Efficient LLM Checkpointing with Memory and Storage

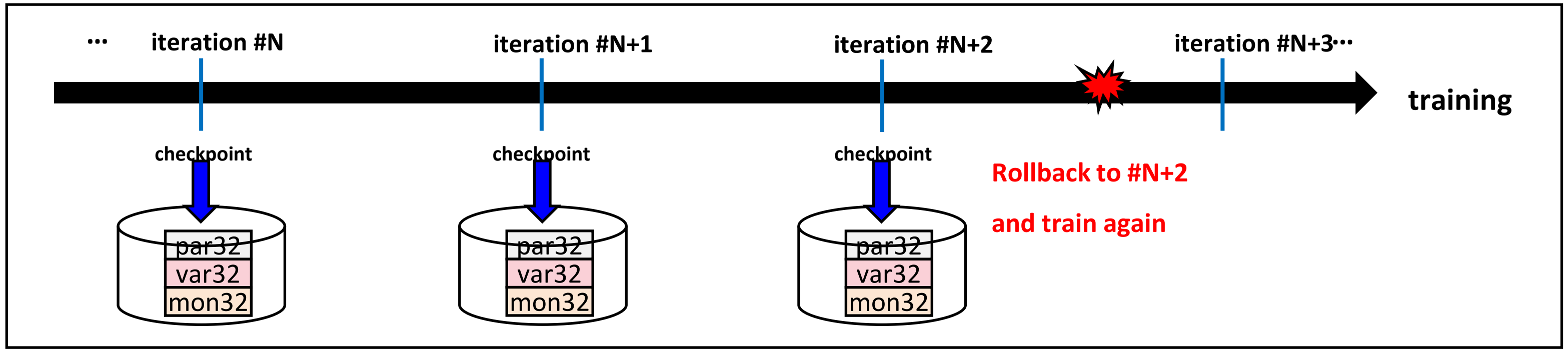
Jongryool Kim

SK hynix

Checkpointing for LLM Training System

Dealing with various issues while training such as H/W, power, interrupt issues, LLM systems need to be checkpointed frequently to be rolled back to a stable state

To restore the specific version for Model-Tuning : Resume from specific meaningful version for a model-tune training



LLM checkpointing writes the model parameters and optimizer state to persistent storage

Challenges on LLM Checkpointing

SK hynix

Large Checkpoint Size !!

With growing model size, total checkpoint size grow.

→ Large size of LLMs incurs significant I/O overheads

Model	# params (p)	Model size (p*2byte)	total checkpoint state size
GPT-3	175 B	350 GB	2.4 TB
LLaMa	544 B	1088 GB	7 TB
GPT	1 T	2 TB	13.8 TB

https://arxiv.org/pdf/2406.10707v1, DataStates-LLM: Lazy Asynchronous Checkpointing for LLMs

Frequent Checkpointing !!

More frequent LLM checkpoints (becomes mandatory due to frequent GPU failures with large LLM data)

→ incurs system overheads such LLM training performance degradation and storage capacity

Checkpointing every 5 iterations

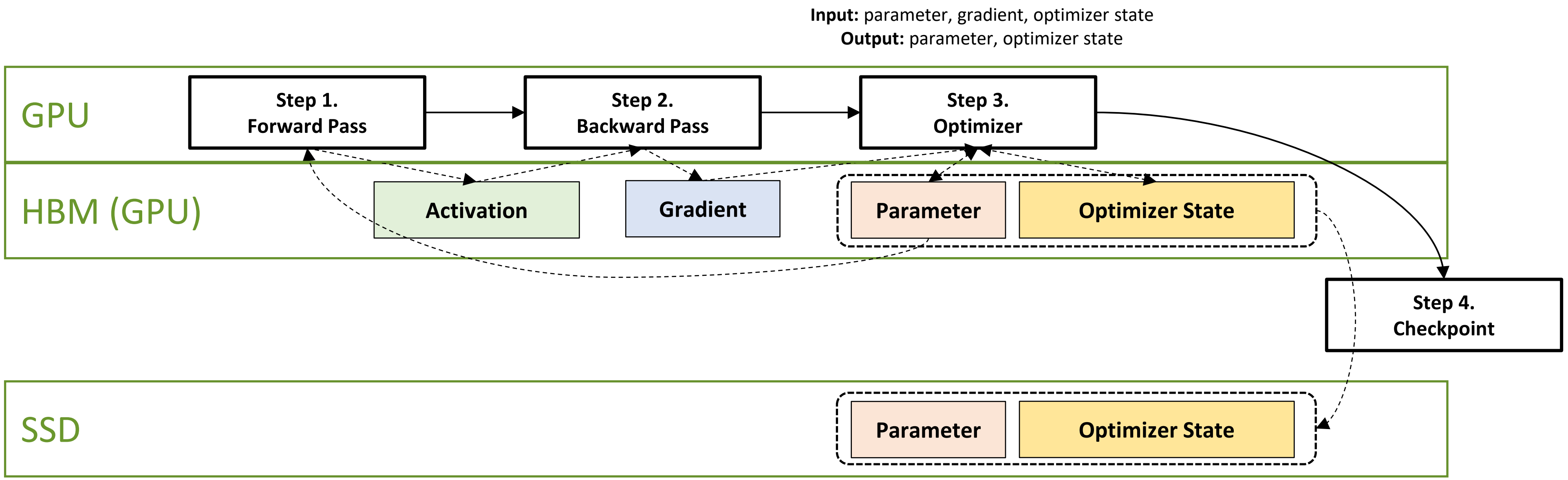
Iteration	1	2	3	4	5	6	7	8	9	10
Para32										
Mon32										
Var32										

