BACK MATTER · APPENDIX A

Hardware Quick Reference & Cheat Sheet

GPU Specifications · HBM Generations · Network Interconnects · Key Formulas · Parallelism Rules · The 30 Numbers to Know Cold · MFU Reference Table

Book: AI/ML Infrastructure from Silicon to Scale — V2 Edition | **Author:** Venkat Vinjam | **Cross-ref:** All chapters · Ch 3B: GPU roadmap detail · Ch 4: Memory hierarchy

> *"Know these numbers cold. They are the vocabulary of every architecture conversation, every system design interview, and every capacity planning discussion. An engineer who hesitates on H100 HBM bandwidth in a design review has given away their preparation level."*
>
> — Principal Performance Architect's Interview Principle

This appendix consolidates the hardware specifications, formulas, parallelism rules, and reference numbers used throughout the book. Keep it open alongside every chapter. When a formula appears in the text, verify it here. When a design question arises, start here.

A.1 NVIDIA GPU Generation Comparison [SHIPPED] [ROADMAP for GB300]

GPU	Arch	Process	BF16 TFLOPS	FP8 TFLOPS	HBM Type	HBM Cap	HBM BW	NVLink BW	TDP
V100 SXM2	Volta	12nm	125	N/A	HBM2	32 GB	900 GB/s	300 GB/s	300W
A100 SXM4	Ampere	7nm	312	N/A	HBM2e	80 GB	2.0 TB/s	600 GB/s	400W
H100 SXM5	Hopper	4N TSMC	**989**	1,979	HBM3	80 GB	**3.35 TB/s**	900 GB/s	700W
H200 SXM5	Hopper	4N TSMC	989	1,979	HBM3e	**141 GB**	**4.8 TB/s**	900 GB/s	700W
B200 SXM6	Blackwell	4NP TSMC	**~2,250**	~4,500	HBM3e	192 GB	**8.0 TB/s**	1,800 GB/s	1,000W
GB200 NVL72	Blackwell	4NP TSMC	**~2,250/ GPU**	~4,500/ GPU	HBM3e	192 GB/ GPU	8.0 TB/s	**130 TB/s rack**	1,000W/ GPU
GB300 SXM [ROADMAP]	Blk Ultra	3nm est.	~2,800	~6,000	HBM4	288 GB	**12.0 TB/s**	1,800 GB/s	~1,400W

> **The H100 is the 2025 baseline.**
>
> Design every new architecture to run efficiently on H100 SXM5. If it runs efficiently on H100, it will run on H200 with better memory economics, on B200 with better compute and bandwidth, and on MI300X with more memory. The H100 is the common denominator of the current fleet.

A.2 AMD GPU Reference [SHIPPED] [ANNOUNCED for MI355X+]

GPU	Arch	BF16 TFLOPS	FP8 TFLOPS	HBM Type	HBM Cap	HBM BW	Infinity Fabric BW	TDP
MI250X	CDNA 2	383	N/A	HBM2e	128 GB	3.28 TB/s	800 GB/s	500W
MI300X	CDNA 3	**1,307**	~2,614	HBM3	**192 GB**	**5.3 TB/s**	896 GB/s	750W
MI325X [ANNOUNCED]	CDNA 3+	1,307	~2,614	HBM3e	**256 GB**	**6.0 TB/s**	896 GB/s	~1,000W
MI355X [ANNOUNCED]	CDNA 4	~2,600	~5,200	HBM3e	288 GB	~8.0 TB/s	~1,500 GB/s	~1,400W
MI400 [ROADMAP]	CDNA 4+	TBA	TBA	HBM4	400+ GB	~12 TB/s	TBA	TBA

The MI300X memory moat: 192 GB HBM3 allows LLaMA-3 70B (140 GB BF16) to fit on a single GPU — no tensor parallelism required. Every other current GPU requires TP=2 (two GPUs). At 100 serving instances, that's 100 GPUs saved: $2.5M capex and $1.3M/year OpEx at standard cloud rates.

