Wine Development Updates, Performance and the D3D9 State Tracker

Stefan Dösinger
stefandoesinger@gmail.com

Outline

- Progress report
- Why we think the d3d9 state tracker is a bad idea
- Wishlist / Interface ideas

Wine Updates

- Fullscreen focus loss handling
- Continued work on d3d10
- Multithreaded command stream stalled
- Performance monitoring updates

Focus Handling

- Switch away from fullscreen d3d window
  - Minimize, restore resolution
- And back on focus restore
- Works on OSX, KDE, FVWM
- Not yet on Metacity forks and Compiz
  - They Refuse XIconifyWindow without MWM_FUNC_MINIMIZE
- Semi-related: Resolution on game crash

D3D 10/11 Status

- Incremental progress
- Recently implemented texture sampling
- Still missing: D3D10 style resource handling
  - Can't sample from buffers
  - Format reinterpretation
- D2D and DirectWrite on top of D3D10
  - Used by Microsoft Office 2013

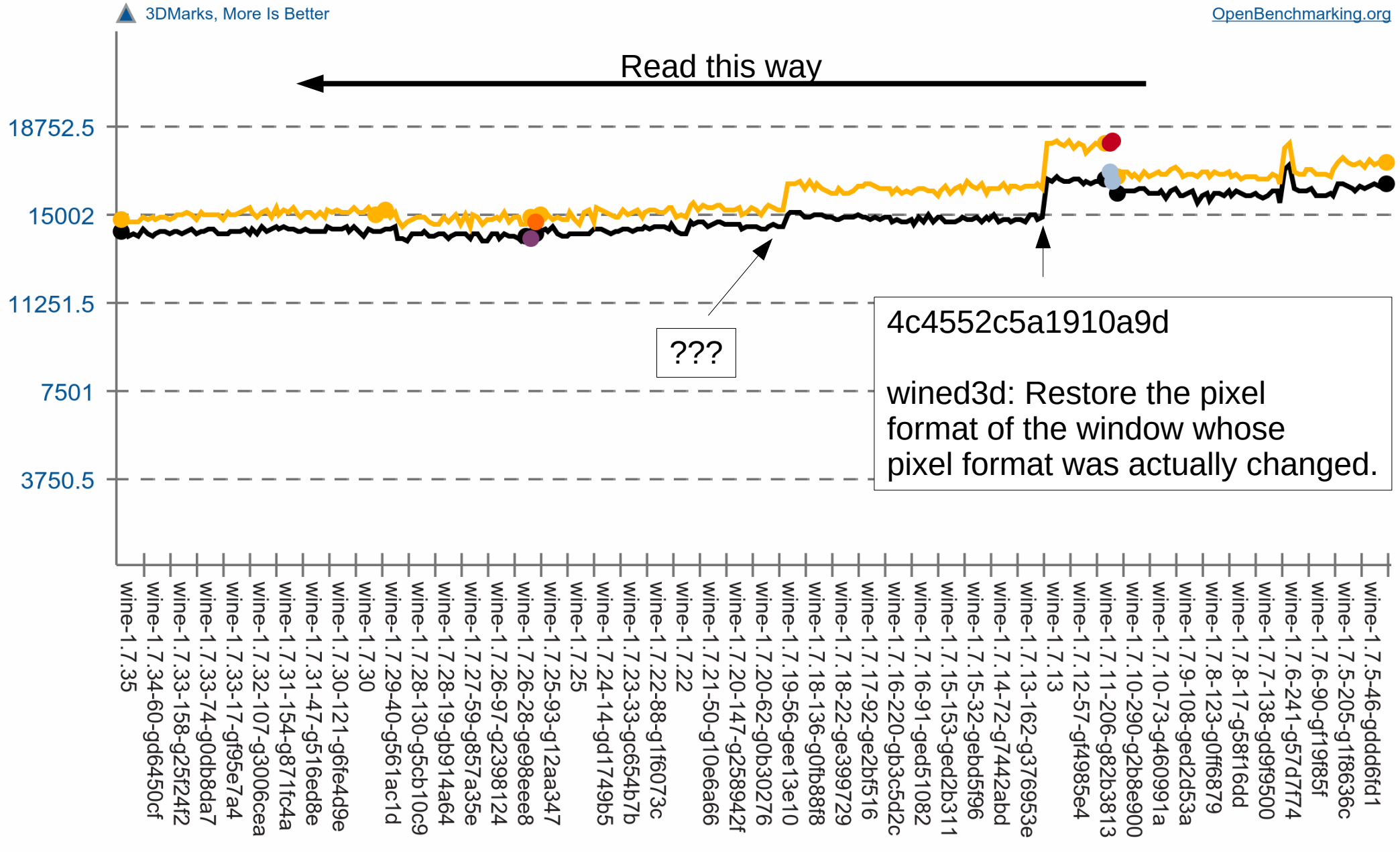
Core Contexts

- Finally working on it
- Needed for d3d10 on some HW
- Hopefully makes things easier for drivers
- ETA: A month or two
  - We're good at missing deadlines
  - Sticking point: ddraw blitters and color keys

Command Stream Status

- Blocked on d3d10 resource changes

Performance Monitoring – r600g
3DMark2000



Development Environment

- Henri is using r600g
- I am mostly using Nvidia Blob
- Work at CodeWeavers forces us to keep an eye on OSX
- r200, r300g, Geforce <= 7 bitrotting
- Intel? Not really tested, few bug reports
  - It either works or people gave up

D3d9 state tracker

D3d9 state tracker

- We see it as a testing / debugging tool rather than a long-term solution
- Main Problem: Massive code duplication for one corner case

API / GPU / OS

	ddraw	d3d8	d3d9
Nvidia GF4	Linux, OSX, (Win)	Linux, OSX, (Win)	Linux, OSX, (Win)
Nvidia GF7	Linux, OSX, (Win)	Linux, OSX, (Win)	Linux, OSX, (Win)
Nvidia GF8+	Linux, OSX, (Win)	Linux, OSX, (Win)	Linux, OSX, (Win)
r200	Linux, OSX, (Win)	Linux, OSX, (Win)	Linux, OSX, (Win)
r500	Linux, OSX, (Win)	Linux, OSX, (Win)	Linux, OSX, (Win)
r600+	Linux, OSX, (Win)	Linux, OSX, (Win)	Linux, OSX, (Win)
i915	Linux, OSX, (Win)	Linux, OSX, (Win)	Linux, OSX, (Win)
i945	Linux, OSX, (Win)	Linux, OSX, (Win)	Linux, OSX, (Win)
i965+	Linux, OSX, (Win)	Linux, OSX, (Win)	Linux, OSX, (Win)

