# NVIDIA Vera CPU Architecture

## Max Single-Threaded CPU at Scale for Agents

*Version 1.1*

# Table of Contents

# List of Figures



# List of Tables

# NVIDIA Vera CPU Architecture

## Introduction

AI is rapidly evolving from systems that generate responses to systems that take action. Modern reasoning models increasingly interact with external environments through tool calls, code execution, data processing, search, and software workflows. At the same time, reinforcement learning (RL) has emerged as a key scaling mechanism for improving model capability, requiring vast numbers of environment evaluations, reward calculations, and feedback loops. As a result, CPUs have become a critically important part of the AI execution path, serving as the platform that executes, evaluates, and orchestrates agent actions while keeping GPUs continuously supplied with context.

These emerging workloads represent a new CPU design point. RL and agentic AI systems are not traditional batch workloads; they execute continuous loops of reasoning, acting, observing, and evaluating across thousands of concurrent environments. They are characterized by irregular control flow, short feedback loops, real-time analytics, and massive concurrency, placing simultaneous pressure on single-thread performance, memory bandwidth, latency, and scaling efficiency. Meeting these requirements requires a new kind of CPU architecture purpose-built for agents-removing CPU bottlenecks to increase AI factory output and efficiency.

NVIDIA Vera CPU was designed specifically to address this need, combining the NVIDIA designed Olympus core architecture with high-bandwidth power efficient LPDDR5X memory, the NVIDIA Scalable Coherency Fabric (SCF), NVLink-C2C, and PCIe 6.4 to deliver a platform optimized for agentic inference, reinforcement learning, and next-generation AI factories.

This whitepaper provides an architectural overview of NVIDIA Vera CPU and the technologies that enable its performance, efficiency, and scalability. The following sections describe the Olympus core microarchitecture, memory and IO subsystem, platform scalability, performance and impact to the AI factory.

# NVIDIA Vera CPU

At its heart is the Olympus core architecture, featuring 88 high-performance cores and 176 NVIDIA Spatial Multithreading threads optimized for the frontend-heavy, latency-sensitive, and highly concurrent execution patterns common in agentic AI and reinforcement learning environments. The processor integrates wide, high-IPC microarchitecture designed to maximize single-thread performance (Table 1), efficiently scaling across thousands of concurrent software environments.

| Olympus Feature | Single Thread Specifications |
|---|---|
| **Branch Prediction** | Neural & Special-Purpose Predictors<br>2 taken branches per cycle |
| **Instruction Decode** | 10 Wide |
| **Prefetcher** | Novel Graph and advanced prefetchers |
| **L1D Cache** | 96KB |
| **L1I Cache** | 64KB |
| **L2 Cache** | 2MB |
| **L3 Cache** | 164MB |
| **Memory BW per Core** | 14GB/s |

Table 1. Olympus Core Specification

To meet the growing memory demands of AI infrastructure, Vera integrates eight LPDDR5X memory controllers delivering up to 1.2 TB/s of memory bandwidth. By combining high bandwidth with the superior power efficiency of LPDDR5X memory with datacenter class ECC(Error-Correcting Code), Vera significantly reduces platform-level power consumption compared to traditional DDR-based server architectures. This combination of compute density, memory bandwidth, and efficiency enables higher system utilization while reducing power and cooling requirements across large-scale deployments.

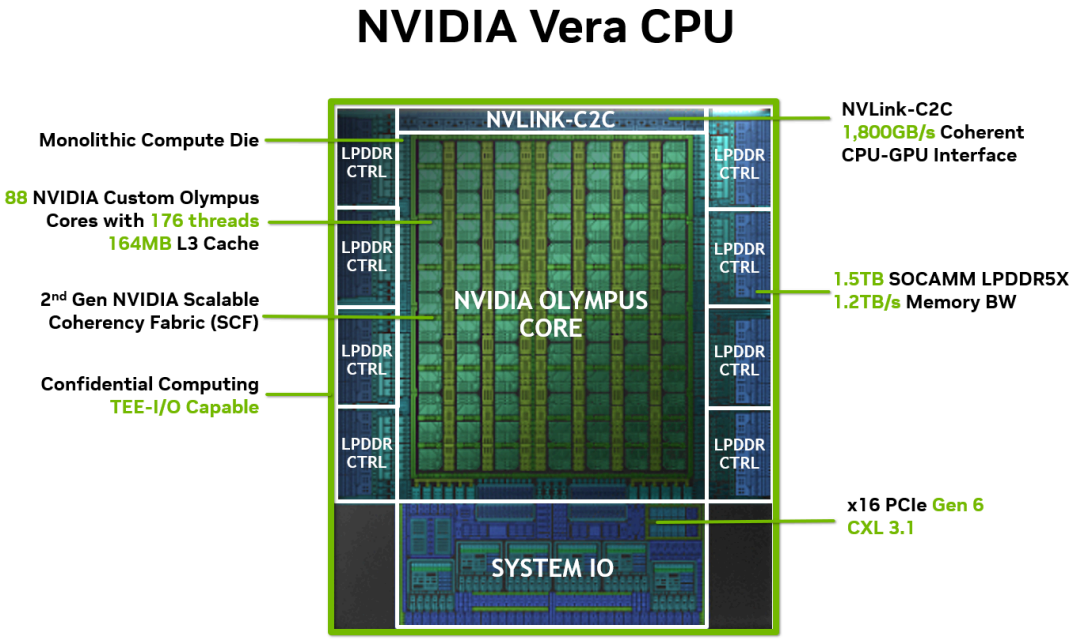


Figure 1. NVIDIA Vera CPU Architecture Block Diagram

Vera is tightly integrated into the NVIDIA AI platform through second-generation NVLink-C2C, providing up to 1.8 TB/s of coherent CPU-GPU bandwidth. The processor also includes a high-performance system I/O subsystem with PCIe 6.4 and CXL 3.1 support, enabling connectivity to accelerators, networking, and storage infrastructure. Together with NVIDIA Scalable Coherency Fabric (SCF), Vera provides a foundation for scale-up and scale-out AI systems that can efficiently orchestrate and feed the next generation of accelerated computing platforms.

Vera builds on the foundation established by Grace's SCF and extends it with a new NVIDIA-designed CPU microarchitecture, higher memory bandwidth, increased CPU-GPU connectivity, Confidential Computing, and next-generation I/O. It also marks the first time an NVIDIA-designed CPU core and NVIDIA coherent fabric come together in a single CPU platform. These enhancements are designed to improve both single-thread responsiveness and overall system throughput for modern AI infrastructure. Table 2 provides a comparison of the key architectural specifications.

| Feature | Vera CPU |
|---|---|
| **Cores** | 88 NVIDIA Custom Olympus cores |
| **Threads** | 176 Spatial Multithreading |
| **Unified L3 Cache** | 164MB |
| **Memory bandwidth (BW)** | Up to 1.2TB/s |
| **Memory capacity** | Up to 1.5TB LPDDR5X |
| **SIMD** | 6x 128b SVE2 \| FP8 |
| **NVLINK-C2C** | 1,800GB/s |
| **PCIe/CXL** | Gen6.4/CXL 3.1<br>176 lanes (2S Vera)<br>96 lanes Vera Rubin |
| **Confidential Computing** | Supported |
| **Power** | 250-450W<br>Configurable |
| **ISA Support** | Armv9.2 |

Table 2. Vera CPU Specification

# Olympus Core

Olympus was developed through extreme co-design across the Vera Rubin platform, spanning CPUs, GPUs, networking, storage, memory systems, and software. This system-level approach enabled NVIDIA to optimize the AI factory as a whole, not just the CPU in isolation. As agentic AI and reinforcement learning place more pressure on CPU-side execution, Olympus was designed to accelerate the irregular, branch-heavy, and latency-sensitive software paths that increasingly shape end-to-end AI factory performance.

The Olympus core is the compute engine of the NVIDIA Vera CPU, built from the ground up to maximize instructions per cycle (IPC) for irregular, branch-heavy, latency-sensitive, and highly concurrent software. Designed specifically for the demands of modern AI infrastructure, Olympus combines high single-thread performance, wide instruction throughput, deep out-of-order execution, and advanced memory acceleration technologies to efficiently execute agentic AI, reinforcement learning, data analytics, and large-scale software workloads. Together, this ground up design enables Olympus to deliver 1.5x higher IPC gain vs prior generation.

At a high level, the Olympus core consists of the following architectural components:

- **Front End: Optimized Branch Predictors**
- **Mid Core: Critical Path Acceleration**
- **Execution Engine: Complex Vector Optimization**
- **Cache Subsystem: Latency Optimized Instruction and Data Paths**

The following sections describe each of these architectural components in greater detail and explain how they contribute to the performance, efficiency, and scalability of the Olympus core.

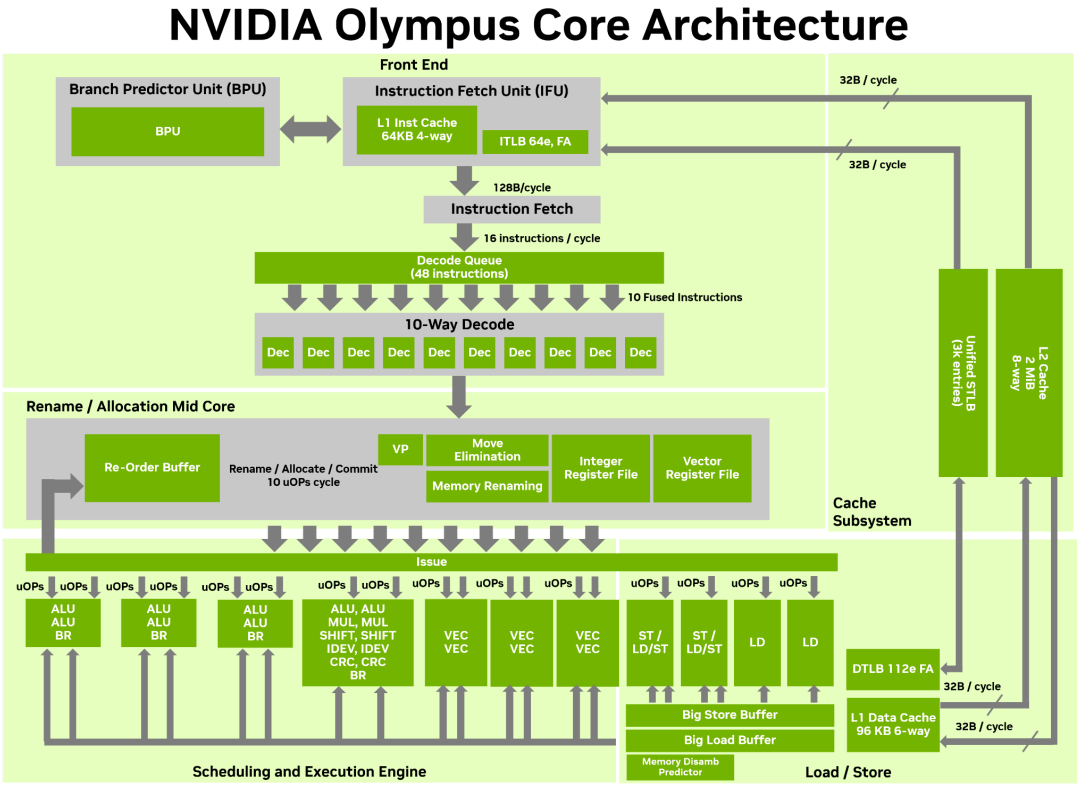


Figure 2. NVIDIA Olympus Core Architecture

# Front End – Optimized Branch Predictors

The Olympus Front End is designed to sustain high instruction throughput across deep software stacks such as PyTorch while minimizing pipeline bubbles caused by branch mispredictions and instruction-cache misses. Agentic AI environments, reinforcement learning frameworks, interpreters, graph analytics, and modern data-processing runtimes frequently execute irregular control flow with large instruction footprints. To address these challenges, Olympus combines a multi-level branch prediction subsystem, large branch target buffers, advanced instruction fetch logic, and a 10-wide decode engine.

At the heart of the Front End is a sophisticated branch prediction architecture that combines a novel neural branch predictor and other special-purpose predictors with robust ahead pipelining mechanisms

### Neural Branch Predictor

For branches that are difficult to predict using conventional history-based techniques, Olympus incorporates the neural branch predictor. It's designed to improve prediction accuracy on complex, branch-heavy software. It learns longer-range control-flow behavior and statistically biased branch patterns that are common in large runtime environments, interpreters, databases, compilers, and agent workloads. By improving prediction accuracy on difficult branches, the neural predictor reduces misprediction penalties and helps keep the wide Olympus Front End supplied with useful instructions. This enables higher instruction throughput and more consistent performance on workloads with irregular control flow and large instruction footprints.

Together, Olympus' advanced branch prediction and wide front-end architecture enable the core to sustain up to two taken-branches per cycle with zero penalties, thereby reducing front-end stalls Olympus also introduces significantly larger branch target structures and instruction delivery resources. Instructions are delivered into a 10-wide decode engine capable of decoding up to ten instructions per cycle. Combined with large instruction caches, instruction TLBs, and wide fetch bandwidth, the Front End is designed to continuously feed the deep execution engine behind it.

# Mid Core – Critical Path Acceleration

The Olympus Mid Core is responsible for transforming decoded instructions into executable operations while maximizing instruction-level and memory-level parallelism. Modern agentic workloads—including Python runtimes, reinforcement learning

environments, data-processing frameworks, simulation engines, and tool orchestration systems—often generate long dependency chains that limit performance. These workloads frequently spend cycles waiting on pointer dereferences, memory accesses, object traversal, and sequential software dependencies. The Olympus Mid Core is designed to identify and execute independent work hidden behind these bottlenecks, enabling higher utilization of execution resources and improved overall IPC.

