HPG2012 ~~Hot3D~~ Cool3D

Design Tradeoffs in the Kepler Architecture

June 26, 2012

Steve Molnar

NVIDIA

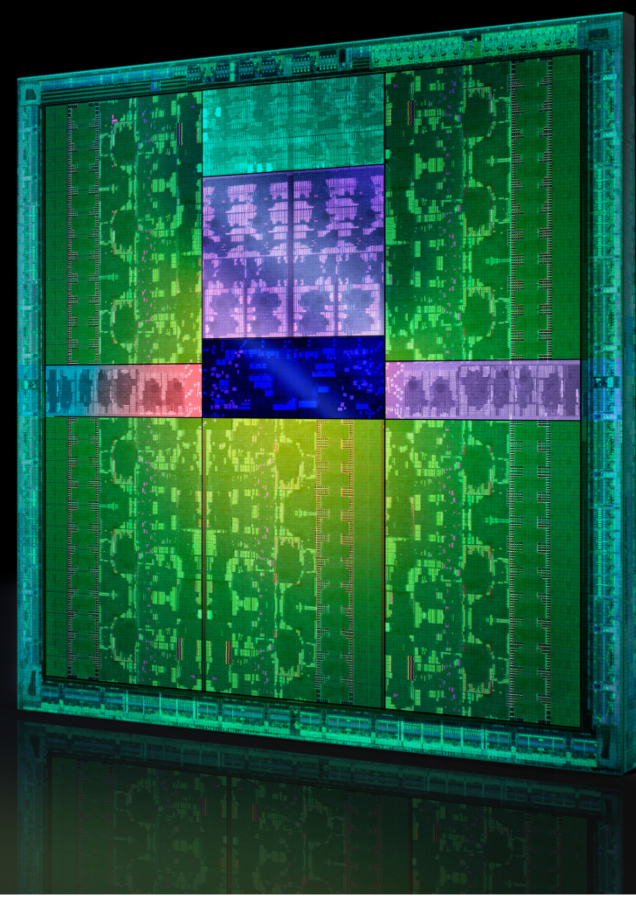
Kepler GK110 Block Diagram

Architecture

- 7.1B Transistors
- 15 SMX units
- > 1 TFLOP FP64
- 1.5 MB L2 Cache
- 384-bit GDDR5
- PCI Express Gen3

The Kepler GPU architecture was designed for

- Performance
- Programmability
- **Efficiency**



The Kepler architecture family had many goals, but the key goal was *efficiency* (perf/watt)

- On Fermi we designed for max performance, but found ourselves power-limited in many cases:
  - Tesla high-performance compute parts were power limited and had to run at lower voltage and clocks than the design allowed
  - Dual-GPU systems had to run at lower clocks
  - Mobile parts required lower-than-desirable voltage and clocks to maintain battery life

On Kepler power was critical

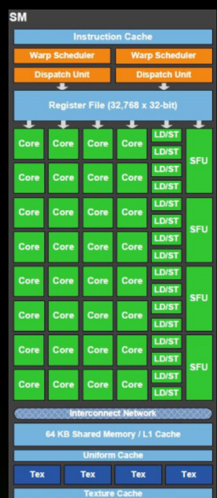
- Below ~0.1µm, CMOS power no longer scales with feature size
  - Moving to a smaller process, a chip of given size burns more power than the previous one
  - You can lower voltage (and clock speed), but that's no solution
- In Kepler's 28nm process, we could be power limited by 50%

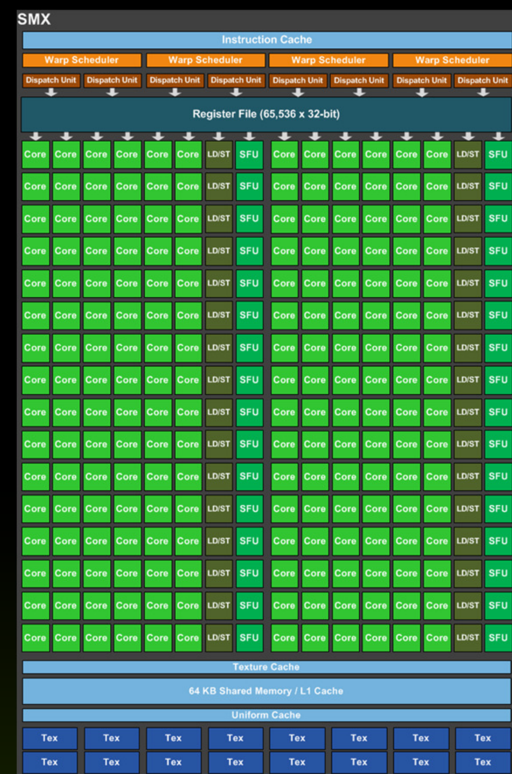
- *Time for decisive action!*

Kepler took holistic view of power

- We had previously designed to reduce power, but on Kepler, we attacked power holistically
  - Tools to measure power consumed by each unit
  - Aggressive clock and power-gating
  - Redesigned shader core to greatly increase efficiency
  - Redesign of GDDR5 DRAM I/O for speed and power
  - Architectural enhancements to reduce work
- In the remainder of this talk I will discuss several of these – and some other system-level design tradeoffs

