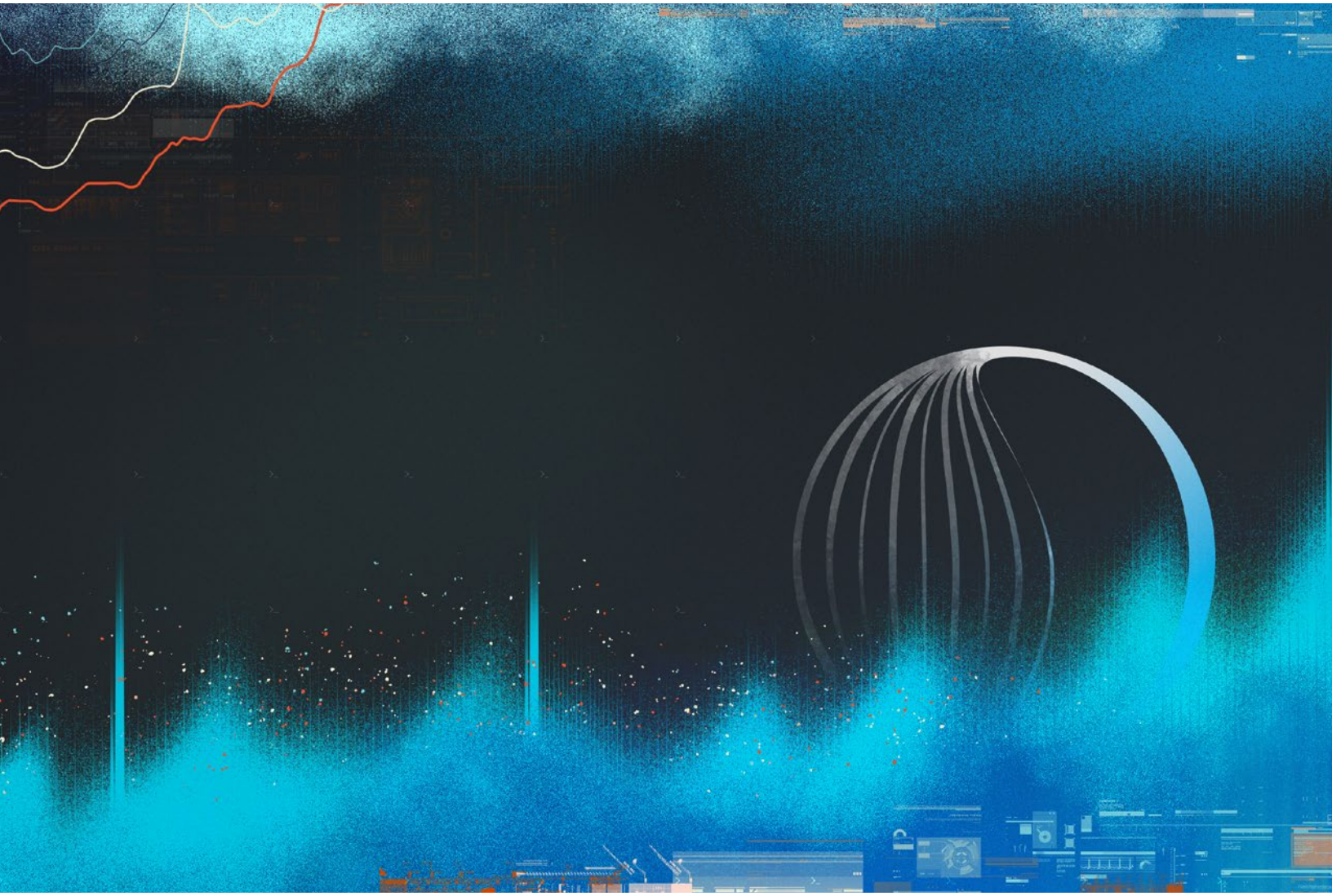
signal65



Deploy Open Models in Minutes: Serving AI Agents at Scale on Microsoft Foundry with AMD Instinct™ MI300X GPUs

AUTHORS

Russ Fellows
VP, Labs | Signal65

Ryan Shrout
President and GM | Signal65

COMMISSIONED BY

Microsoft

NOVEMBER 2026

Executive Summary

A Signal65 study of deployment speed, concurrent agent capacity and cost efficiency for dedicated, private model endpoints.

Key Highlights

14–24 min
From Foundry catalog to a live MI300X endpoint

128 agents
On four MI300X GPUs running DeepSeek V4.1 Flash (1017) with a one-million-token context

93%–100%
Of tuned self-hosted per-agent speed at full agent load

~112,000
gpt-oss-120b agent sessions per $10,000 of MI300X time

Enterprise adoption of generative AI continues to outpace the supply of skilled AI infrastructure engineers. Many organizations want to run open-weight models on dedicated capacity, with predictable performance and a model version they control. Until recently, that meant buying accelerators, waiting through long lead times and hiring scarce engineers to configure and tune GPU serving software.