At the heart of the Mid Core is a wide rename and allocation engine supported by a large reorder buffer and extensive physical register resources. Register renaming eliminates false dependencies between instructions, allowing the processor to expose additional parallelism and execute instructions as soon as their true operands become available. A large instruction window enables Olympus to look significantly further ahead in the instruction stream, finding useful work while other instructions wait on memory or branch resolution.

To further improve performance, Olympus incorporates several advanced dependency-breaking technologies including memory renaming, value prediction and other critical-path acceleration techniques.

- **Memory Renaming**: Accelerates store-to-load dependency chains by allowing dependent instructions to execute before a load completes when the data relationship can be predicted or determined. This is particularly beneficial for pointer-heavy software, graph traversal, runtime frameworks, and complex object-oriented workloads.
- **Value Prediction**:  Identifies stable dependency chains and predicts future values before they are produced, allowing dependent instructions to execute speculatively while correctness is verified later. This capability is especially effective for repetitive software patterns and sequential data structures.
- **Move elimination and value optimization:** Techniques to achieve higher utilization.

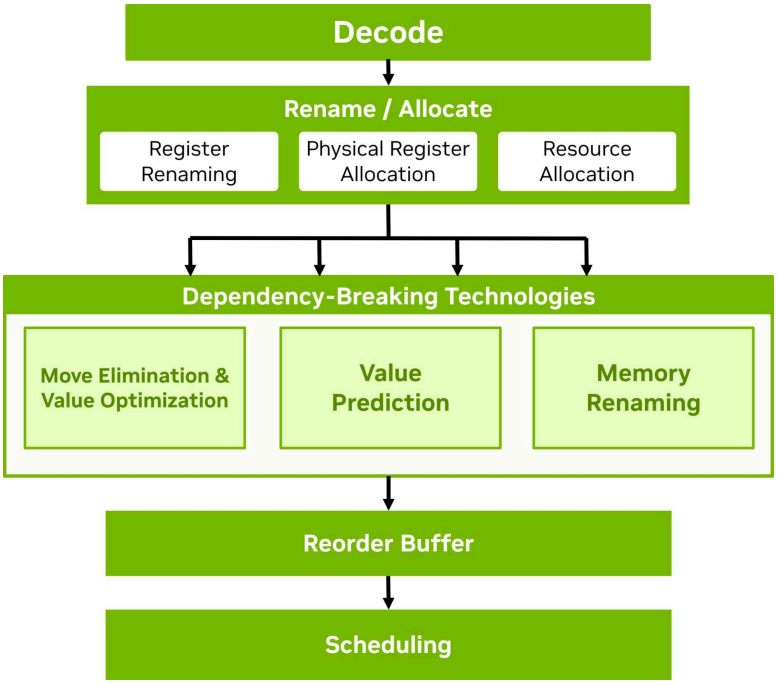


Figure 3. Olympus Mid Core

Together, these features enable Olympus to sustain high performance on modern software workloads that are difficult to optimize through conventional execution techniques alone. By reducing dependency-related stalls and exposing additional parallelism, the Mid Core improves responsiveness of agentic tool calls, reinforcement learning environments, interpreters, and other latency-sensitive software infrastructure that forms the foundation of modern AI factories.

## Execution Engine – Complex Vector Optimization

The Olympus execution engine is designed to sustain high instruction throughput across a diverse range of workloads, from agentic AI runtimes and reinforcement learning environments to databases, graph analytics, scientific computing, and large-scale software frameworks. After instructions pass through the rename and scheduling stages, they are dynamically dispatched to specialized execution resources optimized for integer arithmetic, branch processing, vector computation, and memory operations. The combination of a wide execution backend and a deep out-of-order engine enables Olympus to continue making forward progress even when portions of the workload are stalled by memory accesses, branch resolution, or long dependency chains.

At the core of the execution engine are eight simple integer ALUs, two complex ALUs, and four dedicated branch execution units. The simple ALUs handle common arithmetic and logical operations, while the complex ALUs accelerate higher-latency operations

such as multiplication, division, CRC, and shift-intensive workloads. The dedicated branch units work in conjunction with the advanced branch prediction subsystem to rapidly resolve control-flow decisions and minimize pipeline disruption. This combination is particularly beneficial for modern agentic workloads, which often exhibit irregular control flow and frequent branching as agents reason, evaluate state, and execute actions.

For vector and AI-oriented computation, Olympus integrates six 128-bit SVE vector execution units, including two crypto-capable vector pipelines and supports FP8 precision. These units accelerate vectorized workloads commonly found in data processing, machine learning preprocessing, compression, encryption, scientific computing, and analytics frameworks. Support for Arm Scalable Vector Extension (SVE) enables efficient execution of both traditional HPC workloads and emerging AI data-processing pipelines.

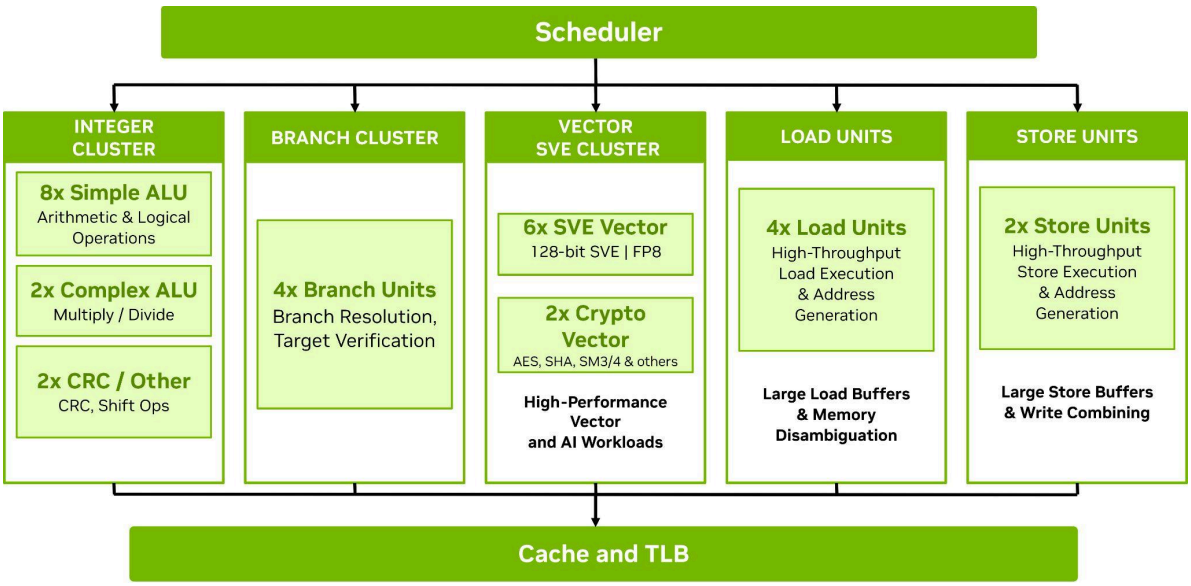


Figure 4. Olympus Execution Engine

The execution engine is tightly coupled to a high-bandwidth memory subsystem through four load pipelines and two store pipelines, enabling substantial memory-level parallelism. Large load-store structures, memory disambiguation prediction, and advanced prefetching technologies help sustain execution throughput even when operating on large datasets or irregular memory-access patterns. Together, these resources enable Olympus to maintain high utilization across both compute-intensive and memory-intensive workloads.

By combining wide integer execution resources, dedicated branch processing, high-throughput SVE vector engines, and substantial memory parallelism, the Olympus execution engine is optimized to efficiently execute the diverse software stacks that underpin modern AI factories.

## Cache Subsystem – Latency Optimized Instruction and Data Paths

The Olympus cache and TLB subsystem is designed to keep the execution engine supplied with instructions and data while minimizing the impact of cache latency. Modern AI infrastructure workloads frequently operate on large datasets that exceed cache capacity and traverse complex software structures with irregular access patterns. Agent runtimes, graph analytics, recommendation systems, sparse matrices, reinforcement learning environments, and large-scale data processing frameworks all place significant pressure on the memory hierarchy. To address these challenges, Olympus combines a deep cache hierarchy, extensive TLB resources, intelligent memory disambiguation, and a collection of advanced hardware prefetch engines.

At the first level of the hierarchy, Olympus integrates a 96 KB L1 data cache and a 64 KB L1 instruction cache, both optimized for low-latency access. The core further incorporates a fully-associative data TLB, instruction TLB, and a large unified second-level TLB. These structures significantly reduce address translation overheads for large software frameworks, containerized workloads, databases, and AI infrastructure services that routinely operate across large virtual memory footprints.

A key innovation within the Olympus memory hierarchy is its advanced collection of hardware prefetch engines. Traditional prefetchers are highly effective at detecting sequential or regular access patterns but often struggle with the irregular memory accesses common in graph processing and sparse data structures. Olympus addresses this challenge through several specialized prefetchers, including stream, indirect, correlated-miss, spatial, and graph-oriented prefetch engines. Among these, the graph prefetcher represents one of the most significant architectural enhancements.

### Graph Prefetcher

Agentic workloads increasingly depend on graph-like data structures. Agents traverse retrieval indexes, tool graphs, code dependencies, and environment objects where each step often points to the next piece of data. These access patterns are irregular and difficult for conventional prefetchers, since the next address is frequently unknown until a prior memory lookup completes. Olympus includes a graph prefetcher to accelerate exactly this behavior, reducing stalls in the pointer-heavy and indirection-heavy memory patterns common in agent execution. The Olympus graph prefetcher is specifically designed to accelerate array indirection. These workloads frequently access memory using a two-stage pattern where the final address depends on data returned from an earlier memory access.

A common example is:

```
result += A[B[i]];
```

In this pattern:

- B[i] represents an index lookup (Producer)
- A[B[i]] represents the final data access (Consumer)

Traditional prefetchers cannot predict the consumer address until the producer value has been returned from memory. As a result, the processor must often wait for the first access to complete before issuing the second.

The Olympus graph prefetcher learns this producer-consumer relationship in hardware. Once the pattern has been observed, the prefetcher monitors future accesses to the producer stream and automatically generates consumer prefetches before the demand access occurs.

## Spatial Multithreading

Multithreading has gained more significance as many agentic and sandboxed workloads are not purely compute loops. A sandbox often runs a primary execution thread while also handling management activity such as async I/O, timers, runtime services, interrupts, logging, networking, and environment orchestration. A second hardware thread gives the system a place to run this supporting work without moving it to another physical core or forcing the primary sandbox thread to yield more often. This can improve core utilization and responsiveness when agent environments frequently block, wake, or interact with external tools.

Traditional SMT achieves this by allowing both hardware threads to opportunistically share the same core resources. In densely threaded sandbox environments, that sharing can create contention across branch prediction, decode, execution, load/store, cache, and memory resources. This creates a “noisy neighbor” effect, where activity on one thread can reduce the performance of the other thread sharing the same core within a sandbox. For agentic AI, reinforcement learning, and cloud services—where many short-running tasks execute concurrently and tail latency matters—this can increase variability and make performance less predictable.

NVIDIA Vera takes a different approach with Spatial Multithreading. Instead of relying primarily on opportunistic time-sharing inside a narrower core, Vera’s wide Olympus core is designed so resources can be partitioned across two hardware threads as shown in Figure 5. This allows Vera to operate as a high-throughput single-threaded core when maximum per-thread performance is needed while the twin thread handles management, or as two more isolated execution contexts when higher thread density is

required. With 88 physical cores and 176 hardware threads, Vera provides the flexibility to support both high single-thread performance and dense vCPU-style deployments.

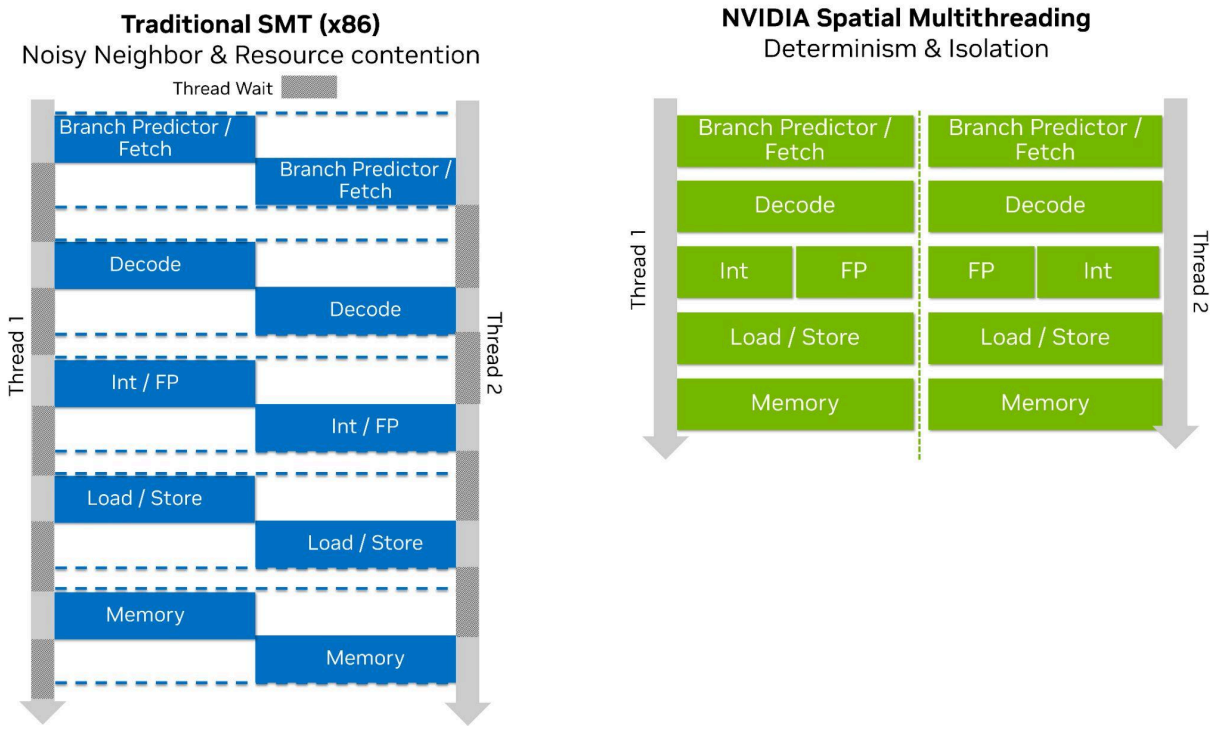


Figure 5. NVIDIA Spatial Multithreading built for agents

This design is especially valuable for agentic AI, RL environments, cloud services, and data-processing pipelines that need predictable performance under heavy concurrency. By reducing resource interference between threads, Spatial Multithreading improves determinism, isolation, and quality of service compared to traditional SMT approaches. The result is a CPU architecture that can run large numbers of concurrent agent tasks while maintaining more consistent latency and throughput.

