Project SHIELD and Tegra 4: Redefining AFK

Andrew Edelsten (Manager, Tegra Developer Technologies)
Paul “Hodge” Hodgson (Manager, Tegra Developer Technologies)

NVIDIA

Overview

Andrew

- Tegra 4 & Project SHIELD
- Game considerations for Project SHIELD
- NVIDIA development tools for Android

Hodge

- Tegra 4’s new GPU features
- Anatomy of Tegra 4’s GPU

Tegra 4

"*NVIDIA Tegra 4 is a promising processor that's going to bring a whole new level of gaming to mobile devices.*"

SLASHGEAR

CNN

Hottest gadgets MWC 2013

Better photography:
NVIDIA Tegra 4 HDR camera

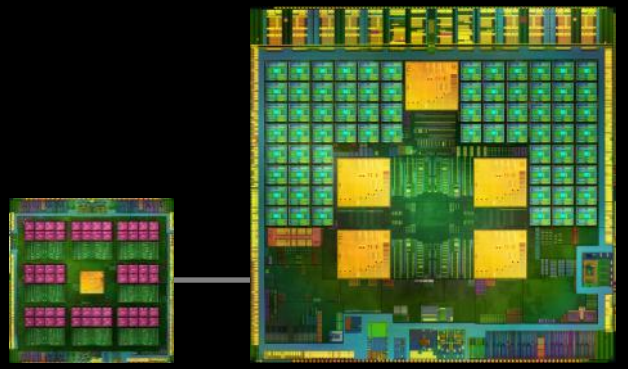
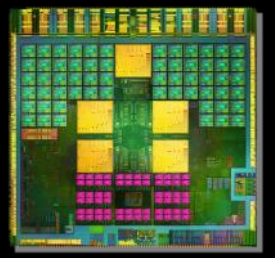
"*if you enjoy the web browsing experience on your iPad, you're going to be pretty pleased what NVIDIA has to offer here.*"

engadget

"*If you want to take better pictures on your mobile device, NVIDIA's Chimera computational photography engine is the technology you've been waiting for.*"

