Jungwoo Kim, Samsung mobile

Advanced Mobile Gaming with Vulkan

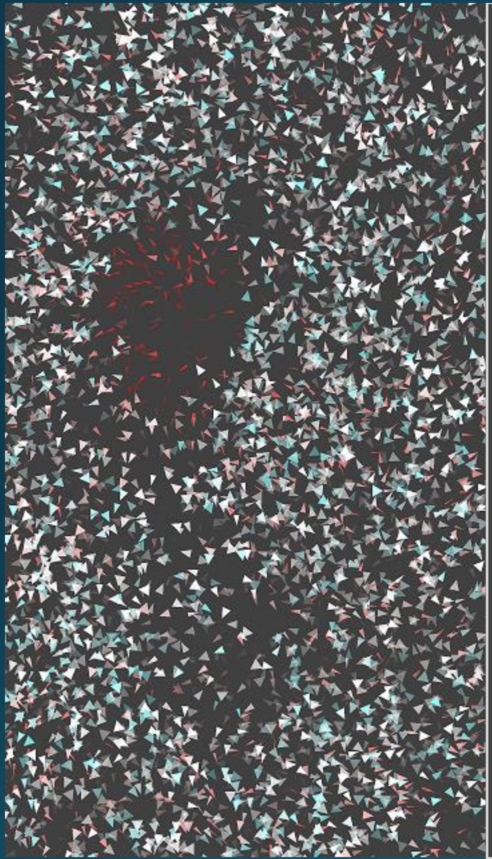
SAMSUNG mobile

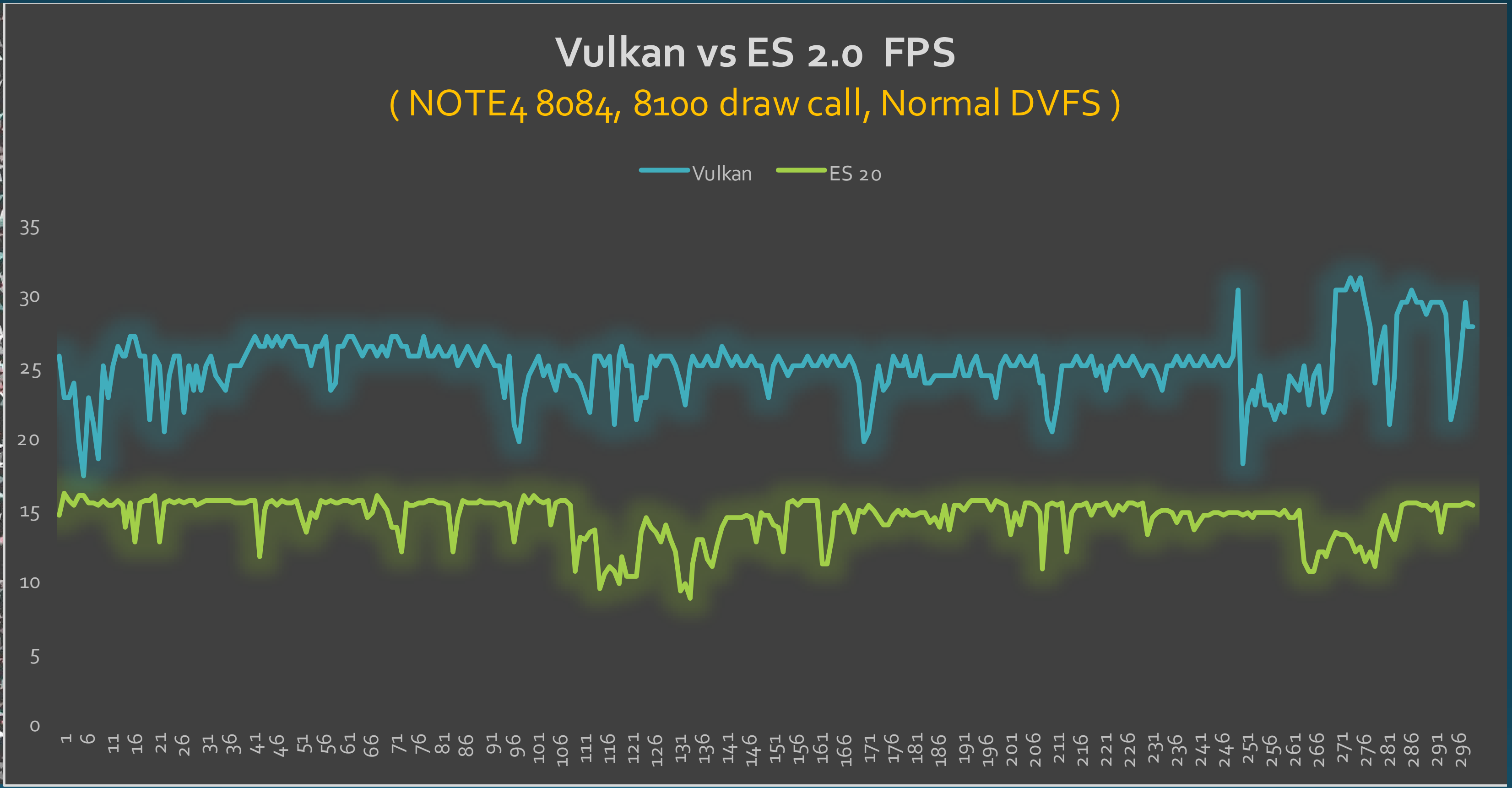
We did several things about Vulkan

- Early studies with corner case examples
- Proof of concept level demo implementation
- Do collaboration for porting real game engine