Microsoft Foundry Managed Compute offers another path. It pairs dedicated GPU and infrastructure, like the AMD Instinct™ MI300X GPUs featured here, with curated deployment templates that Microsoft maintains, so a team can stand up a dedicated, production-ready model endpoint without buying hardware, signing a long-term commitment or tuning GPU kernels. In Signal65's testing, open models on Microsoft Foundry went from catalog selection to a live MI300X endpoint in 14 to 24 minutes, or under an hour for DeepSeek V4.1 Flash (1017) with its one-million-token context, by choosing the pre-configured deployment templates. The 192 GB of HBM3 on each MI300X GPU lets a model as large as gpt-oss-120b run on a single GPU with room for dozens of concurrent agents, which keeps the GPU count and the hourly bill low.

Signal65 evaluated the service the way a customer would. We deployed open-weight models from the Foundry catalog, used each template exactly as provided, and drove the endpoints with agentic and conversational workloads. To see what a managed service might give up, we then ran the same models and workloads on MI300X servers in our own lab, tuned by our engineers.

This report covers four widely used open-weight models, OpenAI's gpt-oss-120b, Google's Gemma 4 31B, Z.ai's GLM-5.3 Flash and DeepSeek V4.1 Flash (1017). It is organized around three questions: how easily the models deploy, how many concurrent AI agents each supports for a defined workload, and what a practical budget delivers.

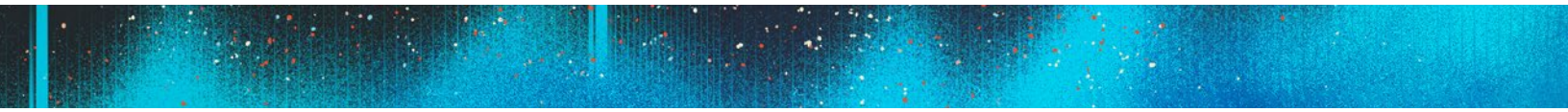
Key Findings

- **14 to 24 minutes (under an hour at one-million-token context)** from the Foundry catalog to a live endpoint, with teardown in two to eight minutes and billing only while a deployment was live
- **No serving configuration** required from the customer, because the templates chose the serving kernels, sized memory and context, and set multi-GPU parallelism
- **Up to 32 concurrent enterprise agents** on a single MI300X GPU running gpt-oss-120b, at 701 output tokens per second
- **128 concurrent enterprise agents** on four MI300X GPUs running DeepSeek V4.1 Flash (1017) with a one-million-token context, at 1,723 output tokens per second

© 2026 Signal65. All rights reserved.

- **96 concurrent enterprise agents** on four MI300X GPUs running GLM-5.3 Flash, at 1,316 output tokens per second, plus 3,397 tokens per second on short-prompt chat
- **93% to 100%** of the per-agent speed of the same models on our tuned MI300X systems at full agent load
- **About $0.09** of GPU time per complete gpt-oss-120b agent session

Dedicated, private AI on MI300X GPUs no longer depends on a hardware purchase or a specialist operations team. In Foundry, a team can deploy a dedicated endpoint in minutes and delete it when the work is done.



Choosing a Deployment Model for Open-Weight AI

Many enterprises want an open-weight model matched to the workload, control over where their data goes, and the ability to move without long lead times or large capital commitments. Each common deployment approach balances those needs differently, and most organizations will use more than one of them, choosing by workload.

- **Serverless model APIs:** Offer the fastest start, with pay-per-token pricing and no capacity to manage, and fit variable or early-stage workloads well. Capacity is shared, and the provider manages the model version and the serving stack.
- **Self-managed infrastructure:** Gives full control over hardware, software and model versions, on premises or on dedicated cloud instances. OEM systems, validated solutions and vendor containers shorten set-up, but the team still owns driver and runtime choices, kernel selection, KV-cache sizing, parallelism and hardware health.
- **Foundry Managed Compute:** Provides dedicated capacity and a fixed model version per deployment, with Microsoft operating the serving stack through curated templates and billing by the GPU-hour.

	Serverless model APIs	Self-managed infrastructure	Foundry Managed Compute
Time to first endpoint	Minutes	Weeks to months, depending on hardware lead times and in-house expertise	14 to 24 minutes, under an hour at 1M context
Capacity	Shared	Dedicated	Dedicated
Model version	Managed by the provider	Fixed by your team	Fixed per deployment
Serving stack	Run by the provider	Built, tuned and run by your team	Run by Microsoft through curated templates
Billing	Per token	Up-front hardware plus operations	Per GPU-hour, no commitment, stops when deleted
Skills needed	Application developers	GPU, serving and systems engineers	Application developers

For workloads that need dedicated capacity and a stable model version, managed compute delivers both without an in-house serving team.

© 2026 Signal65. All rights reserved.



Platform Overview

Each MI300X GPU carries 192 GB of HBM3 memory and 5.3 TB/s of peak memory bandwidth, which lets large models run on relatively few GPUs. In our testing, a single MI300X GPU ran OpenAI's gpt-oss-120b, a 120-billion-parameter mixture-of-experts model, and supported 32 concurrent agents within the service thresholds described in Users Supported by Workload. Its memory holds the model weights with room for the working context of every agent. Memory bandwidth matters as much as capacity, because generating each token streams model weights and cached context from memory.