Checkpointing every iteration

Iteration	1	2	3	4	5	6	7	8	9	10
Para32										
Mon32										
Var32										

LLM Training

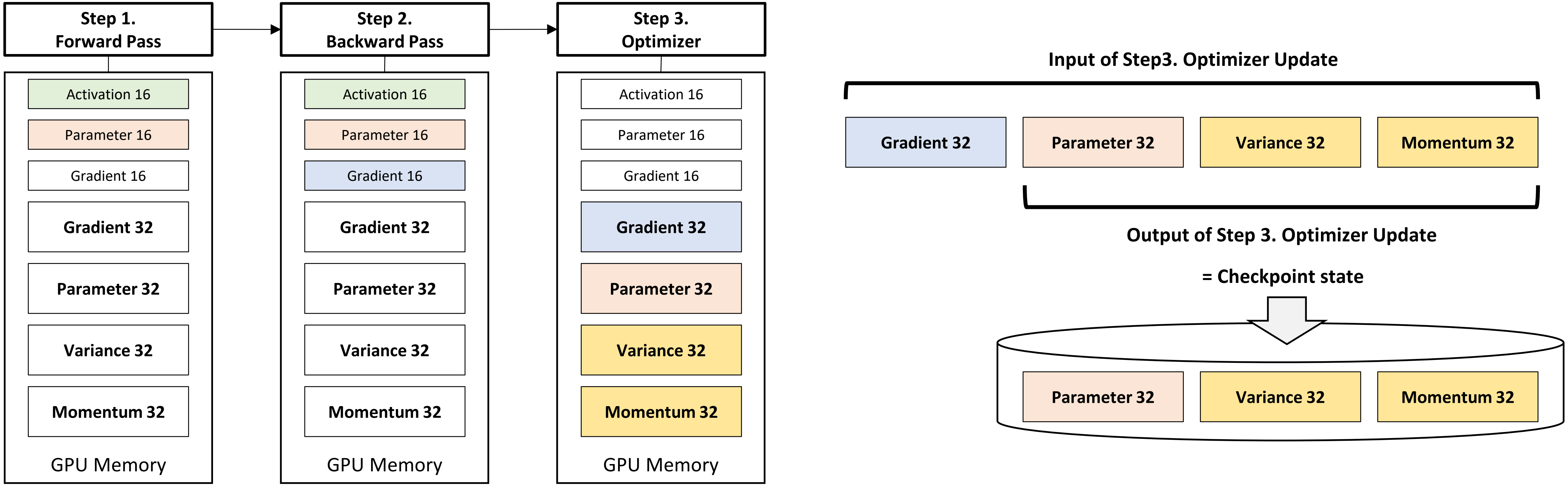
Forward → Backward → Optimizer Update → Checkpoint



