计算网络（*IPCN*）

二维网格 IPCN 旨在促进存内计算处理单元宏的无缝集成，从而能够在网络中高效协调数据流，以支持各种人工智能工作负载。DMAC 操作、部分和累加以及激活运算等计算任务均在路由器中执行，以提高处理效率。为满足动态工作负载需求，IPCN 集成了专用指令集架构，可对数据搬运和计算进行可重新编程的控制，从而确保适应性和可扩展性。

单元路由器配备四个专用于路由器间通信的平面端口，以及两对 AXI-Stream 适配器，分别连接带有阻变存储器和静态随机存取存储器的处理单元。每个通信端口都包含一个先进先出缓冲区，用于临时存储数据并确保数据流畅通。为支持灵活的数据处理，路由器支持在其缓冲区与宏之间进行内部数据传输。这一功能可实现高效的路由器内数据搬运，从而优化各种计算任务的性能。集成式内部控制器负责命令接收和状态报告，确保与网络主控制器 (NMC) 无缝协调。

图 3 所示的 NMC 负责管理二维网格 IPCN 内的数据搬运，并协调路由器的运行，通过存储在指令存储器中的专用指令集，为特定大语言模型工作负载建立经过优化的专用数据流。由于大语言模型工作负载中存在操作冗余，控制器可通过指令使发往路由器的每条命令重复执行。

III. 映射与数据流编排

A. 映射

分区后的权重矩阵$\mathbf{W_Q}$、$\mathbf{W_K}$、$\mathbf{W_V}$和$\mathbf{W_O}$以空间映射的方式映射到处理单元交叉开关阵列上，而相应的中间矩阵**Q**、**K**、**V**和**O**（分别表示查询、键、值和输出）则分配到分布式暂存存储器中。对于处理单元交叉开关上的权重放置，每个矩阵都通过启发式方法限制在一个按列排列的

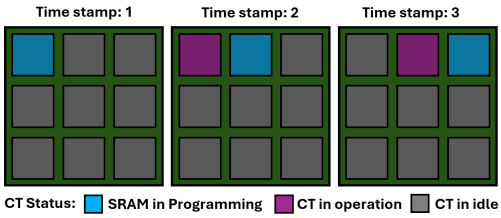

图 5. 我们提出的 SRAM 重编程与电源门控（SRPG）方案。

表 I 系统参数

系统级			
位宽	64	频率	1 GHz
计算瓦片级			
IPCN 维度	32×32	PE #	1024
宏级（每个单元路由器-处理单元对）			
RRAM-ACIM 阵列	256×256	SRAM-存内计算阵列	256×64
暂存器容量	32 KB	FIFO 大小（各）	128 B
DMAC 数量	16	I/O 对数	6

矩形区域内（图 4）。通过调节三个关键因素来优化映射过程：矩阵内形状、矩阵间形状和行列顺序。中间数据与其对应的权重共置，以最大限度地降低通信开销。具体而言，每个中间矩阵都存储在与其权重矩阵所在处理单元阵列相对应的暂存区中。例如，$\mathbf{Q}$ 被放置在连接到$\mathbf{W_Q}$ 预映射位置的路由器的暂存器中。这种共置策略可实现局部输出归约，并提高数据复用效率。

预训练权重和 LoRA 权重采用相同的映射策略，因为 LoRA 引入了低秩适配矩阵，其结构与原始权重矩阵对齐，因此无需额外约束即可采用相同的空间划分和放置策略。

B. 数据流

大语言模型执行中的数据流包括三种主要模式：广播、归约和单播。首先，将输入嵌入广播至承载 $\mathbf{W_Q}$、$\mathbf{W_K}$ 和 $\mathbf{W_V}$ 的处理单元，以进行并行计算。这样即可在多个处理单元上并行计算 $\mathbf{Q}$、$\mathbf{K}$ 和 $\mathbf{V}$ 投影。由于权重在多个列之间进行空间分布，因此归约阶段会汇总多个处理单元产生的部分 SMAC 结果。随后，通过使用 DMAC 执行 $\mathbf{Q} \cdot \mathbf{K^T}$ 来计算注意力分数，并进行 Softmax 激活。单播确保数据以点对点方式传递，从而减少成对点积计算期间不必要的流量。

