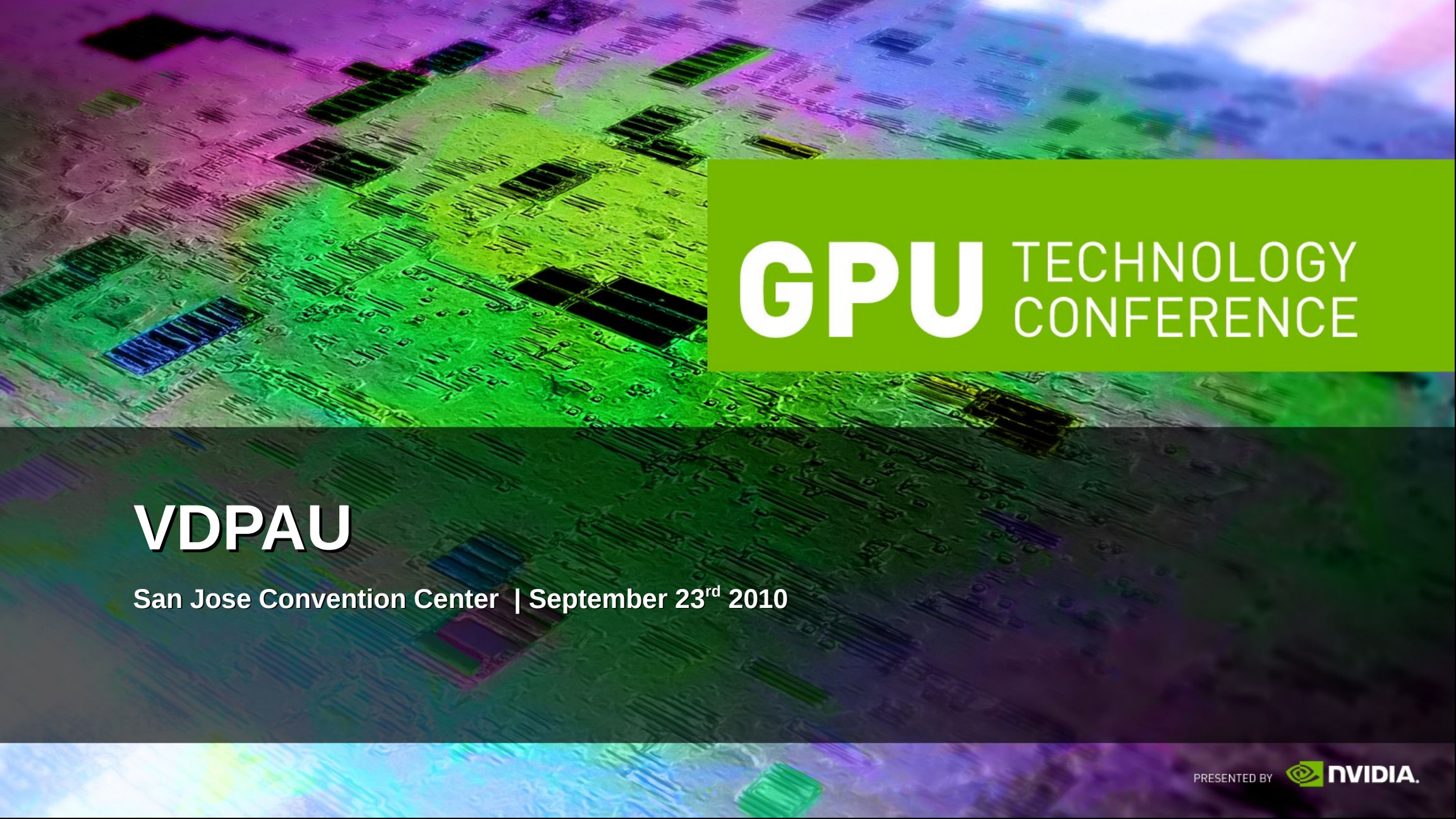


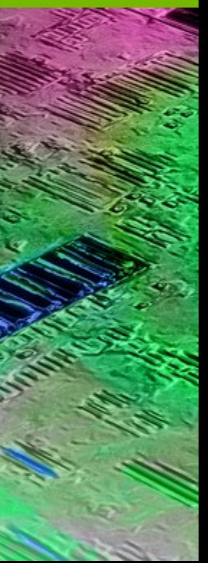
VDPAU

San Jose Convention Center | September 23rd 2010

PRESENTED BY NVIDIA.

