Cnuas Virtual AI HPC Infrastructure Datasheet

Full platform reference, switches, NICs, GPUs, fabric and host requirements

Document DS-CNU-001 • **Revision** D • **Issued** 12 August 2026 • **Status** Published

Item	Value
Part	Cnuas Virtual AI HPC Infrastructure
Type	Software-emulated virtual AI and HPC infrastructure
Version	Platform 2026.07 (`9fbdddf-dirty`)
Status	Active development
Product	cnuas.io

1. Overview

Cnuas (Irish *Gaeilge* for ”cluster”) is the **Cnuas Virtual AI HPC Infrastructure**, a 100% software-emulated virtual AI and HPC infrastructure for teaching and developing HPC, AI, and RDMA networking (RoCEv2 and native InfiniBand) without physical GPU clusters or InfiniBand switches. Every device is a QEMU/userspace model presented to unmodified guest software, so a virtual GPU or NIC behaves like the real part it emulates. Cnuas is developed under the internal **Cnuas** engineering programme.

It composes four emulated device families, CnuasSwitch (TOR fabric switch), CnuasNIC (RDMA NIC), CnuasGPU (SIMT accelerator), and CnuasLink (GPU fabric), over a forked QEMU and a forked Linux kernel, orchestrated as VMs on a single Ubuntu management host.

Components usable without a guest

The full rack runs as virtual machines, but three of the four device families are also usable on their own, as host processes with no hypervisor, no guest and no kernel module. This matters when only one part is wanted, for example an accelerator inside an application, or the fabric alone for protocol work.

Component	Usable with no guest	Form
CnuasGPU	Yes	Soft-GPU in the calling process, including a ggml backend for ggml based applications
CnuasSwitch	Yes	`cnuas-vswitchd` host daemon; ports are UNIX sockets any host process can attach to
CnuasLink	Yes	`cnuasgpu-link-switchd` host daemon
CnuasNIC	No	Needs a guest, because the drivers bind an emulated PCI function and the provider uses the kernel `ib_uverbs` ABI