# Fabric & Memory Subsystem

As AI factories continue to scale, the performance of the CPU is increasingly determined not only by core microarchitecture, but also by the efficiency of the fabric and memory subsystem that connects compute, memory, accelerators, and I/O resources. Agentic AI, reinforcement learning, large-scale data processing, and modern inference pipelines routinely operate on datasets that exceed the capacity of local caches and require efficient movement of data across CPUs, memory controllers, GPUs, and networking infrastructure. These workloads demand a platform architecture capable of delivering

high bandwidth, low latency, deterministic performance and efficient scalability while maintaining a coherent programming model.

To address these requirements, Vera integrates the 2nd generation NVIDIA Scalable Coherency Fabric (SCF), high-bandwidth SOCAMM2 LPDDR5X memory, and second-generation NVLink-C2C connectivity into a unified platform architecture. Together, these technologies enable efficient communication between Olympus cores, memory controllers, system-level caches, accelerators, and I/O resources while providing the bandwidth and capacity required by modern AI workloads.

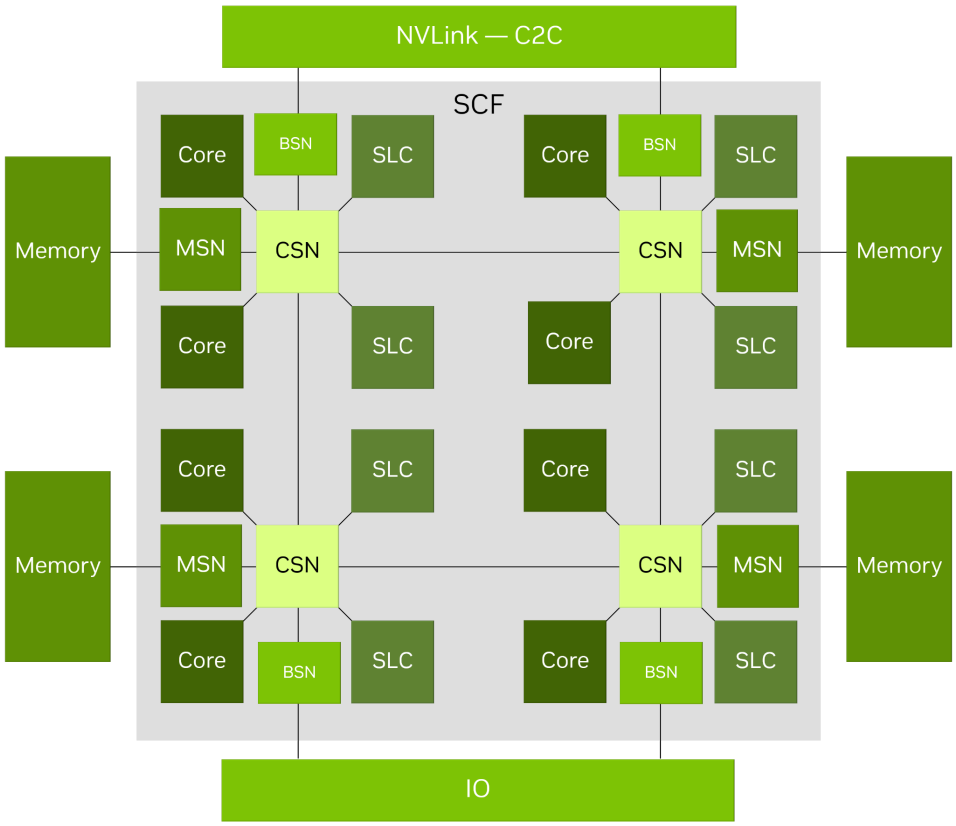


Figure 6. Vera fabric and memory subsystem

## NVIDIA Scalable Coherency Fabric

The NVIDIA 2nd generation Scalable Coherency Fabric (SCF) with 3.4TB/s bisectional bandwidth serves as the communication backbone of the Vera processor. The simplified view in Figure 6 shows a slice of Vera's SCF fabric that scales across the full 88-core CPU. SCF provides a low-latency, high-bandwidth interconnect between Olympus cores, shared cache resources, memory controllers, I/O subsystems, and NVLink-C2C interfaces. The fabric is responsible for maintaining memory coherency across all processor resources while enabling scalable access to memory and accelerators.

SCF is organized as a distributed coherent interconnect connecting multiple cores through a collection of Coherency Switch Nodes (CSNs). Each CSN connects up to two cores and serves as a routing point within the fabric, enabling efficient movement of requests between cores, caches, memory controllers, and external interfaces. The distributed architecture allows memory traffic to scale efficiently as core count and memory bandwidth increase.

Unlike traditional server architectures that rely heavily on centralized coherence structures, SCF distributes coherency management across the fabric. This approach reduces contention, improves scalability, and minimizes latency for cache-coherent memory accesses. The fabric enables all Olympus cores to access local and remote socket through a unified coherent address space while maintaining efficient cache coherency across the processor.

The SCF integrates a 164 MB distributed System-Level Cache (SLC) that serves as a unified L3 cache shared across all Olympus cores. Unlike many chiplet-based CPU architectures where portions of the last-level cache reside on separate compute dies and remote cache accesses must traverse inter-die links, Vera's monolithic compute die and distributed SLC enable uniformly low-latency cache access across the processor. This organization reduces cache access variability, improves cross-core data sharing, and allows frequently accessed data to remain on-chip, reducing DRAM traffic while maintaining high effective memory bandwidth. The distributed cache architecture also provides an efficient balance between latency, bandwidth, silicon area, and scalability, making it particularly well suited for branch-heavy, data-sharing workloads commonly found in agentic AI, graph analytics, and data processing applications.

The architecture additionally provides direct connectivity to the memory subsystem through Memory Switch Nodes (MSNs). These interfaces distribute memory traffic across multiple memory controllers while balancing bandwidth utilization throughout the system. The result is a fabric capable of efficiently feeding both CPU workloads and accelerator-attached workloads that require sustained high-bandwidth memory access.

SCF also acts as the integration point for NVIDIA's broader accelerated computing platform. Through dedicated Bridge Switch Nodes (BSN) interfaces, the fabric connects directly to second-generation NVLink-C2C interfaces, allowing coherent communication between Vera CPUs and NVIDIA GPUs. This enables CPUs and GPUs to share memory efficiently while significantly reducing software overhead associated with traditional PCIe-based accelerator communication.

Key benefits of the NVIDIA Scalable Coherency Fabric include:

- Low-latency communication between CPU clusters
- Efficient access to shared system-level cache resources
- High-bandwidth connectivity to memory controllers

- Native integration with NVLink-C2C and accelerator infrastructure

Vera supports MPAM (Memory System Resource Partitioning and Monitoring), allowing software to partition System-Level Cache (SLC) capacity and memory bandwidth across co-located workloads for quality of service and noisy-neighbor isolation. Integrated cache and memory-bandwidth usage monitors provide the telemetry needed to tune allocations and sustain more consistent, predictable performance in consolidated, multi-tenant deployments.

# SOCAMM2 LPDDR5X

Modern AI workloads place unprecedented demands on memory bandwidth. Agentic execution environments, RL simulations, data processing, graph analytics, and large-scale inference pipelines frequently operate on working sets that extend far beyond processor cache capacity. In many of these workloads, memory bandwidth becomes the limiting factor for overall system performance.

To address these challenges, NVIDIA Vera adopts a memory architecture built around SOCAMM2 (Small Outline Compression Attached Memory Module) and LPDDR5X memory. SOCAMM2 combines the bandwidth and power efficiency advantages of LPDDR5X with the modularity and serviceability required for enterprise server deployments. The architecture delivers up to 1.2 TB/s of memory bandwidth.

SOCAMM2 LPDDR5X was designed around three primary objectives:

- **High Bandwidth** – Deliver the memory bandwidth required to sustain modern AI, analytics, and HPC workloads.
- **Power Efficiency** – Maximize bandwidth per watt using LPDDR5X technology to reduce overall platform power.
- **Enterprise Serviceability** – Provide a modular, field-replaceable memory architecture suitable for hyperscale and enterprise deployments.

### High Bandwidth

Traditional server memory architectures rely on large numbers of DDR DIMMs connected through long electrical traces and complex motherboard routing. While effective for general-purpose computing, these architectures often limit the amount of memory bandwidth available to each core as core counts continue to increase.

SOCAMM2 adopts a different approach by placing LPDDR5X memory devices on compact memory modules located close to the processor package. The shorter electrical paths improve signal integrity, allowing LPDDR5X to operate at data rates of up to 9600 MT/s while delivering as much as 1.2 TB/s of aggregate memory bandwidth. With 88

Olympus cores, Vera provides up to 14 GB/s of memory bandwidth per core, more than three times that of many conventional DDR-based server processors. This high bandwidth-per-core ratio ensures each core remains well supplied with data, reducing memory bottlenecks for bandwidth-intensive workloads such as agentic AI, graph analytics, reinforcement learning, data processing, and scientific computing. Eight SOCAMM2 modules provide a scalable memory subsystem supporting capacities from 256 GB to 1.5 TB, enabling both exceptional bandwidth and large memory footprints for AI factory deployments.

### Power Efficiency

As memory bandwidth requirements continue to increase, memory subsystem power has become a significant contributor to total server power consumption. Traditional DDR5 server platforms require large numbers of DIMMs, high-speed memory interfaces, and substantial power delivery infrastructure to achieve higher bandwidths. Emerging MRDIMM platforms further increase bandwidth but do so at the expense of significantly higher memory power. In many high-capacity server configurations, the memory subsystem alone can consume well over 100 W, with fully populated MRDIMM systems exceeding 200 W depending on channel count, configuration, and bandwidth heavy workloads.

SOCAMM2 leverages LPDDR5X to fundamentally change this tradeoff. By integrating high-speed LPDDR5X memory on compact, processor-attached modules, Vera delivers up to 1.2 TB/s of memory bandwidth while consuming approximately 30W - 40W for a fully populated memory subsystem, depending on capacity. This represents a substantial reduction in memory power compared to conventional DDR5 and MRDIMM-based server architectures while simultaneously delivering significantly higher memory bandwidth.

The efficiency benefits extend beyond the memory subsystem itself. Combined with Vera's configurable 250W to 450W processor TDP, LPDDR5X enables a substantially lower CPU-plus-memory platform power envelope than comparable DDR5-based systems. For AI factories deploying thousands of servers, these savings translate directly into lower cooling requirements, improved rack density, reduced operational costs, and the ability to allocate a larger fraction of the datacenter power budget to GPUs and networking infrastructure. This makes SOCAMM2 LPDDR5X a key architectural enabler for energy-efficient accelerated computing at scale.

### Enterprise Serviceability

A key innovation of SOCAMM2 is bringing the modularity expected in enterprise servers to LPDDR memory. Unlike conventional LPDDR implementations where memory devices are permanently soldered to the motherboard or processor package, SOCAMM2 uses removable memory modules that can be individually installed, upgraded, or replaced.

This modular architecture simplifies manufacturing and system qualification while significantly improving field serviceability. System integrators can deploy different memory capacities using a common processor platform, replace failed modules without replacing the CPU or motherboard, and source qualified modules from multiple memory vendors. At the same time, the compact module design preserves the short electrical paths required to achieve LPDDR5X operating speeds.