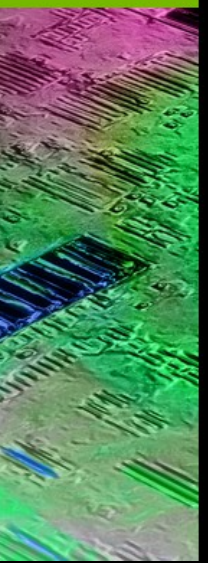
Outline

- What is VDPAU?
- Status
- Data Flow
- Features
- Interoperability
- Demos



What is VDPAU?

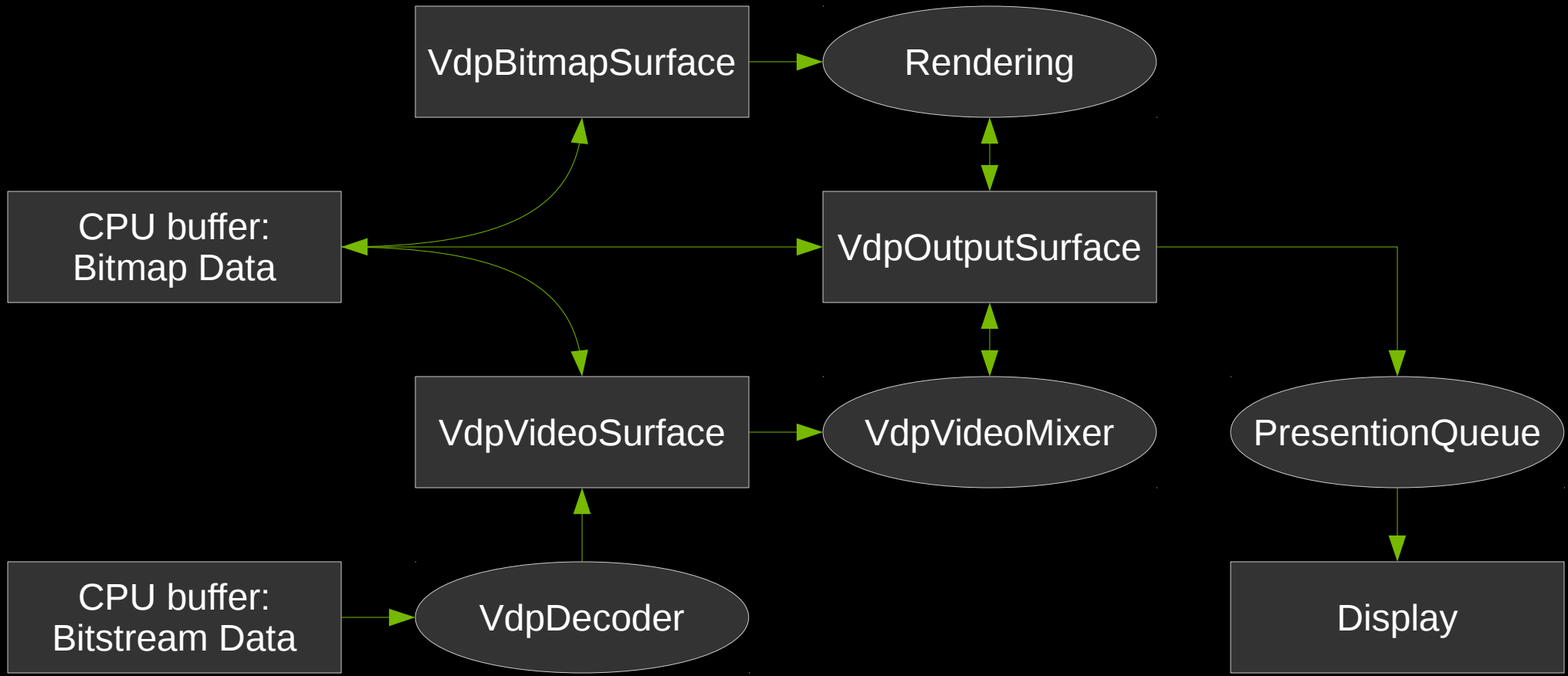
- Unix API for
  - GPU-accelerated video decoding
  - Post-processing
  - Simple UI compositing
  - Display
  - Customization via interop



Status

- Shipping since late 2008
- Advanced interop APIs shipping since early 2010
- Supported on GeForce 8 series and higher
  - With just a few exceptions
- Runs on Linux, Solaris, and FreeBSD
- Supported by many media player applications
- Well supported by NVIDIA