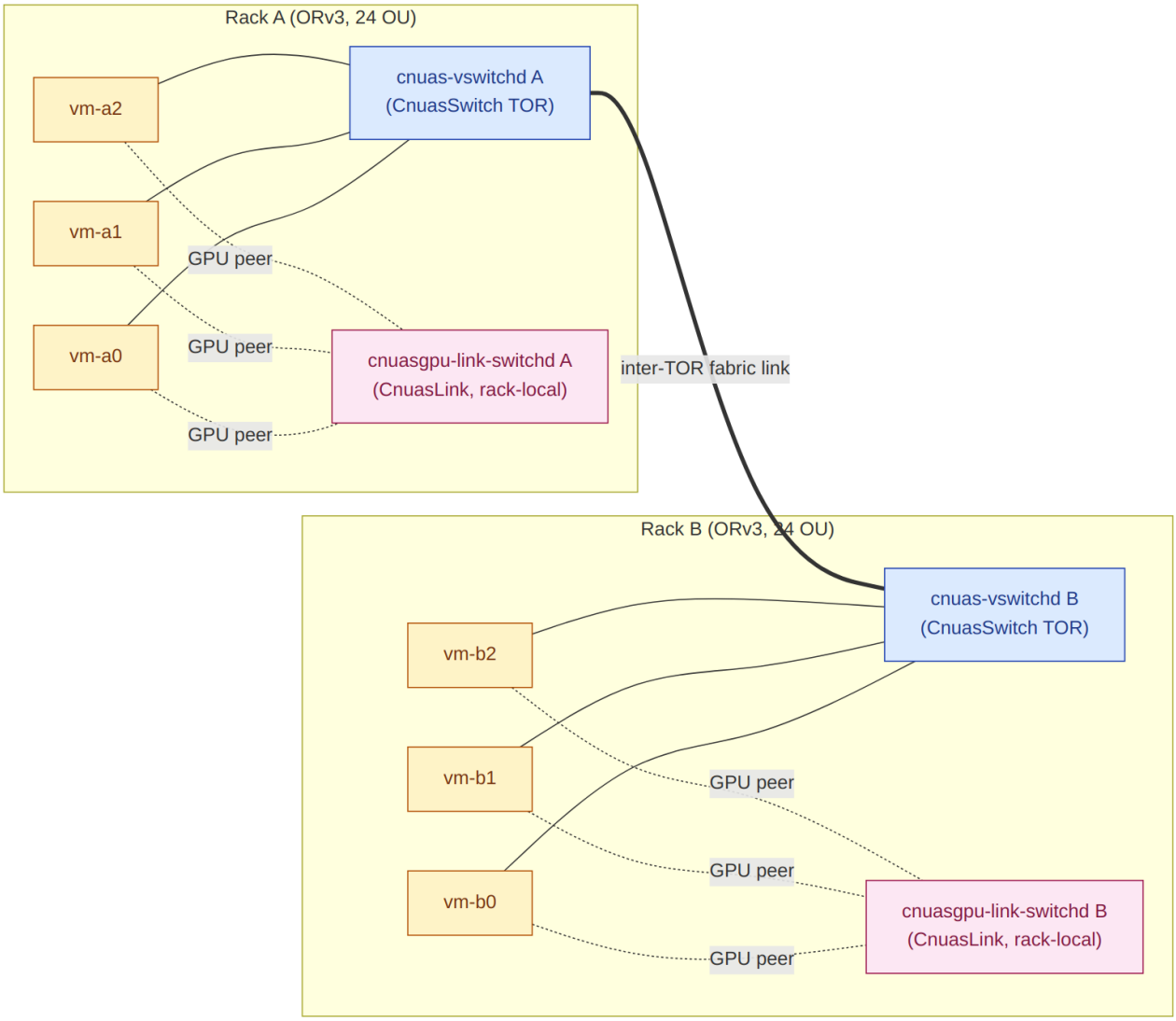
Key features

- Full RDMA fabric, RoCEv2 **and** native InfiniBand, end to end.
- SIMT virtual GPUs with a CUDA-style runtime and multi-architecture SIMD backends.
- Separate GPU-to-GPU fabric (CnuasLink) modelling an NVSwitch/NVLink-class interconnect.
- In-switch Subnet Manager, the fabric comes up without external `opensm`.
- Runs on a single commodity x86-64 host; no special hardware required.

- Consistent management plane spanning CLIs, JSON control sockets, and a browser dashboard.
- Multi-rack topology modelling on the Open Compute Project Open Rack v3 reference.

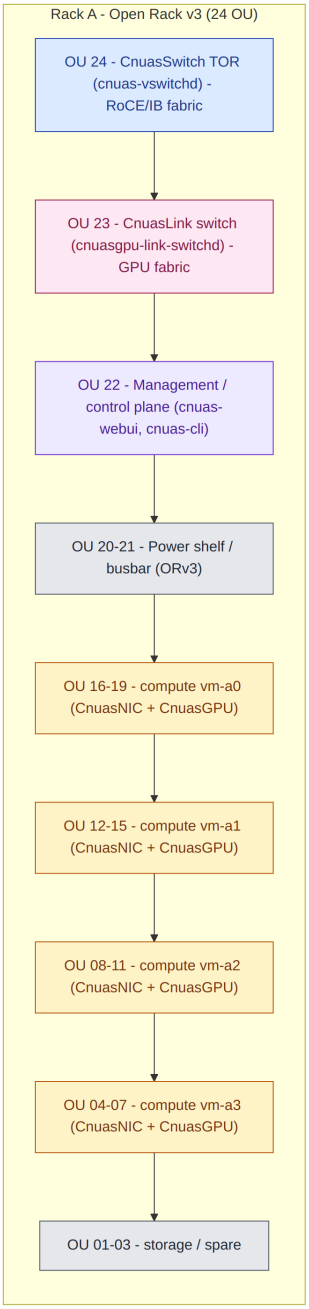
2. Rack-level block diagram

Two-rack reference deployment. Each rack has a CnuasSwitch top-of-rack (TOR) for the RoCE/IB fabric and a rack-local CnuasLink switch for the GPU fabric. VMs attach to both fabrics; racks are joined by an inter-TOR link.