Fermi SM compared to Kepler SMX

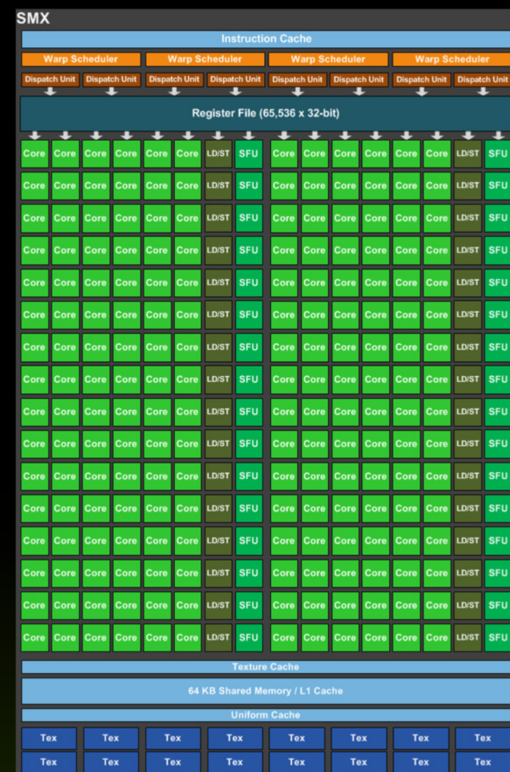




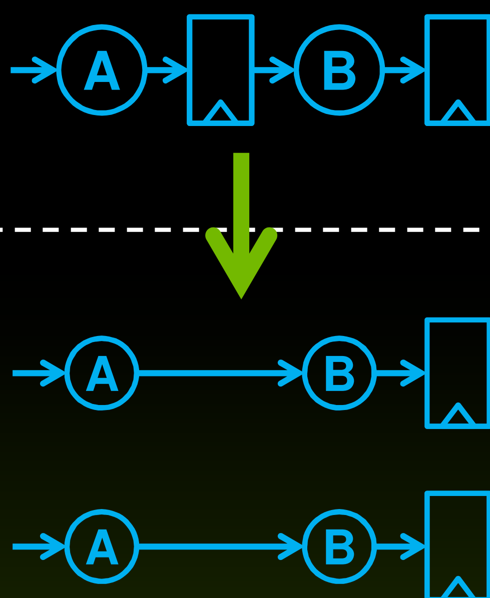
- 3x the math units
- Greatly increased efficiency

SM redesigned with power efficiency in mind

- 2x hardware at ½ clock frequency
  - Reduces power consumption
  - 40nm to 28 nm provides more area
- Overall result
  - SMX Performance is up
  - SMX Power is down
  - Perf/watt metric benefits from both

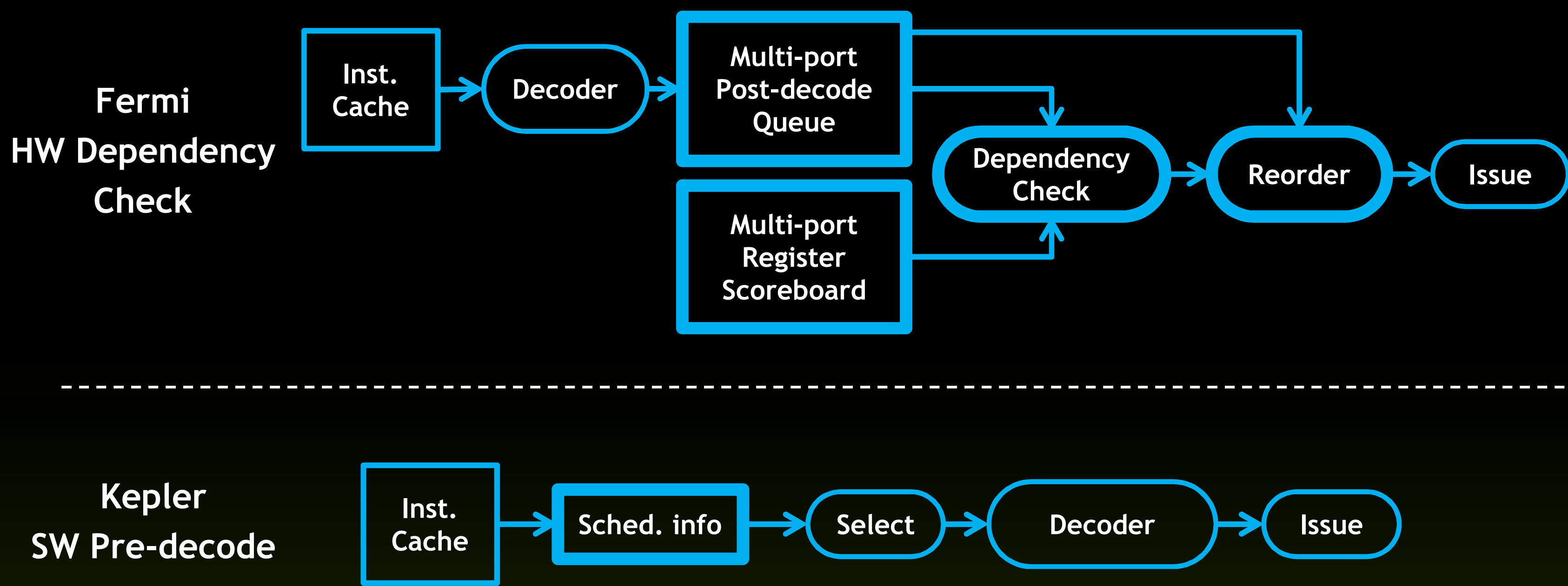


Optimizing for area vs. optimizing for power



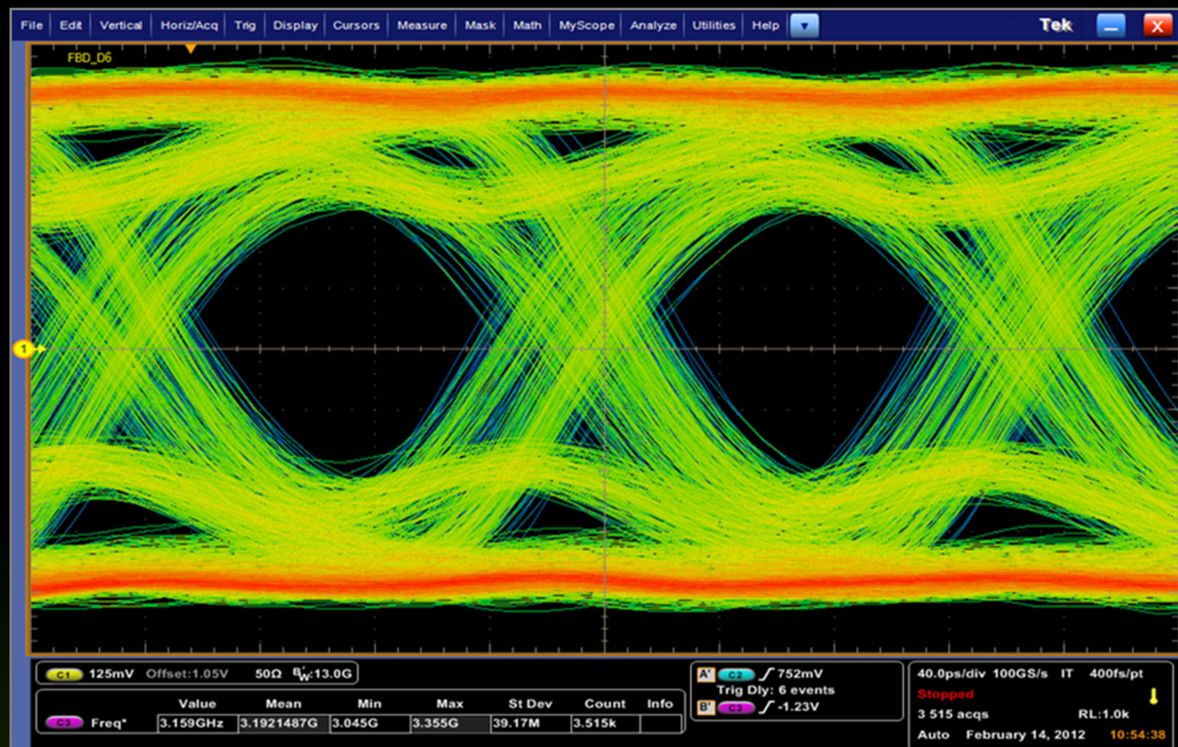
	Logic		Clocking	
	Area	Power	Area	Power
Fermi 2x clock	1.0x	1.0x	1.0x	1.0x
Kepler 1x clock	1.8x	0.9x	1.0x	0.5x