- Do collaboration for developing cool Vulkan demo
- Developing real Vulkan games for market

Exciting 1 year for Vulkan with Khronos and great game companies!

Early studies : Heavy Drawcalls!!!



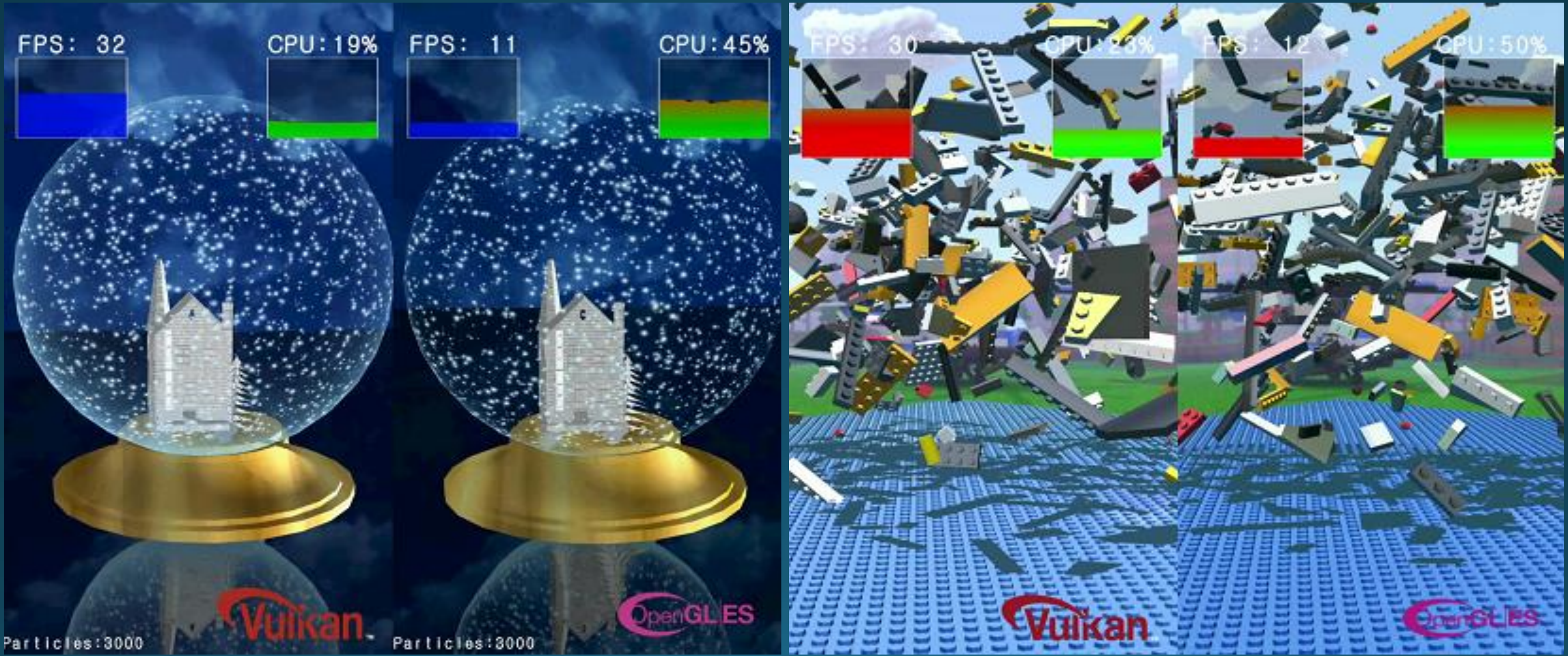


Vulkan vs ES 2.0 FPS

(NOTE4 8084, 8100 draw call, Normal DVFS)