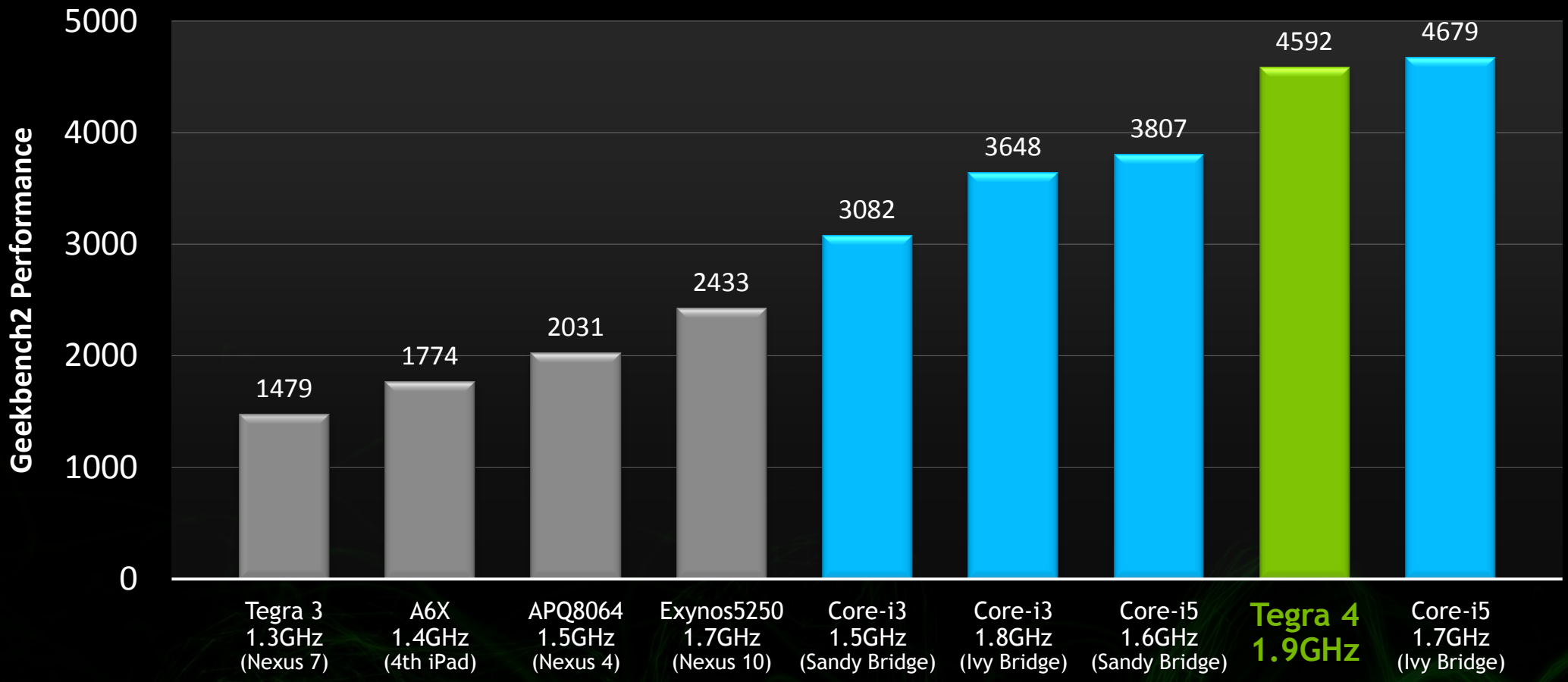
LAPTOP
THE PULSE OF MOBILE TECH

Tegra 4 Family

	Tegra 4 (“Wayne”) *World’s Fastest Mobile Processor* Superphone / Tablet	Tegra 4i (“Grey”) *1st Integrated Tegra 4 LTE Processor* Smartphone
		
Quad CPU	Cortex A15, 4+1	Cortex A9 r4, 4+1
NVIDIA GPU	72 Core	60 Core
LTE	Optional with i500	Integrated i500
Chimera*	✓	✓

* Chimera is NVIDIA’s Computational Photography

Mobile Processor, Ultrabook Performance



Intel Core i3-2377m 1.5GHz, Core i3-3217U 1.8GHz & Core i5-2467m 1.6GHz, Core i5-3317U 1.7GHz all have 17W maximum TDP
Competitive data published on Geekbench website; Tegra 4 1.9GHz measured on reference platform