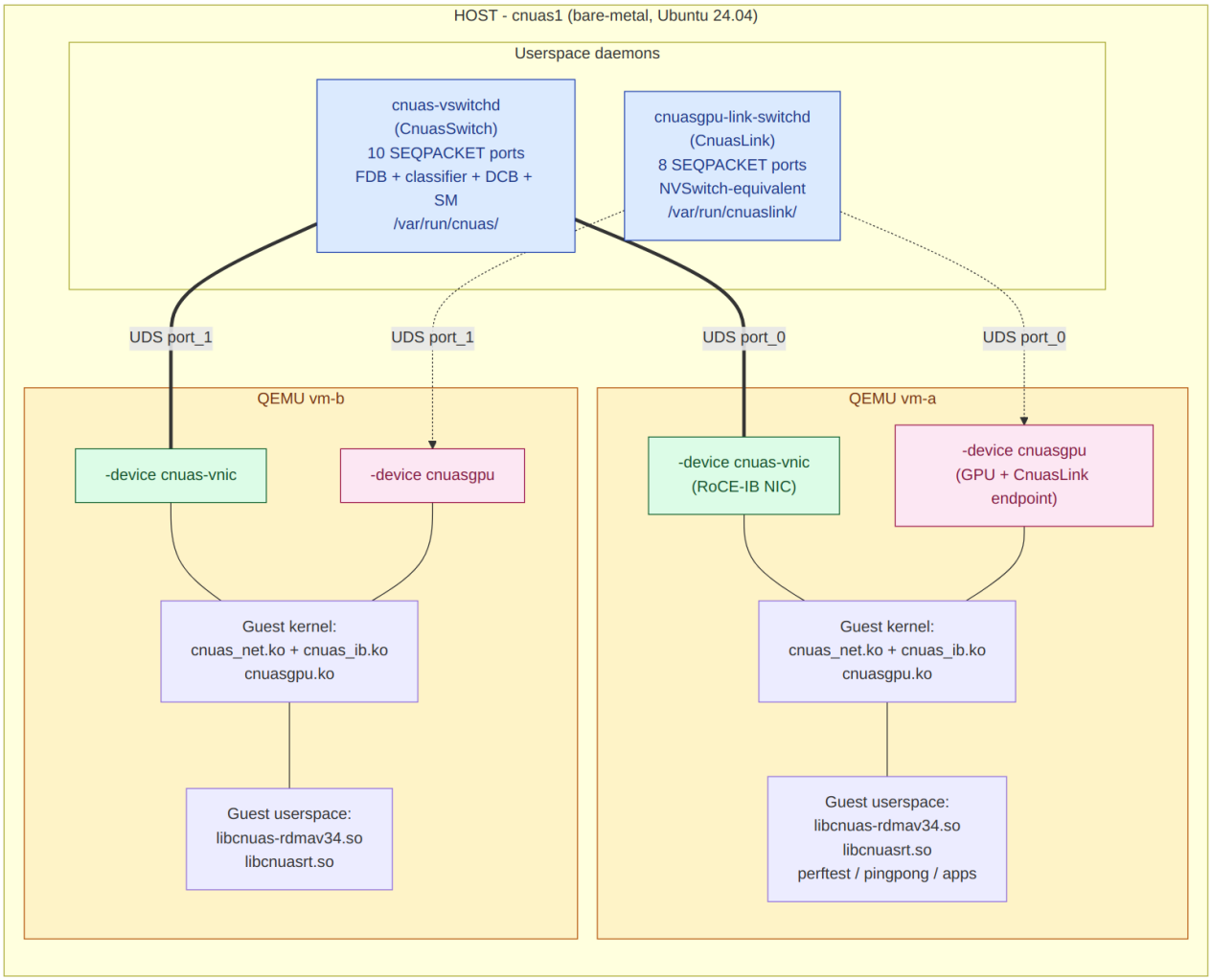
2.1 Rack elevation (ORv3, 24 OU)

Logical OU (OpenU) slot assignment for a single rack. CnuasSwitch (TOR) and the rack-local CnuasLink switch occupy the top OUs; compute VMs fill the remainder. Two such racks form the reference deployment above.