Memory Inefficiency

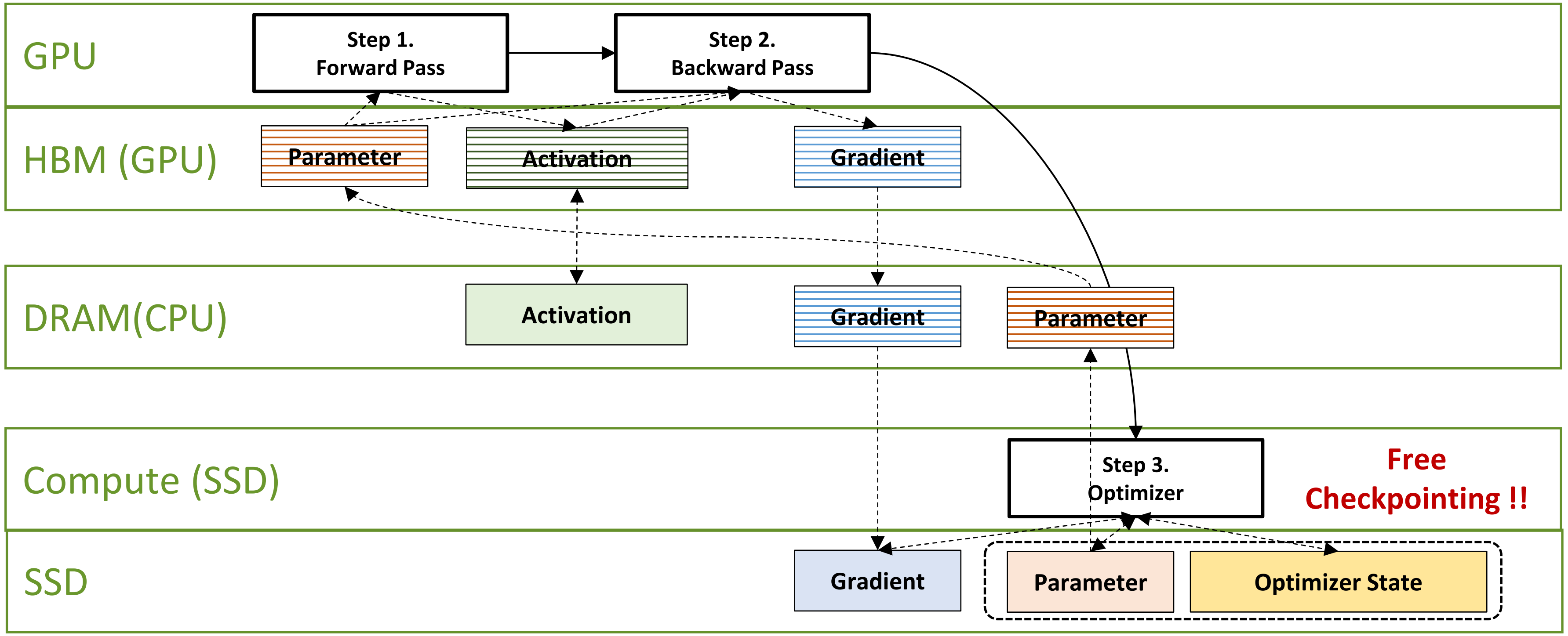
Forward & Backward only use paramter16, activation16, and gradient16

Optimizer state is kept in high precision (FP32 – G32, P32, V32, M32)