A.3 HBM Generation Reference [SHIPPED] [ROADMAP for HBM4]

Standard	BW / Stack	Cap / Stack	Data Rate / Pin	Interface Width	Notable GPUs	Status
HBM2	256 GB/s	8–16 GB	2 Gbps	1024-bit	V100, MI100	[SHIPPED]
HBM2e	307 GB/s	16 GB	3.2 Gbps	1024-bit	A100, MI250X	[SHIPPED]
HBM3	819 GB/s	16 GB	6.4 Gbps	1024-bit	H100, MI300X	[SHIPPED]
HBM3e	**~1.2 TB/s**	24 GB	9.6 Gbps	1024-bit	H200, B200, MI325X	[SHIPPED]
HBM4 [ROADMAP]	**~2 TB/s**	32+ GB	~16 Gbps	1024-bit+	GB300, MI400	[ROADMAP]

Why HBM capacity scales slower than compute: Each HBM stack is area-limited by the silicon interposer (reticle ~858 mm²), yield-limited by TSV defect rates, and thermally limited (5–10W per stack). These constraints produce ~2× capacity growth per generation vs ~3× compute — the root cause of the widening ridge point.

A.4 Network Interconnect Reference [SHIPPED]

Interconnect	BW (bidirectional)	Latency	Topology Scope	Primary AI Use	Constraint
NVLink 3.0 (A100)	600 GB/s	<1 µs	8-GPU intra-node	Tensor Parallelism	NVSwitch domain only
NVLink 4.0 (H100)	**900 GB/s**	<1 µs	8-GPU intra-node	Tensor Parallelism	NVSwitch domain only
NVLink 5.0 (B200)	**1,800 GB/s**	<1 µs	8-GPU intra-node	Tensor Parallelism	NVSwitch domain only
NVLink-C2C (GB200)	**10,000 GB/s**	<1 µs	CPU-GPU same chip	KV transfer, unified mem	Same GB200 superchip only
NVLink Switch (NVL72)	130 TB/s rack total	<1 µs	72-GPU rack	Full-mesh TP/EP	One rack "computer"
InfiniBand HDR	50 GB/s/port	~1 µs	Inter-node cluster	Pipeline + Data Parallelism	Not suitable for TP
InfiniBand NDR	**100 GB/s/port**	~1 µs	Inter-node cluster	AllReduce, KV transfer	Not suitable for TP
InfiniBand XDR	200 GB/s/port	<1 µs	Inter-node cluster	AllReduce at scale	Emerging, 2025+
PCIe Gen5 x16	128 GB/s	~5 µs	CPU-GPU same server	Model loading, checkpoints	26× slower than HBM
400GbE (RoCEv2)	50 GB/s/port	~5–20 µs	Inter-node cluster	Scale-out alternative to IB	Higher latency, less predictable

TP over InfiniBand is unusable.

Tensor Parallelism at TP=8 on 70B requires one AllReduce per layer (80 layers) each ~0.5 GB. Over IB NDR (50 GB/s): 0.5/50=10ms/layer × 80 = 800ms overhead per inference step. Over NVLink (900 GB/s): 0.5/900=0.56ms × 80 = 44ms. NVLink is 18× better — TP must stay within the NVSwitch domain.

A.5 Key Capacity Planning Formulas

```
MODEL PARAMETER MEMORY:
  model_bytes = N_params × bytes_per_param
  70B BF16:  70 × 10⁹ × 2 bytes = 140 GB
  70B FP8:   70 × 10⁹ × 1 byte  =  70 GB
  70B INT4:  70 × 10⁹ × 0.5 bytes = 35 GB

KV CACHE MEMORY (per request):
  KV_bytes = 2 × L × H_kv × d_head × S × bytes_per_elem
  LLaMA-3 70B (L=80, H_kv=8, d_head=128, S=4096, BF16):
    = 2 × 80 × 8 × 128 × 4096 × 2 = 5.37 GB per request
  Per token per layer: 2 × 8 × 128 × 2 = 4,096 bytes = 4 KB

TPOT MINIMUM (batch=1, bandwidth-bound):
  TPOT_min = model_bytes / HBM_bandwidth
  H100 BF16:  140 GB / 3.35 TB/s = 41.8 ms/token
  H200 BF16:  140 GB / 4.8 TB/s  = 29.2 ms/token
  MI300X BF16: 140 GB / 5.3 TB/s = 26.4 ms/token
  H100 INT4:   35 GB / 3.35 TB/s = 10.4 ms/token

PREFILL FLOPS (approximate):
  FLOPs_prefill ≈ 2 × N_params × S  (forward pass rule of thumb)
  LLaMA-3 70B, S=2048: 2 × 70e9 × 2048 = 286 TFLOP
  On H100 (989 TFLOPS, MFU=50%): 286 / (989 × 0.50) = 579 ms

MFU:
  MFU = (FLOPs_per_step × steps_per_second) / Peak_TFLOPS

ALLREDUCE TIME (ring algorithm):
  t = 2 × (N-1)/N × M / BW     [M = message bytes, N = GPUs]
  Example: N=8, M=256 MB, BW=900 GB/s:
    t = 2 × (7/8) × 256e6 / 900e9 = 0.497 ms

FLEET SIZING:
  raw_GPUs = peak_tok/s / tokens_per_GPU_per_s
  total_GPUs = raw_GPUs / target_util × burst_headroom
              (target_util=0.75, burst_headroom=1.20)
```

