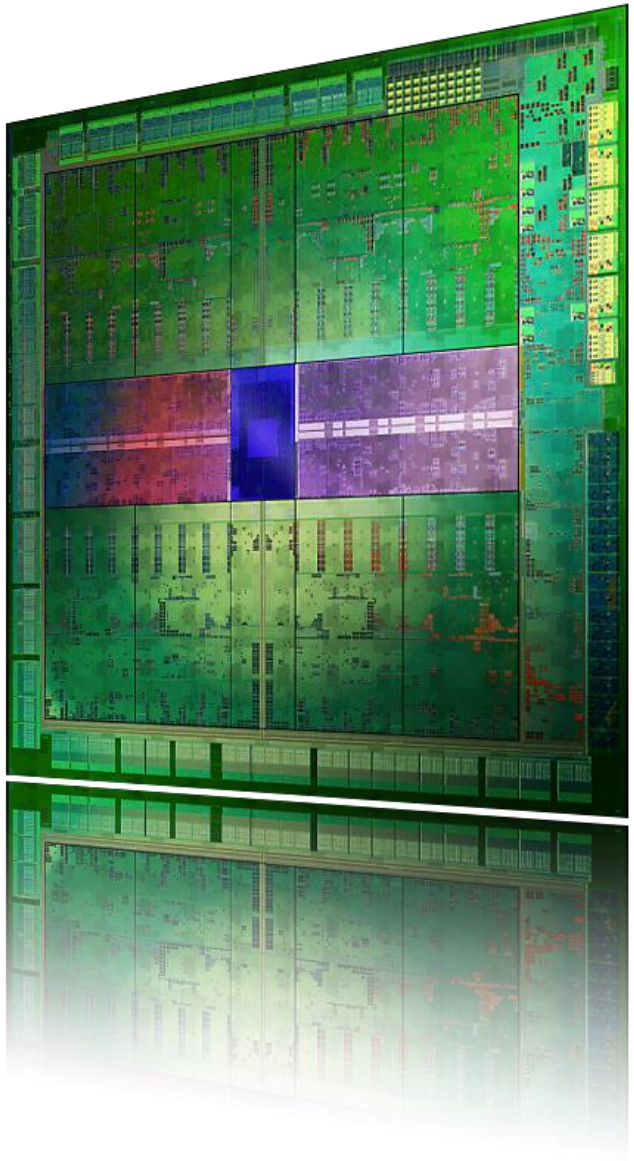
GPUDirect RDMA and Green Multi-GPU Architectures

GE Intelligent Platforms
Mil/Aero Embedded Computing

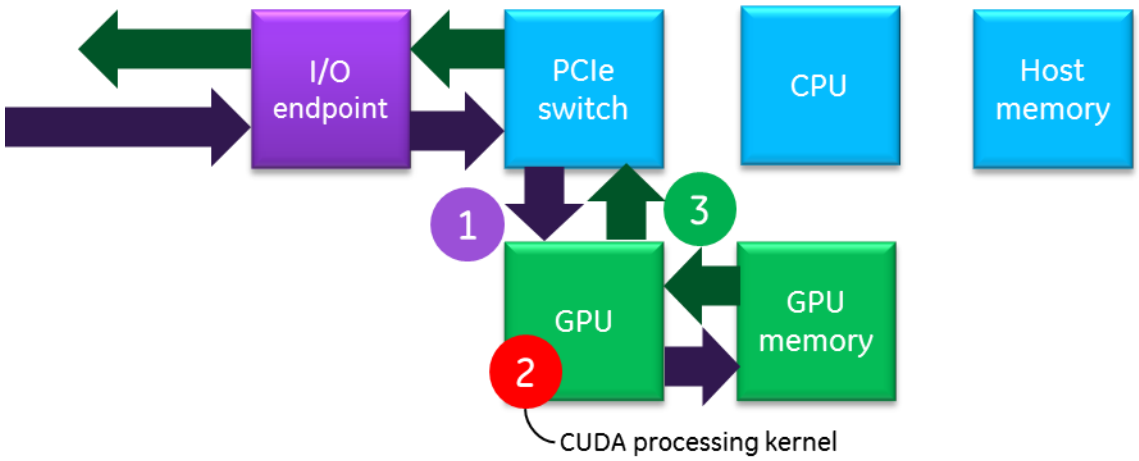
Dustin Franklin, GPGPU Applications Engineer
dustin.franklin@ge.com
443.310.9812 (Washington, DC)