API / GPU / OS

	ddraw	d3d8	d3d9
Nvidia GF4	Linux, OSX, (Win)	Linux, OSX, (Win)	Linux, OSX, (Win)
Nvidia GF7	Linux, OSX, (Win)	Linux, OSX, (Win)	Linux, OSX, (Win)
Nvidia GF8+	Linux, OSX, (Win)	Linux, OSX, (Win)	Linux, OSX, (Win)
r200	Linux, OSX, (Win)	Linux, OSX, (Win)	Linux, OSX, (Win)
r500	Linux, OSX, (Win)	Linux, OSX, (Win)	Linux, OSX, (Win)
r600+	Linux, OSX, (Win)	Linux, OSX, (Win)	Linux, OSX, (Win)
i915	Linux, OSX, (Win)	Linux, OSX, (Win)	Linux, OSX, (Win)
i945	Linux, OSX, (Win)	Linux, OSX, (Win)	Linux, OSX, (Win)
i965+	Linux, OSX, (Win)	Linux, OSX, (Win)	Linux, OSX, (Win)

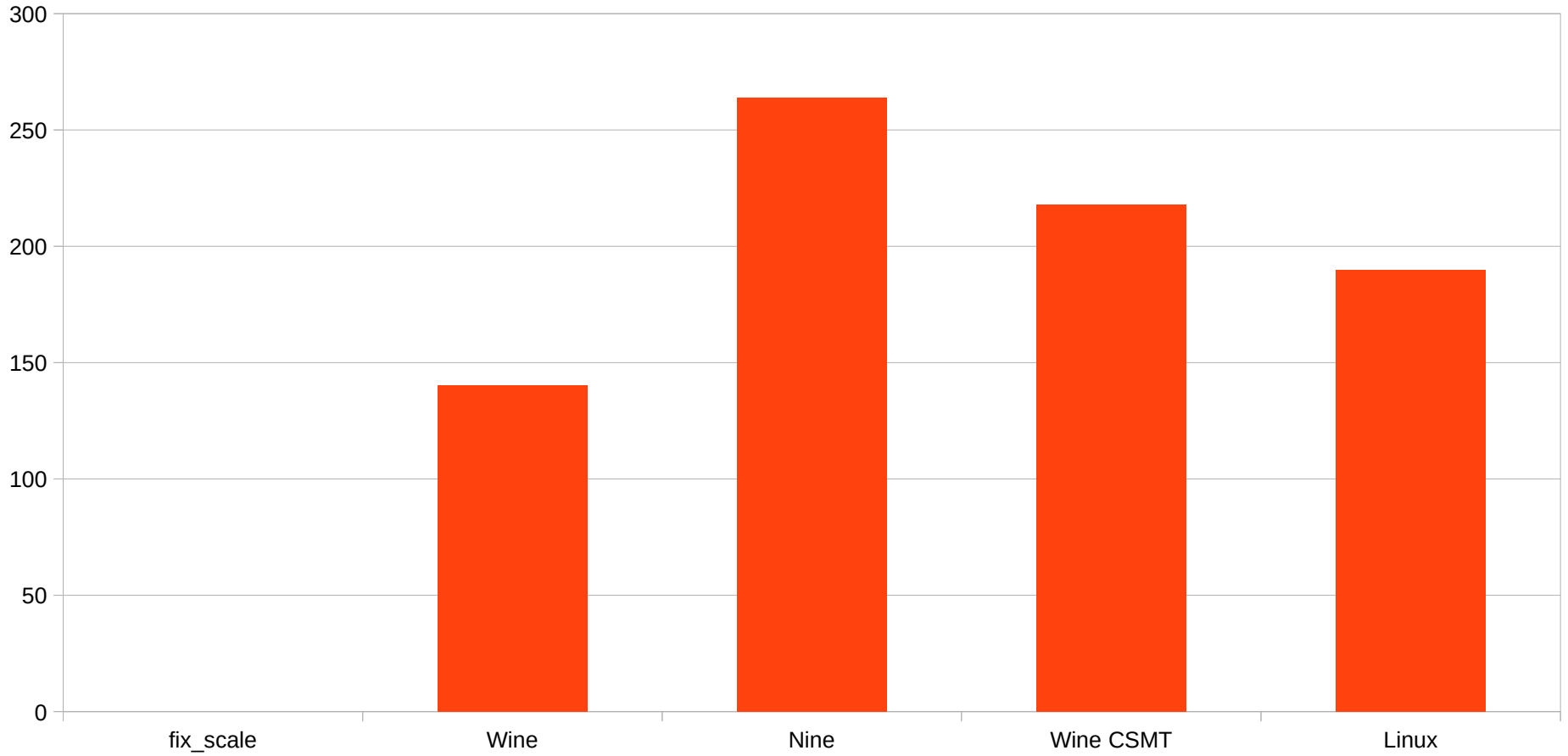
D3d9 state tracker

- We see it as a testing / debugging tool rather than a long-term solution
- Main Problem: Massive code duplication for one corner case
  - wined3d + d3d9 + d3d8 + ddraw: 80,000 LOC
  - Nine: 25,000 LOC for just d3d9 on Radeon on Linux
- Integration issues
- Doesn't solve the actual problems

