# FlatAttention: Dataflow and Fabric Collectives Co-Optimization for Efficient Multi-Head Attention on Tile-Based Many-PE Accelerators

**Chi Zhang** chizhang@iis.ee.ethz.ch

Luca Colagrande, Renzo Andri, Thomas Benz,
Gamze Islamoglu, Alessandro Nadalini,
Francesco Conti, Yawei Li and Luca Benini


@pulp_platform

pulp-platform.org

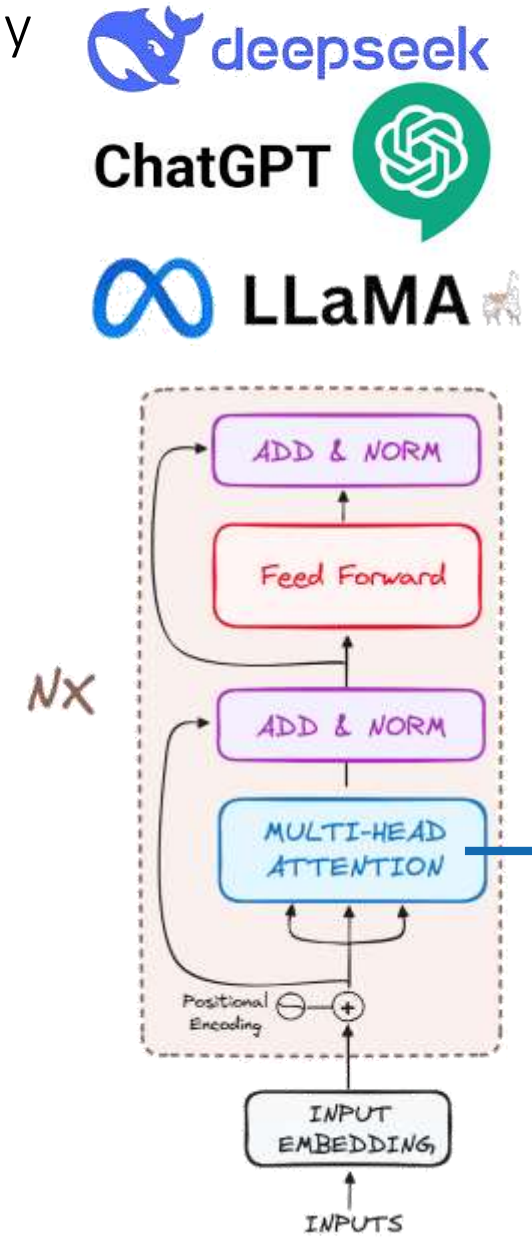
youtube.com/pulp_platform

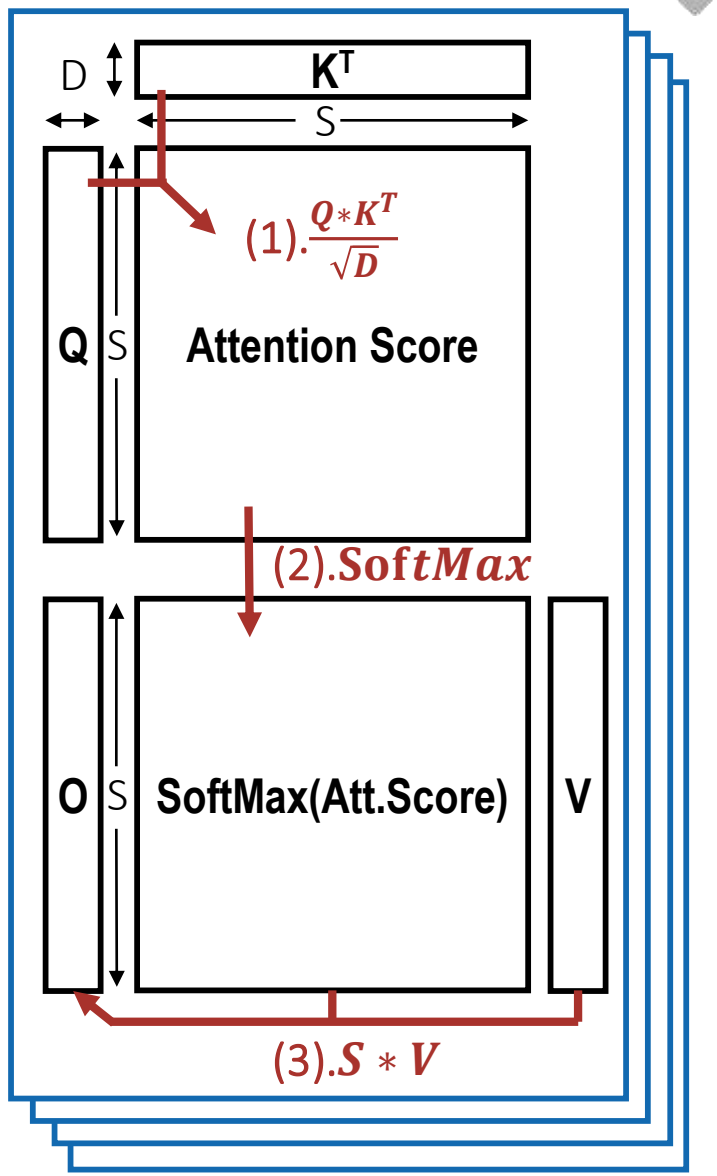
**PULP Platform**

Open Source Hardware, the way it should be!

# The Attention Bottleneck in Transformer-Based LLMs

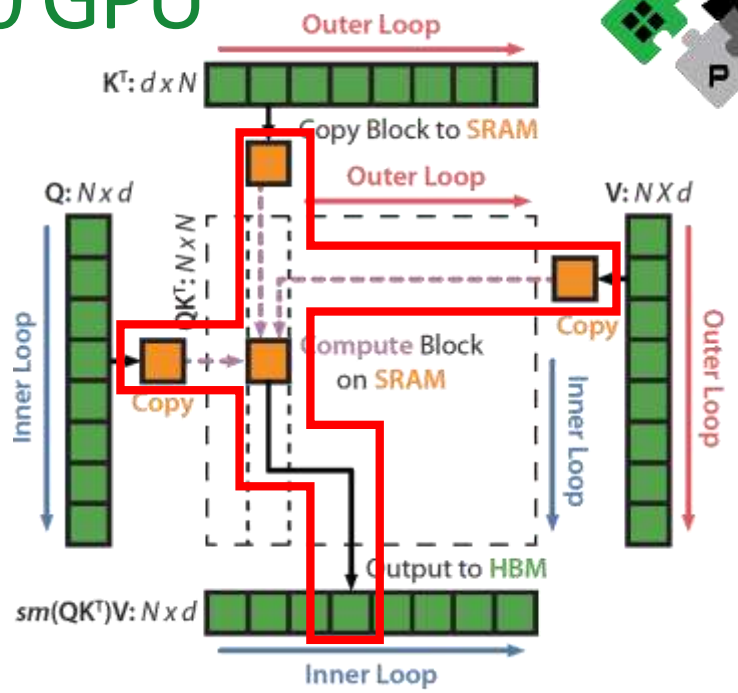
- **LLM workloads** have become increasingly impactful
  - DeepSeek-V3, ChatGPT, Llama
- Dominating LLM are **Transformer-based**
- The attention bottleneck
  - **Multi-Head Attention (MHA)** exhibits quadratic complexity over sequence length. $O(S^2)$
  - Most operations in MHA are bottlenecked by memory accesses[1]
  - Performance bottleneck for
    - LLM inference
      - Especially during long context prefill
    - LLM training

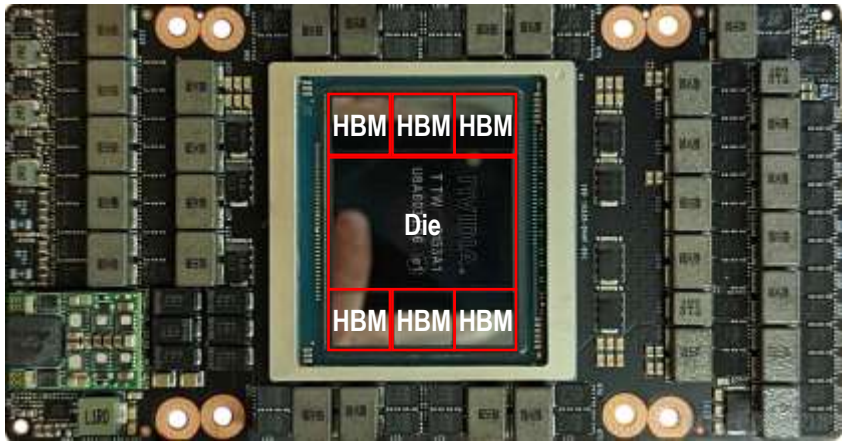


# SoA Solution: FlashAttention-3 on Nvidia H100 GPU

- The attention bottleneck has driven **optimizing MHA dataflows** on the dominant AI hardware platform – **Nvidia GPU**
- **FlashAttention**[2] – most widely adopted solution
  - Efficiently fuses MHA microkernels
  - FlashAttention-**2**[8]: algorithmic optimizations
  - FlashAttention-**3**[9]: leverages asynchronous execution
- Drawback of SoA solution: FlashAttention-3 on H100
  - Suboptimal performance
    - No more than 75% utilization on H100
  - High cost of hardware platform
    - 814 mm² die on TSMC's 5nm process node
    - 6 HBM3 stacks – up to 50% total cost
    - 700W Thermal Design Power(TDP)

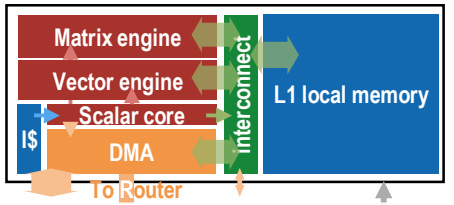


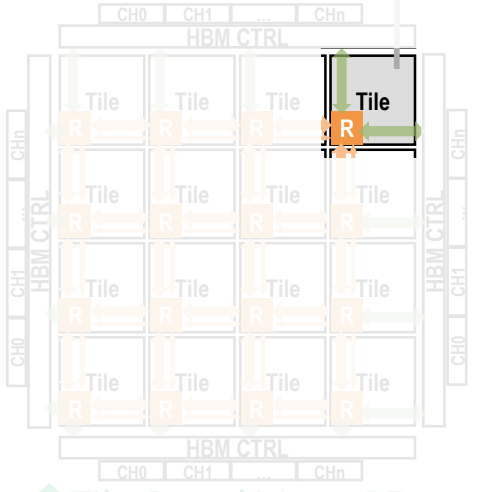




https://www.servethehome.com/nvidia-h200-launched-with-141gb-of-hbm3e-at-sc23/

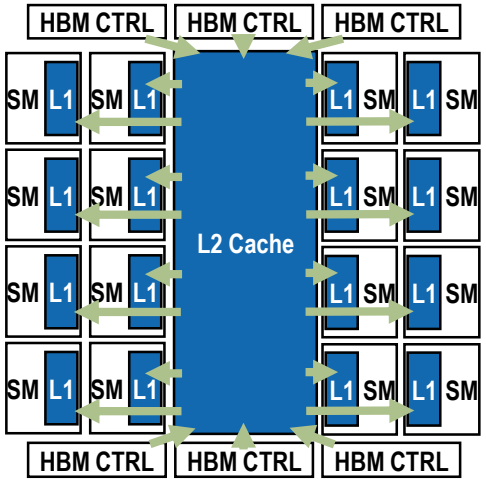
# Emerging AI Accelerator: Tile-Based Many-PE Architecture

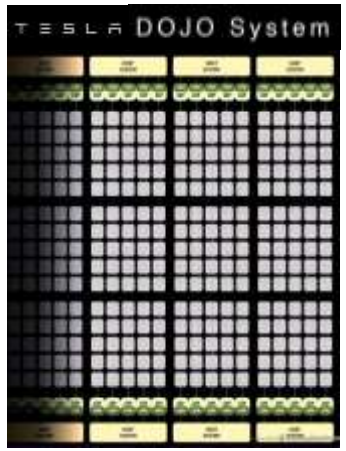
- Tile-based many-PE accelerators
  - Meshes of compute tiles
    - Matrix, vector and scalar engines
    - Local explicitly managed memories
  - Scalable NoC, Main memory at die boundaries
- Favors silicon efficiency and scalability
  - A dense compute tile placement
  - Software-controlled partitioned memory hierarchy
  - Eliminates L2 on-chip cache for area efficiency
- Examples
  - Tesla's Dojo, Tenstorrent's Blackhole, Huawei's Ascend910, Meta's MTIA