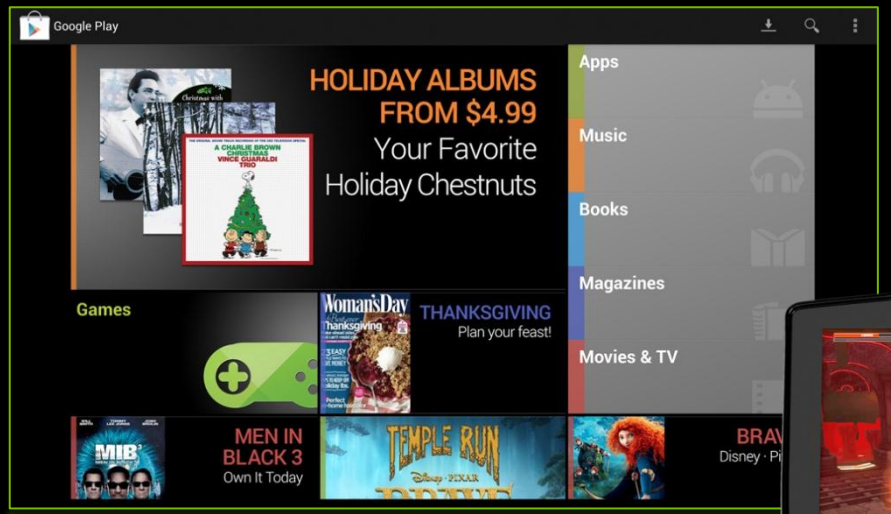

NVIDIA
Project SHIELD

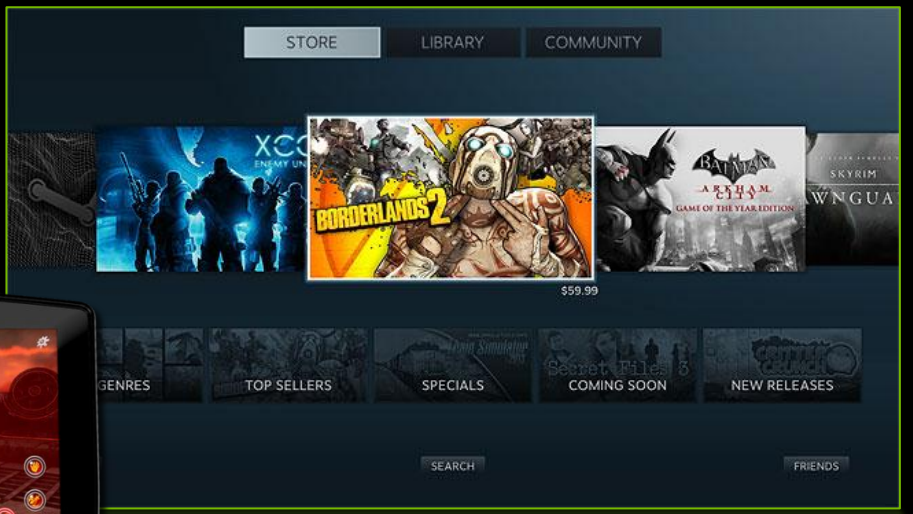
Project SHIELD

- Tegra 4 powered
- 5 inch 720p & multitouch display
- Console grade controller
- High speed Wi-Fi
- Full connectivity (HDMI, USB, microSD, headphone)
- Pure Android (currently Jellybean)


Tuned Port, Bass Reflex Speakers

Two Open Platforms — One Amazing Portable





Android

PC

SHIELD Development Considerations

- Support landscape screen orientation
  - Don’t assume device is a phone and lock to portrait based on DPI
- Don’t *require* touch (for Android games) or a mouse (for PC games)
- Test using HDMI
  - Is everything possible without getting up?
  - How does it look on a big screen?
- Optimize your PC game for Streaming (see next NVIDIA session in this room)
- Controller is King!

Controller is King

- Auto-detect the controller
- Include a controller map overlay
- Sub 20fps extremely noticeable
- Remove all on-screen touch elements
- Use Android Input and code to the Built for Tegra standard
- UI should:
  - Have visual focus indicator (highlight, arrows, etc)
  - Use classic standards for navigation (9 & 6 for OK etc)
  - Allow use of all elements (menu items, checkboxes, sliders etc)
  - Include “exit” in the main and pause menus

Controllers Everywhere

See the Tegra developer documentation “How To: Support Android Game Controllers” and NativeGamepad sample for a great guide on how to handle multiple controllers on all Android devices

Developing for Android

- Setting up an Android development environment can be tricky
- Android SDK, NDK, ANT, Eclipse, adb.. Grrr!
- Native debugging.. Double grrr!
- Is that gcc configuration quite right?

Tegra Android Development Pack

- **GET STARTED** in minutes NOT hours
- **INSTALLS** all tools required for Tegra Android