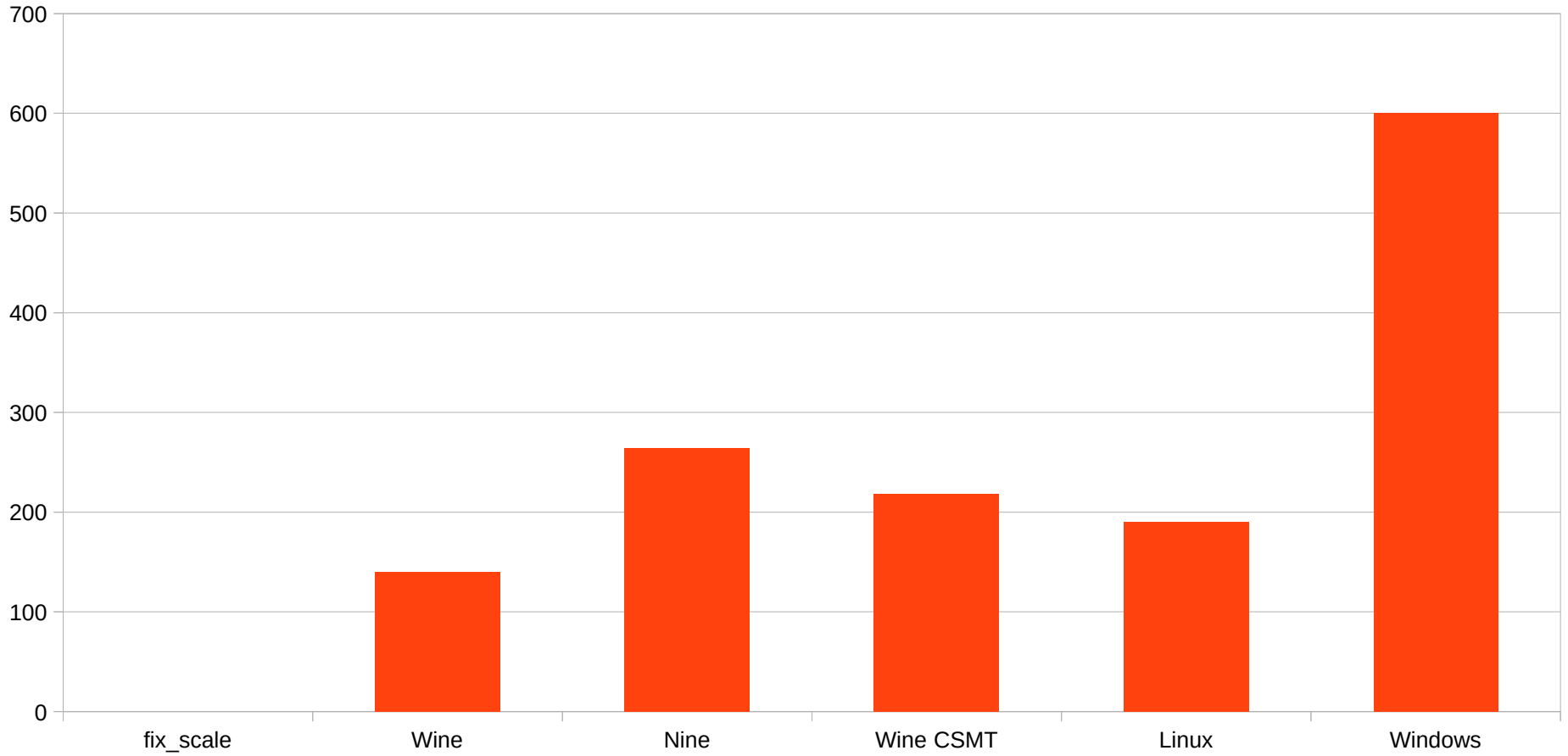
Test Machine

- Intel Core i7
- Radeon HD 5770
  - Mesa git from January 2015
- Geforce GTX 460
  - Nvidia 346.35 blob
- 16 GB RAM
- Windows 7, Gentoo