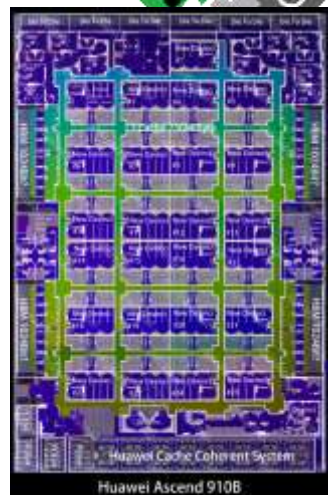


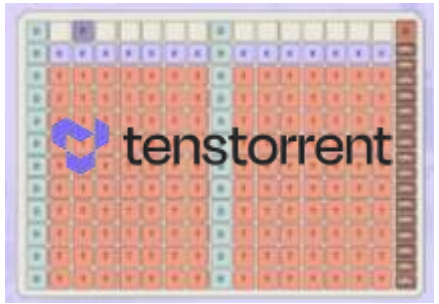



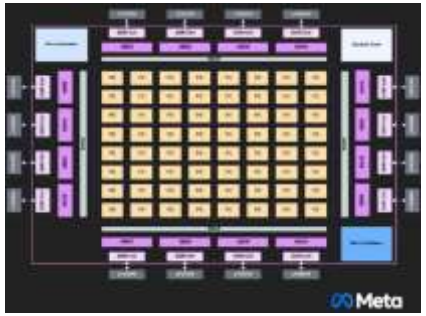

Tile-Based Many-PE
vs Nvidia GPU Arch



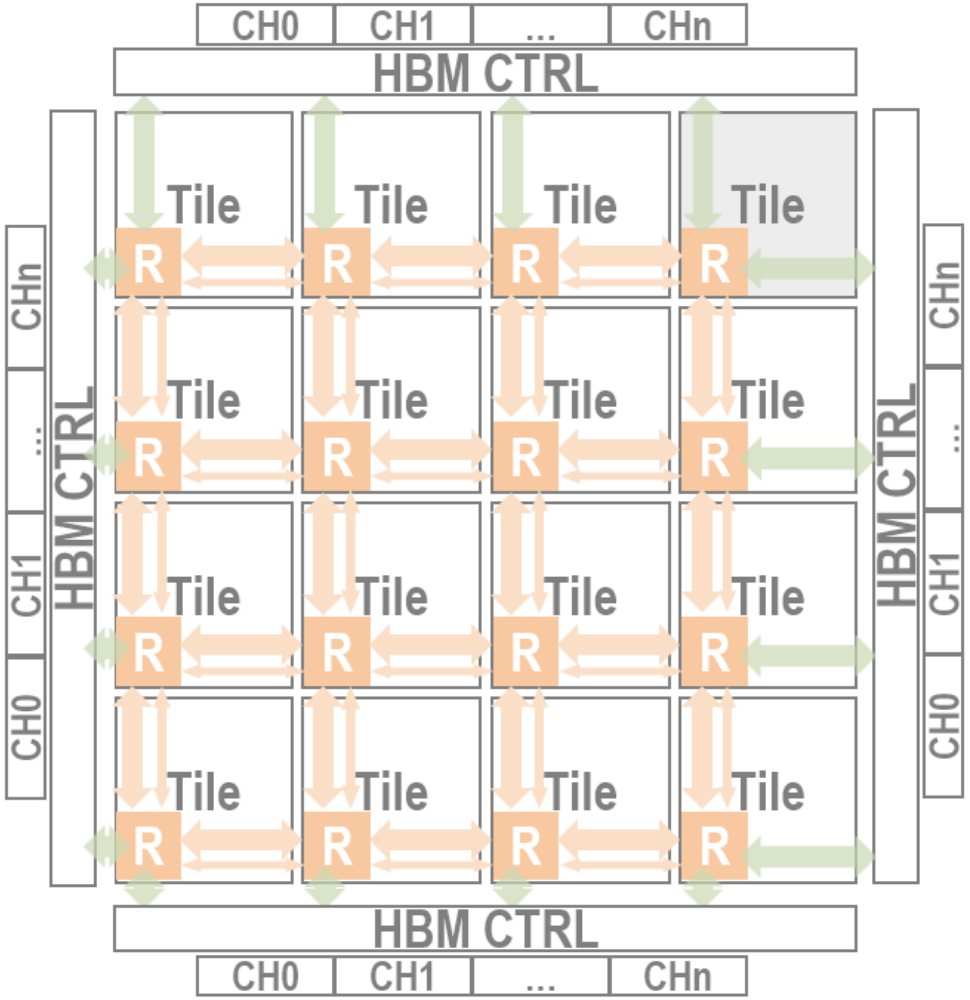





https://chipsandcheese.com/p/hot-chips-34-teslas-dojo-microarchitecture
https://www.tomshardware.com/tech-industry/artificial-intelligence/huaweis-homegrown-ai-chip-examined-chinese-fab-smic-produced-ascend-910b-is-massively-different-from-the-tsmc-produced-ascend-910
https://hc2024.hotchips.org/assets/program/conference/day1/88_HC2024.Tenstorrent.Jasmina.Davor.v7.pdf
https://www.servethehome.com/meta-ai-acceleration-in-the-next-gen-meta-mtia-for-recommendation-inference-risc-v/

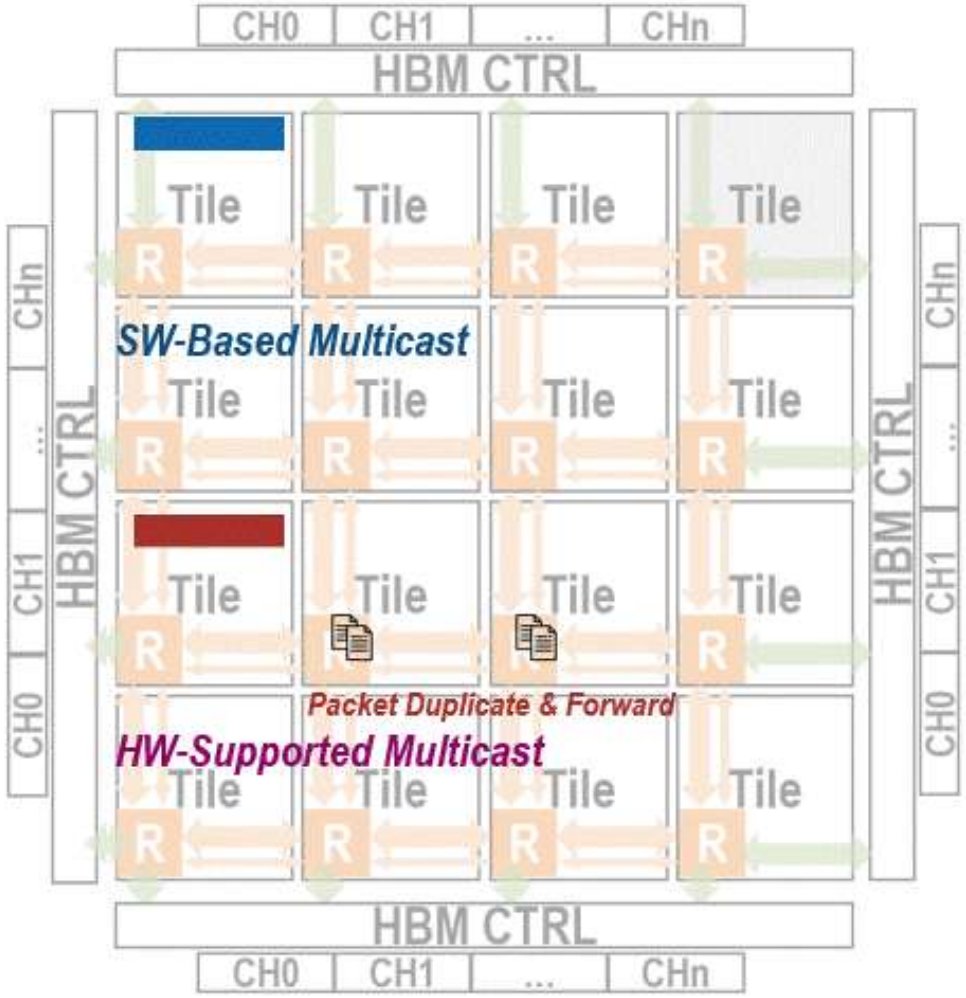
# Challenge: MHA Dataflow on Tile-Based Many-PE Architecture

- The goal is to
  - Achieve high utilization of the tiles' matrix engines
  - Minimize energy-hungry off-chip accesses
- Architecture and dataflow need to be co-explored
  - Dataflow leverages HW feature, e.g., collective primitives on NoC fabric



# Challenge: MHA Dataflow on Tile-Based Many-PE Architecture

- The goal is to
  - Achieve high utilization of the tiles' matrix engines
  - Minimize energy-hungry off-chip accesses
- Architecture and dataflow need to be co-explored
  - Dataflow leverages HW feature, e.g., collective primitives on NoC fabric
    - Accelerate inter-tile collective communications
  - Determine optimal architecture design parameters alongside dataflow exploration

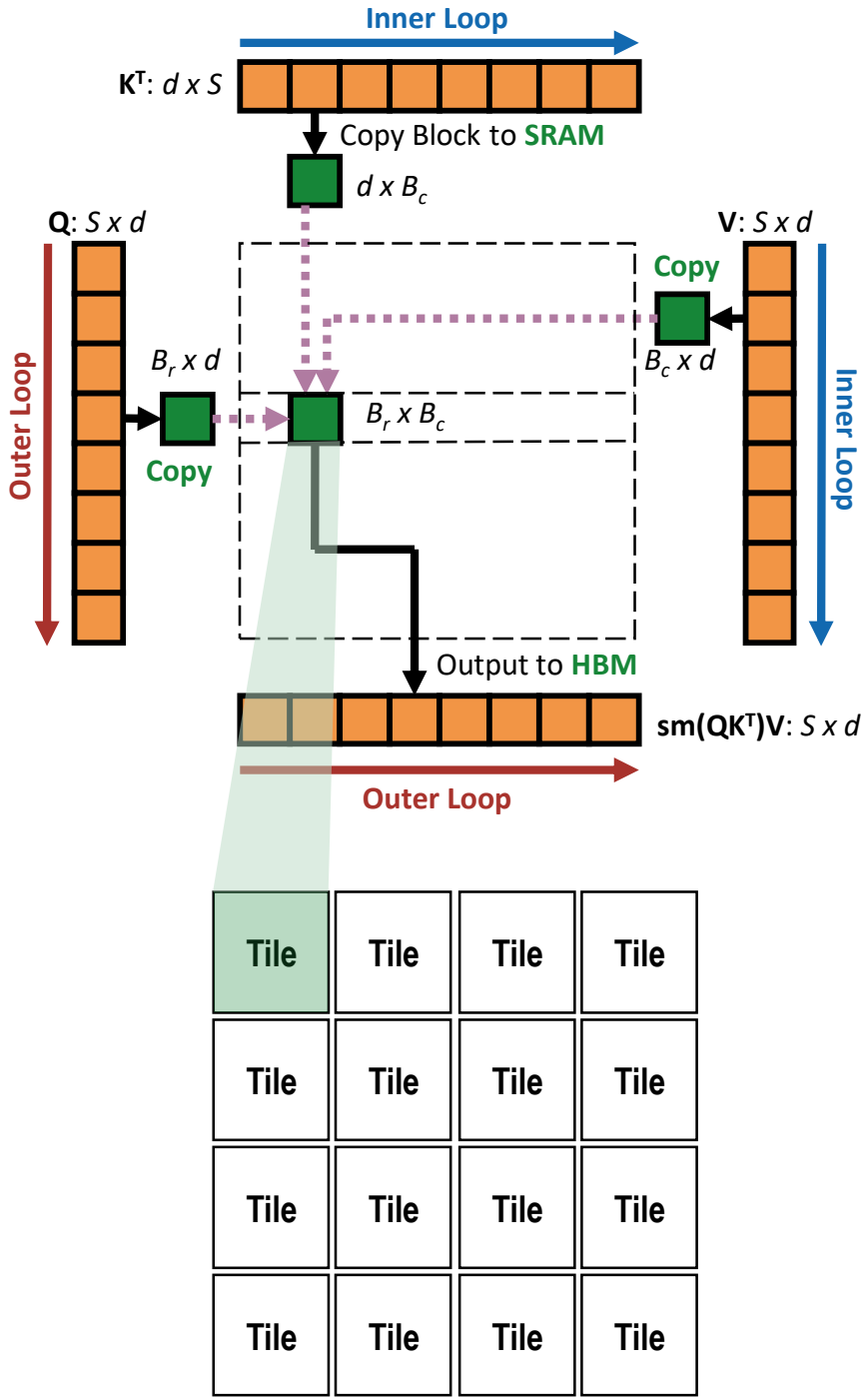


# Our Contributions


- Modeling and simulation framework for tile-based architecture template (**SoftHier**)
  - Estimating the performance of a large set of tile-based accelerators
  - Enabling the co-design of network collective primitives
- **FlatAttention** MHA dataflow
  - Leverages collective primitives on the NoC fabric and minimizes off-chip HBM traffic
  - Co-exploration of accelerator architecture and FlatAttention parameters
- **Key Results**
  - FlatAttention vs. FlashAttention-3 on tile-based accelerator
    - Up to 89.3% utilization, 4.1× speedup, 16× HBM traffic reduction
  - Algorithm-architecture co-exploration
    - Peak FP16 Perf(1024 TFLOPS) matches H100 (989 TFLOPS)
    - 40% less available HBM BW, 1.8× die size reduction (estimated at TSMC 5nm)
    - Up to 1.3× utilization speedup (FlatAttention on *BestArch* vs. FlashAttention-3 on H100)

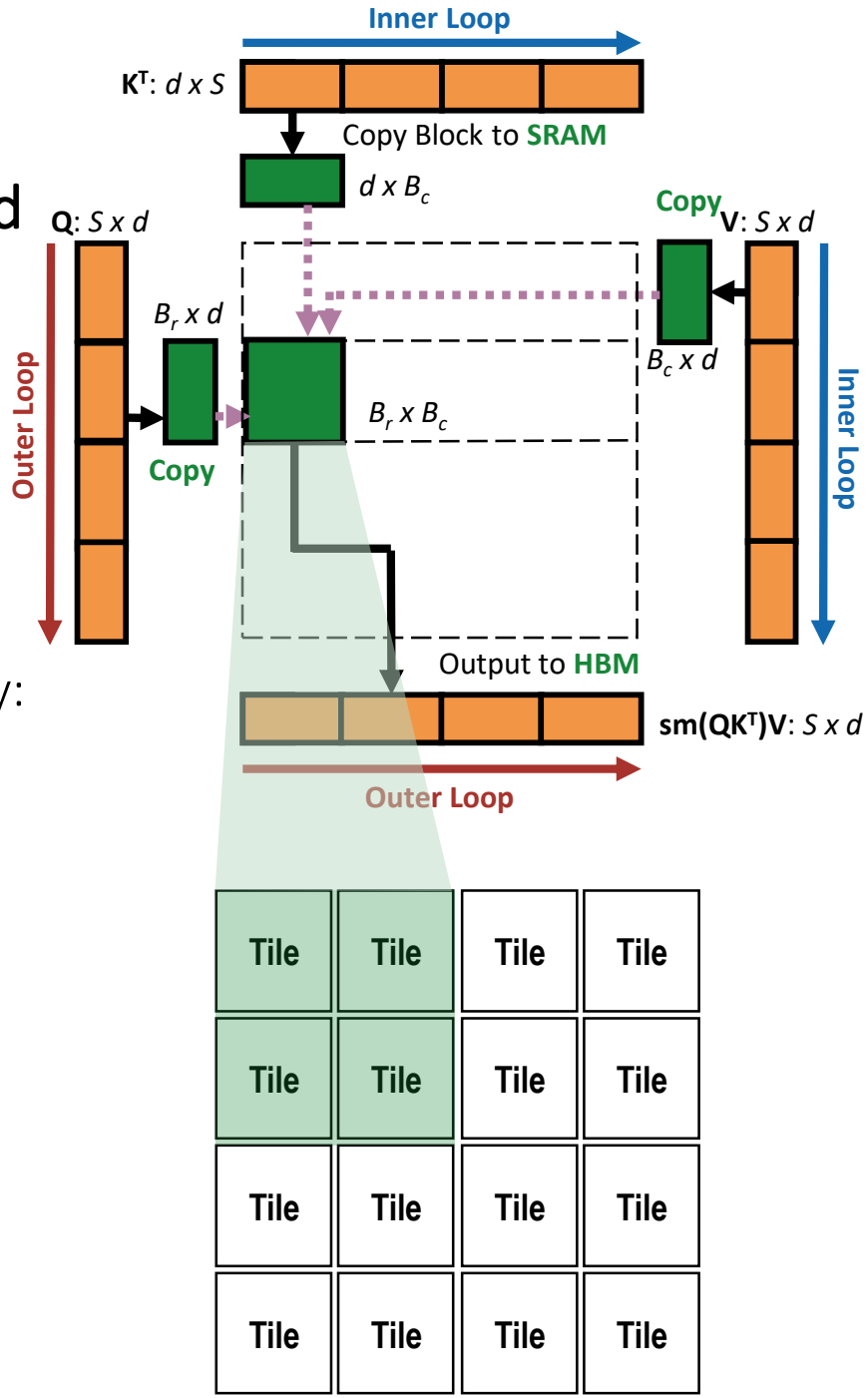
# FlatAttention Motivation

- Analysis start from FlashAttention
  - Fuse microkernels of each head attention
  - MHA workload is partitioned to tiles over:
    - Batch size, number of heads, output sequence dimension
  - Every tile processes independently
    - Every tile need to access in HBM
    - No communication between tiles is required
  - Results in an HBM I/O complexity of
    - $\mathrm{IO} = 2 \cdot H \cdot B \cdot D \cdot S \cdot \left(1 + \frac{S}{M}\right)$
    - Sequence length S, head dimension D, number of heads H, batch size B and block size M:=Br=Bc