Objects and Data Flow



Features – Decoding

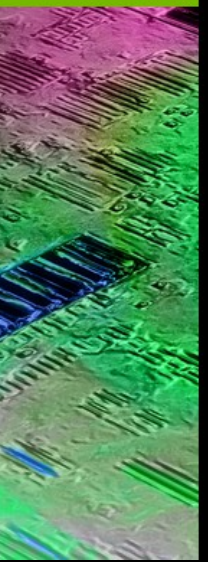
- All formats supported by the GPU
  - MPEG-1/2
  - H.264 / AVC / MPEG-4 Part 10
  - VC-1 / WMV3
  - MPEG-4 Part 2 / DivX
- VLD level API only
  - Applications work the same way on all HW

Features – Post-Processing

- De-interlacing
  - Weave, Bob, and two more advanced algorithms
  - Inverse-telecine
- Color-space conversion
- “Procamp” (brightness, contrast, saturation, hue)
- Noise reduction
- Sharpness

Compositing

- During post-processing of video ...
- Scale video
- Extract a sub-rectangle
- Alpha-blending with other *n* surfaces
  - Each scaled and with a sub-rectangle extracted
- Result can be looped back to blend *n* video streams

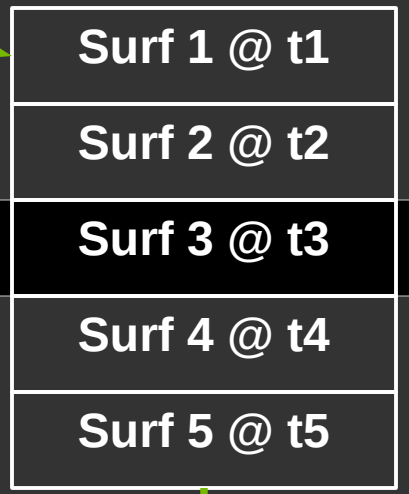


Presentation Queue

CPU

```
for ever:
    parse bitstream
    vdp_decoder_render(bitstream, vid_surf)
    vdp_mixer_render(vid_surf, out_surf)
    vdp_pq_display(out_surf, timestamp)
```

Push asynchronously



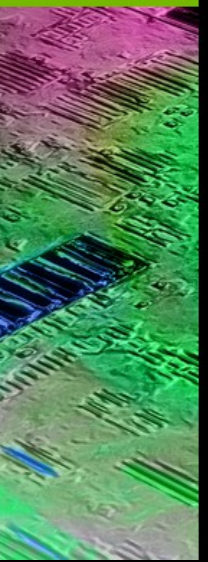
Display

Pop at first VSYNC after t

GPU

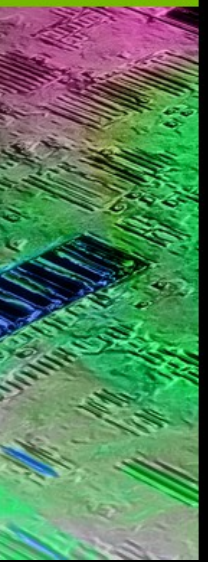
Interop – CPU

- “Put Bits”: Upload surface content from CPU to GPU
  - VdpVideoSurface, VdpOutputSurface, VdpBitmapSurface
- “Get Bits”: Download surface content from GPU to CPU
  - VdpVideoSurface, VdpOutputSurface
- Limited format conversions available in some cases
- CPU memory can be used by any other API
  - File I/O, X11 rendering, OpenGL texture upload, ...
- Slowest path if extracted data to be processed on GPU



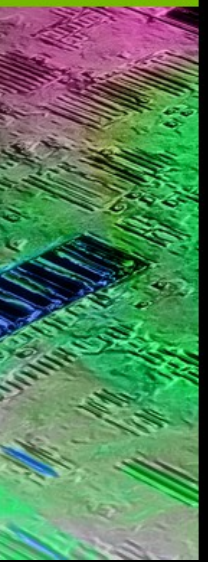
Interop – X Pixmaps

- Presentation Queue can render to
  - X Window, for end-user display
  - X Pixmap, for interop
    - GLX_EXT_texture_from_pixmap
- Only a single X pixmap per presentation queue
  - Limits pipelining/concurrency