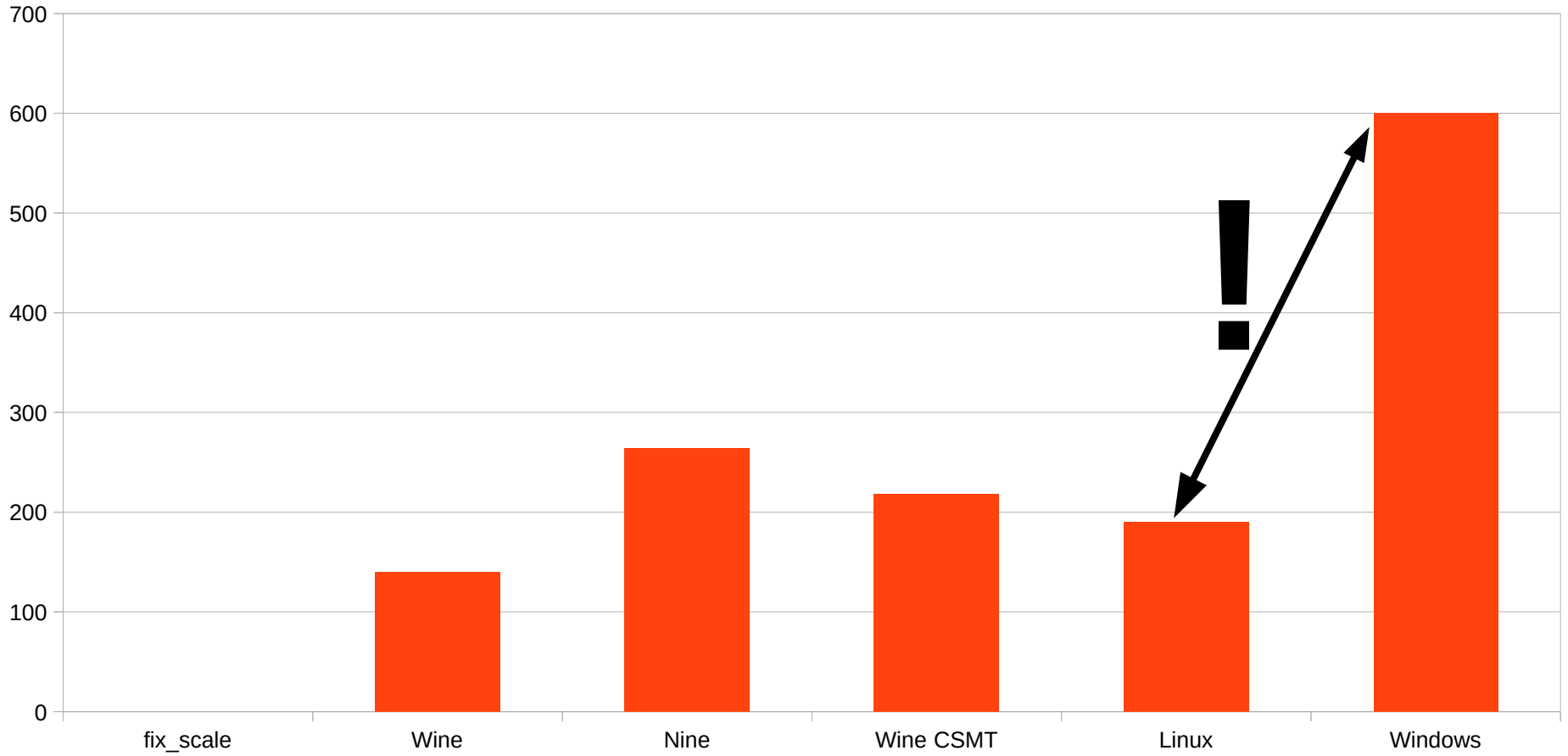
Example: Half Life 2



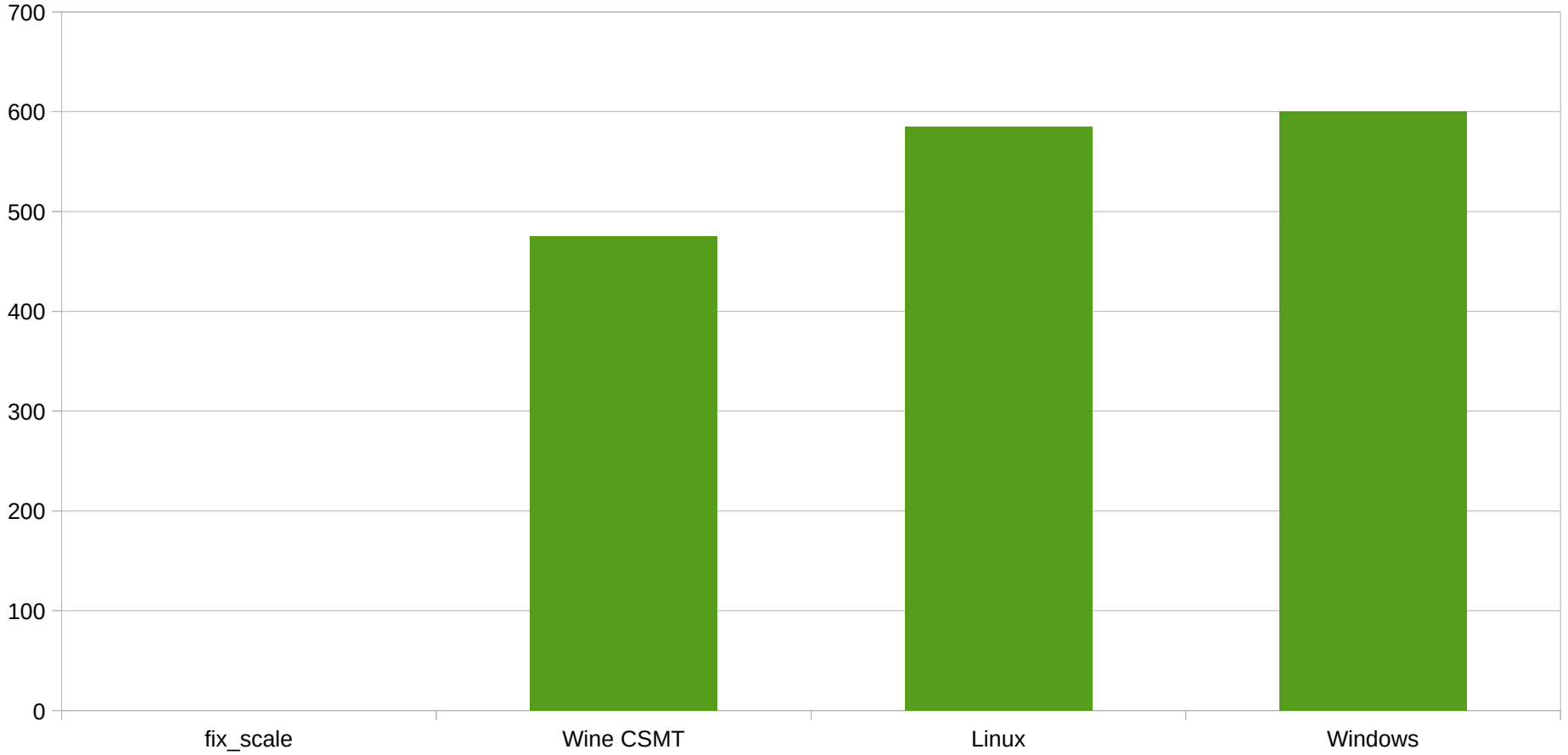
Example: Half Life 2



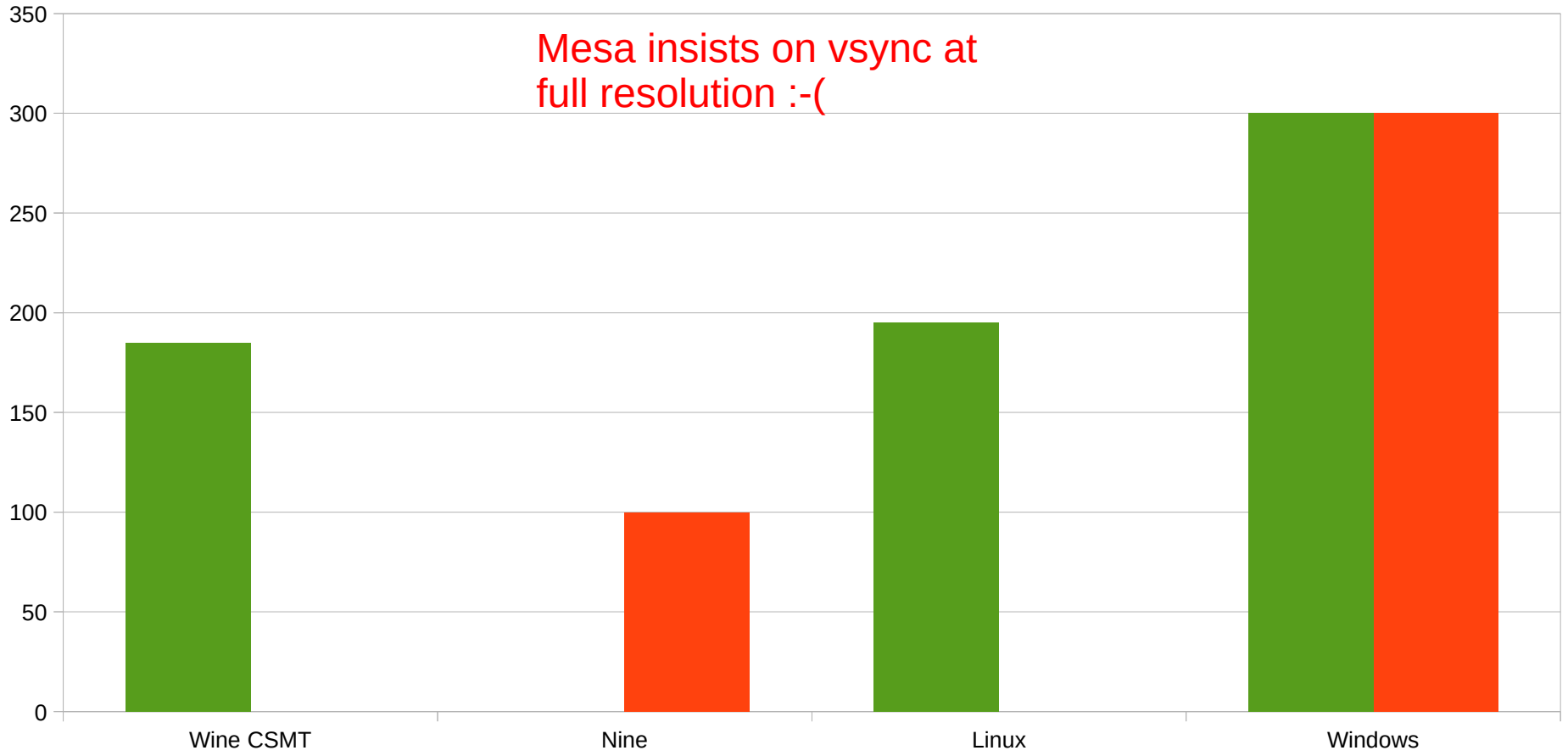