Offloading Optimizer to SSD

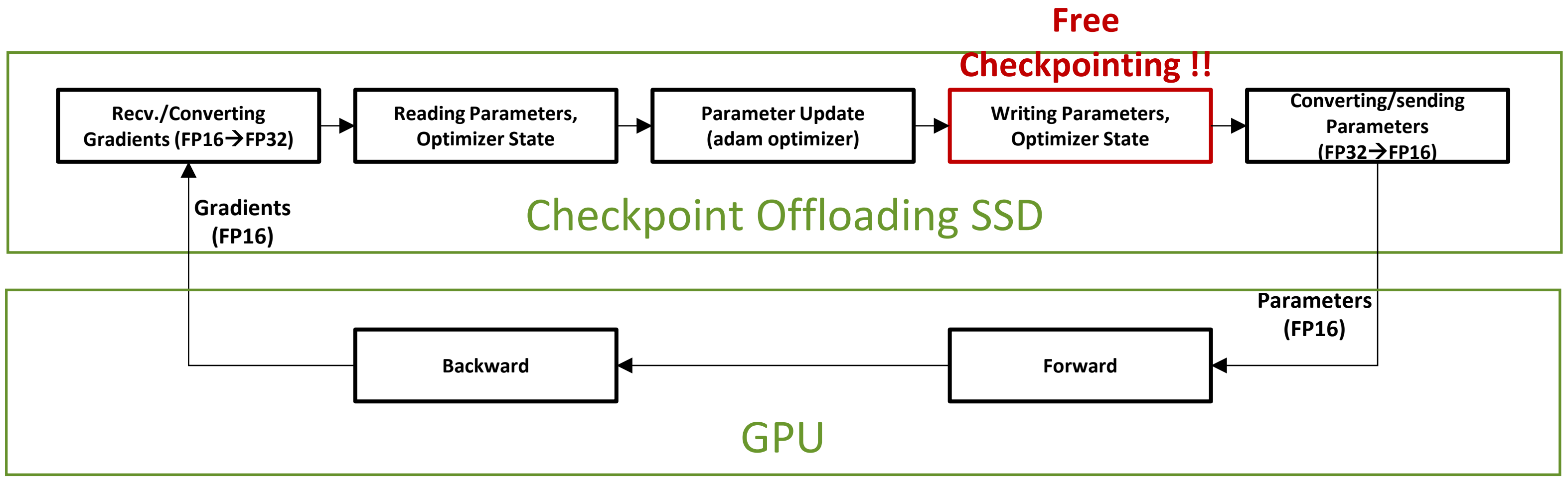
Forward → Backward → Optimizer Update



Checkpoint Offloading SSD

Checkpoint Offloading SSD generates checkpoint states in SSD

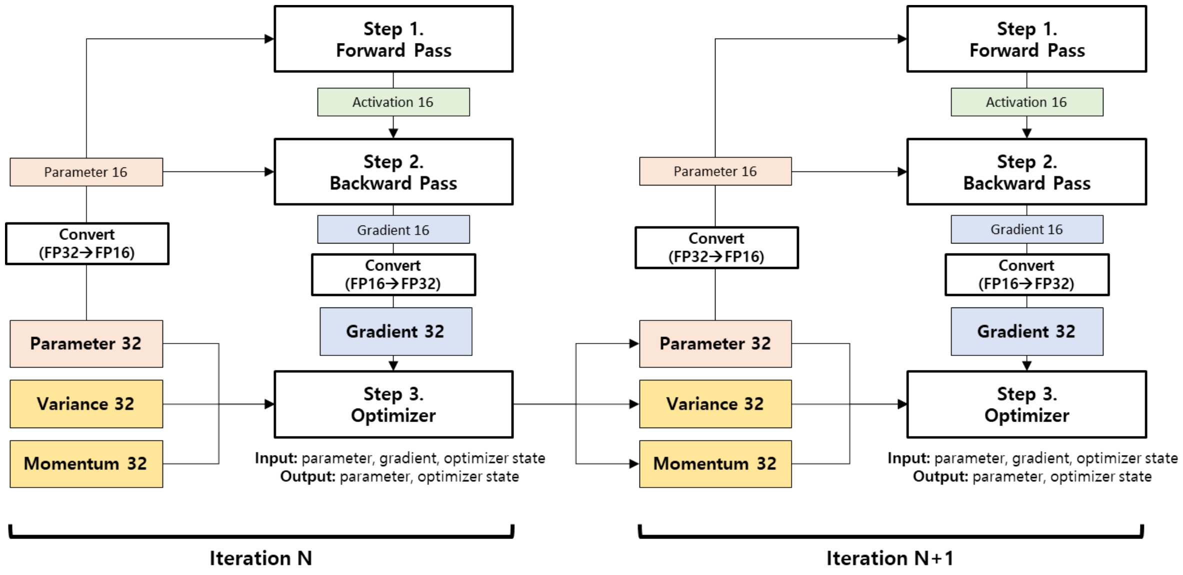
- Generating checkpoint states is an elementwise operation with low operational intensity (# arith. ops. / # mem. ops.) → can be offloaded from GPU
- GPU memory saving → larger model training
- Parameter restore (not optimizer) → shorten checkpoint restoration time
- high frequency checkpointing → Save PCIe + network B/W by avoiding transfer of optimizer state



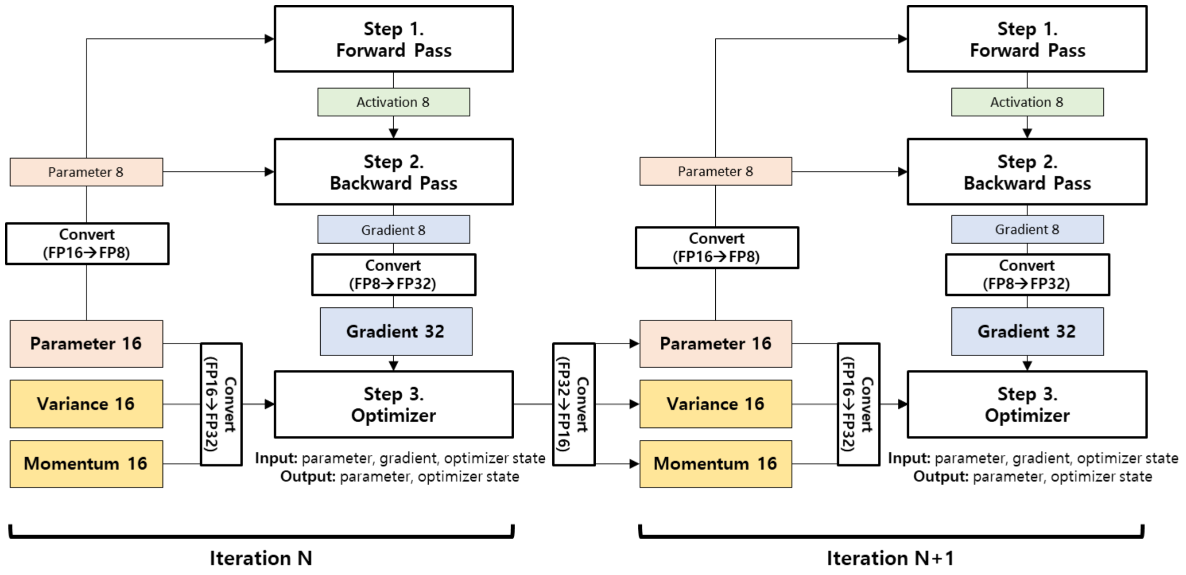
Mixed Precision and Optimizer

Support mixed-precision training to take advantage of low precision compute speedup while maintaining accuracy