Foundry users do not provision virtual machines or build containers. They choose a curated template from the Foundry catalog that names the model, the number of MI300X GPUs and the context window, such as "openai--gpt-oss-120b--128k-amd-mi300x." Each deployment is dedicated capacity running a fixed model version, and Microsoft supports scaling a deployment out to several instances behind one endpoint. Every deployment in our tests used a single instance.

Microsoft offers two placement options. Global Managed Compute schedules deployments across regional capacity pools for broad availability, and DataZone Managed Compute keeps processing within a defined geography for organizations with data-residency requirements.

Every deployment exposes a private HTTPS endpoint with an OpenAI-compatible API, which we secured with Foundry account keys. Tool calls worked on all four endpoints, and gpt-oss-120b and GLM-5.3 Flash returned their reasoning in a separate field, as OpenAI-compatible clients expect. Most applications built on OpenAI SDKs can connect by changing only an endpoint and a key.

MI300X model coverage in Foundry

As of November 4th, 2026, Microsoft listed 54 MI300X templates covering 47 models from 18 publishers, ranging from compact text-embedding models on a single GPU to models of nearly 400 billion parameters on eight. The table below shows a selection of current models from that list, including the four from our testing, with context windows from 16K tokens to one million.

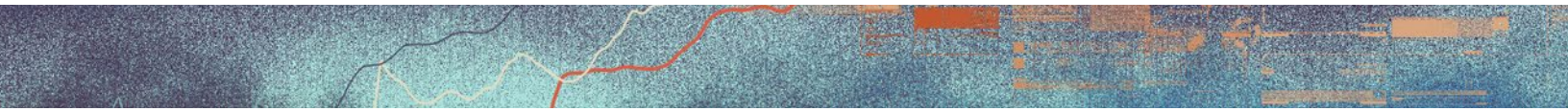
Several models offer more than one MI300X layout. GLM-5.3 Flash comes in a four-GPU template and two eight-GPU templates that divide the GPUs differently between tensor and data parallelism, and DeepSeek V4.1 Flash (1017) serves its one-million-token context on either four or eight GPUs. Teams can pick the layout that fits their concurrency and context needs, and applications connect to every template through the same OpenAI-compatible API.

Model	Provider	MI300X templates	MI300X GPUs	Context window
gpt-oss-120b	OpenAI	128k-amd-mi300x	1	128K
gpt-oss-20b	OpenAI	128k-amd-mi300x	1	128K
Gemma 4 31B	Google	256k-amd-mi300x	1	256K
Gemma 4 26B A4B	Google	256k-amd-mi300x	1	256K

© 2026 Signal65. All rights reserved.

Model	Provider	MI300X templates	MI300X GPUs	Context window
Gemma 4 E4B	Google	128k-amd-mi300x	1	128K
GLM-5.3 Flash	Z.ai	512k-amd-4xmi300x 512k-tp2-dp4-amd-8xmi300x 512k-tp4-dp2-amd-8xmi300x	4 or 8	512K
GLM-5.2 FP8	Z.ai	512k-amd-8xmi300x	8	512K
DeepSeek V4.1 Flash	DeepSeek	512k-amd-4xmi300x	4	512K
DeepSeek V4.1 Flash (1017)	DeepSeek	1m-amd-4xmi300x 1m-amd-8xmi300x	4 or 8	1M
Qwen3.8 27B	Qwen	64k-amd-mi300x 64k-amd-2xmi300x	1 or 2	64K
Granite 4.0 H Small	IBM	128k-amd-mi300x	1	128K
Granite 4.0 H Tiny	IBM	128k-amd-mi300x	1	128K
Phi-4	Microsoft	16k-amd-mi300x	1	16K
Fara1.5 27B	Microsoft	256k-amd-mi300x	1	256K
Mistral NeMo Instruct FP8	Mistral AI	128k-amd-mi300x	1	128K
Text embedding models (BGE, E5, GTE, Arctic Embed and others)	Various	amd-mi300x	1	Not applicable

Selected from the MI300X Managed Compute templates Microsoft listed as of November 4th, 2026. Template names are shortened to the part after the model name, so the full name of the gpt-oss-120b template is openai--gpt-oss-120b--128k-amd-mi300x.



The Deployment Experience

Standing up a private model endpoint took four choices and less than half an hour, or under an hour for the one-million-token DeepSeek V4.1 Flash (1017) template, with no container setup or infrastructure orchestration.

- **Step 1:** Select an open model from the Microsoft Foundry catalog.
- **Step 2:** Choose Managed Compute and the matching MI300X template.
- **Step 3:** Set the capacity, either one instance or several behind a single endpoint.
- **Step 4:** Deploy, then point applications at the endpoint with a standard OpenAI SDK.

The same steps can be automated with a single Azure REST call that names the model, the template, the accelerator type and the instance count, and deleting a deployment is a single call as well.

© 2026 Signal65. All rights reserved.

During provisioning, the service allocates the GPUs, loads the model weights, starts the serving runtime and runs health checks before the endpoint accepts traffic. Time from deployment to a ready endpoint was consistent for each model, and longer for the four-GPU, one-million-token DeepSeek V4.1 Flash (1017) template.

