OFVWG: GPUDirect and PeerDirect

Connecting co-processors to the fabric

Shachar Raindel and Davide Rossetti

Outline

- Revisit GPUDirect and PeerDirect family of technologies
- Next level of peripheral integration – GPUDirect Async and PeerDirect Sync
- Early benchmark results

GPUDirect and PeerDirect family

- GPUDirect Shared GPU-Sysmem for inter-node copy optimization
- GPUDirect P2P for intra-node, between GPUs in the node
- GPUDirect RDMA[1] for inter-node copy optimization
  - Based on PeerDirect technology from Mellanox
  - Available in Mellanox OFED
  - Submitted to upstream for review[2]

[1] http://docs.nvidia.com/cuda/gpudirect-rdma

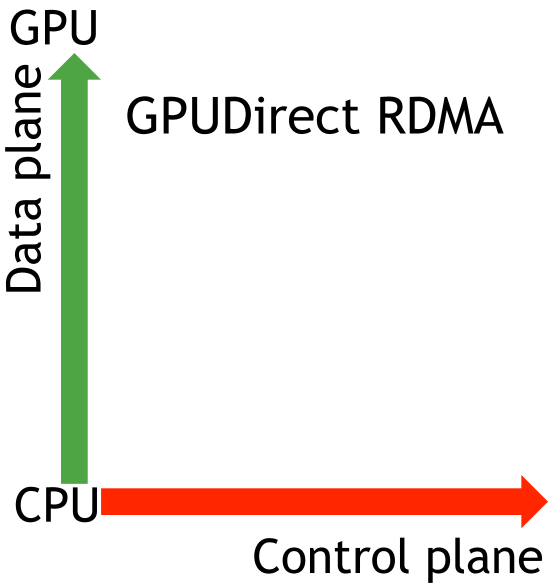
[2] http://comments.gmane.org/gmane.linux.drivers.rdma/22093

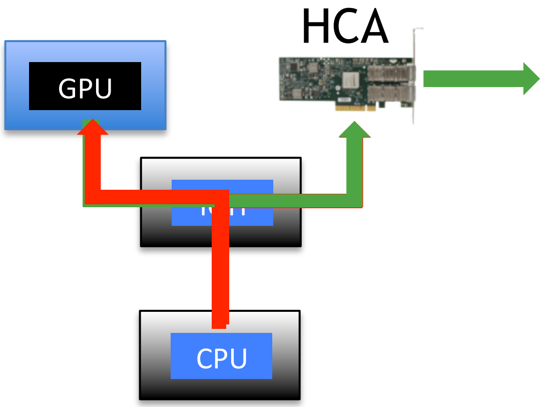
GPUDirect RDMA capabilities & limitations

- GPUDirect RDMA
  - direct HCA access to GPU memory

- CPU still driving computing + communication
  - Fast CPU needed
  - Implications: power, latency, TCO
  - Risks: limited scaling …

