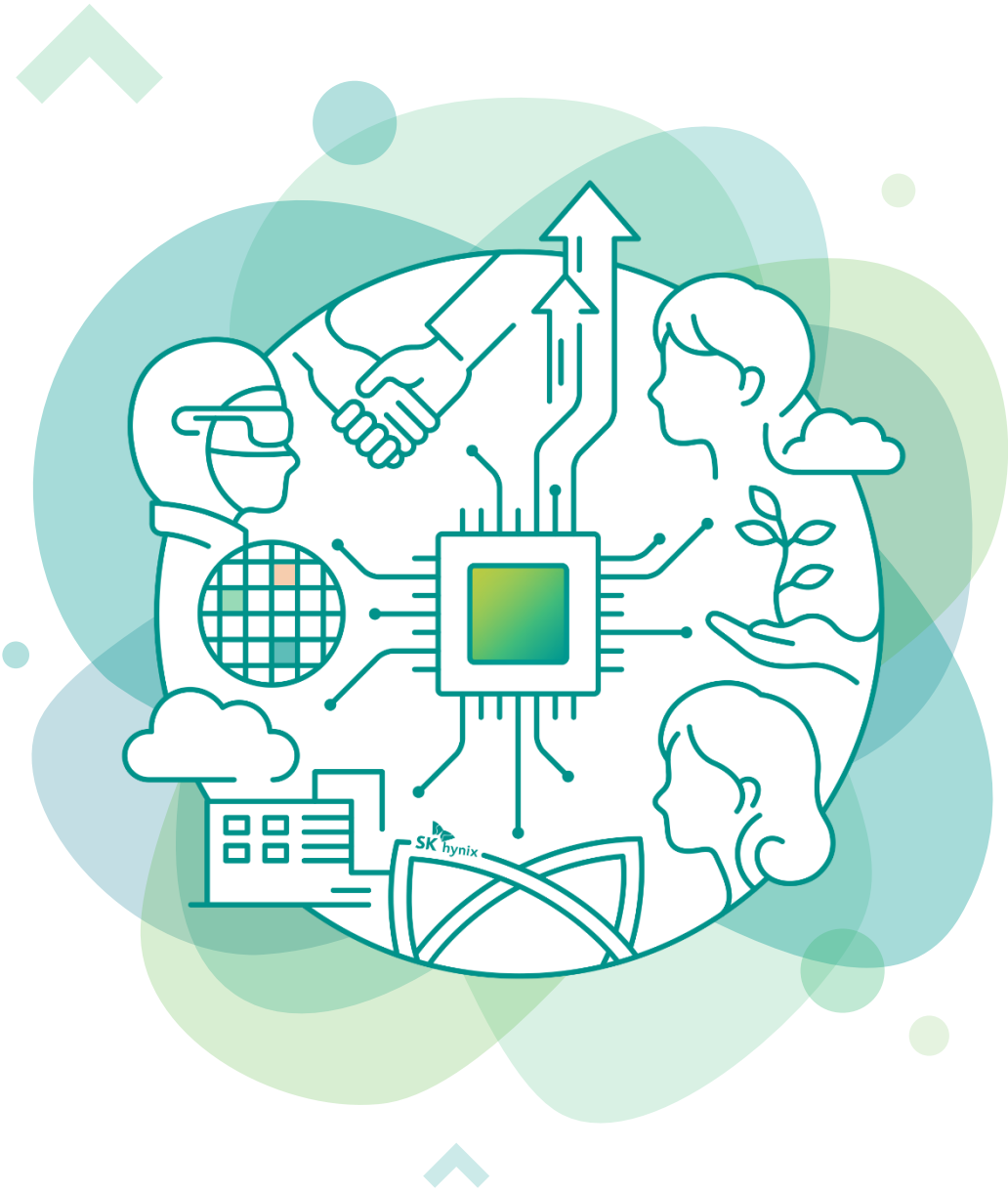
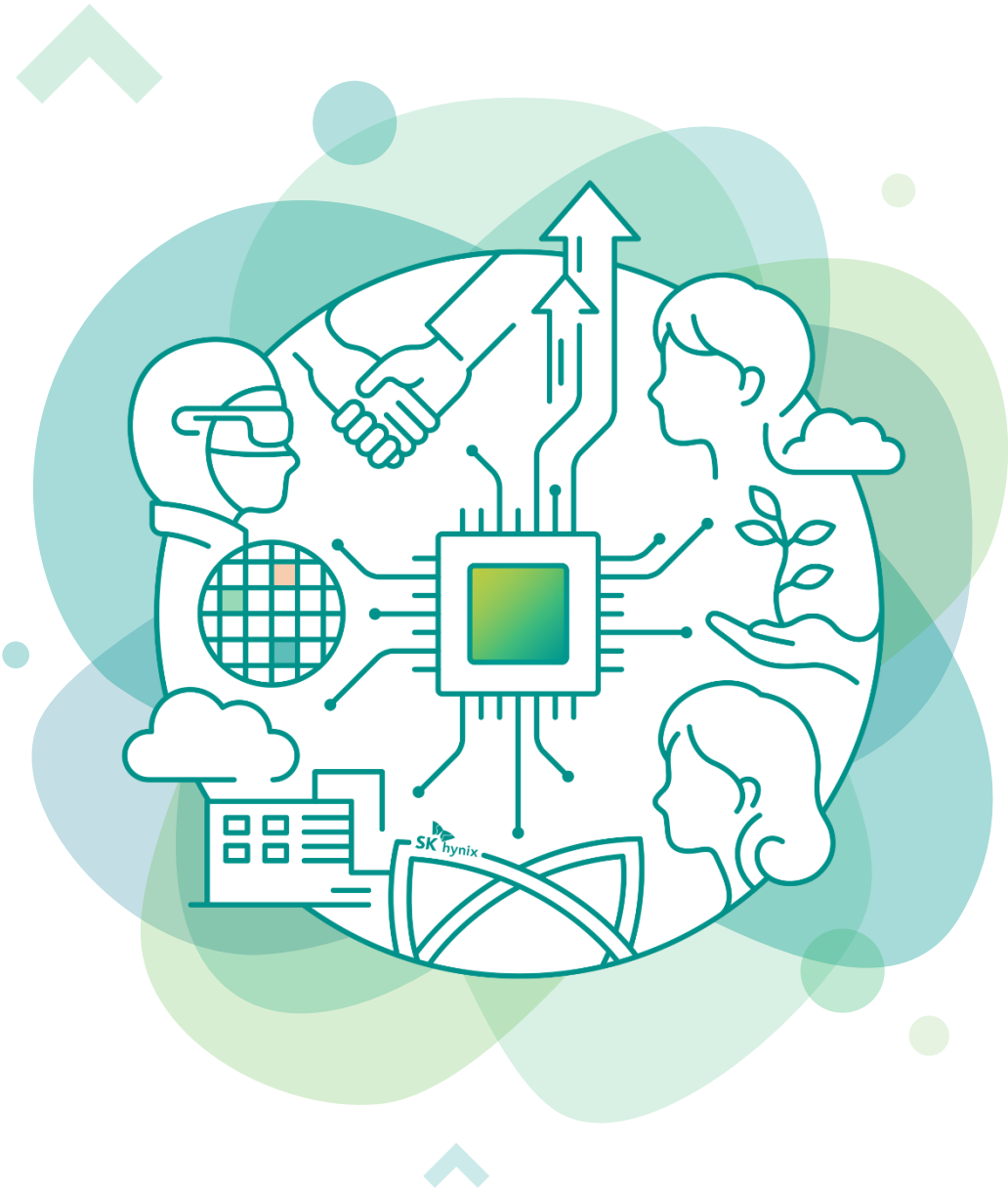
# $H^3$: Hybrid Architecture Using High Bandwidth Memory and High Bandwidth Flash for Cost-Efficient LLM Inference



