Nouveau: Cooking an Open Source Nvidia driver

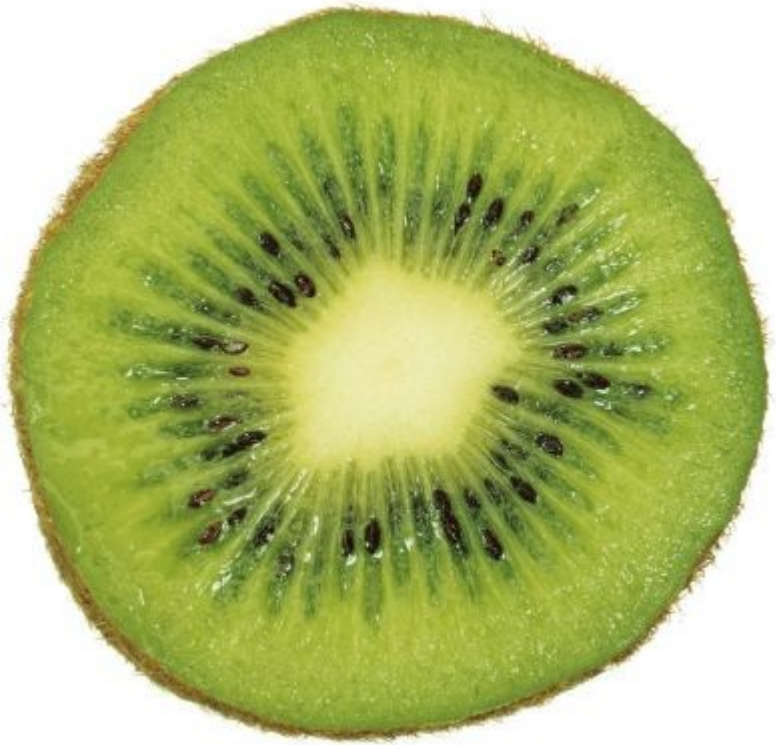
Stéphane Marchesin <marchesin@icps.u-strasbg.fr>

Talk outline

- Introduction
- Nvidia hardware primer
- Reverse engineering
- Current 2D developments
- Current 3D developments
- Future works
- Conclusion/demo

Introduction

- Coding drivers is like eating sane food
- Good cooking keeps you fit
- Good drivers keep you fit, too
  - But only the open source ones



Introduction

- Two years ago
  - Project introduction at Fosdem
  - Initial reverse engineering
- Last year
  - Some EXA
  - Most specs reverse engineered
  - glxgears
- This year ?

Introduction

- Community
  - Bi-weekly news updates (TiNDC)
  - Associate with news sites
  - Gather more developers
- Currently
  - Half a dozen developers
    - All spare-time
  - Heavy use of IRC
  - Unwritten policies
    - Broken code is better than no code