### Reliability, Availability, and Serviceability (RAS)

Vera further enhances the enterprise readiness of LPDDR5X through a comprehensive set of Reliability, Availability, and Serviceability (RAS) capabilities. These features improve fault detection, error correction, predictive maintenance, and long-term platform reliability, making SOCAMM2 suitable for mission-critical enterprise and AI factory deployments.

| RAS Feature | Description |
|---|---|
| **Multi-symbol ECC Correction** | Corrects multi-symbol memory errors, providing the foundation for enterprise-class memory reliability. |
| **Multi-Bit Correctable Error Reporting** | Reports multi-bit correctable ECC events to firmware and the operating system, enabling early diagnostics and proactive maintenance. |
| **RAS Error Records** | Implements RAS records for consistent error reporting, diagnostics, and operating system integration. |
| **Hard Post-Package Repair (PPR)** | Permanently repairs defective memory locations identified during manufacturing or deployment, improving long-term memory reliability. |
| **Page Offlining** | Offlines memory pages that have experienced uncorrectable errors, reducing failure blast radius and improving system availability. |
| **Channel Sparing** | Redirects memory accesses away from degraded channels or channel resources, improving platform availability and fault tolerance. |

| | |
|---|---|
| **SOCAMM2 Modular Support** | Enables field replacement and servicing of memory modules without replacing the processor or motherboard. |
| **Error Injection Support** | Provides hardware-assisted error injection to validate firmware, operating system recovery flows, and platform RAS mechanisms. |

Table -3 - Vera RAS features

The memory subsystem is tightly integrated into the SCF architecture through multiple memory controllers distributed around the processor. This organization allows memory traffic to be balanced efficiently across available channels while minimizing contention within the fabric. Combined with Olympus' advanced cache hierarchy and hardware prefetch engines, the memory subsystem helps maintain high utilization of execution resources even under bandwidth-intensive workloads.

# Security, IO and Scale Up

A high-performance CPU must not only execute workloads efficiently, but also move data securely throughout the system. Modern AI factories depend on CPUs to orchestrate communication between accelerators, memory, storage, networking, and other processors, making system interconnects as important as core performance, all in a secure environment. NVIDIA Vera is designed with a balanced I/O architecture that provides high-bandwidth, low-latency connectivity for both scale-up and scale-out deployments. Together, second-generation NVLink-C2C, PCIe 6.4, CXL 3.1, and coherent two-socket connectivity with Confidential Computing enables Vera to efficiently integrate into next-generation AI infrastructure.

The Vera I/O and scale-up architecture consists of several key technologies

- **Second-Generation NVLink-C2C** – High-bandwidth coherent interconnect for CPU-to-GPU and CPU-to-CPU communication.
- **Two-Socket Scale-Up** – High-speed coherent processor-to-processor connectivity for scalable dual-socket systems.
- **PCI Express 6.4** – High-performance I/O connectivity for storage, networking, and accelerator devices.
- **Compute Express Link (CXL) 3.1** – Extends the PCIe infrastructure to support memory expansion, pooling, and composable system architectures.
- **Confidential Computing -** Secure scale up environment across 2-socket and NVL rack.

## NVLINK C2C

NVLink-C2C (Chip-to-Chip) is NVIDIA's coherent interconnect technology for tightly coupling CPUs and GPUs within a single system. First introduced with the NVIDIA Grace platform, NVLink-C2C extends the NVLink architecture to provide cache-coherent communication between processors while delivering significantly higher bandwidth and lower latency than conventional PCIe-based interconnects. In Vera, the second-generation implementation provides up to 1.8TB/s of bidirectional coherent bandwidth (900GB/s per direction), enabling CPUs, GPUs, and coherent system components to efficiently share memory and exchange data as a unified computing platform.

NVLink-C2C serves as the primary scale-up interconnect throughout the Vera platform. It provides coherent connectivity between two Vera processors in a dual-socket configuration, allowing both CPUs to operate as a unified system with high socket-to-socket bandwidth. The same interface also connects Vera directly to the Rubin GPU, enabling CPUs and GPUs to efficiently share memory and exchange data without the software overhead associated with traditional I/O interconnects. At rack scale, NVLink-C2C forms the building block of NVIDIA NVL72 systems, where coherent CPU-GPU connectivity combines with the broader NVLink fabric to enable tightly integrated accelerated computing infrastructure spanning 72 GPUs and 36 CPU nodes.

## Dual-Socket Vera

Vera scales from single-socket deployments to high-performance dual-socket systems through second-generation NVLink-C2C, creating a fully coherent two-socket platform optimized for AI factories, HPC, and enterprise computing. The coherent interconnect enables both processors to operate as a unified system with high socket-to-socket bandwidth and low latency, allowing applications to efficiently share data and scale across both sockets.

A dual-socket Vera system combines 176 Olympus cores and 352 hardware threads with NVIDIA Spatial Multithreading enabled. The platform supports up to 3 TB of LPDDR5X memory, delivers as much as 2.4 TB/s of aggregate memory bandwidth, and provides up to 176 PCIe 6.4 lanes for networking, storage, and accelerator connectivity. Together with the NVIDIA Scalable Coherency Fabric and second-generation NVLink-C2C, the dual-socket architecture provides a balanced platform capable of supporting large-scale data analytics, agentic AI, reinforcement learning, and CPU-intensive infrastructure workloads while maintaining high bandwidth, low latency, and efficient processor-to-processor communication.

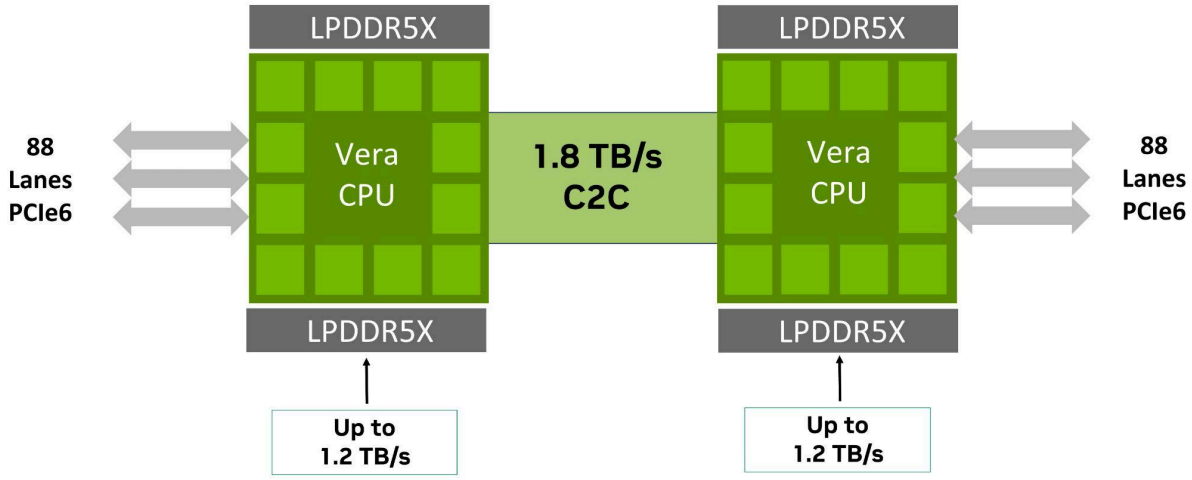


Figure 7. Coherent Dual-Socket NVIDIA Vera

**Simplified NUMA Topology**

As processor core counts continue to increase, memory topology has become an increasingly important aspect of server architecture. Many modern x86 processors achieve higher core counts through chiplet-based designs, partitioning the processor into multiple NUMA (Non-Uniform Memory Access) domains, each with its own local memory and cache resources. This approach can be effective for cloud infrastructure where CPU resources are typically rented by the hour in relatively small vCPU configurations. However, for AI infrastructure, enterprise applications, HPC, and data analytics that routinely utilize large portions of a processor or an entire server, the resulting non-uniform memory access characteristics require operating systems, hypervisors, runtime libraries, and applications to carefully manage thread placement and memory allocation to achieve optimal performance. In large dual-socket systems, this complexity can result in as many as 32 NUMA domains, significantly increasing software tuning and optimization requirements.

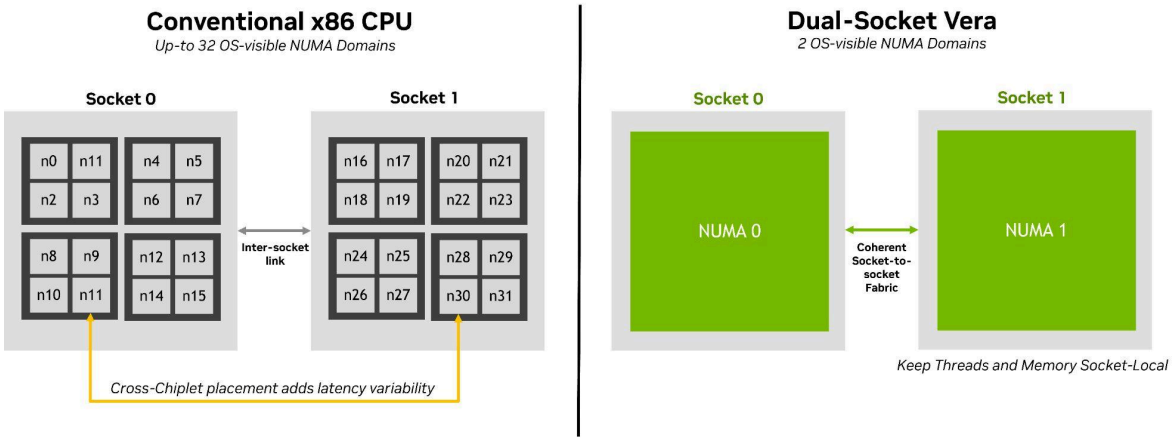


Figure 8. Software-first unified NUMA design with Vera CPU

Vera takes a fundamentally different software-first approach. Each processor presents a single NUMA domain, resulting in a simple two-NUMA-node topology in a dual-socket system. Built around a monolithic compute die, the NVIDIA Scalable Coherency Fabric, and a unified system-level cache, Vera provides uniform access to on-chip compute and memory resources without the latency variability associated with traversing multiple chiplets. This eliminates the chiplet tax often encountered in distributed processor architectures, where accessing data located on remote dies incurs additional latency and bandwidth overhead. The simplified NUMA topology reduces software complexity while improving application portability across deployments.

Another consequence of chiplet-based processor architectures is increased I/O topology complexity. Because PCIe controllers are distributed across multiple chiplets and fabric domains, PCIe devices often need to be carefully assigned to specific processor regions or I/O zones to achieve optimal bandwidth and latency. Improper device placement can increase cross-fabric traffic, and reduce I/O efficiency, particularly for bandwidth-intensive accelerator, networking, storage workloads. Vera's monolithic compute architecture and unified I/O subsystem eliminate these topology constraints, providing a simpler PCIe hierarchy with more predictable performance and reducing the need for platform-specific PCIe zoning and software tuning.

## PCI 6.4

Vera incorporates a next-generation I/O subsystem built around PCI Express 6.4, providing the bandwidth and flexibility required for modern AI factories and enterprise servers. A single-socket Vera processor supports up to 88 PCIe 6.4 lanes, while dual-socket configurations provide up to 176 PCIe 6.4 lanes for networking, storage, accelerators, and high-speed peripherals. Operating at 64 GT/s per lane, PCIe 6.4 doubles

the throughput of PCIe 5.x, ensuring sufficient bandwidth for next-generation SmartNICs, DPUs, NVMe storage, and accelerator devices.

To support a wide variety of deployment configurations, Vera provides flexible lane bifurcation, allowing each x16 interface to be partitioned according to system requirements. This enables platform designers to optimize I/O resources for high-bandwidth networking, dense NVMe storage, CXL devices, or mixed peripheral configurations without additional switching hardware.

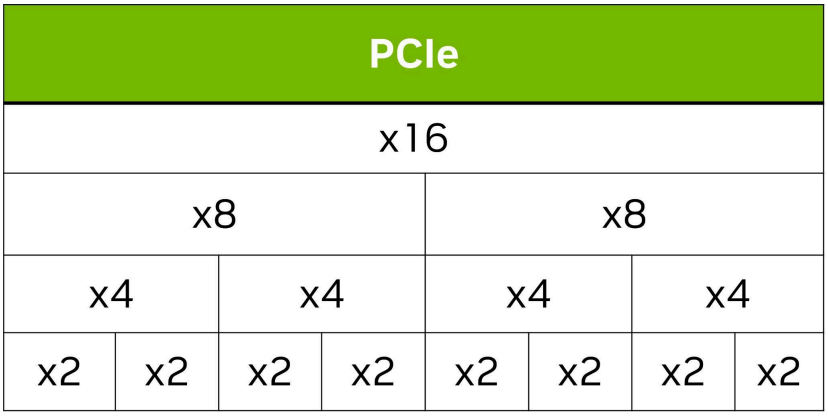


Figure 9. Vera offers PCIe 6.4 with ports bifurcated down to x2

## CXL

NVIDIA Vera extends its PCIe 6.4 infrastructure with support for Compute Express Link (CXL) 3.1, enabling next-generation memory expansion and composable infrastructure. Vera supports CXL Type-3 memory devices, allowing systems to seamlessly expand main memory beyond directly attached LPDDR5X while maintaining cache-coherent access. This capability is particularly valuable for memory-capacity-intensive workloads such as in-memory databases, graph analytics, and datasets that benefit from larger memory footprints.

Beyond memory expansion, Vera implements several key CXL 3.1 capabilities that improve platform flexibility, security, and future scalability. Support for memory pooling enables multiple hosts to efficiently share external memory resources, improving overall memory utilization across a datacenter.

# Confidential Computing

NVIDIA Vera CPU includes Confidential Computing capabilities to protect sensitive data and AI models and help accelerate secure enterprise AI adoption. By addressing critical privacy and security requirements without compromising performance for data in use, Vera provides a trusted foundation for organizations to deploy and scale transformative AI workloads with confidence.

Vera supports Arm CCA, RME-DA, and RME-CDA, providing the foundation for confidential VM isolation and secure device assignment. Per-VM encryption keys enable cryptographic isolation between tenants running on the Vera CPU, while support for TDISP-capable devices extends the trusted execution environment beyond CPU. As the industry's first Confidential Computing CPU to enable TDISP for coherent devices such as GPUs, Vera provides high-performance, coherent access without bounce buffers or unnecessary data copies while maintaining strong workload isolation.