imagination at work

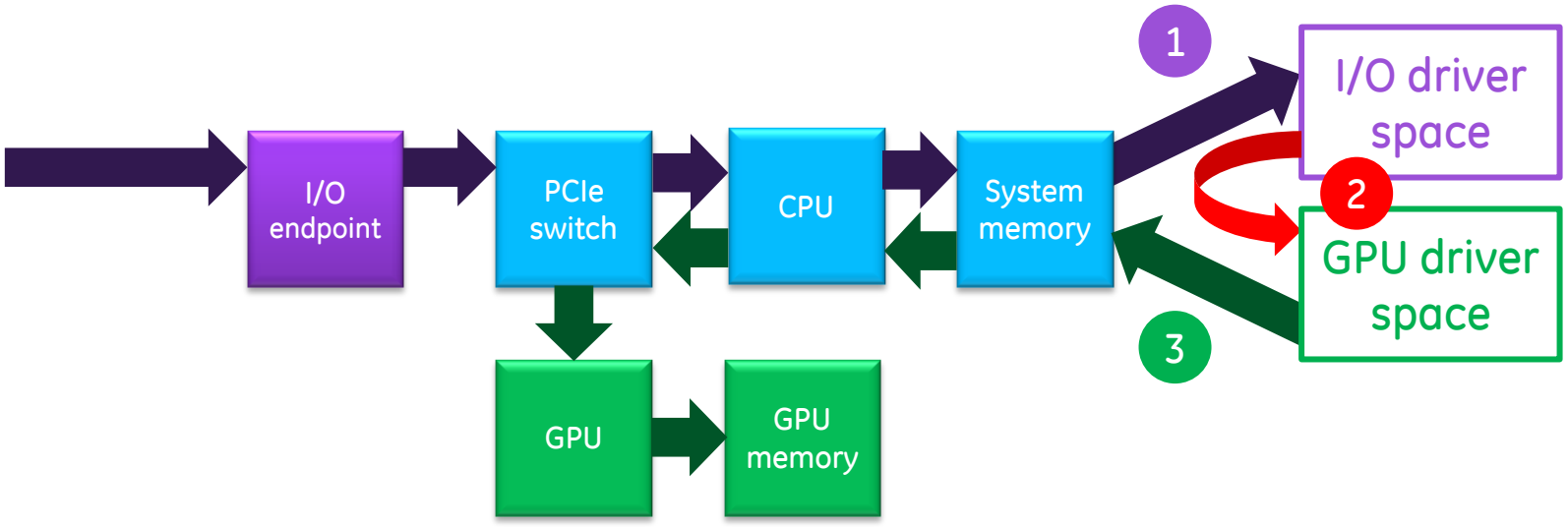
What this talk is about

- GPU Autonomy
- GFLOPS/watt and SWAP
- Project Denver
- Exascale



Without GPUDirect

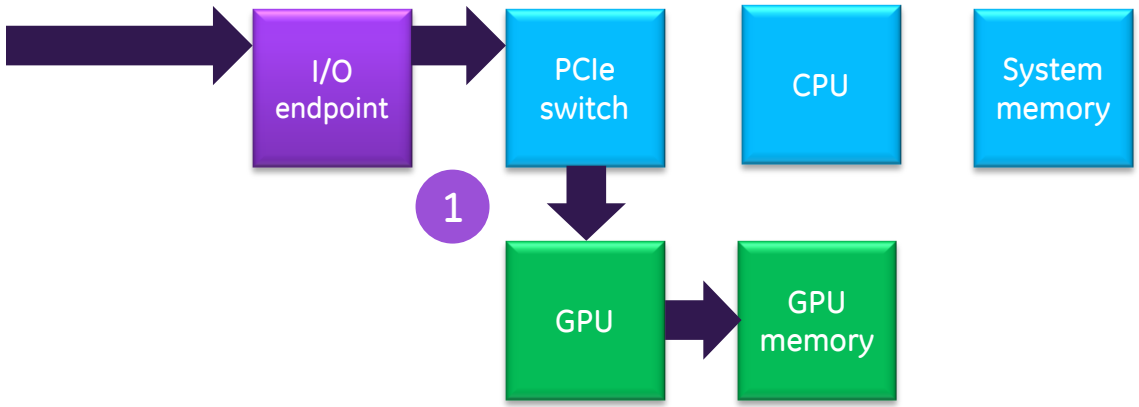
- In a standard plug & play OS, the two drivers have separate DMA buffers in system memory
- Three transfers to move data between I/O endpoint and GPU



1. I/O endpoint DMA's into system memory
2. CPU copies data from I/O driver DMA buffer into GPU DMA buffer
3. GPU DMA's from system memory into GPU memory

GPUDirect RDMA

- I/O endpoint and GPU communicate directly, only one transfer required.
- Traffic limited to PCIe switch, no CPU involvement in DMA
- x86 CPU is still necessary to have in the system, to run NVIDIA driver



1 I/O endpoint DMA's into GPU memory

Endpoints

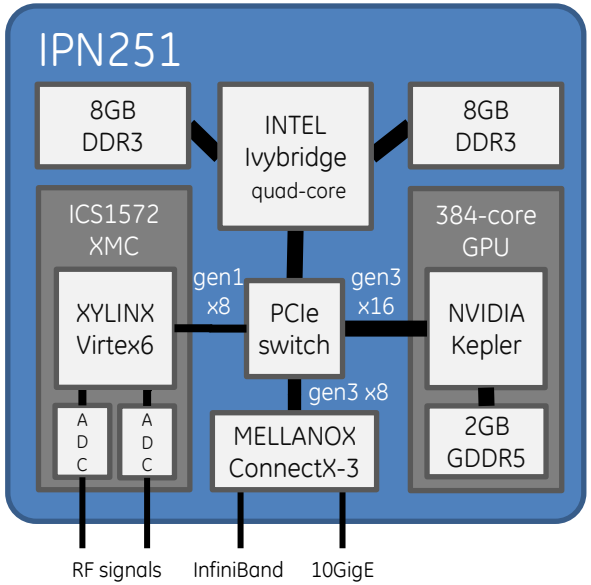
- **GPUDirect RDMA is flexible and works with a wide range of existing devices**
  - Built on open PCIe standards
  - Any I/O device that has a PCIe endpoint and DMA engine can utilize GPUDirect RDMA
  - GPUDirect permeates both the frontend ingest and backend interconnects

    - **FPGAs**
    - **Ethernet / InfiniBand adapters**
    - **Storage devices**
    - **Video capture cards**
    - **PCIe non-transparent (NT) ports**

- **It's free.**
  - Supported in CUDA 5.0 and Kepler
  - Users can leverage APIs to implement RDMA with 3rd-party endpoints in their system

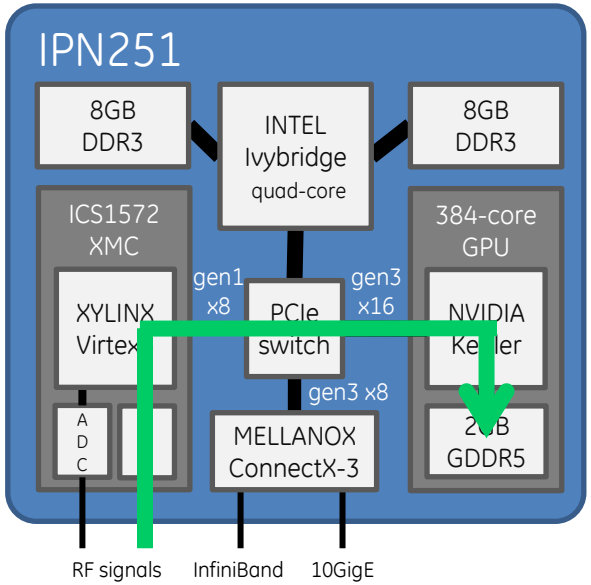
- **Practically no integration required**
  - No changes to device HW
  - No changes to CUDA algorithms
  - I/O device drivers need to use DMA addresses of GPU instead of SYSRAM pages