Model	MI300X GPUs	Deployments measured	Deploy to ready
gpt-oss-120b	1	5	14 to 21 minutes
Gemma 4 31B	1	2	14 to 21 minutes
GLM-5.3 Flash	4	1	24 minutes
DeepSeek V4.1 Flash (1017)	4	4	46 to 59 minutes

Teardown took two to eight minutes, with a median of about five. Billed hours matched the time from a ready endpoint to the delete request, and failed deployments were not billed. Because capacity comes and goes on demand, a team can deploy, test and remove several candidate models in a single day and pay only for the hours each one runs.



What the Templates Manage

Serving a large language model well depends on many interacting settings, and the Foundry templates make the decisions that would otherwise fall to a dedicated AI infrastructure team. We used every template exactly as provided and changed nothing on the serving side.

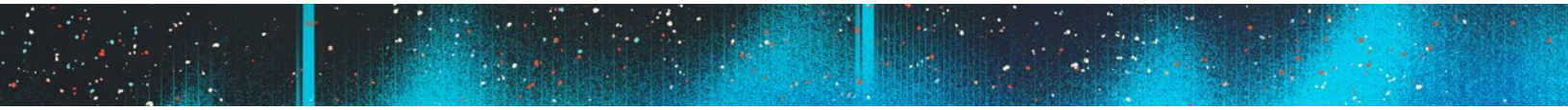
Area	Self-managed infrastructure	Foundry Managed Compute
Hardware	Procurement, installation, power and cooling	Provisioned on demand in minutes
Drivers and runtimes	Matching GPU drivers, libraries, containers and serving software	Curated and maintained by Microsoft
Attention kernels	Chosen and tested per model	Selected per model in the template
Memory and context	KV-cache precision, memory ratios and served context length set by hand	Preconfigured per template
Parallelism	GPU placement, tensor parallelism and CPU pinning	Set per template, from one to eight GPUs
Output parsing	Tool-call and reasoning parsers configured and maintained	Built into the endpoint
Health	Monitoring hardware and replacing faulty components	Operated by Microsoft

© 2026 Signal65. All rights reserved.

To build a fair comparison, Signal65 engineers stood up the same models on MI300X servers in our lab. Matching the managed endpoints took specialists several rounds of setup, measurement and repair.

- **Kernel selection:** No single attention backend suited every model. AMD's AITER attention kernels were the right choice for gpt-oss-120b, while Gemma 4 ran faster on the default backend, and on one model the wrong backend decoded long prompts up to about 6x more slowly. We found the right combination for each model only by measuring it under load.
- **Memory and context sizing:** Our GLM-5.3 Flash configuration served a 262,144-token context, and under heavy agent load one long-running agent outgrew it. The Foundry template served the same workload without cutting off any agent.
- **Ongoing software maintenance:** When we added KV-cache offloading to CPU memory and NVMe to support more sessions, an upstream software defect crashed the server at the end of heavy runs. We had to diagnose it, build patched software images and revalidate before continuing.

Teams that run GPU inference full time handle this work routinely, but most enterprises do not have that team. On Foundry, the templates handled each of these choices for us.



Users Supported by Workload

We measured how many agents each model supports with Signal65 PINNACLE, which replays real multi-turn agent sessions, complete with tool calls, at rising levels of concurrency. Each agent is an AI agent, a software process that works through a multi-step task by calling the model and its tools in a loop, not a human user. A load level counts only if the median per-agent decode speed stays at or above 10 tokens per second and the 95th-percentile time to first token stays at or under 20 seconds. Beyond that point, we no longer consider the experience usable.

Model	Suited to	MI300X GPUs	Agents supported	Per-agent decode at that load	First token, 95th percentile
DeepSeek V4.1 Flash (1017)	Long-context agents, large codebases	4	128	13.5 tok/s	1.6 s
GLM-5.3 Flash	Agent fleets, high-volume chat, document analysis	4	96	13.7 tok/s	2.1 s
gpt-oss-120b	Coding agents, software engineering, tool execution	1	32	21.9 tok/s	0.75 s
Gemma 4 31B (256K context)	Departmental assistants, summarization, internal search	1	8	20.3 tok/s	7.6 s

We ran the same agentic workload on every model so the results compare directly. The "Suited to" column reflects how each model is typically deployed, and the conversational tests below add a second, shorter workload.

© 2026 Signal65. All rights reserved.

GLM-5.3 Flash, deployed across four GPUs, supported 96 concurrent agents. On a single GPU, gpt-oss-120b supported 32 agents, which makes it an efficient starting point for coding assistants. DeepSeek V4.1 Flash (1017), on four GPUs with a one-million-token context, supported 128 agents, the most of any model we tested.

Gemma 4 31B supported 8 agents on one GPU, the same limit we reached on our own tuned system. We tested it with its 256K-token context template. Gemma 4 31B is a dense model that runs every token through all of its parameters, while gpt-oss-120b, a mixture-of-experts model, activates only a fraction of its parameters for each token, so Gemma 4 31B does more work for every token it generates. Signal65 PINNACLE agent sessions grow past 100,000 tokens of context over dozens of rounds, which makes this a demanding workload for any model on a single GPU. On short-prompt conversational work, Gemma 4 31B served more throughput than gpt-oss-120b on the same single GPU, 1,065 versus 903 output tokens per second at 128 concurrent requests.

