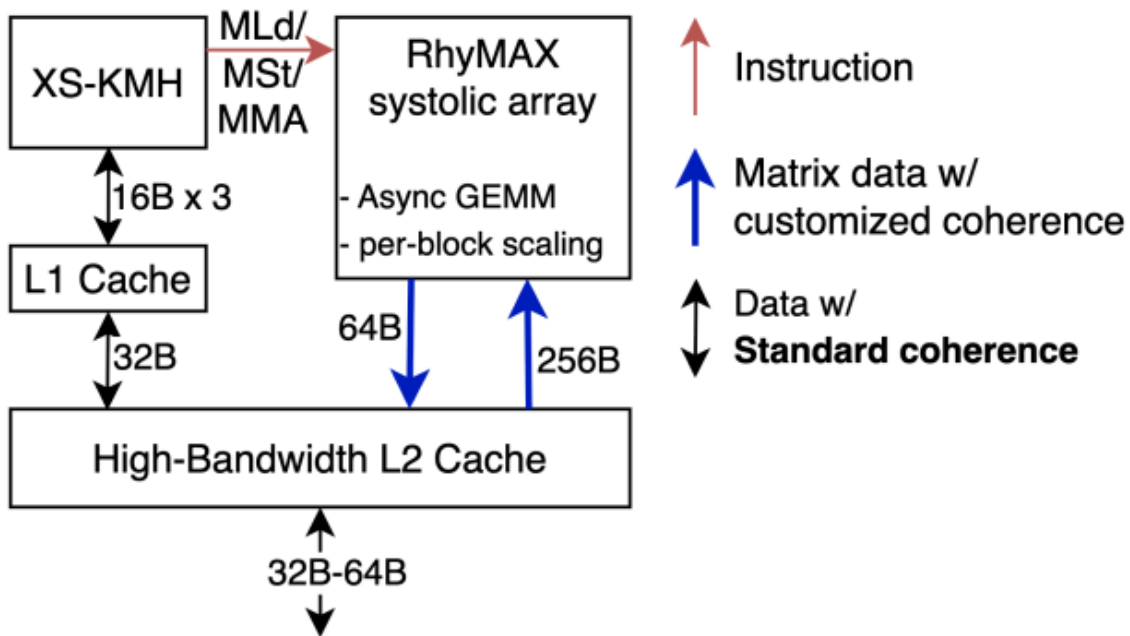
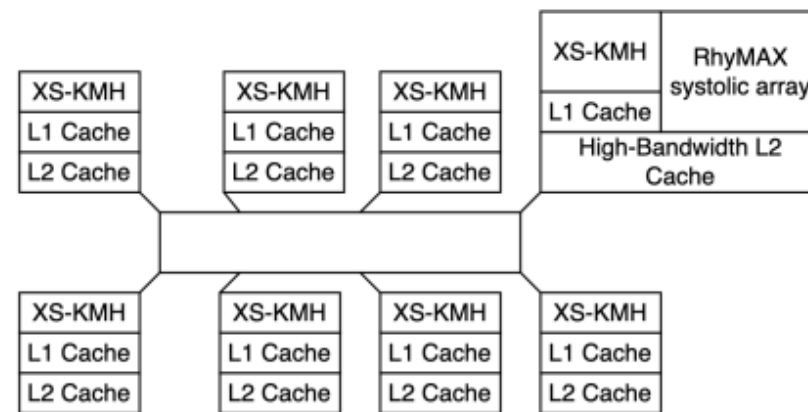