Example: Half Life 2



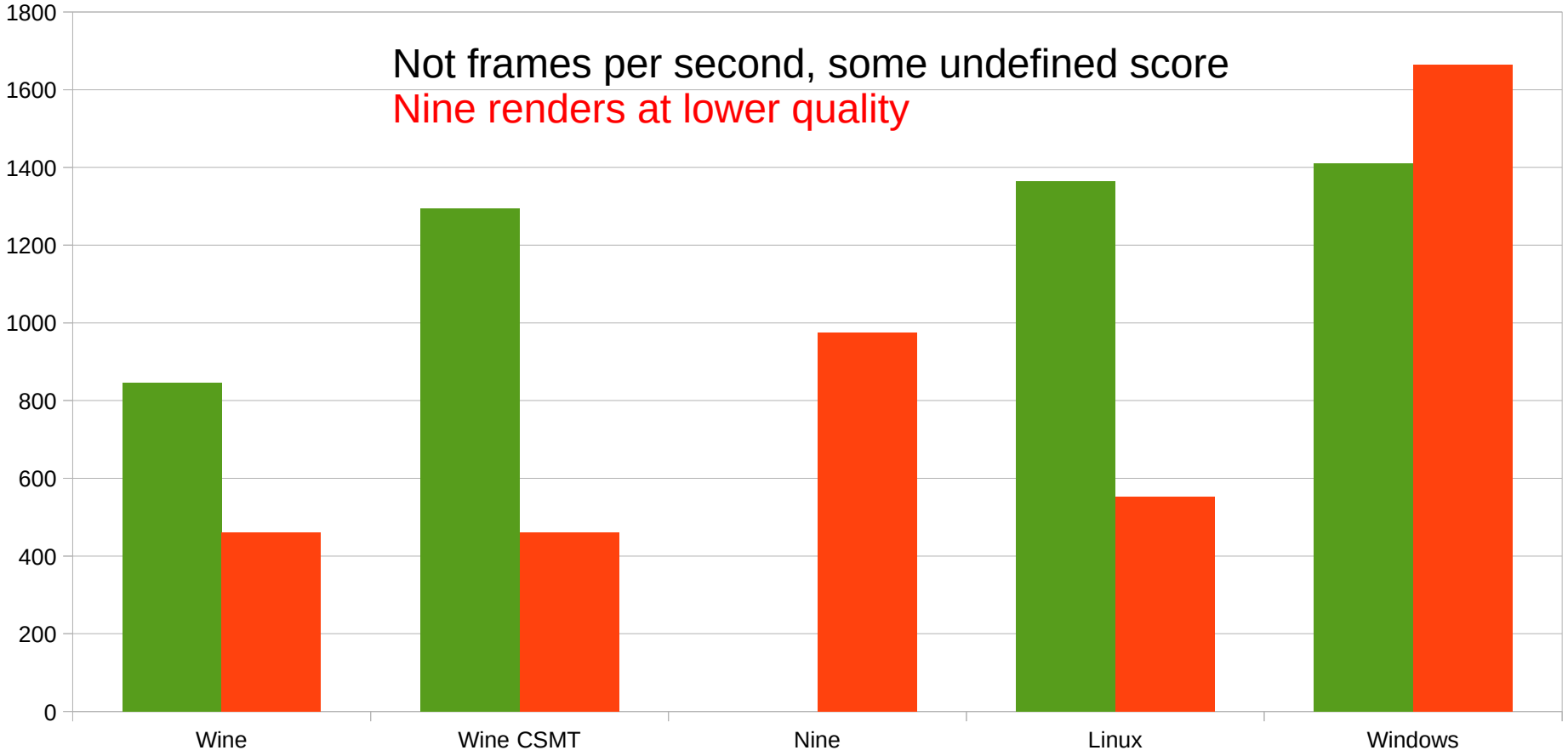
NV Blob: Fast OpenGL is possible



HL2 GPU Limited



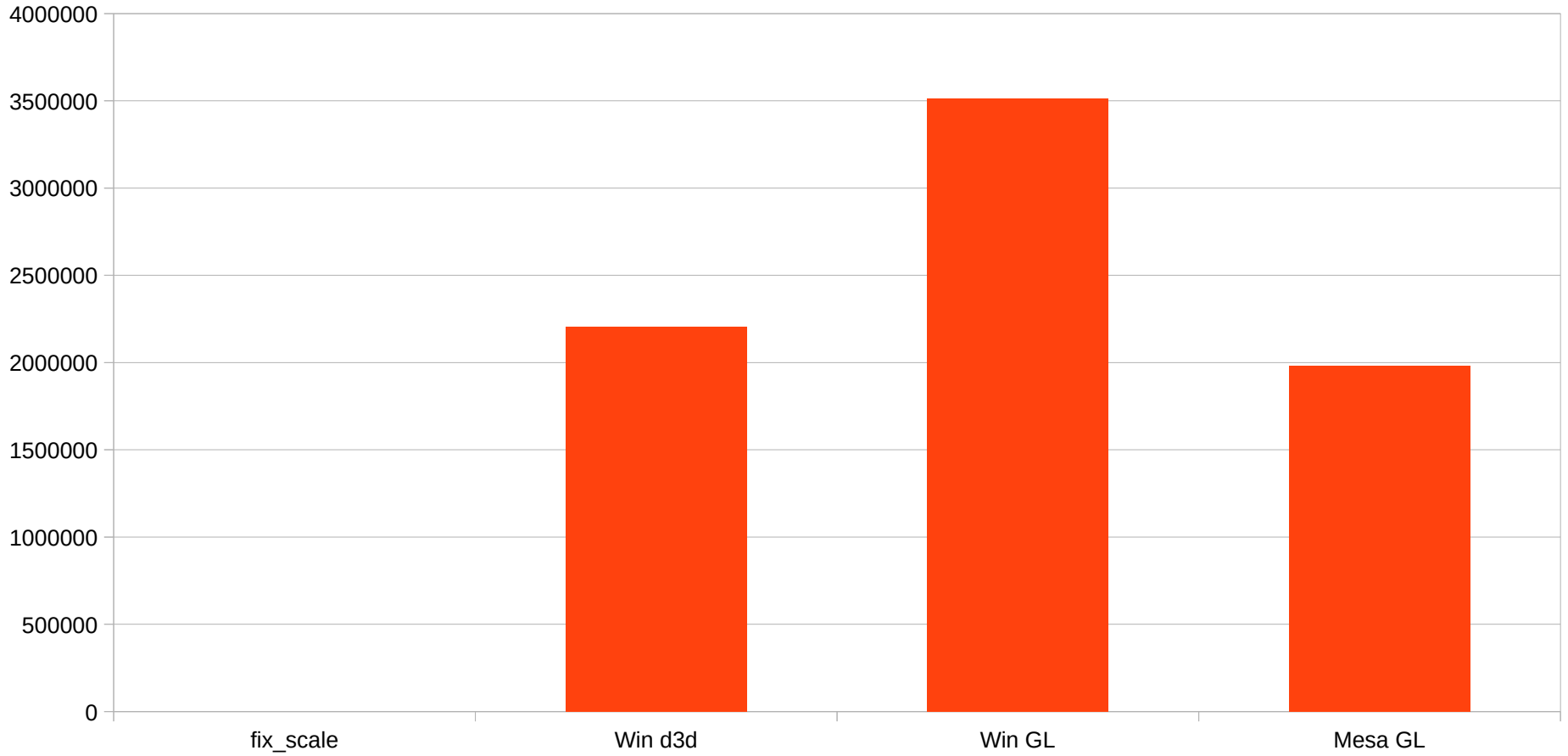
Civilization V