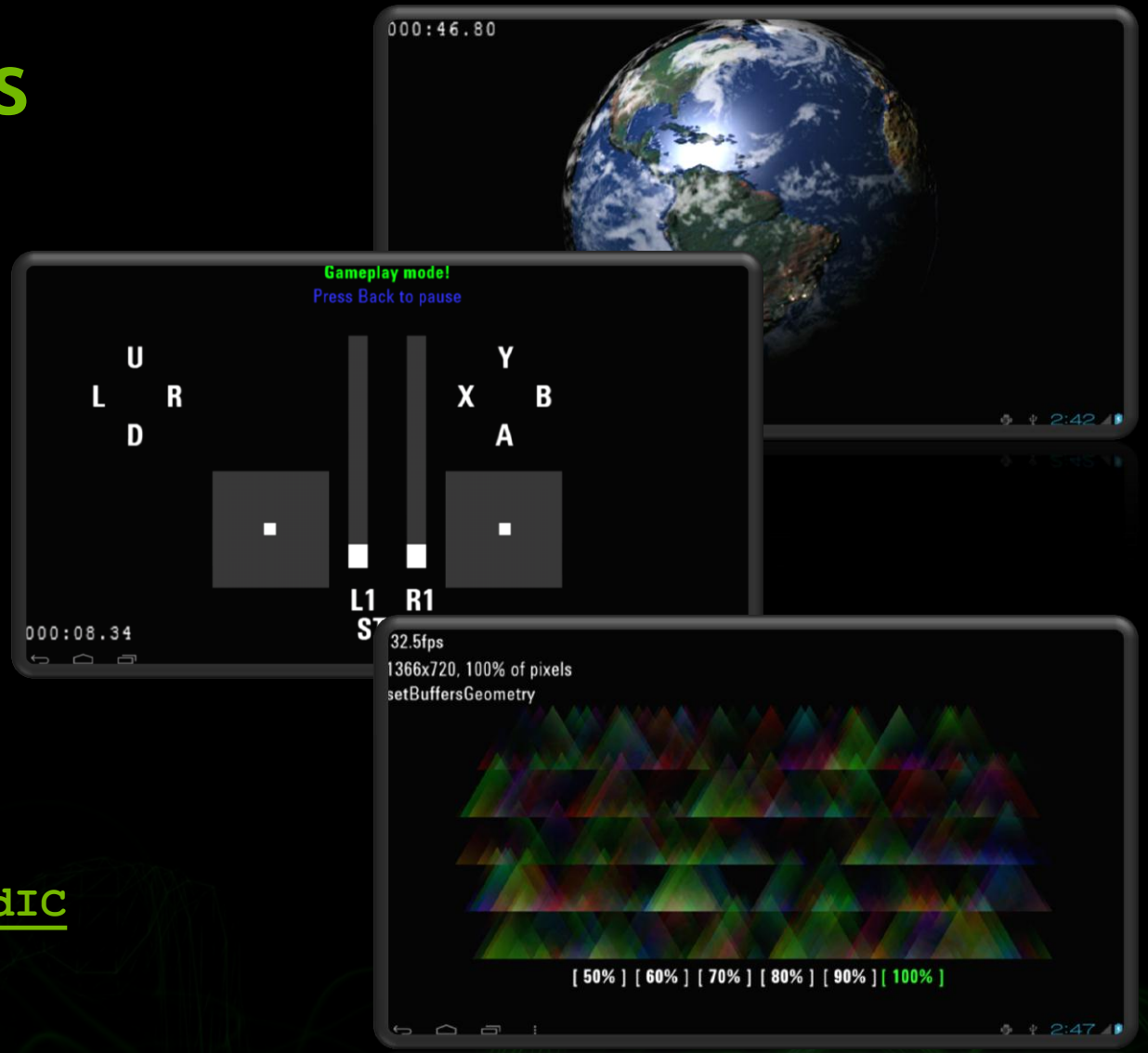
- **CPU DEBUGGING** with Nsight Tegra
- **GPU DEBUGGING** with PerfHUD ES
- **OPTIMIZE** applications with Tegra Profiler
- **REFERENCE** docs, samples & tutorials

- **OPTIMIZED** for Tegra Android development
- **FLASHES** Tegra DevKit with OS Image
- **CONFIGURED** for debugging and profiling
- **INCLUDES** Kernel symbols and DS-5 support

http://developer.nvidia.com/develop4tegra

Native Code Samples

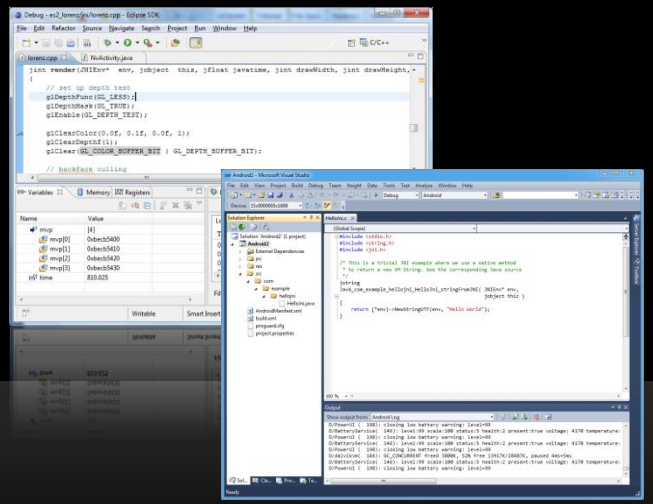
- Android lifecycle
  - Lifecycle can be tricky
  - Highly recommend using "Native Basic" as a base
- OpenGL ES
- Input device handling
  - Multitouch
    - Beware the stylus!
    - Use getToolType()
      -- see http://goo.gl/eRdIC
  - Sensors
  - Gamepad