# **XSAI ( $\xi$ ) : Hardware Support for Modern LLM Kernels in a CPU Paradigm**

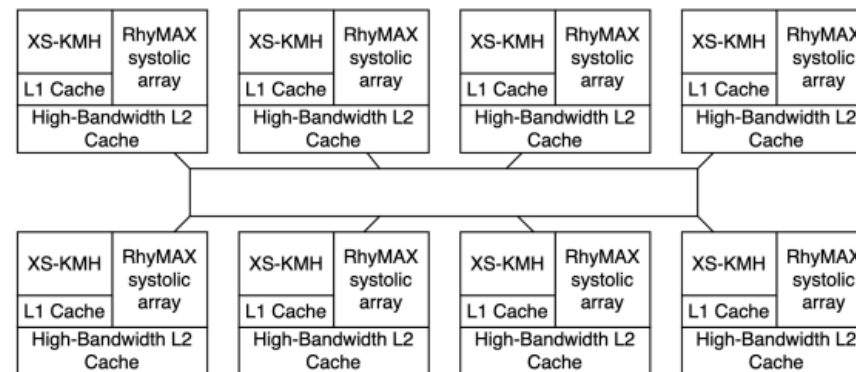
# XSAI ( $\xi$ ) : Xiangshan-KMH + RhyMAX



## • Consumer SoC (AIPC)



## • Cloud SoC



# Agenda

## Motivation of RISC-V + GEMM

- Why GEMM
- Why RISC-V CPU

## Current status

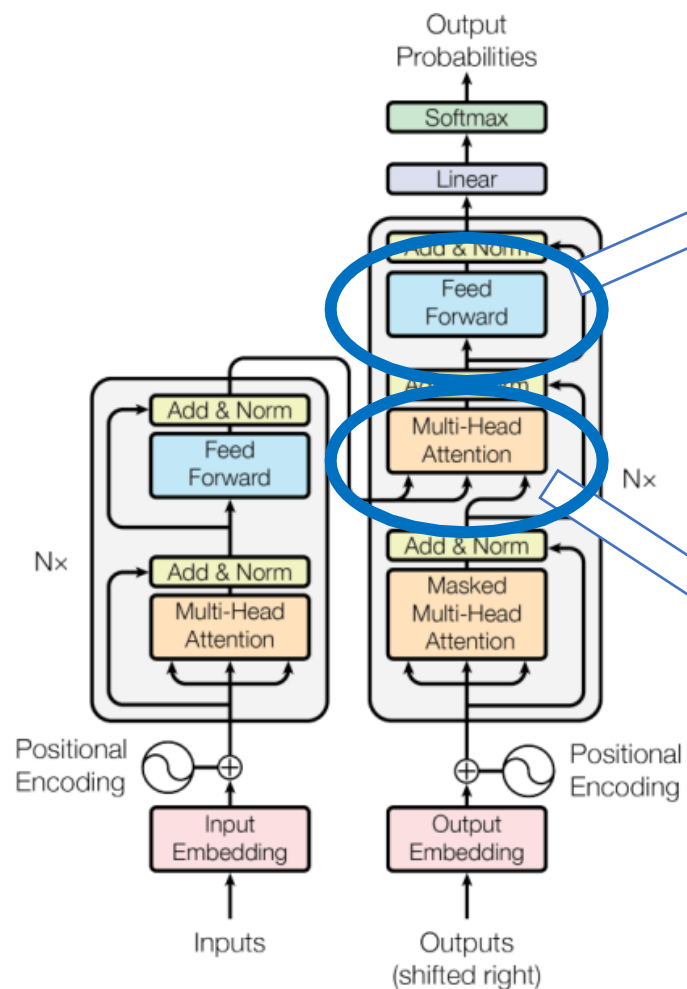
- High-bandwidth L2-cache
- Asynchronous GEMM
- MXFP8

# GEMM is important, even for decoding & consumer-side

FAQ: But decoding is GEMV/mem-bound?

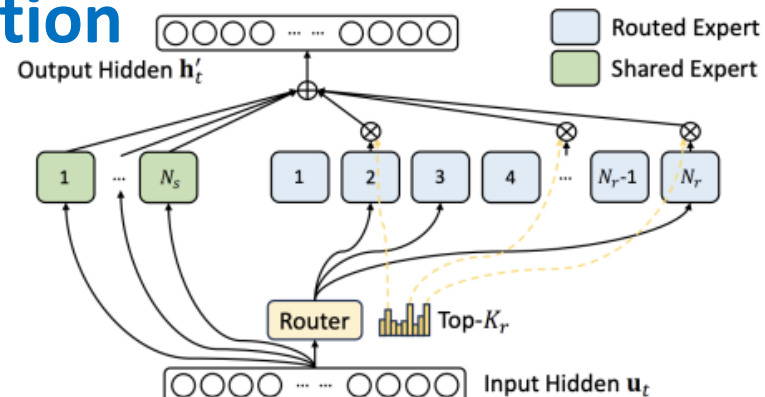
- Takeaway1: Decoding in cloud is not memory-bound
- Takeaway2: Decoding in consumer-side can be GEMM
- Takeaway3: Speculative decoding is cheaper than memory channel