Moving data around with GPUDirect



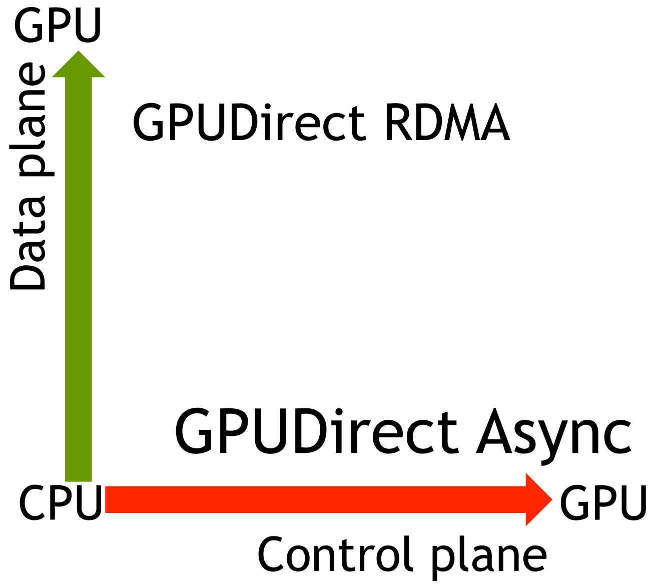


HCA directly accesses GPU memory

CPU synchronizes with GPU tasks

CPU prepares and queues communication tasks on HCA

Accelerating the control plane



GPU

Data plane

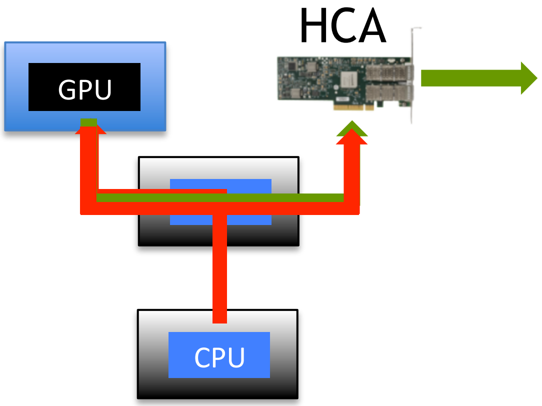
GPUDirect RDMA

GPUDirect Async

CPU

GPU

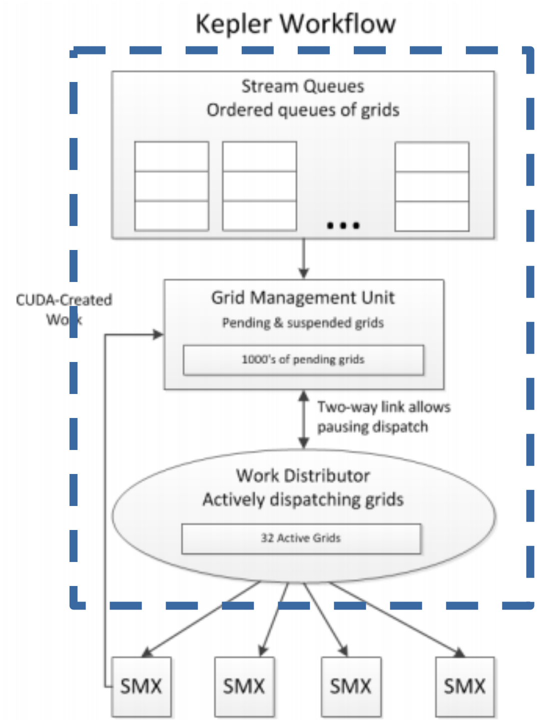
Control plane



CPU prepares and queues
compute and communication tasks on GPU
GPU triggers communication on HCA
HCA directly accesses GPU memory

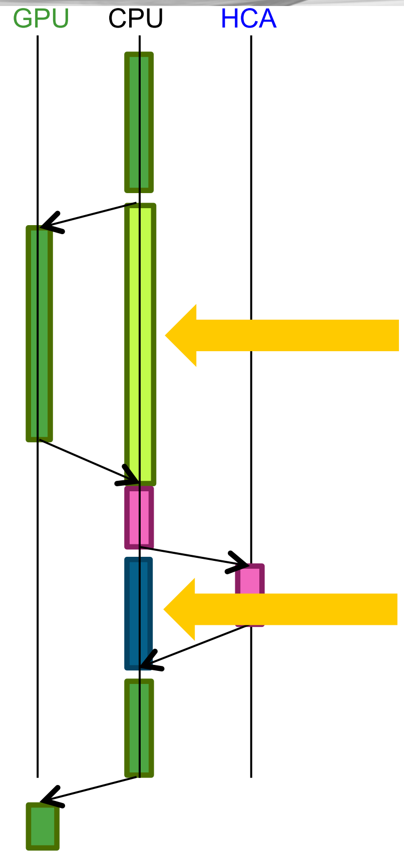
CPU off the critical path

- CPU prepares work plan
  - hardly parallelizable, branch intensive
- GPU orchestrates flow
  - Runs on optimized front-end unit
  - Same one scheduling GPU work
  - Now also scheduling network communications