With a single agent in PINNACLE, decode ran at roughly 40 to 90 tokens per second across the four models, well above reading speed. We also ran a second suite of MLCommons-style tests with fixed chat, retrieval and code request shapes.

The table below shows the Foundry results for each model on those three request shapes, at the highest concurrency where both Foundry and our self-hosted systems were measured.

Model	Workload shape	Input / output tokens	Concurrency	Output tok/s	p95 first token	Share of self-hosted
gpt-oss-120b	Chat	1K / 400	128	903	6.85 s	89%
gpt-oss-120b	Retrieval	8K / 400	45	180	13.37 s	100%
gpt-oss-120b	Code	10K / 1K	32	410	14.67 s	97%
Gemma 4 31B	Chat	1K / 400	128	1,065	15.19 s	110%
Gemma 4 31B	Retrieval	8K / 400	8	152	10.38 s	101%
Gemma 4 31B	Code	4K / 1K	32	586	17.95 s	105%
GLM-5.3 Flash	Chat	1K / 400	256	3,397	6.65 s	92%
GLM-5.3 Flash	Retrieval	8K / 400	23	483	9.79 s	83%
GLM-5.3 Flash	Code	4K / 1K	64	1,663	11.03 s	101%
DeepSeek V4.1 Flash (1017)	Chat	1K / 400	256	2,244	14.70 s	97%
DeepSeek V4.1 Flash (1017)	Retrieval	8K / 400	32	434	19.87 s	95%
DeepSeek V4.1 Flash (1017)	Code	4K / 1K	32	912	10.27 s	98%

Share of self-hosted compares aggregate output tokens per second at the same concurrency. The gpt-oss-120b code row uses the 10K-token code shape, because its 4K-token code points on Foundry ran without randomized prompt prefixes and have no comparable self-hosted counterpart.

© 2026 Signal65. All rights reserved.

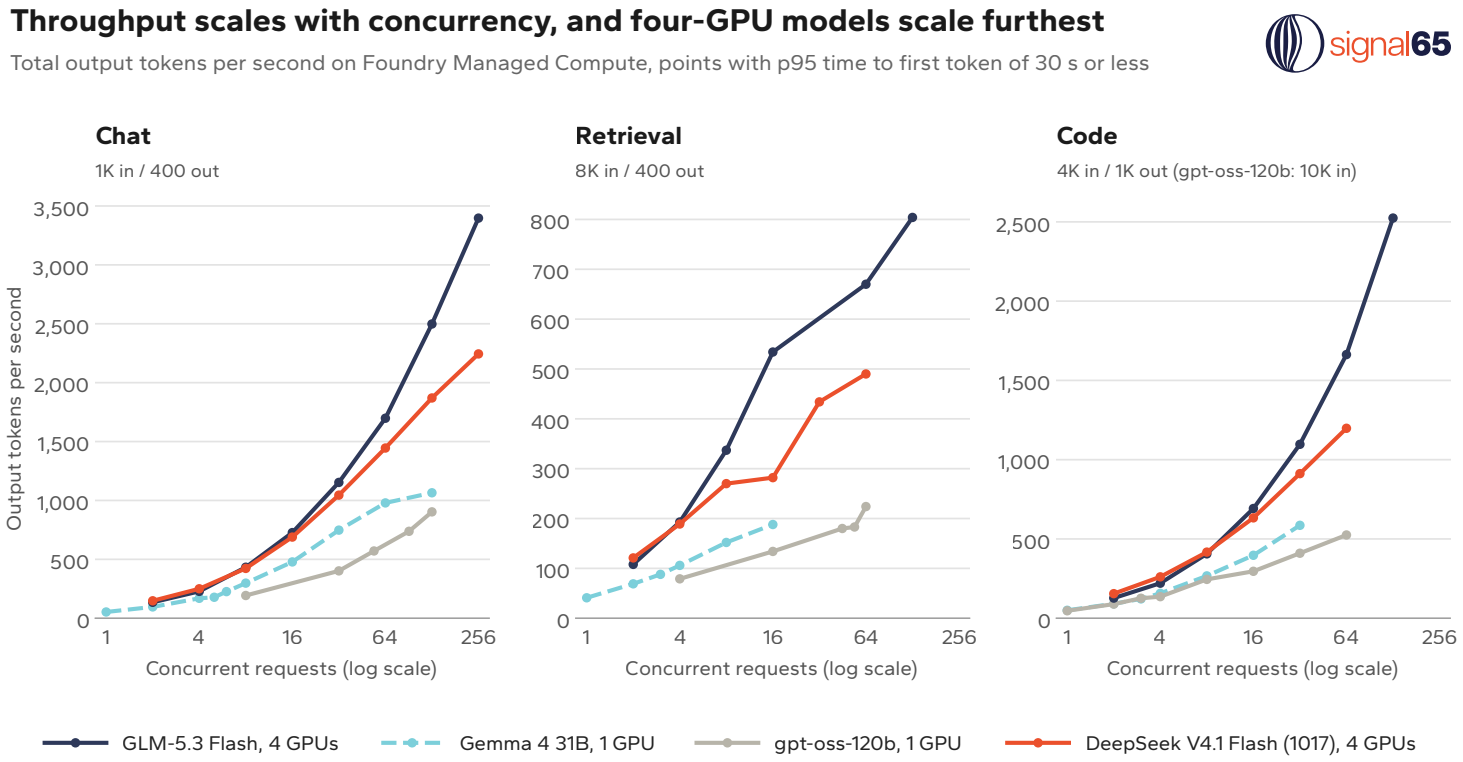


Figure 1. *Total output tokens per second against concurrency on Foundry Managed Compute for the chat, retrieval and code shapes, by model. Points shown have a p95 time to first token of 30 seconds or less. The gpt-oss-120b code line uses 10K-token prompts. GLM-5.3 Flash points ran without randomized prompt prefixes, and its retrieval point at 32 concurrent requests is omitted because it benefited from prompt-cache reuse.*