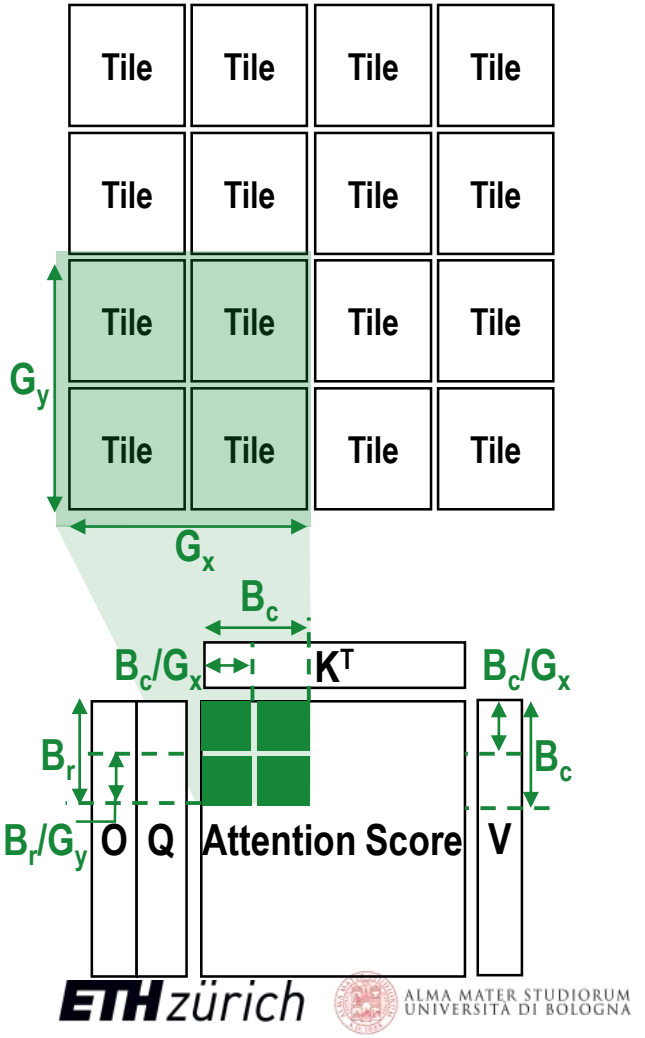
# FlatAttention Motivation

- FlatAttention – Redefine how MHA is parallelized
  - Leverages multiple tiles as a unified entity
  - Process an MHA block of a significantly larger size
    - The aggregate L1 memory of a group of tiles
    - Collectively store the block
  - When N tiles are grouped together, HBM I/O complexity:
    - $\mathrm{IO} = 2 \cdot H \cdot B \cdot D \cdot S \cdot \left(1 + \frac{S}{\sqrt{N} \cdot M}\right)$
  - More tiles are grouped, less HBM accesses needed



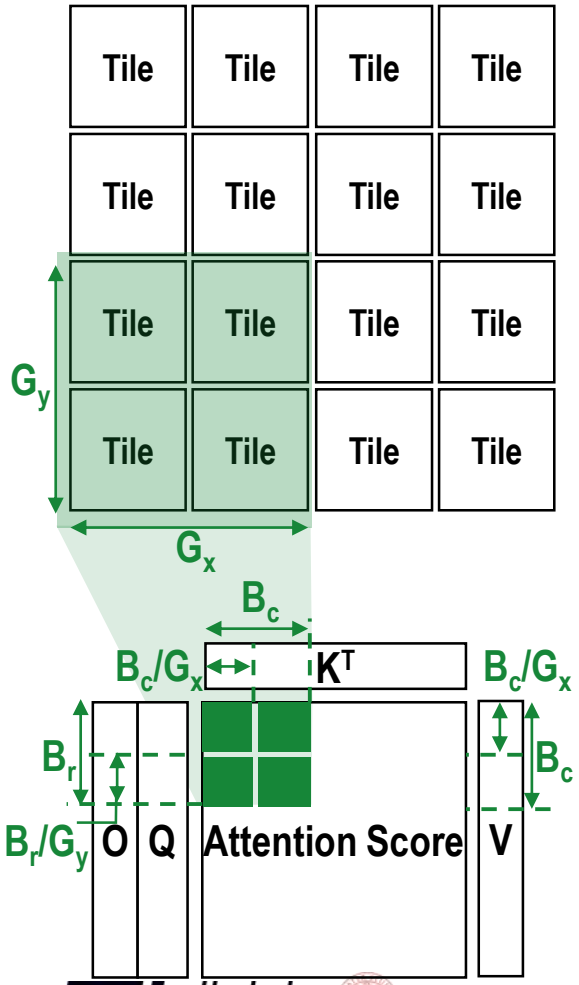
# Detailed FlatAttention Dataflow

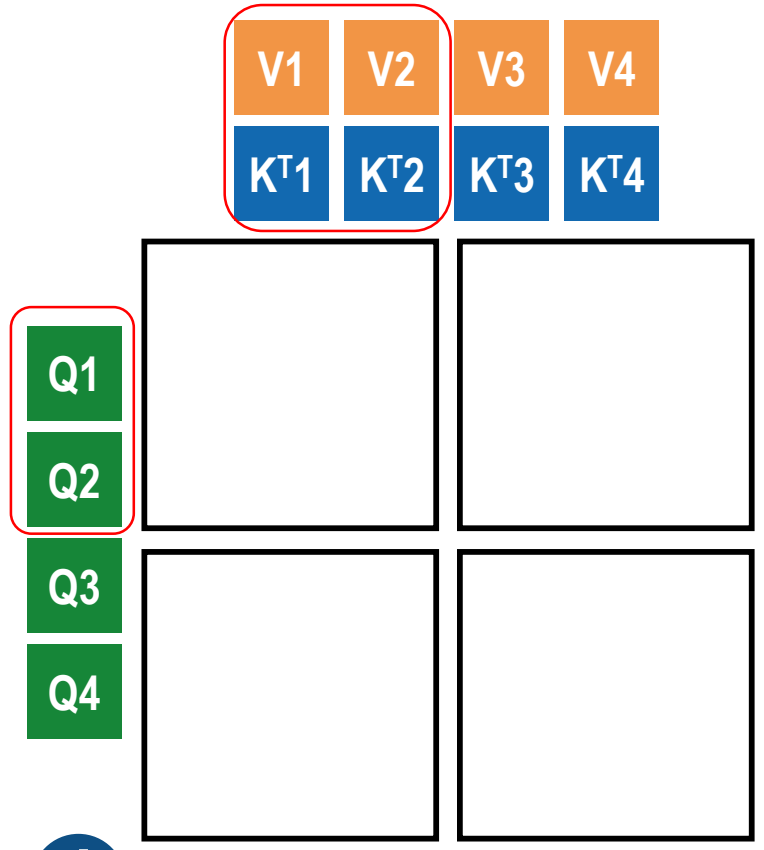
- Distinct data movement patterns within the tile group



# Detailed FlatAttention Dataflow

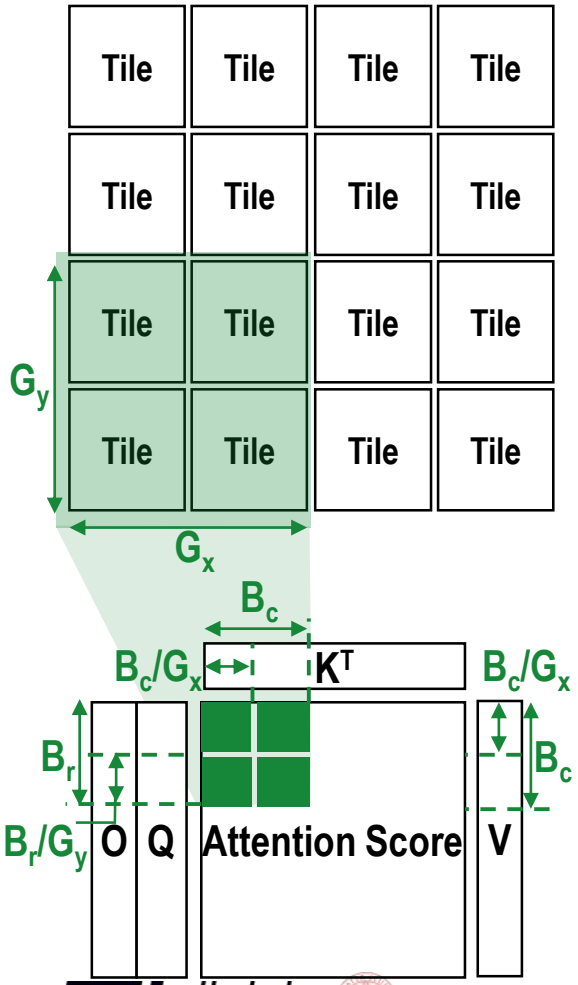
- Distinct data movement patterns within the tile group

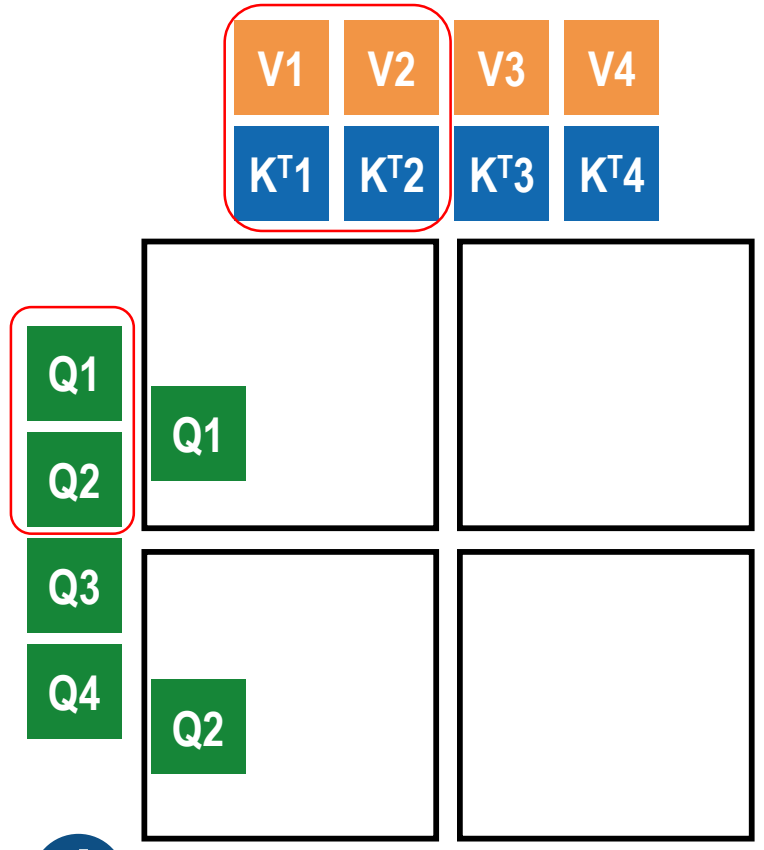


# Detailed FlatAttention Dataflow

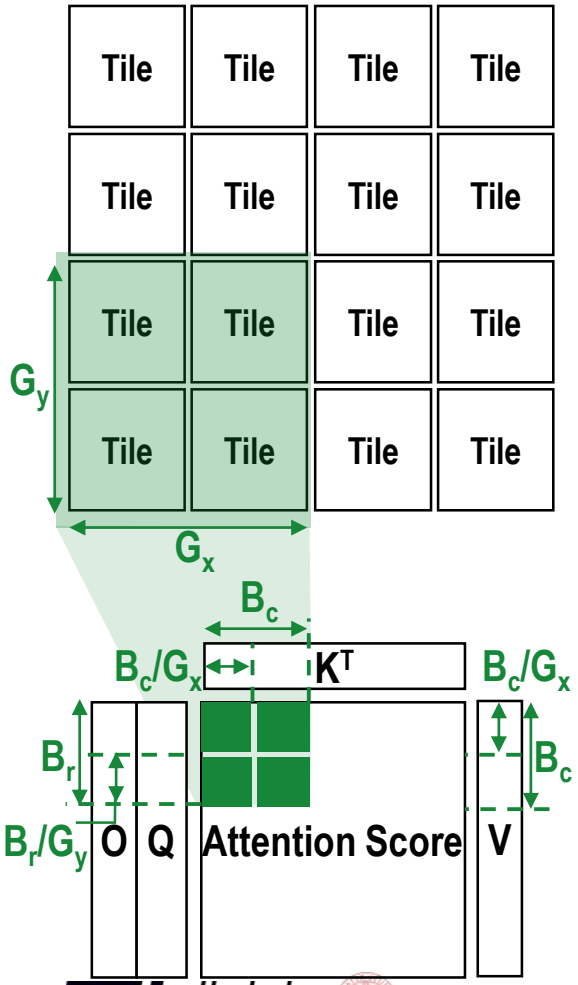
- Distinct data movement patterns within the tile group

