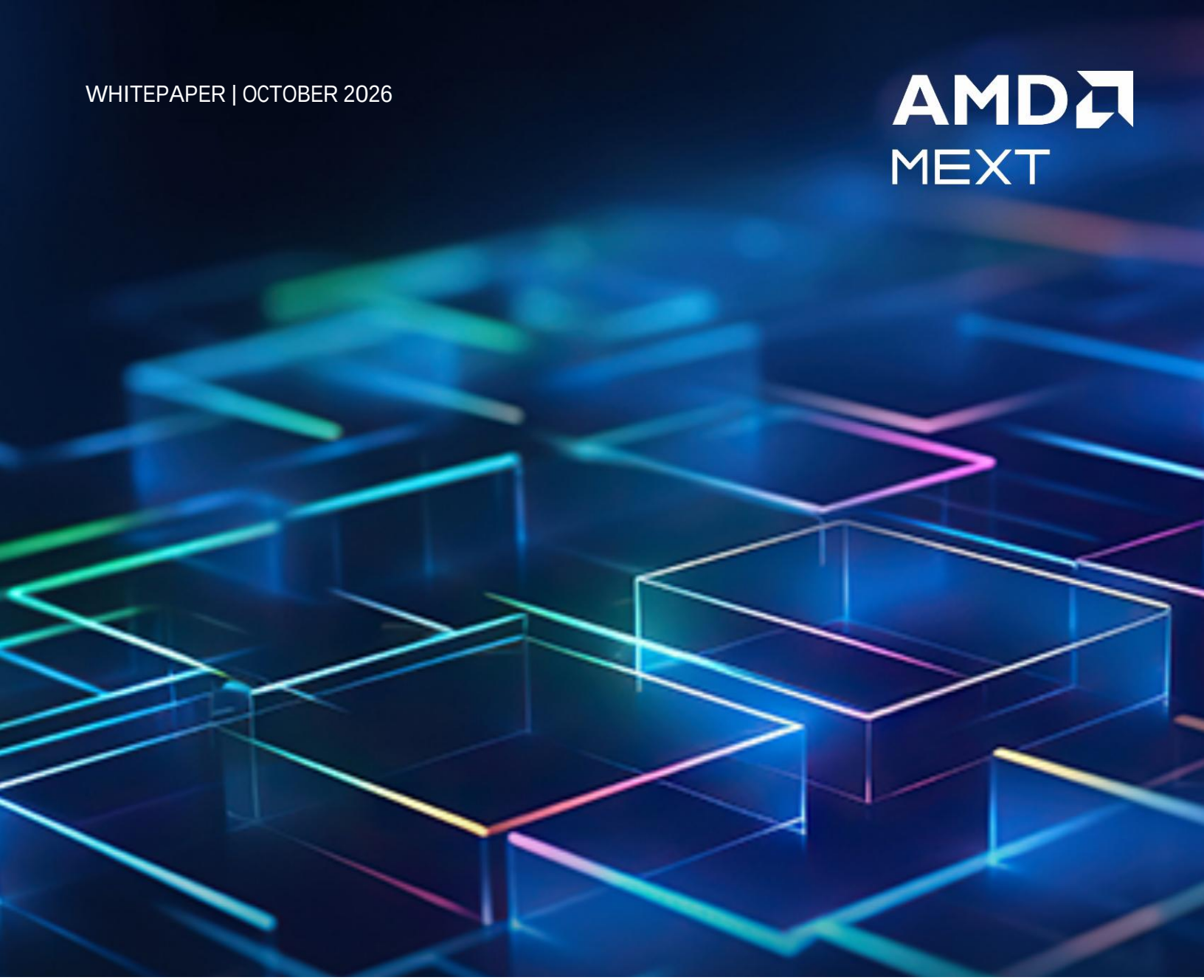
WHITEPAPER | OCTOBER 2026

AMD MEXT



DRIVING ~40% MORE THROUGHPUT ON THE SAME CLOUD INSTANCES FOR STATIC TIMING ANALYSIS

AMD MEXT for Synopsys® PrimeTime® on Microsoft Azure HX-Series VMs

EXECUTIVE SUMMARY

This whitepaper highlights how AMD MEXT Predictive Memory™ software can increase throughput for memory-intensive EDA signoff workloads by adding flash-backed memory capacity. This study increased workload concurrency and measured completed work per unit time on the same Microsoft Azure HX-series virtual machine. AMD internal testing showed that MEXT enabled four concurrent Synopsys® PrimeTime® jobs on the VM, compared with two concurrent jobs on the 1.4TB DRAM-only baseline instance. The MEXT configuration delivered 0.6029 jobs per hour versus 0.4314 jobs per hour for the baseline, translating to 39.7% higher throughput.[1]

2x	~40%
concurrent jobs	higher throughput

INTRODUCTION

Signoff is one of the most computationally demanding steps in EDA workflows, requiring precise modeling of timing paths across billions of transistors and interconnects. These analyses involve massive, highly interconnected data structures that strain system memory, often pushing DRAM capacity and bandwidth to their limits. As design complexity grows with each process node, the sheer volume of timing data leads to frequent paging, reduced throughput, and underutilized compute resources – creating a significant bottleneck in design turnaround time and overall engineering efficiency. Optimizing memory usage without compromising accuracy has therefore become a critical challenge, directly impacting schedule predictability, tool utilization, and total compute cost.