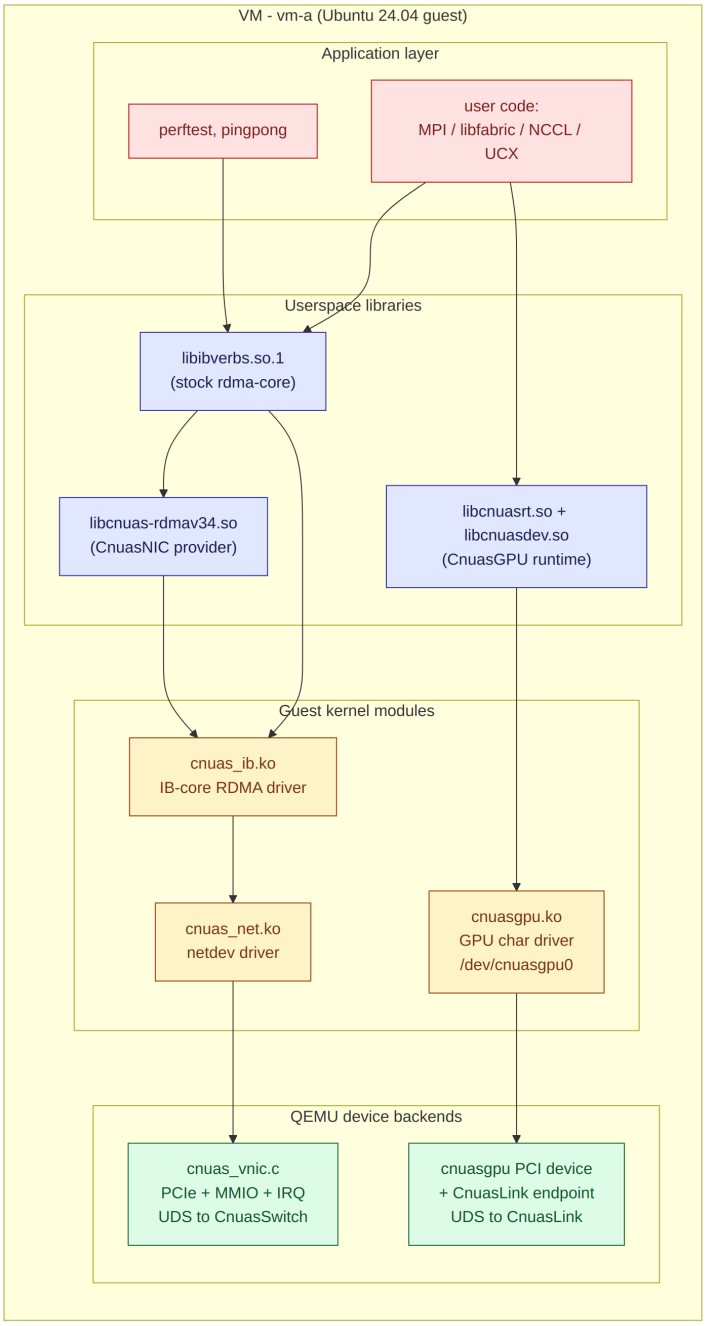
3. Host / node block diagram

Inside a single physical host, the switch daemons run in host userspace; each VM is a QEMU process exposing a `cnuas-vnic` (NIC) and a `cnuasgpu` (GPU) PCIe device. Devices connect to the daemons over UNIX-domain sockets (UDS).



4. Guest software stack

Per-VM stack, from application down to the QEMU device backends. The RDMA path uses stock `libibverbs`, which `dlopen()`s the CnuasNIC provider; the GPU path uses `libcnuasrt` over the `cnuasgpu.ko` char device.



5. Platform specifications

Parameter	Value
Emulation model	QEMU custom PCIe devices + userspace switch daemons
Host OS	Ubuntu 24.04 LTS (x86-64)
Guest kernel	Linux fork `cnuas-v6.19.0` (6.19.0-cnuas+)
Hypervisor	QEMU fork `cnuas-qemu-v0.1.0`
Inter-VM data transport	UNIX domain sockets (`SOCK_SEQPACKET`); ivshmem shared memory (GPU fast path)
Management transport	JSON over UNIX/TCP sockets
Reference topology	Open Compute Project Open Rack v3 (ORv3), up to 24 OU