对于键值（KV）缓存，在解码阶段，每个生成词元对应的$\mathbf{K}$ 和$\mathbf{V}$ 向量会追加到暂存缓冲区中；这些缓冲区已静态预分配，并在预填充期间预先存储了$\mathbf{K}/\mathbf{V}$ 。这些缓冲区以循环方式分布在各个分布式存储单元中，从而实现负载均匀分布并缓解内存争用。

表 II PRIMAL 基准测试：吞吐量和功率

模型	LoRA* 矩阵（输入/输出）	上下文长度	吞吐量（词元/秒）	平均值 功率（瓦）	效率（词元/焦耳）
Llama 3.2 1B	Q	1024/1024	966.32	2.23	433.33
		2048/2048	565.46		253.57
	Q, V	1024/1024	963.47		432.04
		2048/2048	564.48		253.13
Llama 3 8B	Q	1024/1024	308.76	9.58	32.23
		2048/2048	221.37		23.11
	Q, V	1024/1024	307.89		32.12
		2048/2048	220.77		23.04
Llama 2 13B	Q	1024/1024	191.68	14.76	12.99
		2048/2048	145.81		9.88
	Q, V	1024/1024	190.98		12.94
		2048/2048	145.40		9.85

*LoRA，秩 8

表 III PRIMAL 延迟：首令牌生成时间和令牌间延迟

模型	LoRA* 矩阵	上下文长度（输入/输出）	TTFT (s)	ITL (ms)
Llama 3.2 1B	Q	1024/1024	0.370	1.708
		2048/2048	1.192	2.955
	Q, V	1024/1024	0.373	1.711
		2048/2048	1.199	2.958
Llama 3 8B	Q	1024/1024	0.710	5.726
		2048/2048	2.012	8.052
	Q, V	1024/1024	0.782	5.738
		2048/2048	2.037	8.065
Llama 2 13B	Q	1024/1024	0.962	9.494
		2048/2048	2.494	12.499
	Q, V	1024/1024	0.982	9.513
		2048/2048	2.533	12.518

*LoRA 秩 8

循环放置策略确保暂存区利用率不受序列长度影响，始终保持均衡，从而在长上下文推理场景中维持吞吐量并降低延迟。

集合通信模式通过生成树算法进行调度，该算法为每个阶段确定路由路径。该算法利用权重和中间数据规则且对齐的映射，确保流量均衡且无拥塞。

C. SRAM 重编程与电源门控 *(SRPG)*

对于不同的下游任务，必须更新 LoRA 矩阵，因此需要对 SRAM-DCIM 宏进行重编程。此外，大语言模型推理工作负载严格按照逐层顺序执行 [7]。在计算某一层时，所有其他层均处于空闲状态，这为功率优化提供了显著机会。为同时利用顺序执行模式和 SRAM 重编程需求，PRIMAL 采用基于 CT 的逐层权重分配策略，将每一层映射到相邻的 CT。

如图 5 所示，在针对不同下游任务的初始化阶段（时间戳 1），第一个 CT 中的静态随机存取存储器会重新编程。当第一个 CT 开始执行分配给它的工作负载时，下一个 CT 中的静态随机存取存储器会