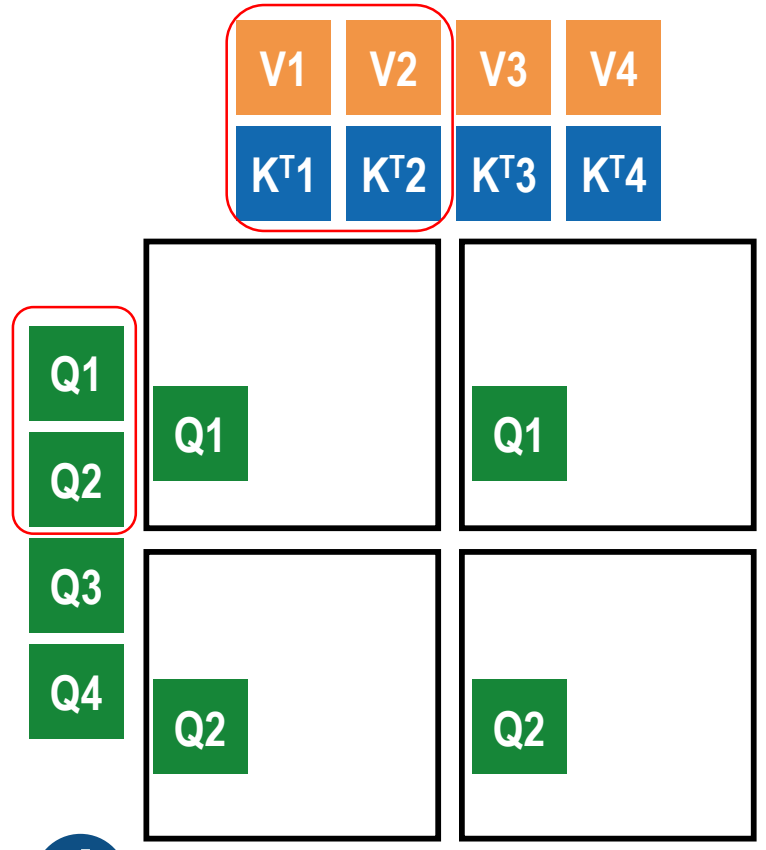


# Detailed FlatAttention Dataflow

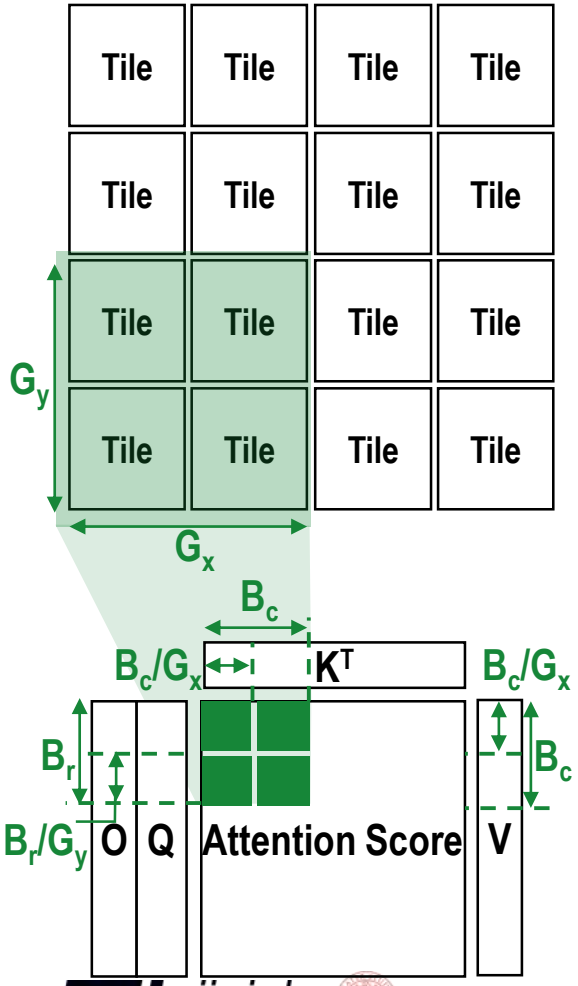
- Distinct data movement patterns within the tile group

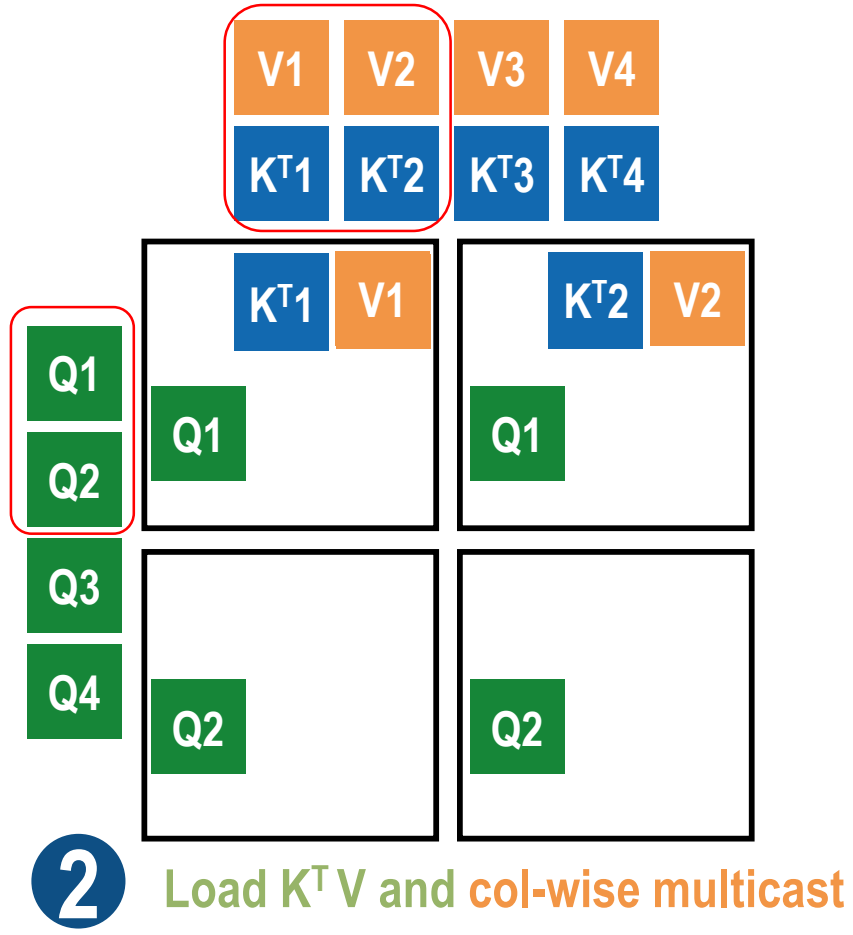


# Detailed FlatAttention Dataflow

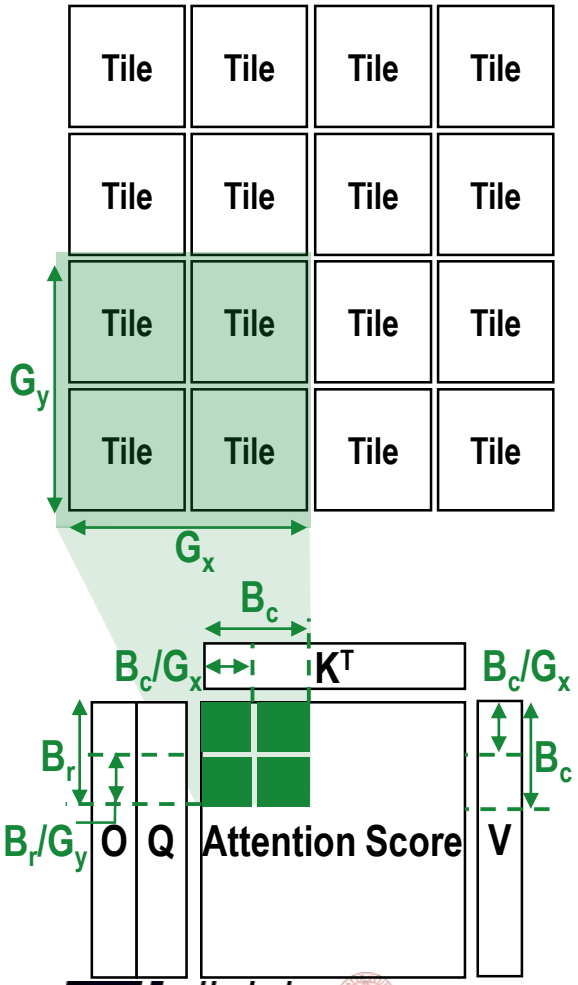
- Distinct data movement patterns within the tile group