Scheduling complexity moved from hardware to compiler



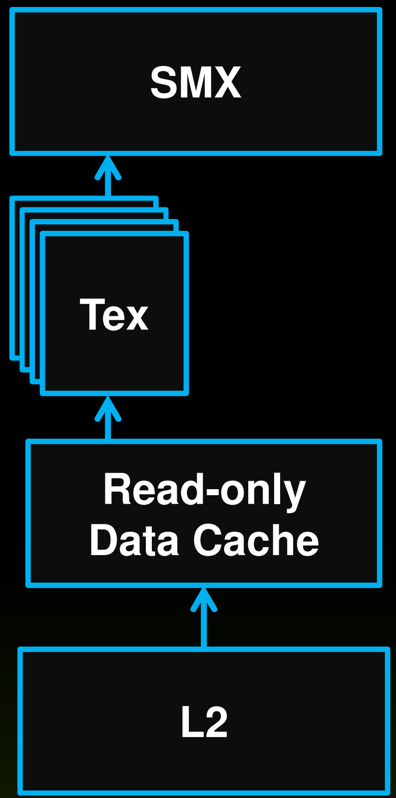
New GDDR5 DRAM controller

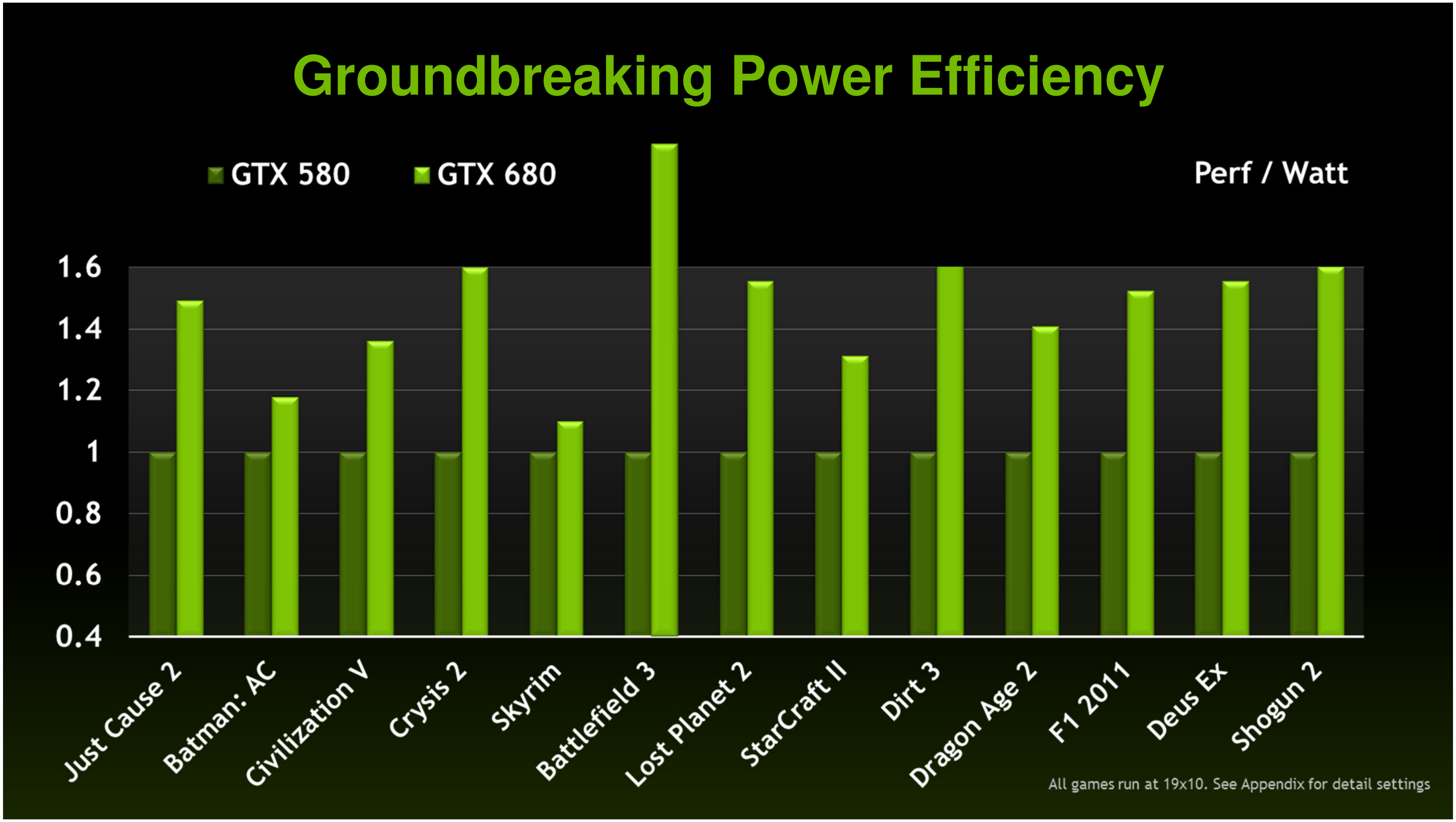
- Clean slate design to:
  - Achieve peak GDDR5 speed
  - Minimize power
- World's first 6Gbps GDDR5



