《StreamPIM: Streaming Matrix Computation in Racetrack Memory》论文解析报告

21307174 刘俊杰

May 2024

1 相关背景

在当前计算系统中，面临着内存墙问题，即处理器与内存之间的速度差距。这导致了大规模应用程序的性能瓶颈，特别是对于数据密集和计算密集型任务，如机器学习、科学计算和图形分析。传统的 DRAM 存储系统由于技术限制，如保持时间违规、感知边际不足和低可靠性等，无法有效应对这一挑战。

非易失性内存（NVM）作为一种新型存储技术，引起了人们的关注。NVM 相比于 DRAM 具有更高的内存密度，并且支持内存中处理（PIM）技术，可以在内存中直接进行并行矩阵计算，从而避免了数据移动的开销。然而，不同类型的 NVM 解决方案各有优劣，如阻抗性随机访问存储器（ReRAM）在内存密度和能耗方面优于 DRAM，但易受可靠性问题的影响；相变随机存取存储器（PCRAM）具有较低的可靠性和较长的访问延迟等。

赛道存储器（RM）作为一种新兴的 NVM 技术，具有较好的性能、能耗、密度和可靠性平衡，被认为是解决内存墙问题的有前景的解决方案。在赛道存储器中，有许多研究探索了内存中处理矩阵计算的设计空间，但现有的解决方案受到内存核心和计算单元之间松散耦合所带来的数据传输开销的影响。

2 现有解决方法及问题

2.1 现有解决方法

在以往的研究中，已经提出了一些解决方案来在赛道存储器内进行矩阵计算。例如，SPIM 方案集成了基于 skyrmion 的计算单元在赛道存储器中，而 DW-NN 方案则采用了基于 CMOS 的替代计算单元。CORUSCANT 方案是最先进的方案之一，它在赛道存储器内部使用基于 CMOS 的计算单元，并利用一种称为横向读取的机制来降低计算延迟和提高可靠性。然而，这些解决方案中的计算单元需要频繁地从存储核心加载数据并将中间结果存回，导致了数据转换的开销，特别是在写入延迟和功耗较高的赛道存储器中，这一问题变得尤为突出。

2.2 存在问题

尽管赛道存储器具有潜力成为解决内存墙问题的有前景解决方案，但现有的解决方案仍然存在一些问题:

内存核心和计算单元之间的松散耦合导致了数据传输开销，降低了性能。

计算单元需要频繁地从存储核心加载数据和存储中间结果，增加了数据转换的开销，影响了系统的整体性能。

传统的基于 CMOS 的计算单元在处理复杂矩阵计算时可能效率不高，需要进行大量的数据传输和转换操作，增加了能耗和延迟。

因此，需要一种新的解决方案来克服这些问题，提高赛道存储器内部矩阵计算的效率和性能。

3 论文贡献点

3.1 针对流处理的新 RM 内处理架构

作者提出了一种全新的 RM 内处理架构，通过域壁纳米线构建计算单元和内部总线，以消除传统的数据转换开销，并通过简化的 RM 移位操作执行计算和数据传输，提高了系统的性能和吞吐量。

