GPU SUBMISSION STRATEGIES

MATTIAS LILJESON

GPU DEVELOPER TECHNOLOGY ENGINEER, AMD

WHOAMI

I work as a DevTech at AMD!

My role can be summarized as:

I help game developers make the most out of our hardware

Fun fact:

I always have a Justin Bieber poster in my office!

GPU WORK SUBMISSION

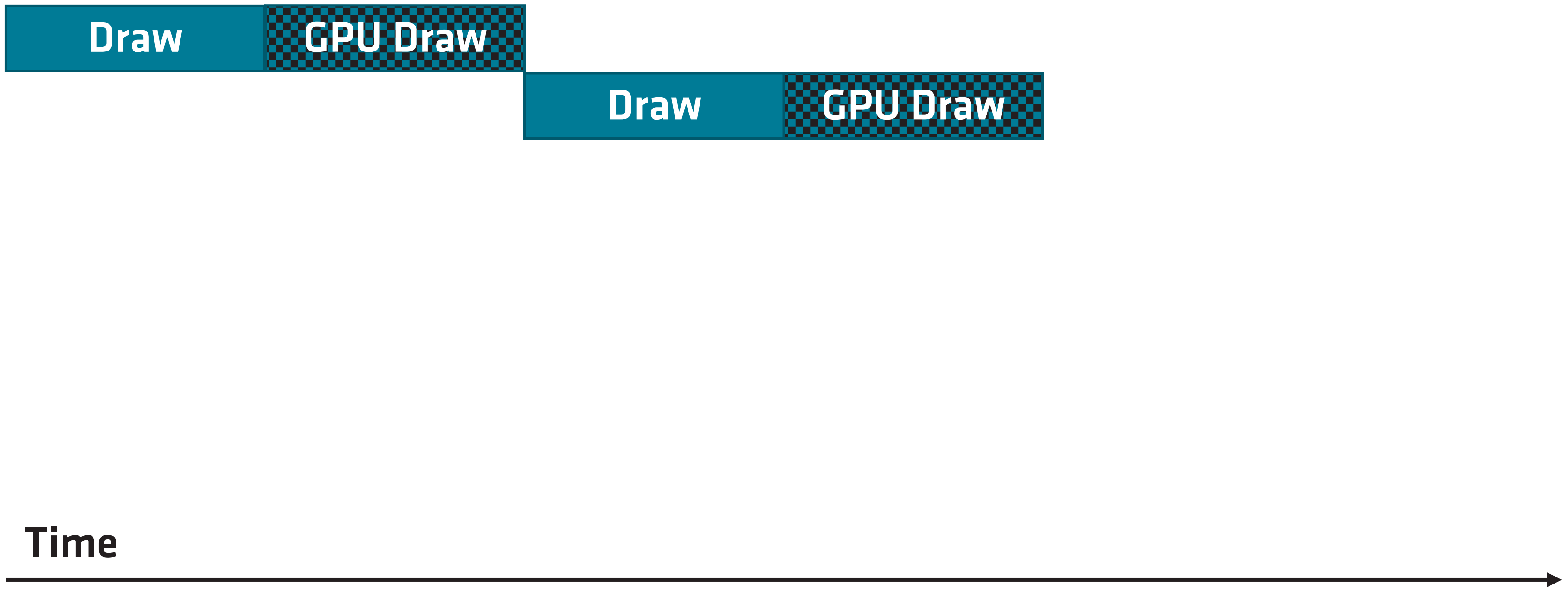
Some background

FROM A HIGH-LEVEL VIEW

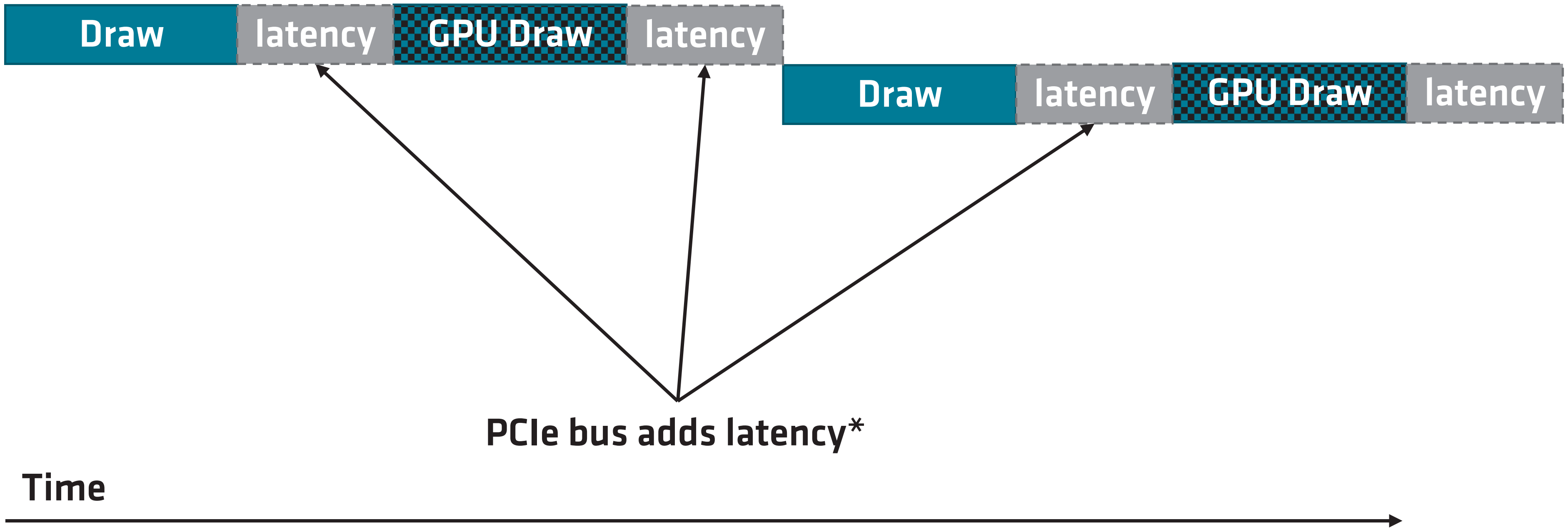
- Draw…
- Draw…
- Draw…

Draw → Draw

NAÏVE APPROACH



NAÏVE APPROACH



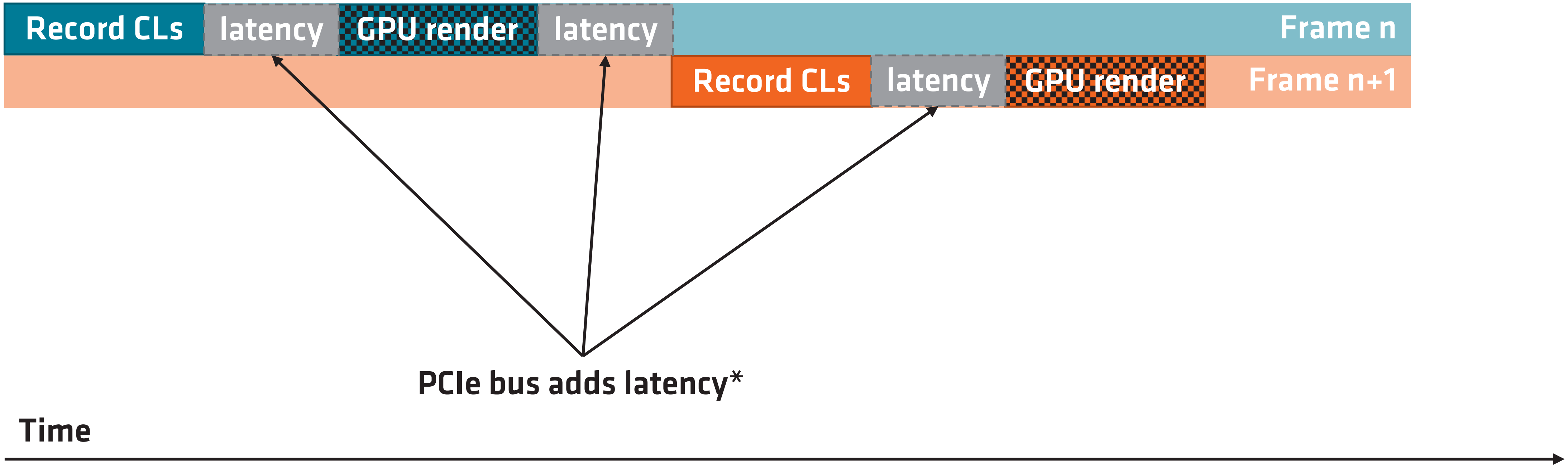
** Also the driver stack, we'll talk about that a bit later*

HOW WORK IS SUBMITTED TO THE GPU

- Command list:
  - Draw…
  - Draw…
  - Draw…

[Public]

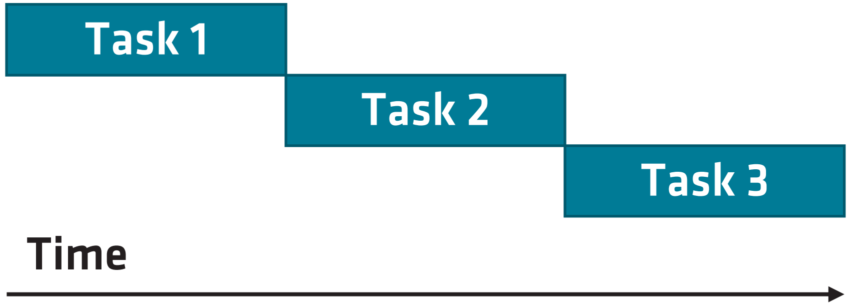
IF DRAWING WAS SYNCHRONOUS

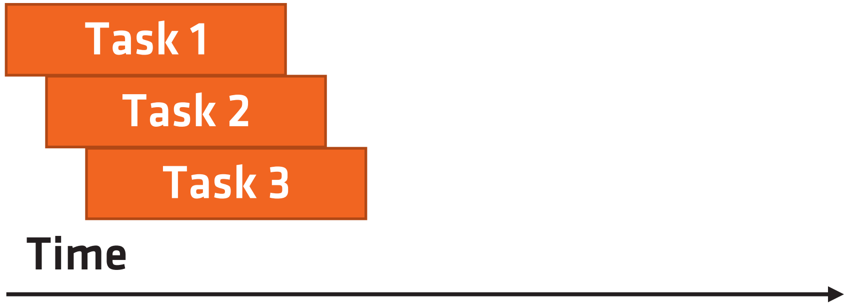


** Also the driver stack, we'll talk about that a bit later*

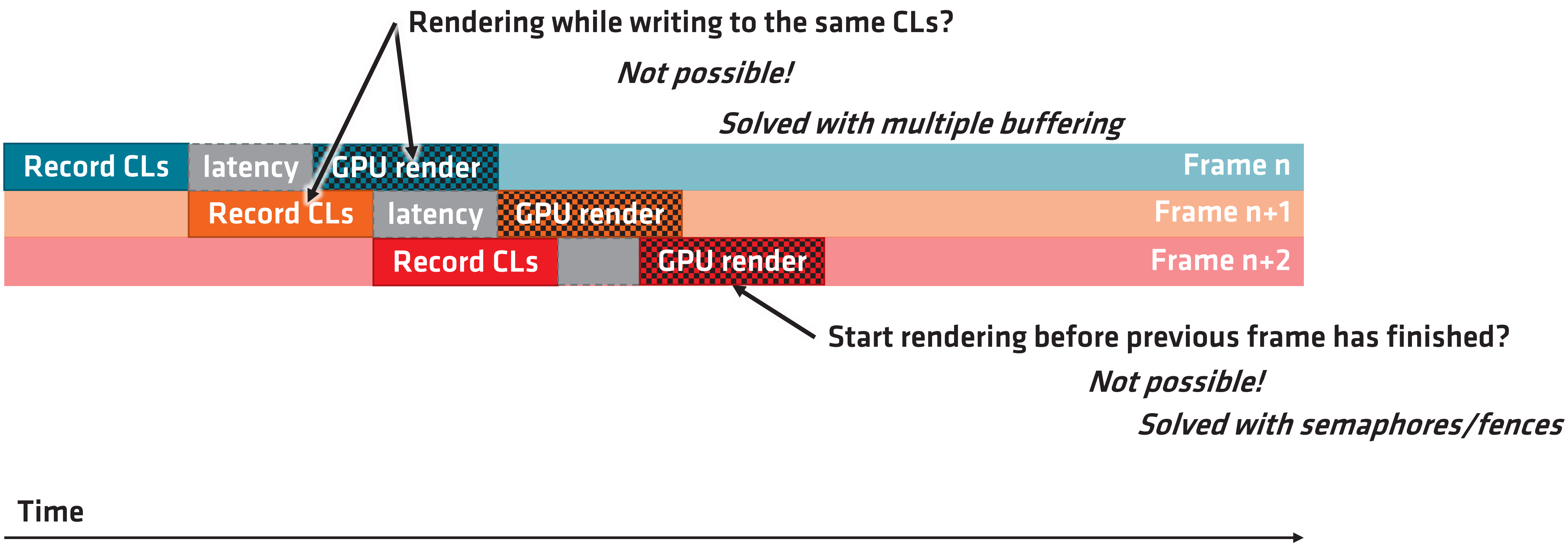
[Public]

SYNCHRONOUS VS ASYNCHRONOUS





WITH AN ASYNC APPROACH



WHAT ABOUT MULTIPLE SUBMISSIONS?