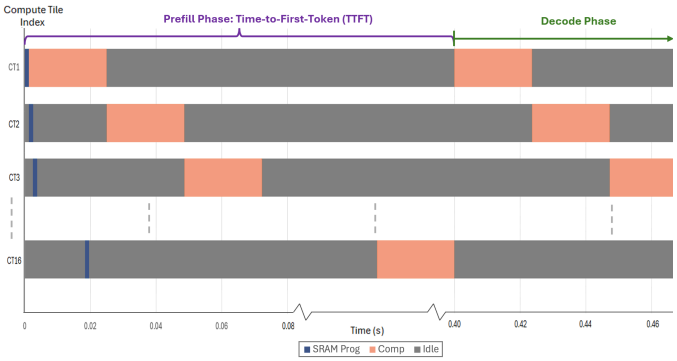

图6. PRIMAL上 Llama 3.2-1B 的硬件调度时序图。

并行重新编程。对于处于空闲状态的 CT，其 IPCN 和 RRAM-ACIM宏会进行电源门控以节省能量。然而，所有 CT 中的静态随机存取存储器和暂存存储器宏仍保持通电，分别用于保留易失性LoRA权重，并确保为 KV缓存保留上下文窗口数据。这种操作方案通过最大限度地减少空闲 CT 的能耗，有效降低了系统功耗。

IV. 系统评估

系统采用全面的硬件软件协同验证方法进行评估。数字硬件组件使用 Verilog硬件描述语言设计并验证。硬件综合使用*SynopsysDesign Compiler*完成，随后使用*CadenceInnovus*进行布局布线。暂存存储器宏的功率和面积使用*CACTI* [8], 获取，而其他硬件宏则通过数学抽象在软件中建模和仿真。推理仿真和基准测试采用基于IPCN指令集及映射方案的周期精确指令级模拟器进行。

A. 性能基准测试

1) 吞吐量和功率：在 PRIMAL 中，系统平均功率取决于 CT 的数量。随着模型规模增大，为存储和处理扩展后的模型权重而启用更多 CT，功耗也随之上升。同时，较大的模型会增加 IPCN 内部的数据搬运和计算需求，导致系统吞吐量下降。

与 Nvidia H100 相比，在 Llama-13B（2048/2048，批次大小 1）上运行 LoRA 秩 8 （Q 、 V）时，PRIMAL 实现了 1.5× 吞吐量和 25× 能效（9.85 tks/J 对比 0.4 tks/J）。

2) 首令牌时间（*TTFT*）：TTFT 表示从接收到提示词到大语言模型生成第一个输出令牌之间的延迟。在 PRIMAL 中，这对应于预填充阶段，包括初始 CT 中 SRAM-存内计算的重新编程，以及所有大语言模型层的总计算时间（图6）。得益于 PRIMAL 的流水线执行方案 SRPG，后续 CT 中 SRAM-存内计算的重新编程实际上与计算重叠进行，因此不会影响 TTFT。

表 IV 硬件宏的平均功率与面积分布（单位）

宏	功率（uW）分布		面积（mm^2）分布	
RRAM-ACIM [5]	120	9.9%	0.1442	65.2%
SRAM-存内计算 [6]	950	78.1%	0.035	15.8%
暂存存储器	42	3.5%	0.013	5.9%
路由器	103	8.5%	0.029	13.1%
总计（路由器处理单元对）	1215	100%	0.2212	100%

#工艺节点：7纳米；每个 CT芯粒的面积：227.5 mm^2

因此，TTFT 主要受每个 CT 的计算时间限制，该计算时间构成流水线中的关键路径。

3) 令牌间延迟（*ITL*）：令牌间延迟对应解码阶段，主要由大语言模型中的总层数决定。随着模型规模扩大，层数也按比例增加，导致计算流水线变得更深。因此，在 PRIMAL 中，数据必须经过更多层，并映射到额外的 CT 上，从而使连续生成词元之间的延迟增加（表 III）。

B. 硬件可扩展性

SRPG方案的实现可对空闲状态下 CT 内的 IPCN 和 RRAM-ACIM宏进行电源门控，从而大幅降低整体系统功耗。对不同规模模型的实证评估表明，与未采用电源门控的基准配置相比，SRPG最高可节省 80% 的功率。因此，系统功耗相对于大语言模型规模呈次线性增长（表 II），使 PRIMAL 能够高度可扩展地运行更大型的大语言模型。