3.2 定制的矩阵计算 RM 处理器设计

设计了定制的 RM 处理器，用于流式执行整个矩阵计算，避免了中间结果的频繁存取和存储。优化了矢量操作的执行方式，提高了处理效率和性能。

3.3 基于域壁纳米线的总线设计

作者提出了基于域壁纳米线的新型总线（RM 总线），通过移位操作直接传输数据，减少了传统电磁转换带来的传输开销。

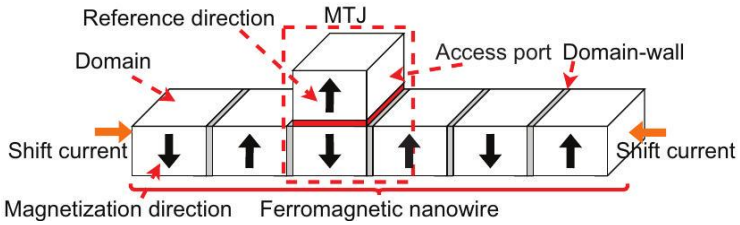
3.4 用于并行矩阵处理的架构设计

作者引入跨层设计，更好地利用 RM 的内部并行性，提高了系统的并行处理能力和效率，解决了 RM 读/写和移位操作之间的阻塞问题，进一步优化了并行性探索的效果。

4 StreamPIM 体系结构设计

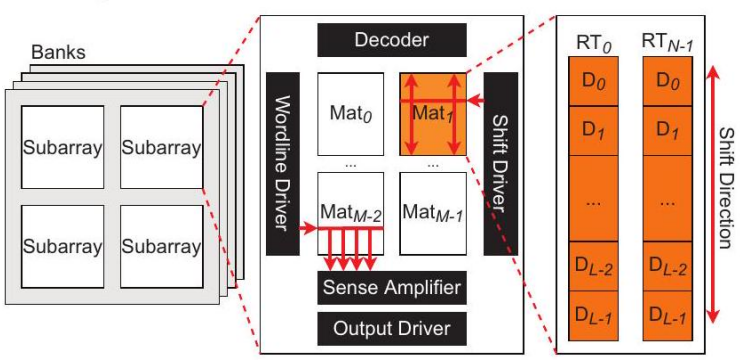
4.1 赛道存储器（RM）的总体架构

赛道存储器（RM）是一种基于自旋电子学的新型非易失性存储技术，也称为域壁内存（DWM）。RM 由多个域壁纳米线组成，每个域壁纳米线都包含多个存储域，每个域存储一个比特数据。域壁纳米线还包括访问端口，用于执行读写操作。读取操作通过在域壁纳米线上施加电压来执行，利用磁隧道结构（MTJ）中的磁阻来感知位值。写入操作则通过施加高写入电流来改变目标域的磁化方向。RM 的核心结构如下图所示。

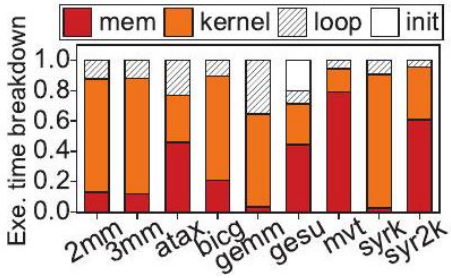


考虑到访问端口的成本，多个域共享同一个访问端口。因此，RM 需要执行移位操作将特定域移动到与访问端口对齐的位置，然后执行读取或写入操作。移位操作通过在纳米线两侧施加自旋极化电流来执行，使域和域壁沿电流方向同步移动。

RM 的内存组织如下图所示，包括多个银行，每个银行包含多个子阵列，每个子阵列由多个矩阵和外围电路组成。子阵列是为内存请求提供服务的基本单位，通过交错请求来同时提供请求服务。

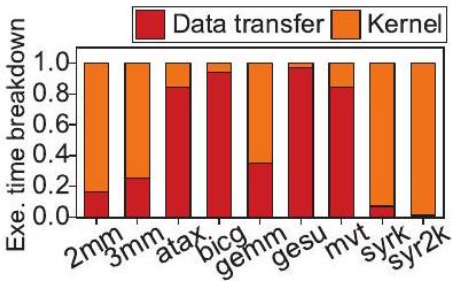


RM 集成的挑战在于当前基于 DRAM 的尺寸缩放面临多重挑战，而 RM 通过将多个位紧密集成到一个纳米线中，有望解决密度问题。然而，RM 无法缓解由处理器和内存之间的数据迁移引起的开销，这在下图中显示了在基于 CPU 和 GPU 的计算系统上的执行时间分解。

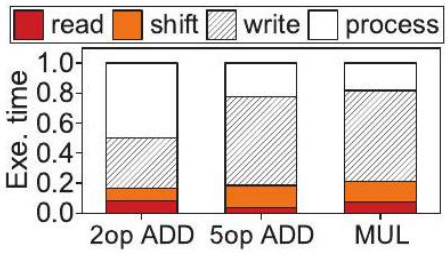


(a) CPU 上的执行时间分解。

(b) GPU 上的执行时间分解。



下图展示了 CORUSCANT 中支持的操作的执行时间和能量成本分解。RM 写操作占据了总执行时间的大部分，这表明 RM 的写入延迟和能量比读取操作要高得多。



(a) 执行时间分解。

(b) 能量成本分解。

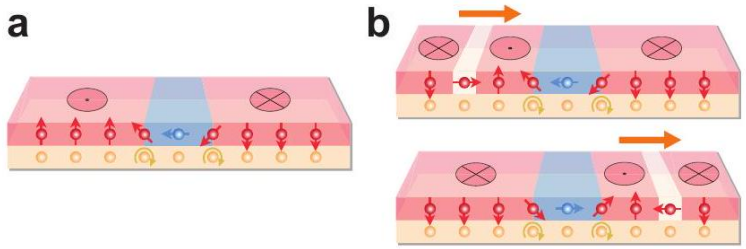
B. 赛道存储器中的处理

现有的 PIM 解决方案受到计算单元和存储阵列之间数据转换开销的影响。因此，如何消除数据转换成为提高 PIM 性能和能效的关键。

先前的研究通过在内存阵列内或附近放置定制的计算单元，构建了基于 RM 的处理内存 (PIM) 来解决这些挑战。DW-NN 提议将基于 CMOS 的算术单元放置在 RM 阵列附近，通过读取 RM 阵列中的数据并进行处理。DW-AES 将 AES 加密的部分实现为 RM 的移位操作，从而减轻了数据转换开销。然而，这些方法仍受到数据转换开销的影响。