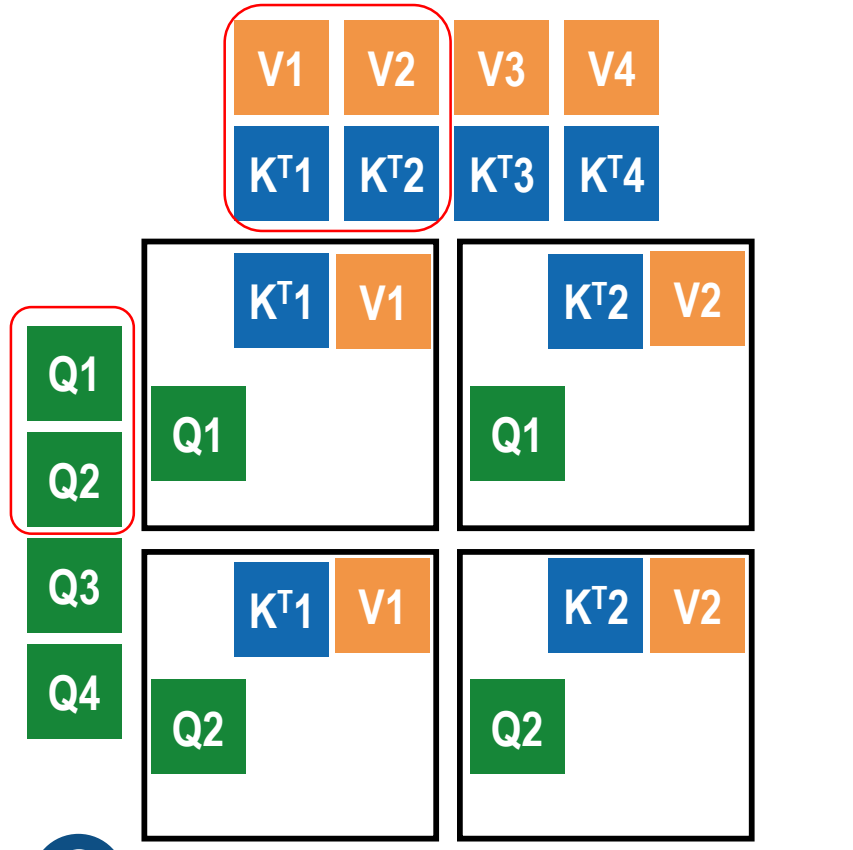


# Detailed FlatAttention Dataflow

- Distinct data movement patterns within the tile group

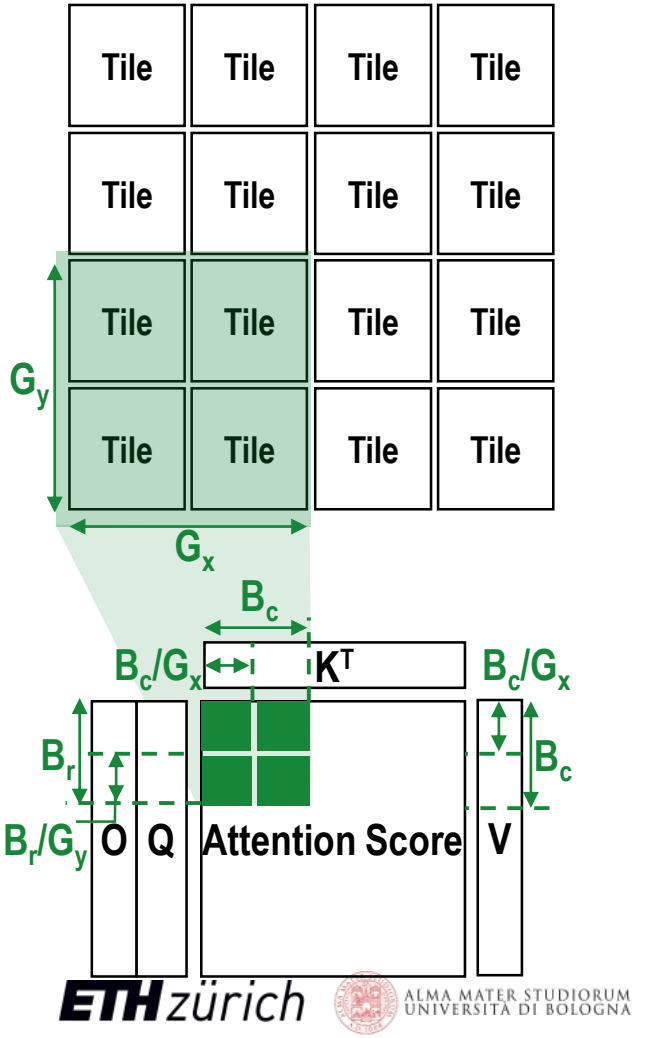


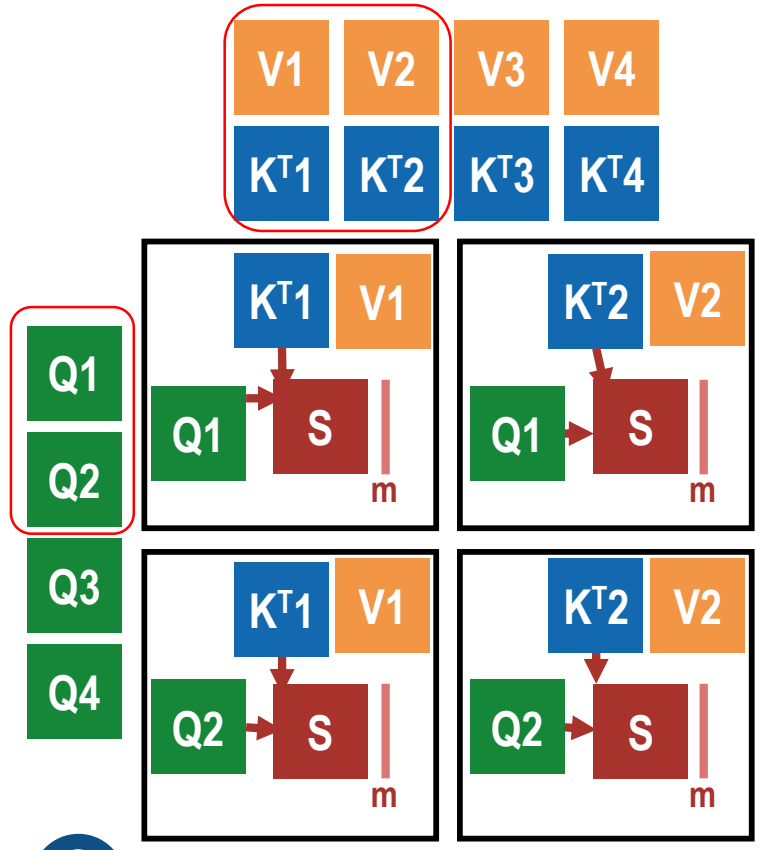




2 Load $K^T$ V and col-wise multicast

# Detailed FlatAttention Dataflow

- Distinct data movement patterns within the tile group

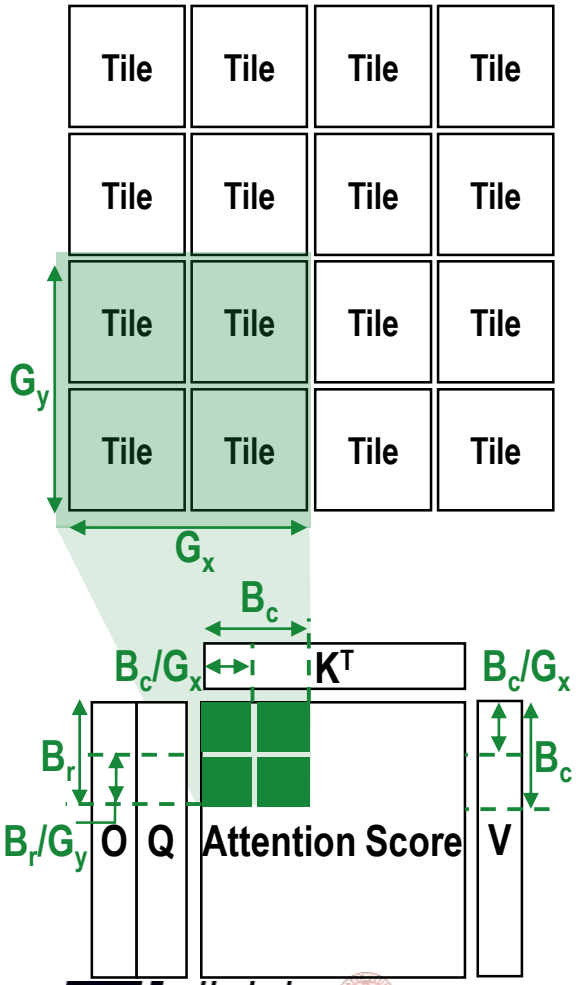


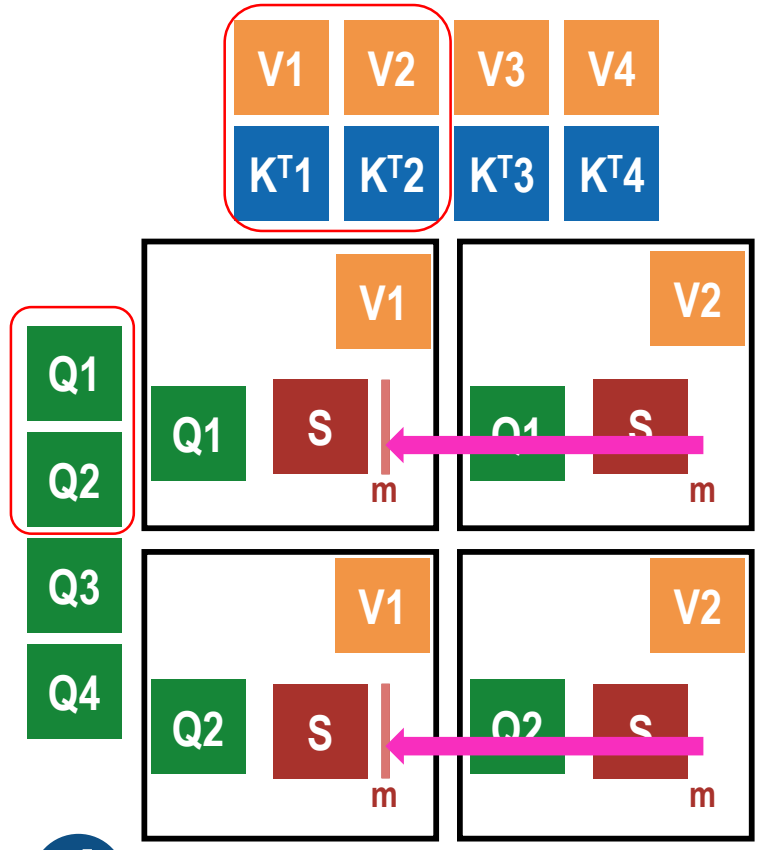




3 Calculate att.score, scaling, and local rowmax m

# Detailed FlatAttention Dataflow

- Distinct data movement patterns within the tile group

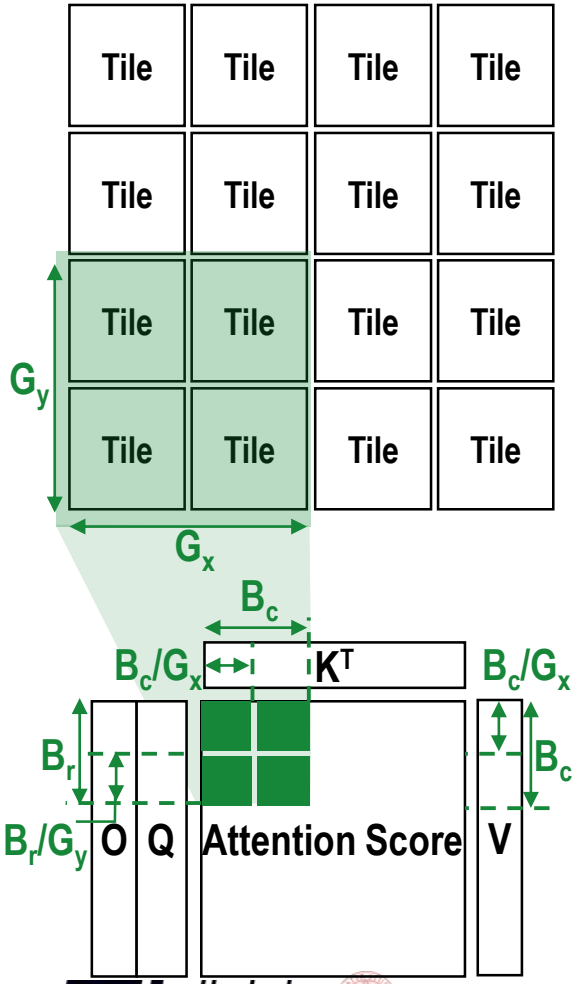


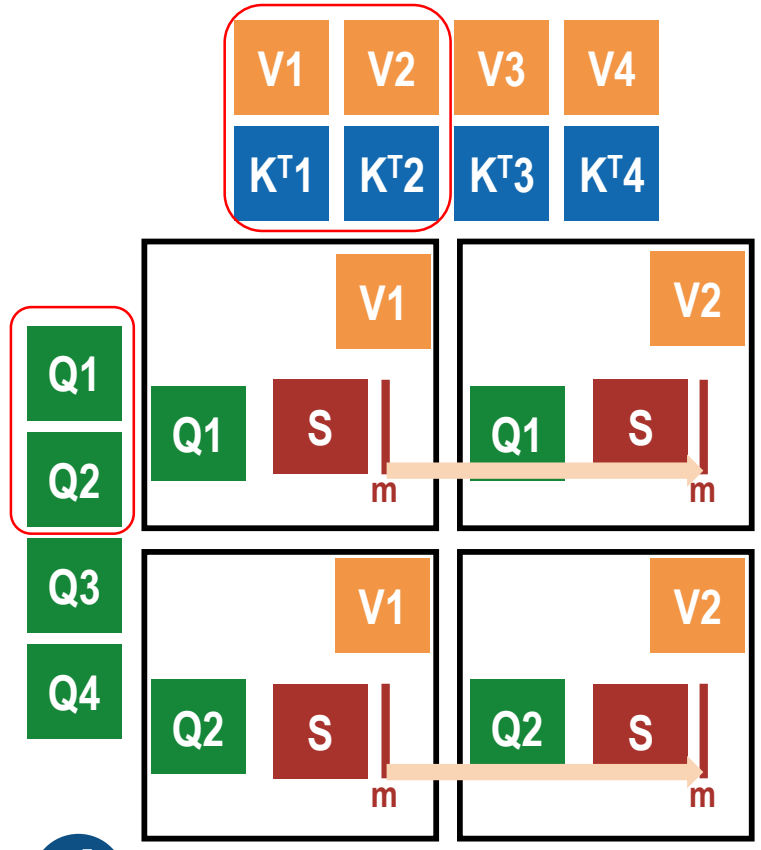




4 Row-wise red.max m and multicast -> global max

# Detailed FlatAttention Dataflow

- Distinct data movement patterns within the tile group

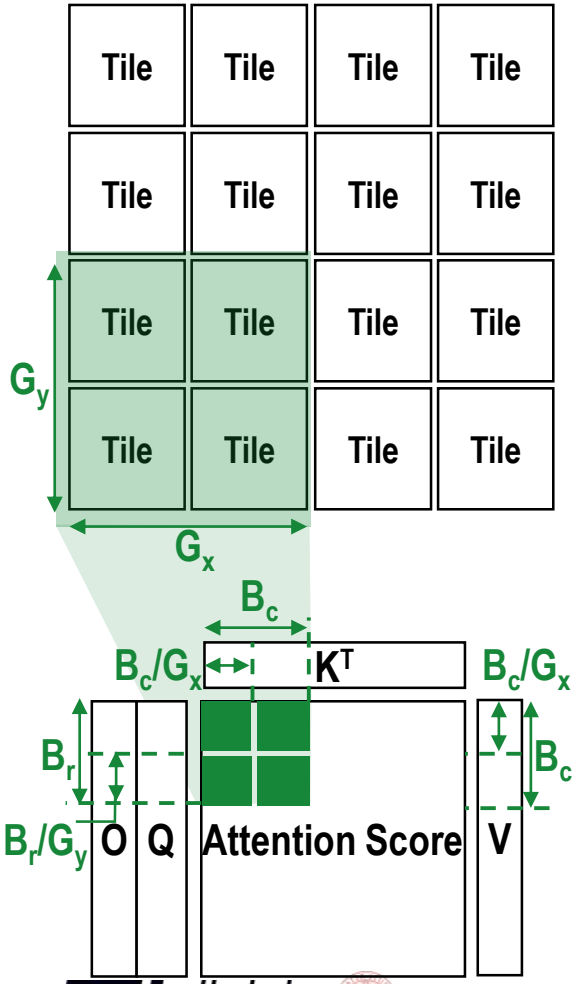


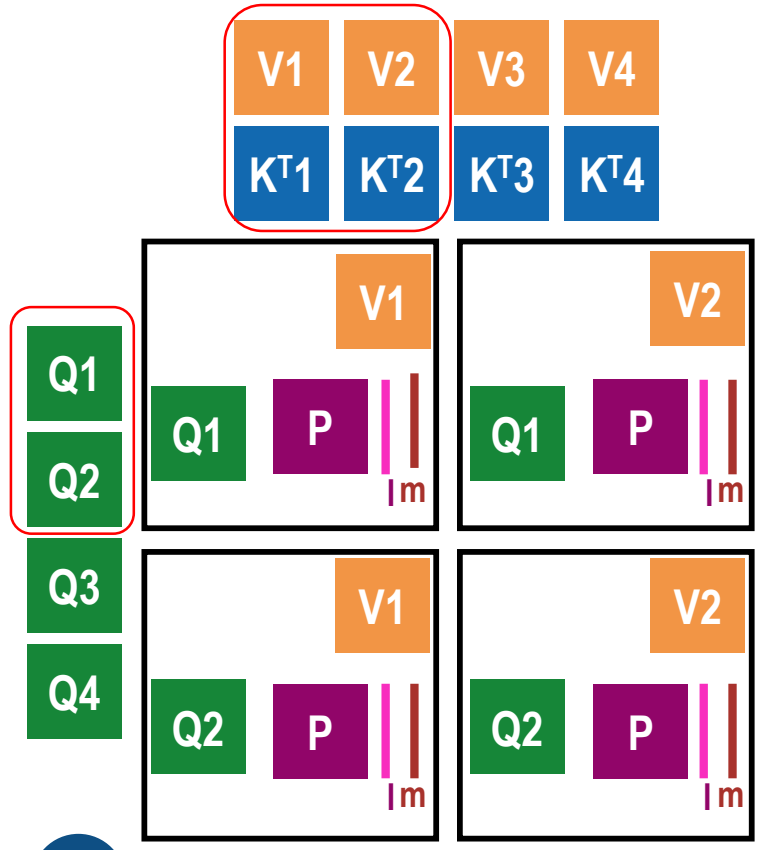