**Minho Ha, Ph.D.**
**Senior Staff Engineer**
**Advanced Technology Pathfinding**
**Memory Systems Research**

**© 2026. SK hynix Inc. All Rights Reserved.**
Specifications and designs are subject to change without notice.
All data were deemed correct at time of creation. SK hynix is not liable for errors or omissions.

# In-depth Discussion on HBF: Opportunities and Challenges



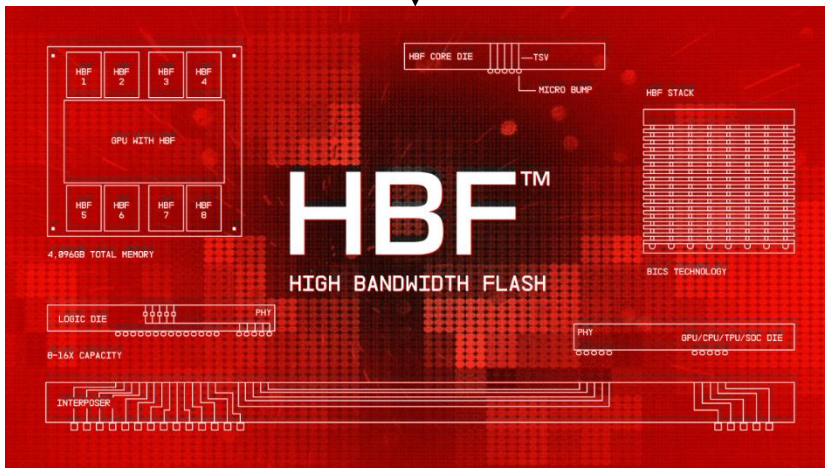
**Minho Ha, Ph.D.**
**Senior Staff Engineer**
**Advanced Technology Pathfinding**
**Memory Systems Research**

**© 2026. SK hynix Inc. All Rights Reserved.**
Specifications and designs are subject to change without notice.
All data were deemed correct at time of creation. SK hynix is not liable for errors or omissions.

# HBF Timeline

Today's Talk

| 2025.02 | 2025.07 | 2025.08 | 2026.02 | 2026.02 | 2026.08 | 2026.08 |
|---|---|---|---|---|---|---|
| HBF debut Sandisk Investor Day | HBF Technical Advisory Board | SK hynix–Sandisk MOU | $H^3$ paper published IEEE CAL | HBF Consortium Kickoff | OCP HBF Spec v0.7.0 released | HBF presentations FMS 2026 |




Source: Sandisk




Source: SK hynix Research page



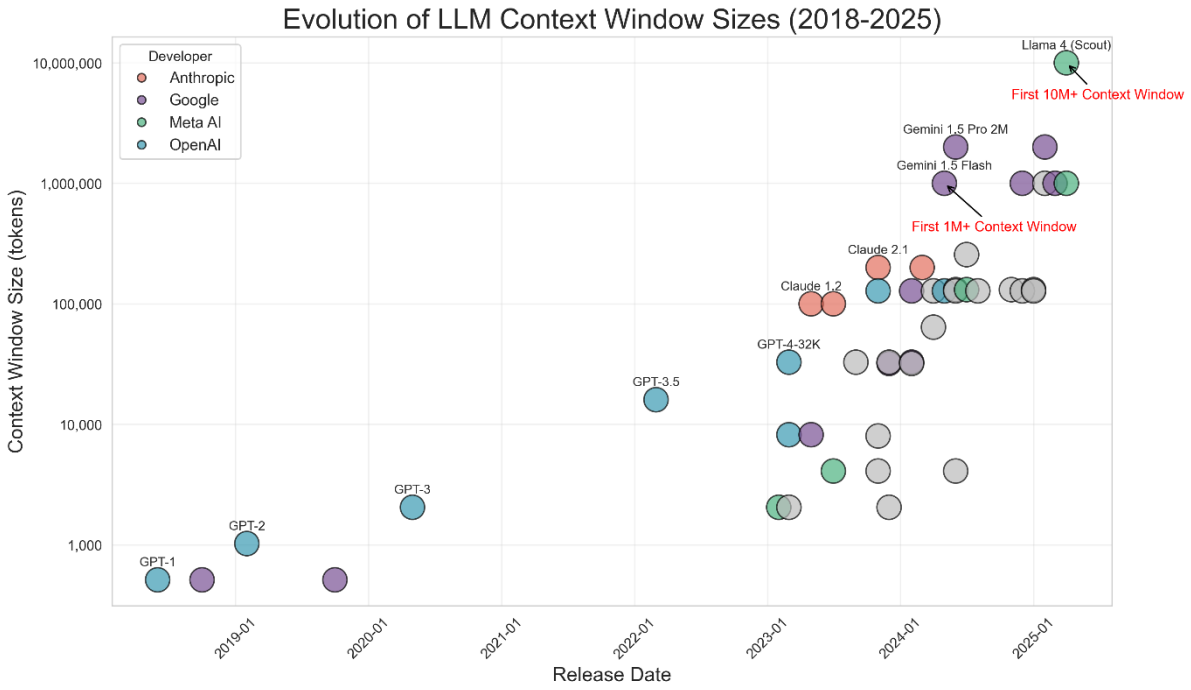

Source: SK hynix newsroom

SK hynix © 2026. SK hynix Inc. All Rights Reserved.