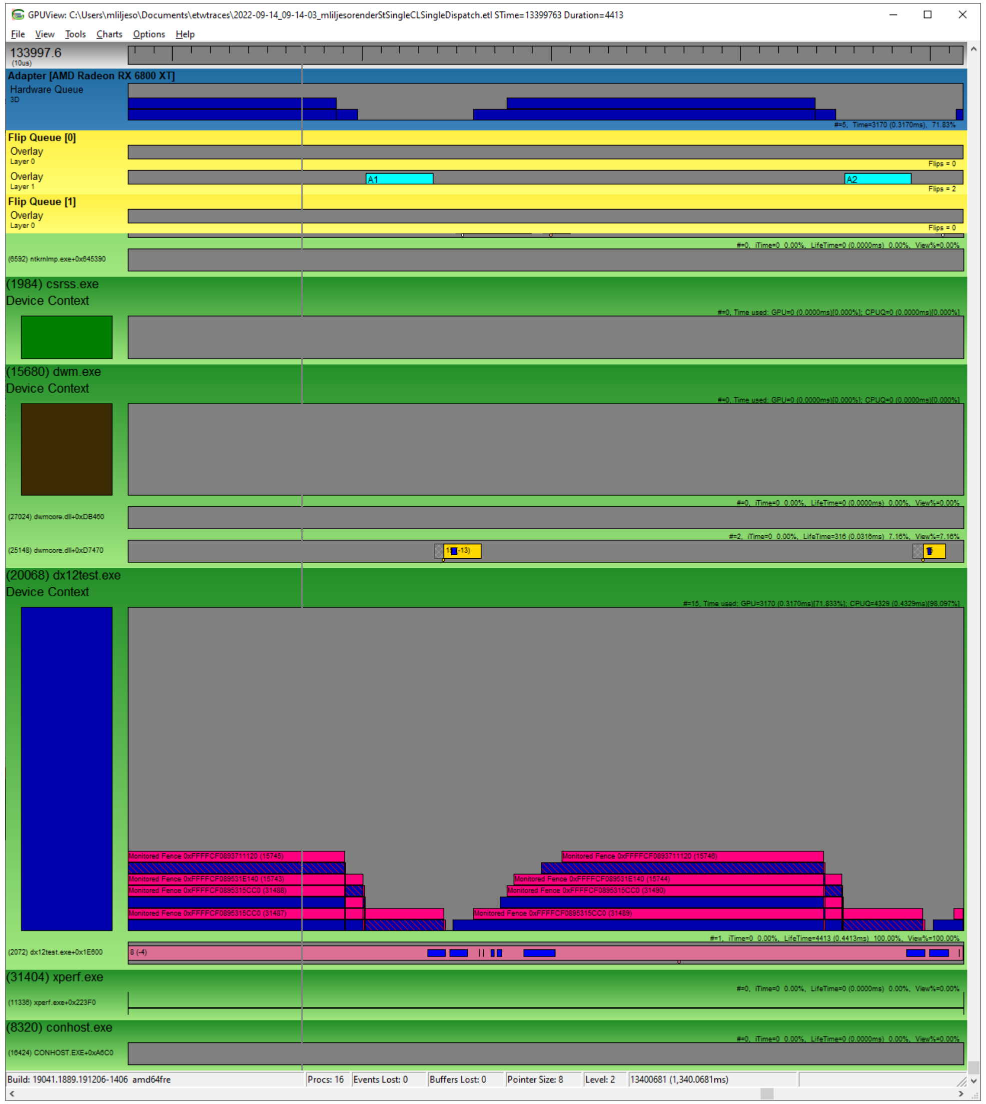
In most games we are not doing a single submit per frame though

- Let’s look at different submission patterns
- How they will be executed on the HW
- We will use a real-world example
- And some suitable tools

TODAY'S TOOLS

GPUView

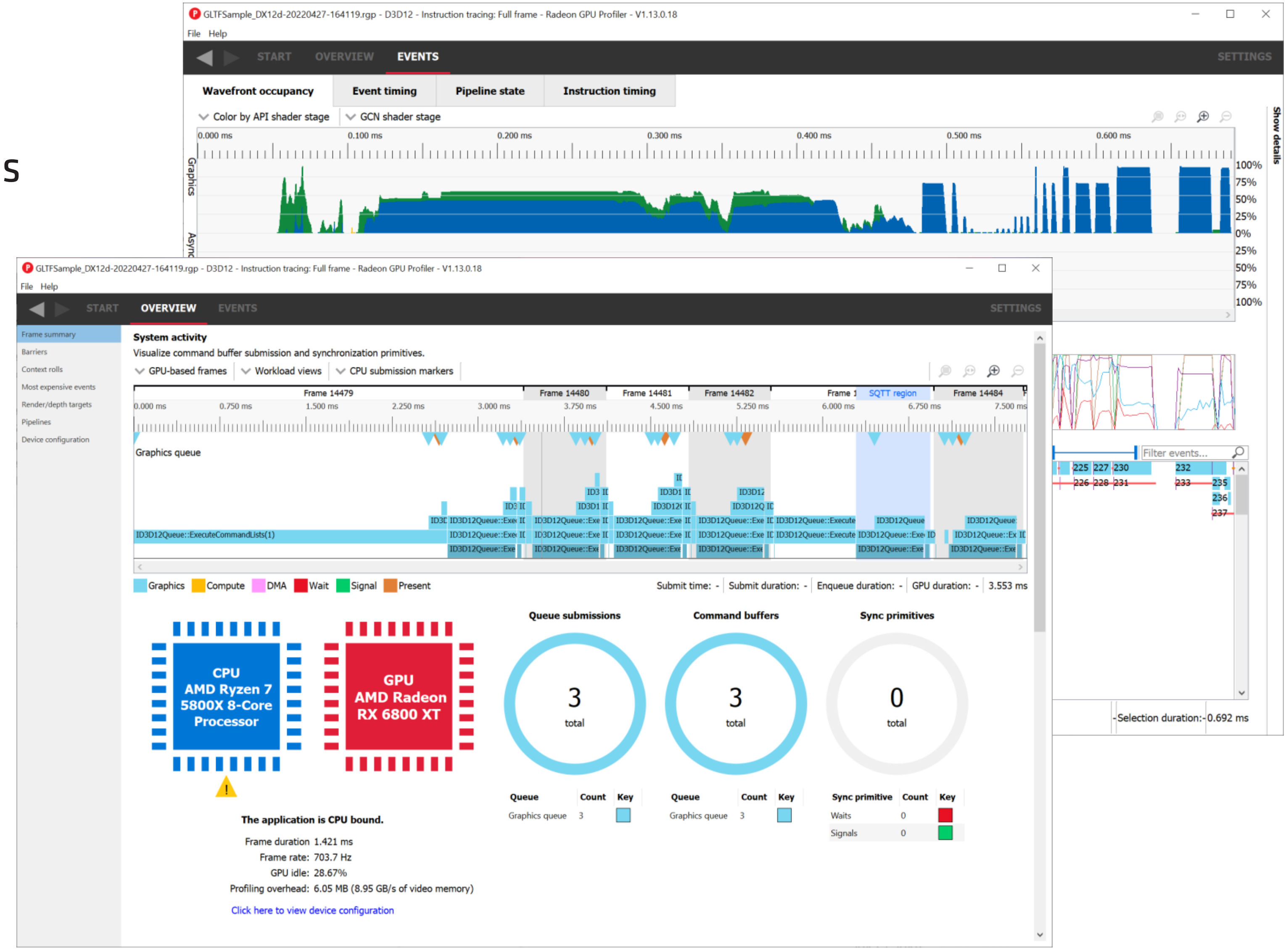
- Tool developed by Microsoft
- Holistic CPU and GPU overview
- Can be used to answer questions like:
  - Are we CPU bound or GPU bound?
  - Or both, during different parts of the frame?
  - Are stutters due to memory allocations in-between frames or something else?



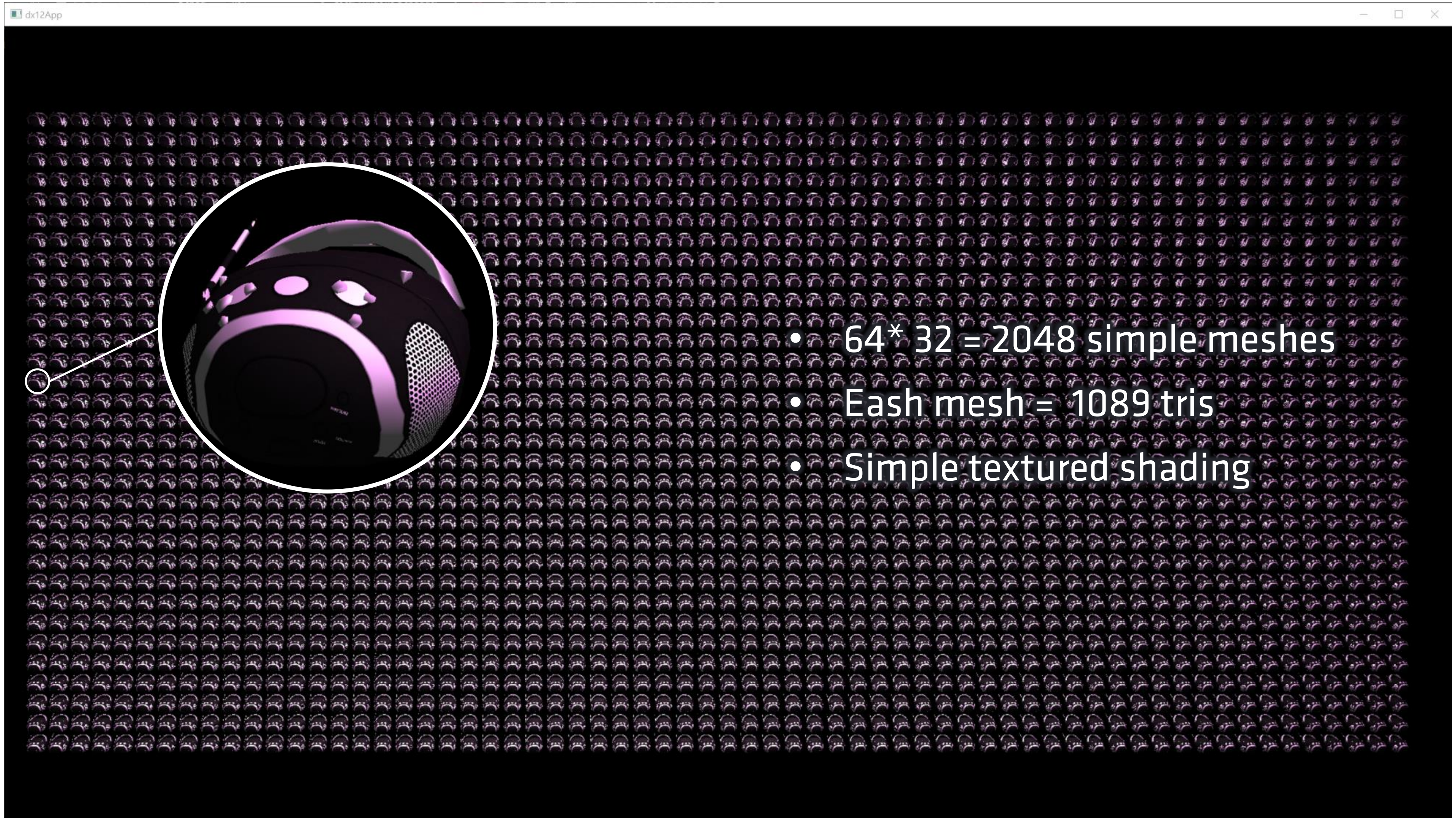
TODAY'S TOOLS

Radeon™ GPU Profiler (RGP)

- A profiling tool for AMD graphic cards
- In depth analysis of a GPU frame
- Graphics and async compute usage
- Event timing
- Barriers
- Instruction timing
- …



THE EXAMPLE APPLICATION



CONTENT WARNING

This application is designed to show limits

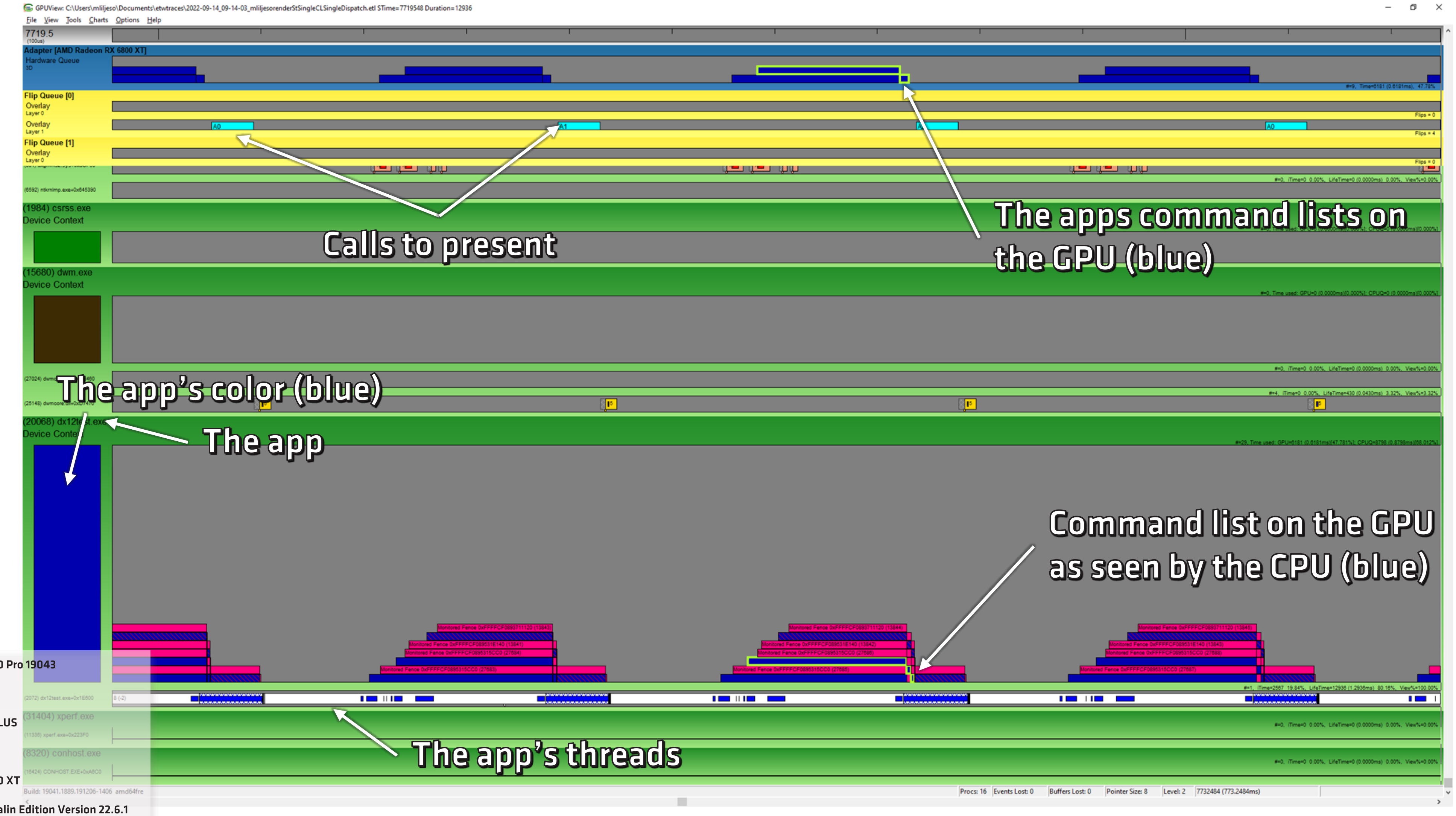
It’s not an example of good renderer design

THE BASE CASE

1 Command list for everything in the whole frame

- Barrier: framebuffer from **present** to **render target**
- Clear render target (back buffer) and depth buffer
- Draw 2048 boomboxes
- Barrier: framebuffer from **render target** to **present**