Nvidia hardware primer

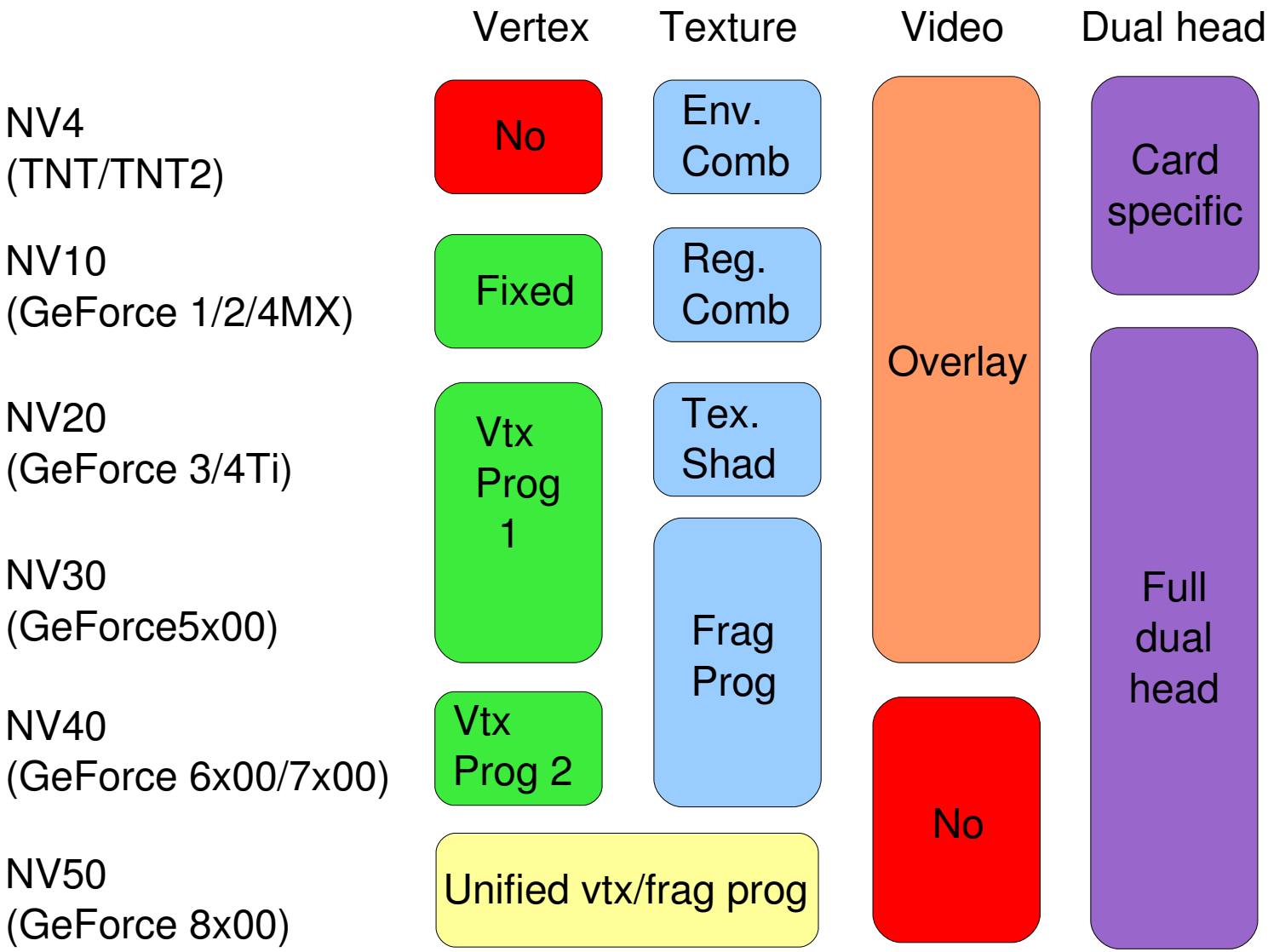
Nvidia hardware primer

- 10-year timeframe:
  - NV04 (1998)
  - NV50 (2008)
- Similarities between the cards
  - Fifo handling
  - Some registers
- Differences
  - 3D handling