Frontend Ingest



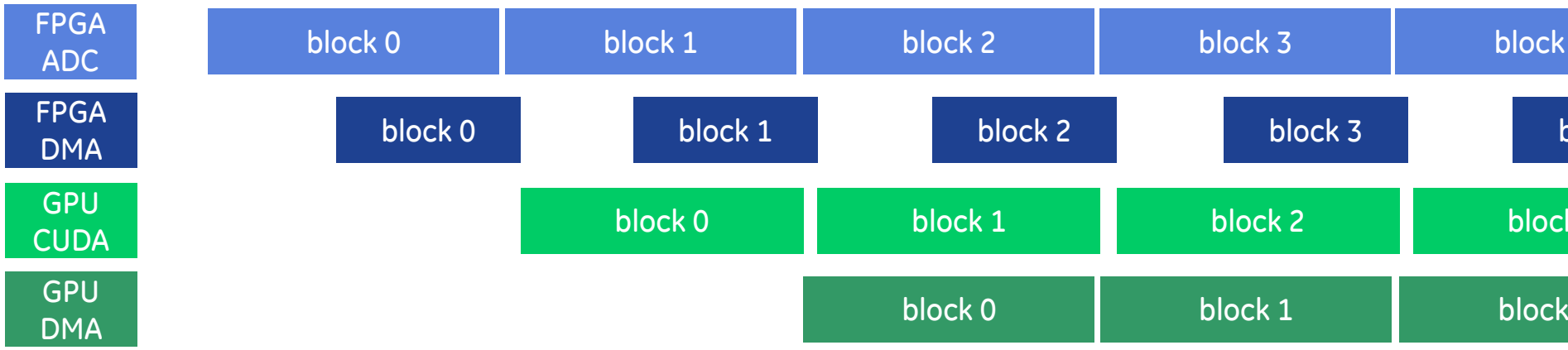
DMA Size	DMA latency (µs)			DMA throughput (MB/s)	
	no RDMA	with RDMA	Δ	no RDMA	with RDMA
16 KB	65.06 µs	4.09 µs	↓15.9X	125 MB/s	2000 MB/s
32 KB	77.38 µs	8.19 µs	↓9.5X	211 MB/s	2000 MB/s
64 KB	124.03 µs	16.38 µs	↓7.6X	264 MB/s	2000 MB/s
128 KB	208.26 µs	32.76 µs	↓6.4X	314 MB/s	2000 MB/s
256 KB	373.57 µs	65.53 µs	↓5.7X	350 MB/s	2000 MB/s
512 KB	650.52 µs	131.07 µs	↓5.0X	402 MB/s	2000 MB/s
1024 KB	1307.90 µs	262.14 µs	↓4.9X	400 MB/s	2000 MB/s
2048 KB	2574.33 µs	524.28 µs	↓4.9X	407 MB/s	2000 MB/s

Frontend Ingest



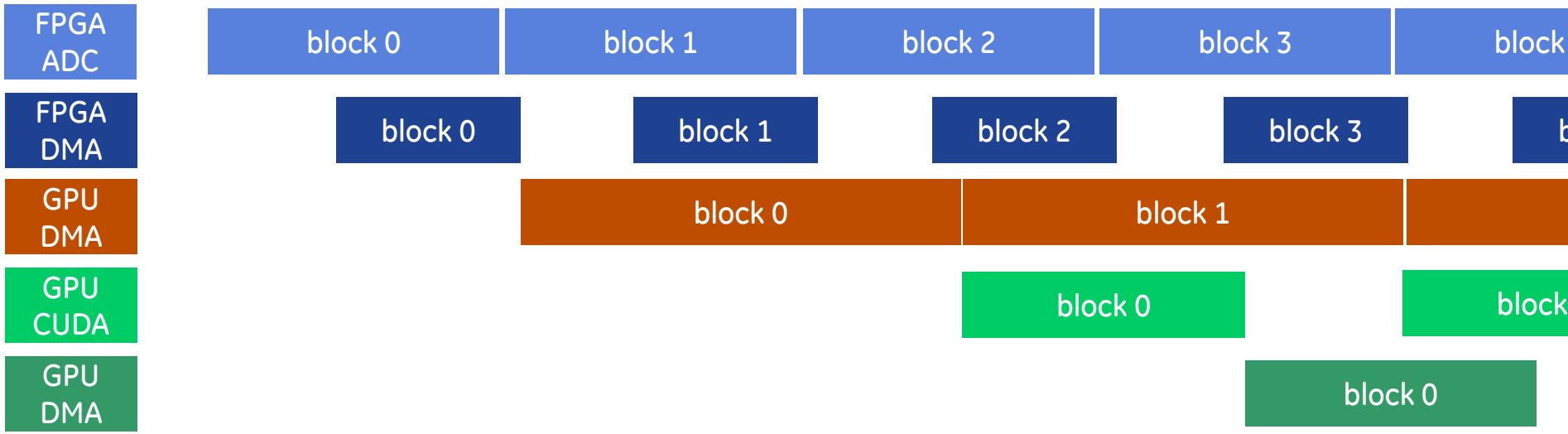
DMA Size	DMA latency (µs)			DMA throughput (MB/s)	
	no RDMA	with RDMA	Δ	no RDMA	with RDMA
16 KB	65.06 µs	4.09 µs	↓15.9X	125 MB/s	2000 MB/s
32 KB	77.38 µs	8.19 µs	↓9.5X	211 MB/s	2000 MB/s
64 KB	124.03 µs	16.38 µs	↓7.6X	264 MB/s	2000 MB/s
128 KB	208.26 µs	32.76 µs	↓6.4X	314 MB/s	2000 MB/s
256 KB	373.57 µs	65.53 µs	↓5.7X	350 MB/s	2000 MB/s
512 KB	650.52 µs	131.07 µs	↓5.0X	402 MB/s	2000 MB/s
1024 KB	1307.90 µs	262.14 µs	↓4.9X	400 MB/s	2000 MB/s
2048 KB	2574.33 µs	524.28 µs	↓4.9X	407 MB/s	2000 MB/s

Pipeline with GPUDirect RDMA



FPGA DMA	Transfer block directly to GPU via PCIe switch
GPU CUDA	CUDA DSP kernels (FIR, FFT, ect.)
GPU DMA	Transfer results to next processor