- Fwd./Bwd.: High compute, low memory requirement (low precision model, gradients, activations)
- Optimizer: Low compute, high memory requirement (high precision optimizer state)



[Mixed Precision (FP32 ←→ FP16)]

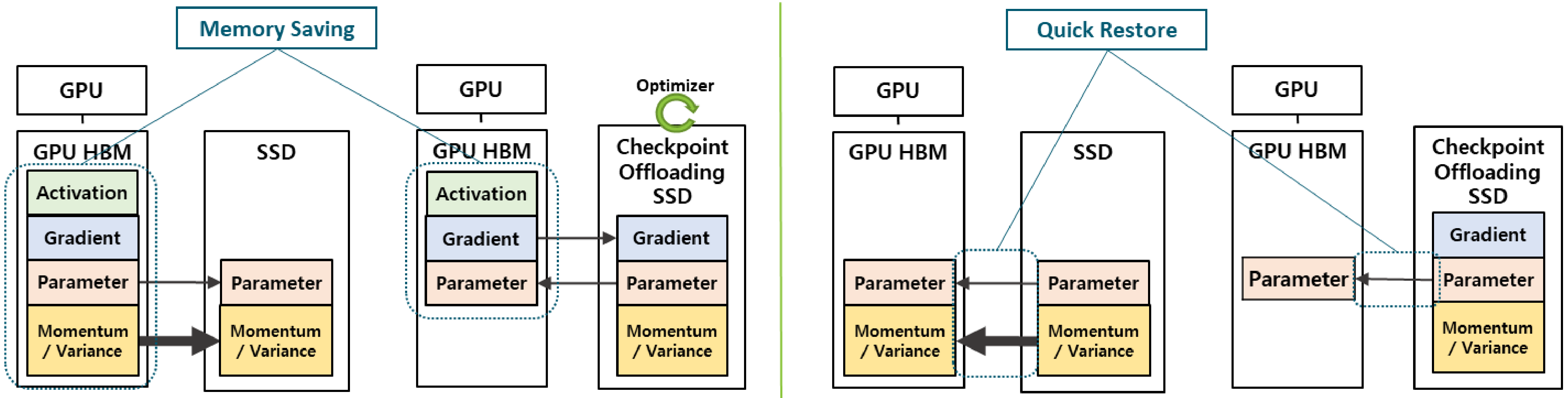


[Mixed Precision (FP16 → FP8, FP32 ←→ FP16, FP8 → FP32)]

Benefit of Checkpoint Offloading SSD

Can save GPU memory usage to train a larger model or increase the batch size

Shorten checkpoint restoration time



[LLM Training System]

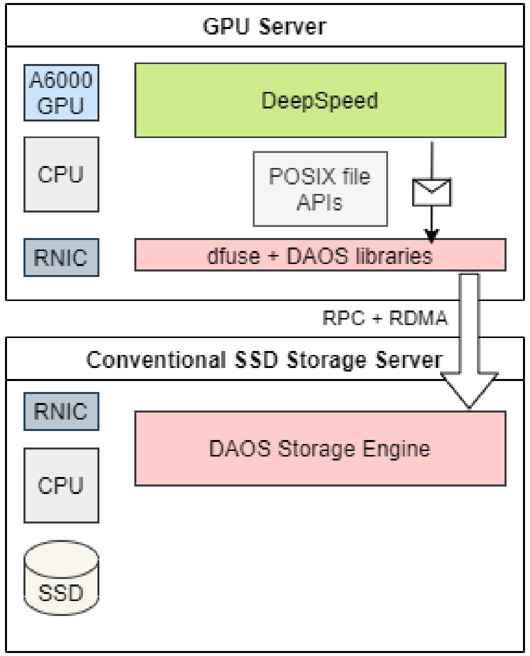
[LLM Training System w/ Checkpoint offloading SSD]

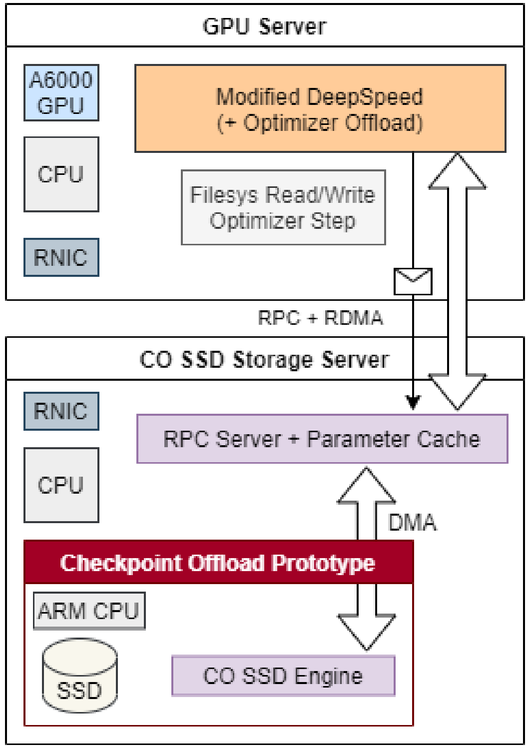
Free Checkpointing !!

