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 XIIconifyWindow 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
wined3d: Restore the pixel format of the window whose pixel format was actually changed.
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
<table>
<thead>
<tr>
<th></th>
<th>ddraw</th>
<th>d3d8</th>
<th>d3d9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nvidia GF4</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
</tr>
<tr>
<td>Nvidia GF7</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
</tr>
<tr>
<td>Nvidia GF8+</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
</tr>
<tr>
<td>r200</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
</tr>
<tr>
<td>r500</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
</tr>
<tr>
<td>r600+</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
</tr>
<tr>
<td>i915</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
</tr>
<tr>
<td>i945</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
</tr>
<tr>
<td>i965+</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
</tr>
</tbody>
</table>
## API / GPU / OS

<table>
<thead>
<tr>
<th></th>
<th>ddraw</th>
<th>d3d8</th>
<th>d3d9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nvidia GF4</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
</tr>
<tr>
<td>Nvidia GF7</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
</tr>
<tr>
<td>Nvidia GF8+</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
</tr>
<tr>
<td>r200</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
</tr>
<tr>
<td>r500</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
</tr>
<tr>
<td>r600+</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
</tr>
<tr>
<td>i915</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
</tr>
<tr>
<td>i945</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
</tr>
<tr>
<td>i965+</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
<td>Linux, OSX, (Win)</td>
</tr>
</tbody>
</table>
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

- fix_scale
- Wine
- Nine
- Wine CSMT
- Linux
Example: Half Life 2
Example: Half Life 2
NV Blob: Fast OpenGL is possible
Mesa insists on vsync at full resolution :-(

HL2 GPU Limited

<table>
<thead>
<tr>
<th>Platform</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wine CSMT</td>
<td>180</td>
</tr>
<tr>
<td>Nine</td>
<td>60</td>
</tr>
<tr>
<td>Linux</td>
<td>280</td>
</tr>
<tr>
<td>Windows</td>
<td>300</td>
</tr>
</tbody>
</table>
Civilization V

Not frames per second, some undefined score

Nine renders at lower quality
Lower Draw Overhead

- It is possible with OpenGL
- No need for Nine, Mantle or other wheel reinventions
- Not the holy grail
r600g draws / sec

- fix_scale
- Win d3d
- Win GL
- Mesa GL
Nvidia draws / sec

- fix_scale
- Win d3d
- Win GL
- Linux GL
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
4c4552c5a1910a9d

wined3d: Restore the pixel format of the window whose pixel format was actually changed.
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
- \( \text{dst} = \text{src0} \geq 0 \ ? \ \text{src1} : \ \text{src2}; \)
  - 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