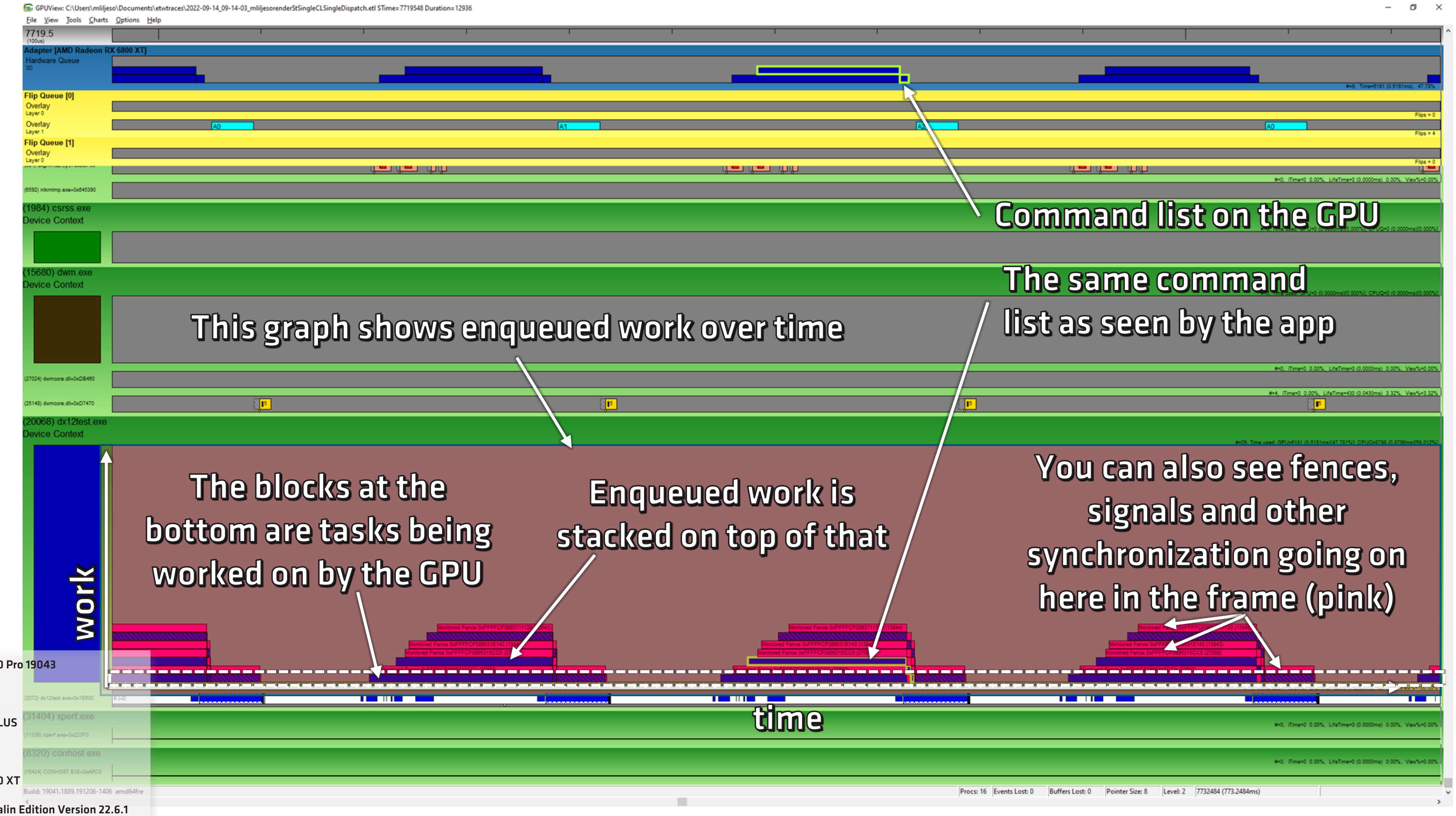
THE BASE CASE



OS	Microsoft Windows 10 Pro 19043
Processor	AMD Ryzen 7 5800X
Motherboard	TUF GAMING X570-PLUS
RAM	32,0 GB DDR4
GPU	AMD Radeon RX 6800 XT
Driver	AMD Software Adrenalin Edition Version 22.6.1

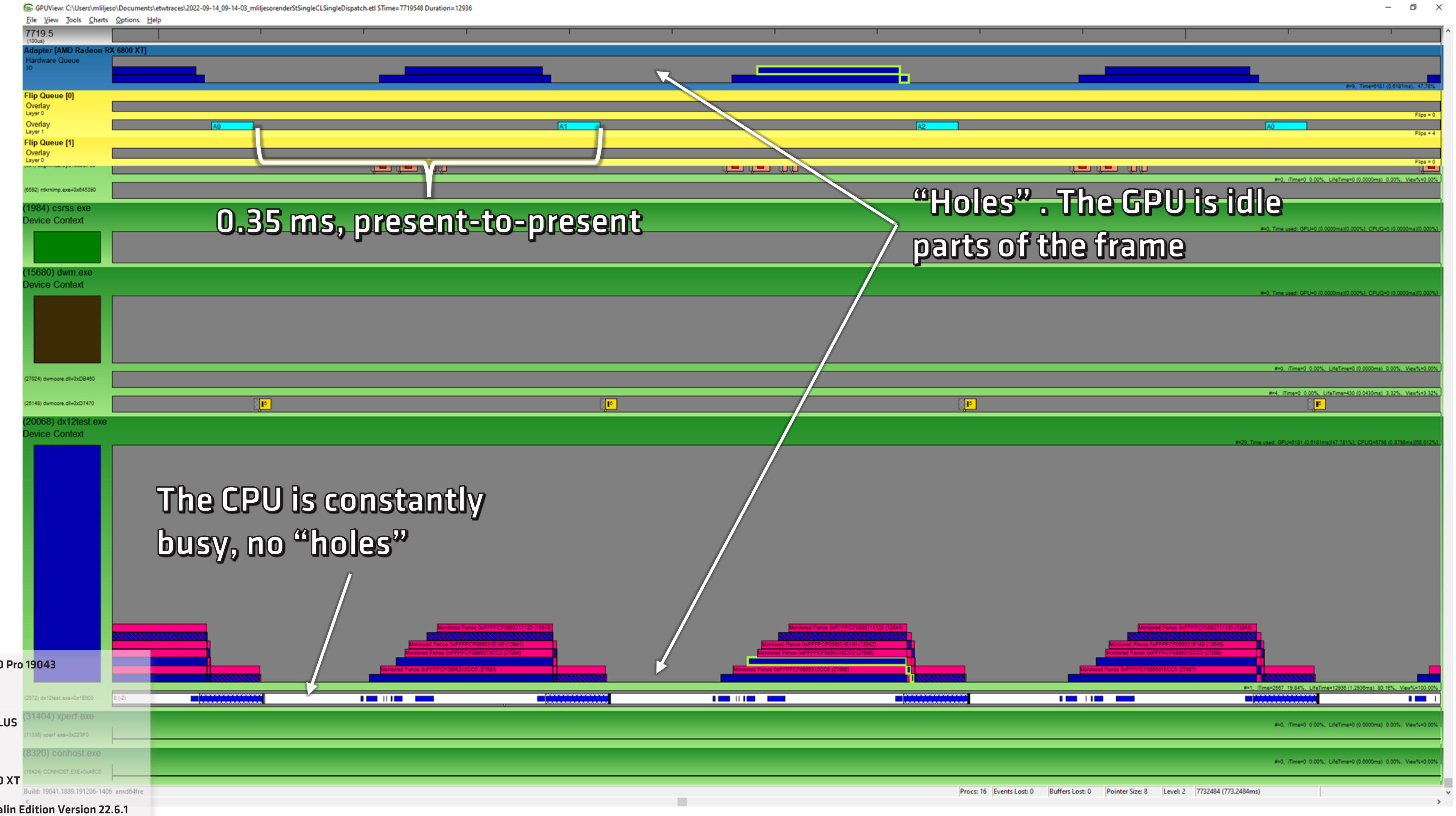
[Public]

THE BASE CASE



OS	Microsoft Windows 10 Pro 19043
Processor	AMD Ryzen 7 5800X
Motherboard	TUF GAMING X570-PLUS
RAM	32,0 GB DDR4
GPU	AMD Radeon RX 6800 XT
Driver	AMD Software Adrenalin Edition Version 22.6.1

THE BASE CASE

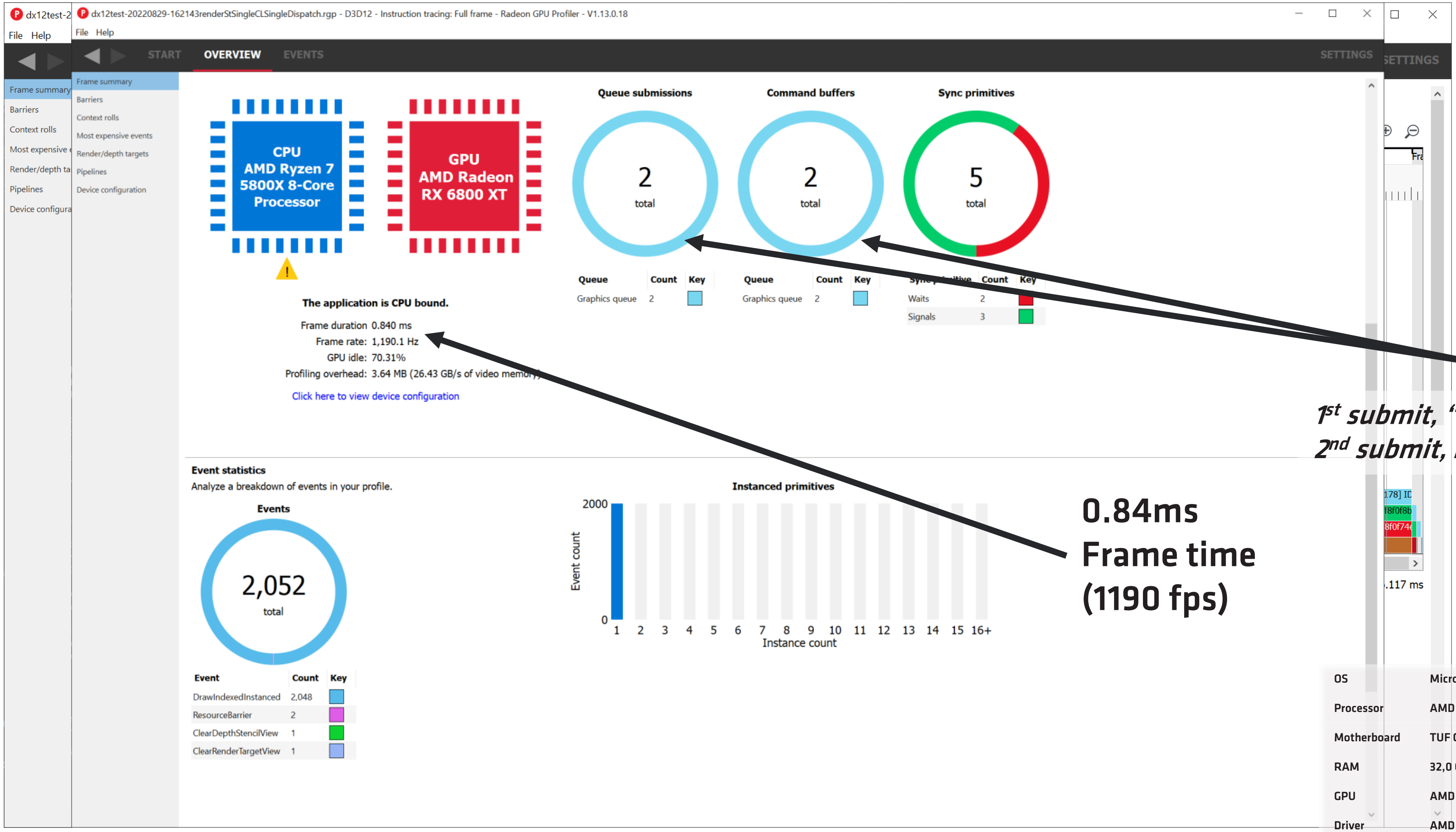


OS	Microsoft Windows 10 Pro 19043
Processor	AMD Ryzen 7 5800X
Motherboard	TUF GAMING X570-PLUS
RAM	32,0 GB DDR4
GPU	AMD Radeon RX 6800 XT
Driver	AMD Software Adrenalin Edition Version 22.6.1

THE BASE CASE

OS	Microsoft Windows 10 Pro 19043
Processor	AMD Ryzen 7 5800X
Motherboard	TUF GAMING X570-PLUS
RAM	32,0 GB DDR4
GPU	AMD Radeon RX 6800 XT
Driver	AMD Software Adrenalin Edition Version 22.6.1

THE BASE CASE



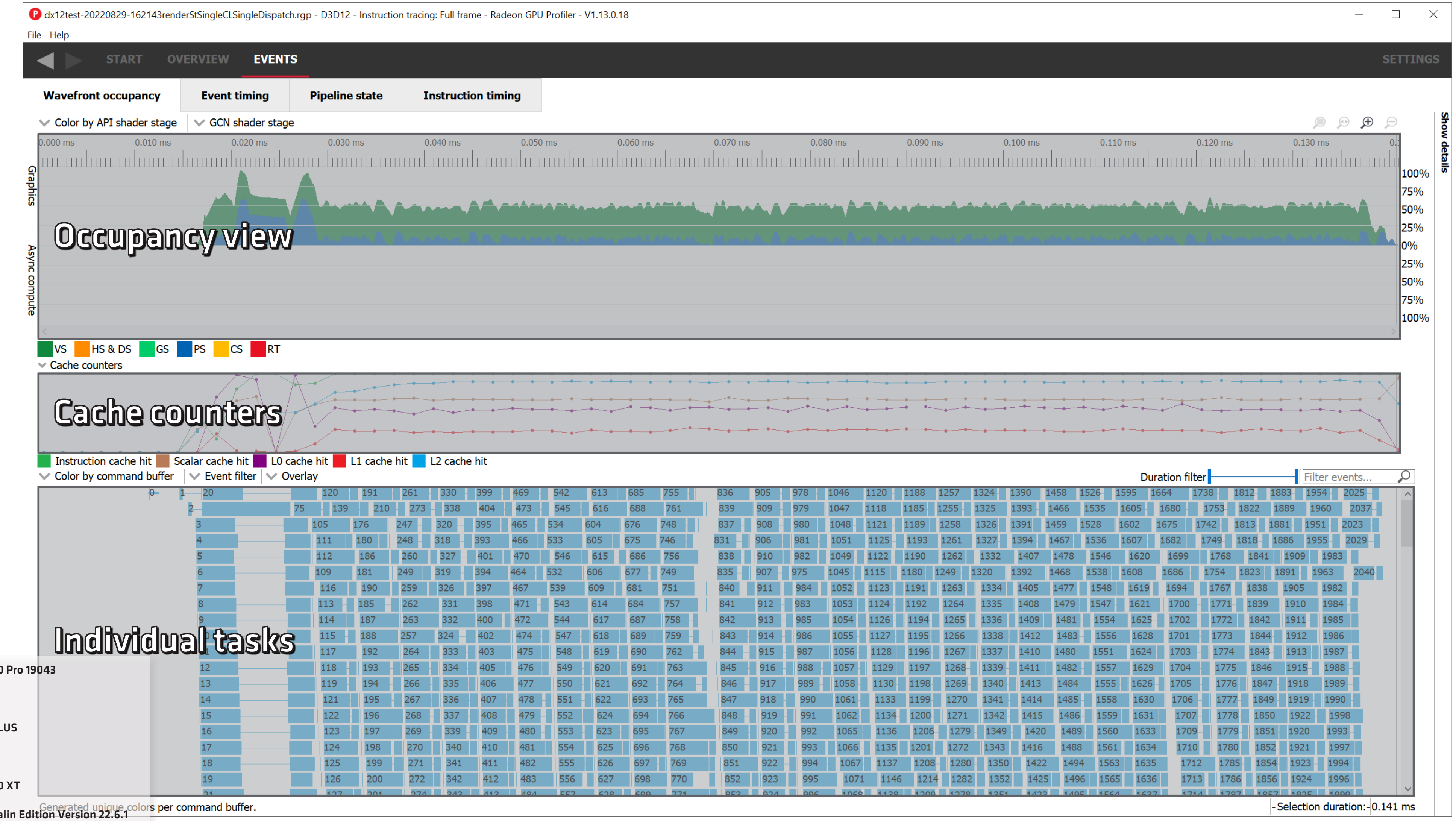
2 ?!