最先进的工作 CORUSCANT 利用了 RM 的横向读取和横向写入机制来减轻 RM 读取开销。横向读取允许同时感知一组连续的领域中的数据，而横向写入机制则同时执行位移和写操作，进一步减少操作延迟。尽管 CORUSCANT 具有更低的延迟和更高的可靠性，但仍受到电磁转换开销的影响。为了量化这种影响，作者在 CORUSCANT 中执行了三个常见操作的实验，结果显示 RM 写操作占据了总执行时间的大部分。

综上所述，如何消除数据转换开销将成为提高基于 RM 的处理内存性能和能效的关键。



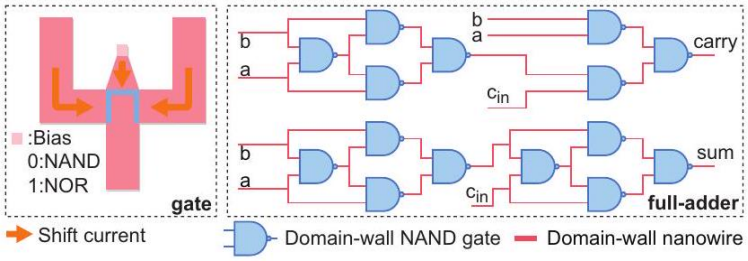
上图为领域壁纳米线逻辑机制。

4.2 StreamPIM 体系结构设计

理解了，我将逐个部分进行扩展和完善:

4.2.1 关键观察

最近的研究发现，领域壁纳米线可以用于实现布尔逻辑操作，从而为处理器内存集成提供了新的可能性。通过耦合磁性金属和重金属，可以实现领域壁纳米线的逻辑操作，从而实现了基于 RM 的逻辑门。作者描述了这一机制，并展示了领域壁纳米线逻辑结构的示意图：



领域壁纳米线的逻辑操作是基于磁性和电性相互作用的，其核心机制是利用磁性和电性的变化来实现逻辑门的功能。通过调节磁性金属和重金属之间的耦合强度，可以控制领域壁的磁性和电性状态，从而实现布尔逻辑操作。这一机制为处理器内存集成提供了新的思路和方法，可以显著提高系统的性能和能效。

4.2.2 StreamPIM 的内部架构

StreamPIM 的内部架构包括了处理器、内存和总线的集成。作者采用了传统的 RM 架构，但在内部添加了处理器和总线模块，实现了数据的流水线传输和处理

StreamPIM 架构的内部架构主要包括处理器、内存、总线和控制单元。处理器负责执行计算任务，内存存储数据，总线负责处理器和内存之间的数据传输，控制单元负责协调各个模块之间的工作。通过这种紧密集成的架构，作者能够实现高效的数据处理和传输，从而提高系统的整体性能和能效。

4.2.3 RM 处理器

RM 处理器专为矩阵计算而设计，包括标量加法、标量乘法和向量点积等操作。作者描述了 RM 处理器的结构和工作原理，以及其在 StreamPIM 架构中的作用。

RM 处理器采用了流水线和向量化指令等技术来提高计算效率，同时还采用了低功耗设计和异步电路等技术来降低能耗。通过这些优化，RM 处理器能够实现高性能和高能效的矩阵计算，为 StreamPIM 架构的整体性能提供了重要支持。

4.2.4 RM 总线

为了实现处理器和内存之间的数据传输，作者提出了一种新型的 RM 总线结构，利用领域壁纳米线进行数据传输。该总线结构消除了传统总线的数据转换开销，提高了数据传输效率和能效。

RM 总线采用了多通道和流水线传输等技术来提高数据传输带宽和速度，同时还采用了错误检测和纠正等技术来提高数据传输的可靠性。通过这些优化，RM 总线能够实现高效的数据传输，为 StreamPIM 架构的整体性能提供了重要支持。