# Motivation: Need for Larger Memory in LLM Inference

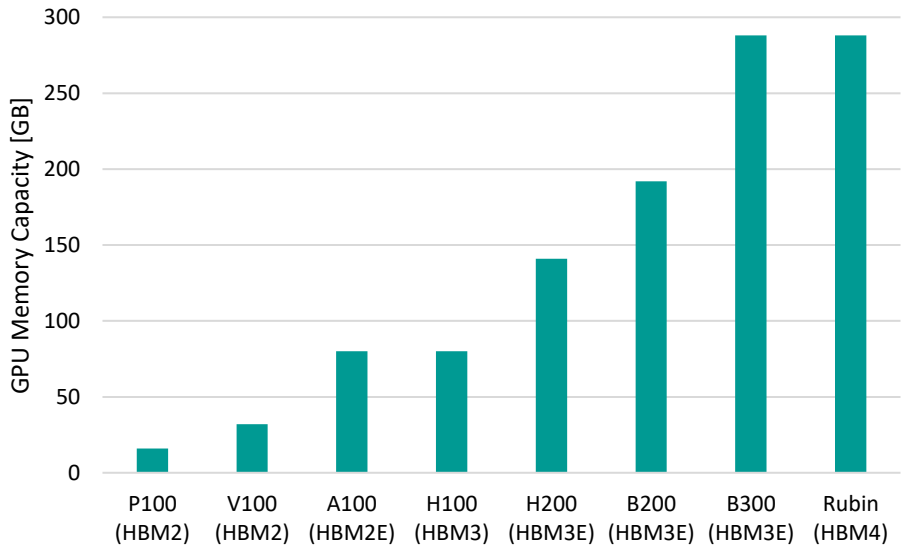
- ○ LLM은 Chatbot에서 Multimodal AI/Agentic AI로 진화하고 있으며, 모델의 Context Window 또한 커지고 있음 (예: Llama 4는 최대 10M 길이 지원)
- ○ Long Sequence 처리에 필요한 거대한 KV Cache 저장 공간이 HBM 용량 한계로 제한됨
- ○ CAG (Cache-Augmented Generation)와 같은 거대 읽기 전용 데이터 사용 사례는 HBM 용량을 극한으로 소모

### LLM Model Size 변화




Source: https://www.meibel.ai/post/understanding-the-impact-of-increasing-llm-context-windows

### NVIDIA GPU HBM 용량 변화



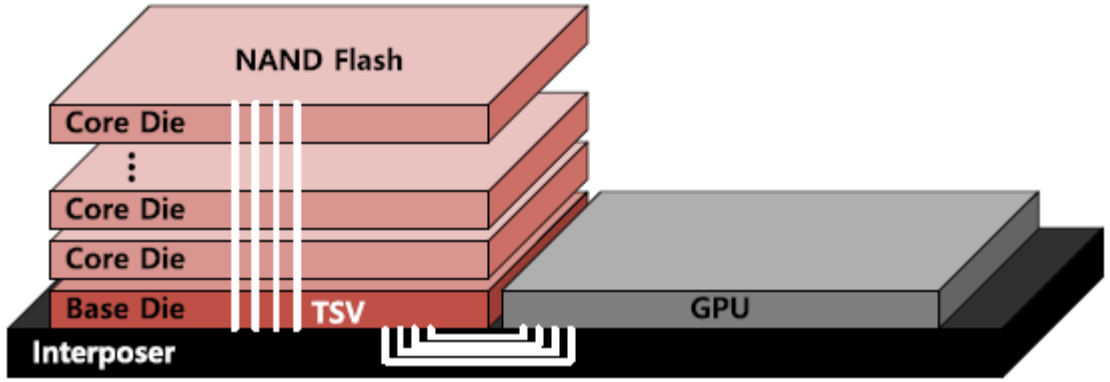

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Capacity [GB]** | 16 | 32 | 80 | 80 | 141 | 192 | 288 | 288 |
| **Bandwidth [TB/s]** | 0.72 | 0.9 | 2 | 3.35 | 4.8 | 8 | 8 | 22 |

SK hynix © 2026. SK hynix Inc. All Rights Reserved.