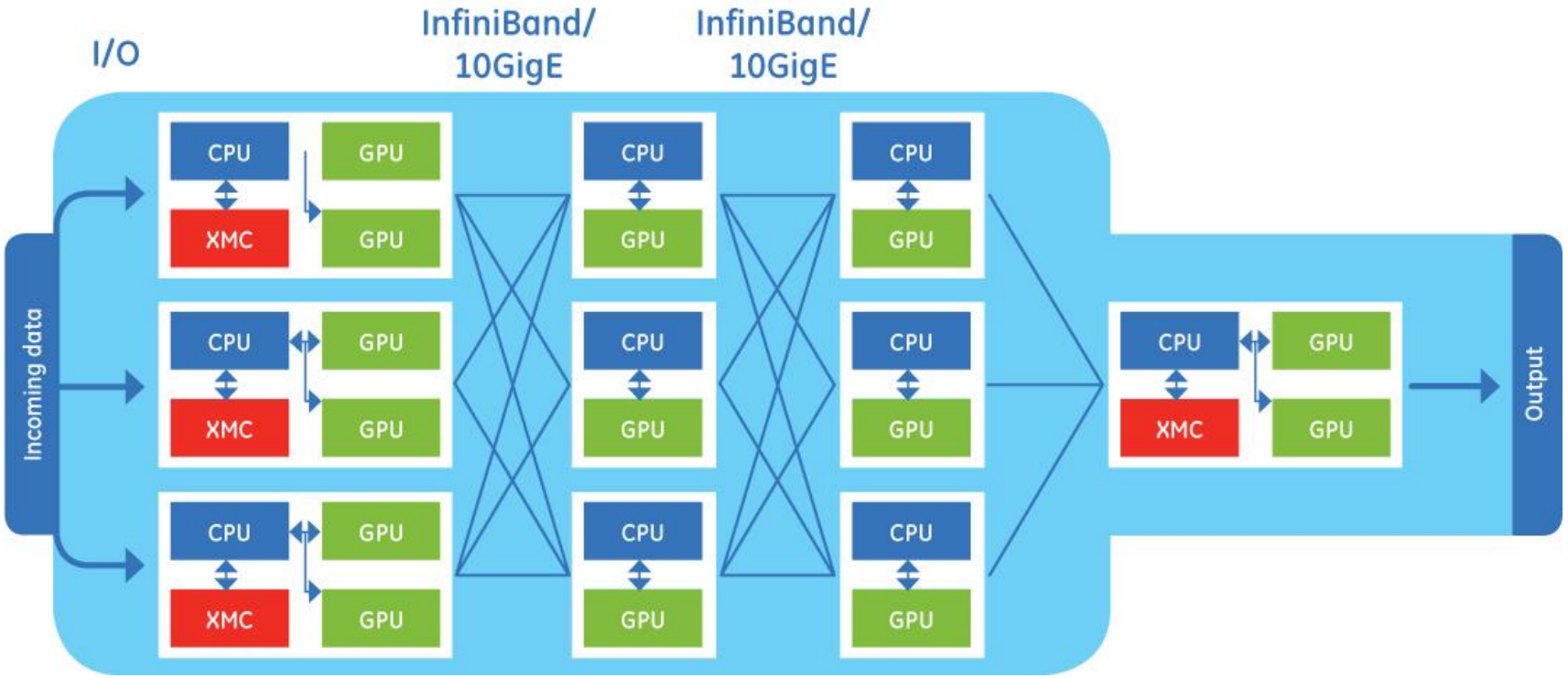
Pipeline without GPUDirect RDMA



FPGA DMA	Transfer block to system memory
GPU DMA	Transfer from system memory to GPU
GPU CUDA	CUDA DSP kernels (FIR, FFT, ect.)
GPU DMA	Transfer results to next processor

Backend Interconnects

- Utilize GPUDirect RDMA across the network for low-latency IPC and system scalability

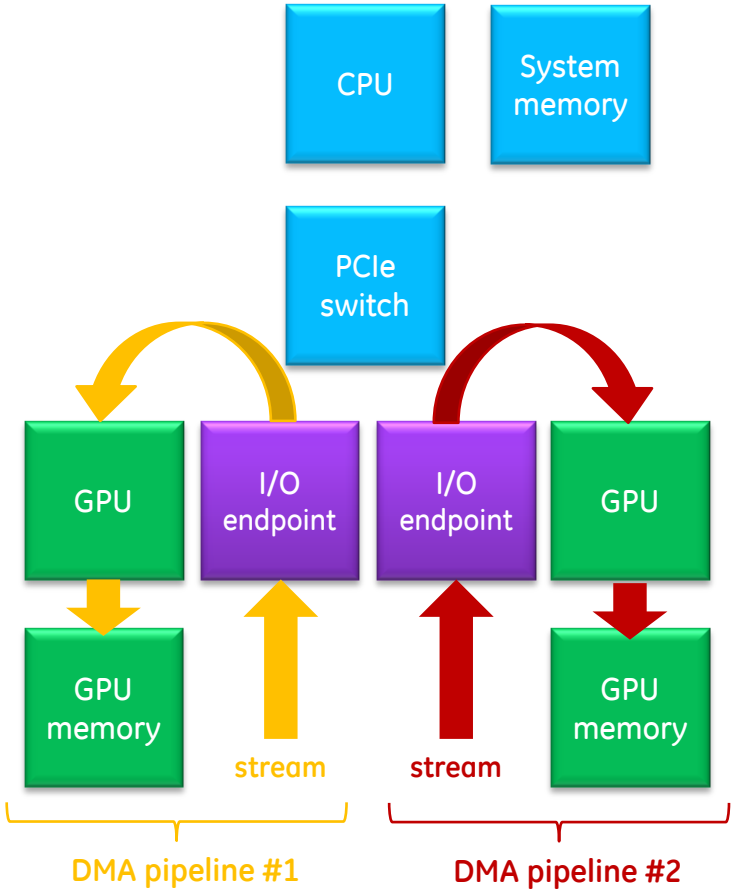


- Mellanox OFED integration with GPUDirect RDMA – Q2 2013

Topologies

- GPUDirect RDMA works in many system topologies
  - Single I/O endpoint and single GPU
  - Single I/O endpoint and multiple GPUs
  - Multiple I/O endpoints and single GPU
  - Multiple I/O endpoints and multiple GPUs

  } with or without PCIe switch downstream of CPU



Impacts of GPUDirect

- **Decreased latency**
  - Eliminate redundant copies over PCIe + added latency from CPU
  - ~5x reduction, depending on the I/O endpoint
  - Perform round-trip operations on GPU in microseconds, not milliseconds

enables new CUDA applications

- **Increased PCIe efficiency + bandwidth**
  - bypass system memory, MMU, root complex: limiting factors for GPU DMA transfers