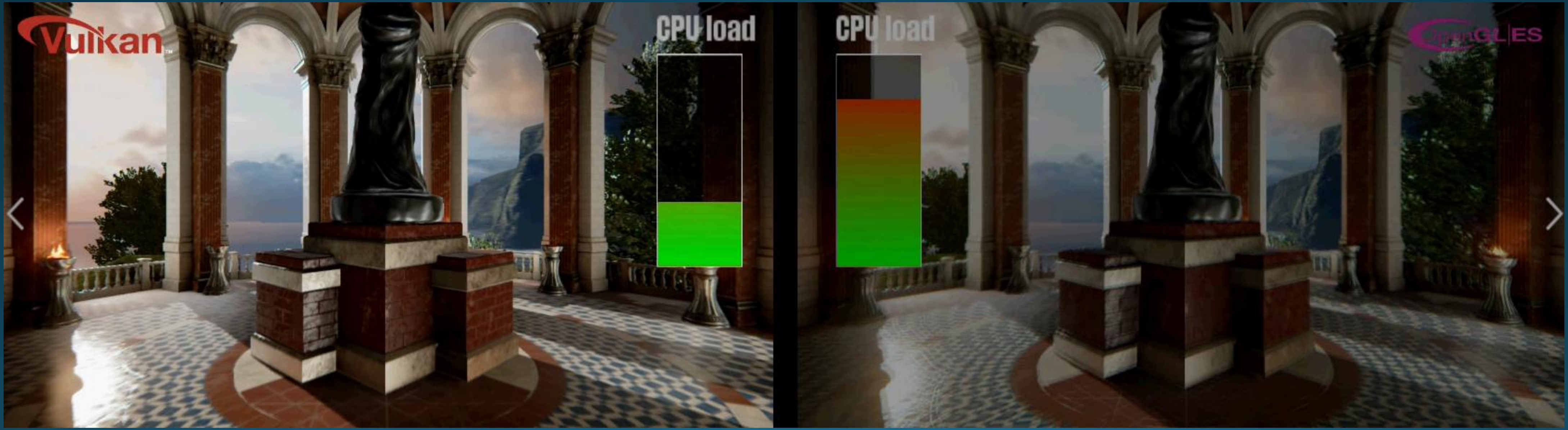
AVERAGE 23.2 FPS / 14.5 FPS

Proof of Concept demo



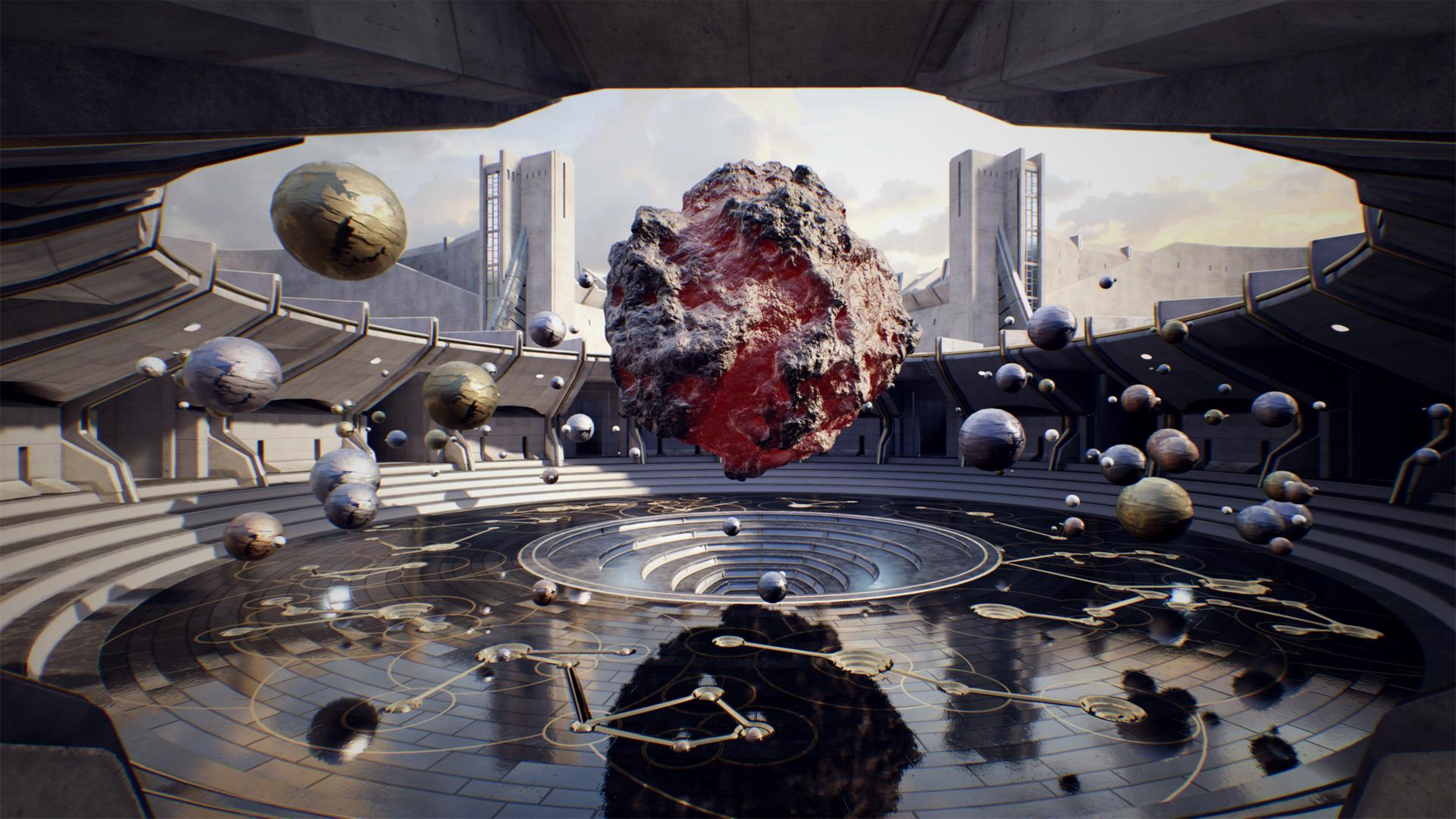
Porting Real Game Engine

- Ronin project with Epic Game for porting UE4 support Vulkan
- First goal was running Tappy chicken and SunTemple