http://developer.nvidia.com/develop4tegra

Tegra Developer Tools

Native Android Development Tools



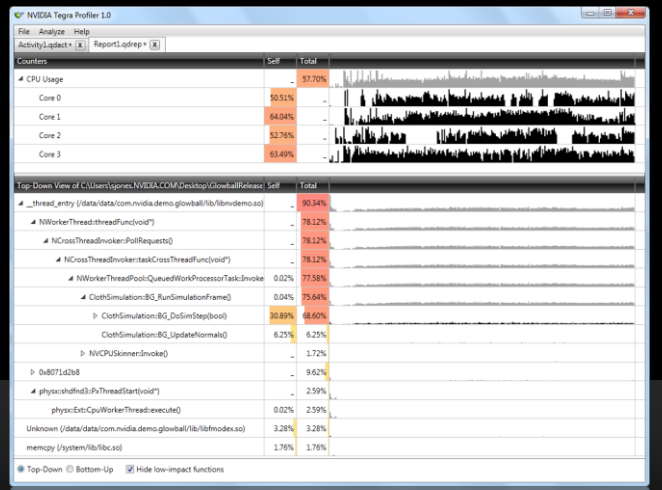
Nsight Tegra

Visual Studio and Eclipse integrations

Full Android build management

Native Android CPU debugging

Breakpoints in both Java and Native

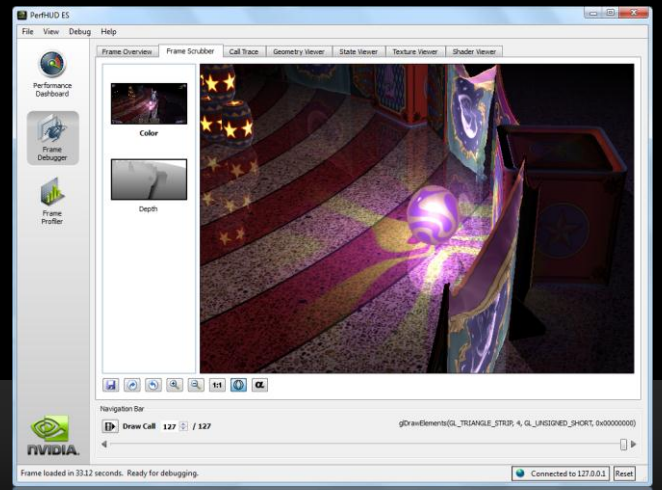


Tegra Profiler

Maximize multi-core CPU utilization

Quickly identify CPU “hot spots”

Identify thread contention issues



PerfHUD ES

Examine and debug OpenGL ES frames

Automated bottleneck analysis

Edit shaders at runtime

http://developer.nvidia.com/develop4tegra

The Tegra 4 GPU

Paul “Hodge” Hodgson
Tegra Developer Technologies

Depth on Tegra 4

- Many additional extensions supported
  - OES_depth24
  - OES_depth_texture
  - OES_depth_texture_cube_map
  - EXT_shadow_samplers
  - NV_shadow_samplers_cube
  - NV_shadow_samplers_array
- Hardware PCF

Tegra 4 - Shadows

Shadows

- Depth textures
- Percentage-closer filtering
- Soft-edged, AA shadows
- No fragment shader emulation

Tegra 4 - Shadows

HDR on mobile

- Improved half float support
  - OES_texture_half_float_linear
  - OES_texture_half_float
  - EXT_color_buffer_half_float
- Introducing sRGB
  - EXT_sRGB
  - NV_sRGB_formats
  - NV_generate_mipmap_sRGB