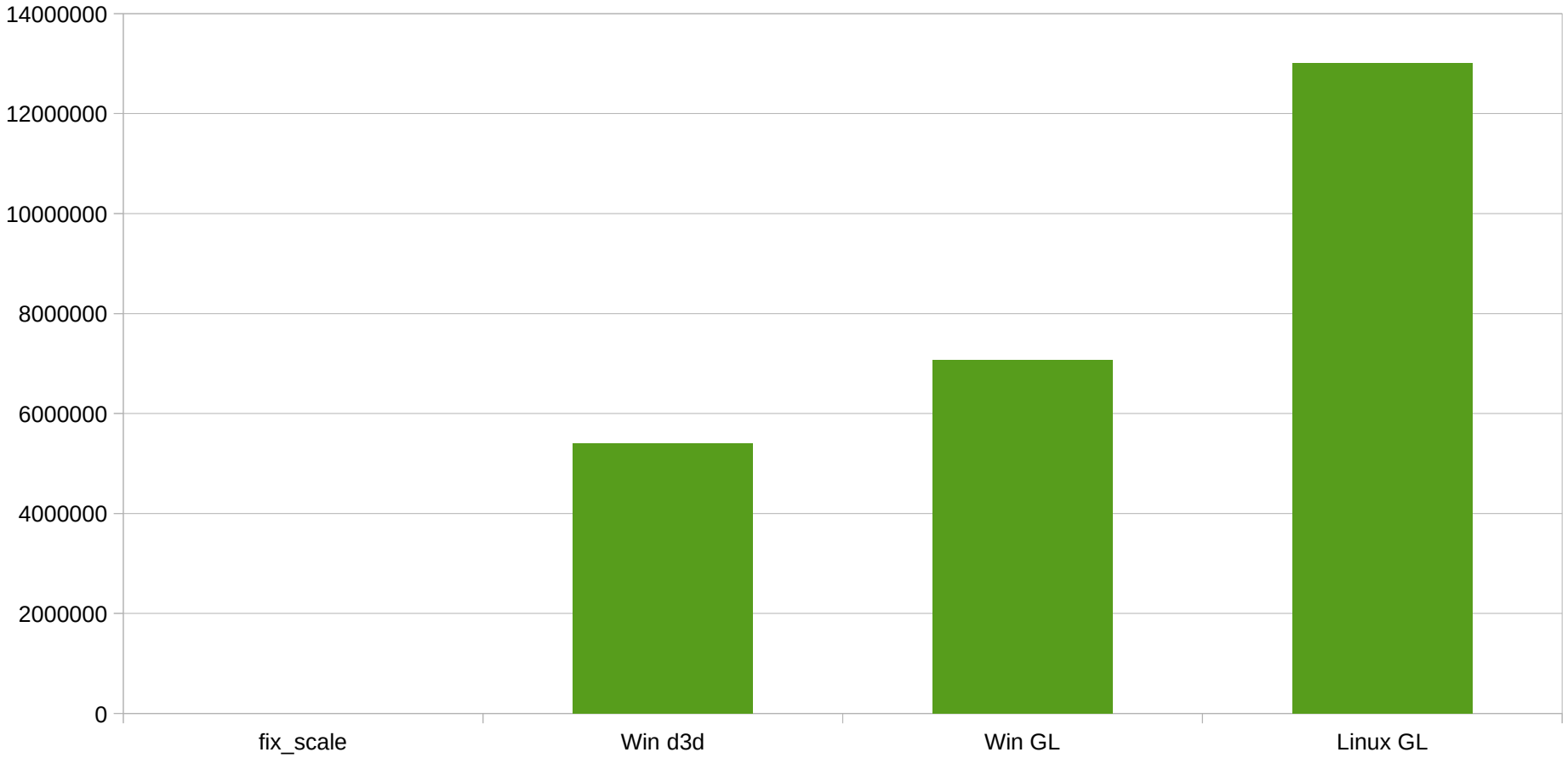
Lower Draw Overhead

- It is possible with OpenGL
- No need for Nine, Mantle or other wheel reinventions
- Not the holy grail