# High Bandwidth Flash (HBF): Opportunities and Challenges

- **HBM의 용량 부족을 극복하기 위한 방안으로 HBF가 제안됨**
- **HBF: NAND Flash Die를 수직으로 적층하고 TSV로 연결하여 HBM 급 대역폭과 최대 16배 용량 제공**
  - 단점: 1) ~1,000배 느린 접근 지연(ns vs. µs), 2) 낮은 Write Endurance, 3) 높은 전력 소모
- **낮은 Write Endurance와 Write Performance로 인해 HBF는 Read-Heavy Workload에서 효과적일 것으로 예상**

**HBF Concept**




**HBM vs. HBF***

| Metric | HBM (Normalized) | **HBF (Relative to HBM)** |
|---|---|---|
| Capacity | 1 | **16** |
| Bandwidth | 1 | **1** |
| Power Consumption | 1 | **4** |

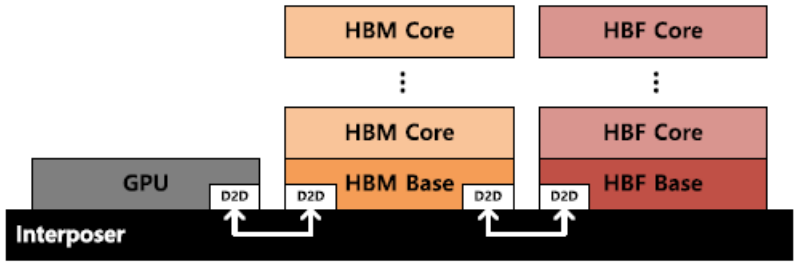
* OCP Spec. v0.7 공개 전, IEEE CAL 논문 내 사용된 수치로 실제 PoC 및 제품과는 상이

SK hynix © 2026. SK hynix Inc. All Rights Reserved.

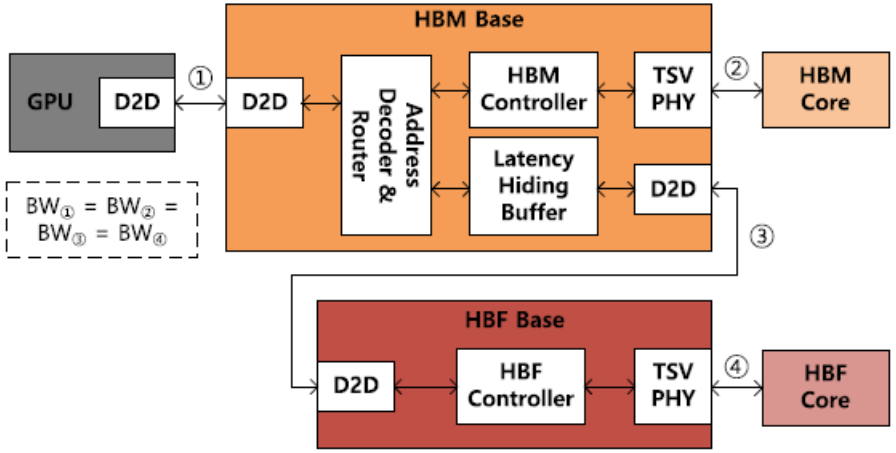
# $H^3$: Hybrid Architecture with HBM and HBF (1/2)

- ○ HBM과 HBF를 병렬 활용하는 Hybrid Memory Architecture 제안
- ○ GPU Shoreline에는 HBM을 연결하고, HBM과 HBF는 Daisy-Chain 연결
- ○ GPU가 HBM Base Die의 Address Decode를 통해 HBF에 직접 접근 ➔ Unified Address Space
  - ▪ GPU 입장에서 HBM과 HBF 모두 “Main Memory”

$H^3$ Architecture Overview




(a) Side View of $H^3$




(b) $H^3$ Sub-system Architecture




(C) Address Space Example of $H^3$

SK hynix © 2026. SK hynix Inc. All Rights Reserved.