Texture improvements

- SMX vs Fermi SM :
  - 4x filter ops per clock
  - 4x cache capacity
- In most texture-heavy regimes, shader is not limited by texture

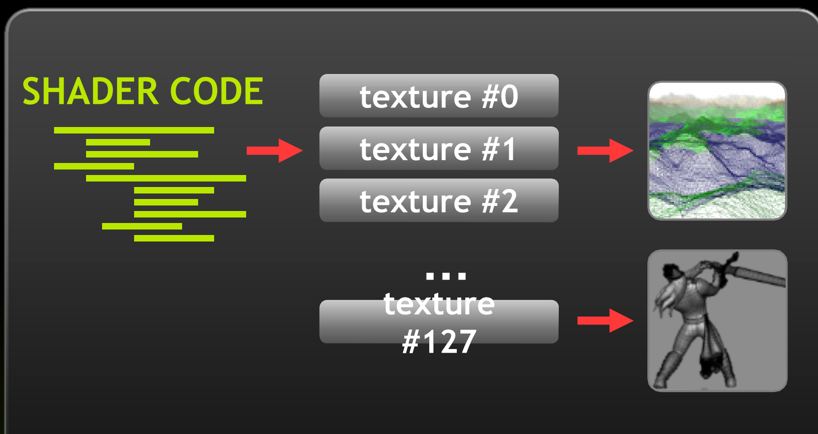



Groundbreaking Power Efficiency
GTX 580
GTX 680
Perf / Watt
1.6
1.4
1.2
1
0.8
0.6
0.4
Just Cause 2
Batman: AC
Civilization V
Crysis 2
Skyrim
Battlefield 3
Lost Planet 2
StarCraft II
Dirt 3
Dragon Age 2
F1 2011
Deus Ex
Shogun 2
All games run at 19x10. See Appendix for detail settings

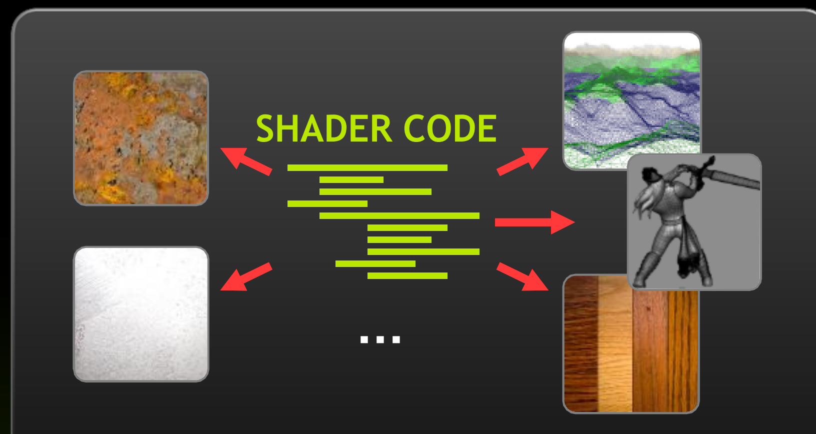
Other Kepler improvements

Bindless Textures

- Dramatic increase in the number of unique textures available to shaders at run-time
- More different materials and richer texture detail in a scene



Pre-Kepler texture binding model
128 simultaneous textures



Kepler bindless textures
over 1M simultaneous textures

Atomic instruction enhancements

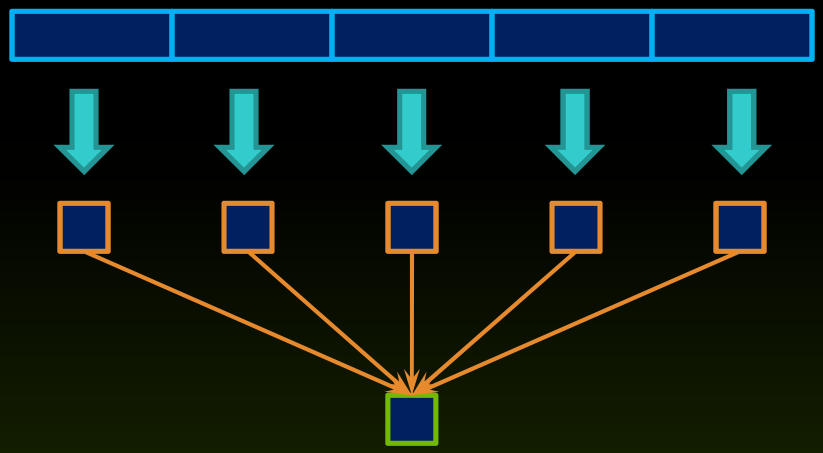
- Shorter processing pipeline
- More atomic processors
- Slowest 10x faster
- Fastest 2x faster
- Added int64 functions to match existing int32

- 2x-10x performance increase

High speed atomics enable new uses

Atomics are now fast enough to use within inner loops

- **Example: Data reduction (sum of all values)**



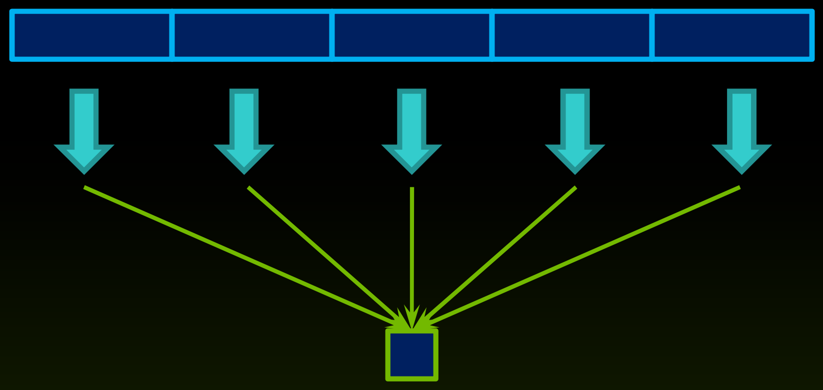
Without Atomics

1. Divide input data array into N sections
2. Launch N blocks, each reduces one section
3. Output is N values
4. Second launch of N threads, reduces outputs to single value

