Groq

Fast inference on GPU's

Tokens drive compute.

Fastest inference speed for models

TTFB is the most important metric for LLM's. in a real-time setting

GroqChip

→ First built HW

→

Chip & GroqCard ⇒ GroqNode

& GroqNode ⇒ GroqRack

Process anything seq in nature. V FAST

GroqChip:

Start from sand (i.e the chip)

AIM: Hlw thats easy to program

Jonathan - Founder of Groq. Inventor of TPU

TPU hard to program
AIM: Slw should be easy on Hlw

How did we get here?

End of Dennard Law →
Single core ILP hit a wall

Multicore
Shift parallelism responsibility from Hlw to Slw

How to fix it?
Custom Hlw for applications.

AI algos are well behaved (data flow is static, algo, etc).

Problem is memory is not deterministic
This is why AI compilers struggle

Solution? Make 100't deterministic hardware

LPU → Simpler than H100
→ 14 nm
→ 3 tech holder
→ No HBM
→ No silicon imposer

Magic ? Architecture

Lot of waiting for H100. Adds to latency. Using HBM has penalty

LPU
→ Fully deterministic
→ No cache. Direct memory ?
→ How do you load Llama 2 - 70B model into a device that doesn't have HBM ?
→ Entire system is deterministic
→ All chips are synced to act like one device i.e like a megachip.
→ Within 2-3 NS, have access to

TB worth of memory

→ can do the above in a deterministic way

→ soons to access any other device within range

→ Assign work to idle units fast

→ Low latency because everything is pre-scheduled & orchestrated by software

→ Massively scale.

→ Groq has shown 600x improvent in cyber security over GPUs.

→ Work in fin space. Fast processing & of data

→ see papers on website.

→ Also work with sparse models.

→ TSP: Tensor Streaming Processor
But LPU built for LM

Building blocks

SIMD Unit: 320-element vector.

Lightweight instruction dispatch.

Different types of special SIMD units

MXM - Matrix vector
VXM - Vector vector
SXM - Data reshapes
MEM - On chip SRAM

Flat memory
Significant bandwidth to compute! 80 TB/s
⇒ Almost 100x the bandwidth of HBM

Single one dimensional interconnect for inter FU communication.

Instruction Set:

320 element vector ops. (compared to 10k instructions for

explicit resource selection

pytorch)

Cool parts:

→ Activation functions like ReLU, TanH are part of the core instruction set.

MLIR used in compiler.

Groq: ⇒ 35 developers
Nvidia ⇒ 50K developers

Groq only focuses on inference for now

hmm chip next!!

For 70b LLM ⇒ Didn't answer it.

Interconnect:-

Visual 3 layers :- S/W, Network & Compute

CPU S/W + N/W + compute
S/W controlled network

→ Use SW to sync chips to act like one big chip (layer)
→ Use 'DragonFly' mode of connection.

→ One effective global clock
→ No routing
→ 3NS ⇒ Access to 2 TB of SRAM.

Groq can reach max bandwidth in a few ns.
This is why inference is fast.

SW scheduled communication
Efficient use of resources is the key reason.

Scalability

→ Linear increase in perf as number of LPU's increase

GPU vs LPU :-

Both are matrix processing engines.

GPU: Have a 'production' stop.
DGX box is limited of 8 packed chips

LPU: Assembly line for making tokens
No HBM
Don't go through switches to move data between devices
Linear movement.
Low latency
Lower power consumption.

Power metric

Joules per token
LPU is 10x better.

Reasons

→ No storage in HBM (because no HBM)

→ Don't access NIC

→ Because of scaling, don't need to fight back pressure, adaptive routing.