

Panthor status update (season 34 episode 26...)

Liviu Dudau, Karunika Choo and Boris Brezillon

XDC 2026



COLLABORA

Open First

Who we are

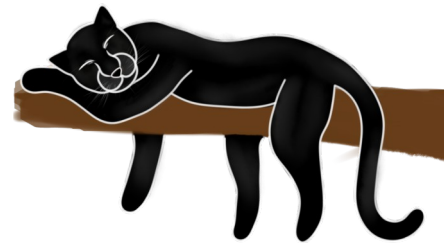
- Liviu Dudau
 - Maintainer for a lot of Arm graphics drivers (display and now GPU)
 - Bridging the company's interests and the upstream community
- Karunika Choo
 - Arm GPU kernel driver developer
 - Contributing to new GPU bring-up and GPU virtualization
- Boris Brezillon
 - Long-term Panfrost, Panthor and PanVK contributor



COLLABORA

Panthor 2026 and beyond

A relatively calm year



- ~170 commits since last XDC
- New generation of GPUs supported: v14 AKA Mali G1
- Relatively few feature added:
 - We now have transparent memory reclaim hooked up
 - Optimized support for the Vulkan sparse binding sink
 - Huge pages support
 - CPU-cached mappings
- The rest is just fixes and minor improvements

Around the corner

- Perf counters (Lukas Zapolskas, Arm)
- Support for protected content (Ketil Johnsen, Arm)
- Enforcement of priorities at the SW-scheduler level
 - Real-time workqueues for contexts with RT priority (Tvrtko Ursulin, Igalia)
 - Panthor needs per-prio workqueues to pass to `drm_sched_init()`
 - Panthor event processing also needs to be directed to the proper workqueue
 - Finer grained locking needed to avoid contention in the `::run_job()` path

Further down the road

- Panthor-specific page table implementation
 - Needed for stage2 MMU (virtualization) support which io-pgtable-arm doesn't support
 - Will let us optimize other things, like large region updates or sparse mappings
- A shmem-based page cache to speed up memory allocation
 - If you're interested or already thought about an implementation design, please reach out to us
- Support for v15 gens is starting to show up on mailing lists



A closer look: how firmware based scheduling works on new GPUs (v15+)

Firmware scheduling

- More groups jobs (CSGs: command stream groups)
 - Firmware exposes upto 4K CSGs
- Panthor can bundle the groups with a ContextID
- Panthor prepares the resources and starts the CSG, but firmware decides when the groups are actually scheduled on the hardware
- When the GPU is virtualized, the whole world can disappear so state needs restoring on re-acquiring the GPU
- Synchronizing state between FW and Panthor becomes more complicated

Firmware scheduling (cont)

- Some registers that used to be associated to particular CSGs have now moved into global arrays that are sparsely populated depending on enabled CSGs
- Sync objects are now allocated from per-VM buffers.
- Firmware uses a small list of active address spaces that needs to be maintained with the help of the kernel driver to match the AS of the running CSGs

Other changes in v15+

- Firmware now reports to the host the version of the CS interface that it supports. This allows for GPU generations to share interfaces or for different classes of GPU from the same generation to have different behaviours.
- Power management decisions are moving towards the virtualization manager that decides when the HW is powered down or at what frequency it is running.



A closer look: how virtualization works on new gens (v15+)