1st submit, "our" submit
2nd submit, inserted by driver

0.84ms Frame time (1190 fps)

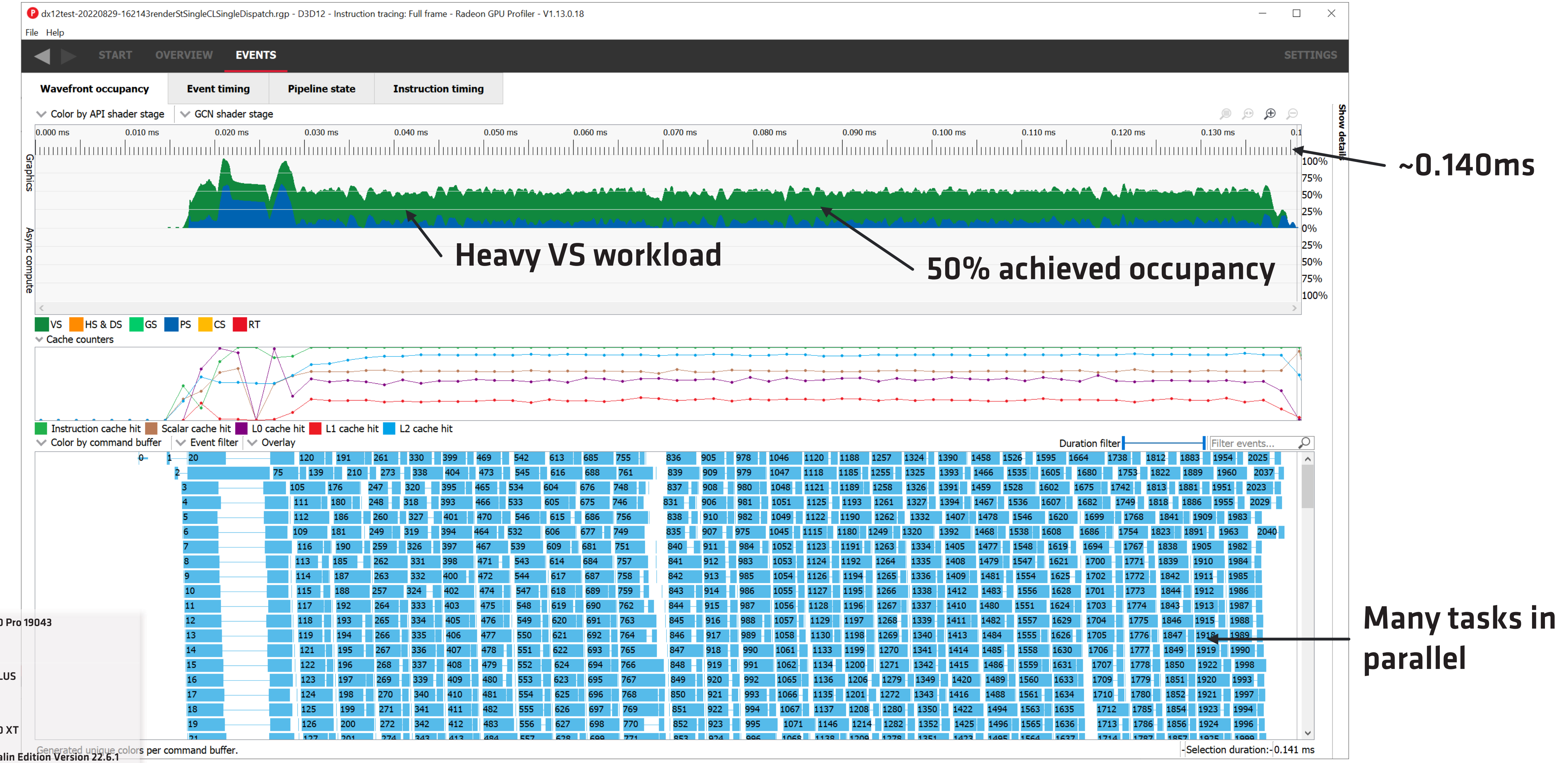
OS	Microsoft Windows 10 Pro 19043
Processor	AMD Ryzen 7 5800X
Motherboard	TUF GAMING X570-PLUS
RAM	32,0 GB DDR4
GPU	AMD Radeon RX 6800 XT
Driver	AMD Software Adrenalin Edition Version 22.6.1

THE BASE CASE



OS	Microsoft Windows 10 Pro 19043
Processor	AMD Ryzen 7 5800X
Motherboard	TUF GAMING X570-PLUS
RAM	32,0 GB DDR4
GPU	AMD Radeon RX 6800 XT
Driver	AMD Software Adrenalin Edition Version 22.6.1

THE BASE CASE



OS	Microsoft Windows 10 Pro 19043
Processor	AMD Ryzen 7 5800X
Motherboard	TUF GAMING X570-PLUS
RAM	32,0 GB DDR4
GPU	AMD Radeon RX 6800 XT
Driver	AMD Software Adrenalin Edition Version 22.6.1

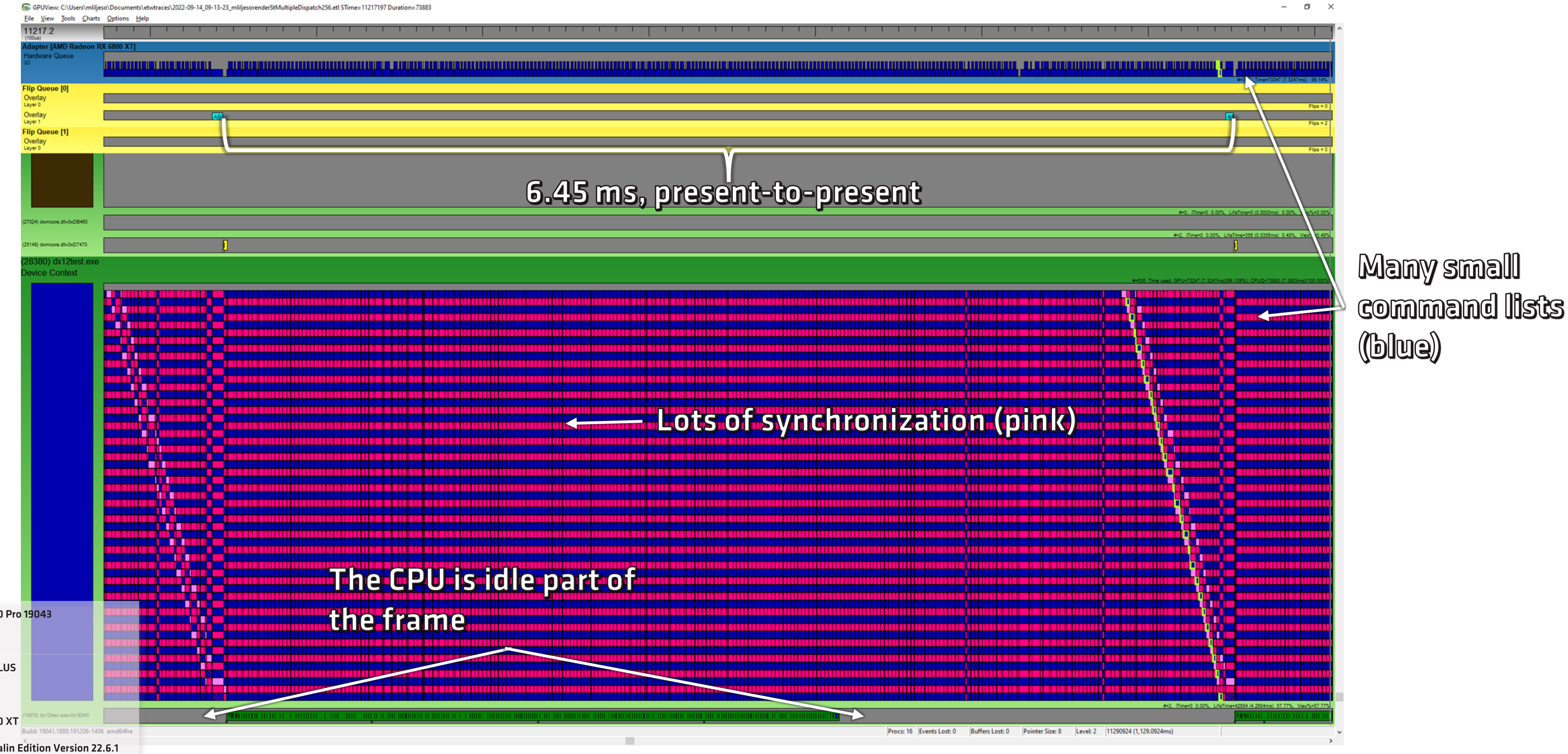
WITH SMALL BATCHES

Command lists with a few draws

1:	Barrier: framebuffer from present to render target Clear render target (back buffer) and depth buffer
2 – 257:	Draw 2048 boomboxes in batches of 8. (8 meshes per CL)
258:	Barrier: framebuffer from render target to present