4.2.5 矩阵设计

作者提出了一种新型的矩阵设计，利用保存赛道和传输赛道实现了非破坏性读取操作。该设计有效地解决了传统矩阵设计中的电磁转换问题，并提高了系统的整体性能。

新型矩阵设计采用了双端口存储器和读写分离等技术来实现非破坏性读取操作，同时还采用了数据压缩和编码等技术来提高数据存储和传输效率。通过这些优化，新型矩阵设计能够实现高效的数据存储和读取，为 StreamPIM 架构的整体性能提供了重要支持。

4.2.6 Subarray 内部数据流程

在 Subarray 内部处理 PIM 任务涉及到 mats、RM 总线和 RM 处理器。作者描述了数据从 mats 转移到 RM 处理器的过程，以及在处理器中的处理过程，最后数据返回到目标 mat 的过程。

Subarray 内部数据流程主要包括数据传输、处理和返回三个过程。数据传输过程通过 RM 总线实现处理器和内存之间的数据传输，数据处理过程通过 RM 处理器实现数据计算和处理，数据返回过程通过 RM 总线实现处理器和内存之间的数据返回。通过这些过程，作者能够实现高效的数据处理和传输，为 StreamPIM 架构的整体性能提供了重要支持。

5 实验评估

A. 实验设置

为了探索 RM 启用系统的全部设计空间，作者将现有的主存储系统在全系统模拟器 gem5 中替换为 RM 延迟模型。延迟模型是从流行的 NVM 内存建模工具 (RTSim 和 NVSim) 中衍生出来的。为了精确模拟 RM 中的 PIM 延迟，作者从头开始构建了一个周期精确的模拟器。作者的模拟器采用了基于实验室环境中的真实研究样本推导出的域墙逻辑结构模型。

作者设置了 6 个不同的计算平台进行评估:

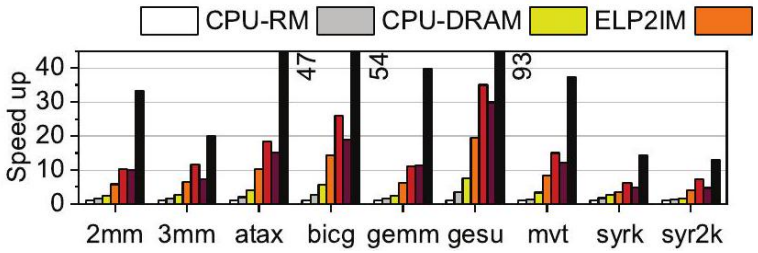
1. CPU-RM: 采用 CPU 和 RM 的传统计算平台; 2. CPU-DRAM: 传统的计算平台，采用 CPU 和 DDR4 DRAM, IO 速度为 2400MHz; 3. StPIM: StreamPIM (简称为 StPIM) 平台，具有 512 个 PIM 子阵列，并采用第 IV 节中提出的优化方案 (即分发和解除阻塞); 4. StPIM-e: 类似于 StPIM，但子阵列内的 RM 总线被传统的电气总线替换，这有助于分析 RM 总线的效率; 5. ELP2IM: 实现了最先进的基于 DRAM 进程的工作 [78]，通过串行比特级逻辑操作执行算术运算; 6. FELIX: 实现了最先进的基于 NVM 进程的工作 [19]，通过串行比特级逻辑操作执行算术运算; 7. CORUSCANT: 实现了最先进的基于 RM 进程的工作 [53]。

工作负载选择自 polybench [59]，共 9 个代表性的工作负载，涵盖了广泛的线性代数应用。这些工作负载包括不同的矩阵处理操作，包括矩阵、向量和标量的加法和乘法。

B. 总体性能

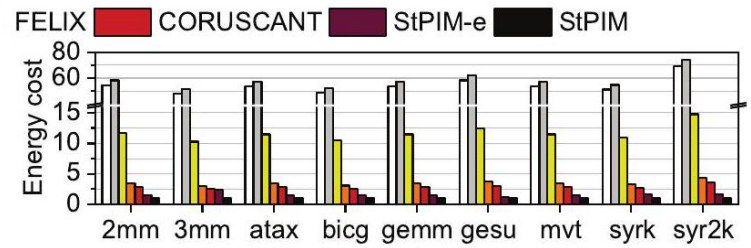
作者通过相对于 CPU-RM 的加速比来评估不同计算平台在各种工作负载下的性能。作者提供了 CPU-DRAM 的性能，以更好地说明基于 RM 的架构