# Evolution of transformer block



Dense FFN -> MoE FFN

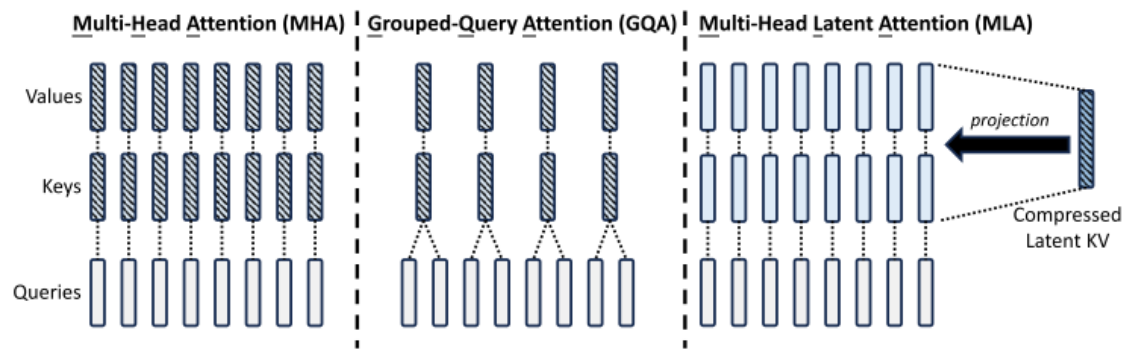
Less Communication



MHA -> GQA -> MLA

Less KV Cache occupation

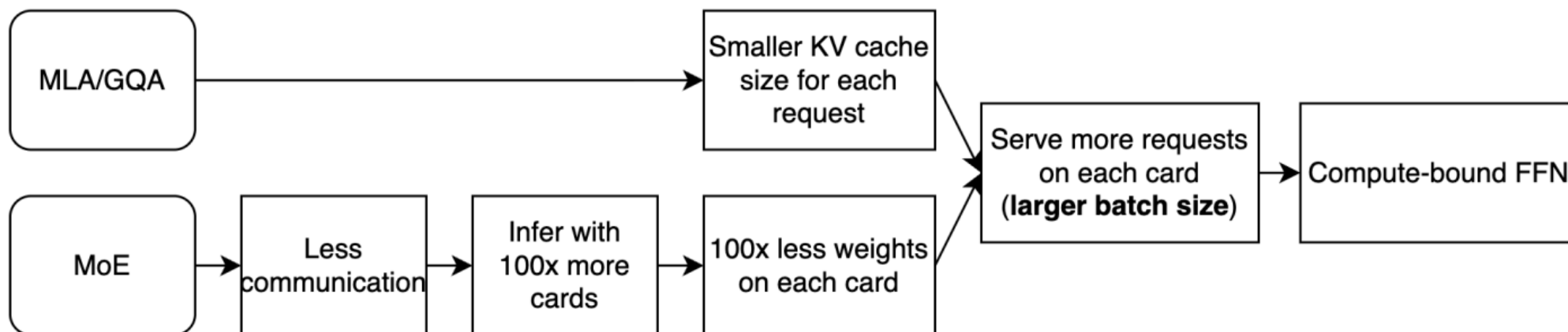
Higher computation/mem ratio



<https://arxiv.org/pdf/1706.03762>  
<https://arxiv.org/abs/2405.04434>

# Takeaway1: Decoding (MoE) in cloud is not mem-bound

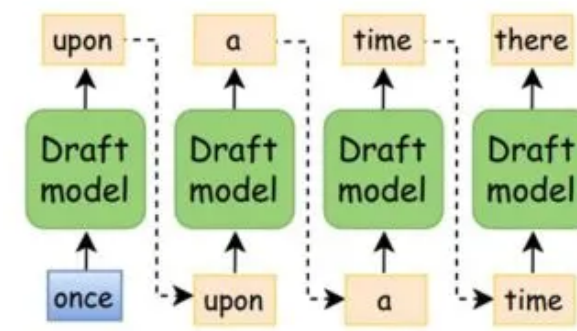
- With EP, DSV3 is either **compute-bound or network-bound**
  - **NOT memory bound**
- Hard to make profit without EP