# $H^3$: Hybrid Architecture with HBM and HBF (2/2)

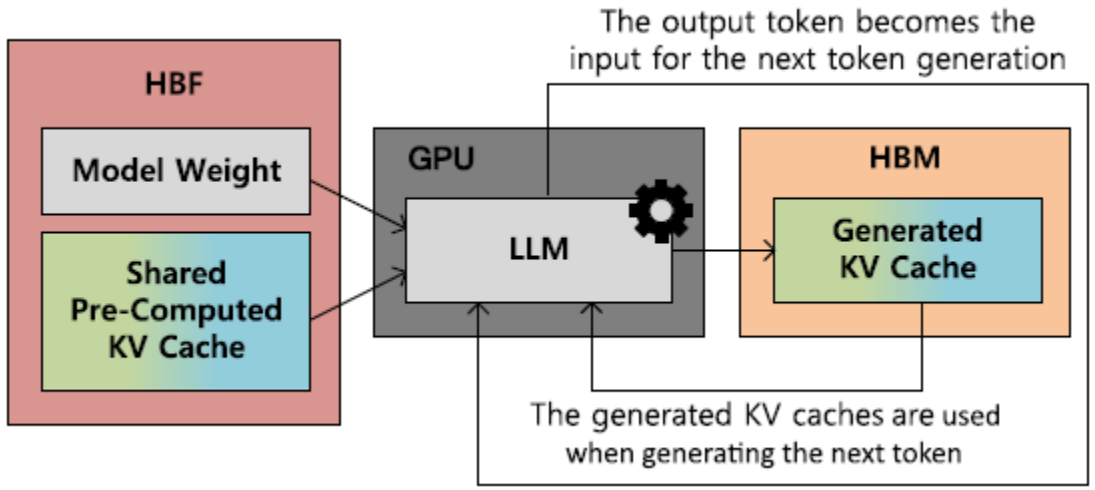
## ○ Strategic Data Placement for Read-Only Workloads

- **Read-only Data (Model Weight + Shared Precomputed KV Cache) ➔ HBF 저장**
- **Write Data (Generated KV Cache) ➔ HBM 저장**
- **LLM Inference의 Deterministic Data Pattern에 기반한 Tensor-level Scheduling을 통해 HBM-HBF 경쟁 최소화 가능**

## ○ Latency Hiding Buffer (LHB)

- **HBF의 긴 NAND Flash Access Latency (예: 20µs)을 숨기기 위한 LHB 도입**
- **LLM Inference의 Sequential·Deterministic 특성으로 인해 다음 Layer에서 필요한 Tensor 예측 가능**
- **Double Bufferiing 가정 시 40MB Capacity, 8.06mm² Silicon Area로 Base Die의 6.7% 수준으로 구현 가능**
  - Capacity: 2×1TB/s×20µs = 40MB (Double Buffering, SLC NAND Flash Latency 기준)
  - 3nm SRAM 기준 Core Area 6.72mm², Additional Circuit Overhead 포함 8.06mm²

**$H^3$ Usage: CAG and Model Weights**




SK hynix © 2026. SK hynix Inc. All Rights Reserved.

# Evaluation Methodology

○ **실기 측정 불가로 Analytical Modeling 기반 In-house Simulator 개발**

- 최근 진행 중인 연구에서는 NVIDIA AIConfigurator (https://github.com/ai-dynamo/aiconfigurator) 기반으로 확장

○ **B200 GPU, HBM3e (192GB, 8TB/s, 40W), HBF (3TB, 8TB/s, 160W) 가정**

○ **Workload: Llama 3.1 405B 기반 1M/10M Sequence Length, 각 최소 GPU 수(8개/32개)로 비교**

- 실제 Llama 3.1 405B의 지원 가능 Context Window를 초과

SK hynix © 2026. SK hynix Inc. All Rights Reserved.