4 Row-wise red.max m and multicast -> global max

# Detailed FlatAttention Dataflow

- Distinct data movement patterns within the tile group

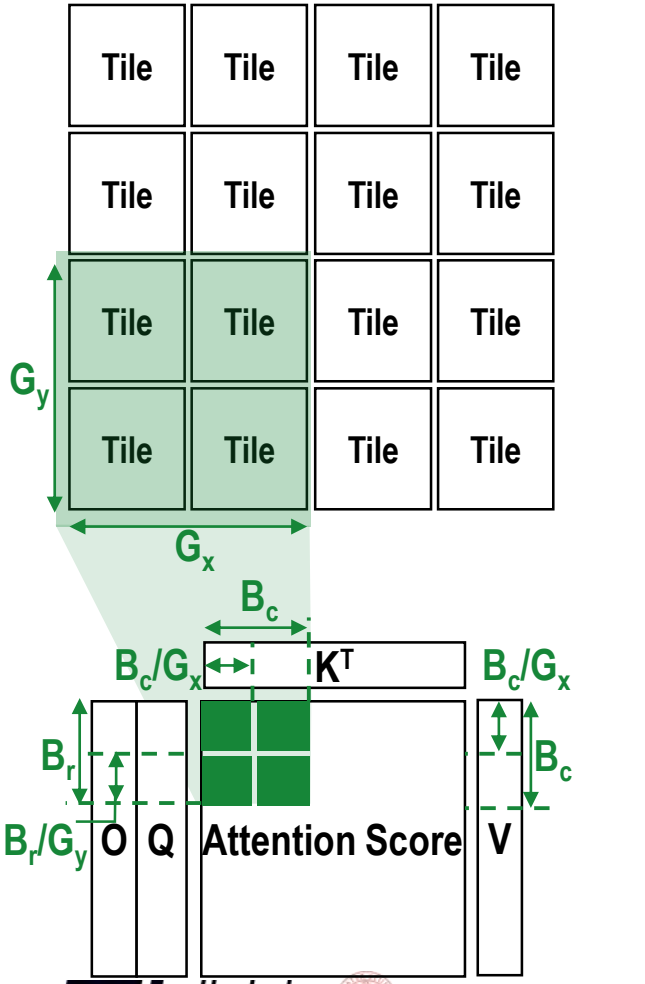


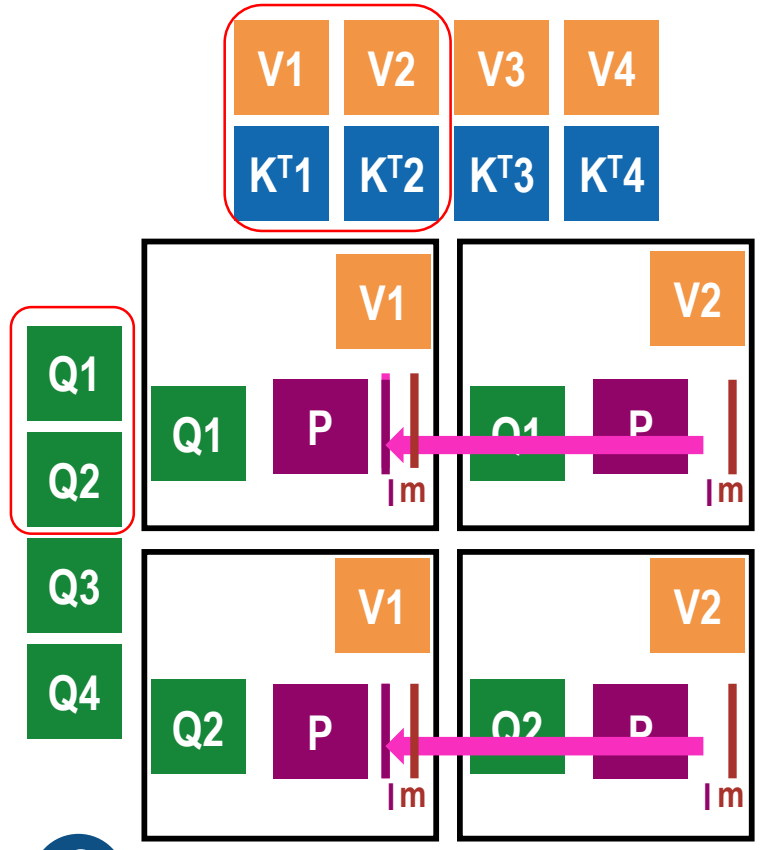




5 Calculate softmax numerator P and denominator l

# Detailed FlatAttention Dataflow

- Distinct data movement patterns within the tile group

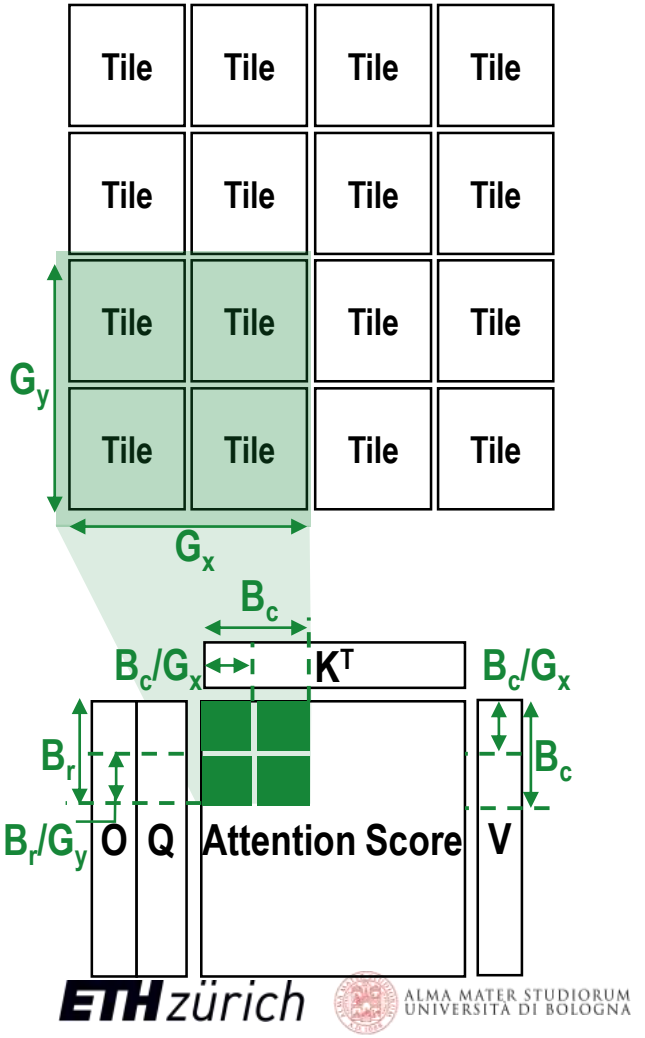


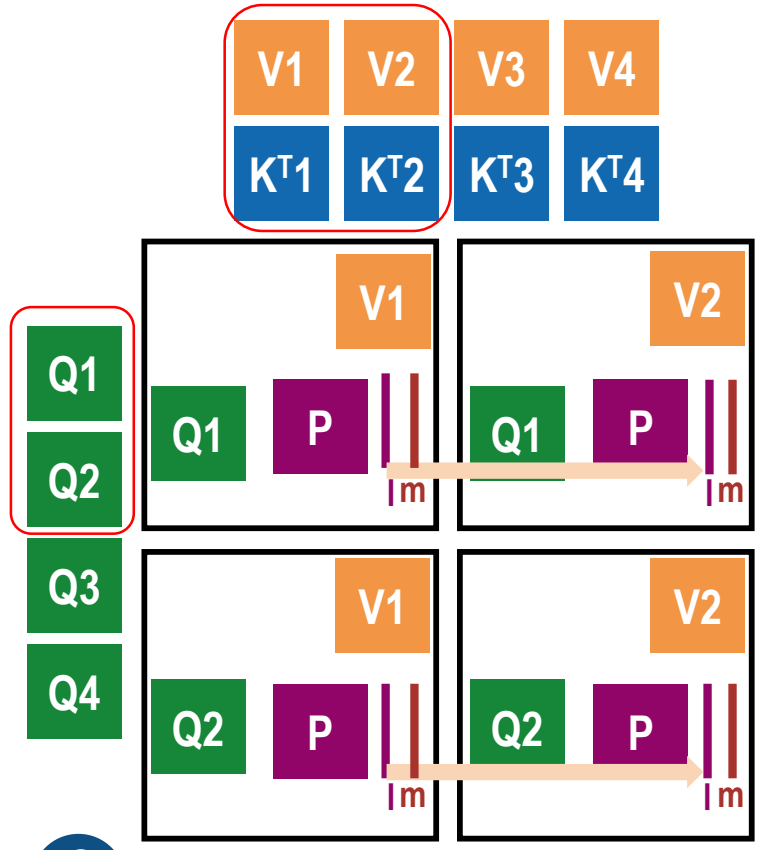




6 Row-wise red.sum l and multicast -> global denominator

# Detailed FlatAttention Dataflow

- Distinct data movement patterns within the tile group

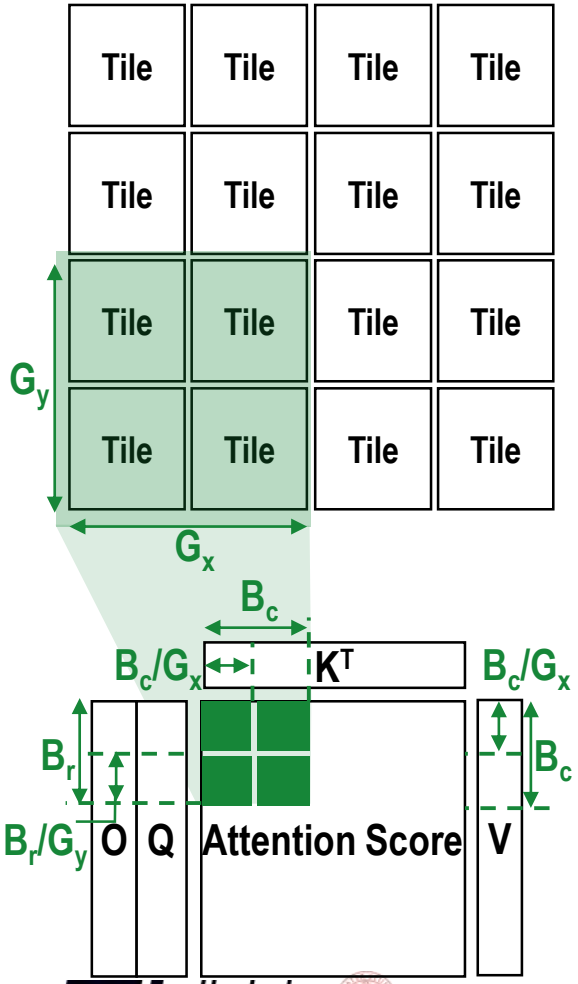


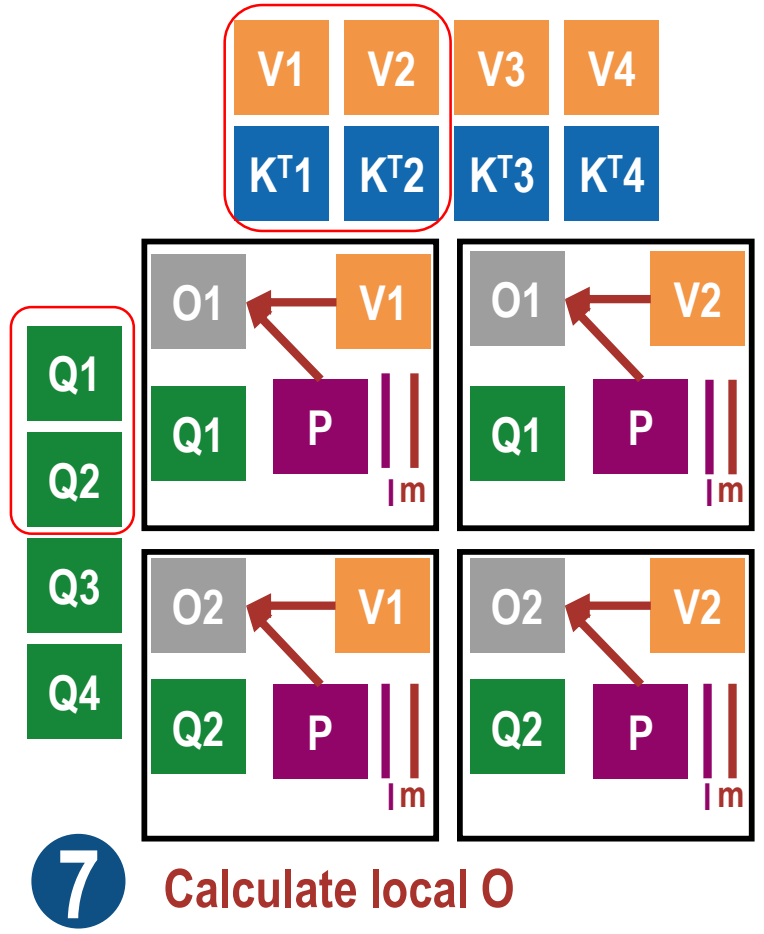




6 Row-wise red.sum l and multicast -> global denominator

# Detailed FlatAttention Dataflow

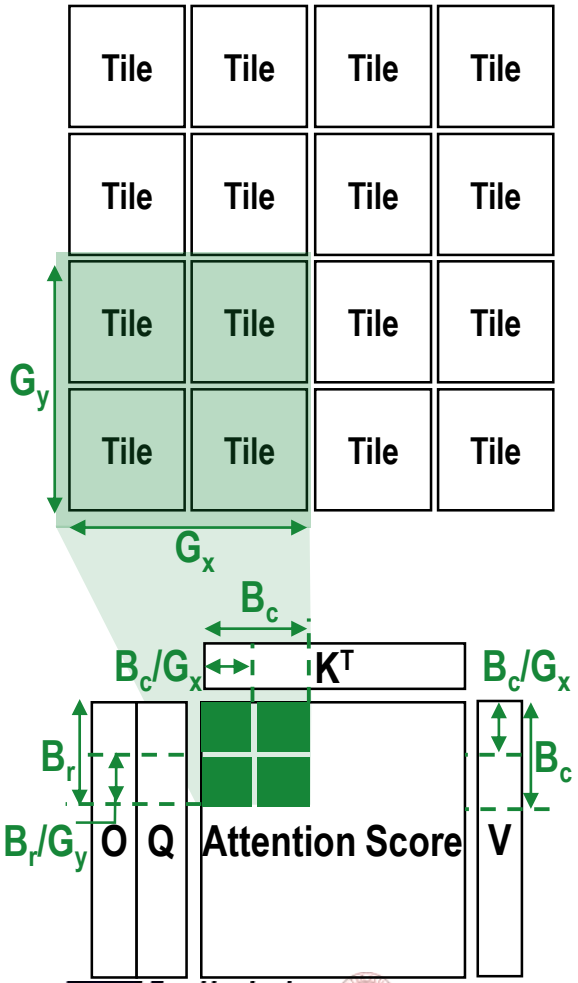
- Distinct data movement patterns within the tile group

