

Cnuas Kernel Modules, Datasheet



Guest kernel modules for the emulated network, RDMA and accelerator devices

Document DS-CNU-013 • **Revision** A • **Issued** 03 August 2026 • **Status** Published

Item	Value
Part	cnuas_net.ko, cnuas_ib.ko, cnuasgpu.ko
Type	Guest Linux kernel drivers
Packages	cnuas-nic-modules, cnuas-gpu-modules
Version	v0.1.0-26-gbc3a3c9 (NIC), v0.2.0-25-gc54883d (GPU)
Repos	PacketFive/CnuasNIC, PacketFive/CnuasGPU

1. Overview

Three out of tree kernel modules bind the emulated PCI devices inside a Cnuas guest. `cnuas_net.ko` presents a network interface, `cnuas_ib.ko` registers an RDMA device with the kernel `ib_core` subsystem, and `cnuasgpu.ko` presents an accelerator through a character device. Together they are the boundary between guest software and the QEMU device models, and they are what makes unmodified RDMA and accelerator software run against emulated parts.

All three build against the Cnuas kernel fork and are shipped as prebuilt modules inside the golden image.

Key features

- Standard subsystem registration, so `ip`, `ethtool`, `ibv_devinfo`, `rdma` and `perftest` work without patches.
- Dual personality RDMA, RoCEv2 over Ethernet or native InfiniBand link layer.
- Reliable connected, unreliable datagram, shared receive queue, multicast and atomic operation support in the RDMA path.
- Direct mapping of accelerator device memory into user space.
- Sysfs attributes for inventory and telemetry collection.

2. Module summary

Module	Version	Licence	Registers with	Node or interface
<code>cnuas_net</code>	0.1.0	Dual MIT/GPL	PCI, netdev, NAPI, ethtool	network interface
<code>cnuas_ib</code>	0.2.0	Dual MIT/GPL	PCI, <code>ib_core</code>	InfiniBand device
<code>cnuasgpu</code>	0.1.0	GPL	PCI, character device	<code>/dev/cnuasgpuN</code>

3. Network module, `cnuas_net.ko`

Parameter	Value
PCI identity	Vendor <code>0x1af4</code> , device <code>0x10f0</code>
Class	Network, other



Parameter	Value
Register window	BAR0, 4 KiB memory mapped
Maximum frame	9216 bytes, interface MTU 9202 bytes
Interrupts	MSI or legacy line, one vector, receive completion and link change
Receive path	NAPI poll
Host transport	UNIX domain socket, SOCK_SEQPACKET

Network device operations

`ndo_open`, `ndo_stop`, `ndo_start_xmit`, `ndo_get_stats64`, `ndo_set_mac_address`, `ndo_validate_addr`.

Ethtool operations

`get_drvinfo`, `get_link`, `get_link_ksettings`, `get_strings`, `get_sset_count`, `get_ethtool_stats`.

4. RDMA module, `cnuas_ib.ko`

`cnuas_ib.ko` layers on the same PCI function as `cnuas_net.ko` and registers an InfiniBand device named `cnuas_ib` with the kernel RDMA core.

4.1 Reported device attributes

Attribute	Value
Firmware version	0x000100
Hardware version	1
Maximum queue pairs	256
Maximum work requests per queue pair	4096
Maximum completion queues	256
Maximum completion queue entries	4096
Maximum memory regions	256
Maximum protection domains	64
Maximum scatter gather entries	16
Outstanding read and atomic operations	16
Atomic capability	Performed by the adapter
Partition keys	1
Multicast groups	256
Queue pairs per multicast group	256
GID table length	1024 in RoCE mode, 1 in native InfiniBand mode
Path MTU	Up to 4096 bytes
Maximum message size	2 GiB less one byte
Reported speed and width	EDR, 4X

4.2 Verbs implemented



Group	Operations
Device and port	query_device, query_port, query_gid, query_pkey, get_link_layer, get_port_immutable
Management	process_mad
Context	alloc_ucontext, dealloc_ucontext
Protection domain	alloc_pd, dealloc_pd
Address handle	create_ah, create_user_ah, destroy_ah
Queue pair	create_qp, modify_qp, destroy_qp, post_send, post_recv
Completion queue	create_cq, destroy_cq, poll_cq, req_notify_cq
Memory region	reg_user_mr, dereg_mr
Shared receive queue	create_srq, modify_srq, query_srq, destroy_srq, post_srq_recv
Multicast	attach_mcast, detach_mcast

4.3 Wire protocol

The host and guest agree on the header layout declared in `cnuas_proto.h`.