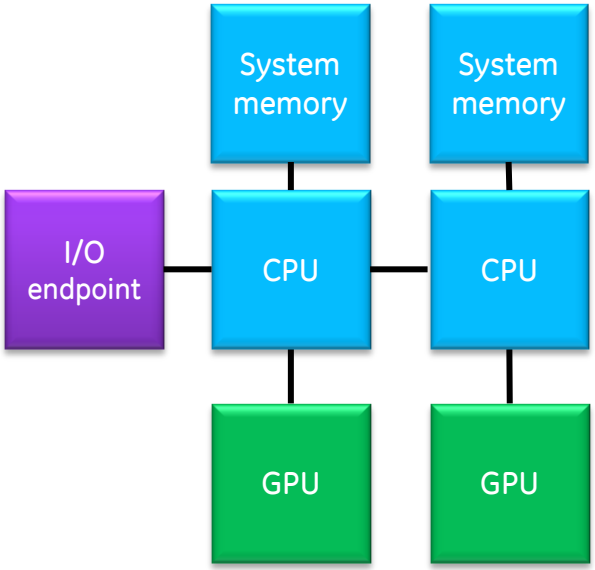
- **Decrease in CPU utilization**
  - CPU is no longer burning cycles shuffling data around for the GPU
  - System memory is no longer being thrashed by DMA engines on endpoint + GPU
  - Go from 100% core utilization per GPU to < 10% utilization per GPU

promotes multi-GPU

Before GPUDirect…

- Many multi-GPU systems had 2 GPU nodes
- Additional GPUs quickly choked system memory and CPU resources

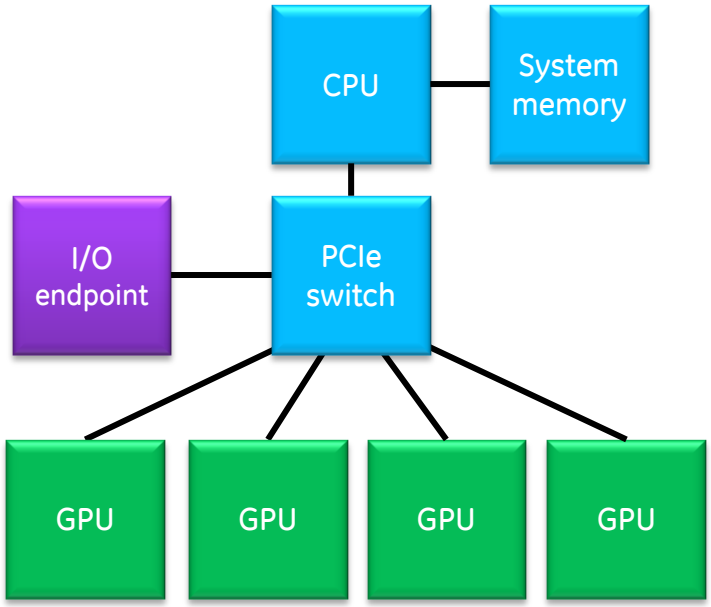
- Dual-socket CPU design very common
- Dual root-complex prevents P2P across CPUs (QPI/HT untraversable)



GFLOPS/watt			
	Xeon E5	K20X	system
SGEMM	2.32	12.34	8.45
DGEMM	1.14	5.19	3.61

Rise of Multi-GPU

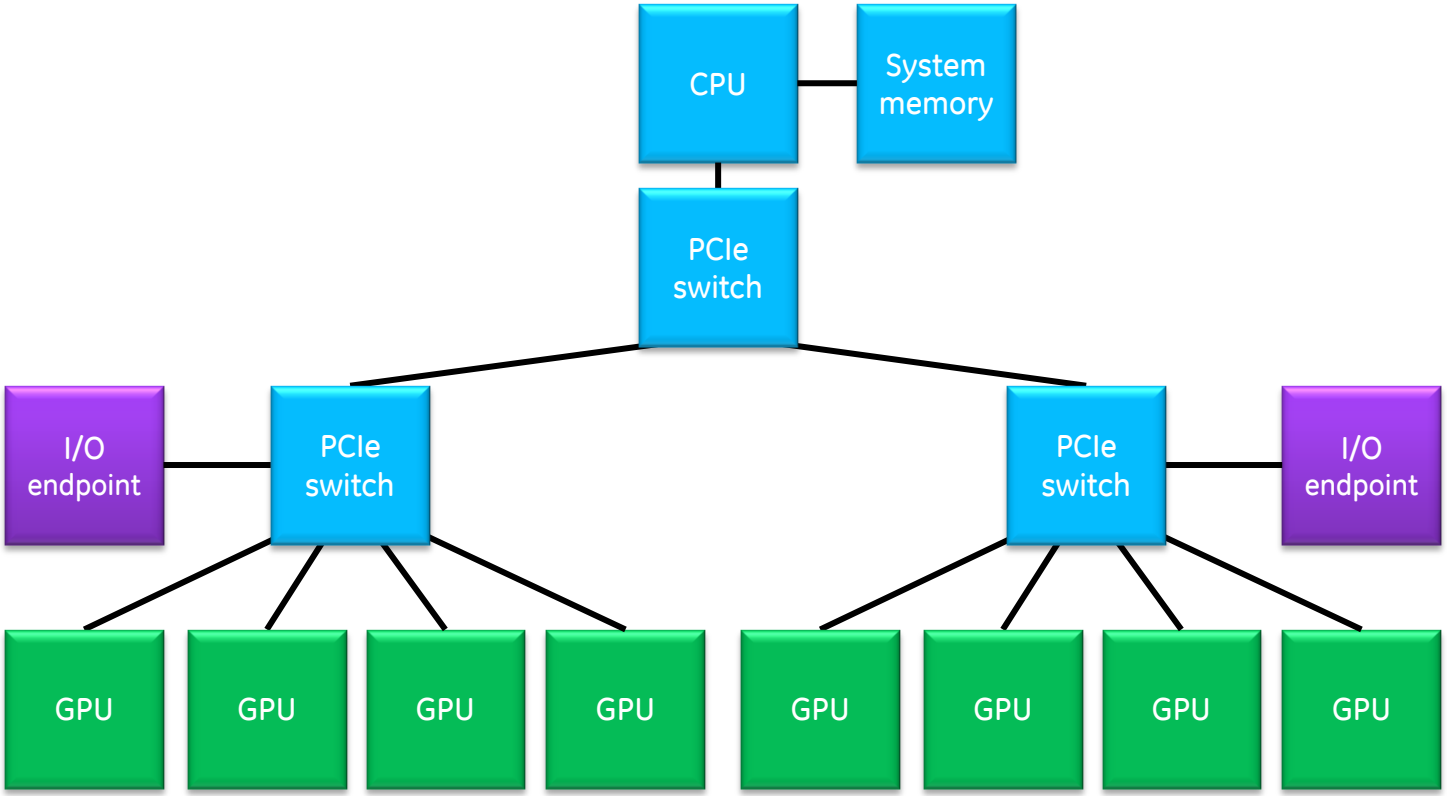
- Higher GPU-to-CPU ratios permitted by increased GPU autonomy from GPUDirect
- PCIe switches integral to design, for true CPU bypass and fully-connected P2P