Tegra 4 - HDR

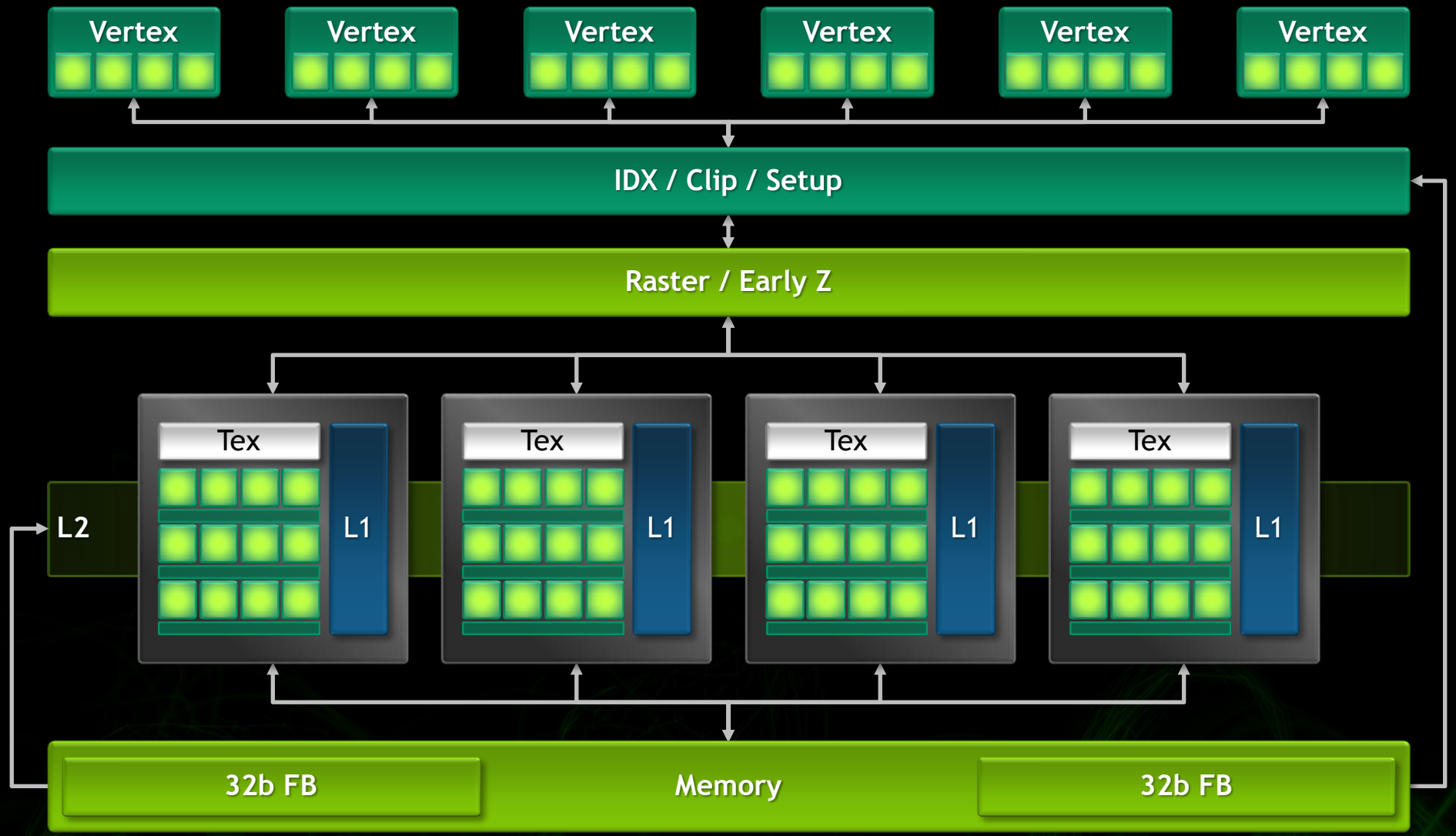
HDR

- FP16 Filter
- FP16 Blend
- Multiple Render Targets
- sRGB

Tegra 4 GPU Features

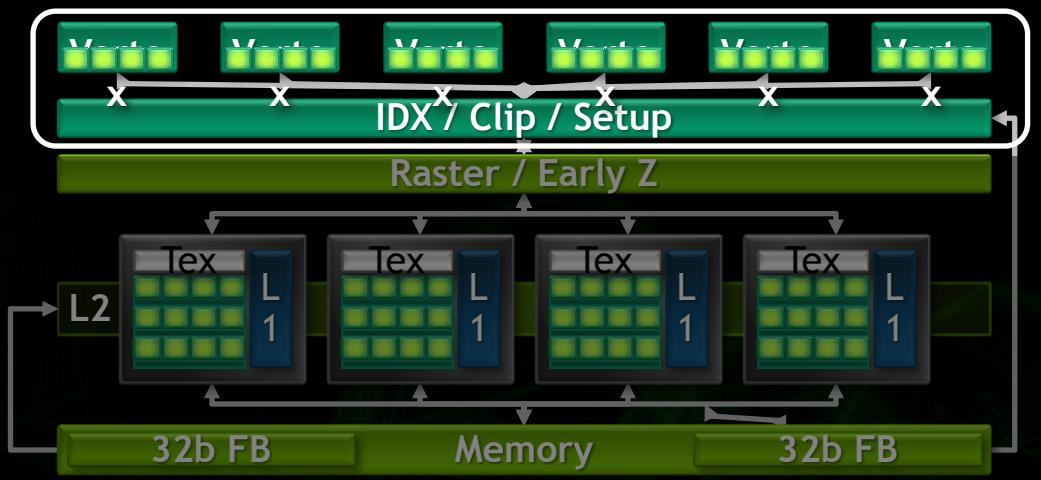
ES Features	Tegra 4
FBO_render_mipmap	✓
Uniform Buffer Objects	✓
Separate Shader Objects	✓
Framebuffer Blit	✓
Copy Buffer (ARB_copy_buffer)	✓
Explicit Attribute Locations	✓
Surface-less context creation	✓
Texture Storage	✓
Pixel Buffer Objects	✓

ES Features	Tegra 4
24-bit Depth	✓
FP16 Texture Filtering	✓
Multisampling	✓