HARDWARE CONFIGURATION

- **Cloud platform:** Microsoft Azure HX-series VM running on Microsoft Hyper-V
- **Processors:** 2× AMD EPYC™ 9V33X 96-Core processors (Genoa-X, Azure-custom)
- **CPU resources:** Azure exposes 176 total cores out of 192 total cores, with SMT disabled; 174 schedulable CPUs and 2 CPUs reserved for MEXT
- **Physical DRAM:** ~1.4 TB DRAM (1,377 GiB visible to the OS)
- **Local scratch storage:** 1.92 TB Microsoft NVMe Direct Disk used for /tmp
- **MEXT storage/memory pool:** Separate 1.92 TB Microsoft NVMe Direct Disk, claimed exclusively by MEXT and presenting 1,191.06 GiB (~1.2 TB) as the MEXT Memory™ pool

SOFTWARE STACK

- **Application:** Synopsys PrimeTime 2025.06-SP2
- **Operating system:** Red Hat Enterprise Linux 8.10 (Ootpa)
- **Kernel:** Linux 6.17.13-mextpcihv
- **MEXT software:** MEXT Predictive Memory™ Version 1.2

AMD MEXT PREDICTIVE MEMORY SOFTWARE

How It Works

The MEXT software-only solution makes flash storage appear as DRAM-like memory to the OS.

Here's how it works:

- MEXT continuously monitors which memory pages in DRAM are actively being utilized (hot) and which have gone cold
- MEXT offloads the cold memory from DRAM to flash
- MEXT AI engine mitigates the effects of flash latency to keep the system performant
  - AI engine continually predicts which offloaded pages might be requested by the application soon (in other words, which pages are likely to soon go from cold to hot)
  - AI engine transparently moves those pages back into DRAM before the requests are even made
  - As a result, the application stays performant because from its perspective, the relevant memory pages are always already resident in DRAM

THE MARKET: DRAM CHALLENGES

Server memory (DRAM) comes with some major challenges.

First, it is now the most expensive data center component, with cost-per-bit increasing dramatically and showing no signs of slowing.[2]

Second, predicting how much is needed for a particular job is very difficult, and getting it wrong can result in out-of-memory (OOM) errors and job failures; as a result, most teams over-provision to avoid the "memory cliff".

Third, for larger memory systems, the granularity of available memory sizes is quite coarse; even if the exact requirement is known, it's impossible to fine-tune the purchased memory to the requirement plus a small buffer (e.g., 10%).

Lastly, even if the precisely correct memory size is somehow implemented, utilization is often quite poor.

In short, DRAM is extremely expensive, often over-provisioned, hard to fine-tune, and usually poorly utilized.

MEXT Value

MEXT addresses the major DRAM challenges outlined above.

First, by leveraging low-cost flash as memory, MEXT enables much more cost-effective computing.

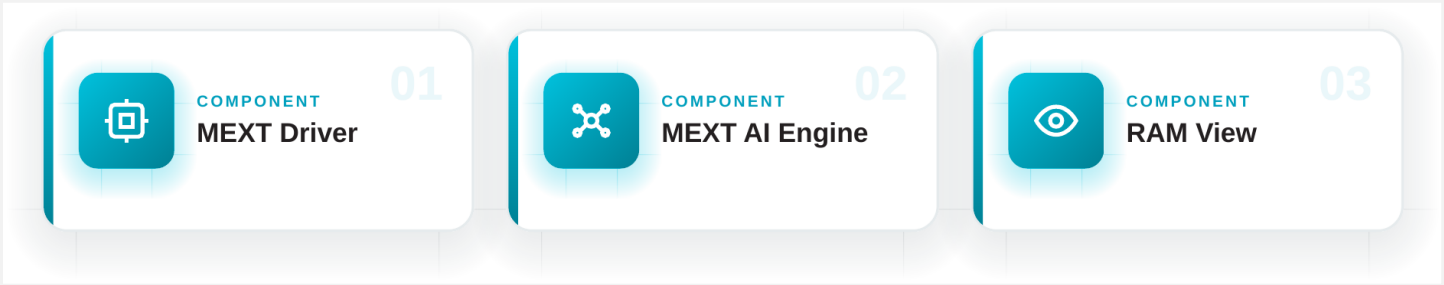
Second, MEXT can prevent OOM crashes by enabling applications to go out to flash-as-memory if DRAM ever runs out.

Third, MEXT enables more precise right-sizing of memory sizes to true requirements.

Lastly, MEXT solves the utilization issue by intelligently tiering across DRAM and flash to keep DRAM mostly all hot.