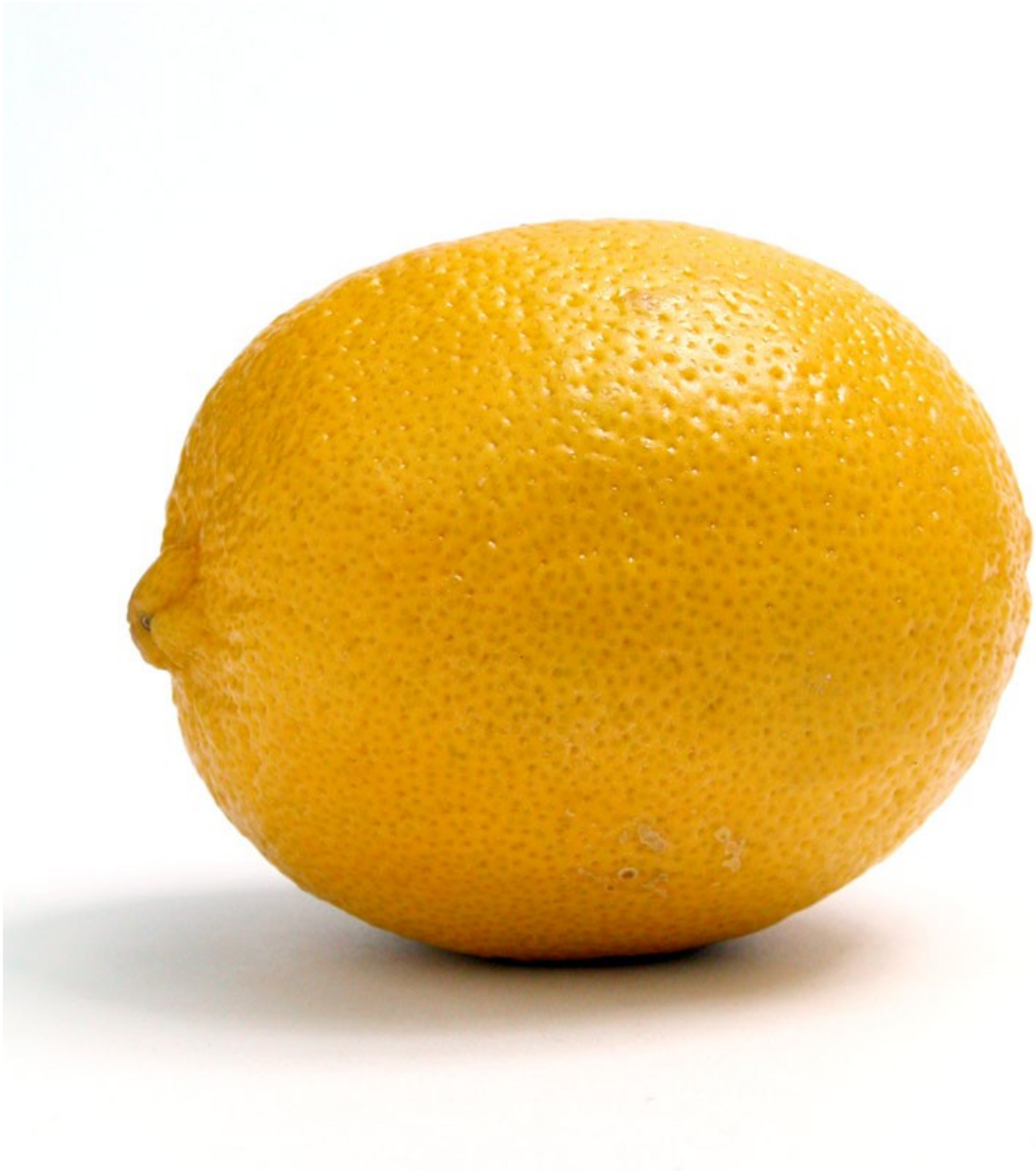
Nvidia hardware primer

- Multiple hardware contexts
  - A hardware context = a fifo + a graphics state
- Full hardware context switching
  - Each context gets a time slice
  - Once the time slice is up, the hardware switches the context
    - Save/Restore the graphics state
- Hardware speaks OpenGL almost natively
  - Hardware uses OpenGL values (e.g. for blending)
  - Few fallbacks when implementing OpenGL

Nvidia hardware primer



Reverse engineering



Reverse engineering

- Mmio-trace
  - In-kernel tracing
    - Trap ioremap calls, remove the mapping
    - Next access to the mapping triggers a page fault
    - Map the page
    - Step the instruction
    - Remove the page again
  - Used by other projects
  - Broken in recent kernels
  - Aiming for mainline kernel inclusion (2.6.26 ?)