A.6 Parallelism Quick Reference

Strategy	What is Sharded	Comm Pattern	BW Needed	Scale Constraint	Best For
Data Parallelism (DP)	Gradient only (model replicated)	AllReduce gradients	IB acceptable	Model must fit on 1 GPU	Small-to-mid models, high throughput
ZeRO-1	Optimizer states	AllGather params before fwd	IB acceptable	Lower comm vs ZeRO-3	Opt state memory relief
ZeRO-3 / FSDP	Weights + optimizer + grads	AllGather + ReduceScatter	IB acceptable (3× comm)	Highest comm overhead	Very large models, memory-constrained
Tensor Parallelism (TP)	Weight matrices	AllReduce per layer	**NVLink MANDATORY**	NVSwitch domain (≤8 GPU)	Inference latency, very large models
Pipeline Parallelism (PP)	Layers across stages	P2P activations	IB acceptable	Bubble = 1/ PP_degree	Cluster-scale training
Sequence Parallelism (SP)	Sequence dim in activations	AllGather + ReduceScatter	NVLink preferred	Requires TP	Very long context training
Expert Parallelism (EP)	MoE expert weights	All-to-All tokens	IB+ acceptable	Load balance critical	MoE models at scale

A.7 Numerical Precision Reference

Format	Bits	Exp Bits	Mantissa	Range	GPU Support	Typical Use
FP32	32	8	23	$\pm 3.4 \times 10^{38}$	All	Optimizer states, loss scaling accumulator
TF32	32 (19 effective)	8	10	$\pm 3.4 \times 10^{38}$	A100+	Tensor Core internal accumulation
BF16	16	8	7	$\pm 3.4 \times 10^{38}$	A100+	Training activations & weights (default)
FP16	16	5	10	±65,504	V100+	Legacy training; loss scaling required
FP8 E4M3	8	4	3	±448	H100+	Forward pass weights + activations
FP8 E5M2	8	5	2	±57,344	H100+	Gradient backward pass
INT8	8	—	—	−128 to 127	All	LLM.int8(), SmoothQuant activation quant
INT4 / GPTQ	4	—	—	−8 to 7	All (CUDA)	Memory-constrained inference, AWQ
NVFP4 [SHIPPED]	4	2	1	block-scaled	B200+	B200 native training + inference