# Evaluation Results

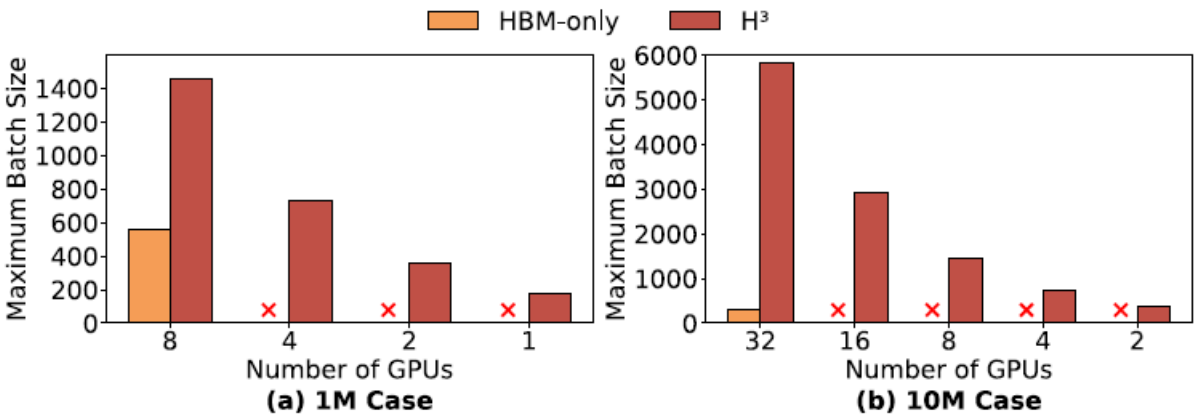
## ○ Maximum Batch Size

- 1M Sequence Length: $H^3$ 적용 시 최대 2.6배 Batch Size 확장
- 10M Sequence Length: 최대 18.8배 Batch Size 확장
- $H^3$ 적용 시, 1M Sequence는 단일 GPU, 10M Sequence는 2개 GPU로 실행 가능 ➔ 극저비용 Long Sequence 처리 입증

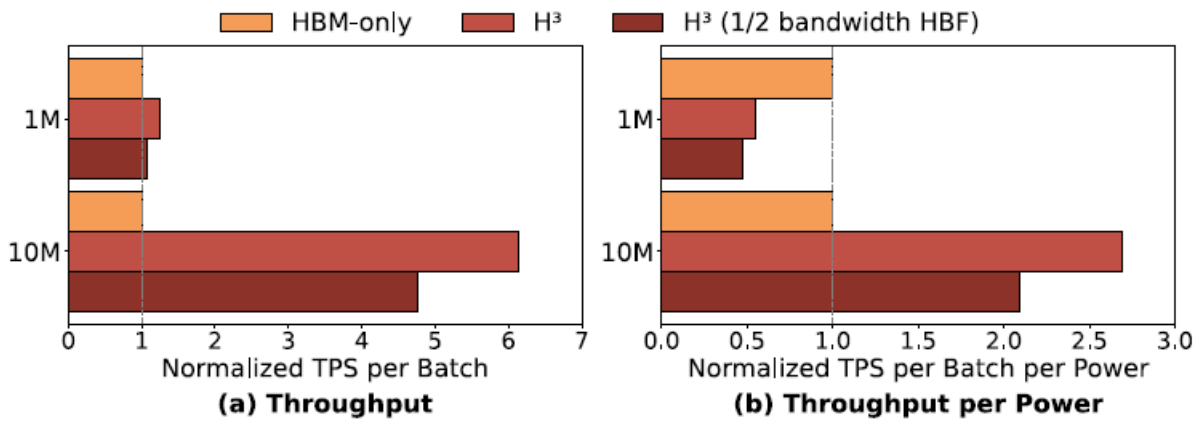
## ○ System Throughput & Power Efficiency

- System Throughput: 1M Sequence 1.25배, 10M Sequence 6.14배 증가
- Power Efficiency: 최대 2.69배 향상 (HBF 추가로 인한 전력 증가 Overhead를 뛰어넘음)
- HBF 대역폭이 절반으로 줄어도 HBM 전용 대비 10M Sequence에서 2.09배 Power Efficiency 달성

**GPU 개수에 따른 Maximum Batch Size**




**System Throughput & Power Efficiency**




SK hynix © 2026. SK hynix Inc. All Rights Reserved.