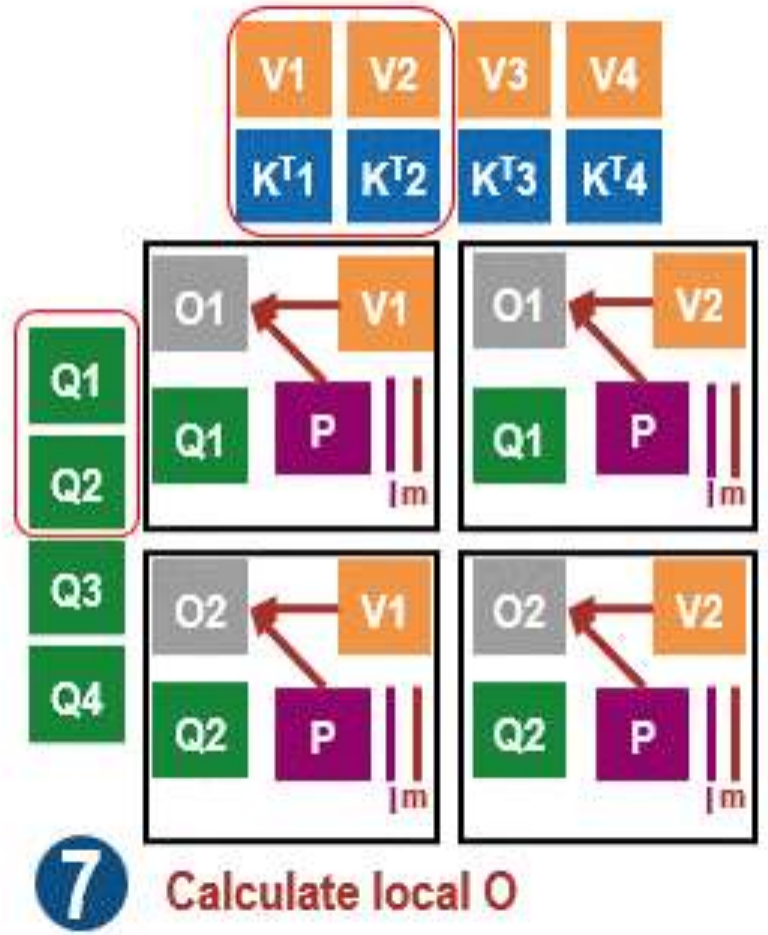


# Detailed FlatAttention Dataflow

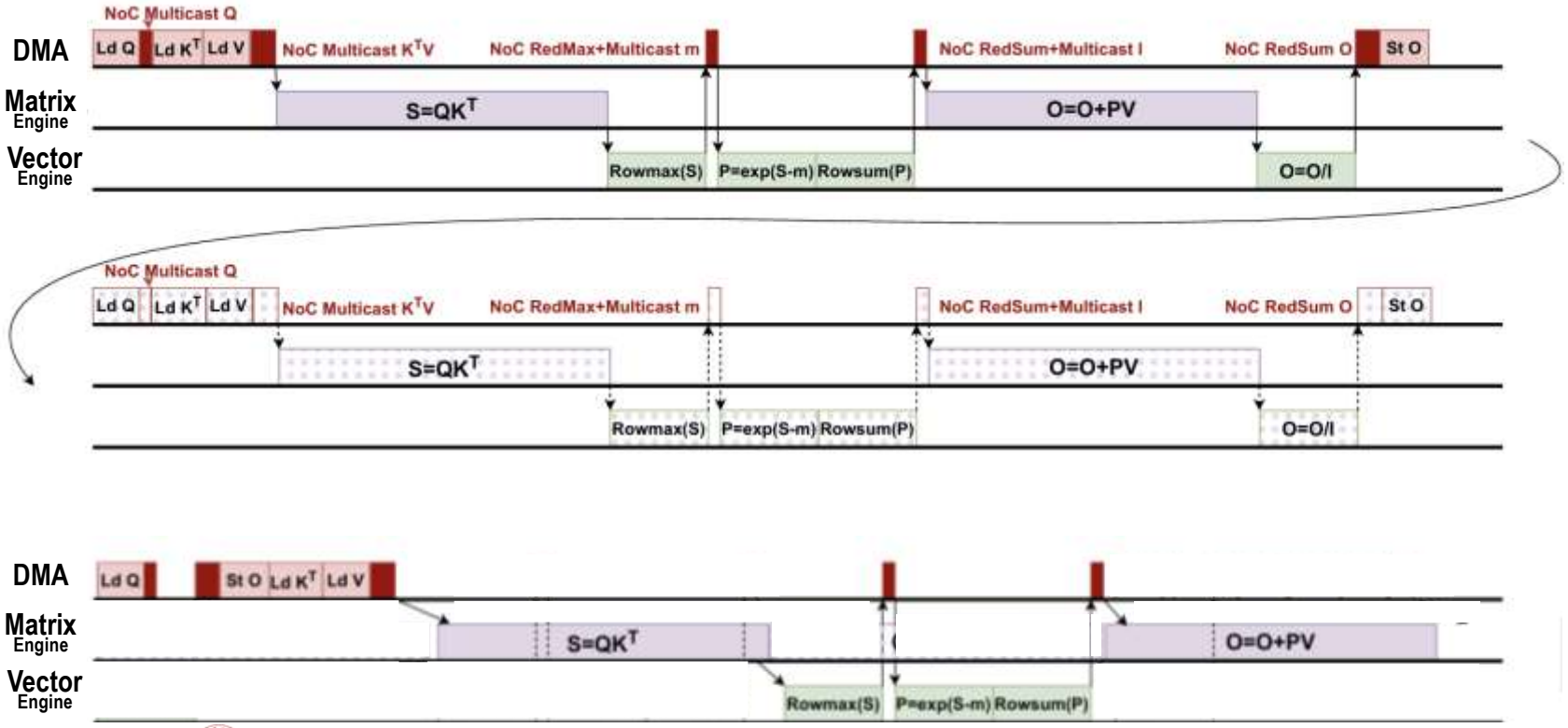
- Distinct data movement patterns within the tile group
  - Edge tiles collect data from HBM once and inter-tile data exchange via collective communications



# Asynchronous FlatAttention

- To further improve utilization
  - Using asynchronization nature of {DMA, Matrix, Vector} engines invoking
  - One tile group dealing with 2 heads concurrently
  - Overlap the runtime of {DMA, Matrix, Vector} engines

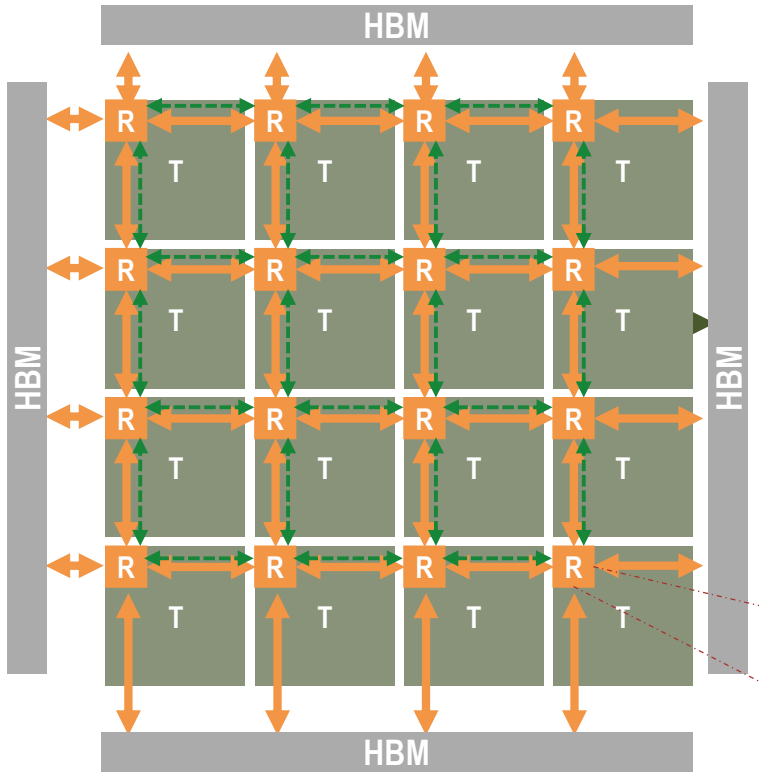


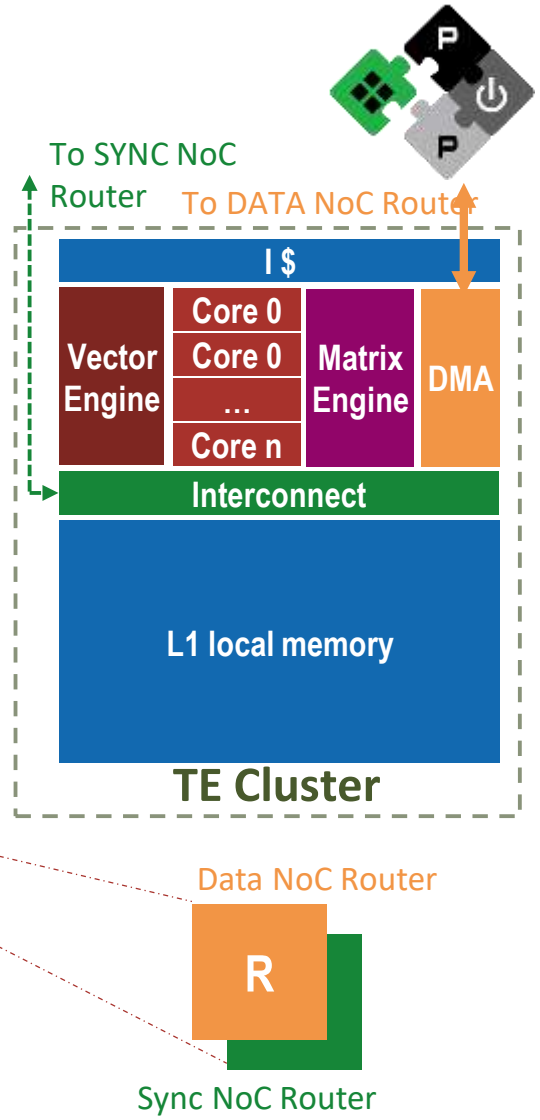

**Naïve FlatAttention**

**Asynchronous FlatAttention**

# High-Level Simulation Model – SoftHier System

- **Fully Parameterizable through Configuration File**
  - System configuration
    - #Tiles (#row, #column)
    - NoC link data width
    - #HBM channels and placement on 4 edges
  - Cluster configuration
    - L1 memory capacity and bandwidth
    - Matrix engine systolic array (shape, CE pipeline stage)
    - DMA burst length, #outstanding burst
- **Based on GVSoC[10] Platform**
  - Event-based simulator, fast simulation speed up to 25 MIPS
- **Calibrated with RTL Models**

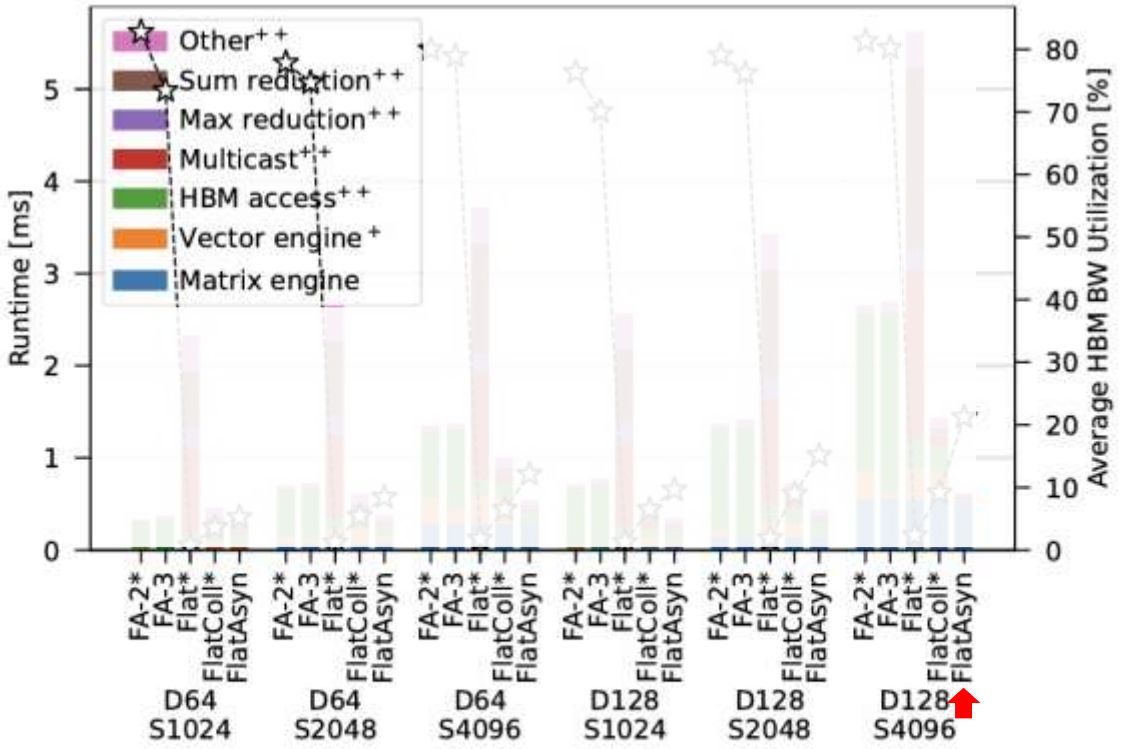



https://github.com/gvsoc/gvsoc/tree/soft_hier_release

# FlashAttention vs. FlatAttention

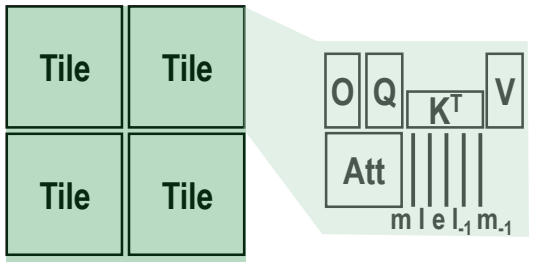
- Compare different MHA implementations
  - **FlashAttention-2**(*FA-2*) and **FlashAttention-3** (*FA-3*)
  - **Naïve FlatAttention** without (*Flat*) and with (*FlatColl*) Collective primitves on NoC fabric.
  - **Asynchronous FlatAttention** (*FlatAsyn*) with NoC Coll
- With FlatAttention **group includes all tiles**
- Results
  - FlashAttention is highly memory-bound
  - FlatAttention significantly reduces HBM access
  - NoC supported collective primitives are essential to accelerate inter-tile data exchange
  - With NoC collective primitives + Asynchronous:
    - Up to 89.3% uti, 4.1× perf speedup, 16× less HBM traffic




Runtime breakdown (bars) and average HBM BW utilization (star markers) for different MHA implementations and layer sizes. +Runtime not overlapped with RedMulE. ++Runtime not overlapped with either Spatz or RedMulE. *Implementations without double buffering.