SAMSUNG mobile

Developing cool demo : ProtoStar



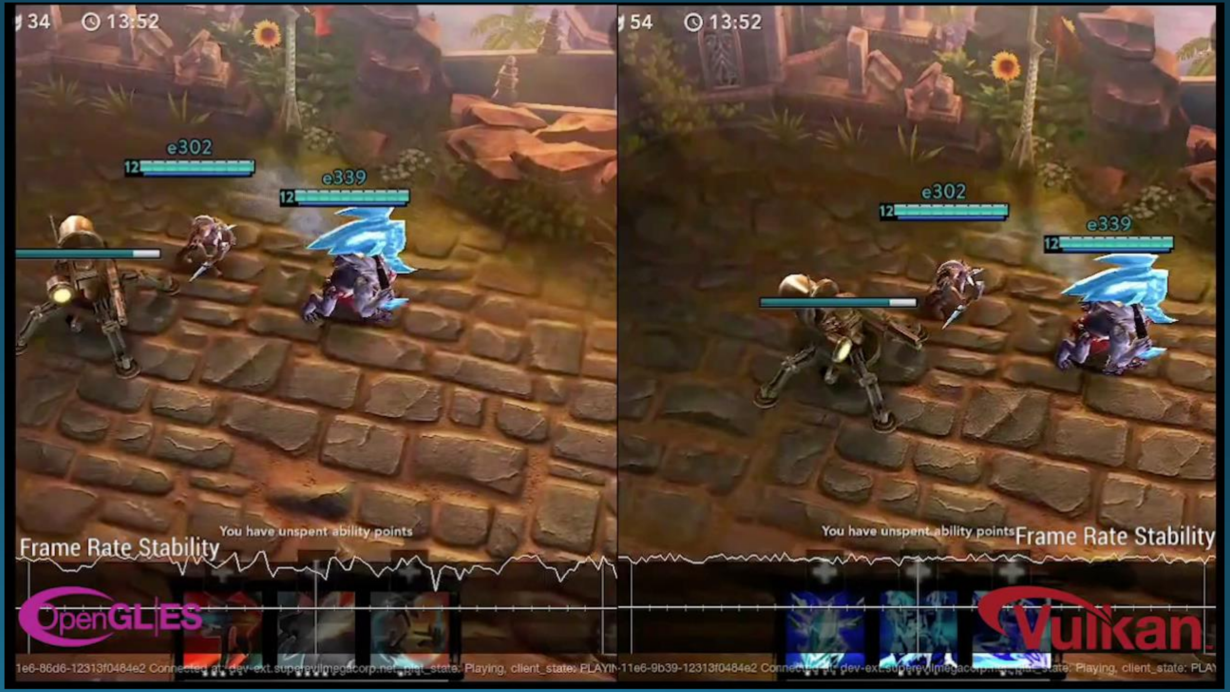
Developing Real Vulkan Games

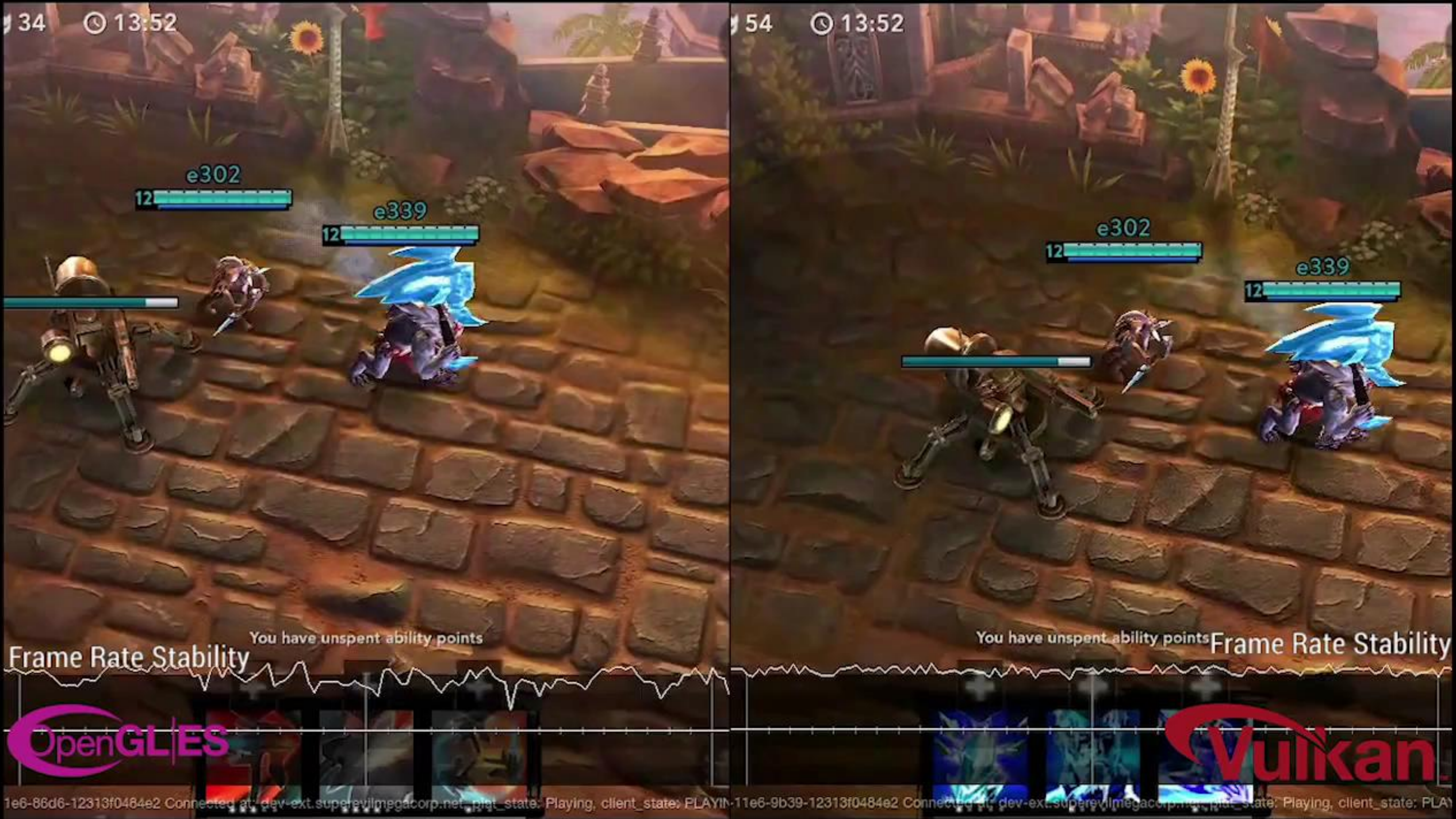
- We decided support game companies to port their games
- Tight schedule pushed us to focus on some specific directions

VainGlory : Performance Gain

- VainGlory was already well optimized and still near 60 fps in GLES
- But Vulkan gives everlasting 60 fps in any case with huge reduction in memory usage by ASTC