High speed atomics enable new uses

Atomics are now fast enough to use within inner loops

- **Example: Data reduction (sum of all values)**



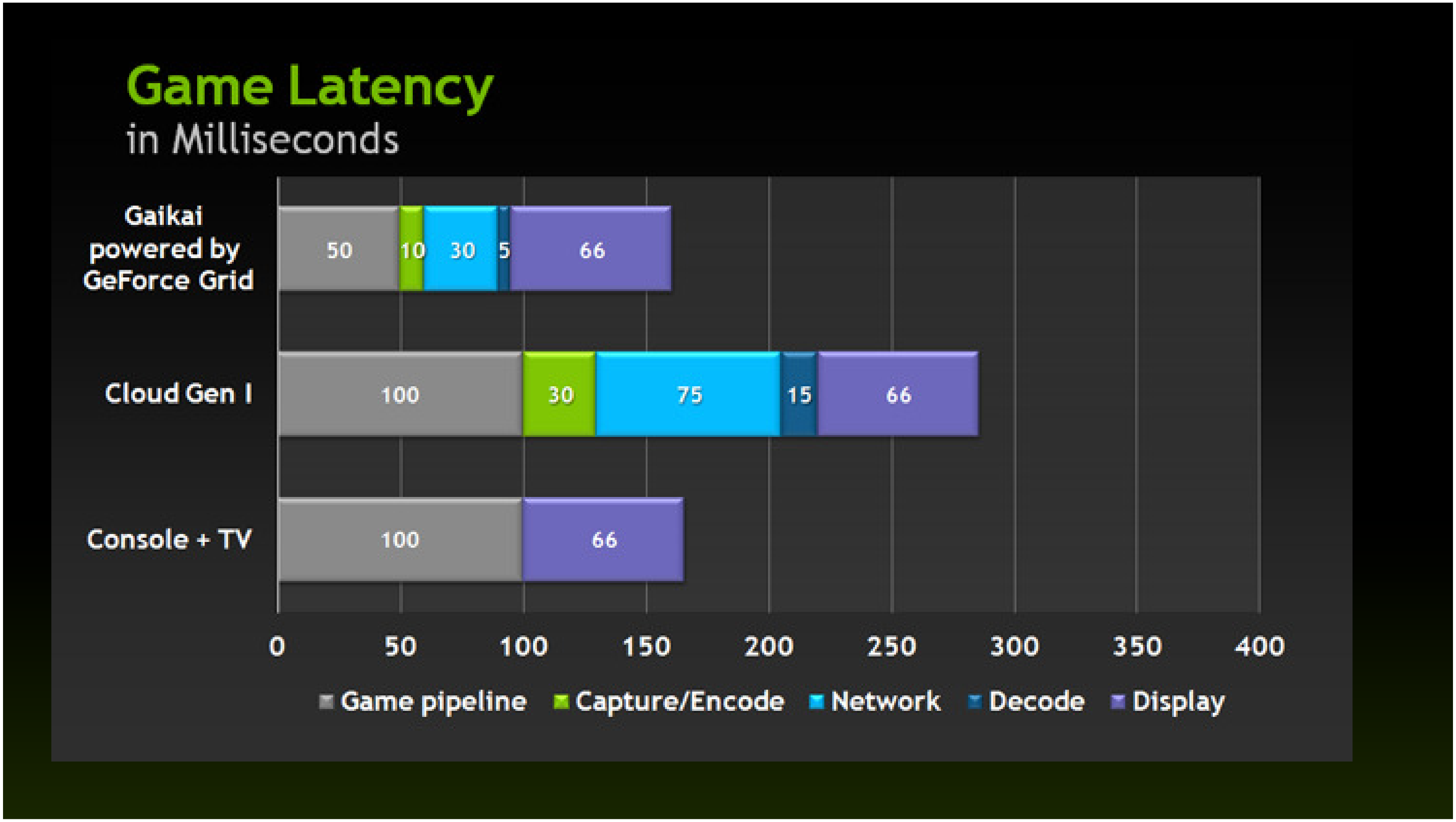
With Atomics

1. Divide input data array into N sections
2. Launch N blocks, each reduces one section
3. Write output directly via atomic.
   No need for second kernel launch.

GPU virtualization enables cloud gaming

- **Kepler host and memory virtualization allow multiple virtual GPUs to be hosted on a single physical GPU**
- **Other critical pieces:**
  - **Fast hardware encoder works directly from render target**
  - **Cloud servers**
  - **Fast, low-latency WAN**
- **Used by Gaikai (www.gaikai.com)**

GeForce VGX
Cloud GPU



Game Latency
in Milliseconds
Gaikai powered by GeForce Grid
50
10
30
5
66
Cloud Gen I
100
30
75
15
66
Console + TV
100
66
0
50
100
150
200
250
300
350
400
Game pipeline
Capture/Encode
Network
Decode
Display

GAIKAI

Showcase | Open Platform | Ad Network | Company

The fastest, most powerful, lowest-latency cloud network on the market

Gaikai offers a bleeding-edge open platform technology that allows the most demanding games and applications to be seamlessly streamed via any connected video capable device including PCs, digital TVs, tablets and smart mobile devices. Having the fastest, lowest-latency, most sophisticated cloud network in the world enables users to experience rich media content as if they were running it locally.

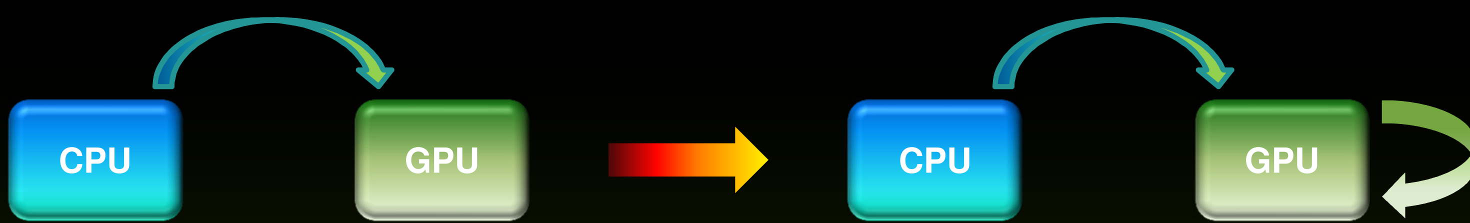
Contact Us

Dynamic Parallelism (GK110+)

What is Dynamic Parallelism?