GFLOPS/watt		
GPU:CPU ratio	1 to 1	4 to 1
SGEMM	8.45	10.97
DGEMM	3.61	4.64

Nested PCIe switches

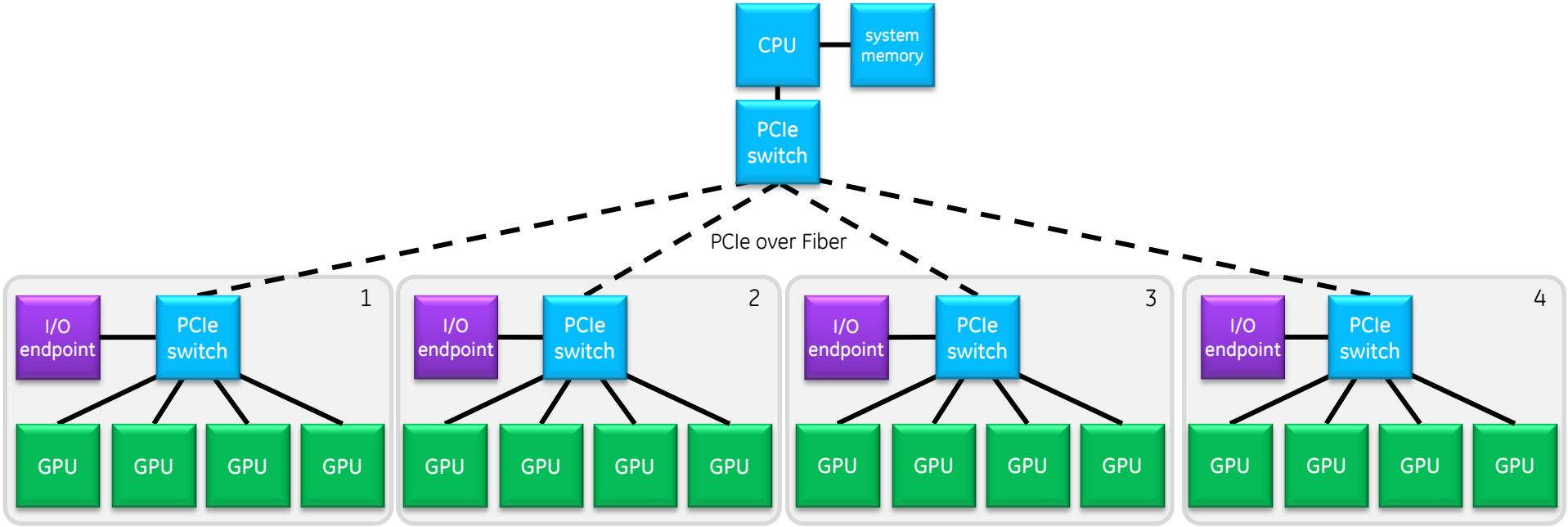
- Nested hierarchies avert 96-lane limit on current PCIe switches



GFLOPS/watt			
GPU:CPU ratio	1 to 1	4 to 1	8 to 1
SGEMM	8.45	10.97	11.61
DGEMM	3.61	4.64	4.91

PCIe over Fiber

- SWaP is our new limiting factor
- PCIe over Fiber-Optic can interconnect expansion blades and assure GPU:CPU growth
- Supports PCIe gen3 over at least 100 meters



GFLOPS/watt				
GPU:CPU ratio	1 to 1	4 to 1	8 to 1	16 to 1
SGEMM	8.45	10.97	11.61	11.96
DGEMM	3.61	4.64	4.91	5.04

Scalability – 10 petaflops

Processor Power Consumption for 10 petaflops				
GPU:CPU ratio	1 to 1	4 to 1	8 to 1	16 to 1
SGEMM	1184 kW	911 kW	862 kW	832 kW
DGEMM	2770 kW	2158 kW	2032 kW	1982 kW

Yearly Energy Bill				
GPU:CPU ratio	1 to 1	4 to 1	8 to 1	16 to 1
SGEMM	$1,050,326	$808,148	$764,680	$738,067
DGEMM	$2,457,266	$1,914,361	$1,802,586	$1,758,232

Efficiency Savings				
GPU:CPU ratio	1 to 1	4 to 1	8 to 1	16 to 1
SGEMM	--	23.05%	27.19%	29.73%
DGEMM	--	22.09%	26.64%	28.45%

Road to Exascale

GPUDirect RDMA

Integration with peripherals & interconnects for system scalability

Project Denver

Hybrid CPU/GPU for heterogeneous compute

Project Osprey

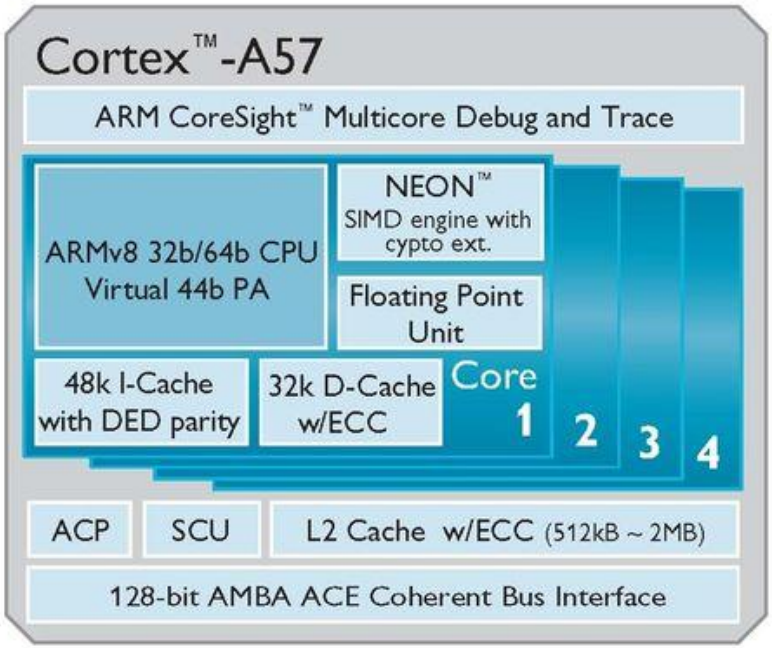
Parallel microarchitecture & fab process optimizations for power efficiency