- Flag-ship open-weight models are becoming MoE
  - Deepseek V3, Qwen3, LLAMA4



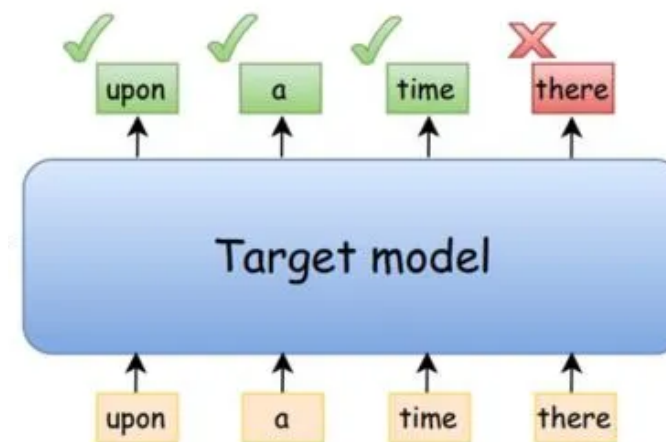
# Takeaway2: Decoding in consumer-side can be GEMM

## Speculative decoding

- Guess multiple tokens with a draft model
- **Verify** them in parallel (load weights once)
- **Similar with prefill (GEMM)**
- For QWEN3 8B + speculation tree size=64
  - GQA:  $QO\_heads = 64 \times 8 = 512$ , head\_dim=128
  - Up proj:  $M=64$ ,  $K=5k$ ,  $N=50k$
  - Down proj:  $M=64$ ,  $K=25k$ ,  $N=5k$
- **Main ops are ~~GEMV~~ GEMM during decoding**