Element	Value
Headers	Local route, global route, base transport, RDMA extended, atomic extended, atomic acknowledge extended, datagram extended
RoCEv2 encapsulation	UDP destination port 4791
Global route next header	0x1b
Base transport version	0
Default partition key	0xffff
Datagram multicast destination queue pair	0x00ffffff
Opcodes	Reliable connected send, write, read, acknowledge and atomic; unreliable datagram send

4.4 Link layer selection

The link layer is chosen by module parameter. In Ethernet mode the device reports RoCEv2 with a full GID table. In native InfiniBand mode it reports the InfiniBand link layer, advertises subnet manager, notice and trap port capabilities, and reduces the GID table to a single entry. A single port cannot present both at once in this revision.

No user space management datagram device

Management datagrams are handled inside `process_mad`. There is no `ib_umad` character device, so user space tools that open `/dev/infiniband/umadN`, including a stock `opensm`, cannot attach. Native InfiniBand fabrics are instead managed by the subnet manager built into `CnuasSwitch`. Adding the character device is tracked work and gates kernel upstreaming.

5. Accelerator module, `cnuasgpu.ko`

Parameter	Value
PCI identity	Vendor 0x1af4, device 0x10f1



Parameter	Value
Class	Processing accelerator, co-processor
Register window	BAR0, 64 KiB memory mapped
Device memory	BAR1, RAM backed, 256 MiB by default
Interrupts	MSI, one vector, interrupt pin 1
Character device	/dev/cnuasgpuN
Sysfs class	/sys/class/cnuasgpu/
Sysfs attributes	gpu_id, sm_count, devmem_size, devmem_used, link_up, link_rx_ready
Allocator	Page aligned bump pointer, reclaimed on file descriptor close
User mapping	mmap of BAR1 with write combining

Register map, BAR0

Offset	Register
0x000	Vendor identity
0x004	Device identity
0x008	Revision
0x00c	Firmware version
0x010	Accelerator identity
0x014	Streaming multiprocessor count
0x018	Lanes per multiprocessor
0x01c	Tensor tile size
0x020, 0x024	Device memory size, low and high halves
0x100	Interrupt status
0x104	Interrupt mask
0x200	Fabric link status
0x204, 0x208	Fabric transmit offset, low and high halves
0x20c	Fabric transmit length
0x210	Fabric transmit doorbell
0x214, 0x218	Fabric receive offset, low and high halves
0x21c	Fabric receive buffer size
0x220	Fabric receive length
0x224	Fabric receive consume

The fabric registers at 0x200 and above carry CnuasLink traffic. The device model connects to the link switch daemon over a UNIX domain socket given by the `cnuaslink_socket` property.

6. Validation

Feature	Test
RoCEv2 end to end between two guests	tests/test_rocev2_e2e.py
Native InfiniBand end to end	tests/test_ib_e2e.py
Ethernet and InfiniBand forwarding	tests/test_eth_forwarding.py, tests/test_ib_forwarding.py

Feature	Test
Accelerator information, allocate, copy, free	driver/test/cnuasgpu_smoke.c

Current results are recorded in the Validation Matrix.

7. Integration information

Item	Value
Sources	src/cnuasnic/kernel/, src/cnuasgpu/driver/
Kernel	6.19.0-cnuas, from PacketFive/linux
Build	Kbuild against /lib/modules/\$(uname -r)/build
Packages	cnuas-nic-modules, cnuas-gpu-modules
Guest operating system	Ubuntu 24.04
Emulator	PacketFive/qemu fork

Module and kernel versions are coupled through the compat.json document that ships with every release. See Deployment.

8. Revision history

Revision	Notes
A	First publication. Attributes, verbs, opcodes and register offsets read from the driver sources and headers.

PacketFive, Packet Five Networks Ltd., Dublin, Ireland.

Cnuas Virtual AI/HPC Infrastructure is published at <https://github.com/PacketFive/cnuas> under the Apache License 2.0; read it at <https://github.com/PacketFive/cnuas/blob/main/LICENSE>.

Cnuas Virtual AI/HPC Infrastructure is emulation software. It is not affiliated with, endorsed by, or derived from any hardware vendor, and every device it models is a software artefact. The work is published by its authors in a personal capacity and is not sponsored or endorsed by any employer.

Specifications describe the referenced revision of the software and may change without notice. Contact info@packetfive.com.