# Tile Group ScaleTrade-offs for FlatAttention

- The group scale is a sensitive parameter to utilization of different MHA layers
  - MHA with long sequences benefit from large tile group
    - Reduced overall HBM traffic
  - MHA with shorter sequence length would suffer from "over-flatten effect".
    - Reduced slice size per tile
    - Reduced matrix engine utilization
    - Increased synchronization overhead
  - For every sequence length there exists **an optimal group scale balancing the two effects**



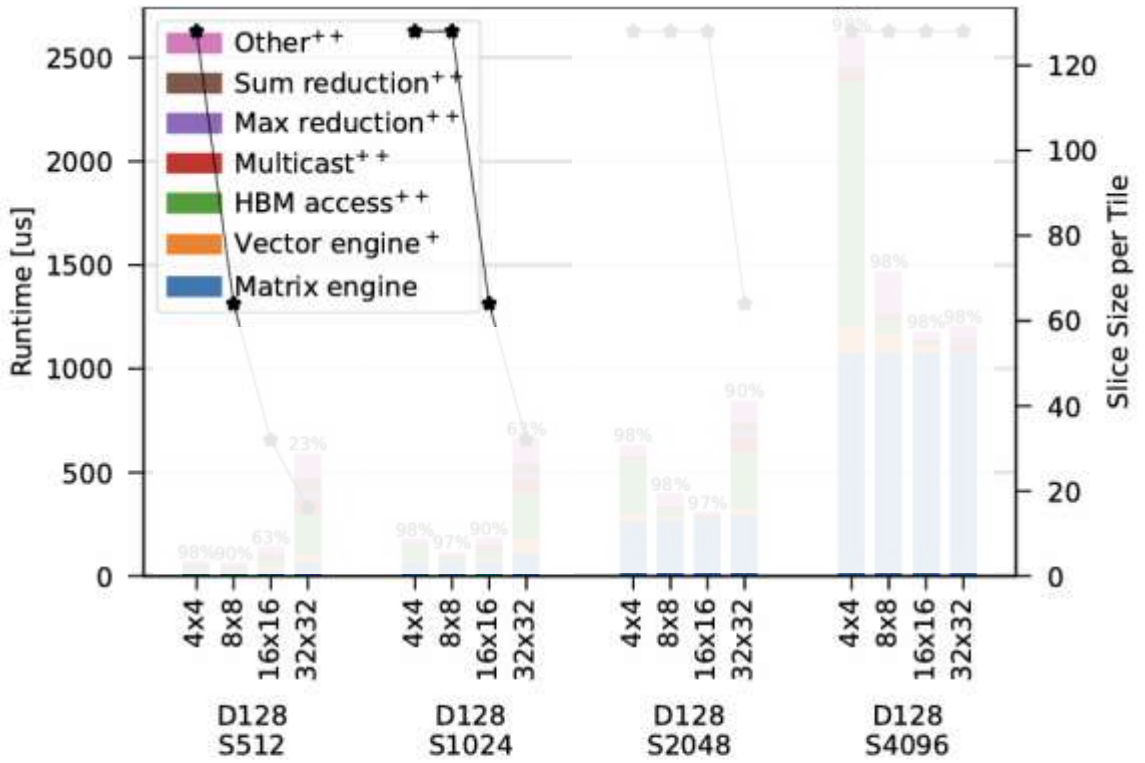
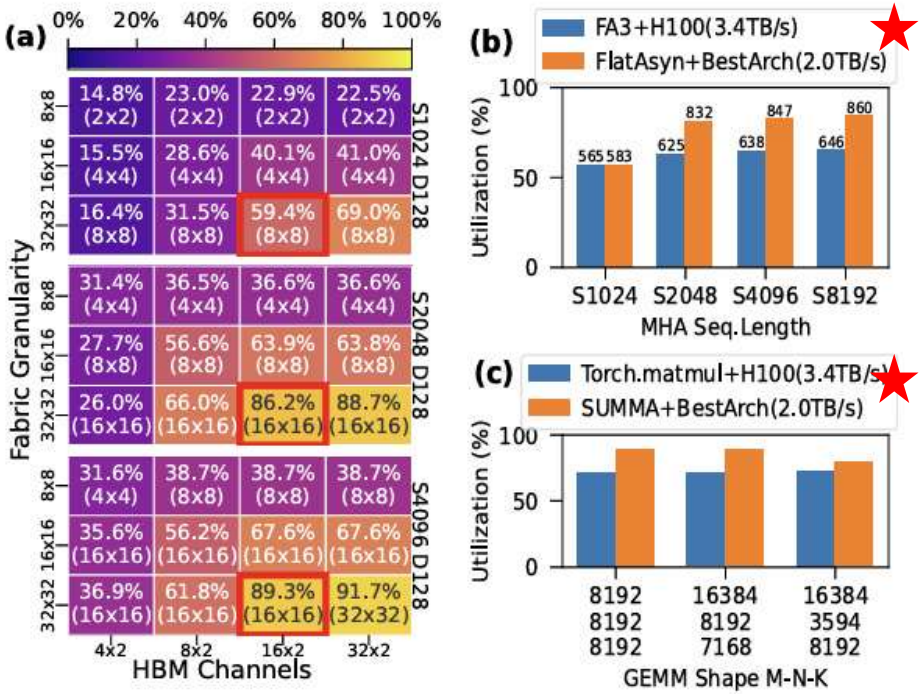




Fig. 4: Runtime breakdown for different (square) flattening scales and layer sizes. Percentage labels above the bars indicate the average utilization of the RedMulE units when active. +Runtime not overlapped with RedMulE. ++Runtime not overlapped with either Spatz or RedMulE.

# Co-exploration of Architecture and Algorithm Parameters

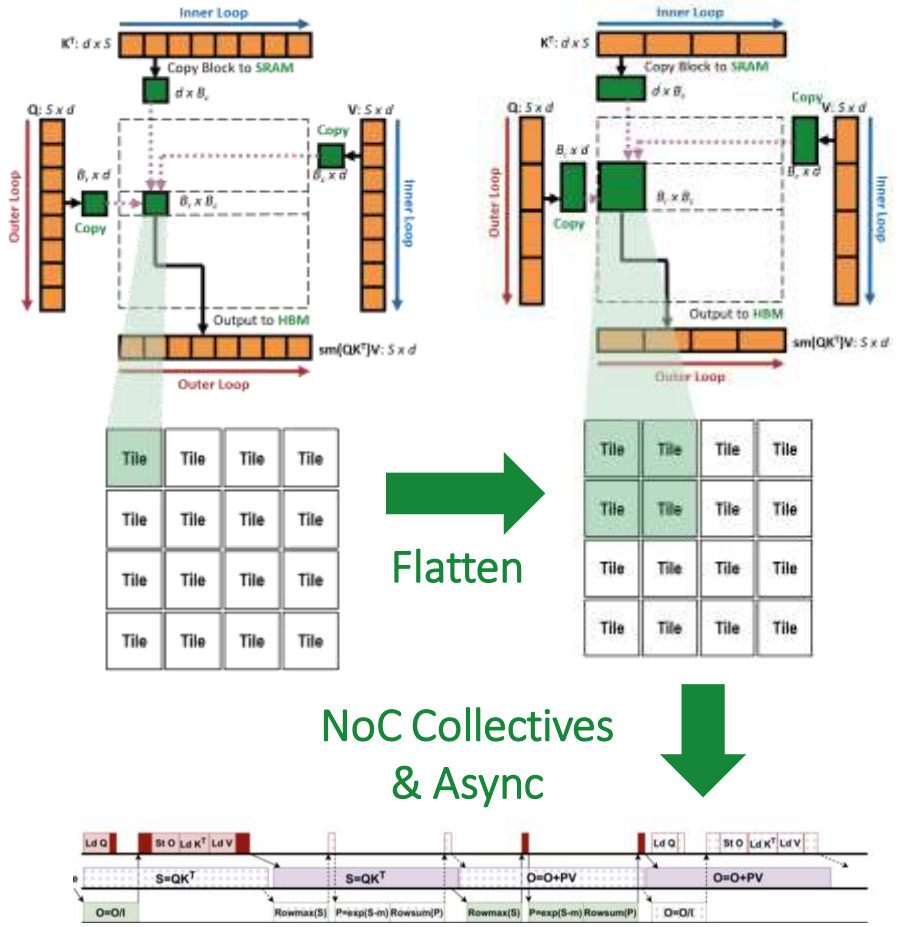
- Our goal
  - Design a tile-based accelerator with comparable peak performance to Nvidia's H100
  - While improve utilization and reducing overall HBM bandwidth requirements on MHA workloads
- Searching for both architecture and MHA dataflow parameter design space
  - Select BestArch for performance over cost
  - Up to 1.3× utilization speedup in MHA
  - Up to 1.2× utilization speedup in GEMM
  - Require 40% less available HBM BW to H100
  - 1.8x die size reduction to H100 (TSMC 5nm estimated)

| Fabric Granularity | 32×32 | 16×16 | 8×8 |
|---|---|---|---|
| RedMulE CE Array | 32×16 | 64×32 | 128×64 |
| Spatz FU Count | 16 | 64 | 256 |
| Local Memory Size (KB) | 386 | 1526 | 6144 |
| Local Memory Bandwidth (GB/s) | 512 | 2048 | 8192 |



# Conclusion For FlatAttention

- We propose **FlatAttention**
  - An optimized dataflow for MHA on tile-based many-PE accelerators
    - Co-designed with NoC collective primitives
    - Generalizable Design: Applicable to MQA, GQA, and MLA variants
  - Outstanding Performance
    - Up to 89.3% utilization, 4.1× performance speedup, 16x HBM traffic reduction
    - Leads to designing scalable AI accelerator with PPA outperforming SoA GPU
- Future Work
  - Co-design Dataflow and NoC-collectives for
    - End-to-End LLM Inferencing
    - Chiplet System and 3D-staked memory







## Related Works

Legend: ✓ = strong / explicitly supported ○ = partially addressed — = not addressed

| Work (venue 'yr) | Peak PE Util [%] | Scalability | NoC-level collectives | Design Space Exploration | Attention-specific optimisation |
|---|---|---|---|---|---|
| This Work | 89.3 | ✓ | ✓ | ✓ | ✓ |
| H²-LLM (ISCA '25) [13] | 80–85 | ✓ | ○ | ✓ | ✓ |
| FuseMax (MICRO '24)[14] | ≈99 | — | — | ○ | ✓ |
| LAD (HPCA '25) [15] | 72 | ○ | ○ | — | ✓ |
| ALISA (ISCA '24) [16] | n/a | — | — | — | ✓ |

Thank you!

# References


[1] A. Ivanov et al., “Data movement is all you need: A case study on optimizing transformers,” in MLSys, 2021.

[2] T. Dao et al., “FlashAttention: Fast and memory-efficient exact attention with IO-awareness,” in NeurIPS, 2022.

[3] F. Zaruba et al., “Snitch: A tiny pseudo dual-issue processor for area and energy efficient execution of floating-point intensive workloads,” IEEE TCOMP, 2020.

[4] T. Benz et al., “A high-performance, energy-efficient modular DMA engine architecture,” IEEE TCOMP, 2023.

[5] T. Fischer et al., “FlooNoC: A 645-Gb/s/link 0.15-pJ/B/hop open-source NoC with wide physical links and end-to-end AXI4 parallel multistream support,” IEEE TVLSI, 2025.

[6] Y. Tortorella et al., “RedMule: A mixed-precision matrix–matrix operation engine for flexible and energy-efficient on-chip linear algebra and TinyML training acceleration," FGCS, 2023.

[7] M. Perotti et al., “Spatz: Clustering compact RISC-V-based vector units to maximize computing efficiency,” IEEE TCAD, 2025.

[8] T. Dao, “FlashAttention-2: Faster attention with better parallelism and work partitioning,” in ICLR, 2024.

[9] J. Shah et al., “FlashAttention-3: Fast and accurate attention with asynchrony and low-precision,” in NeurIPS, 2024.

[10] N. Bruschi et al., “GVSoC: a highly configurable, fast and accurate full-platform simulator for RISC-V based IoT processors,” in ICCD, 2021.

[11] D. Schor, “TSMC N3, and challenges ahead,” 2023. [Online]. Available: https://fuse.wikichip.org/news/7375/tsmc-n3-and-challenges-ahead/

[12] Rausch, Oliver, et al. "A data-centric optimization framework for machine learning." in SC, 2022.

# References

[13] C. Li et al., “H²-LLM: Hardware-Dataflow Co-Exploration for Heterogeneous Hybrid-Bonding-based Low-Batch LLM Inference,” in ISCA, 2025.

[14] N. Nayak et al., “FuseMax: Leveraging Extended Einsums to Optimize Attention Accelerator Design,” in MICRO, 2024.

[15] H. Wang et al., “LAD: Efficient Accelerator for Generative Inference of LLM with Locality Aware Decoding,” in HPCA, 2025.

[16] Y. Zhao *et al.*, “ALISA: Accelerating Large Language Model Inference via Sparsity-Aware KV Caching,” in *ISCA*, 2024.