A.8 The 30 Numbers Every Principal Engineer Knows Cold

```
COMPUTE (BF16 Tensor Core):
  H100 BF16:  989 TFLOPS      H100 FP8:   1,979 TFLOPS
  H200 BF16:  989 TFLOPS      B200 FP8:   4,500 TFLOPS
  MI300X BF16: 1,307 TFLOPS   GB300 FP4:  ~9,000 TFLOPS [ROADMAP]

HBM CAPACITY AND BANDWIDTH:
  H100:   80 GB  | 3.35 TB/s     H200:  141 GB | 4.8 TB/s
  B200:  192 GB  | 8.0 TB/s      MI300X: 192 GB | 5.3 TB/s
  GB300: 288 GB  | 12.0 TB/s [ROADMAP]

INTERCONNECT:
  NVLink 4.0 (H100):    900 GB/s      NVLink 5.0 (B200):  1,800 GB/s
  NVLink-C2C (GB200):  10,000 GB/s   IB NDR (per port):     50 GB/s
  PCIe Gen5 x16:          64 GB/s (one-way)

RIDGE POINTS (BF16 dense, FLOP/byte):
  A100: 156    H100: 295    H200: 206    B200: ~281    MI300X: 247

MODEL MEMORY (LLaMA-3 70B):
  BF16: 140 GB    FP8: 70 GB    INT4: 35 GB    Fits single GPU: MI300X only (BF16)

TPOT MINIMUMS (70B BF16, batch=1):
  H100: 41.8 ms    H200: 29.2 ms    MI300X: 26.4 ms    B200: 17.5 ms

KV CACHE (LLaMA-3 70B, GQA, 4K context, BF16):
  Per request: 5.37 GB    Per token per layer: 4 KB

TRAINING REFERENCE (LLaMA-3 70B, H100, MFU=45%):
  Tokens/GPU/day: ~13 million    Tokens/GPU-hour: ~541K
  Cost @ $3/GPU-hr, 1T tokens: ~$5.5M

PRODUCTION THRESHOLDS:
  Max GPU util target:   80%    Max KV cache util:   80%    MFU world-class: >55%
  TP AllReduce max IB:   NEVER  (use NVLink always)  PP bubble: 1/PP_degree
  SDC rate (H100):       ~1 event / 2 weeks on 10K-GPU cluster
  Checkpoint overhead:   <2% of training time at well-tuned frequency
```

A.9 Roofline Ridge Points by GPU and Precision

GPU	BF16 Ridge (FLOP/byte)	FP8 Ridge (FLOP/byte)	HBM BW	Implication
V100	124	N/A	900 GB/s	Most transformers compute-bound
A100	156	N/A	2.0 TB/s	Large GEMM still compute-bound
H100	**295**	590	3.35 TB/s	Many ops now memory-bound vs V100
H200	**206**	412	4.8 TB/s	Lower ridge → easier to be compute-bound
B200	~281	~562	8.0 TB/s	Higher compute with proportional BW
MI300X	247	~493	5.3 TB/s	Lower ridge than H100 — BW-rich design

> **The H200 insight:**
>
> H200 has a lower ridge point (206) than H100 (295) because NVIDIA added more HBM bandwidth without increasing compute. This means operations that are memory-bound on H100 are more efficiently served by H200, and operations that are just barely compute-bound on H100 may become memory-bound on H200. Re-evaluate regime classification when migrating from H100 → H200.

A.10 MFU Reference Table — What's Normal in Production

Workload	Model	GPU	MFU Range	Source
LLM Pretraining (well-tuned)	LLaMA-3 70B, 4D parallel	H100	**45–52%**	Meta (representative)
LLM Pretraining (Megatron-LM)	GPT-4 class, TP=8 PP=16	H100	**40–48%**	NVIDIA MLPerf Training
Inference Prefill (TRT-LLM)	LLaMA-3 70B, FP8, batch=32	H100	**50–65%**	NVIDIA TRT-LLM benchmarks
Inference Decode (batch=1)	LLaMA-3 70B, BF16	H100	**3–8%**	Bandwidth floor → expected
Inference Decode (batch=64)	LLaMA-3 70B, BF16	H100	**30–45%**	Improving toward ridge
RLHF / PPO	70B policy + 70B reward	H100	**15–30%**	Memory constraints limit batch
Fine-tuning (LoRA)	7B, A/r=64	A100	**25–40%**	Small batch, frequent I/O

MFU Diagnostic Rule:

If MFU <25% during training, investigate in this order: (1) communication overlap failure (profile with nsys), (2) pipeline bubble inflation (check PP schedule), (3) data pipeline stall (check DataLoader CPU util), (4) kernel-level issues (check NCU SM utilization). Fix the highest layer first — kernel tuning while a communication bottleneck exists is wasted effort.

Key Takeaways

CORE CONCEPTS

- H100 is the 2025 baseline: 989 TFLOPS BF16, 3.35 TB/s HBM3, 80 GB, ridge point 295 FLOP/byte. Every new architecture should be designed to run efficiently on H100 first.
- TPOT_min = model_bytes / HBM_bandwidth — a physics floor. For LLaMA-3 70B BF16 on H100: 41.8ms. No software can beat this; only quantization or batching can change it.
- TP must use NVLink — IB-based TP is unusable (800ms overhead per inference step vs 44ms on NVLink). PP can cross IB because it communicates activations (P2P, infrequent), not weights per layer.