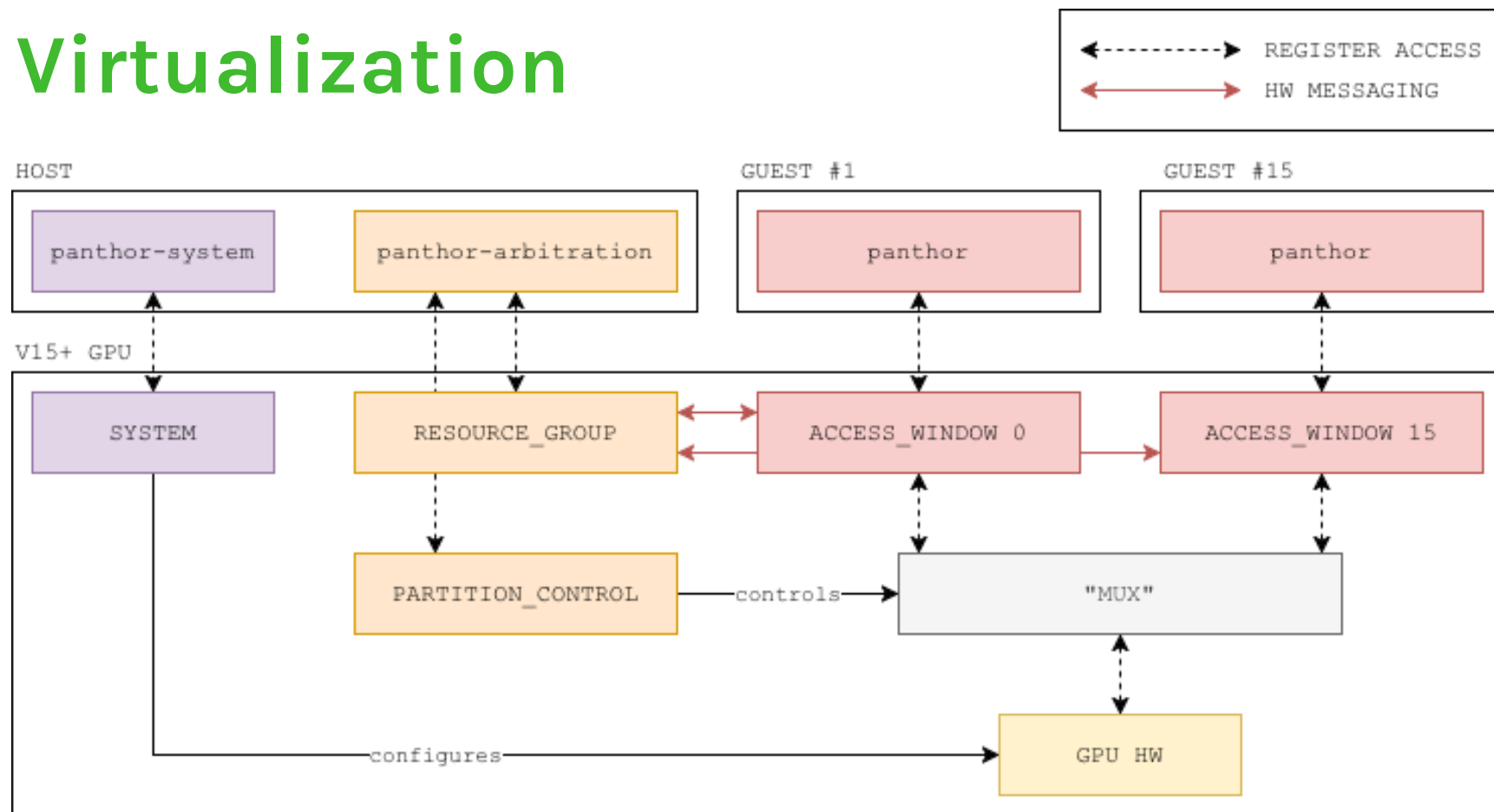
v15 Virtualization

- HW-assisted para-virtualized GPU (G78AE but with CSF)
- GPU provides up to 16 register pages (Access Windows) for job submission (can be assigned to a VM)
- GPU is time-sliced between requesting AWs

v15 Virtualization: SW responsibilities

- Panthor-system
 - GPU-wide configuration.
 - Highest privilege (typically in host / Dom0 / SCP).
- Panthor-arbitration
 - Handles time-slicing of the GPU.
 - 2nd-highest privilege (typically alongside panthor-system).
- Panthor
 - Aware of GPU time-slicing. Mesa is not.
 - Typically in guest OS.

v15 Virtualization



v15 Virtualization: FW-driven handover

- 1) Panthor sends **GPU_REQUEST** to **panthor-arbitration**.
- 2) **panthor-arbitration** assigns GPU to Panthor's AW when available.
- 3) On time-slice expiry, it sends Panthor a **YIELD_NOW** IRQ.
- 4) FW suspends active jobs & yields the GPU
- 5) **panthor-arbitration** assigns the GPU to another requesting AW.
- 6) Repeat until all jobs are complete.

Benefit:

- No need to schedule Panthor to suspend active work

Challenge:

- Panthor must reconcile internal state after FW yields the GPU.



Thank you!
Questions?



COLLABORA

Open First



Backup slides

Shmem-based page cache: experiments/ideas

- Approach 1:
 - Single big file that represents our cache
 - Panthor GEM objects allocated from this cache points to this file and gets assigned a chunk of it (offset+size)
 - Bottom-up allocation with `drm_mm`
 - On free we keep the allocated `drm_mm_node` in an free-tracking `drm_mm` object
 - On reclaim, we walk the free `drm_mm` top-down, remove the free nodes that we reclaim, and `shmem_truncate` (punch-hole) the portions of file that got reclaimed

Shmem-based page cache: experiments/ideas

- Approach 2:
 - Single big file that represents our cache
 - Allocated pages get tracked in an xarray, and have an allocated/free state
 - Panthor GEMs also have an xarray where each entry is an index in the cache xarray
 - When a GEM is freed, we flag the entries it owned in the cache xarray as free
 - On reclaim, we walk the cache xarray and return the free pages to the shmem layer

Shmem-based page cache: pros/cons

- Approach 1 pros:
 - Relatively simple
 - Uses shmem punch-hole feature to deal with free chunks caused by random GEM alloc/free patterns
- Approach 1 cons:
 - Reclaim/alloc granularity is not great



Shmem-based page cache: pros/cons

- Approach 2 pros:
 - Reclaim/alloc at the folio granularity
- Approach 2 cons:
 - Extra indirection: two jumps to get to a page/folio
 - Extra complexity