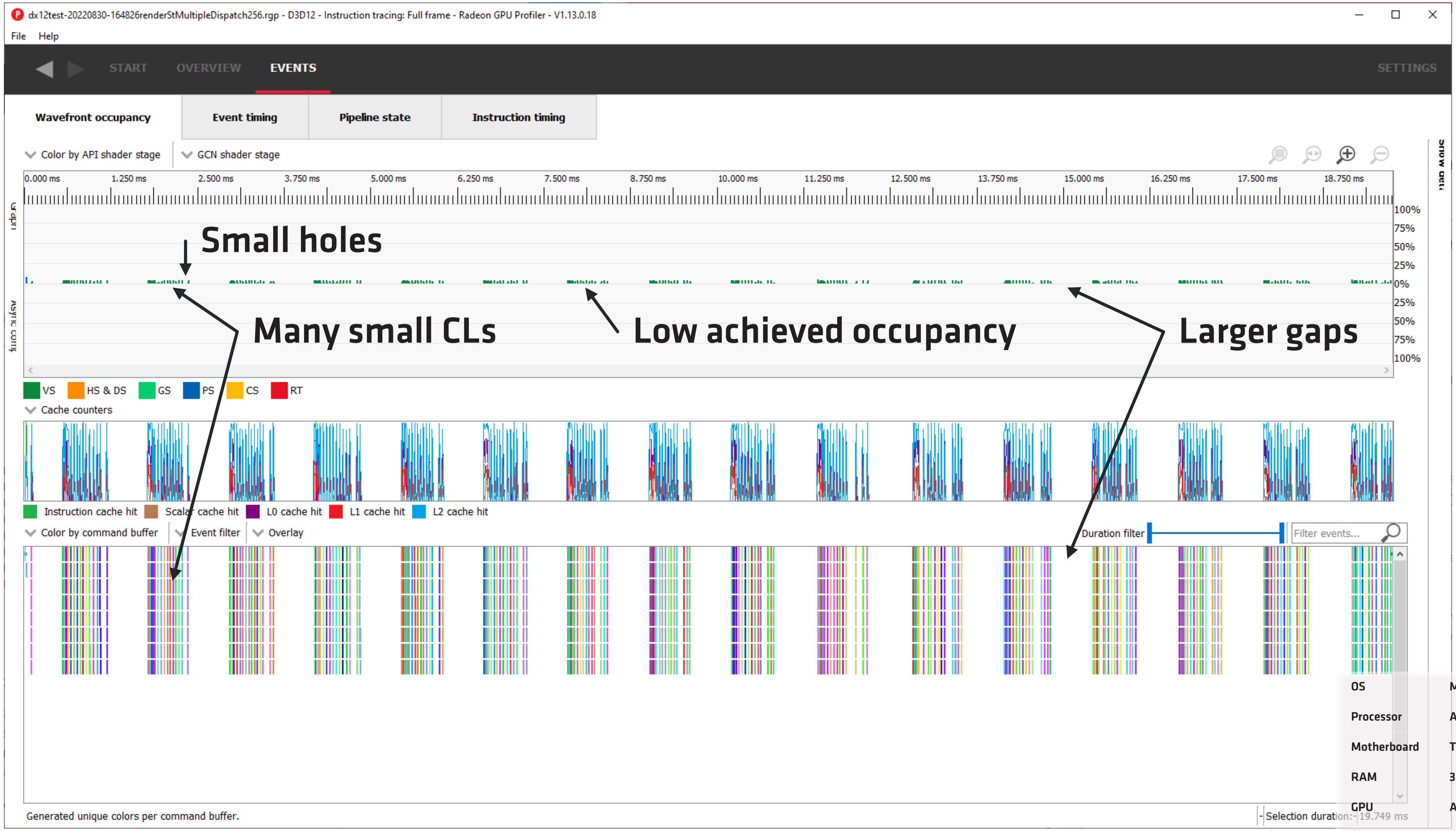
All CLs submitted one-by-one

WITH SMALL BATCHES



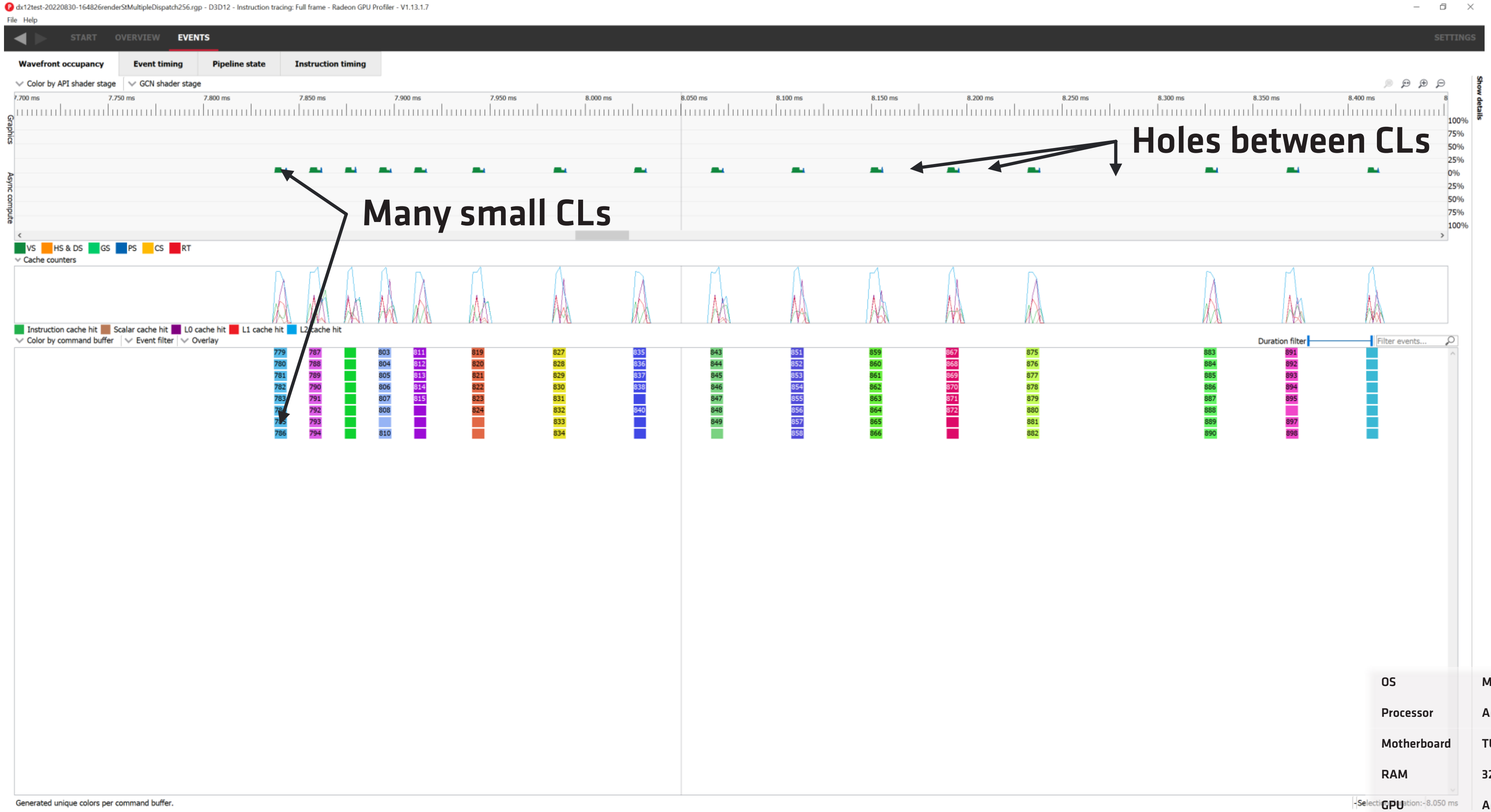
OS	Microsoft Windows 10 Pro 19043
Processor	AMD Ryzen 7 5800X
Motherboard	TUF GAMING X570-PLUS
RAM	32,0 GB DDR4
GPU	AMD Radeon RX 6800 XT
Driver	AMD Software Adrenalin Edition Version 22.6.1

WITH SMALL BATCHES



OS	Microsoft Windows 10 Pro 19043
Processor	AMD Ryzen 7 5800X
Motherboard	TUF GAMING X570-PLUS
RAM	32,0 GB DDR4
GPU	AMD Radeon RX 6800 XT
Driver	AMD Software Adrenalin Edition Version 22.6.1

WITH SMALL BATCHES



OS	Microsoft Windows 10 Pro 19043
Processor	AMD Ryzen 7 5800X
Motherboard	TUF GAMING X570-PLUS
RAM	32,0 GB DDR4
GPU	AMD Radeon RX 6800 XT
Driver	AMD Software Adrenalin Edition Version 22.6.1

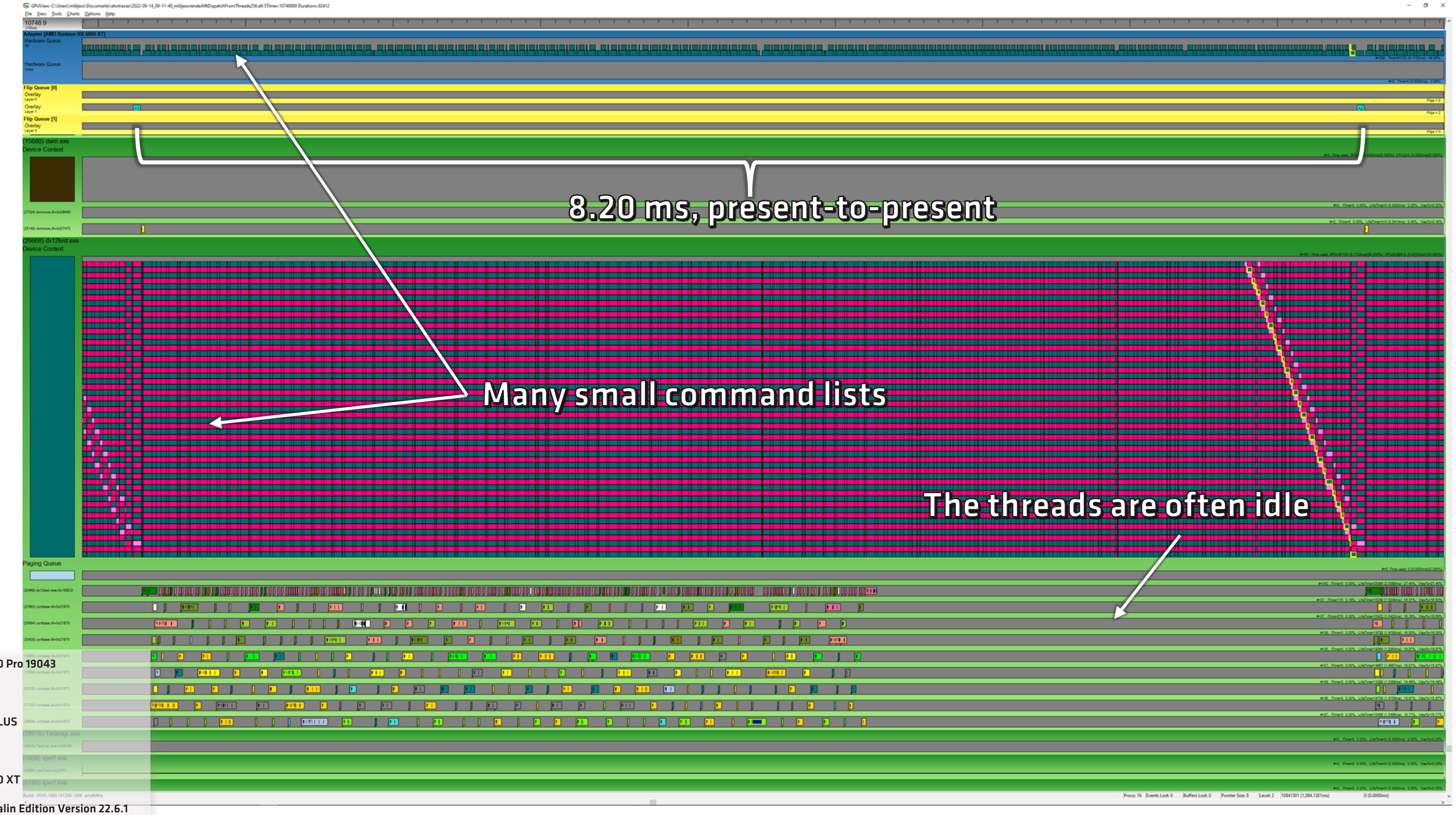
WITH MULTI THREADING

Command lists with a few draws

1:	Barrier: framebuffer from present to render target Clear render target (back buffer) and depth buffer
2 – 257:	Draw 2048 boomboxes in batches of 8. (8 meshes per CL) As 256 tasks from a thread pool
258:	Barrier: framebuffer from render target to present

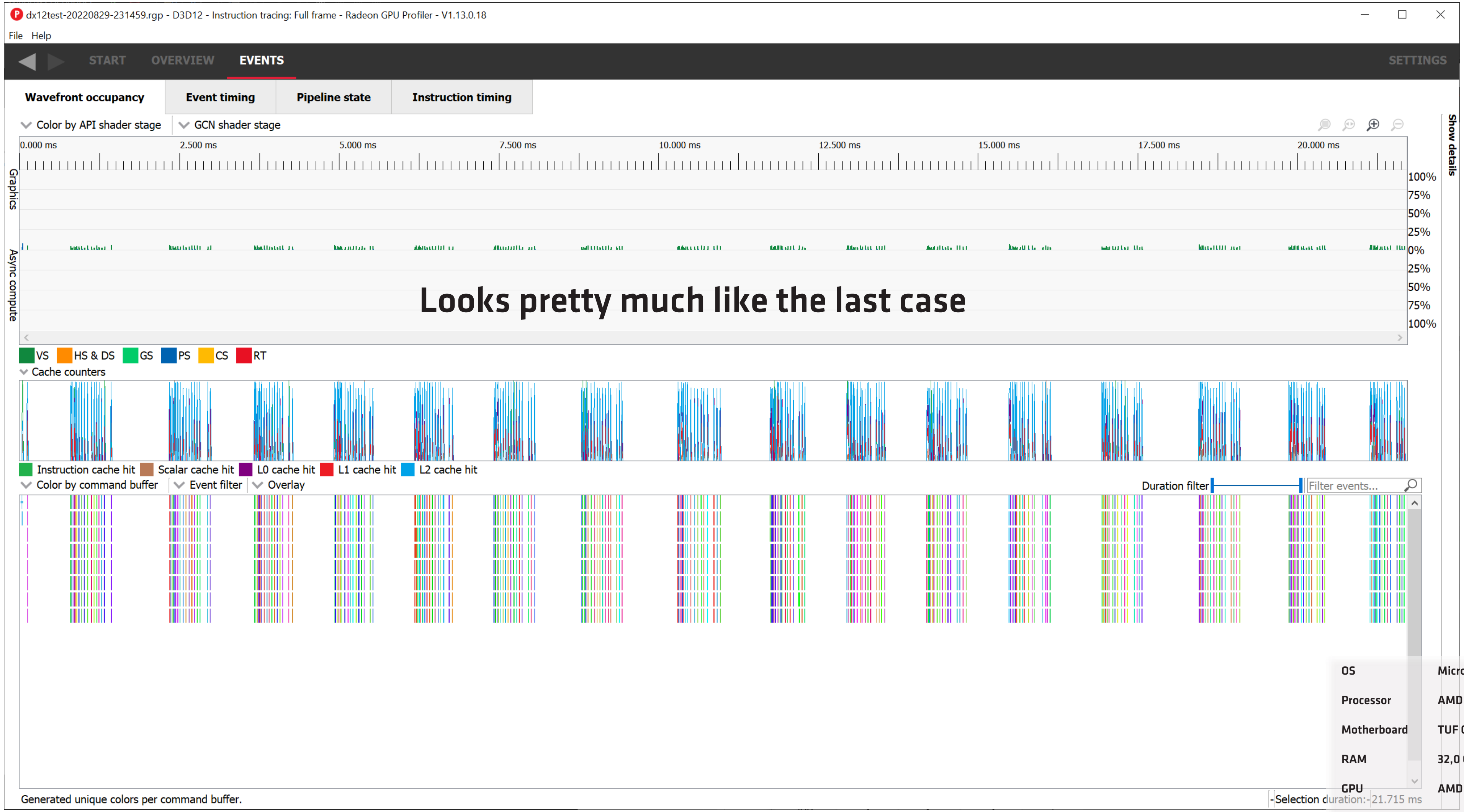
All CLs submitted one-by-one from their respective thread

WITH MULTI THREADING



OS	Microsoft Windows 10 Pro 19043
Processor	AMD Ryzen 7 5800X
Motherboard	TUF GAMING X570-PLUS
RAM	32,0 GB DDR4
GPU	AMD Radeon RX 6800 XT
Driver	AMD Software Adrenalin Edition Version 22.6.1

WITH MULTI THREADING



OS	Microsoft Windows 10 Pro 19043
Processor	AMD Ryzen 7 5800X
Motherboard	TUF GAMING X570-PLUS
RAM	32,0 GB DDR4
GPU	AMD Radeon RX 6800 XT
Driver	AMD Software Adrenalin Edition Version 22.6.1

THAT DOESN'T LOOK TOO GOOD

SO WHY DO WE SEE WHAT WE SEE?

- Why does performance tank so badly with many submits?
- Why doesn't multithreading help?

Let's look at the driver stack!

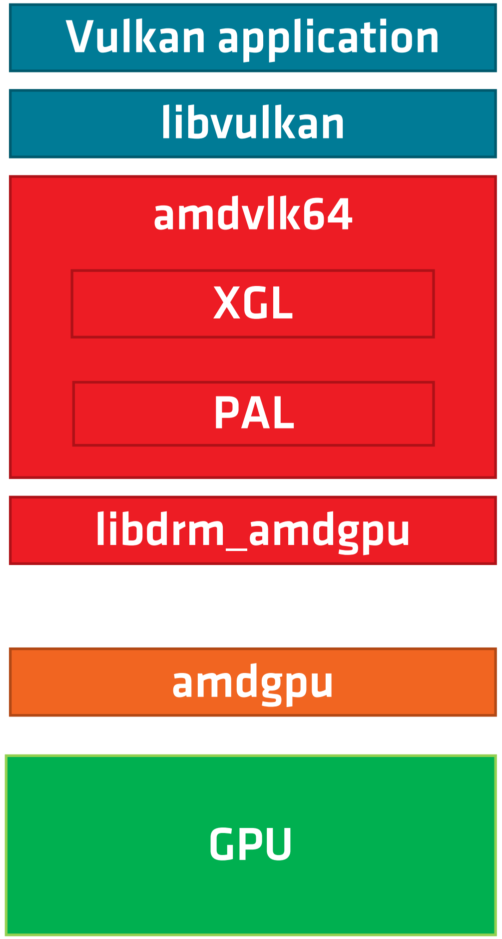
- In this case, let's use our open-source vulkan driver as an example
- Since it's open source, we can freely discuss its details
- The source code is available online here: https://github.com/GPUOpen-Drivers/AMDVLK

THE DRIVER STACK

- This is what our Open Source driver look like
  - The application is on top and calls the vulkan runtime, libvulkan
  - The runtime talks to the upmost part of the driver, amdvlk64
  - The red upper part, amdvlk64 talks to libdrm_amdgpu which communicates with the lower orange part amdgpu
  - amdgpu talks to the graphics card