Software

CUDA, OpenCL, OpenACC
Drivers & Mgmt Layer
Hybrid O/S

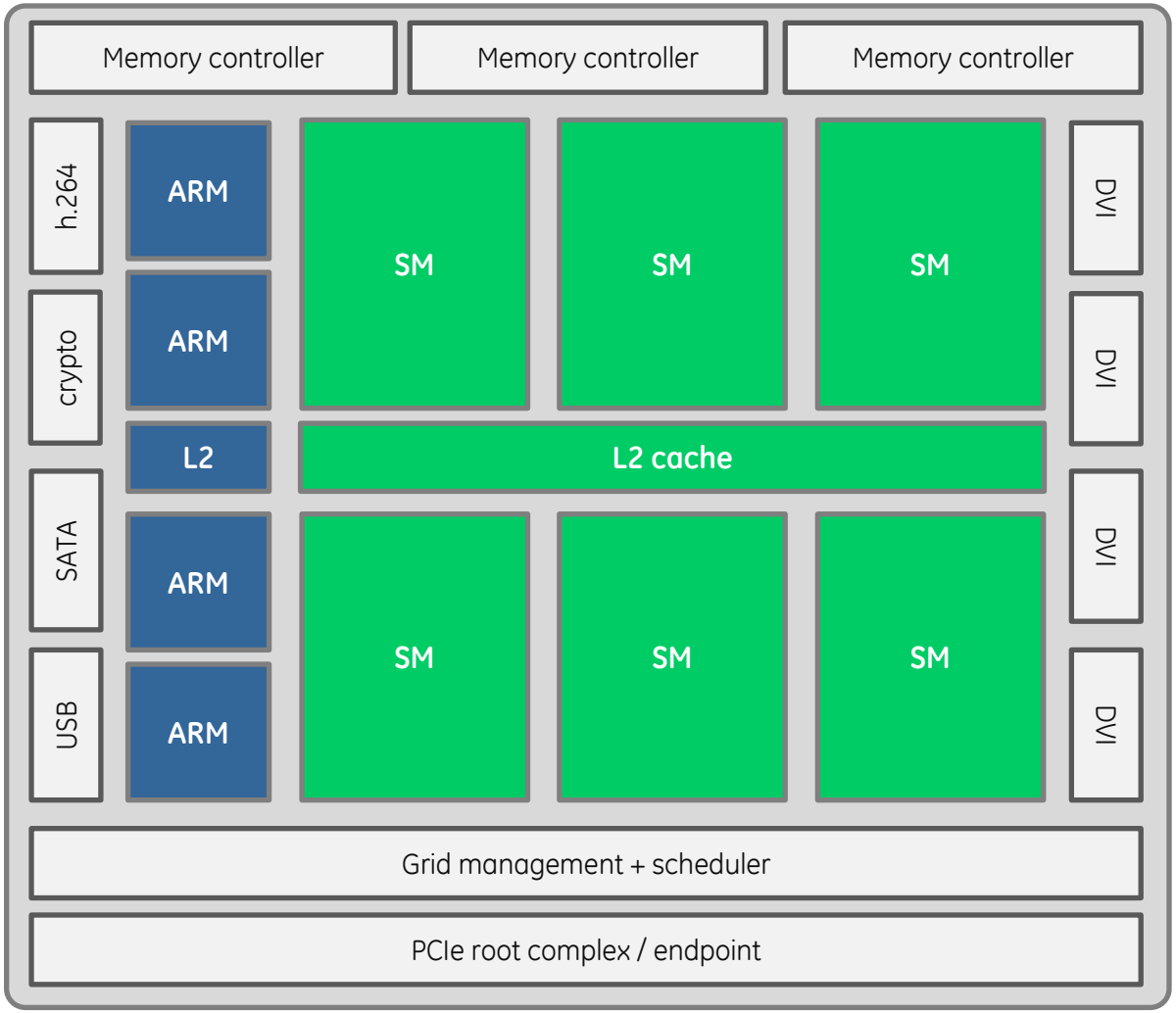
Project Denver

- 64-bit ARMv8 architecture
- ISA by ARM, chip by NVIDIA
- ARM's RISC-based approach aligns with NVIDIA's perf/Watt initiative