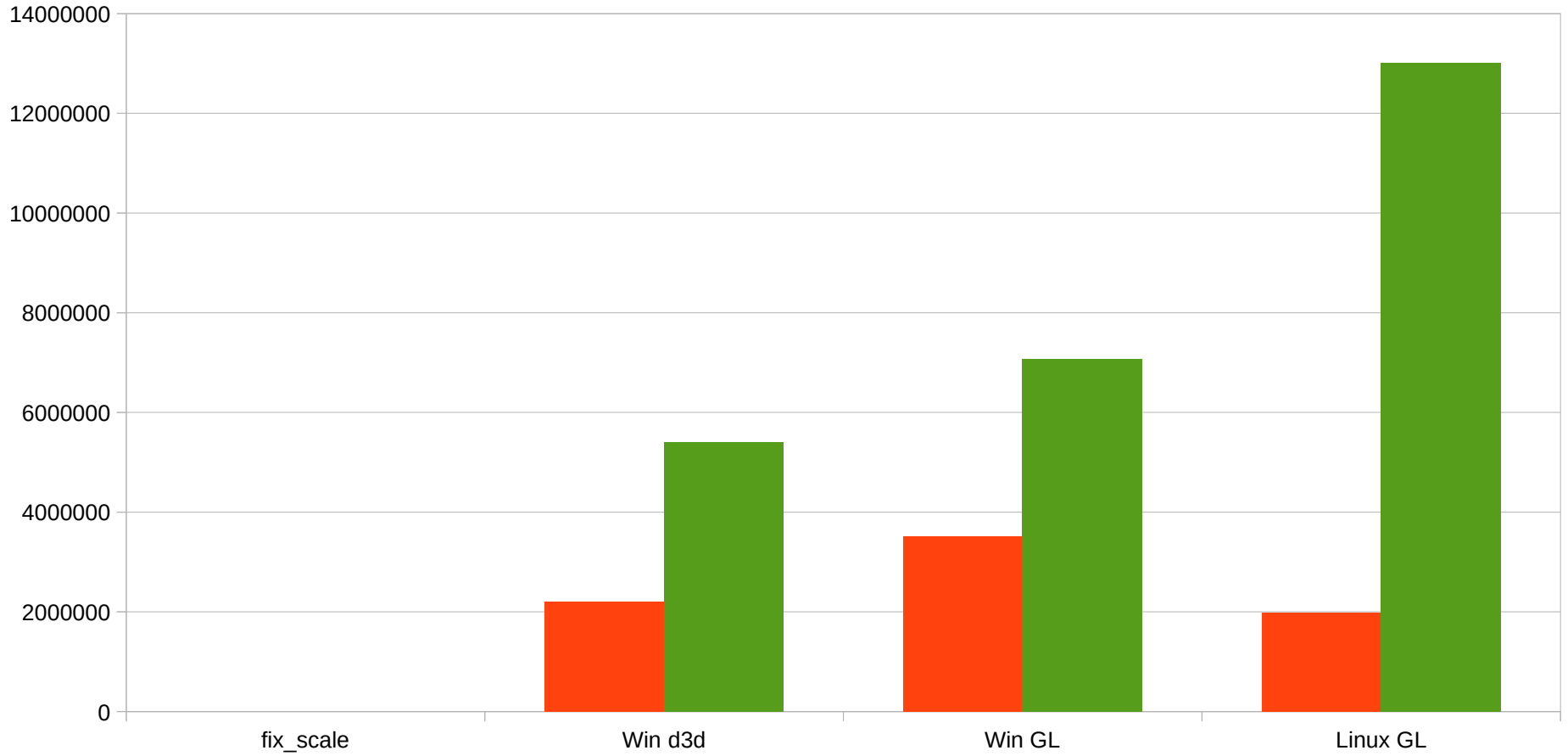
r600g draws / sec



Nvidia draws / sec



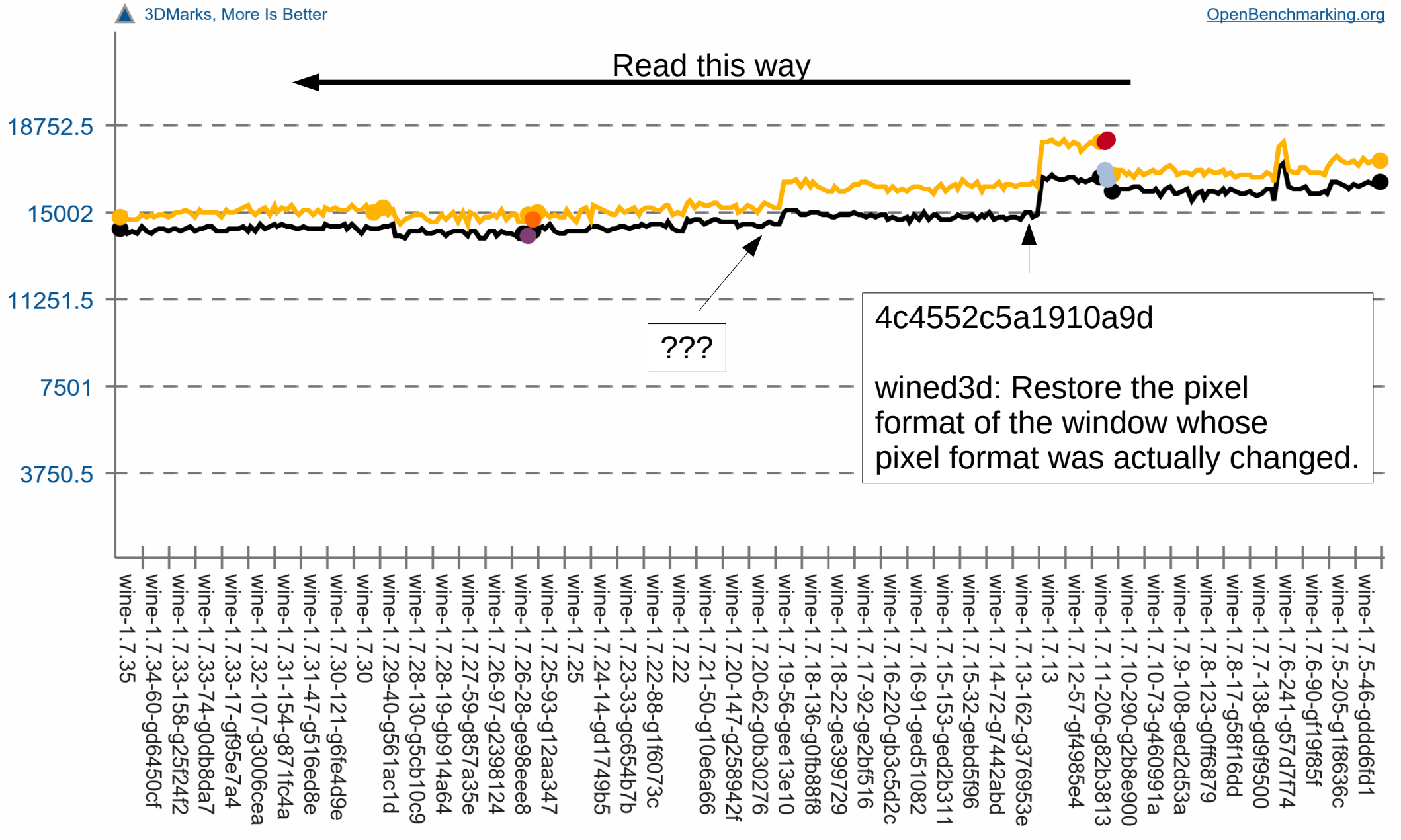
Nvidia vs AMD



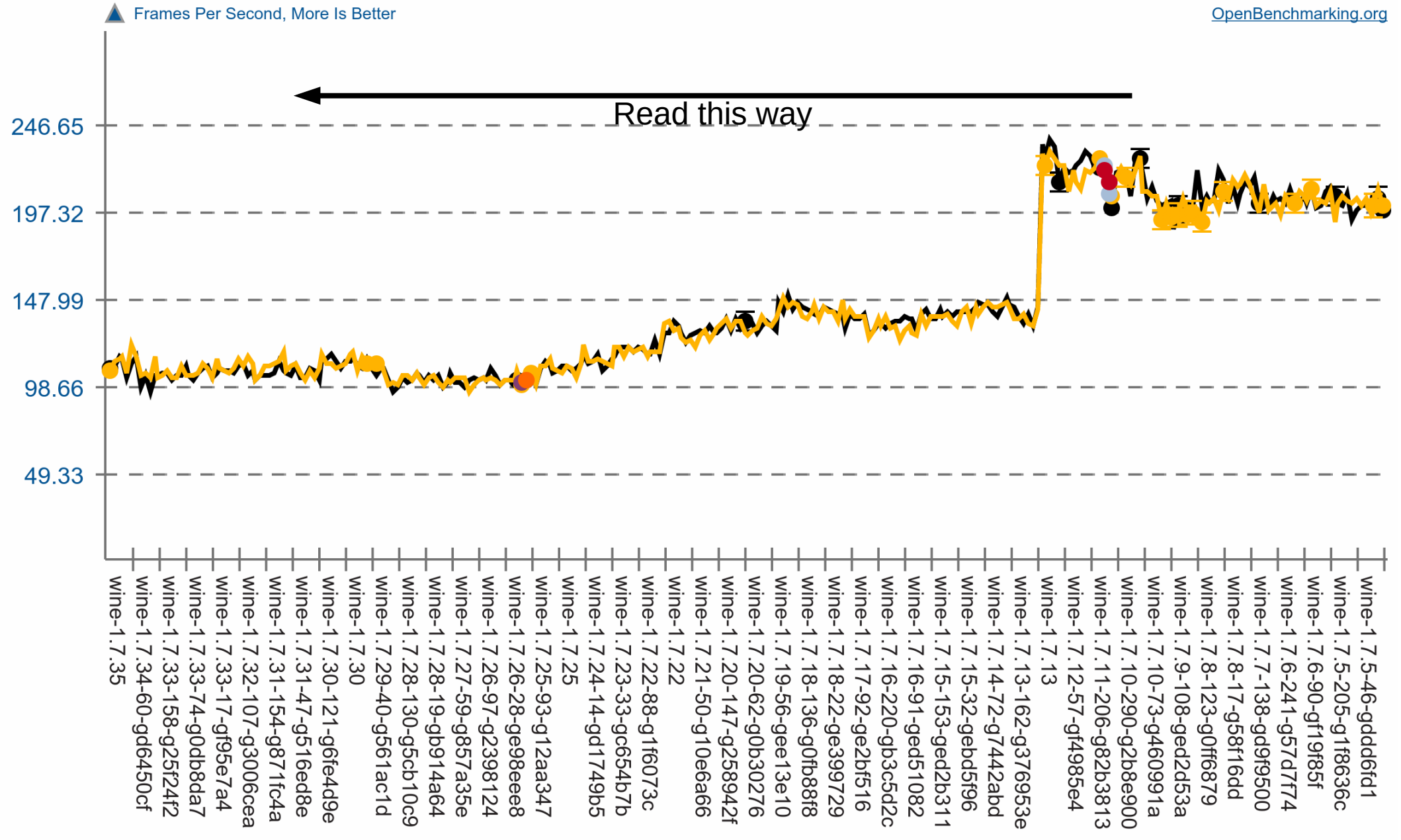
Lower Draw Overhead

- It is possible with OpenGL
- No need for Nine, Mantle or other wheel reinventions
- Not the holy grail
  - Otherwise glxgears would be a benchmark
- But it correlates to real game performance

3DMark2000


3DMarks, More Is Better
OpenBenchmarking.org
Read this way
18752.5
15002
11251.5
7501
3750.5
???
4c4552c5a1910a9d
wined3d: Restore the pixel format of the window whose pixel format was actually changed.
wine-1.7.35
wine-1.7.34-60-gd6450cf
wine-1.7.33-158-g25f24f2
wine-1.7.33-74-g0db8da7
wine-1.7.33-17-gf95e7a4
wine-1.7.32-107-g3006cea
wine-1.7.31-154-g871fc4a
wine-1.7.31-47-g516ed8e
wine-1.7.30-121-g6fe4d9e
wine-1.7.30
wine-1.7.29-40-g561ac1d
wine-1.7.28-130-g5cb10c9
wine-1.7.28-19-gb914a64
wine-1.7.27-59-g857a35e
wine-1.7.26-97-g2398124
wine-1.7.26-28-ge98eee8
wine-1.7.25-93-g12aa347
wine-1.7.25
wine-1.7.24-14-gd1749b5
wine-1.7.23-33-gc654b7b
wine-1.7.22-88-g1f6073c
wine-1.7.22
wine-1.7.21-50-g10e6a66
wine-1.7.20-147-g258942f
wine-1.7.20-62-g0b30276
wine-1.7.19-56-gee13e10
wine-1.7.18-136-g0fb88f8
wine-1.7.18-22-ge399729
wine-1.7.17-92-ge2bf516
wine-1.7.16-220-gb3c5d2c
wine-1.7.16-91-ged51082
wine-1.7.15-153-ged2b311
wine-1.7.15-32-gebd5f96
wine-1.7.14-72-g7442abd
wine-1.7.13-162-g376953e
wine-1.7.13
wine-1.7.12-57-gf4985e4
wine-1.7.11-206-g82b3813
wine-1.7.10-290-g2b8e900
wine-1.7.10-73-g460991a
wine-1.7.9-108-ged2d53a
wine-1.7.8-123-g0ff6879
wine-1.7.8-17-g58f16dd
wine-1.7.7-138-gd9f9500
wine-1.7.6-241-g57d7f74
wine-1.7.6-90-gf19f85f
wine-1.7.5-205-g1f8636c
wine-1.7.5-46-gddd6fd1

DrawPrimitive()



Wishlist