Drm in this case stands for Direct Rendering Manager

- Basically an API for modern GPU drivers in linux



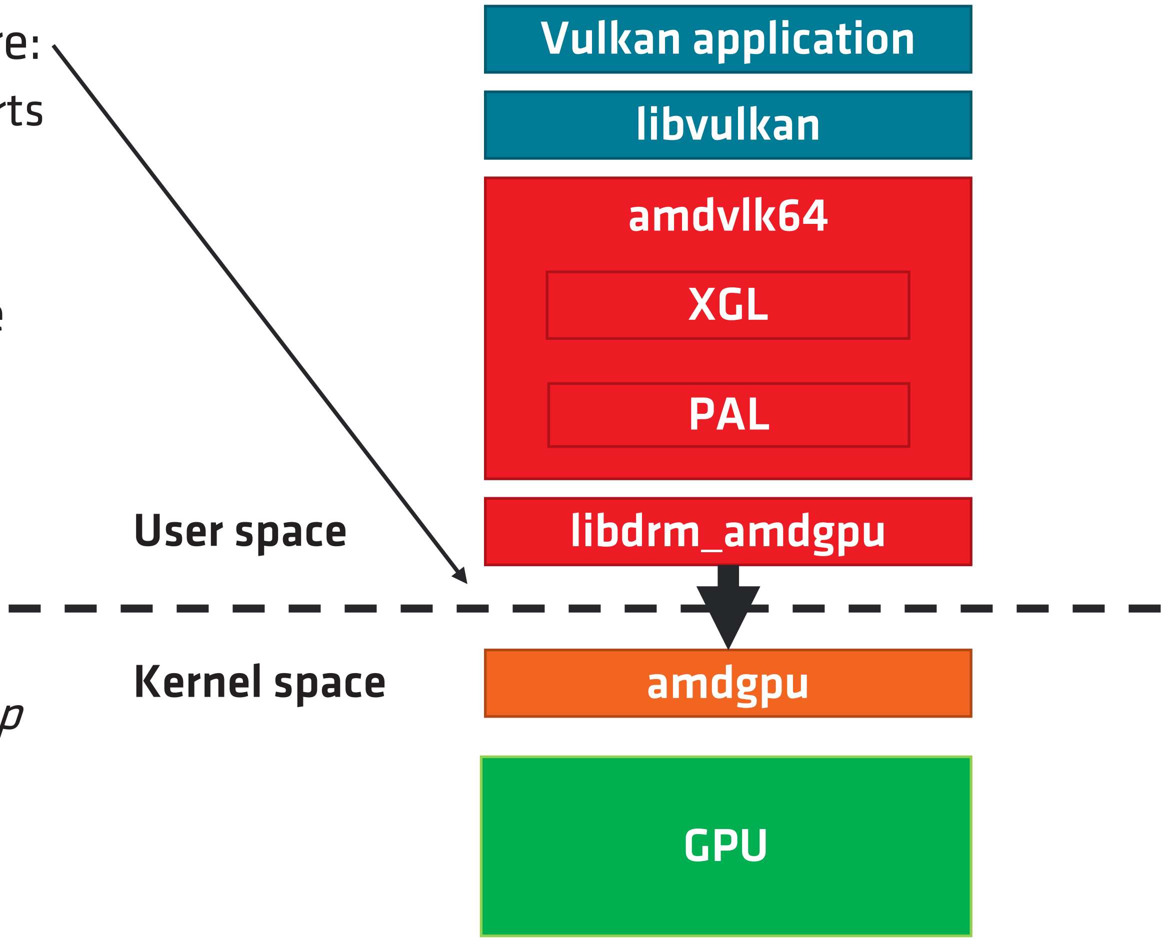
[Public]

THE DRIVER STACK

- The interesting part of this picture is this line here:
  - It divides the driver into user and kernel space parts
- To protect the system and the user, only the operating system kernel can access the hardware
  - The CPU can therefore execute in **user space** and **kernel space**
  - Switching between the two takes time

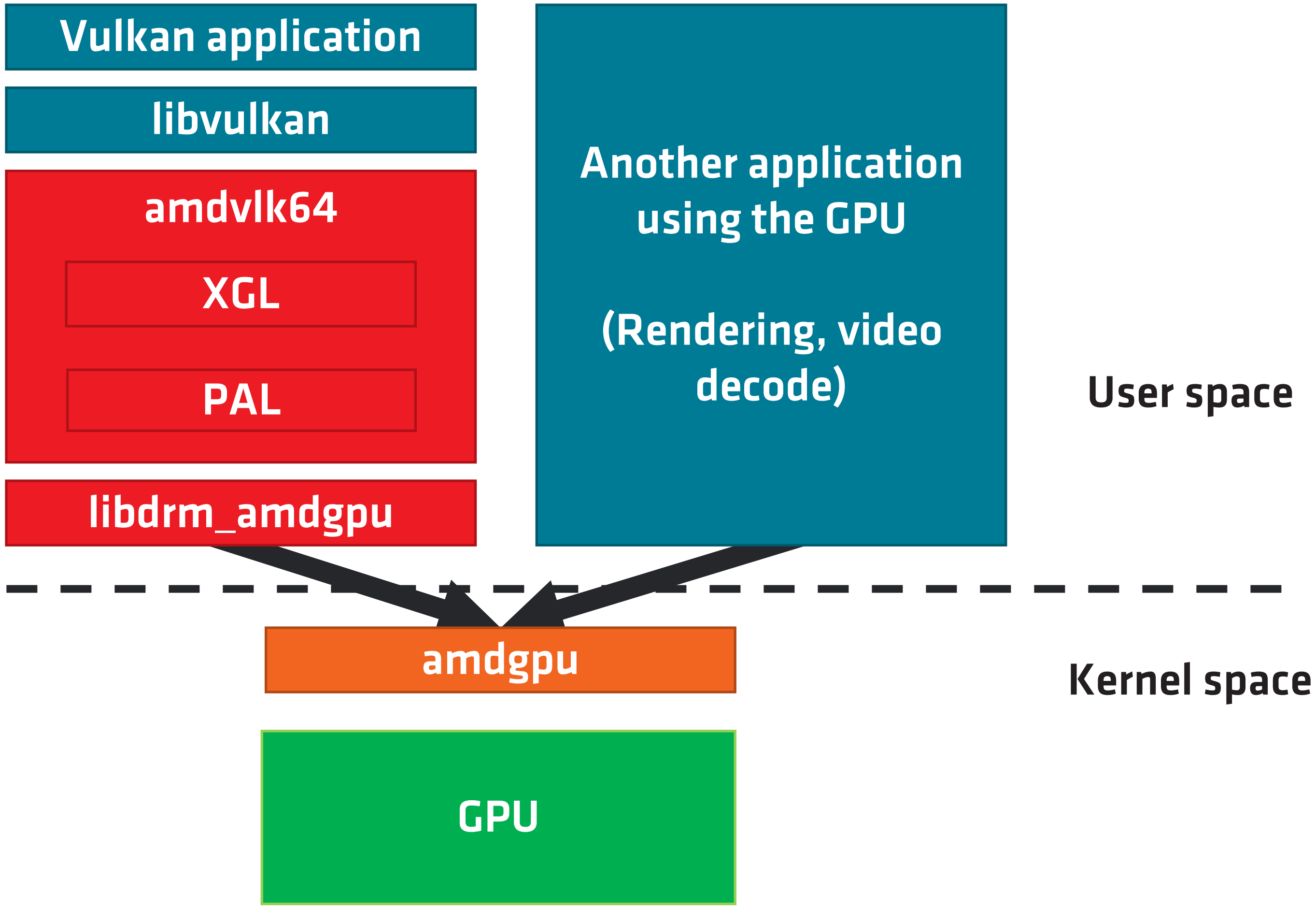
Important note:

Not all calls to the user mode part of the driver end up causing a mode switch to kernel mode



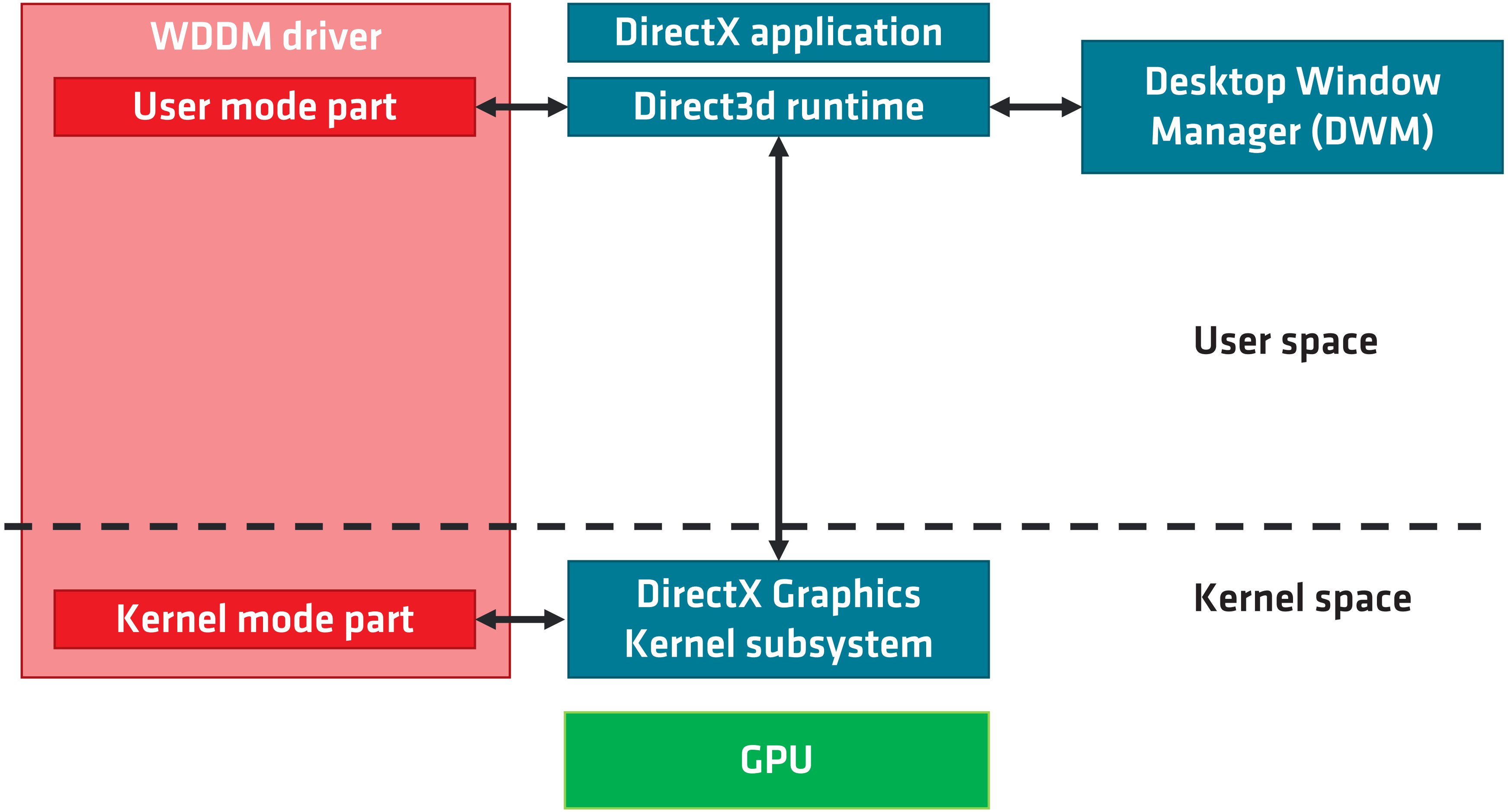
THE DRIVER STACK

- The driver also needs to support multiple applications using the graphics card
- The kernel mode part therefore supports scheduling GPU work for multiple applications

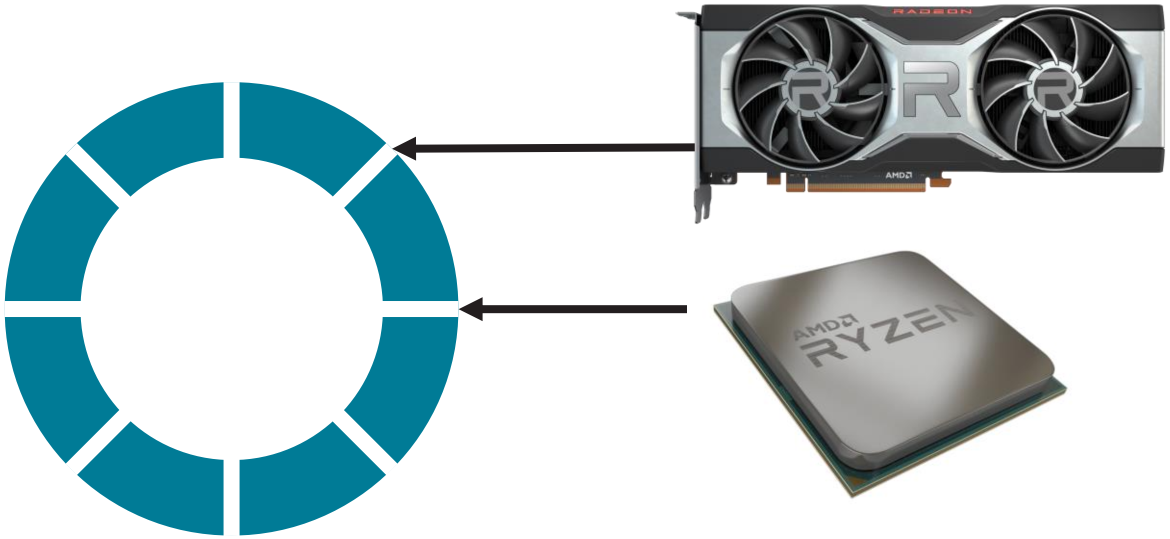


THE DRIVER STACK, ON WINDOWS

- There's a similar setup on windows
- The driver is split up in user mode and kernel mode parts
  - The model is called: *Windows Display Driver Model*
- There is a compositor that handles multiple applications rendering at the same time
  - It is called: *Desktop Window Manager*

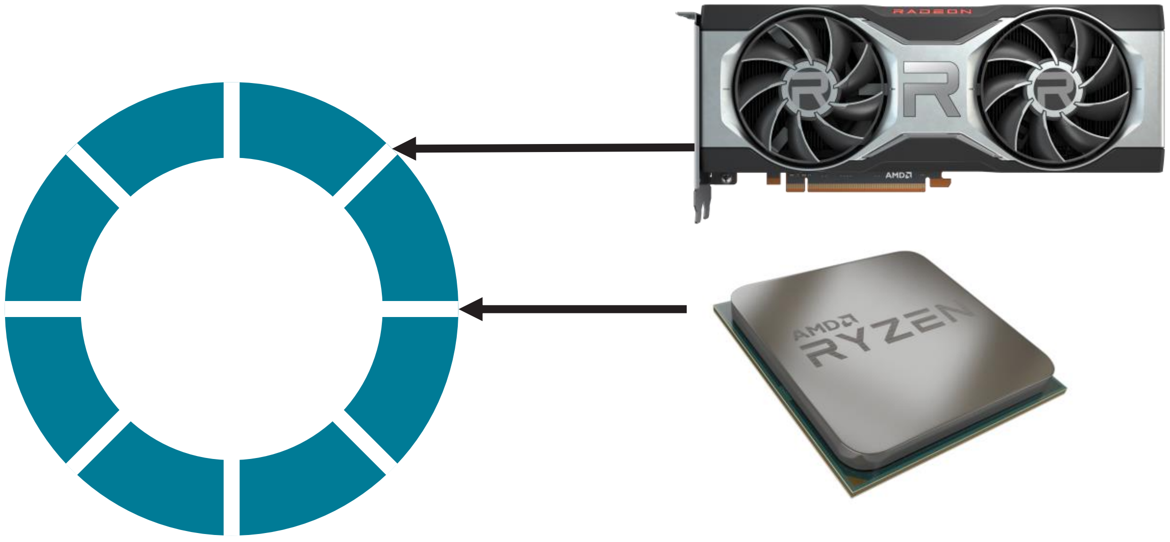


CPU → GPU COMMUNICATION



- The CPU communicates with the GPU over a *ring buffer*
- The ring buffer has two pointers:
  - The CPU writes data and moves the *write pointer* forward
  - The GPU reads data and moves the *read pointer* forward