Performance	Normal	4 %
	Throttling	30 %
Power usage		5 %
Memory usage		25 %



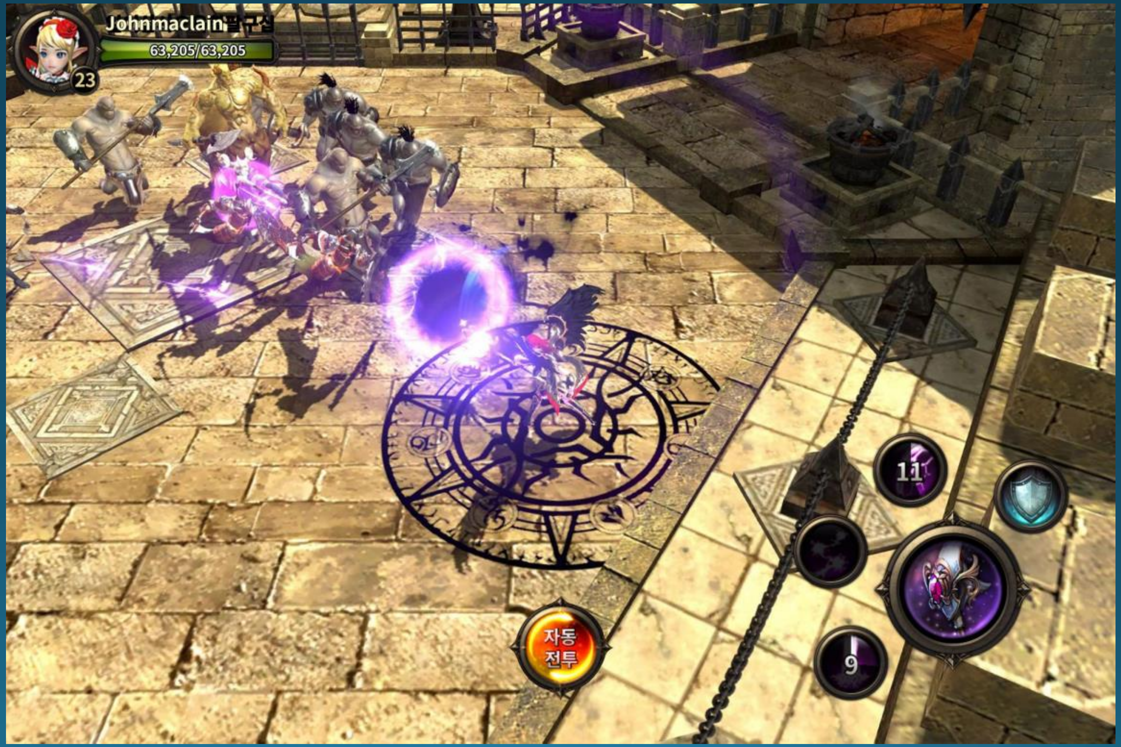


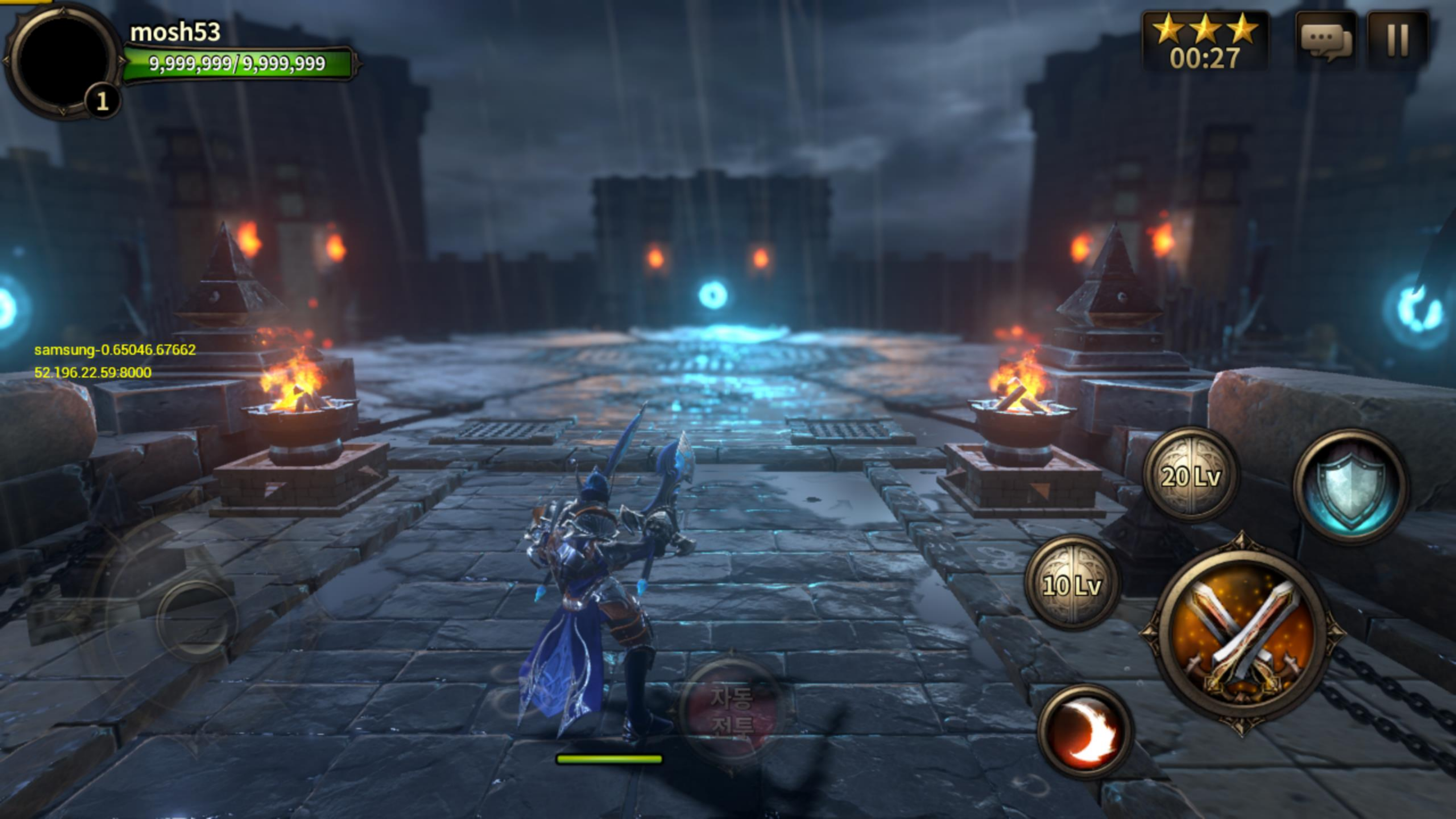
1e6-86d6-12313f0484e2 Connected at: dev-ext.superevilmegacorp.net, plat_state: Playing, client_state: PLAYIN