The ability to launch new work from the GPU

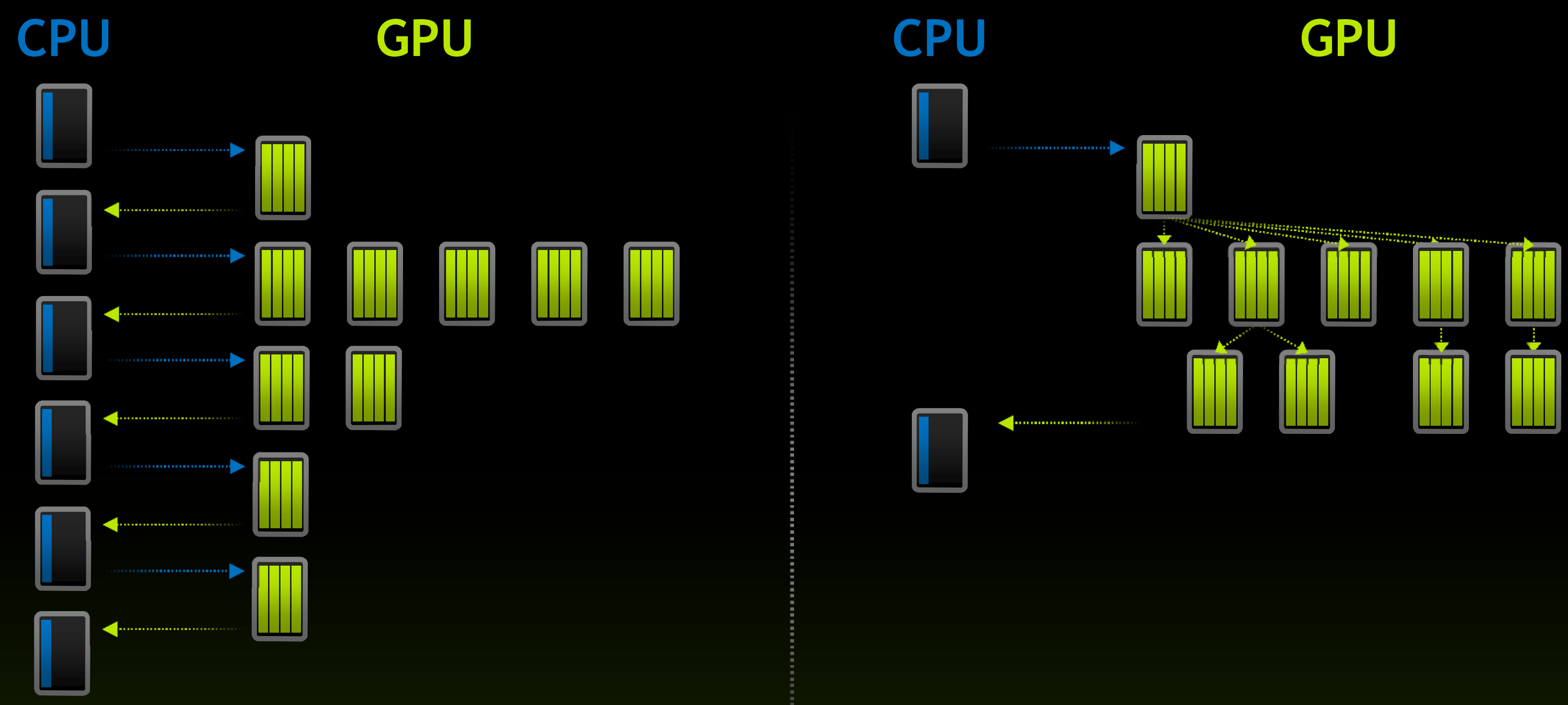
- Dynamically
- Simultaneously
- Independently