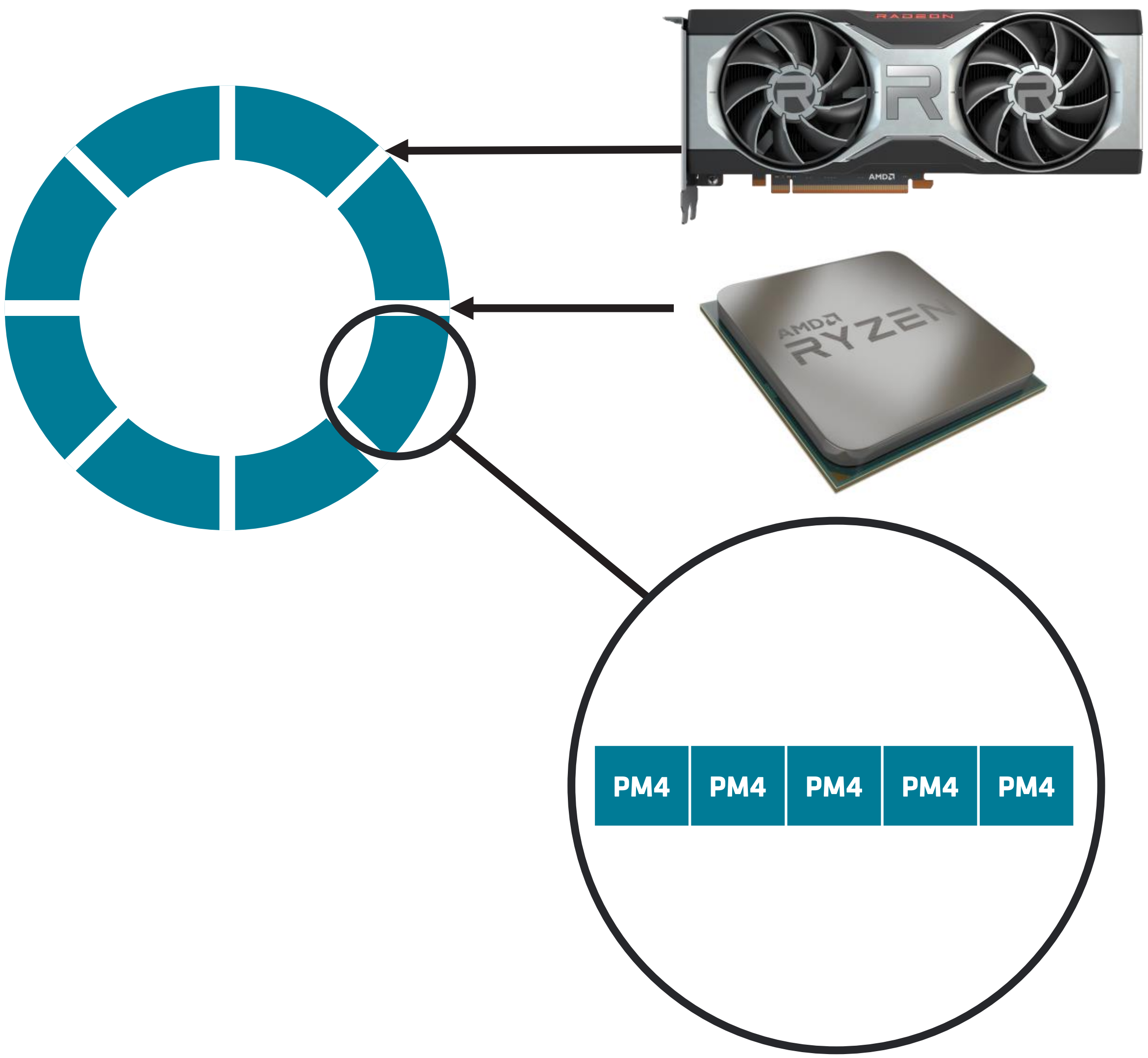
CPU → GPU COMMUNICATION



- The CPU communicates with the GPU over a *ring buffer*
- The ring buffer has two pointers:
  - The CPU writes data and moves the *write pointer* forward
  - The GPU reads data and moves the *read pointer* forward

[Public]

CPU ➔ GPU COMMUNICATION



- The ring buffer contains *PM4 packets* which contain:
  - State settings
  - Draws
  - And other things telling the GPU what to do
- PM4 packets travel over the PCIe bus
  - The *Command Processor* on the GPU receives these packets and configures the GPU accordingly

WHAT CAN WE DO WITH THIS KNOWLEDGE?

- We learnt that the driver and runtime is a few layers thick
- Some calls will end up in the kernel
  - Causing a mode switch
  - Not cheap
  - We should try to limit the amount of mode switches

- We also saw how packets with instructions are sent to the GPU
  - By writing to the ring buffer and moving pointers forward
  - That communication over the PCIe bus is not free either
  - We also mentioned the latency issues previously 😁

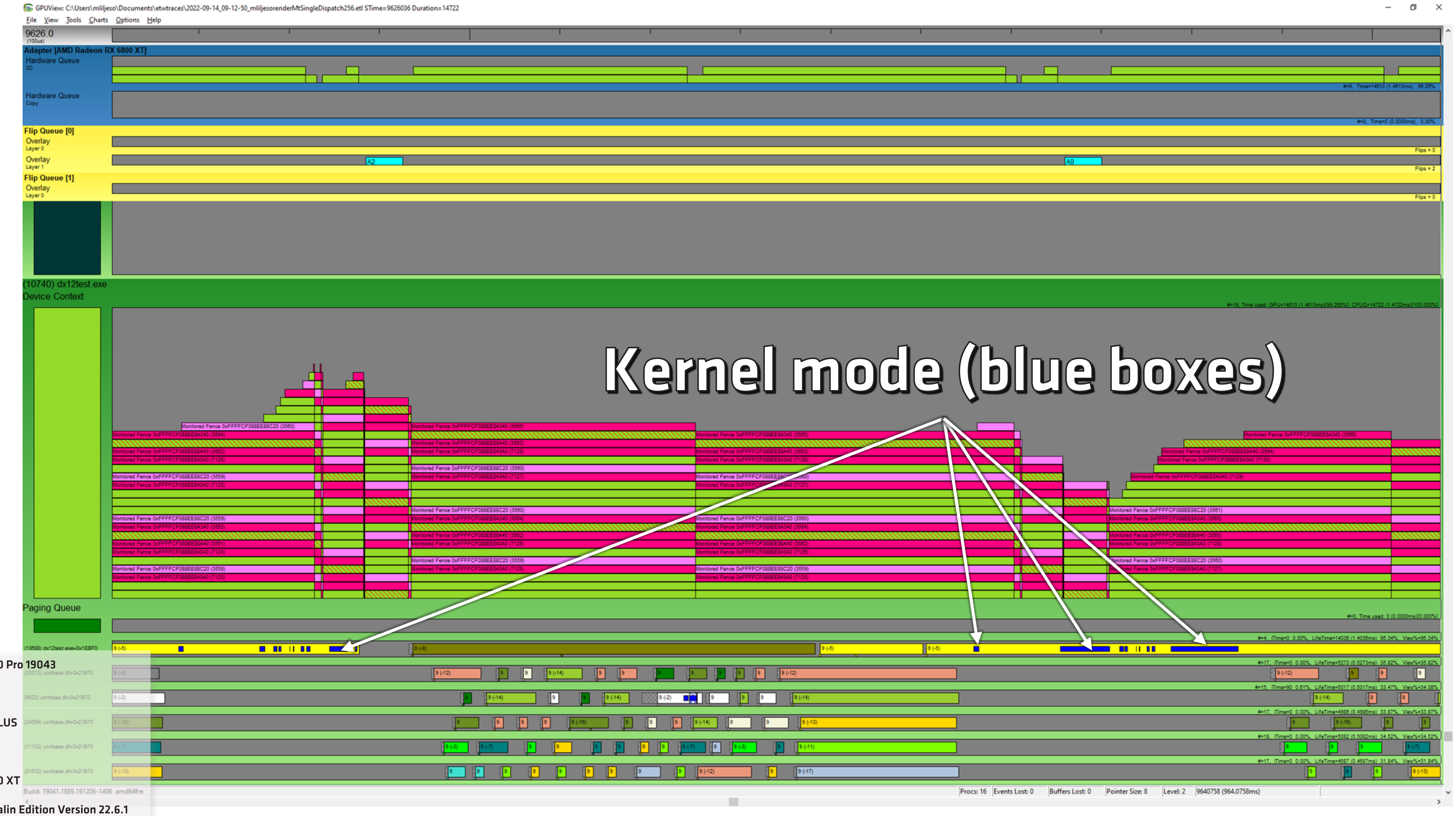
WITH MULTI THREADING, BUT WITH A SINGLE SUBMIT

Command lists with a few draws

1:	Barrier: framebuffer from present to render target Clear render target (back buffer) and depth buffer
2 – 257:	Draw 2048 boomboxes in batches of 8. (8 meshes per CL) As 256 tasks from a thread pool
258:	Barrier: framebuffer from render target to present

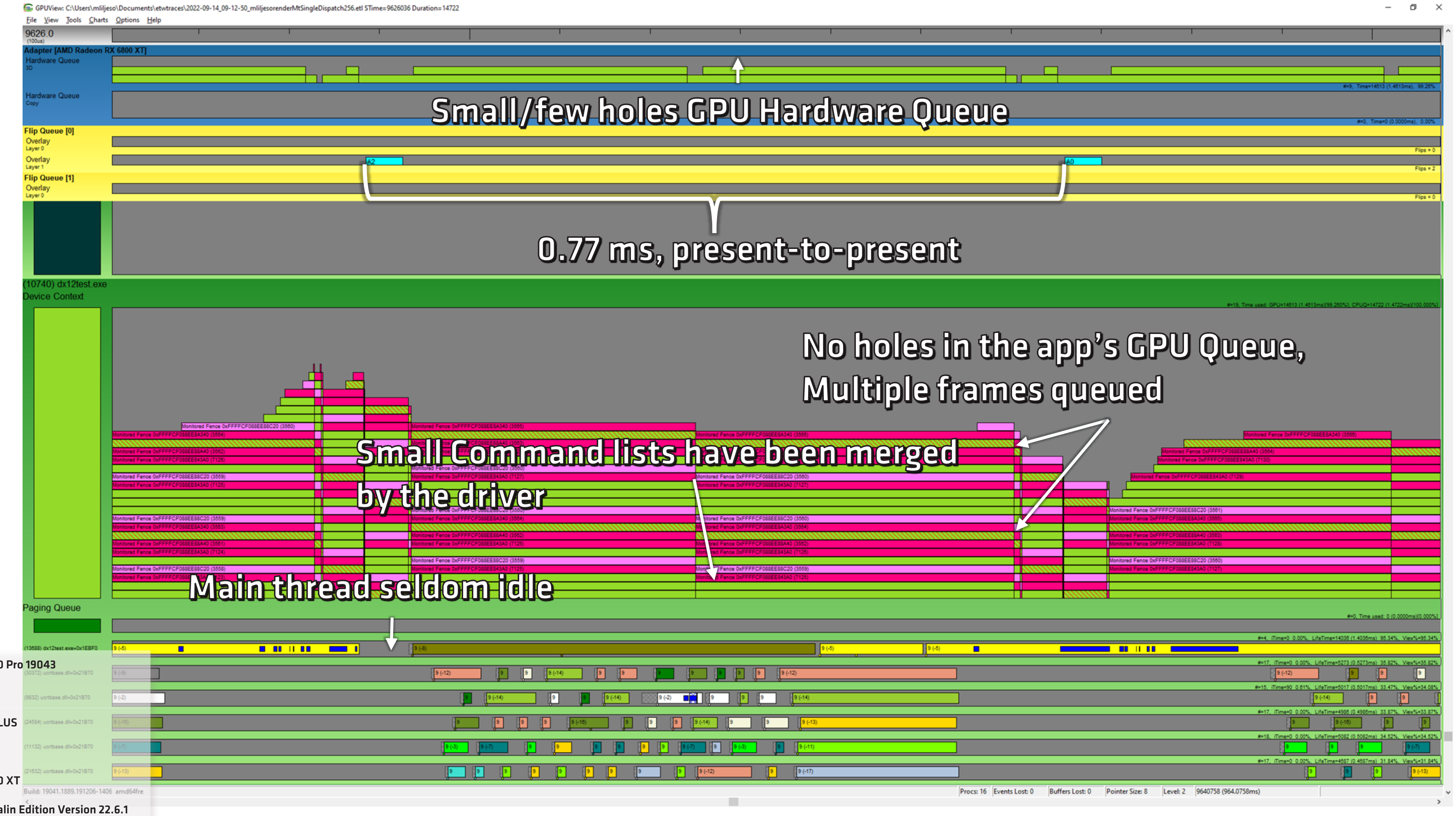
All CLs submitted in one go from the main thread

WITH MULTI THREADING, BUT WITH A SINGLE SUBMIT



OS	Microsoft Windows 10 Pro 19043
Processor	AMD Ryzen 7 5800X
Motherboard	TUF GAMING X570-PLUS
RAM	32,0 GB DDR4
GPU	AMD Radeon RX 6800 XT
Driver	AMD Software Adrenalin Edition Version 22.6.1

WITH MULTI THREADING, BUT WITH A SINGLE SUBMIT



OS	Microsoft Windows 10 Pro 19043
Processor	AMD Ryzen 7 5800X
Motherboard	TUF GAMING X570-PLUS
RAM	32,0 GB DDR4
GPU	AMD Radeon RX 6800 XT
Driver	AMD Software Adrenalin Edition Version 22.6.1

[Public]

WITH MULTI THREADING, BUT WITH A SINGLE SUBMIT

Clustered, as seen in GPUView

Low "height"

OS	Microsoft Windows 10 Pro 19043
Processor	AMD Ryzen 7 5800X
Motherboard	TUF GAMING X570-PLUS
RAM	32,0 GB DDR4
GPU	AMD Radeon RX 6800 XT
Driver	AMD Software Adrenalin Edition Version 22.6.1