Occlusion Queries	✓
Non-square Matrices	✓
Multiple Render Targets	✓
R8, RG8, RGB8, RGBA8, RGB565	✓
SRGB8_ALPHA8, RGBA4, RGB5_A1	✓
{R, RG, RGBA}{8}{I,UI}	✓


Vertex
Vertex
Vertex
Vertex
Vertex
Vertex
IDX / Clip / Setup
Raster / Early Z
Tex
L1
Tex
L1
Tex
L1
Tex
L1
L2
32b FB
Memory
32b FB

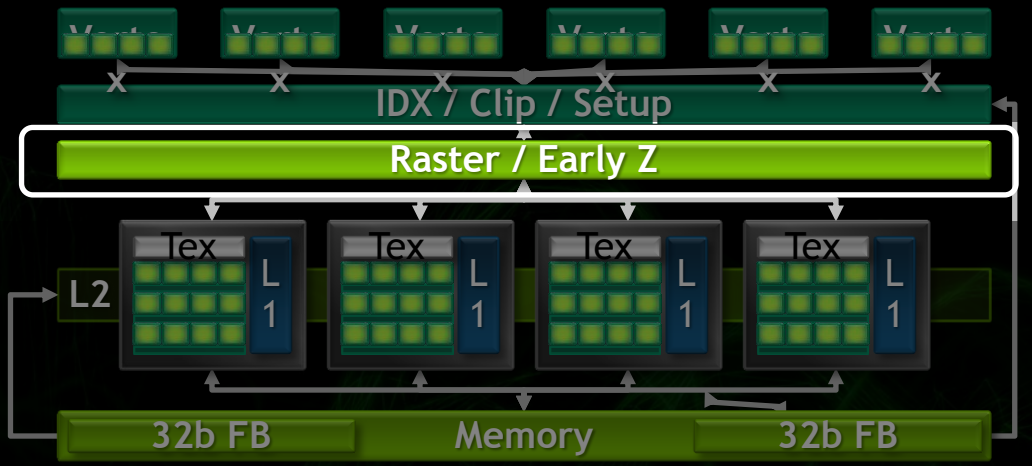
Vertex and primitive processing

- ~60 cycles in vertex shader per visible primitive
  - 60 DP4
  - 15 vec4*mat4
- Parallel to fragment shader
- Vertex cache
- Use optimized triangle lists