AMD MEXT SOLUTION COMPONENTS

The solution consists of 3 primary components: the MEXT Driver, the MEXT AI Engine, and the RAM View™ Observability Platform.



Driver

The MEXT Driver is a dynamically loadable kernel module (which does not alter the standard Linux kernel) that sends process and memory page telemetry data to the AI engine.

MEXT AI Engine

The AI engine is a user-space process that feeds predictions of which memory pages should be pushed from flash to DRAM – making predictions / inferences in under a fraction of a second. It runs entirely on the local Linux operating system (on a single CPU core) and does not require a GPU.

It was inspired by modern AI techniques based on neural networks. Instead of using these techniques to predict words or natural language patterns (like ChatGPT does), it predicts sequences of future memory page accesses. It consists of a family of models that work together, combining extremely lightweight heuristic predictors with more powerful neural-network models. For any given workload, it automatically adjusts to use the model or group of models that performs best. Continuous observation of which predicted pages were actually used by the application also enables the engine to acquire real-time feedback regarding model accuracy, supporting ongoing adaptation and self-optimization.

RAM View Observability Platform

RAM View is an observability / visualization tool that illustrates how much memory the application is consuming including real-time insights into page hotness / warmness / coldness. It illustrates the proportion of memory pages that were accessed <15s ago, <1min ago, <2min ago, <3min ago, <5min ago, and >5min ago.

METHODOLOGY

Testing used Synopsys PrimeTime 2025.06-SP2 on the same Microsoft Azure HX-series VM with two AMD EPYC 9V33X processors. The 1.4TB DRAM-only baseline ran two jobs concurrently with 87 threads each, using all 174 cores. The MEXT configuration added a ~1.2 TB MEXT memory pool and ran four jobs concurrently with 43 threads each, using all 174 cores. Per-job wall-clock runtime was recorded, then throughput was calculated as concurrent jobs divided by average runtime per job, multiplied by 3,600 seconds per hour. The baseline was run three times and the MEXT scenario five times. All jobs completed with zero errors and every run passed. Because concurrency changed, this study evaluates jobs completed per hour rather than comparing individual job runtime.

RESULTS

Test Configuration	Throughput (Jobs/Hour)
1.4TB DRAM-only baseline, 2 concurrent jobs	0.4314
1.4TB DRAM + ~1.2TB MEXT, 4 concurrent jobs	0.6029
Throughput improvement with MEXT	39.7%

Capacity and Concurrency

Each PrimeTime job used approximately 600GB of peak resident memory. Four concurrent jobs therefore required approximately 2.4TB of memory, which exceeded the VM's 1.4TB of DRAM. Adding a ~1.2 TB MEXT memory pool enabled the higher concurrency level on the same VM.[1]

Performance

The DRAM-only baseline averaged 16,689 seconds per job while running two jobs concurrently, producing 0.4314 jobs per hour. The MEXT configuration averaged 23,886 seconds per job while running four jobs concurrently, producing 0.6029 jobs per hour. Although average time per job increased because each job shared the VM with three other jobs instead of one, total throughput increased by 39.7%.[1]

CONCLUSION

For memory-intensive signoff workloads, AMD MEXT software can turn available compute into greater aggregate output by expanding effective memory capacity. In this study, the added MEXT Memory enabled twice as many concurrent PrimeTime jobs on the same Azure HX-series VM and delivered approximately 40% more throughput. The result demonstrates how with MEXT, customers can keep existing instances in place, add effective memory capacity in matter of minutes, and use that capacity to increase the amount of work completed over time.

DISCLAIMER

©2026 Advanced Micro Devices, Inc. All rights reserved. AMD, the AMD Arrow logo, AMD EPYC, and combinations thereof are trademarks of Advanced Micro Devices, Inc. Synopsys and PrimeTime are trademarks or registered trademarks of Synopsys, Inc. PCIe is a registered trademark of PCI-SIG Corporation. Other product names used in this publication are for identification purposes only and may be trademarks of their respective companies. Supported features may vary by operating system. Please confirm with the system manufacturer for specific features. No technology or product can be completely secure.

ENDNOTES

1. 9xx5-301: AMD internal testing from 9/25/2026 to 10/4/2026 using Synopsys PrimeTime 2025.06-SP2, an EDA signoff workload, on the same Microsoft Azure HX-series VM with two AMD EPYC 9V33X processors. The 1.4TB DRAM-only baseline ran two jobs concurrently and averaged 16,689 seconds per job, or 0.4314 jobs/hour, across three runs. The MEXT configuration used 1.4TB DRAM plus a ~1.2TB MEXT memory pool, ran four jobs concurrently, and averaged 23,886 seconds per job, or 0.6029 jobs/hour, across five runs. Throughput improvement was calculated as 0.6029 / 0.4314 = 1.397, or 39.7% higher throughput. All jobs completed with zero errors and all runs passed.

2. CounterPoint Research: https://counterpointresearch.com/en/reports/memory-price-tracker-feb-2026