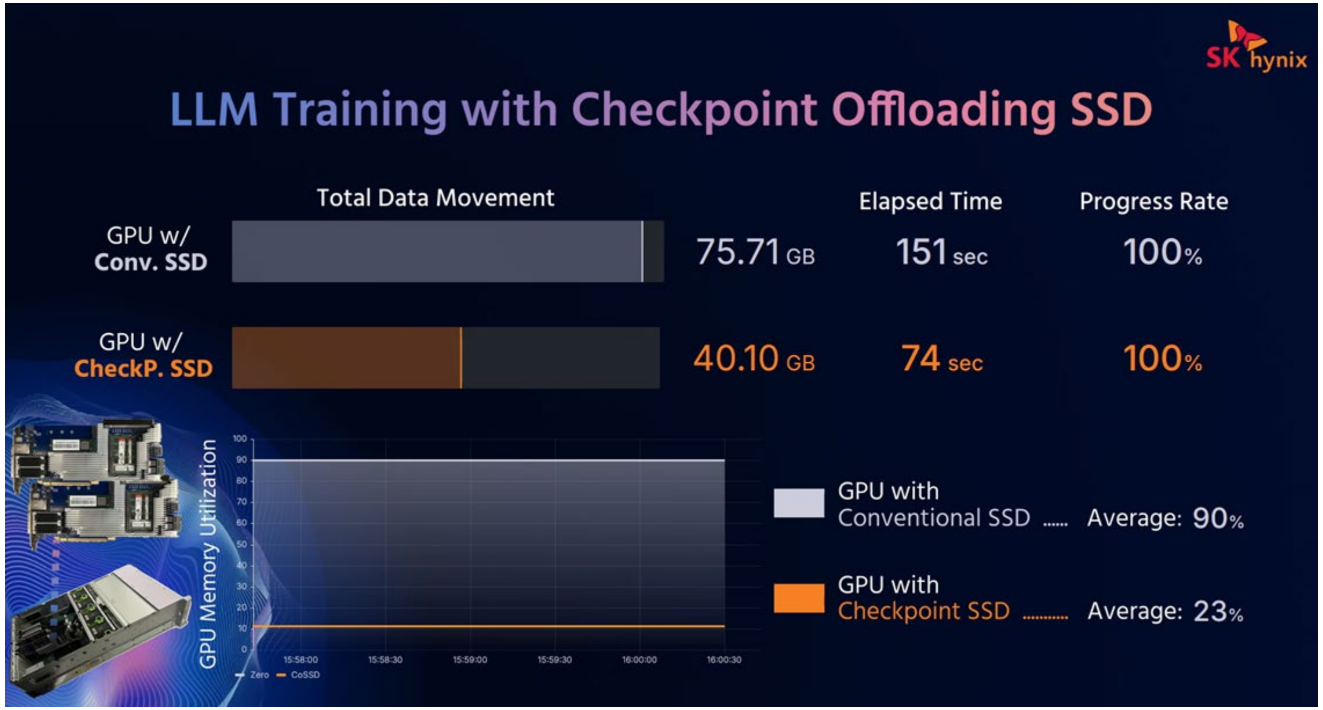
[LLM Training System]

[LLM Training System w/ Checkpoint offloading SSD]