的绝对性能。具体而言，由于具有较短的访问延迟和更高的带宽，CPU-DRAM 的性能平均提高了 1.5 倍，超过了 CPU-RM。



上图显示了不同计算平台的性能加速情况。例如，StPIM-e 相比 CPU-RM 平均提高了 12.7 倍的速度。此外，分布和解除阻塞优化有助于充分利用子阵列级并行性，从而实现了显著的性能提升。相比之下，虽然理想的 CORUSCANT 解决方案相对于 CPU-RM 基线有 15.6 倍的性能，但 StPIM 的平均超过了 CORUSCANT 2.5 倍。ELP2IM 和 FELIX 虽然也有性能提升，但受到了串行位级逻辑操作的限制。

C. 能量消耗



上图显示了不同平台的平均能耗。通过消除在内存和计算单元之间传输数据的能耗密集型过程，StPIM 相比于 CPU-DRAM 基准线实现了 58.4 倍的能效提升。与 FELIX 和 CORUSCANT 相比，StPIM 的能效分别提高了 3.5 倍和 2.8 倍。此外，由于 RM 处理器的能耗比基于 CMOS 的单元更低，StPIM-e 相比于 FELIX 和 CORUSCANT 分别提高了 2.2 倍和 1.8 倍。然而，由于电

气总线的能效低于 RM 总线，StPIM-e 的能耗平均比 StPIM 多 1.6 倍。

D. 敏感性测试

作者评估了 PIM 对 StPIM 性能的影响，包括 PIM 子阵列数量和优化方案。

针对 PIM 子阵列数量的测试结果显示，增加子阵列数量可以显著提高性能，但随着数量的增加，性能增益会逐渐减少。

优化方案方面，通过分布和解锁优化，作者实现了显著的性能提升，特别是解锁优化对性能的提升非常显著。

在段大小方面的测试结果显示，虽然缩小段大小会增加传输时间，但对于整体性能和能耗的影响是微小的。

E. 端到端性能

作者展示了两个深度神经网络应用程序在 StreamPIM 上的性能改进，包括 MLP 和 BERT。通过加速关键矩阵计算部分，StreamPIM 实现了显著的性能提升。

F. 制造过程

在作者的评估中，作者采用了 CORUSCANT 兼容的配置（即 32 nm），在这种配置下，加法和乘法操作的能量消耗分别为 0.03pJ 和 0.18pJ，与基于 CMOS 的算术单元相比，显示出了出色的能量效率。

G. 面积开销

作者通过估算不同组件中域壁存储单元（RM mats）的数量来分析作者设计的面积分布情况。结果表明，作者的设计在性能和面积消耗之间取得了良好的平衡。RM 总线和 RM 处理器仅占 RM 设备总面积的 1.8% 和 0.1%。另一方面，传输轨道的面积开销仅为总面积的 3.1%，而添加的控制逻辑的面积开销与根据之前的研究 [82] 的银行面积相比仅约为 1.0%。

这些评估结果表明，StPIM 架构在性能、能效、制造成本和面积开销之间取得了良好的平衡，展现了其在各种计算任务中的潜力和优势。

6 启发

通过学阅读习论文《StreamPIM: Streaming Matrix Computation in Racetrack Memory》，我对 StreamPIM 这一在赛道内存中进行流矩阵计算的新技术有了更深入的了解。StreamPIM 针对内存墙问题提出了一种创新的解决方案，这个问题限制了处理器与内存之间数据传输的速度，从而限制了整体计算性能的提升。通过紧密耦合内存核心和计算单元，并利用赛道内存独特的移位操作来消除传统读写操作带来的数据转换开销，StreamPIM 有效地克服了内存墙问题。这个创新让我深刻认识到，对于现有问题的深入分析和创新思维是推动技术进步的关键。

此外，StreamPIM 对处理内存储 (Processing-in-Memory，PIM) 架构进行了优化。它直接从域壁纳米线构建矩阵处理器，并通过基于域壁纳米线的总线设计来提高数据迁移效率。这种设计不仅提高了性能，还节省了能耗。这让我意识到，针对特定计算任务定制硬件架构，可以显著提升整体系统效率。

值得思考的是，StreamPIM 对未来计算架构的影响。随着数据密集型应用的不断增加，传统的冯·诺依曼架构已经难以满足需求。StreamPIM 提出的解决方案，可能会成为未来计算架构设计的一个趋势。

通过学习 StreamPIM，我对硬件设计有了更深入的理解。如何将硬件设计与软件优化相结合，以实现更高效的数据处理，是当前和未来技术发展的一个重要方向。

并且从论文中可以看出，StreamPIM 技术的发展也强调了跨学科合作的重要性。从材料科学到计算机科学，从硬件设计到软件应用，多个领域的知识和技术被整合在一起，共同推动了这一创新技术的发展。