Authenticated chip-to-chip (C2C) encryption protects data across coherent links, allowing Confidential Computing environments to scale across multiple CPU sockets. This enables rack-scale confidential computing solutions optimized for the performance requirements of different model configurations. Vera's initiator-side encryption also provides a path to extend data protection to CXL Type 3 devices.

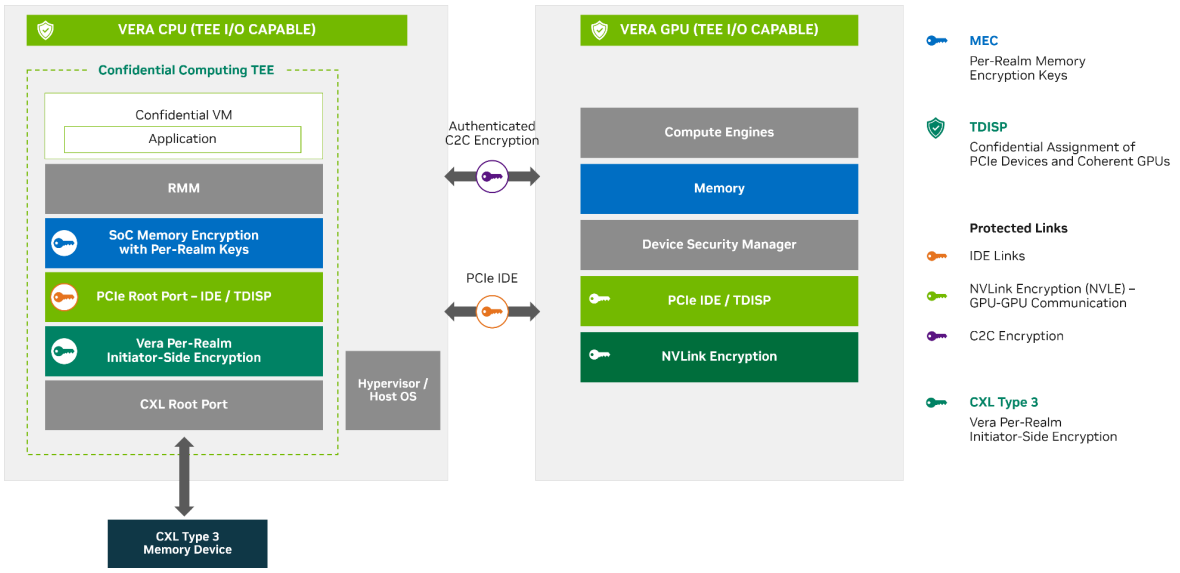


Figure 10. Vera Rubin Confidential Computing Architecture

NVIDIA has advanced AI security over multiple generations. Hopper introduced high-performance Confidential Computing for GPUs. Blackwell expanded these capabilities, eliminating the traditional tradeoff between security and performance. As a

pioneer in secure AI infrastructure, NVIDIA has completed this progression by unifying CPU and GPU security into a single unified trust domain across the entire rack with Vera Rubin NVL72 by delivering an end-to-end Confidential Computing fabric that spans the CPU, the coherent GPU, PCIe devices, CXL memory, and NVLink-connected GPUs to protect AI workloads in the cloud, on premises or at the edge.

# NVIDIA Vera CPU Performance

The preceding sections described the architectural foundation of NVIDIA Vera CPU, the Olympus core, SCF, memory subsystem and IO. This section examines the performance impact of those design choices.

Vera performance is evaluated across four dimensions:

- Memory Bandwidth and Latency – Shows how Vera keeps cores supplied with data through high-bandwidth memory, low-latency access, and a high-throughput coherent fabric.
- Application Workload Performance – Demonstrates performance across agentic benchmarks, data analytics, graph processing and other CPU-intensive workloads.
- Core IPC – Measures the impact of the Olympus microarchitecture, including the wide Front End , advanced branch prediction, deep out-of-order execution, and expanded execution resources.
- RL and Agentic AI Performance – Highlights Vera's value for latency-sensitive, branch-heavy, and highly concurrent agent environments.

## Memory Bandwidth and Latency

### Memory Latency

Loaded memory latency is a critical measure of real-world CPU performance because modern workloads rarely access memory in isolation. Agentic AI, graph analytics, databases, data processing, and RL environments often generate many simultaneous memory requests across cores, stressing both the memory subsystem and the coherency fabric. Traditional chiplet-based CPUs can struggle under these conditions because requests may cross NUMA domains, traverse inter-chiplet fabrics, or access memory attached to a different die. These extra hops increase peak latency, reduce effective bandwidth, and introduce performance variability as system load increases.

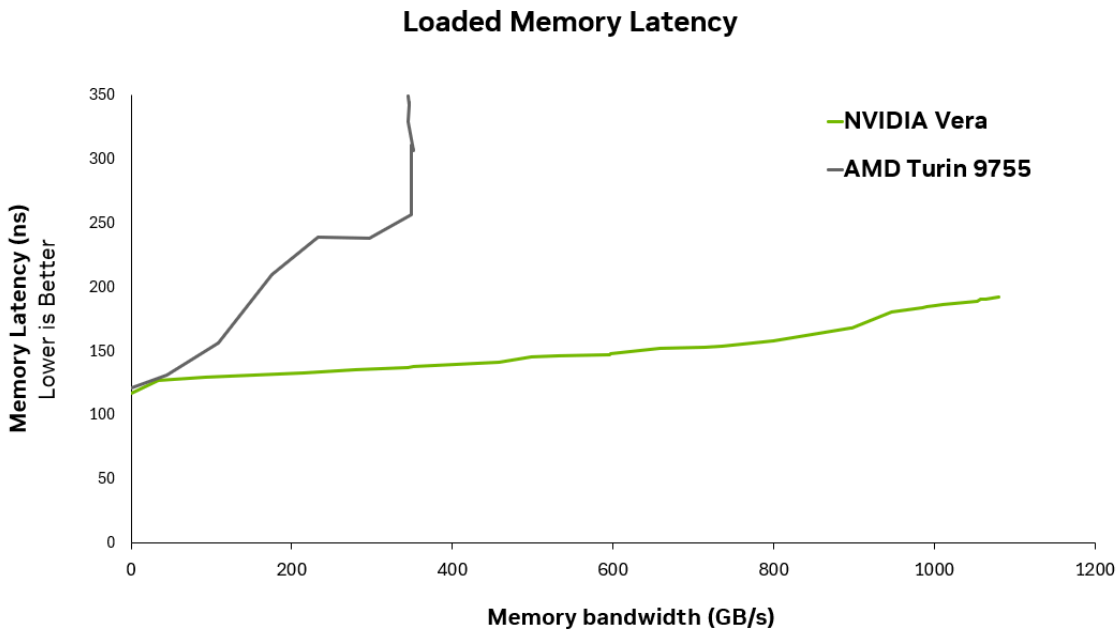


Figure 11. Vera delivers 3x higher bandwidth with 40% lower peak latency

Vera is designed to avoid these bottlenecks. With a single monolithic compute die, unified cache architecture, and high-bandwidth SOCAMM LPDDR5X memory, Vera delivers up to 40% lower peak loaded memory latency. Vera achieves close to 90% DDR utilization, converting more of its available memory bandwidth into effective workload bandwidth and delivering more than 3x  memory bandwidth vs competition.

This combination enables Vera to maintain high throughput even under heavy memory pressure while providing more deterministic performance across cores. This deterministic behavior allows latency-sensitive and highly concurrent workloads to scale more predictably, without the NUMA complexity and chiplet-related variability common in competing architectures.

## Memory BW per Core

Memory bandwidth per core is a better measure of balance than sheer core count, cores only deliver performance when they are consistently fed with data. As shown in Figure 12, Vera delivers 12.7 GB/s of memory bandwidth per core, compared with 3.1 GB/s per core for the competing x86 CPU, providing more than 4x higher bandwidth per core. This balance of high-performance Olympus cores and high memory bandwidth is especially important for agentic AI, graph analytics, data processing, and RL workloads, where many threads operate on large, irregular data sets and can quickly become limited by memory throughput rather than compute resources.

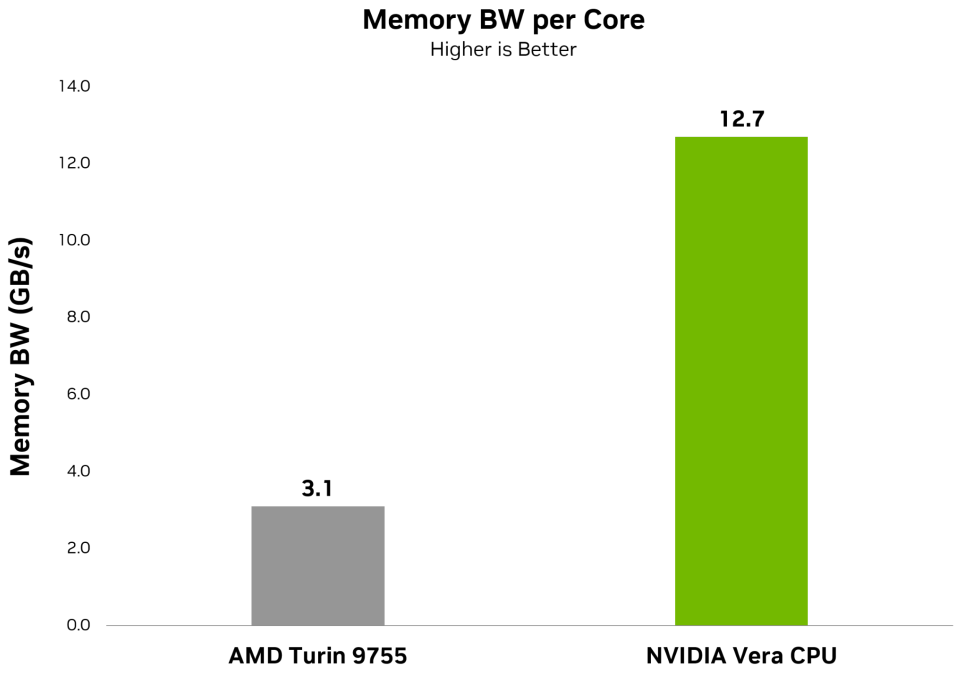


Figure 12. Vera delivers more than 4x memory BW per core

## Core-to-Core Latency

Legacy chiplet architectures can include more than a dozen separate compute and I/O chiplets, and the impact of that topology becomes especially visible in core-to-core transfers. When data moves from one core to another, it may need to leave the source core die, traverse the package fabric through an I/O die, reach a different core die, and then follow a similar path back for coherency responses. Each die hop adds latency, consumes fabric bandwidth, and creates variability depending on where the communicating cores are physically located. The result is higher core-to-core latency, less consistent performance, and greater software sensitivity to thread placement.

NVIDIA built Vera around a monolithic compute die to avoid these die-hop penalties and deliver more uniform core-to-core communication. By keeping the Olympus cores, shared cache resources, and coherent fabric on a single die, Vera reduces latency variation and provides a more predictable execution environment across the processor. This is especially important for latency-sensitive agentic AI, reinforcement learning, graph analytics, and data-processing workloads, where many threads frequently exchange state, synchronize, or access shared data structures. Vera's monolithic design helps these workloads scale with lower latency, better determinism, and fewer topology-related performance cliffs.

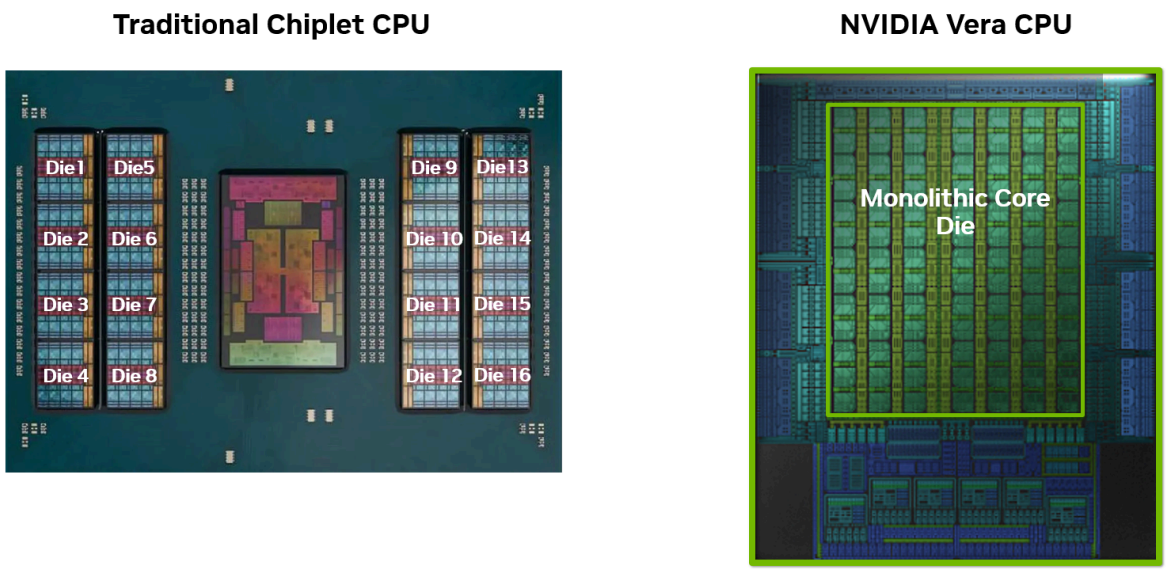


Figure 13. NVIDIA Vera CPU with an unified core die

The result is evident in the heat map shown in Figure 14, illustrating the Core-to-Core latency across Vera CPU and traditional x86 CPU. Vera delivers up to 50% lower core-to-core latency than competing x86 architectures. More importantly, the Vera heat map shows highly consistent latency across core pairs, reflecting a deterministic monolithic-die design rather than the uneven latency bands and topology-driven variation common in chiplet-based CPUs.