GLSL Compile Time

- D3D apps expect shader creation to be FAST
- Wine can improve some corner cases
  - GL_ARB_separate_shader_objects
  - Compile at creation with reasonable assumptions
- Some applications create shaders on the fly
  - So GLSL creation needs to be fast either way
- On-disk shader cache an answer?
  - Maybe, but ugly

Maybe: CMP in GLSL

- CMP dst, src0, src1, src2
- dst = src0 >= 0 ? src1 : src2;
  - Per component
  - Axel Davy tells me this creates ugly code
- INF / NaN semantics
  - Broken on Nvidia

Multithreading

- Not needed for Wine
  - We'll do it ourselves
  - Needed for correctness constraints
  - Can do d3d-based optimizations
- Native GL games profit
  - E.g. main magic in Half Life 2 on Nvidia is __GL_THREADED_OPTIMIZATIONS

Resolution restore

- Windows has CDS_FULLSCREEN
  - Some external process restores screen on exit
  - Crash or exit without cleanup
  - Probably handled by explorer.exe
- Wine can could handle it in explorer
  - But the problem affects native games too

Tell us when we do something stupid

Despite all these numbers

- „Evergreen“ games perfectly playable on Mesa and Wine
- Casual gamer can run his evening StarCraft 2 session on the open source drivers
  - A lot more important than record framerates in Assassin’s Creed Unity
- Hardcore gamers will probably stay with Windows and / or Nvidia for now

Summary

- Wine and Mesa lack manpower
- Focus on one codepath, not two half-baked ones
- D3D9 in Mesa is neither necessary nor sufficient for good performance

Volunteer Tasks

- Help bisect performance regressions
  - Contact stefandoesinger@gmail.com
  - Expect to spend some time on setup
- Play with Mesa and Wine from git
  - Try to catch problems early

Thank you