C. 宏功耗与面积

表 IV 总结了 PRIMAL 宏的平均功耗与面积分布。由于集成了模拟 SMAC 运算，且需要数模转换器和模数转换器，RRAM-ACIM 宏占用了最多面积。相比之下，SRAM-DCIM 宏使用加法树和移位器执行数字 SMAC，无需转换器，因此面积更小；但由于数字开关活动增加，且数字计算效率较低，其功耗更高。

V. 结论

通过与IPCN互连集成，并利用LoRA的适应性，PRIMAL充分发挥存内计算的能效优势，在实现无缝下游任务适配的同时，提供高能效的大语言模型推理。此外，所提出的SRPG机制可确保高效的静态随机存取存储器重编程并优化计算流水线，从而实现相对于大语言模型规模的次线性功耗扩展。评估结果表明，对于Llama-13B（2048/2048，LoRA秩 8 Q、V），其吞吐量和能效较Nvidia H100分别提升1.5×和 25×。我们的设计范式不仅应对了大规模大语言模型日益增长的能耗需求，还使PRIMAL成为一种高度可扩展的架构，能够通过SRPG以极低的功耗开销支持结合LoRA的更大规模大语言模型。

参考文献

[1] Z. Chkirbene、R. Hamila、A. Gouissem和U. Devrim，“大语言模型（LLM）在工业中的应用：应用、挑战与趋势综述”，载于*2024 IEEE*第*21*届智能社区国际会议：利用人工智能、机器人和物联网改善生活质量（*HONET*），2024年，第229–234页。[2] P. Kumar，“大语言模型（llms）：综述、技术框架与未来挑战”，《人工智能评论》，第57卷，第10期，第260页，2024年8月。[3] E. J. Hu、Y. Shen、P. Wallis、Z. Allen-Zhu、Y. Li、S. Wang、L. Wang、W. Chen 等，“大语言模型的低秩适配。”*ICLR*，第1卷，第2期，第3页，2022年。[4] A. Vaswani、N. Shazeer、N. Parmar、J. Uszkoreit、L. Jones、A. N. Gomez、Ł. Kaiser和I. Polosukhin，“注意力就是你所需要的一切”，神经信息处理系统进展，第30卷，2017年。[5] W. Wan、R. Kubendran、C. Schaefer、S. B. Eryilmaz、W. Zhang、D. Wu、S. Deiss、P. Raina、H. Qian、B. Gao 等，“一种基于电阻式随机存取存储器的存算一体芯片”，《自然》，第608卷，第7923期，第504–512页，2022年。[6] Y.-D. Chih、P.-H. Lee、H. Fujiwara、Y.-C. Shih、C.-F. Lee、R. Naous、Y.-L. Chen、C.-P. Lo、C.-H. Lu、H. Mori、W.-C. Zhao、D. Sun、M. E. Sinangil、Y.-H. Chen、T.-L. Chou、K. Akarvardar、H.-J. Liao、Y. Wang、M.- F. Chang和T.-Y. J. Chang，“16.4 一种面向机器学习边缘应用、采用22纳米工艺的89TOPS/W和16.3TOPS/mm2全数字SRAM全精度存算一体宏单元”，载于*2021 IEEE*国际固态电路会议（*ISSCC*），第64卷，2021年，第252–254页。[7] Y. Wang、Y. J. Chong和X. Fong，“LEAP：采用可扩展存算一体网络架构、平衡数据流和细粒度并行的大语言模型推理”，载于*2025 IEEE/ACM*计算机辅助设计国际会议（*ICCAD*），2025年，第1–9页。[8] N. Muralimanohar、R. Balasubramonian和N. P. Jouppi，“Cacti 6.0：大型缓存建模工具”，惠普实验室，第27卷，第28页，2009年。