Parameter	Value
Fabric switch ports	10 per CnuasSwitch (8 fabric + uplink + console)
GPU fabric ports	8 per CnuasLink (configurable, max 32)
Build system	Bazel (daemons/CLI/WebUI/packaging), Make (kernel, QEMU)
Languages	C (GNU C, daemons + kernel), Python 3.11+ (tooling), C++ (QEMU models)
Versioning	Semantic versioning per component (`cnuas-vMAJOR.MINOR.PATCH`)

6. Component summary

Component	Part / daemon	Kernel	Userspace	Datasheet
Fabric switch	`cnuas-vswitchd`	, (host daemon)	`cnuas-cli`, WebUI	CnuasSwitch
RDMA NIC	`cnuas-vnic`	`cnuas_net.ko`, `cnuas_ib.ko`	`libcnuas-rdmav34.so`	CnuasNIC
GPU	`cnuasgpu`	`cnuasgpu.ko` (`/dev/cnuasgpuN`)	`libcnuasrt.so`, `libcnuasdev.so`, `cnuassmi`	CnuasGPU
GPU fabric	`cnuasgpu-link-switchd`	, (host daemon)	`cnuaslink-cli`	CnuasLink

6.1 Component versions

Live from each submodule's git tag at documentation build time:

@@VERSION_TABLE@@

7. Management and control plane

Interface	Description
`cnuas-cli`	Switch/fabric management CLI (port status, SM, DCB)
`cnuaslink-cli`	CnuasLink GPU fabric management CLI
`cnuassmi` / `cnuas-smi`	GPU inventory and telemetry (nvidia-smi style)
`cnuas-tools`	VM lifecycle: image/package build, `vm up/down/ssh/console/lab`
`cnuas-webui`	Browser dashboard (FastAPI backend + web frontend)
Control sockets	JSON over UNIX socket (`/var/run/cnuas/`, `/var/run/cnuaslink/`)

8. Host requirements

Requirement	Minimum	Recommended
CPU	x86-64 with AVX2	AVX-512 (for CnuasGPU compute)
Memory	16 GB	64 GB+ (scales with VM count)
Virtualization	KVM enabled	KVM + nested virt
OS	Ubuntu 24.04 LTS	Ubuntu 24.04 LTS
Toolchain	GCC (GNU C), Make, Bazel, Python 3.11+, `pytest`	+ `mkdocs-material` for docs

9. Roadmap summary

Feature availability at a glance. Full Epic/Task breakdown, status, and the GitHub Projects board are in the Product Roadmap.

Workstream	Available today	Planned
CnuasNIC (RDMA NIC)	RoCEv2 + native IB, RC/UD/SMI/GSI, atomics, SRQ, multicast, ICRC, in-switch SM interop	SA path-record, PMA/perfquery, opensm compat
CnuasSwitch (fabric)	10-port RoCE/IB, DCB, FDB/LFT, in-switch Subnet Manager, telemetry	Switch-side SA/PMA alignment
CnuasGPU (accelerator)	SIMT runtime, libcnuasrt/libcnuasdev, AVX2/AVX-512 SGEMM, multi-GPU	Backend dlopen split, device-backend refactor, CnuasIR, FPGA
CnuasLink (GPU fabric)	Frame protocol, FDB, discovery, sysfs link state, CLI	Collective offload, bandwidth modelling
Platform / tooling	VM lifecycle CLI, packaging, submodule split, CI	Documentation pass

10. Ordering / integration information

Item	Value
Superproject repo	`PacketFive/cnuas` (git submodule superproject)
Submodules	`linux`, `qemu`, `src/cnuasnic`, `src/cnuasswitch`, `src/cnuasgpu`, `src/cnuaslink`
License	Apache-2.0 (see repository `LICENSE`)
Quick start	`./build_and_test.sh`
Documentation	`https://cnuas.io/docs/`
Project site	`https://cnuas.io`

11. Revision history

Revision	Date	Notes
A	2026-07-05	Initial platform datasheet
B	2026-07-05	Added rack/host/stack diagrams, detailed specs, component driver map
C	2026-07-05	Added rack elevation, live version table, roadmap summary
D	2026-08-12	Recorded which components are usable without a guest