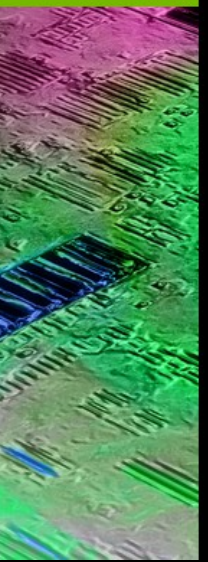
Interop – VdpOutputSurface

- Native format is a single w×h ARGB surface



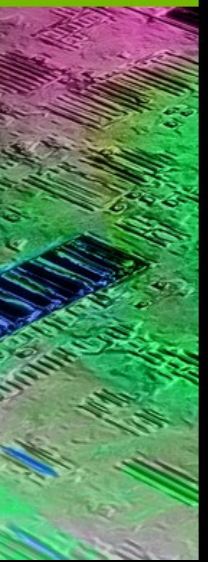
Interop – VdpVideoSurface

- Each field exposed as separate sub-surfaces
- Luma and chroma exposed as separate sub-surfaces
- ⇒Two w×(h/2) surfaces for luma
- With appropriate chroma sub-sampling
- ⇒ Two w×(h/2) or w×(h/4) surfaces for chroma



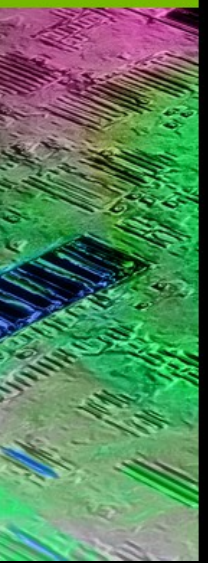
Interop – CUDA

- Exposes VDPAU surfaces as CUDA arrays
- VDPAU surfaces readable and writable
- Well-defined ordering between VDPAU and CUDA accesses
- Access *n* surfaces at a time
- Fits into standard CUDA interop API style in CUDA 3.1
- Both driver-level and toolkit-level APIs



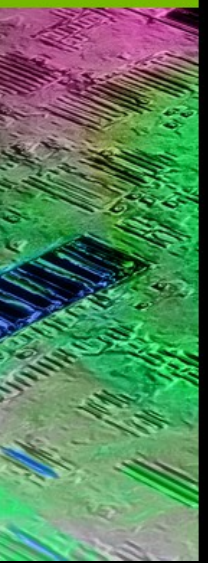
Interop – CUDA

- cuVDPAUCtxCreate
- cuGraphicsVDPAURegister{Video,Output}Surface
- cuGraphicsMapResources
- cuGraphicsSubResourceGetMappedArray
- Read from CUDA array, or memcpy to it
- cuGraphicsUnmapResources
- cuGraphicsUnregisterResource



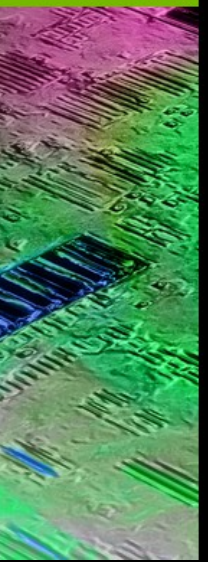
Interop – OpenGL

- Gives VDPAU surfaces OpenGL texture names
- VDPAU surfaces readable and writable
- Well-defined ordering between VDPAU and GL rendering
- Access *n* surfaces at a time
- GL_NV_vdpau_interop



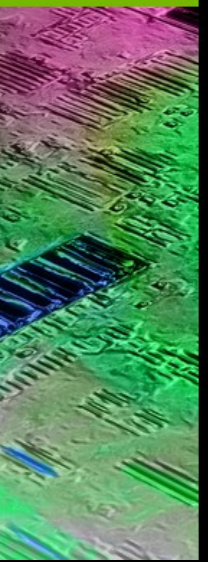
Interop - OpenGL

- VDPAUInitNV
- VDPAURegister{Video,Output}SurfaceNV
- VDPAUMapSurfacesNV
- Texture from the surface, or render to it
- VDPAUUnmapSurfacesNV
- VDPAUUnregisterSurfaceNV
- VDPAUFiniNV

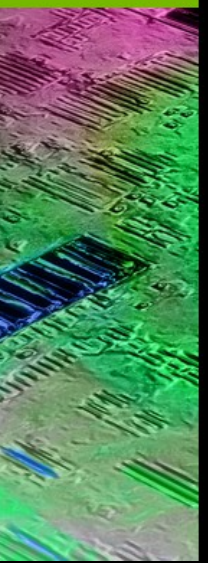


Demos

- Basic Decoding
- Interop with CUDA
- Interop with OpenGL



Questions?



Questions?