Fermi: Only CPU can generate GPU work

Kepler: GPU can generate work for itself

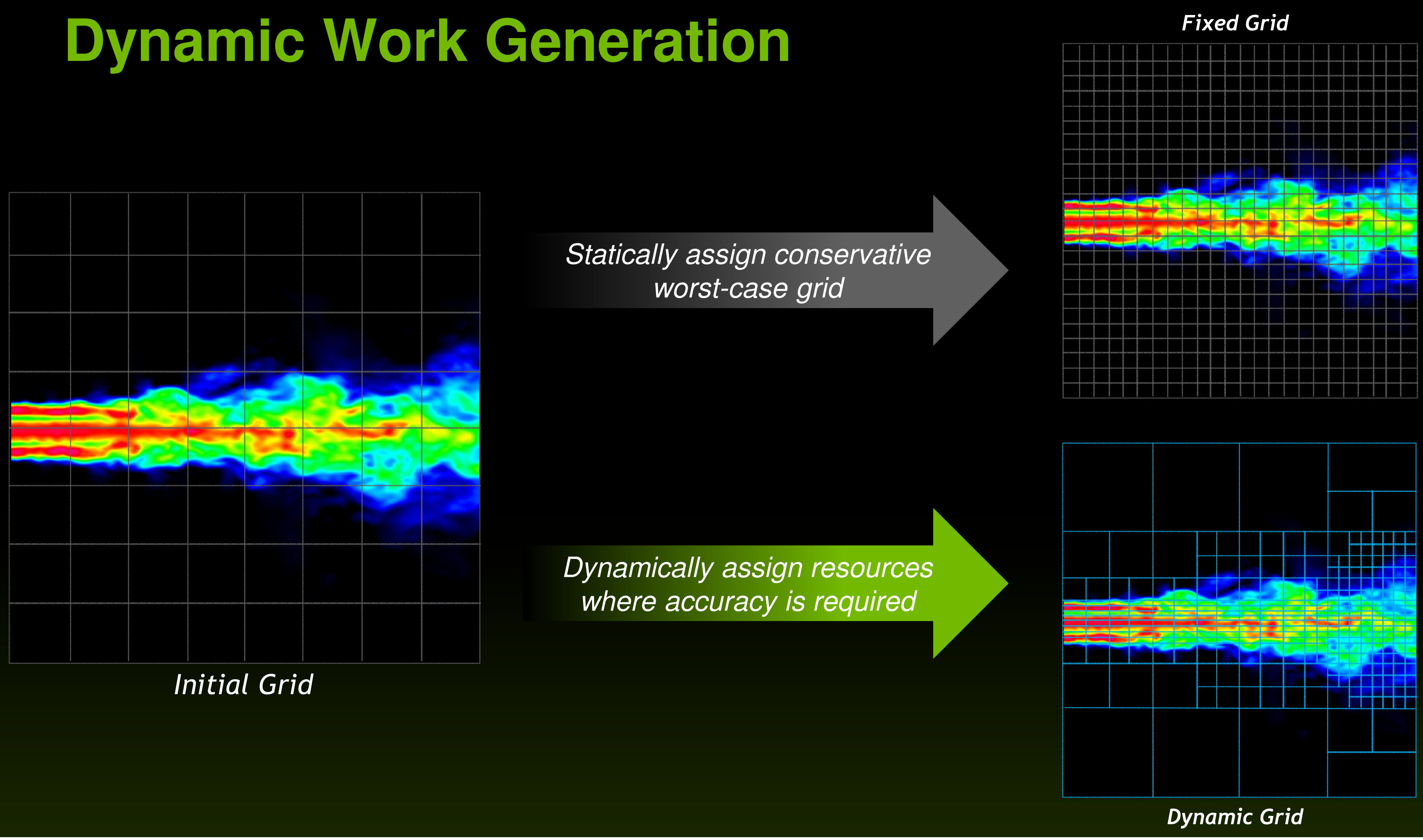
What Does It Mean?



GPU as Co-Processor

Autonomous, Dynamic Parallelism

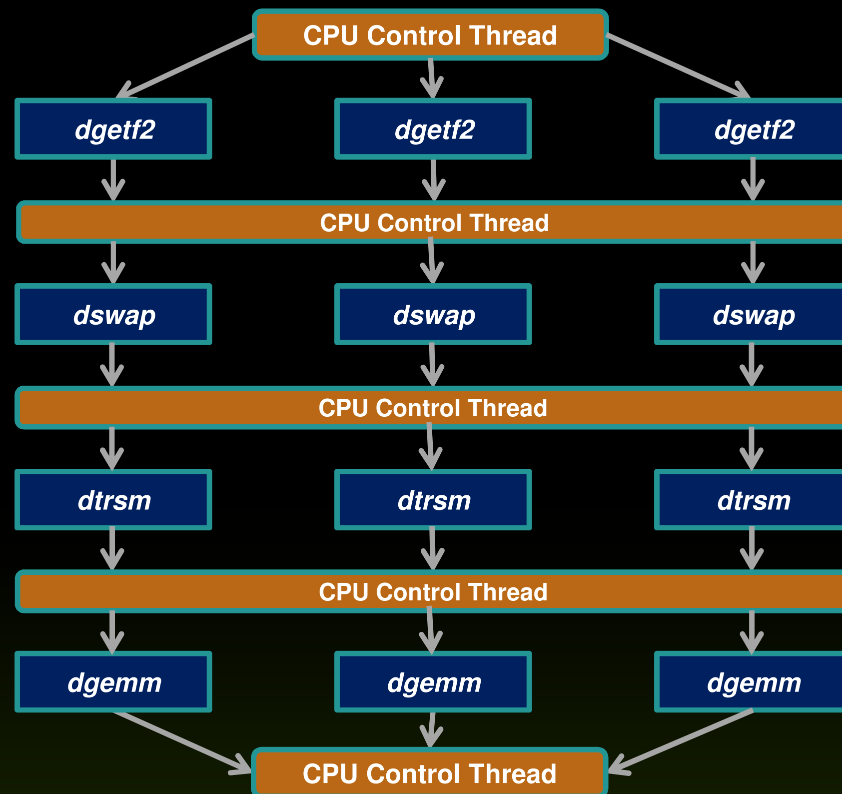
Dynamic Work Generation



Batched & Nested Parallelism

CPU-Controlled Work Batching

- **CPU programs limited by single point of control**
- **Can run at most 10s of threads**
- **CPU is fully consumed with controlling launches**



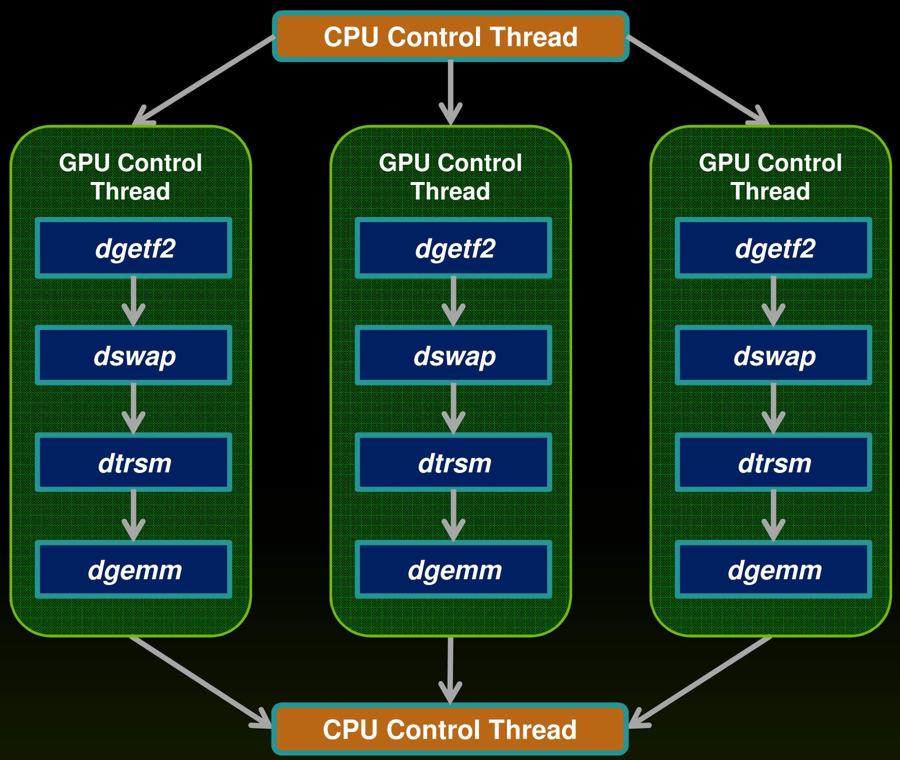
Multiple LU-Decomposition, Pre-Kepler

Algorithm flow simplified for illustrative purposes

Batched & Nested Parallelism

Batching via Dynamic Parallelism

- Move top-level loops to GPU
- Run thousands of independent tasks
- Release CPU for other work