Model choice depends on the task, the context it needs, the latency users expect and the number of users or agents to serve. Because each model is a separate, on-demand deployment, a team can run several side by side and route each workload to the model that suits it.



Managed Versus Self-Hosted: The Same Models on the Same Hardware

Ease of deployment matters only if a managed endpoint performs to within a reasonable measure of what a skilled team could build by hand. To measure that gap, we ran the same models and workloads on MI300X servers in Signal65's lab. Each model used the same number of GPUs as its Foundry template, and our engineers tuned the serving settings for each one.

At full agent load, Foundry's default templates delivered 93% to 100% of the per-agent speed of our tuned systems.

© 2026 Signal65. All rights reserved.

At full agent load, Foundry reached 93% to 100% of tuned self-hosted speed

Foundry per-agent decode speed as % of the same model self-hosted on the same MI300X GPU count

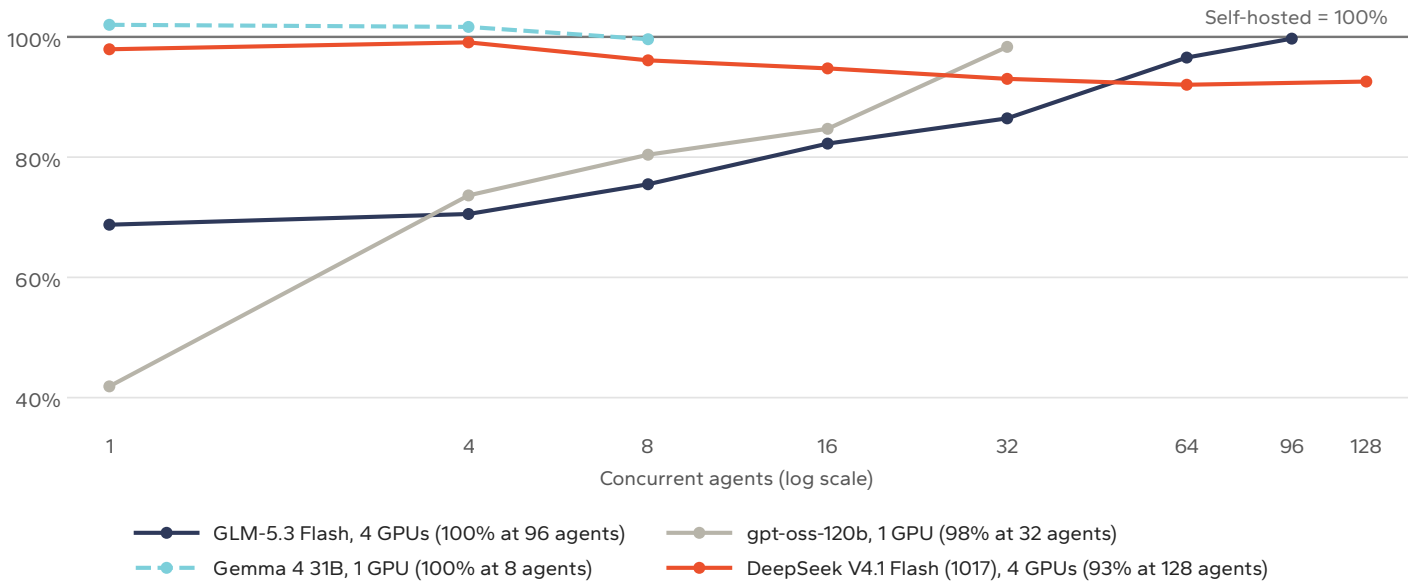


Figure 2. *Foundry per-agent decode speed as a share of self-hosted speed on the same MI300X GPU count, by number of concurrent agents, measured with Signal65 PINNACLE.*

Model	MI300X GPUs	Full-load rung	Foundry aggregate	Self-hosted aggregate
DeepSeek V4.1 Flash (1017)	4	128 agents	1,723 tok/s	1,862 tok/s
GLM-5.3 Flash	4	96 agents	1,316 tok/s	1,320 tok/s
gpt-oss-120b	1	32 agents	701 tok/s	713 tok/s
Gemma 4 31B	1	8 agents	162 tok/s	163 tok/s

gpt-oss-120b and Gemma 4 31B supported the same number of agents on Foundry as on our own systems. GLM-5.3 Flash ran at the same per-agent speed at 96 agents in both environments, although our self-hosted 96-agent run does not count as a pass because one agent outgrew the context we had configured. The Foundry template did not hit that limit. DeepSeek V4.1 Flash (1017) supported 128 agents on Foundry at 93% of our self-hosted per-agent speed; our self-hosted testing of that build ended at 128 agents.