Performance

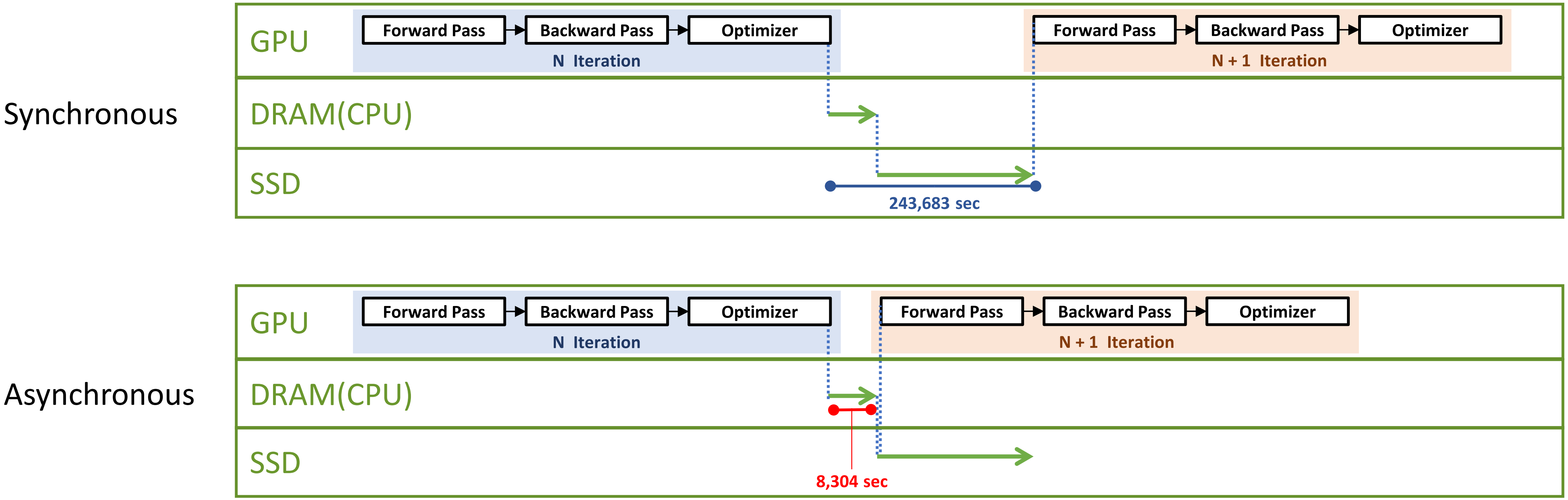




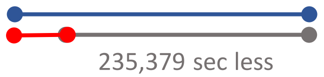


Asynchronous Checkpointing

* Benchmarked with GPT2-2B

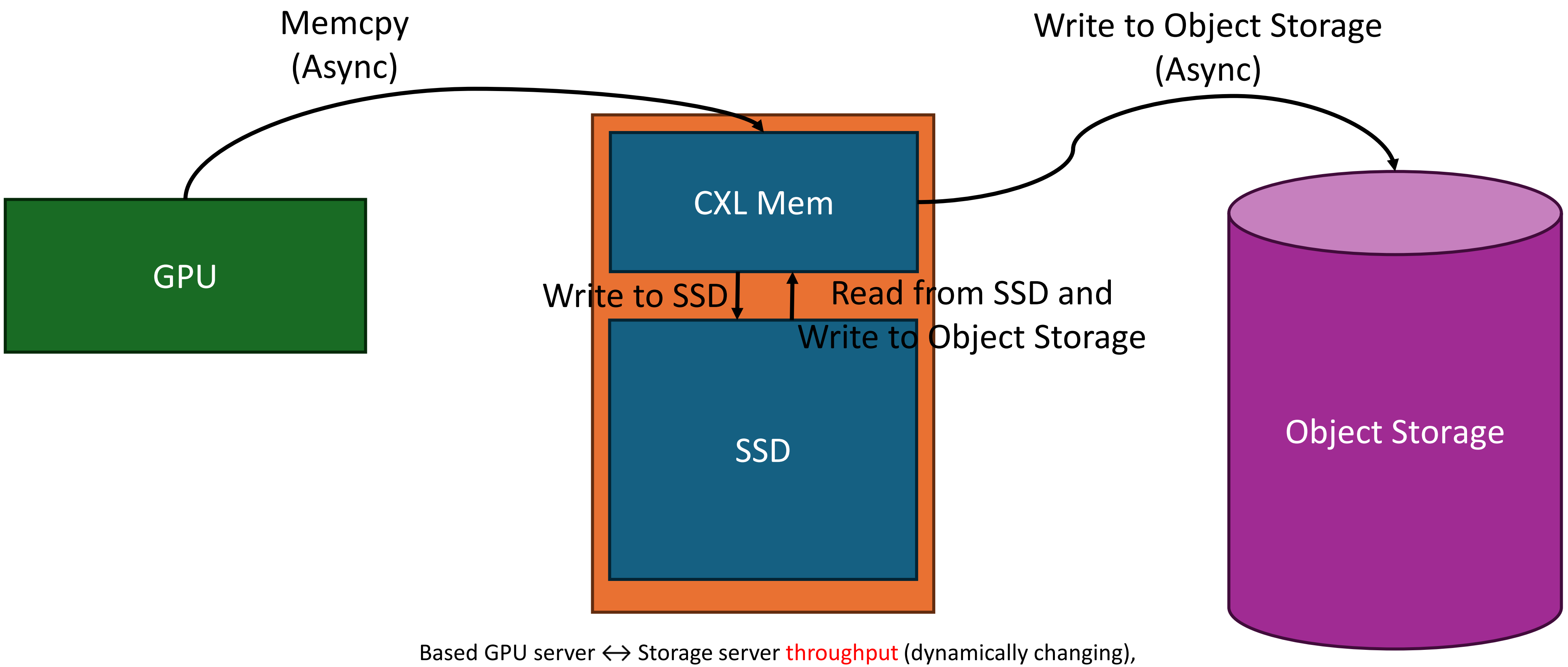


- Asynchronous checkpointing is much faster



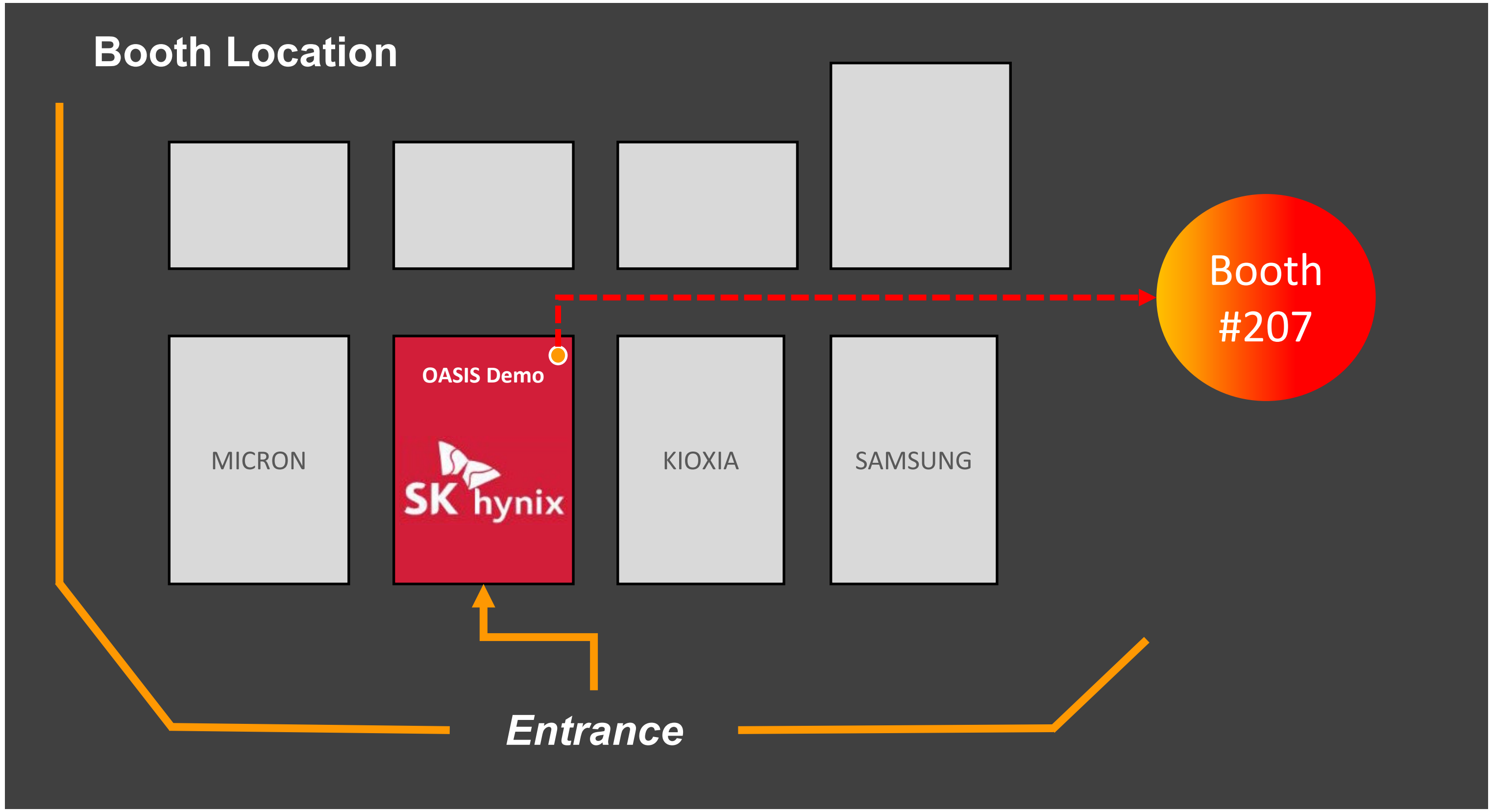
- GPU is making progress faster, but the CPU memory can get filled quickly due to slow SSD

Asynchronous Checkpointing with Hybrid CXL Memory



Based GPU server ↔ Storage server throughput (dynamically changing), policy determines whether to write/keep checkpoint data in local SSD or move to remote object storage.

Learn more about SK hynix



Visit Booth #207 and Experience SK hynix products and demos

Booth #207

Meet the future of memory.
Just steps from the entrance.

Innovation starts here,

Literally.

SK hynix