Reverse engineering

- Valgrind-mmt
  - Similar to mmio trace
  - But in user space
  - Usable on X.Org

Reverse engineering

- Renouveau
  - Old dog (released 2 years ago at Fosdem)
    - User space OpenGL application
    - Nvidia maps the fifo to user space
    - Find the fifo mapping
    - Run a simple OpenGL code
    - Look for changes in the fifo, print them
  - New tricks (regularly updated OpenGL tests)

2D support

EXA

- Simple EXA
  - NV4 and NV50
    - NV4 can't do more
    - NV50 is not widely spread among devs yet
- Full XRENDER
  - NV10/NV20/NV30/NV40
  - Accelerates all the operations that common desktop environments and toolkits need
- Leftover issues
  - Some are in core EXA
  - Performance glitch from synchronous uploads

Xv

- High performance
  - Better than the binary driver
  - Overlaps data upload and rendering
- High quality
  - Bicubic filtering
  - Implements Matthias Hopf's last year Fosdem talk

Bios Parsing

- All nvidia cards use a mini language for the bios
  - Bytecode
  - Similar to assembly
- The interpreter was written early on
  - Adated from a cold booting application and from the BeOS/Haiku driver
- Slept for a long time, but was recently picked up
- Required for cold booting
- Required for modesetting/Randr 1.2

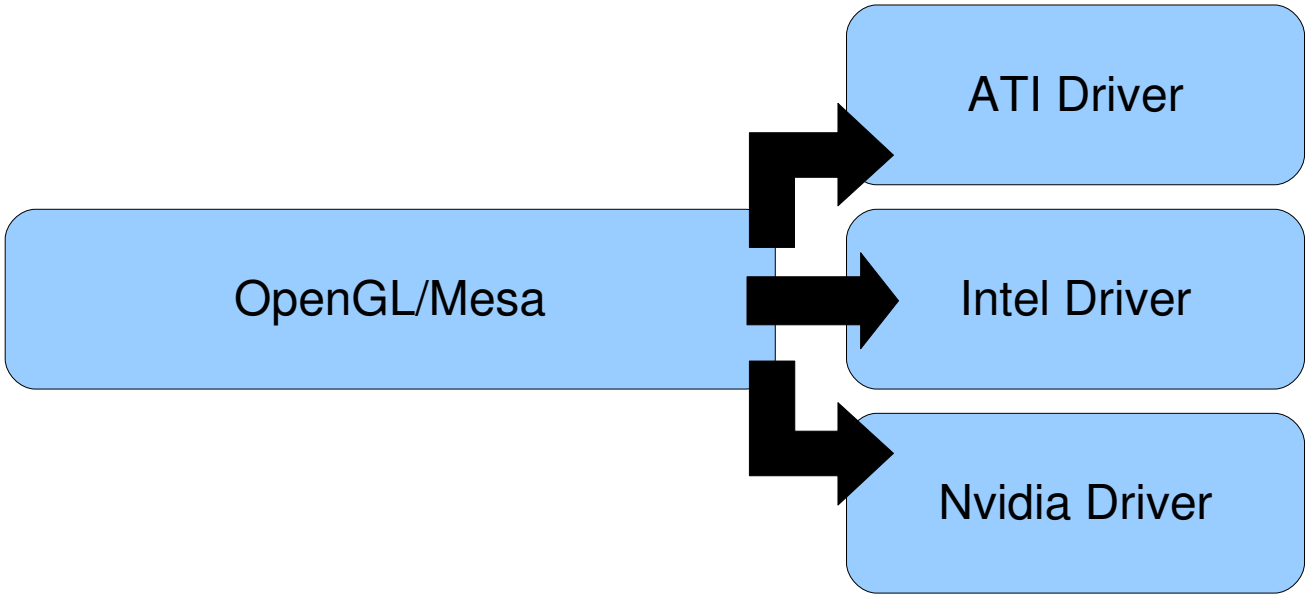
Randr 1.2

- Dual head support
  - NV17 -> NV50 should work
  - Older dual head is more difficult
- Dynamic reconfiguration
- Static front buffer allocation issue