Step1: Autoregressive generation

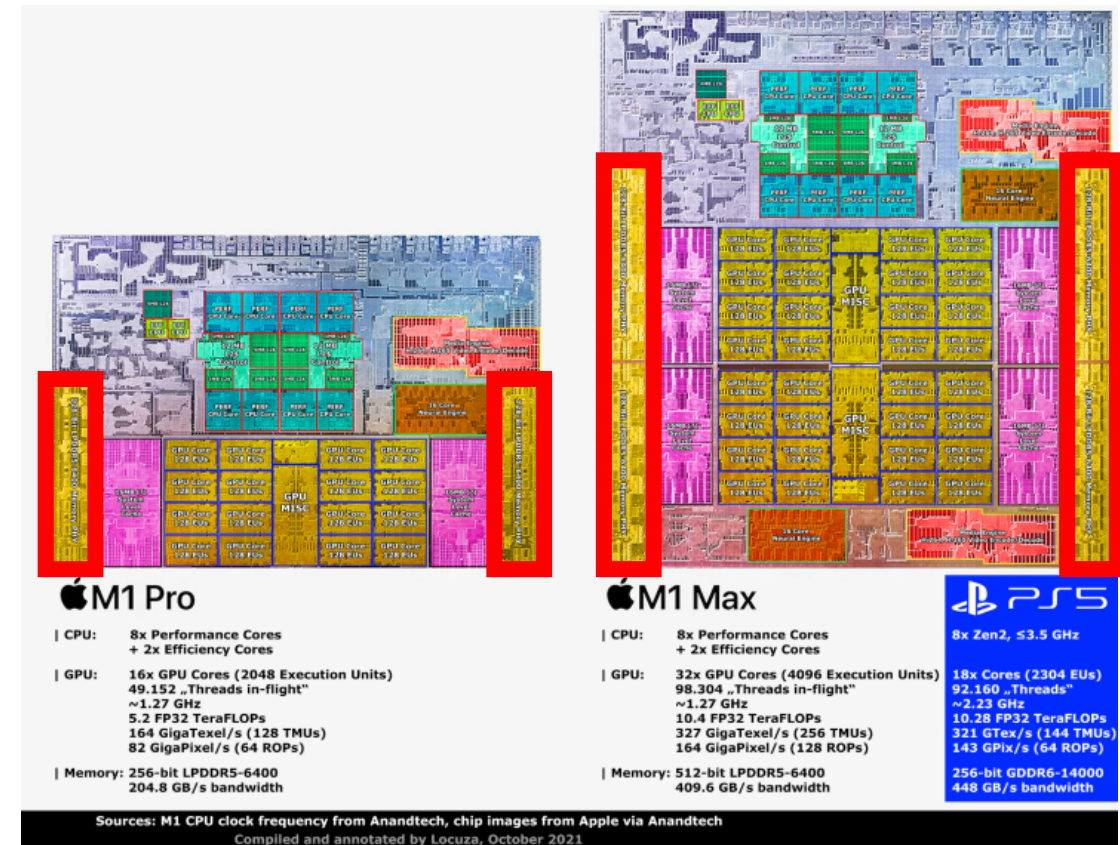


Step2: Parallel verification

<https://arxiv.org/abs/2211.17192>

# Takeaway3: Spec-decoding is cheaper than memory

- **Speculative decoding** VS. larger mem bandwidth
- Increasing memory channels is expensive
  - Double memory channels
    - Impact on total area
  - Double particles
- Increase GEMM FLOPS is cheaper
  - Double height of systolic array
  - Limited impact on SRAM/datapath

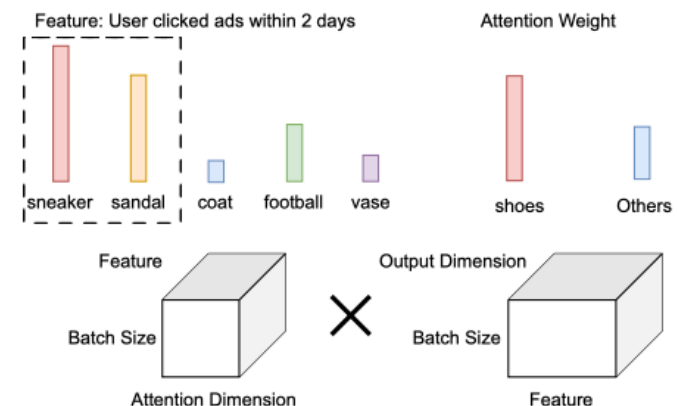
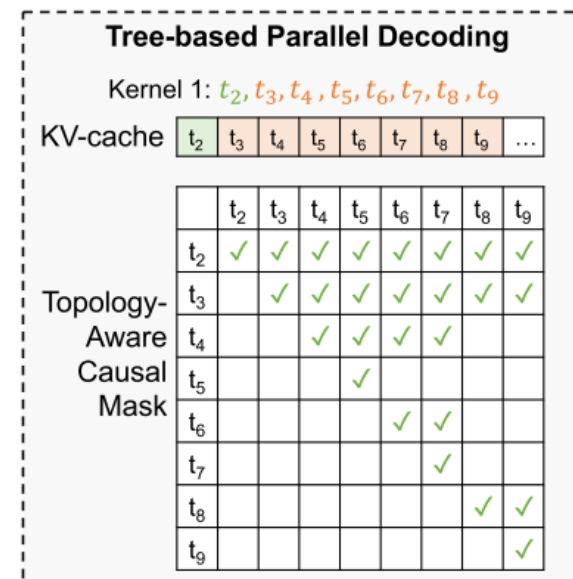


<https://www.anandtech.com/show/17024/apple-m1-max-performance-review>

<https://zhuanlan.zhihu.com/p/12618700803> 北京开源芯片研究院 (BOSC)