11e6-9b39-12313f0484e2 Connected at: dev-ext.superevilmegacorp.net, plat_state: Playing, client_state: PLAY

HIT : More Visual Quality

- HIT is UE4 based mobile action RPG game
- We focus on adding more graphics effects with same performance





mosh53
9,999,999/9,999,999
1
00:27
samsung-0.65046.67662
52.196.22.59:8000
20 Lv
10 Lv
자동
전투

HIT : More Visual Quality

- Samsung back ported all Vulkan RHI and new features related changes from Ronin and UE4.12 into old version of UE4
- Advanced reflection, refraction, GPU particles and cinematic depth of field effect were added
- NATGames changed stage design, assets and camera view to maximize the impact of all these new graphics effects
- After adding and changing things, performance is exactly same and game build passed publisher's QA process with both chipsets
- Spending 6 weeks for E3 demo and another 6 weeks for market

SAMSUNG
mobile

Wrap-up

- Heavy drawcall is not only case having benefit of Vulkan in mobile
- Vulkan gives CPU off-load, predictable behavior by explicit control and various ways to optimize games
- Vulkan drivers are not perfect but stable enough to make a game
- ASTC and SPV is essential, Vulkan will give more power with them
- No more driver magic, so you need to manage things by yourself

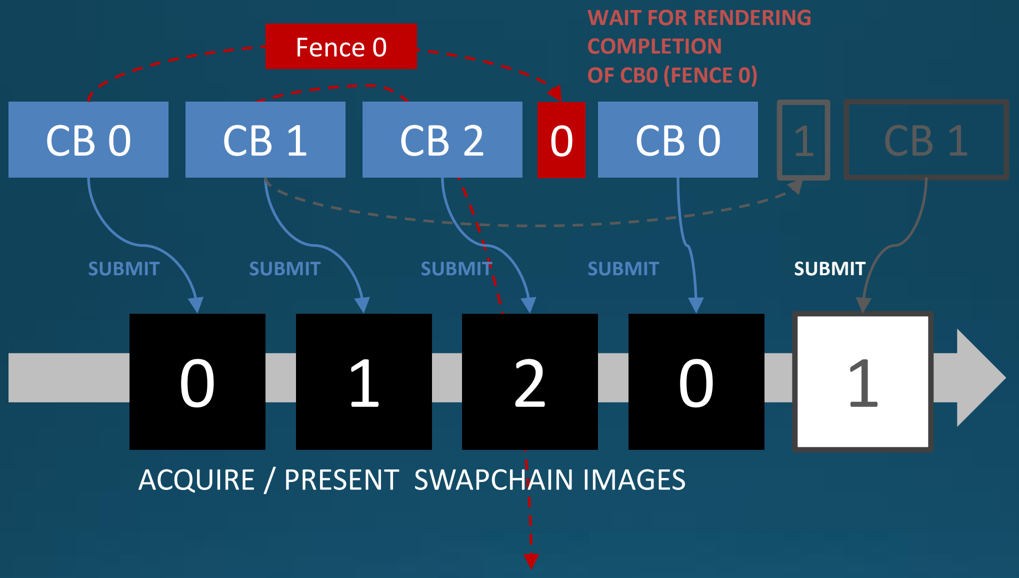
We will keep go on contributing to Vulkan and gaming industry!