3D support

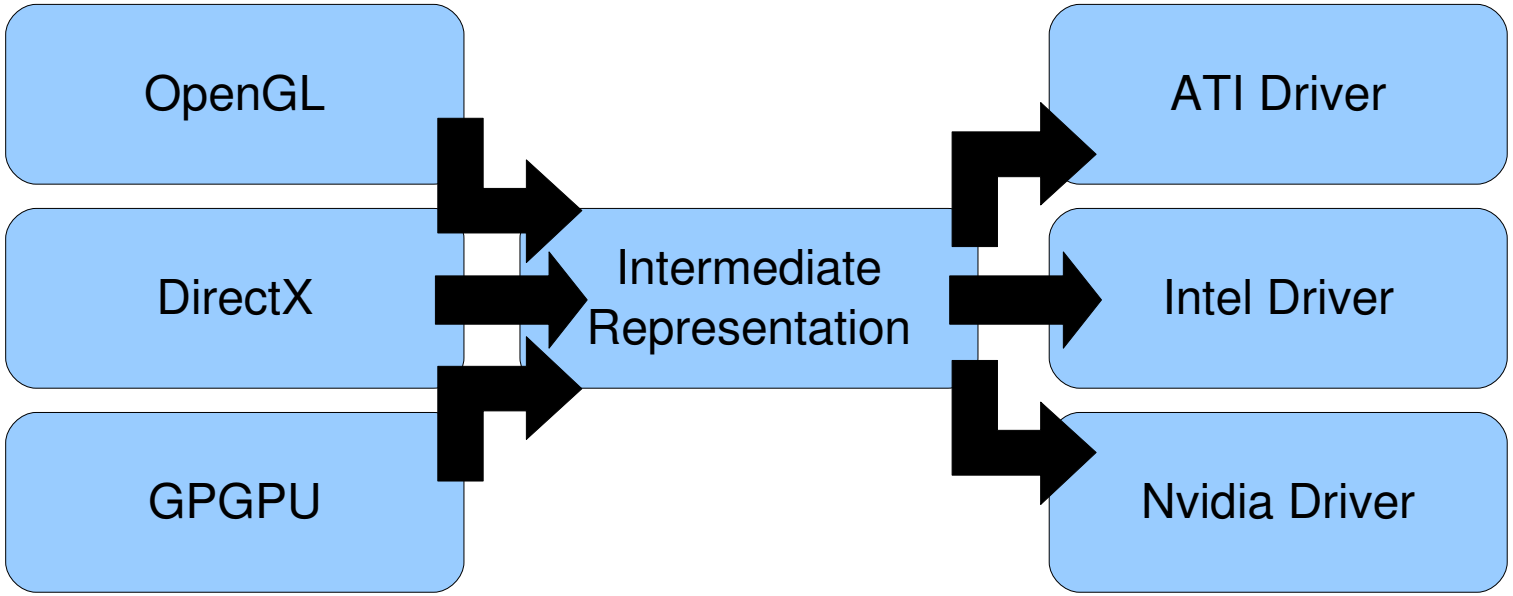
Gallium 3D

- Old DRI model:
  - Most OpenGL functions must be implemented in the driver
  - Lots of work in the driver
  - Each driver implements most of the OpenGL pipeline



Gallium 3D

- Gallium 3D model:
  - Front ends implement different APIs (OpenGL, DirectX, OpenGL 3.0, GPGPU, OpenVG, Coffee maker) by converting all fixed pipe into shaders
  - Intermediate layer optimizing the shaders
  - Back ends turn the intermediate language into card-specific code



Gallium 3D: what about older cards ?

- Older cards do not have shaders
- The previous model does not work
- Unless...