# Conclusion and Future Work

- $H^3$는 HBM 용량 한계를 HBF와의 Hybrid Architecture로 극복
- Gigantic Read-only KV Cache와 Model Weight 저장에 HBF를 활용함으로써 Maximum Batch Size, System Throughput, Power Efficiency 대폭 향상 가능성 확인
- 향후 연구: HBF가 탑재된 다양한 Architecture Exploration, HBF 적합 Workload 검토

SK hynix © 2026. SK hynix Inc. All Rights Reserved.

# After $H^3$ Paper…

- 다양한 HBF 연구들이 등장 (수많은 IEEE CAL, arXiv, IMW'26 논문들 등)
- 그러나, 아직 실물이 없는 HBF ➔ System-/Device-level Simulation 연구 위주

- 많은 사람들이 대용량 KV Cache(=Context)를 저장하고 싶어하지만…
- 많은 사람들이 제2의 HBM이라고 얘기하지만…

**Architecture 관점, Device 관점, Workload 관점 심도있는 논의가 필요**

SK hynix © 2026. SK hynix Inc. All Rights Reserved.

# Discussion 1) HBF-suitable Workloads

- HBF의 장점: 1) HBM 대비 대용량, 2) HBM 수준의 대역폭
- HBF의 단점: 1) 느린 Access Latency, 2) 낮은 Write Performance 및 Endurance, 3) 높은 Cube당 전력 소모, 4) Page-level Granularity

- 장점을 부각하려면? 1) 기존 HBM에 담을 수 없는 크기, 2) Memory BW-bound 응용
- 단점을 상쇄하려면? 1) Prefetch 가능 (즉, 예측 가능), 2) Read-heavy, 3) Coarse-grained Access, 4) Sparse Access

- **Question 1) 위 모든 조건들을 만족하는 응용이 존재할까? 있다면 무엇일까?**
- **Question 2) 위 모든 조건들을 만족할 수 없다면, 그 중에서 우선순위?**
- **Question 3) Write가 발생하는 응용에 HBF를 활용하기 위한 방안은?**

SK hynix © 2026. SK hynix Inc. All Rights Reserved.

# Discussion 2) HBF-equipped System

- 다양한 System 구성 가능: HBF-only, Shorline Sharing, Daisy-chaining, Hybird Stacking 등
- HBF-only
  - 장점: 초고대용량, 면적 증가 최소화
  - 단점: NAND Flash 단점 그대로 승계
- Shoreline Sharing
  - 장점: 면적 증가 최소화
  - 단점: 다른 메모리와 HBF가 BW를 나눠가짐
- Daisy-chaining
  - 장점: 다른 메모리와 HBF 장점 모두 취함
  - 단점: 면적 증가, 전력 증가
- Hybrid Stacking
  - 장점: 다른 메모리와 HBF 장점 모두 취함
  - 단점: 구현 가능성 (개별 TSV? 공유 TSV?)

- Question 4) 어떤 구조가 가장 좋은 구조일까?
- Question 5) HBF와 함께 사용되기 좋은 이기종 메모리 파트너는? HBM? LPDDR? GDDR? 3D-stacked DRAM?

SK hynix © 2026. SK hynix Inc. All Rights Reserved.

# Discussion 3) Others

○ SW가 바라봤을 때 HBM과 HBF의 차이: 1) Access Granularity, 2) Access Latency

○ Question 6) 기존의 HBM을 지원하는 Software가 어떻게 바뀌어야 할까?

○ 아직까지 깊게 고민되고 있지 않은 것들: 1) ECC, 2) Garbage Collection, 3) Wear-leveling

○ Question 7) HBF의 ECC는 어느 수준이어야 할까? LDPC? BCH? RS? Hamming?

○ Question 8) HBF도 SSD처럼 Firmware가 있어야 할까? 필요하다다면 어느 수준? HW로 처리 불가?

SK hynix © 2026. SK hynix Inc. All Rights Reserved.

# One Last Question

**HBF가 HBM을 넘어 차세대 AI Memory의 한 축을 차지할 수 있을까?**

SK hynix © 2026. SK hynix Inc. All Rights Reserved.

# Let's Find the Answers Together




SK hynix © 2026. SK hynix Inc. All Rights Reserved.