# Why RISC-V CPU + GEMM?

- Software eco-system
  - Scalar compilers
  - Auto-vectorization for non-critical operations
  - OpenMP
- General computing
  - variable tree attention mask (SpecInfer and EAGLE-2)
  - Jagged attention



<https://arxiv.org/abs/2409.15373>  
<https://arxiv.org/abs/2305.09781>

# CPU VS. SIMT

- CUTE is trying to express GEMM in SIMT-paradigm (Warp)
  - Obscure (at least for me)
- Mainstream languages are Shifting form Warp-based to tile-based
  - OpenAI Triton
  - Tile-lang
- Support tile-based primitives in CPU is straight-forward
  - Intel AMX
  - RISC-V Attached matrix unit

# How to satisfy computing demand in XSAI

- Better **Model FLOPS utilization (MFU)**
  - High-bandwidth L2 cache
  - Asynchronous GEMM / matrix load / matrix store
- Higher **Tflops**: Hardware support for quantization precision
  - Float: mxfp8
  - Int: Hadamard transform (Future works)

# High-bandwidth L2 cache

- Demands of matrix unit conflicts with high-perf CPU

	Primary demand	what can be traded	Best buffer in existing memory hierarchy
CPU	latency	capacity	L1-cache
Matrix unit	capacity and hit bandwidth	Latency	?

- High-bandwidth L2 cache (HBL2)
  - Capacity: 1-2 MB
  - Bandwidth: 256-512 Bytes per cycle
  - Latency: ~12 cycle hit latency

	Effective hit bandwidth/Bytes/cycle
HBL2 (8-bank)	480
Xiangshan Coupled L2 cache (4-bank)	17
XuanTie C910 L2 cache	5.5

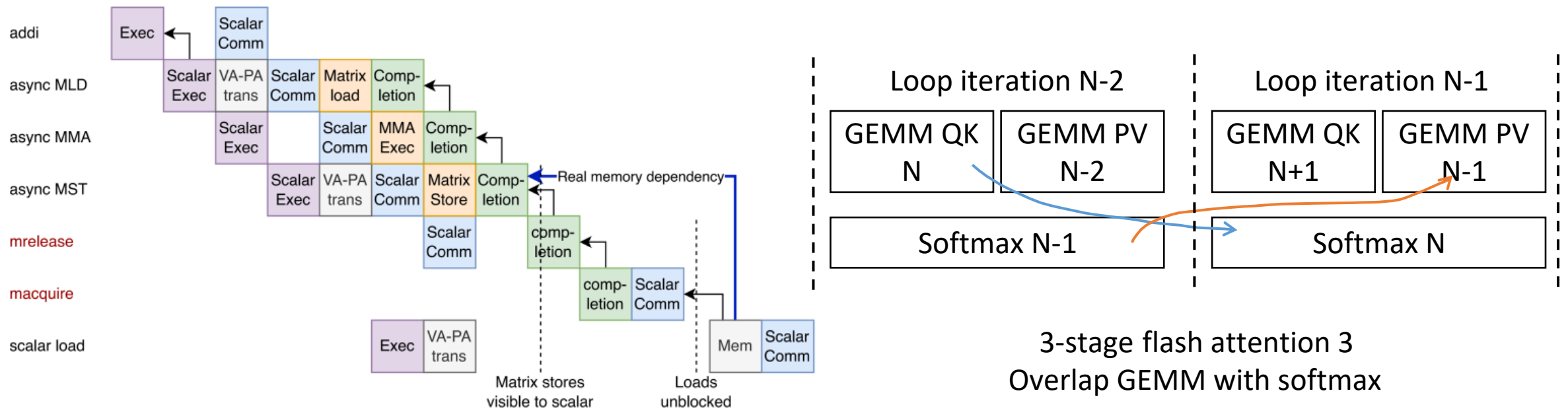
<https://chipsandcheese.com/p/alibabat-heads-xuantie-c910>