Hand-tuning helped most at light load. With one to a few agents, our tuned systems decoded faster per user on gpt-oss-120b and GLM-5.3 Flash, while Gemma 4 31B matched at every load. The gap narrowed steadily as load rose, and the same pattern appeared in our MLCommons-style tests, where Foundry's short-prompt chat throughput at high concurrency was 89% to 110% of self-hosted. DeepSeek V4.1 Flash (1017) was the exception, matching our tuned system at light load and trailing it by 7% to 8% at 64 to 128 agents.

Most enterprises size deployments for production load. At that load, Foundry's managed templates matched or came within 7% of the results of our engineers' tuning, with no tuning effort from the customer.

© 2026 Signal65. All rights reserved.

Outcomes per Dollar

Foundry Managed Compute billed MI300X capacity at $7.91 per GPU-hour during our testing, the pay-as-you-go rate with no discounts applied, and with no up-front commitment. Billing stops when a deployment is deleted, so teams pay only for the hours they use.

A $10,000 budget buys about 1,264 hours of dedicated MI300X time. For development work, counting a 160-hour working month, that funds about 250 agent working-months of gpt-oss-120b agents, or about 190 agent working-months on a 96-agent GLM-5.3 Flash fleet. For production, it keeps one GPU of gpt-oss-120b serving 32 agents around the clock for about seven and a half weeks, or a 96-agent GLM-5.3 Flash fleet running for about two weeks.

Model	Continuous service per $10,000	Agent working-months per $10,000	Agent sessions per $10,000
gpt-oss-120b	About 7.5 weeks on one GPU	About 250	About 112,000
GLM-5.3 Flash	About 2 weeks on four GPUs	About 190	About 63,000
DeepSeek V4.1 Flash (1017)	About 2 weeks on four GPUs	About 250	About 87,000
Gemma 4 31B (256K context)	About 7.5 weeks on one GPU	About 63	About 23,000

At the agent loads measured in Users Supported by Workload, the $7.91 rate works out to the per-agent costs below. A complete agent session in our tests averaged 1.4 to 2.1 million input tokens and 23,000 to 32,000 output tokens.

Model	MI300X GPUs	Cost per hour	Agents supported	Cost per agent-hour	Cost per agent session
gpt-oss-120b	1	$7.91	32	$0.25	$0.09
GLM-5.3 Flash	4	$31.64	96	$0.33	$0.16
DeepSeek V4.1 Flash (1017)	4	$31.64	128	$0.25	$0.12
Gemma 4 31B (256K context)	1	$7.91	8	$0.99	$0.43

Evaluation is inexpensive as well. Because billing runs only while a deployment is live, a team can take a candidate model from first deployment through full load testing for a few hundred dollars of GPU time.

© 2026 Signal65. All rights reserved.

Conclusions and Next Steps

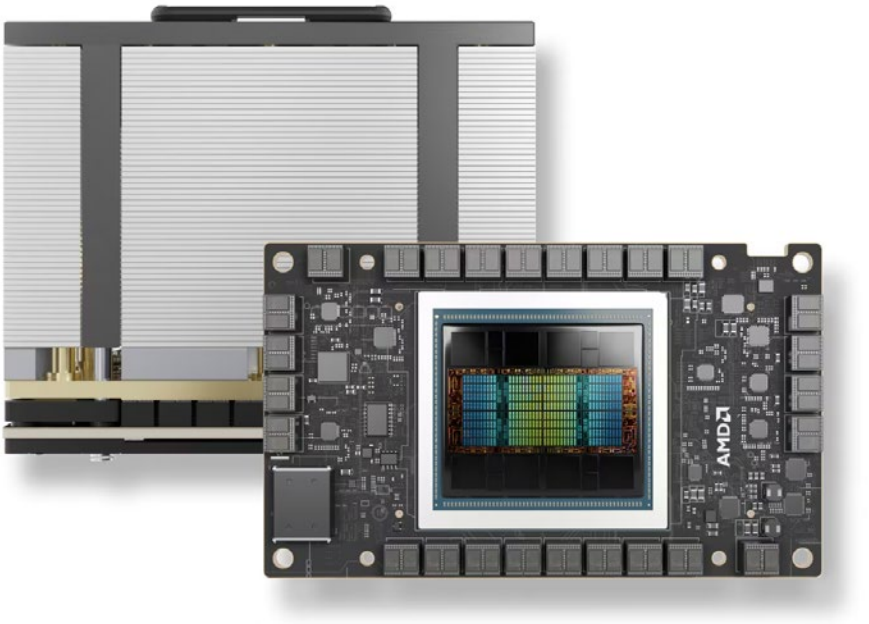
Microsoft Foundry Managed Compute on MI300X GPUs turns dedicated, private model serving into an on-demand service. In our testing, endpoints were live in under half an hour, or under an hour for DeepSeek V4.1 Flash (1017) at one-million-token context. They performed at production load like systems our own engineers had tuned and needed no AI infrastructure team to build or run them. The curated templates absorb kernel selection, memory sizing and multi-GPU layout, and the MI300X's large memory supports dense, high-concurrency serving on few GPUs.