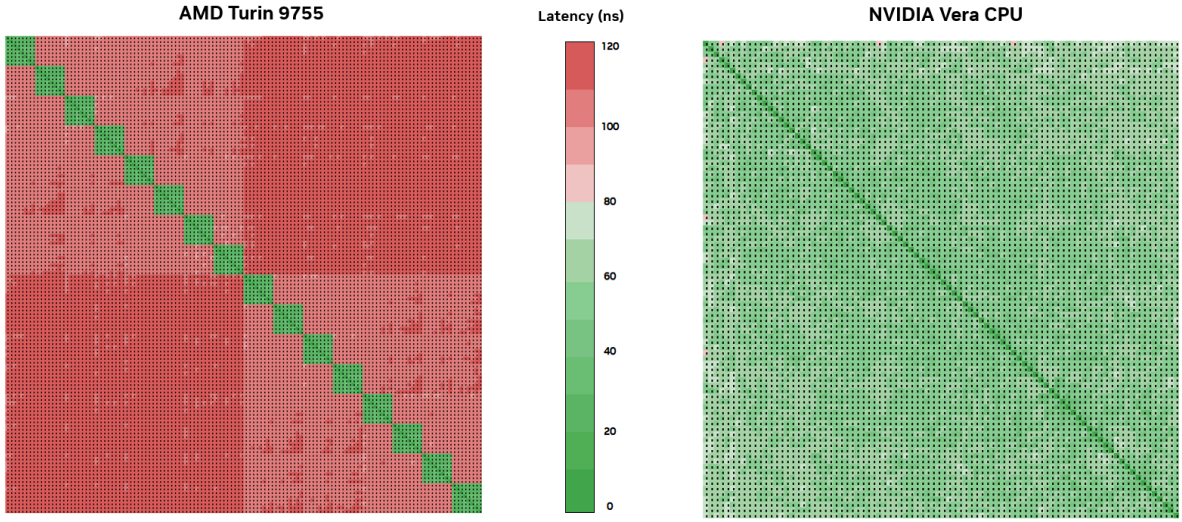


Figure 14. Vera delivers deterministic performance with 50% lower latency

# Workloads

## Agentic Benchmarks

Agentic benchmarks are evolving as AI systems move from simple request-response interactions to executing tools, writing and analyzing code, and operating across software environments. Some of the foundational workloads common across agentic calls include Python execution, code compilation, static analysis, and tool-driven software workflows. These workloads stress the same CPU behaviors that matter for agents: large code footprints, branch-heavy control flow, dynamic runtime behavior, and dependency-heavy execution paths. Figure 15 showcases Vera CPU performance measured across the four key workloads, delivering up to 1.8x higher performance vs latest x86 CPU. SPEC CPU® 2026 refreshes a familiar reference point — 52 benchmarks, twice the source code of SPEC CPU® 2017, and a 64GB footprint that better reflects modern hardware. While still a CPU-level suite rather than an agentic one, its coverage of AI, compilers, databases, and graph analytics makes it a reasonable stand-in for some of the work underneath agentic tool-calling pipelines.

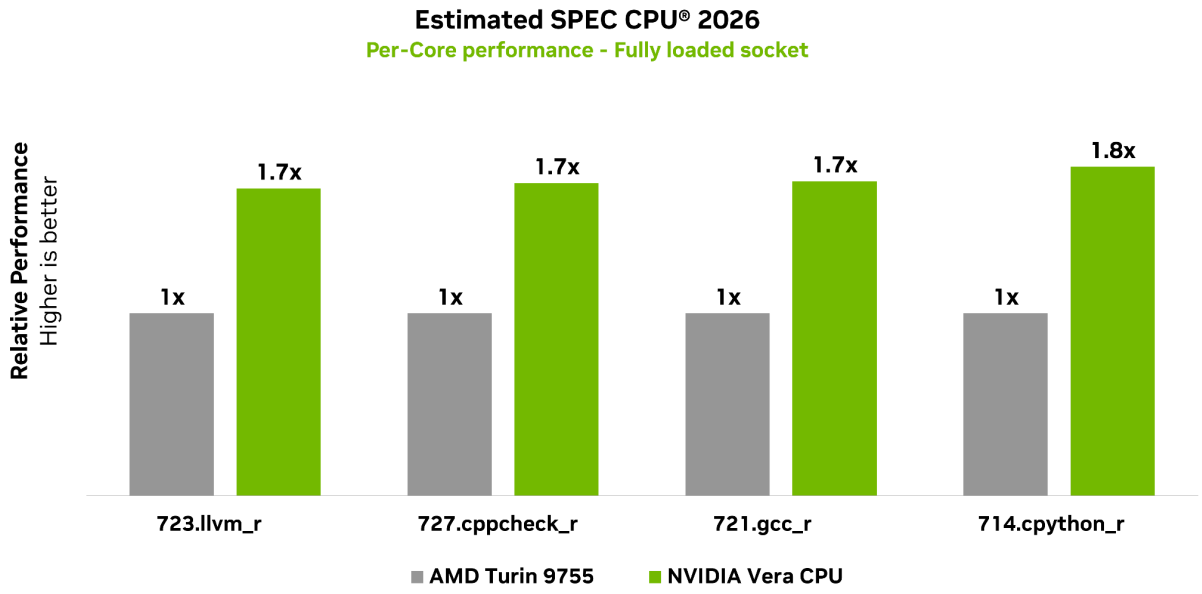


Figure 15. NVIDIA Vera CPU outperforms competition across agentic benchmarks

The figure above compares per-core performance under a fully loaded socket. This metric is non-trivial for agentic AI and RL systems, where many sandboxes, tools, and environments run concurrently rather than as isolated single-thread tests. Fully loaded

per-core performance captures how well each core sustains throughput while sharing socket-level power, memory bandwidth, cache, and fabric resources.

## Graph Traversal

Graph algorithms are foundational to many data-intensive workloads as they model relationships across large, irregular data sets including web links, recommendations, knowledge graphs, dependency graphs, social networks, and search indexes. They are also increasingly relevant to agentic AI, where agents often interact with structured memory, retrieval systems, tool graphs, planning workflows, and environment state represented as connected objects. One such algorithm is PageRank, which repeatedly traverses large sparse graphs, generating irregular memory accesses with relatively low compute per byte. As a result, performance is heavily influenced by memory bandwidth, cache efficiency, and the ability to sustain many outstanding memory requests.

Vera is designed with hardware graph prefetching capabilities specifically for graph workloads like PageRank, where performance is often limited by indirect, irregular memory access patterns rather than raw compute. By learning producer-consumer address relationships in graph traversal, Vera can prefetch data more effectively and keep the Olympus cores supplied with useful work. Combined with high memory bandwidth, strong bandwidth per core, and efficient on-die data movement, Vera delivers 2.6x higher graph traversal performance than AMD Turin. This highlights how Vera's architecture is optimized for graph analytics workloads that stress memory bandwidth, cache efficiency, and memory-level parallelism.

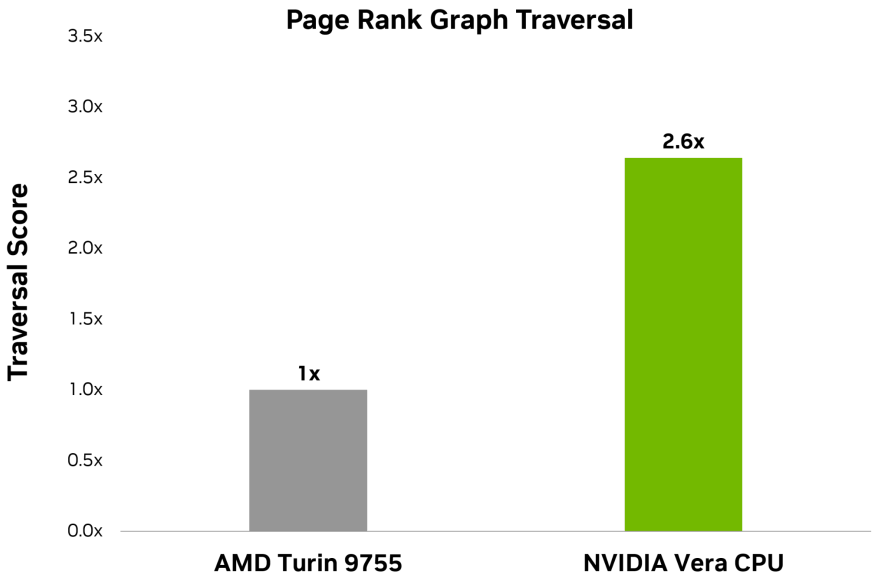


Figure 16. Vera delivers 2.6x faster graph traversal performance

## Graph Traversal Core Scaling

Graph traversal workloads such as PageRank stress both the core pipeline, fabric and memory bandwidth. As core counts increase, performance depends on how efficiently the processor can move graph data across cores, caches, memory controllers, and the coherent fabric. In chiplet-based CPUs, limited inter-chiplet bandwidth and additional die hops can become a bottleneck, causing scaling to flatten as more cores compete for fabric and memory resources.

Vera is purpose-built for this class of agentic and graph-intensive workload. Its monolithic die, high-bandwidth SCF, LPDDR5X memory, and graph prefetching capabilities help sustain data movement as more Olympus cores are activated. This allows Vera to scale more efficiently on irregular graph traversal workloads where memory bandwidth, latency, and fabric efficiency are critical as illustrated in Figure 17.

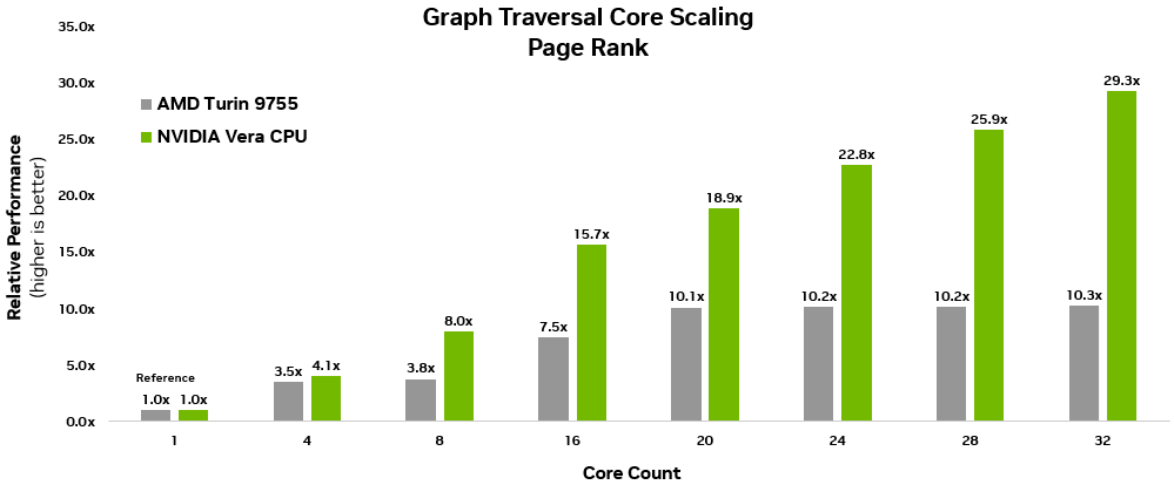


Figure 17. Vera delivers consistent core-scaling performance

## ClickHouse

Vera is built for data analytics and data processing, combining up to 1.2 TB/s of memory bandwidth with a unified on-die fabric that moves data efficiently across cores, cache, and memory. This makes Vera a strong fit for scan-heavy, aggregation-heavy, and concurrency-sensitive analytics and ETL (Extract-Transform-Load) workloads where performance depends on feeding every core with data, not just adding more cores. ClickHouse is a good representative workload due to its high-performance columnar database used for online analytical processing, with query execution patterns that stress memory bandwidth, cache efficiency, and cross-core data movement. Vera delivers 1.2x higher performance compared to competition.