BETTER, BUT STILL HORRENDOUS

WITH SANE MULTI THREADING

Command lists with a few draws

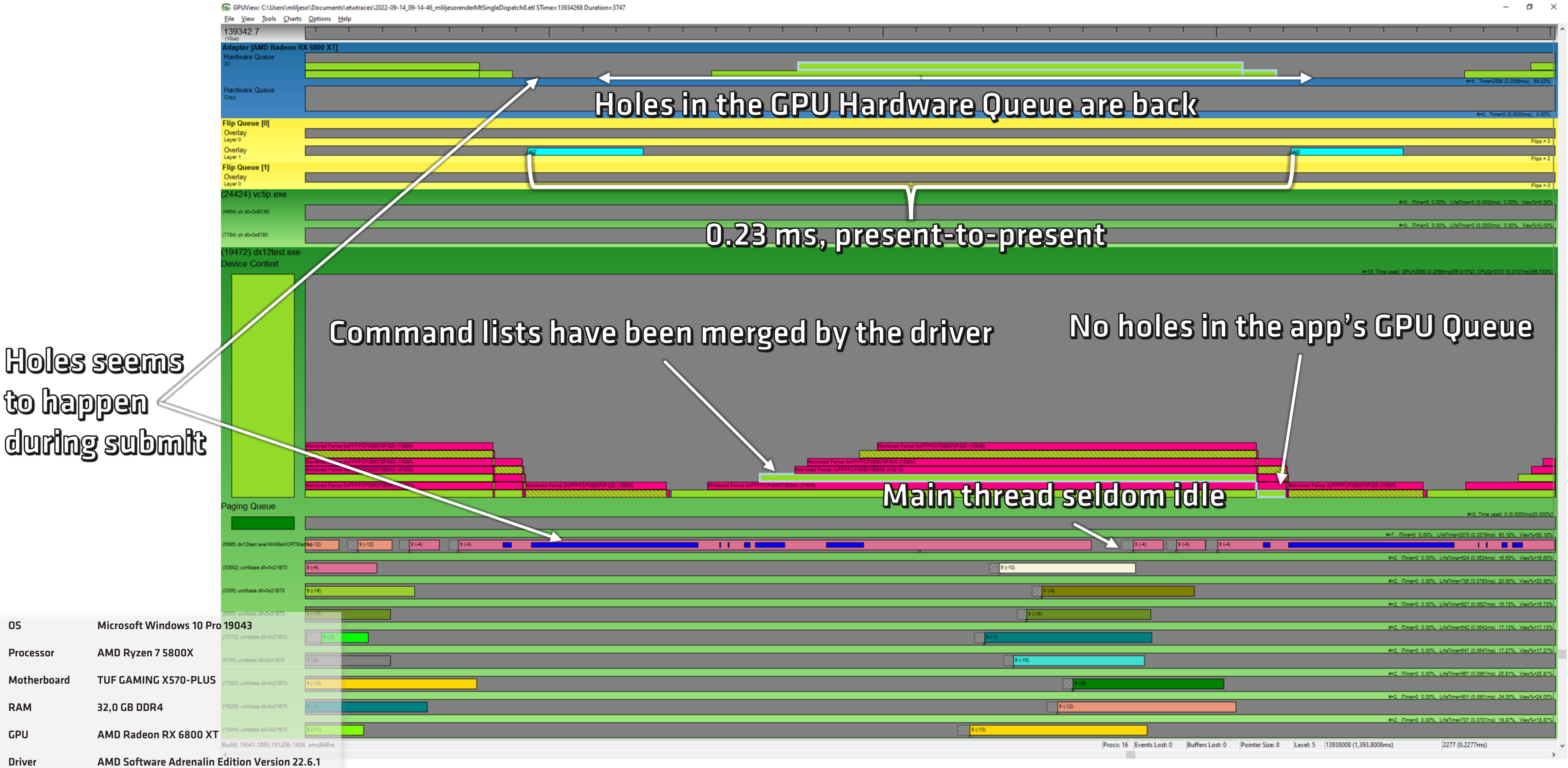
1: Barrier: framebuffer from present to render target
Clear render target (back buffer) and depth buffer

2 – 10: Draw 2048 boomboxes in batches of 256. (256 meshes per CL)
As 8 tasks from a thread pool

11: Barrier: framebuffer from render target to present

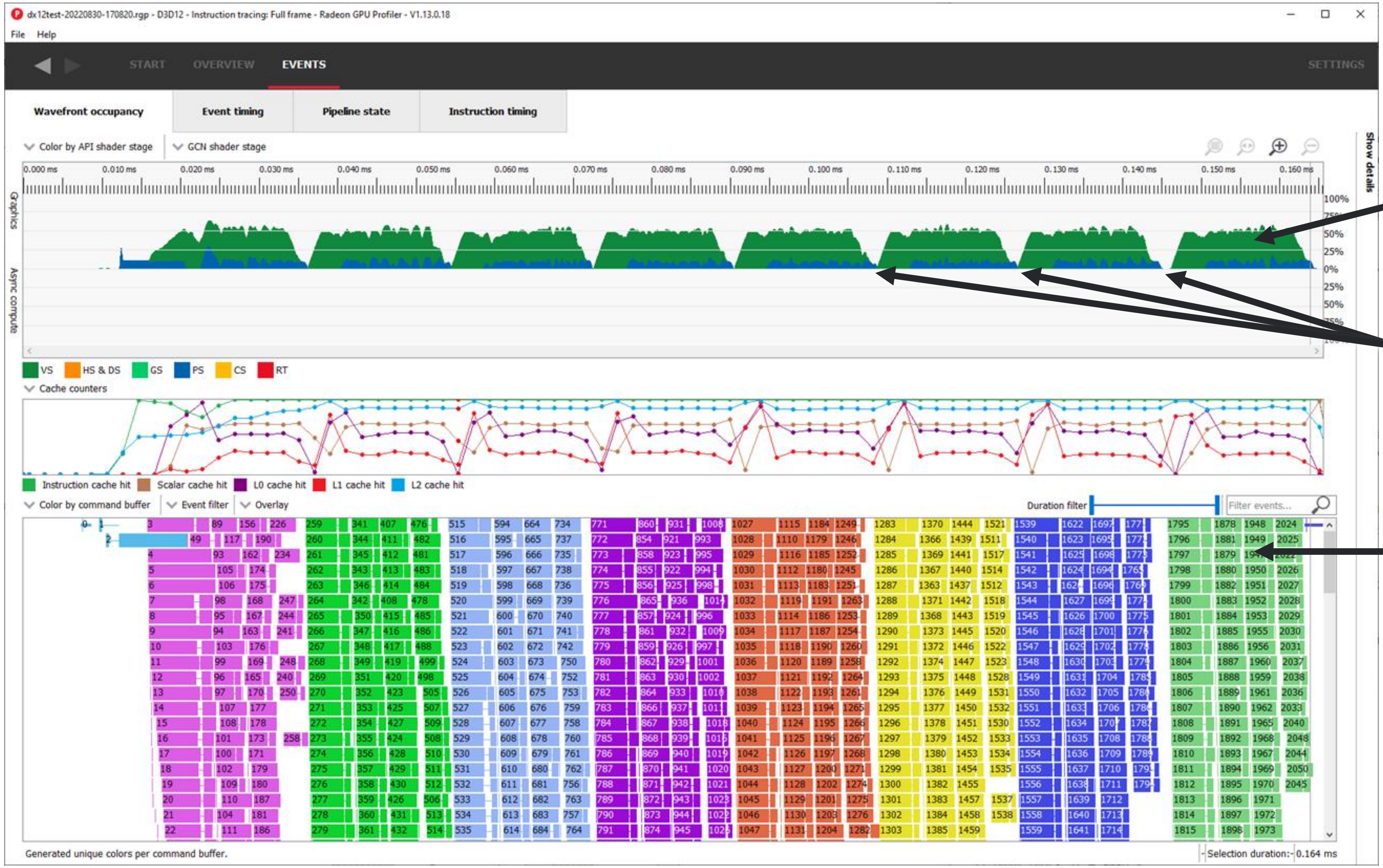
All CLs submitted in one go from the main thread

WITH SANE MULTI THREADING



OS	Microsoft Windows 10 Pro 19043
Processor	AMD Ryzen 7 5800X
Motherboard	TUF GAMING X570-PLUS
RAM	32,0 GB DDR4
GPU	AMD Radeon RX 6800 XT
Driver	AMD Software Adrenalin Edition Version 22.6.1

WITH SANE MULTI THREADING



Occupancy up again

Still visible dips in the occupancy view

Parallelism again

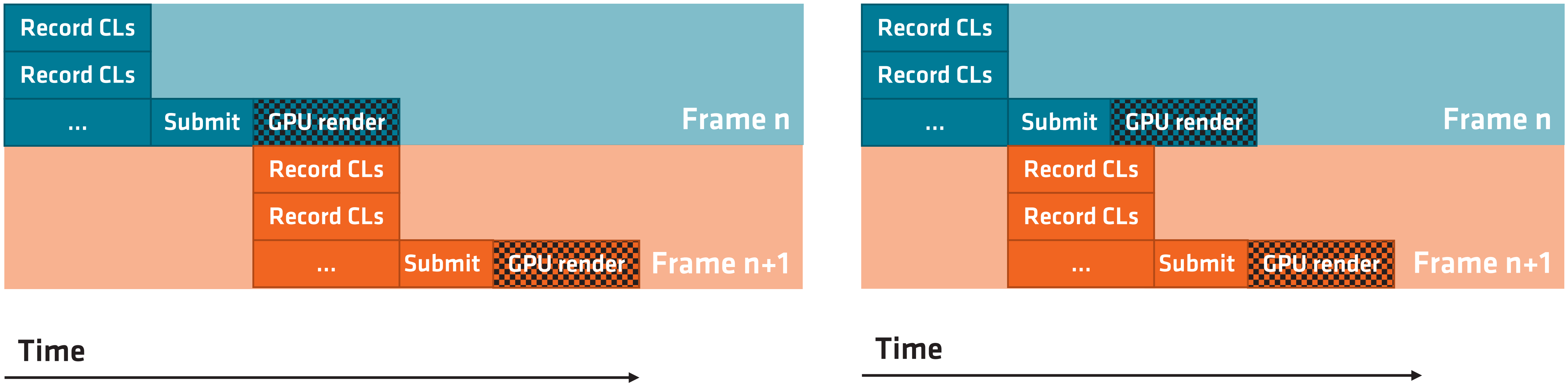
OS	Microsoft Windows 10 Pro 19043
Processor	AMD Ryzen 7 5800X
Motherboard	TUF GAMING X570-PLUS
RAM	32,0 GB DDR4
GPU	AMD Radeon RX 6800 XT
Driver	AMD Software Adrenalin Edition Version 22.6.1

[Public]

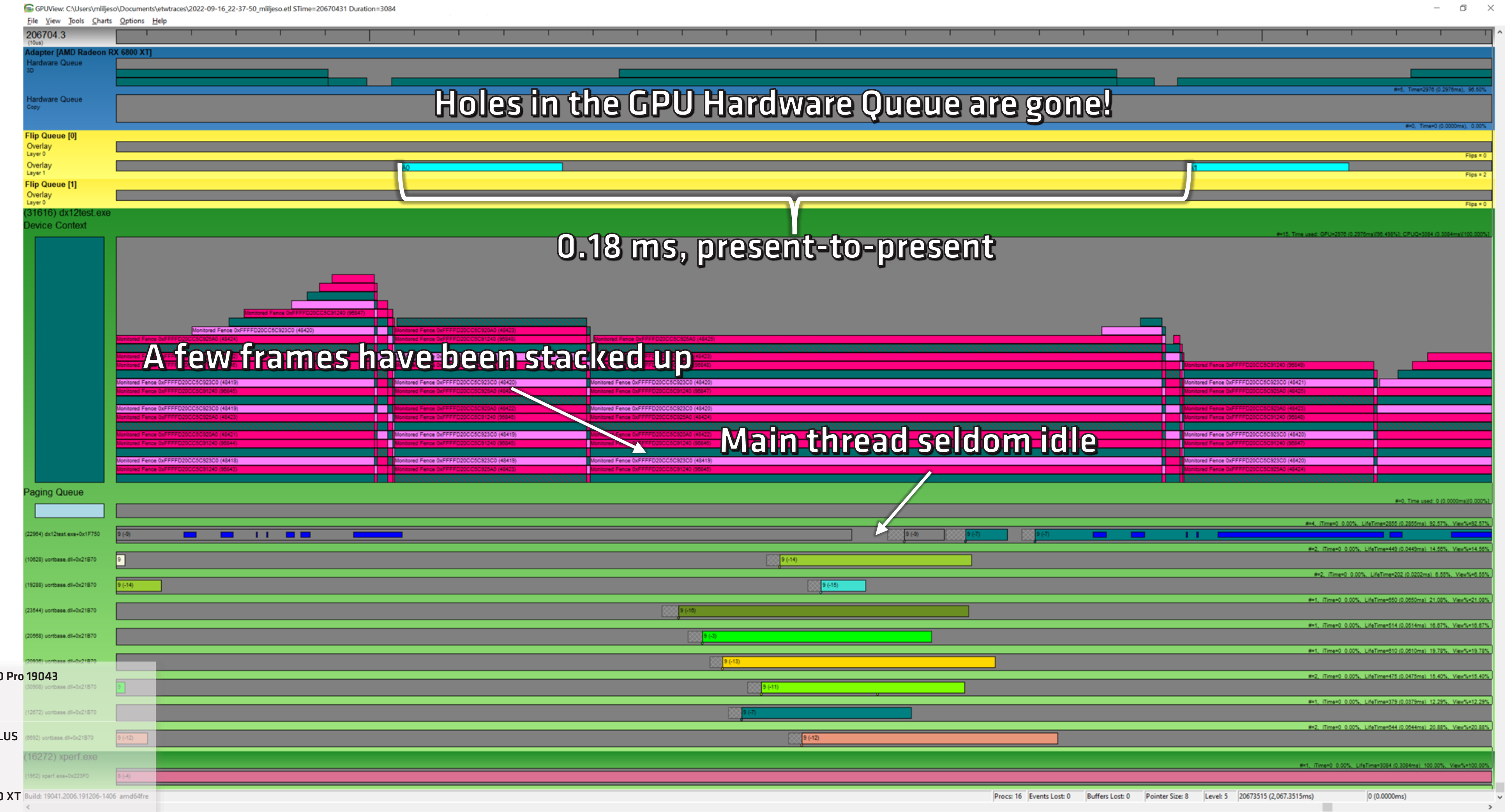
WITH SUBMIT IN PARALLEL WITH CL RECORDING

What we've been doing so far

With async submit



WITH SUBMIT IN PARALLEL WITH CL RECORDING



OS	Microsoft Windows 10 Pro 19043
Processor	AMD Ryzen 7 5800X
Motherboard	TUF GAMING X570-PLUS
RAM	32,0 GB DDR4
GPU	AMD Radeon RX 6800 XT
Driver	AMD Software Adrenalin Edition Version 22.6.1

WITH SUBMIT IN PARALLEL WITH CL RECORDING

OS	Microsoft Windows 10 Pro 19043
Processor	AMD Ryzen 7 5800X
Motherboard	TUF GAMING X570-PLUS
RAM	32,0 GB DDR4
GPU	AMD Radeon RX 6800 XT
Driver	AMD Software Adrenalin Edition Version 22.6.1

SOME CONCLUSIONS

- Submission strategy can affect performance
- Submit size matters: Avoid small CLs
  - That will keep the GPU busy
- Batch CLs up for submission
  - Not one-by-one
- Take advantage of multithreaded command list recording
  - When you see a benefit from it
- Kick off next frame CL recording early

- If unsure of good API usage, take a look at our performance guides on GPUOpen!
  - https://gpuopen.com/performance/
  - https://gpuopen.com/ryzen-performance/

THANKS

- Lou Kramer
- Gareth Thomas
- Dominik Baumeister
- Matthäus Chajdas

QUESTIONS?

mattias.liljeson@amd.com

DISCLAIMER & ATTRIBUTES

Disclaimer:

AMD