What this means for buyers

For many enterprises, the hardest part of private AI has been the infrastructure work that comes before the first useful answer. OEM systems and validated solutions shorten that work for teams that run their own hardware, and Foundry Managed Compute on MI300X GPUs takes nearly all of it off the customer. Our testing showed that the managed templates gave up little or nothing at production load against a serving stack our own engineers tuned by hand.

The service fits best where dedicated capacity and a fixed model version matter, including regulated industries, internal knowledge and agent platforms with steady demand, and development teams that want to evaluate several open models before standardizing on one. Teams with spiky or experimental usage may still start on serverless APIs, and organizations with large, steady fleets and in-house GPU expertise may still choose to own hardware. For the broad group of enterprises between those two cases, managed compute delivers dedicated, private AI capacity without building a serving team first.

AMD deserves credit for its part in these results. The 192 GB of HBM3 on each MI300X GPU is what lets a model the size of gpt-oss-120b serve dozens of concurrent agents on a single GPU. AMD's work across the ROCm software stack, upstream vLLM support for Instinct GPUs and optimized kernels such as AITER underpins every MI300X deployment in Foundry. Making a GPU a first-class option inside a managed service, with templates that hold up at production load without customer tuning, takes sustained engineering across hardware, software and the cloud partnership.



In an AI rollout, the longest delays usually come before the first production workload, while teams wait on hardware, hire specialists and tune software. Foundry Managed Compute on MI300X GPUs removes most of that wait, lets a team test several open models side by side on dedicated capacity, and scales the chosen model to production on the same platform.

© 2026 Signal65. All rights reserved.

Getting started

For teams getting started, we recommend the following sequence.

- **Choose a model:** Select by workload, context requirements and expected concurrency.
- **Deploy the matching MI300X template:** Use the Foundry portal or a single REST call.
- **Connect existing applications:** Point applications built on OpenAI-compatible SDKs at the private endpoint.
- **Profile real traffic:** Confirm latency and throughput against business needs.
- **Scale with demand:** Add instances at peak and delete idle deployments to control spend.

Before moving to production, confirm data-residency requirements, model fit, expected concurrency, latency targets and projected utilization. Then start with one model matched to a clear workload, measure it against real traffic and grow from there.

© 2026 Signal65. All rights reserved.

Appendix: Methodology

- **Foundry setup:** Models were deployed from the Foundry catalog on MI300X Global Managed Compute, using each template as provided. Load came from four Azure D16ads_v6 client VMs in West US 2, the region of the Foundry account.
- **Self-hosted setup:** An 8-GPU MI300X server in the Signal65 lab. Each model ran on the same number of GPUs as its Foundry template, with an open-source serving stack whose settings Signal65 engineers tuned per model.
- **Workloads:** Both environments used the same tools, datasets and pass rules.
    - **Signal65 PINNACLE:** Replays recorded multi-turn agent sessions, with tool calls and conversations that grow beyond 100,000 tokens, at rising concurrency. A load level passes when the median per-agent decode speed is at least 10 tokens per second and the 95th-percentile time to first token is at most 20 seconds. A run in which an agent exceeds the model's served context gets no verdict.
    - **MLCommons-style tests:** Fixed request shapes covering chat, retrieval and code, with prompts up to 32K tokens, run for five minutes per concurrency level. Points used randomized prompt prefixes to prevent artificial cache reuse, except those for GLM-5.3 Flash, which may therefore read slightly optimistic.
- **Scope:** Signal65 evaluated additional models that are not reported here. Their results are excluded pending provider-side updates and may be published separately.
- **Limits:** Most results come from single runs, so differences of a few percentage points are not significant. Foundry does not expose its serving stack, so we report measured behavior only and do not attribute differences to specific software.
- **Disclosure:** This study was commissioned by Microsoft. Signal65 designed and ran the tests and wrote this report independently.

© 2026 Signal65. All rights reserved.

Important Information About this Report

CONTRIBUTORS

Russ Fellows
VP, Labs | Signal65

Ryan Shrout
President and GM | Signal65

PUBLISHER

Ryan Shrout
President and GM | Signal65

INQUIRIES

Contact us if you would like to discuss this report and Signal65 will respond promptly.

CITATIONS

This paper can be cited by accredited press and analysts, but must be cited in-context, displaying author's name, author's title, and "Signal65." Non-press and non-analysts must receive prior written permission by Signal65 for any citations.

LICENSING

This document, including any supporting materials, is owned by Signal65. This publication may not be reproduced, distributed, or shared in any form without the prior written permission of Signal65.

DISCLOSURES

Signal65 provides research, analysis, advising, and lab services to many high-tech companies, including those mentioned in this paper.

ABOUT SIGNAL65

Signal65 exists to be a source of data in a world where technology markets and product landscapes create complex and distorted views of product truth. We strive to provide honest and comprehensive feedback and analysis for our clients in order for them to better understand their own competitive positioning and create optimal opportunities to market and message their devices and services.

COMMISSIONED BY

Microsoft

CONTACT INFORMATION
Signal65 | signal65.com

© 2026 Signal65. All rights reserved.