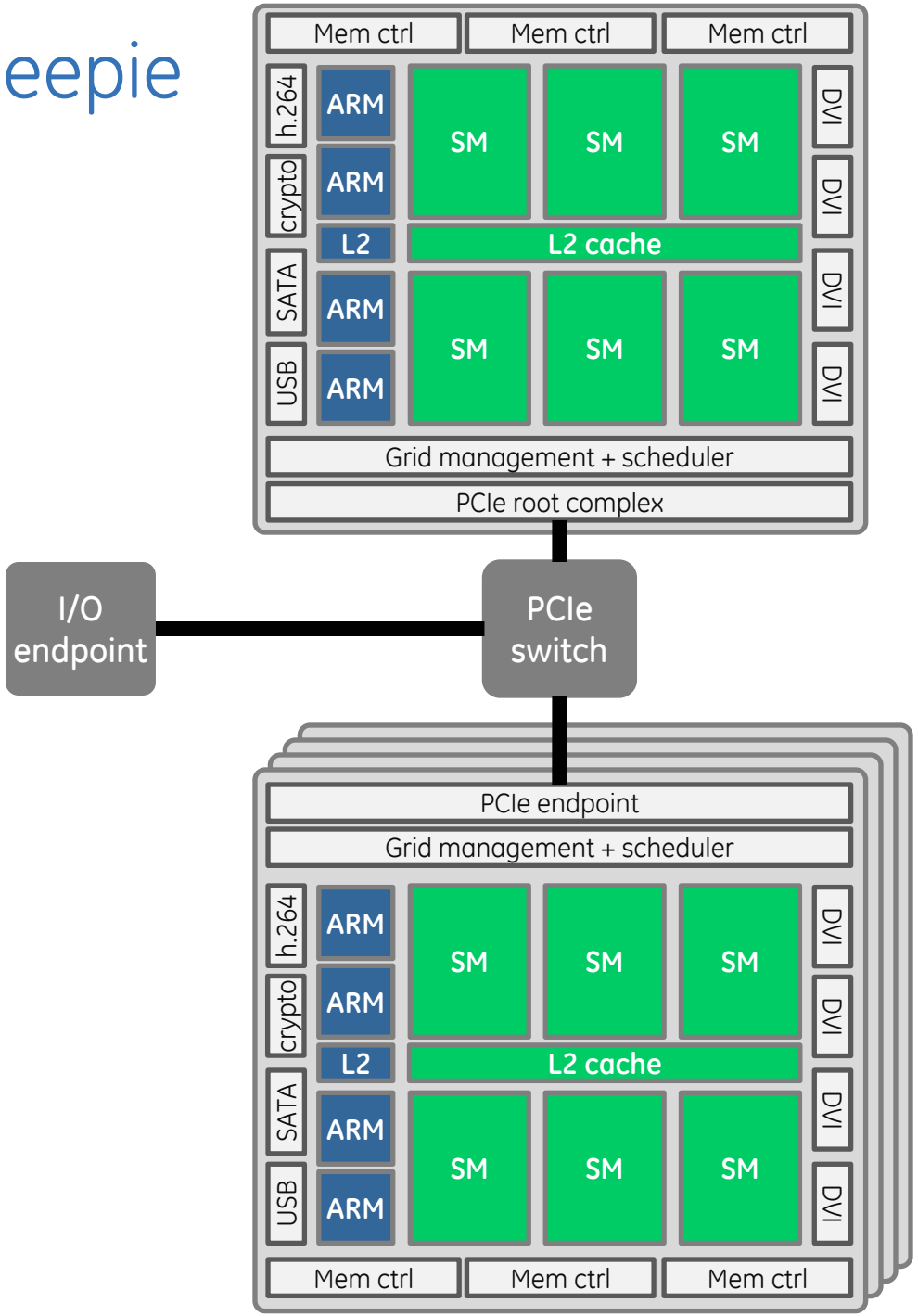


- Unlike licensed Cortex-A53 and –A57 cores, NVIDIA's cores are highly customized
- Design flexibility required for tight CPU/GPU integration

ceepie-geepie



ceepie-geepie

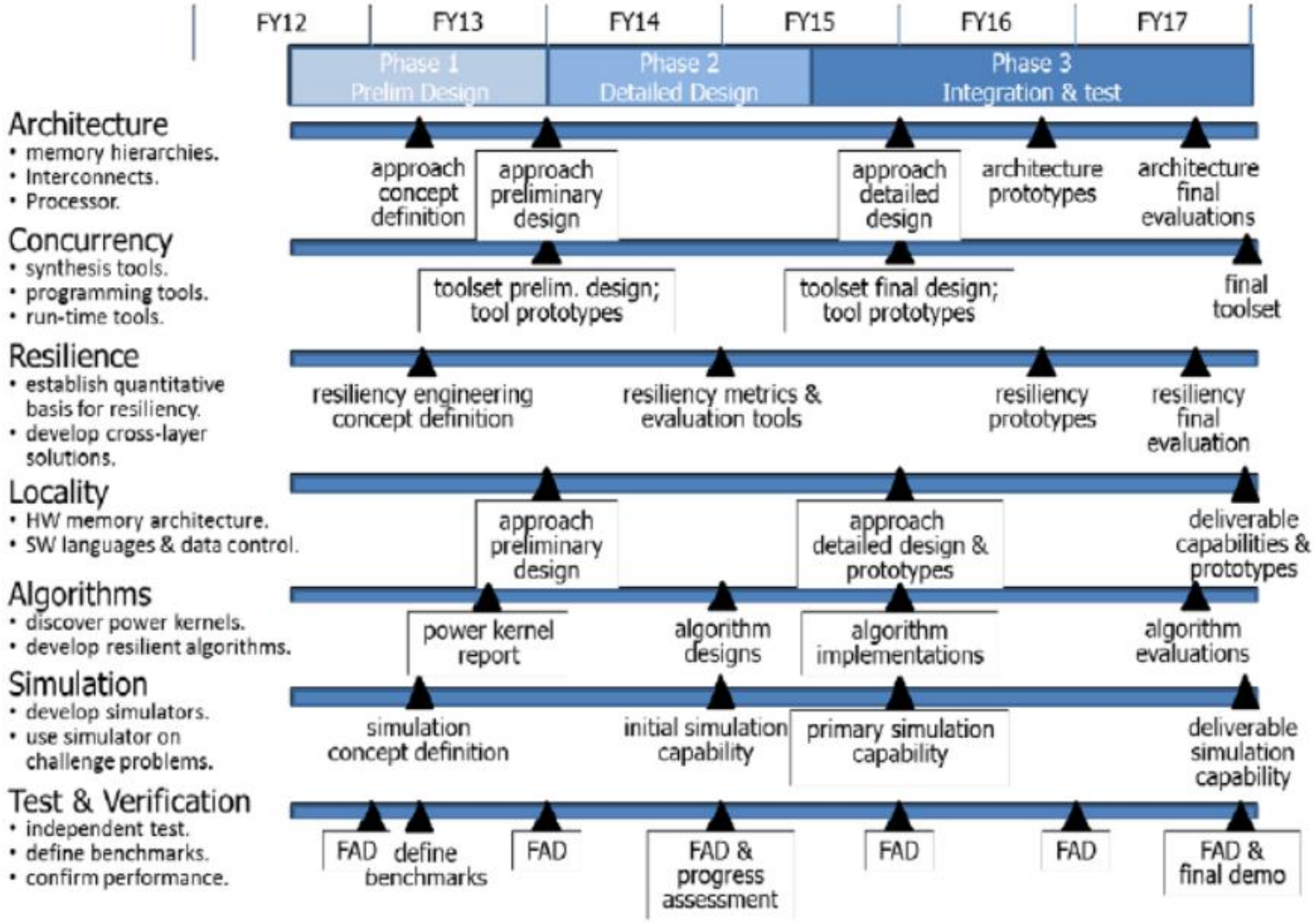


Impacts of Project Denver

- **Heterogeneous compute**
  - share mixed work efficiently between CPU/GPU
  - unified memory subsystem
- **Decreased latency**
  - no PCIe required for synchronization + command queuing
- **Power efficiency**
  - ARMv8 ISA
  - on-chip buses & interconnects
  - “4+1” power scaling
- **Natively boot & run operating systems**
- **Direct connectivity with peripherals**
- **Maximize multi-GPU**

Project Osprey

- DARPA PERFECT – 75 GFLOPS/watt



Summary

- System-wide integration of GPUDirect RDMA
- Increase GPU:CPU ratio for better GFLOPS/watt
- Utilize PCIe switches instead of dual-socket CPU/IOH

- Exascale compute – what's good for HPC is good for embedded/mobile
- Denver & Osprey – what's good for embedded/mobile is good for HPC

questions?

booth 201