# High-bandwidth L2 cache

- When coherent cache meets GEMM
  - Get/Put VS. Acquire/Release
  - Block invalidation from HNF
  - First-fetching permission of weights
  - Block insertion policy
- HBL2
  - Avoid crossbar: Direct datapath to matrix unit
  - Avoid Read/Modify/Write pattern of Acquire by using Get/Put semantic
    - 17 Bytes/cycle → 32 Bytes/cycle
  - Avoid block invalidation from HNF with smart permission speculation

# Support async MMA in XSAI

- Async mma/mld/mst commit in scalar core without execution
  - Precise exception: VA-PA translation & permission check in scalar core
  - Explicit synchronization:
    - Matrix stores are not visible to scalar pipeline/vector pipeline before
    - M.release/M.acquire to enforce dependency explicitly

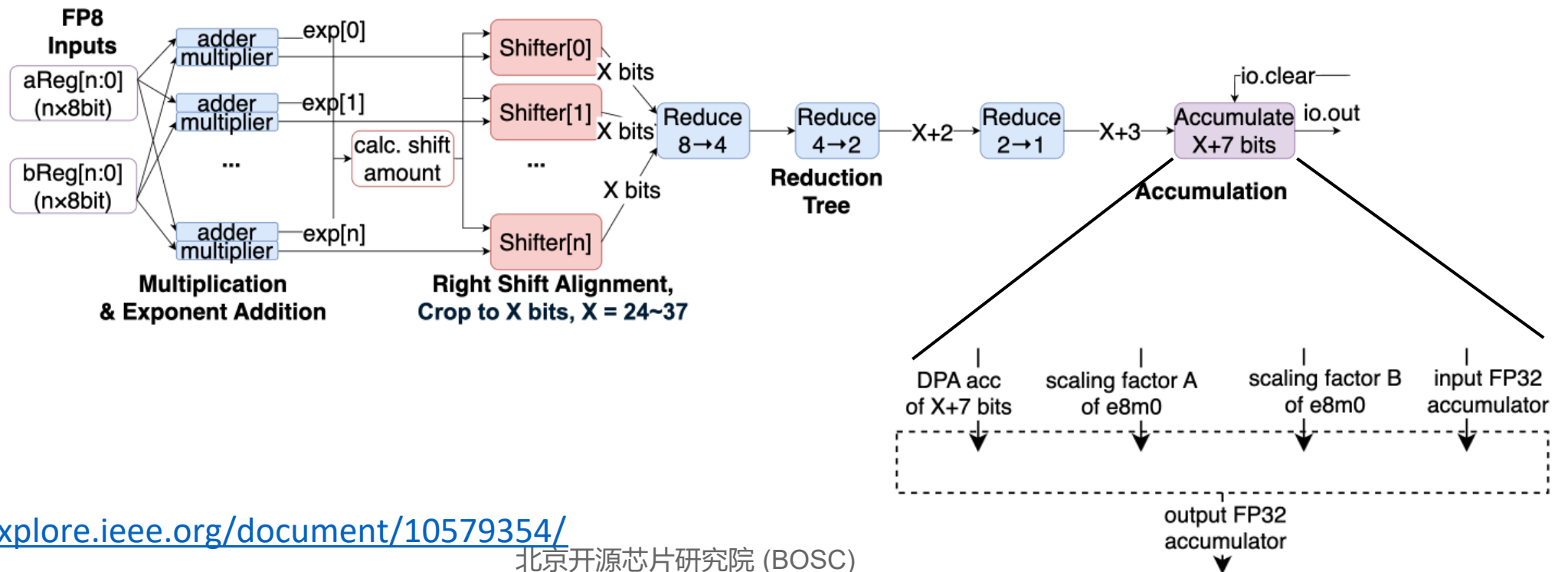


# Achieve higher TFLOPS with quantization

- 8-bit tensor core
  - INT8/INT4
  - MXFP8
- Quantization enhancement
  - High accumulation precision
  - Fine-grained scaling factor

# Mx<sub>fp8</sub> DPA (Dot-Product Accelerator)

- 8-way fp8 DPA
- Every 32 elements share a scaling factor at K dimension
- Optional full-mantissa



# Future works

- Hardware support for Hadamard transform
- Hardware assisted multi-cast
- Pre-silicon end-2-end tuning framework for LLM