COST & PRODUCTIVITY IMPACT

- MI300X's 192 GB HBM eliminates TP=2 for 70B serving. At 100 serving instances: saves $2.5M capex and $1.3M/year OpEx. Memory capacity is a direct cost lever, not a convenience.
- FP8 cuts memory by 2× and GEMM cost by 2×. INT4 cuts memory by 4×. At batch=1 decode, TPOT scales linearly with bytes: H100 70B INT4 = 10.4ms vs 41.8ms BF16. Quantization is the single most impactful decode optimization.
- ZeRO-3 triples communication volume vs ZeRO-1 (AllGather + ReduceScatter for every parameter every step). At 256 GPUs with 70B parameters, this adds 2–4 minutes of AllReduce per hour of training. Know the cost before choosing ZeRO-3.

PRINCIPAL-LEVEL INSIGHT

- The ridge point changes between GPU generations — operations that were compute-bound on V100 (ridge=124) may be memory-bound on H100 (ridge=295). Re-evaluate every kernel's regime classification at each GPU generation transition. The optimization strategy inverts.
- MFU below 25% in training almost never indicates a kernel problem — it indicates a systems problem (communication, scheduling, data pipeline). The most expensive debugging mistake is optimizing Layer 2 (GPU kernels) when a Layer 5 or 6 bottleneck exists.
- Fleet sizing must apply two multipliers after computing raw GPU count: divide by target utilization (0.75) and multiply by burst headroom (1.20). Skipping these produces a fleet that runs at 100% utilization — which means P99 queue time diverges and SLAs fail during any traffic spike.

Review Questions

FOUNDATIONAL Recall and reason — no derivation required

1. Without looking at the table: what is the H100 SXM5 HBM bandwidth, BF16 TFLOPS, and ridge point? What is the equivalent for MI300X? Why is the MI300X ridge point lower than H100's despite higher compute?
2. Explain why the H200 has the same TFLOPS as H100 but a lower ridge point. What kind of workloads benefit from H200 vs H100, and what kind see no benefit at all?
3. You have a 70B BF16 model at batch=1 on H100. MFU is 4.5%. Is this a problem? What should you do? Now at batch=64, MFU is 42%. Is this a problem? What should you do?

INTERMEDIATE Calculate and compare — production context

1. For LLaMA-3 70B: (a) Compute KV cache per request at S=8192 context in BF16. (b) How many H100 GPUs (after TP=2 for model) can serve this model with 20 concurrent requests each? (c) If you switch to INT8 KV and FP8 weights with TP=1 on MI300X, how many concurrent requests per GPU?
2. A 256-GPU H100 training cluster is running ZeRO-3. The model has 70B parameters. Compute the AllGather communication volume per forward pass (each param fetched once). At 900 GB/s NVLink, how long does this take? How does this compare to the GEMM compute time at MFU=50%?
3. TP=8 inference on LLaMA-3 70B, 80 layers, one AllReduce per layer of 0.5 GB each. Compare total per-inference AllReduce overhead on: (a) H100 NVLink 900 GB/s, (b) IB NDR 50 GB/s. At what TPOT SLA does IB become unusable (overhead > 10% of acceptable latency)?

PRINCIPAL LEVEL Design and defend — architecture reviews

1. Design a 70B inference fleet for 10K tokens/sec throughput, P99 TPOT <25ms, with 4K average context. Evaluate H100 (TP=2), H200 (TP=1), and MI300X (TP=1) — all BF16. Include: GPUs needed, capex at current market rates, annual OpEx at $3/GPU-hr, and TCO over 3 years. Recommend one with a quantified justification.
2. A system at 1,024 H100 GPUs achieves 38% MFU during LLaMA-3 405B training (TP=8, PP=8, DP=16). Theoretical maximum for this config is 54%. Walk through the diagnostic hierarchy: what you would check at each of the seven layers, what metric at each, and what the most likely root cause is given the 16-percentage-point gap.