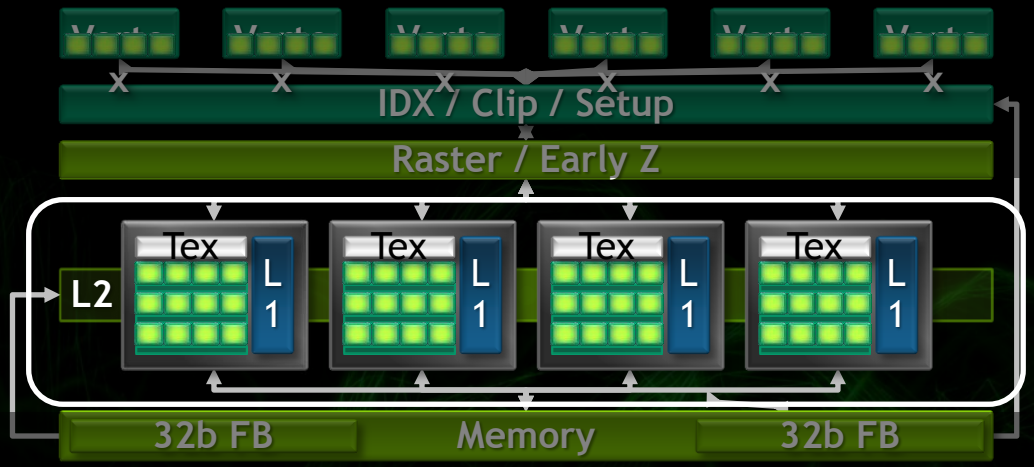
Raster and early z

- 8 fragments per clock
- Can skip fragment shader if depth/stencil killed
- Can skip fragment shader if no surface writes
- Depth compression
- Depth pre-pass, consider it now



Fragment shader

- 4 “mad” units with a single VLIW instruction, can for example
  - 4x MAD
  - 2x DP2A + MFU
  - 1x DP3A + 2x MFU
  - 1x DP4 + MFU
- Use ‘lowp’ precision specifier where possible
  - 2x ‘lowp’ DP4 + MFU
- Single clock lerps per mad unit
- Input & output modifiers
- Free precision conversion
- Ideal ALU/TEX ratio of 3



Tegra 4 GPU - Improved Effective Scaling

- Consumes half the vertex attribute bandwidth of Tegra 3
- Texture L2 cache added
  - reduces over-fetch across pipes
  - increases total effective cache size
- Improvements to pixel pipe to increase ALU utilization

Improvement	Effect
ALU local register state	Reduces power and perf cost of allocating registers
Increase max pixel shader registers over Tegra 3	Up to 24 vs. Tegra 3's 16 fp20 registers per pixel (more threads in flight)
Increase instruction tables for ALU	Improves the efficiency of long programs
Add Multi-Function Unit (MFU) to ALU	Better MFU scaling, improve ALU utilization

Tegra 4 vs Tegra 3 GPU stats

	Tegra 4/ Tegra 3
Vertex Shader	8x
Fragment ALU	8x
Pixel Rate	2.6x
Texture Rate	2.6x
Memory Rate	2.3x
Z-Kill Rate	1.3x
Triangle Rate	1.3x

Tegra 4 - 72 Core GPU @ 672 MHz
4 pixel pipes * 3 ALUs/pipe * 4 MADS/ALU +
6 VPEs * 4 MADS/VPE

Tegra 3 - 12 Core GPU @ 520 MHz
2 pixel pipes * 1 ALU/pipe * 4 MADS/ALU +
1 VPE * 4 MADS/VPE

Wrapping Up

- Questions/Comments?

- Resources
  - NVIDIA Developer Zone - https://developer.nvidia.com
  - NVIDIA Developer Forums - https://devtalk.nvidia.com
- Presentation References
  - “Moving Games into the Cloud, Technologies and Architectures”
    - NEXT UP IN THIS ROOM
  - “Eliminating Texture Waste: Borderless Realtime Ptex”
    - Friday, March 29, 10:35 am – 11:00 am in Room 307, South Hall
- NVIDIA’s GDC Exhibit Booth: #1602