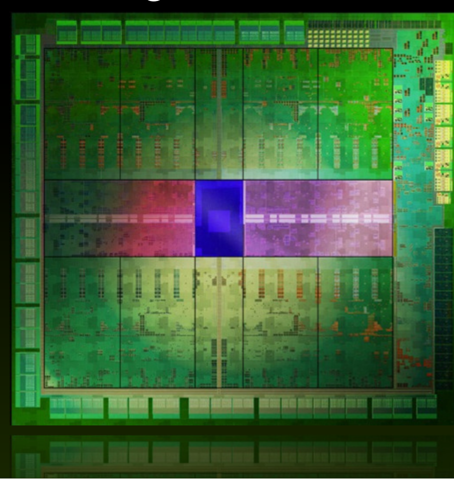
Batched LU-Decomposition, Kepler

Algorithm flow simplified for illustrative purposes

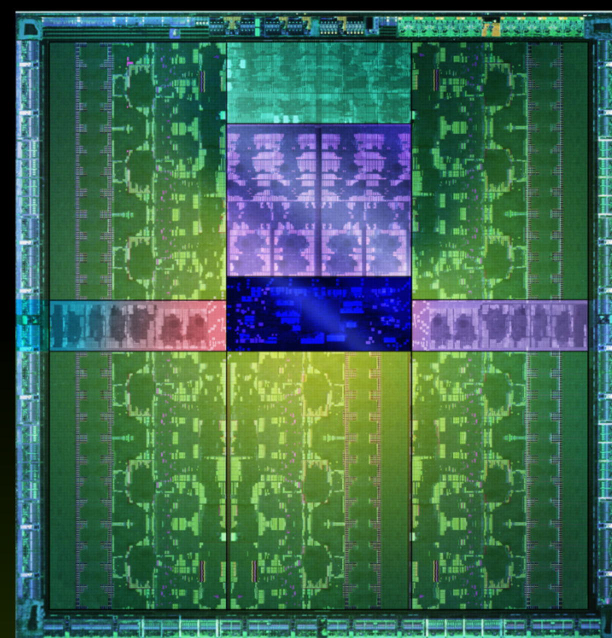
Supporting an Architecture Family

gk107

gk104



gk110



Kepler isn't a chip – it's an architecture family

- Same base architecture must scale over wide range, diverse markets
    - Mobile graphics
    - Consumer desktop and enthusiast graphics
    - Workstation / professional graphics
    - High-performance computing (gk110)

Scaling and feature parameters

- Major configuration parameters (there are many more)
  - Note some are non-power-of-2

Chip	GPCs	SMX per GPC	FBs	L2 size	ECC	Fast FP64
gk104	4	2	4	512K	No	No
gk107	1	2	2	256K	No	No
gk110	5	3	6	1536K	Yes	Yes

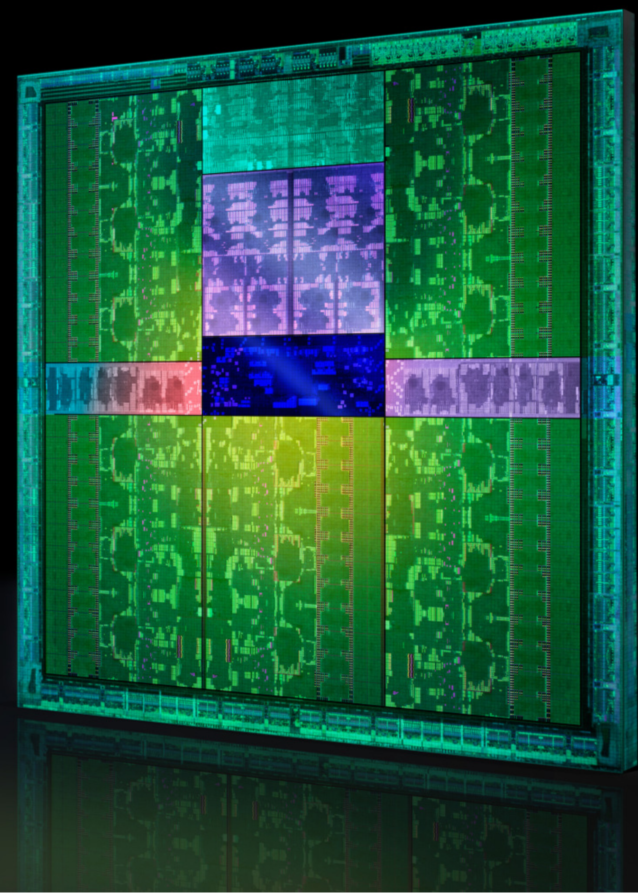
- Market requirements determine configuration
  - Resource balance not the same at each level
  - Can change during project development
  - Arch model is quick; a lot of work remains for physical + pad design

Summary: World’s fastest and most efficient GPUs

Performance

Programmability

Efficiency



Lots more

- GPU Boost
- TXAA
- Adaptive Vsync
- New shader instructions

For more information:

- Kepler whitepaper:
  - http://www.geforce.com/Active/en_US/en_US/pdf/GeForce-GTX-680-Whitepaper-FINAL.pdf
- GeForce GRID (cloud gaming):
  - http://www.geforce.com/whats-new/articles/geforce-grid
- GPU Technology Conference presentations:
  - www.gputechconf.com

Acknowlegments: thanks to Lars Nyland, James Wang, and Jonah Alben for many of the slides used here and to the entire Kepler team.