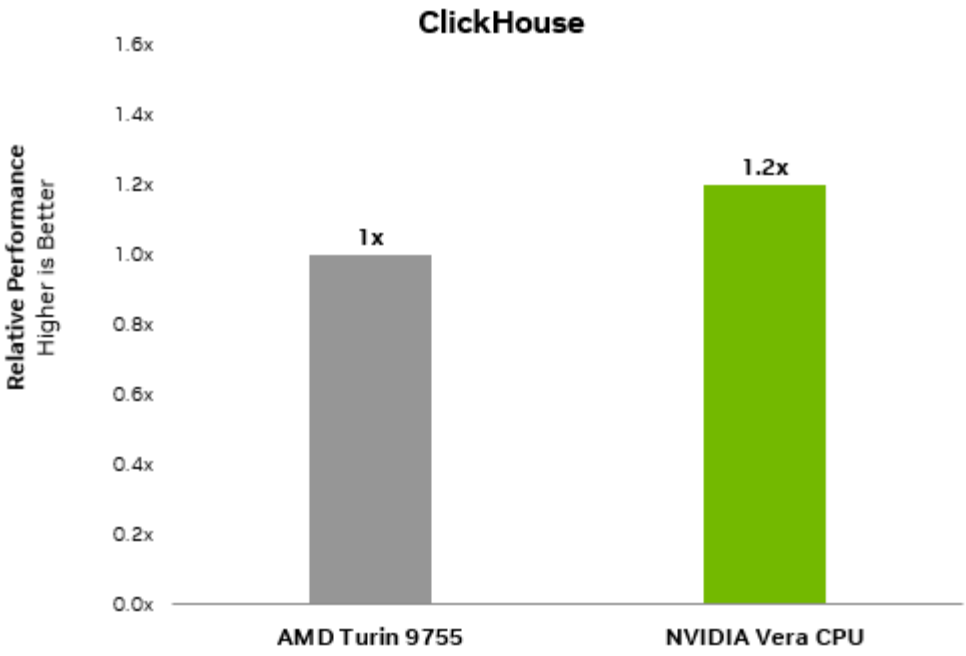


Figure 18. Vera drives higher data processing throughput

# Olympus Core IPC

High IPC is critical for agents as each reasoning, tool-use, and environment step depends on strong single-core performance through irregular control flow and short feedback loops. Olympus core as described in the earlier section, was built for maximum single-thread performance under full socket load through a wider, more efficient core rather than relying only on higher clock frequency. The core combines a high-bandwidth Front End  with advanced branch prediction, 10-wide decode, a deep out-of-order execution engine, wider execution resources, and a deep cache hierarchy to keep useful instructions flowing through the pipeline.

Traditional x86 CPUs often push frequency higher to improve single-thread performance, but that approach can increase power and becomes harder to sustain when all cores are active. Olympus takes a different path, a purpose-built wide core that extracts more work per cycle, enabling strong single-thread performance with better efficiency under loaded, agentic-style conditions.

This IPC advantage is evident across the four representative agentic workloads shown in Figure 19. These workloads stress large instruction footprints, dense control flow, compiler and runtime behavior, and long dependency chains, making them strong proxies for CPU-side agent execution. The results show that Olympus extracts substantially more work per cycle, translating its wider, more efficient core architecture into strong single-thread performance for branch-heavy and dependency-heavy software.

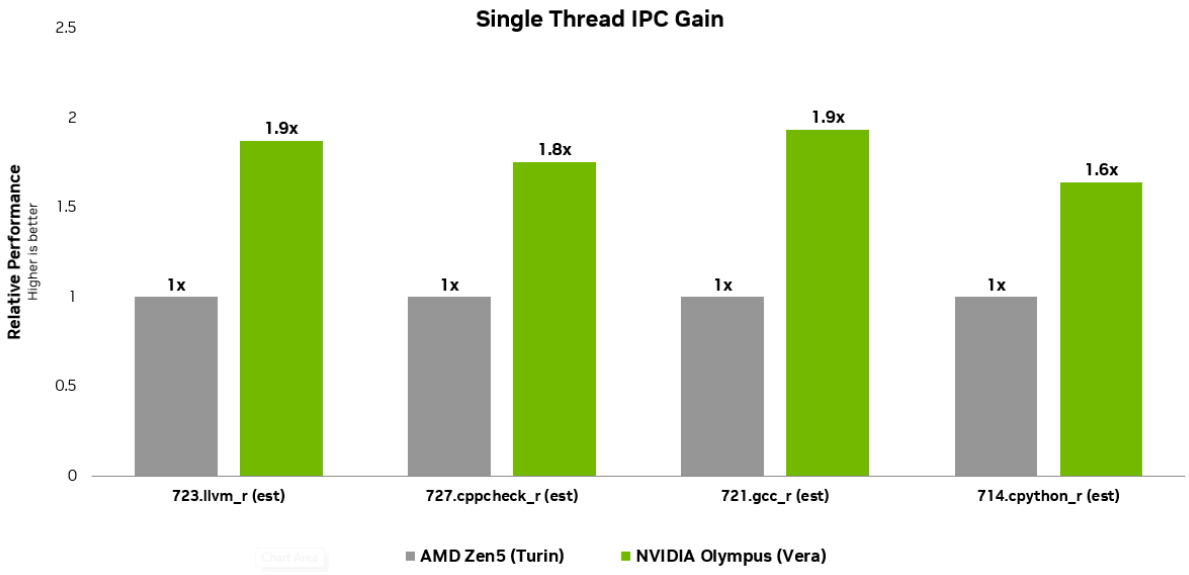


Figure 19. Olympus core delivers up to 1.9x higher single thread IPC in a loaded system

The below areas provide further insight on how Olympus delivers 1.9x higher single thread IPC.

## Branch Prediction

Olympus improves IPC by driving more branch predictions through the frontend each cycle, which is critical for dense control-flow code with short basic blocks. When the branch prediction unit cannot generate targets quickly enough, fetch and decode bandwidth are limited even if the rest of the core has available execution capacity. Olympus implements advanced predictors with robust ahead pipelining mechanisms delivering up to 2.3x higher predictions per cycle vs competition as shown in Figure 20.

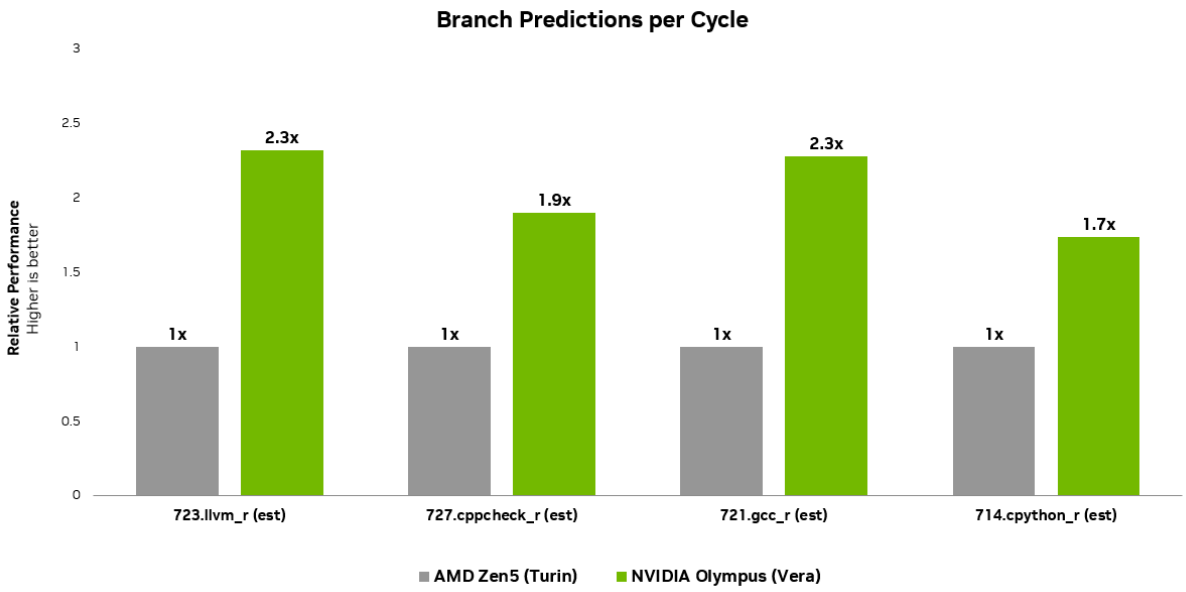


Figure 20. Olympus core delivers up to 2.3x higher branch predictions per cycle

Throughput alone is not enough, the predictions also need to be accurate so the core does not waste cycles executing down the wrong path. While others may advertise support for two taken branches per cycle in their CPUs, the alternatives can often only achieve it under more constrained conditions than Olympus does.

Olympus is designed to sustain high taken-branch processing across dense and varied branch patterns, made possible by the neural branch predictor in addition to other special-purpose predictors. Olympus BPU (Branch Prediction Unit) operates at a higher effective rate while still maintaining a branch MPKI (Missed Prediction per1000 Instructions) advantage, delivering more useful instruction streams into the wide decode and execution pipeline. The result is better front-end utilization, fewer wasted cycles, and more useful instructions, achieving up to 3.5x higher taken-branches per cycle as shown in Figure 21.

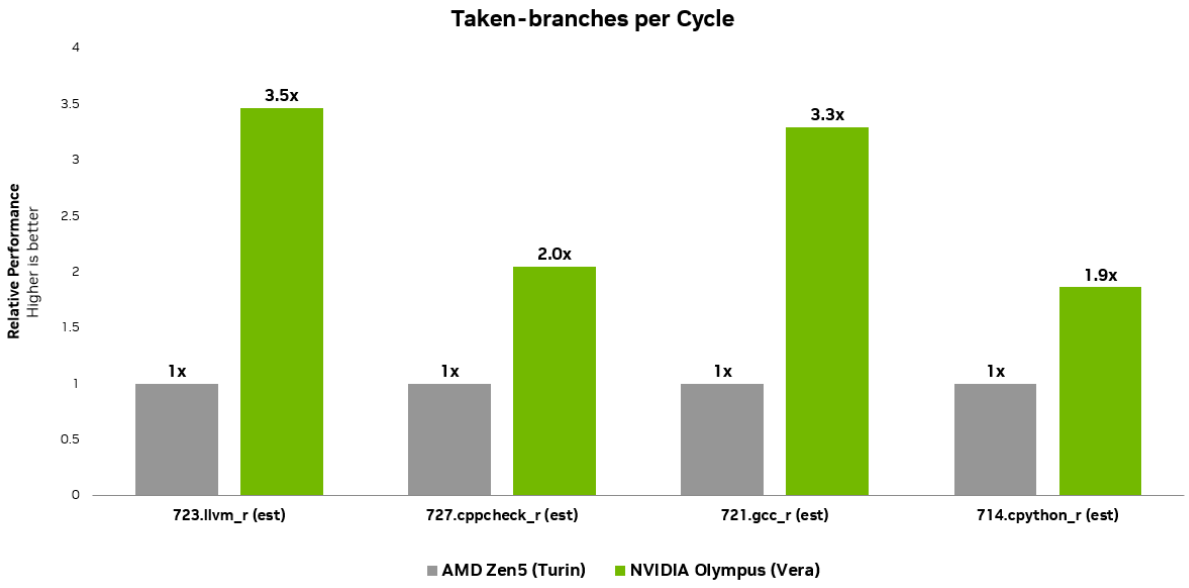


Figure 21. Olympus branch predictors deliver up to 3.5x higher taken-branches

# Instruction Fetch

Olympus improves instruction-fetch throughput by reducing front-end starvation in large-footprint code. The core combines a 64 KB instruction cache, high-bandwidth fetch of 128B per cycle out of the instruction cache, and 10-wide decode to keep the pipeline supplied without relying on an x86-style uOP cache (stores decoded instructions) that can lose effectiveness when code footprints spill beyond its capacity. This is especially important for interpreter, compiler, and static-analysis style workloads, where high instruction-cache pressure can starve the backend before execution resources are fully utilized.

Vera delivers up to 2.4x higher instruction-fetch operations per cycle versus competition, showing that Olympus spends fewer cycles waiting on instruction delivery and more cycles feeding useful work into the core as shown in Figure 22.

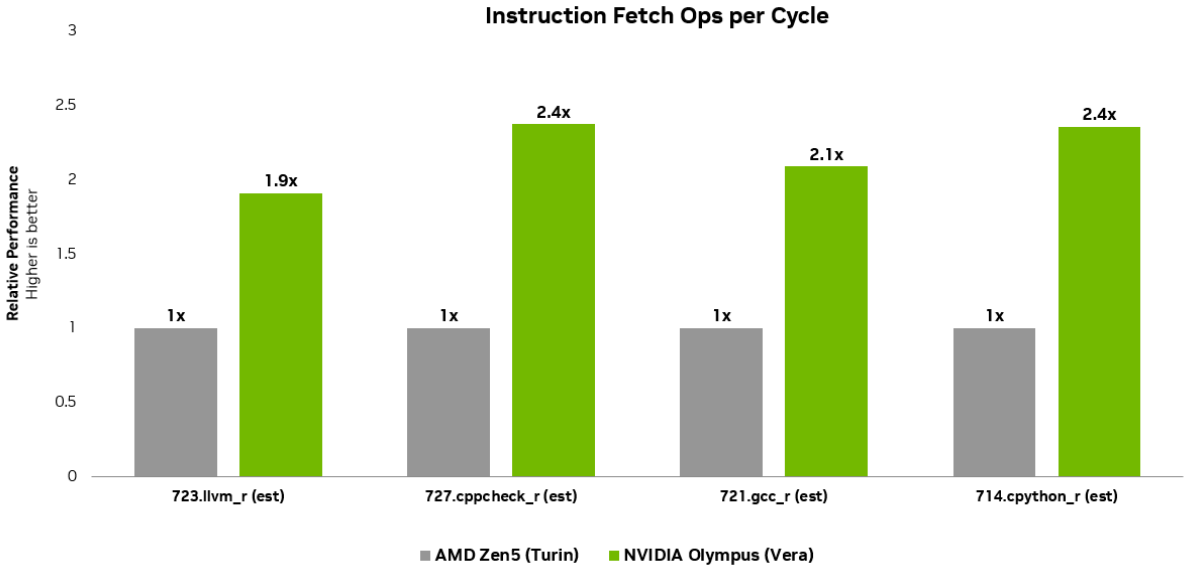


Figure 22. Olympus core delivers up to 2.4x higher Ops per cycle

## Backend Operations

Olympus improves backend throughput by keeping the core productive when workloads encounter cache misses, dependency chains, and irregular memory accesses. Its deep out-of-order engine can look further ahead and schedule independent work while long-latency operations are outstanding, while the larger 96 KB L1D cache and 2 MB private L2 cache reduce backend stalls before requests spill deeper into the memory hierarchy. This allows Vera to bring data into the core and execute instructions in fewer cycles vs competition. As shown in Figure 23, Vera delivers up to 4.3x higher backend operations per cycle, translating backend efficiency directly into higher IPC.