Thank you!

If you have any questions, please contact

jwoo.kim@samsung.com or gamedev@samsung.com

Back-ups : Vulkan Tips 1/3



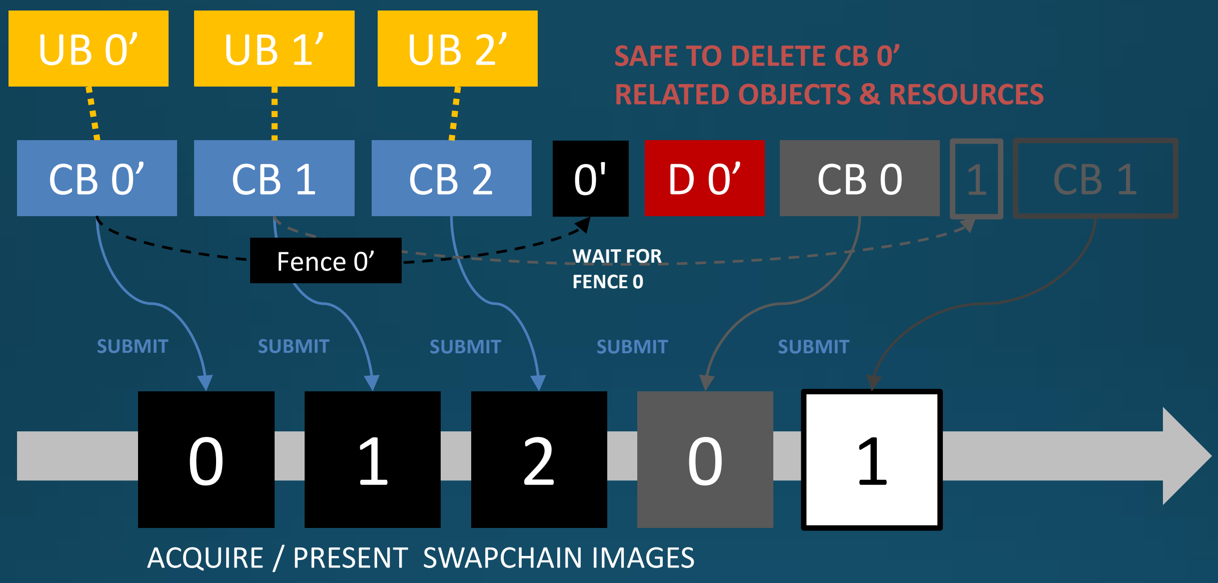
- Recommend to use A big primary command buffer for corresponding back-buffer and need to try to submit once a frame *(e.g. 3 back buffers requires at least 3 command buffers)*
- Should use "FIFO" instead of "MAILBOX" for SwapChain creation on Android to get stable Queue/Dequeue based on vsync

Back-ups : Vulkan Tips 2/3

- Recommend to not to bind render state every draw calls, need to remove duplications.
  - *there will be cases vkCmdSetXXX, vkCmdBindXXX functions are already bound at command buffer.*
- Recommend to use pipeline cache and caching everything as much as you can.
- Efficient managing of vkGraphicsPipeline is key point of rendering optimization. (e.g. using map)
- Recommend to not to use local fence usage. (local blocking command buffer submit)
- Recommend to use proper image layout for image resources. Especially for presentation and copy.
- Recommend to use fixed **UniformBuffer** for static objects to reduce vkUpdateDescriptorSets calls.
  - *and also fixed uniformbuffer can be used for secondary command buffer optimization*
- Recommend to use **UniformBufferPool** for dynamic objects
  - *Calling vkMapMemory, vkUnmapMemory can be optimized by mapping one single big memory and referencing this memory through the several uniform buffers*
  - *But, Please keep mind that should consider minMemoryMapAlignment for memory offset.*

SAMSUNG
mobile

Back-ups : Vulkan Tips 3/3



- Should use deferred deletion for vulkan related resources & objects

  - *Recommend to wait 3 frames till delete candidates are safely detached from it’s command buffer.*

- Recommend to use at least same count of **UniformBuffers** for it’s **CommandBuffers**.
- Please keep in mind that uniformbuffers at shader side should follow **‘std140 layout’** rules.
- Should use **uvec3** instead of **ivec3** or **vec3**, when you using VK_FORMAT_R8G8B8_UINT for attributes type.
- Enabling validation layer will be very helpful to correct API usage but should not trust error result 100%