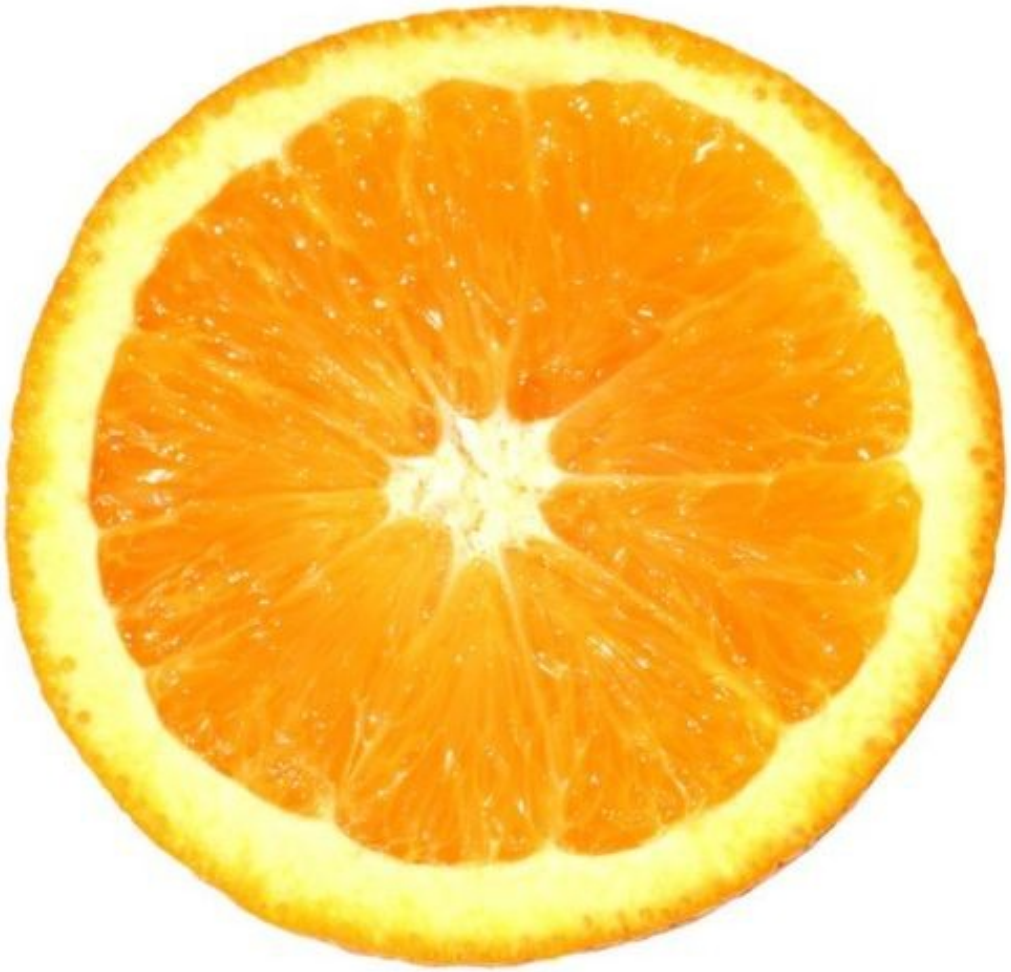
Gallium 3D: what about older cards ?

- Additional intermediate language instructions
- These new instructions represent the old multitexture/texture combiners/texture extensions of NV4/NV10/NV20
- Old cards can be supported with the same architecture

Gallium 3D

- Current developments
  - NV40 driver somehow working
  - NV10, NV30 under development

Future cooking ?



TTM

- More than just memory management
- The kernel is in charge of placing/moving pieces of memory
  - Similar to kernel memory allocation and swapping
  - User space code does not need to track memory addresses itself
  - Kernel “relocates” command buffers
- "Superioctl"
  - (not that super in our case)

DRI2

- Still in the air
- Try to remove the big lock (we don't have one)
- Remove the SAREA (we don't have one)
- Remove shared back buffers (we don't have those)

Kernel modesetting

- Wait for Randr 1.2 stabilization
  - Developing Randr 1.2 in user space is easier
- Move Randr 1.2 into the kernel
- Requires moving the bios parser into the kernel as well
  - «Hi Linus, can we put that bytecode interpreter into the kernel ?»