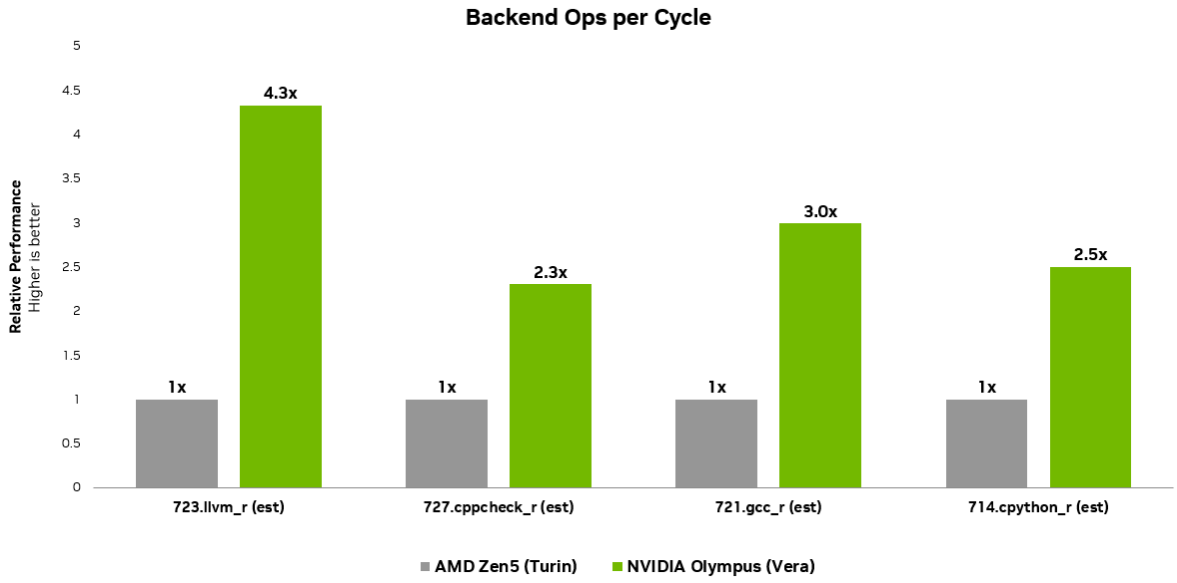


Figure 23. Olympus core delivers up to 4.3x higher backend ops per cycle

# AI Factory Considerations for Reinforcement Learning and Agents

As discussed earlier, reinforcement learning and agentic AI combine large-scale concurrency with sequential, branch-heavy execution inside each rollout or agent trace. In reinforcement learning, rollout execution, reward computation, and feedback preparation determine how many useful environment evaluations complete within a training window, while in agentic inference, tool calls, retrieval, and sandboxed execution determine how quickly the next model step can begin.

For these workloads, the key value of Vera is not only its 88 Olympus cores and 176 Spatial Multithreading threads, but the ability of each core to sustain progress on irregular software. The 10-wide Front End , neural branch predictor, and deep out-of-order engine are directly relevant to RL environments, Python runtimes, and agent frameworks that spend significant time in control-flow-heavy code, while the 2 MB private L2 at approximately 10-cycle access latency and unified 164 MB L3 reduce the cost of repeatedly accessing shared state, reward data, and intermediate results across cores. Vera also delivers lower L3 latency, which is particularly important when these workloads cannot remain inside private core-local structures.

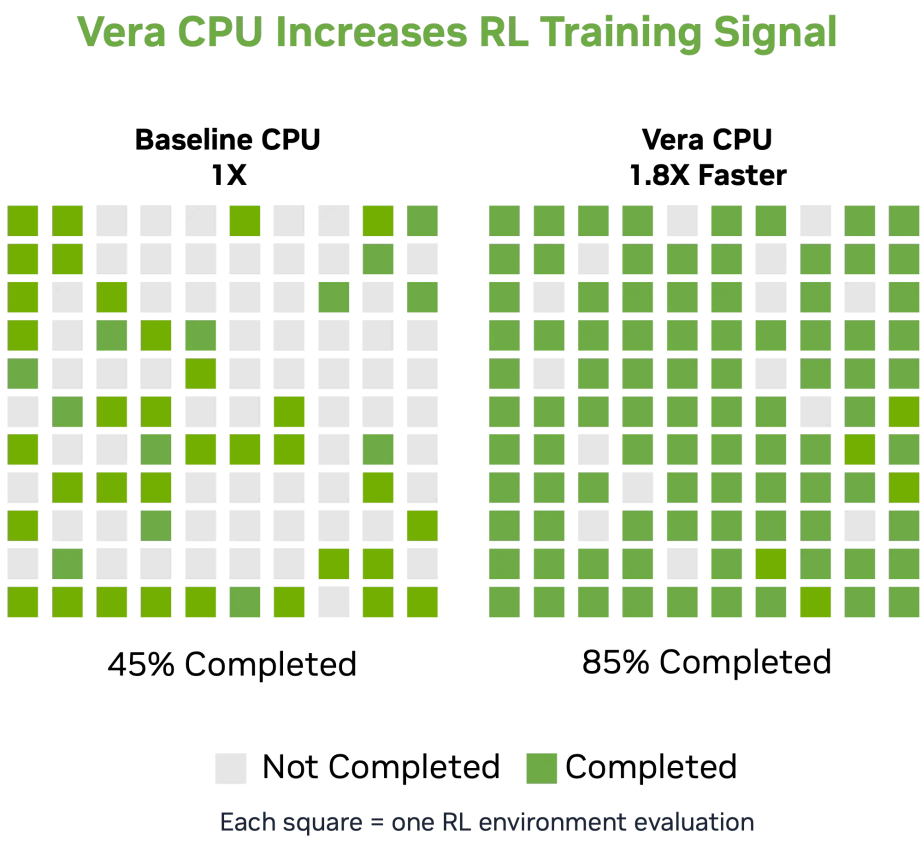


Figure 24. Vera drives 1.8x for RL training

Under full socket load, these workloads are frequently limited by memory behavior as much as by instruction throughput. Vera provides up to 1.2 TB/s of memory bandwidth, 12.7 GB/s of bandwidth per core, more than 3x higher total memory bandwidth versus competition, and up to 40% lower peak loaded memory latency, allowing a larger fraction of CPU time to translate into completed rollout work, tool execution, and data movement rather than stalls. For reinforcement learning, this increases the rate at which rewards and feedback can be generated for downstream policy optimization; for agentic inference, it reduces the CPU-side gap between successive model invocations, improving both responsiveness and overall CPU-GPU utilization across the AI factory.

# Configurations

SPEC CPU®2026 Configurations

Above results using SPEC CPU 2026 compare the relative performance for the benchmarks most relevant to Agnetic AI.  Below are the estimated SPECrate®2026_int_base results.  While every attempt was made to follow the benchmark's run and reporting rules, the Vera hardware used was a reference system, hence does not comply with the availability requirements.

For both systems, the GNU 15.2 compiler was used, including the same flag set, in order to give a better system to system comparison and a more representative view for how Agentic AI applications are compiled.

GNU compiler flag set: "-g -Ofast -flto" with "-mcpu=olympus" or "-march=znver5" and jemalloc, a general purpose "malloc" library implementation.

Results run internally July 2026.

Systems:

"Turin":  Supermicro AS -8126GS-NB3RT-OTO-71

CPU: AMD EPYC 9755, 2 sockets, 128 cores per socket

Memory: 24 x 96 GB DDR5 RDIMM, 2Rx4, PC5-6400B-R

Copies: 512

SPECrate®2026_int_base (est.): 898

"Vera": NVIDIA C2 9600MTS

CPU: NVIDIA Vera CPU, 2 sockets, 88 cores per socket

Memory: 1536 GB (LPDDR5X-9600 MT/s)

Copies: 352

SPECrate®2026_int_base (est.): 925

| Vera Estimated | Copies | Run Time | Rate |
|---|---|---|---|
| 706.stockfish_r | 352 | 324 | 1370 |
| 707.ntest_r | 352 | 251 | 830 |
| 708.sqlite_r | 352 | 250 | 744 |
| 710.omnetpp_r | 352 | 203 | 842 |

| | | | |
|---|---|---|---|
| 714.cpython_r | 352 | 136 | 1240 |
| 721.gcc_r | 352 | 296 | 817 |
| 723.llvm_r | 352 | 196 | 909 |
| 727.cppcheck_r | 352 | 142 | 890 |
| 729.abc_r | 352 | 196 | 823 |
| 734.vpr_r | 352 | 199 | 815 |
| 735.gem5_r | 352 | 131 | 1300 |
| 750.sealcrypto_r | 352 | 231 | 816 |
| 753.ns3_r | 352 | 129 | 1670 |
| 777.zstd_r | 352 | 469 | 483 |
| SPECrate 2026_int_base (est.) | | | 925 |



**NVIDIA Vera** Platform: C2 9600MTS | CPU NVIDIA Vera CPU | Socket Count: 2 | Cores / Threads: 88/176 | Memory: Samsung 1.5 TB 9600 MT/s LPDDR5X | Storage: KIOXIA KXD8DRJ93T84 3.5 TB | NIC: BlueField-3 | BIOS: buildbrain-gcid-sbios-46230282 | OS: Ubuntu 24.04.3 LTS | Kernel: Linux 6.17.0-1018-nvidia-64k| Compiler:GCC15 | **AMD Turin** Platform: Supermicro AS-8126GS-NB3RT | CPU: AMD EPYC 9755 | Socket Count: 2 | Cores / Threads: 128/256 | Memory: Micron 2.3 TB 6400 MT/s DDR5 | Storage: MTFDLCE3T8THA-1BK1DABYY 3.8 TB | NIC: BlueField-3 | BIOS: American Megatrends International, LLC. Version 1.2 | OS: Ubuntu 24.04.4 LTS | Kernel: Linux 6.17.0-19-generic | Compiler: GCC15 | Memory latency https://github.com/dsheffie/mem-lat/ | Stream https://www.cs.virginia.edu/stream/ | GapBS PR https://github.com/sbeamer/gapbs/ | Clickhouse - https://www.phoronix.com/review/nvidia-vera-benchmarks/10/

## Notice

This document is provided for information purposes only and shall not be regarded as a warranty of a certain functionality, condition, or quality of a product. NVIDIA Corporation (“NVIDIA”) makes no representations or warranties, expressed or implied, as to the accuracy or completeness of the information contained in this document and assumes no responsibility for any errors contained herein. NVIDIA shall have no liability for the consequences or use of such information or for any infringement of patents or other rights of third parties that may result from its use. This document is not a commitment to develop, release, or deliver any Material (defined below), code, or functionality.

NVIDIA reserves the right to make corrections, modifications, enhancements, improvements, and any other changes to this document, at any time without notice.

Customer should obtain the latest relevant information before placing orders and should verify that such information is current and complete.

NVIDIA products are sold subject to the NVIDIA standard terms and conditions of sale supplied at the time of order acknowledgement, unless otherwise agreed in an individual sales agreement signed by authorized representatives of NVIDIA and customer (“Terms of Sale”). NVIDIA hereby expressly objects to applying any customer general terms and conditions with regards to the purchase of the NVIDIA product referenced in this document. No contractual obligations are formed either directly or indirectly by this document.

NVIDIA makes no representation or warranty that products based on this document will be suitable for any specified use. Testing of all parameters of each product is not necessarily performed by NVIDIA. It is customer’s sole responsibility to evaluate and determine the applicability of any information contained in this document, ensure the product is suitable and fit for the application planned by customer, and perform the necessary testing for the application in order to avoid a default of the application or the product. Weaknesses in customer’s product designs may affect the quality and reliability of the NVIDIA product and may result in additional or different conditions and/or requirements beyond those contained in this document. NVIDIA accepts no liability related to any default, damage, costs, or problem which may be based on or attributable to: (i) the use of the NVIDIA product in any manner that is contrary to this document or (ii) customer product designs.

No license, either expressed or implied, is granted under any NVIDIA patent right, copyright, or other NVIDIA intellectual property right under this document. Information published by NVIDIA regarding third-party products or services does not constitute a license from NVIDIA to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property rights of the third party, or a license from NVIDIA under the patents or other intellectual property rights of NVIDIA.

Reproduction of information in this document is permissible only if approved in advance by NVIDIA in writing, reproduced without alteration and in full compliance with all applicable export laws and regulations, and accompanied by all associated conditions, limitations, and notices.

THIS DOCUMENT AND ALL NVIDIA DESIGN SPECIFICATIONS, REFERENCE BOARDS, FILES, DRAWINGS, DIAGNOSTICS, LISTS, AND OTHER DOCUMENTS (TOGETHER AND SEPARATELY, “MATERIALS”) ARE BEING PROVIDED “AS IS.” NVIDIA MAKES NO WARRANTIES, EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE MATERIALS, AND EXPRESSLY DISCLAIMS ALL IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE. TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL NVIDIA BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF NVIDIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Notwithstanding any damages that customer might incur for any reason whatsoever, NVIDIA’s aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms of Sale for the product.

## Trademarks

NVIDIA, the NVIDIA logo, are trademarks and/or registered trademarks of NVIDIA Corporation in the U.S. and other countries. Other company and product names may be trademarks of the respective companies with which they are associated.

## Copyright





NVIDIA Corporation | 2788 San Tomas Expressway, Santa Clara, CA 95051
http://www.nvidia.com