Suspend/Resume

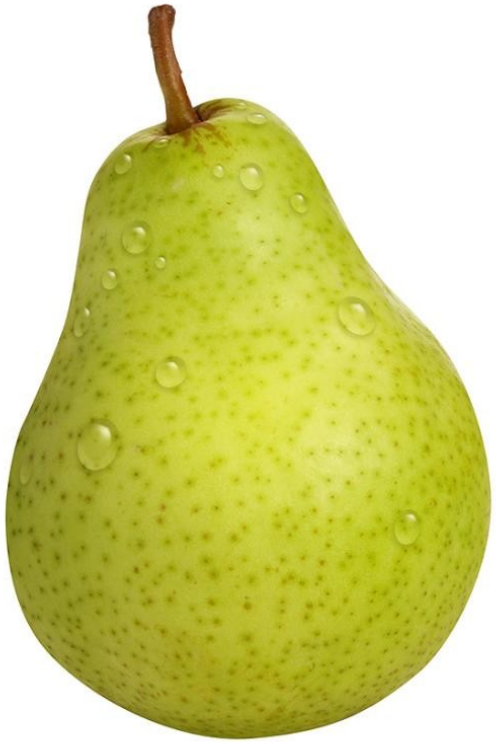
- Work in progress
- Requires a fully working bios parser
- Requires saving & restoring the card state
- Better with kernel modesetting in place
- Not as obvious as it seems

Releases ?

- We get asked a lot about releases
- Stabilize DRM interface
  - Not yet the case
  - Lots of changes left before this happens
  - Stabilizing too early = burden of maintenance later

Conclusions

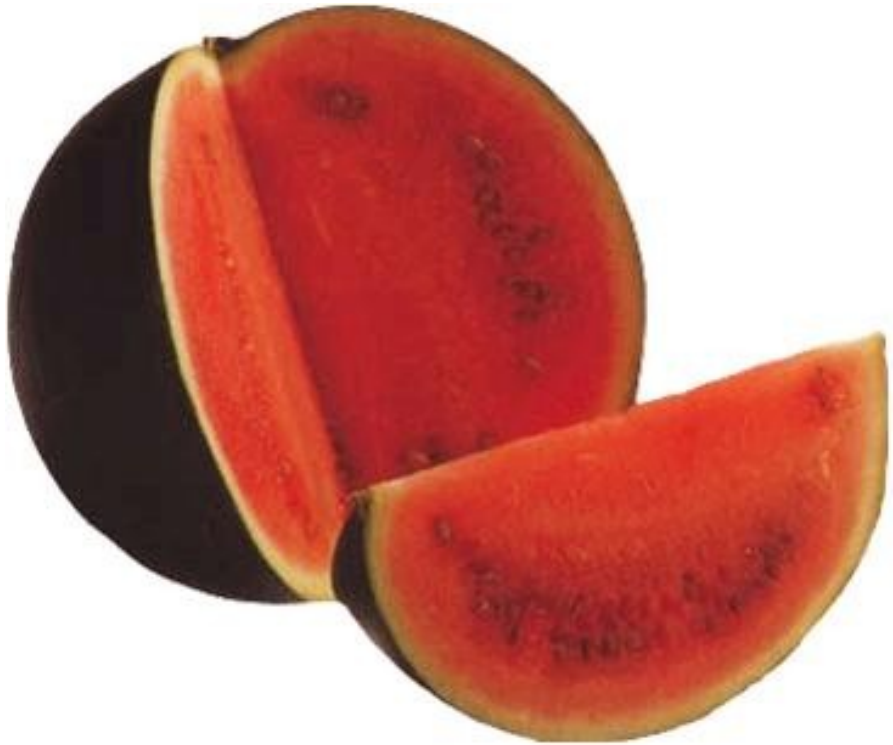
- Mostly finished 2D support
- 3D is largely underway
- NV50 is still the bad boy
- Eating fruits is good



Conclusions

http://nouveau.freedesktop.org

irc: #nouveau on freenode



Demo