Kernel+Send
Normal flow

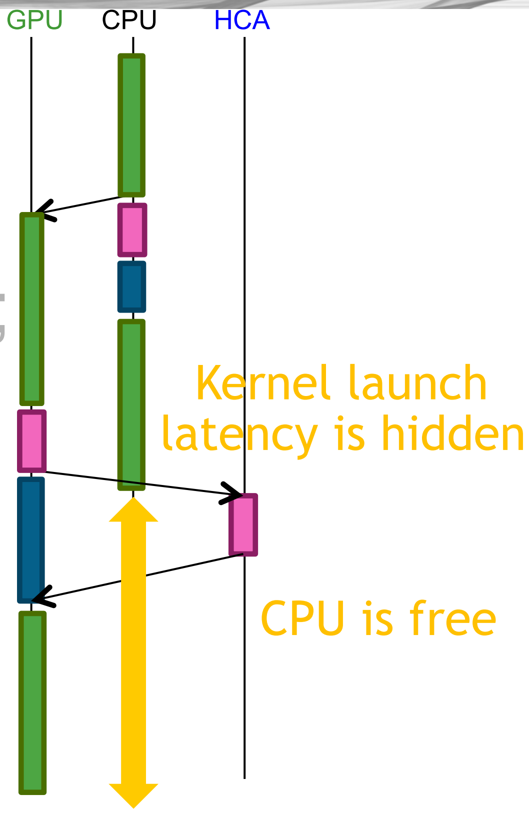
```
a_kernel<<<…,stream>>>(buf);
cudaStreamSynchronize(stream);
ibv_post_send(buf);
while (!done) ibv_poll_cq(txcq);
b_kernel<<<…,stream>>>(buf);
```



100% CPU utilization
Limited scaling!

Kernel+Send GPUDirect Async

```
a_kernel<<<…,stream>>>(buf);
gds_stream_queue_send(stream,qp,buf);
gds_stream_wait_cq(stream,txcq);
b_kernel<<…,stream>>(buf);
```



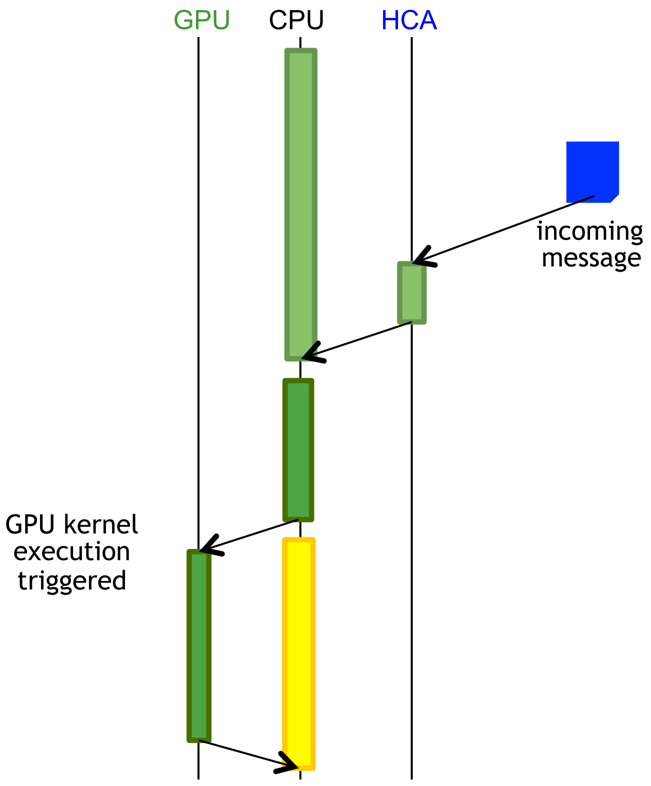
Kernel launch latency is hidden

CPU is free

No CPU in critical path!
Improve Scaling!

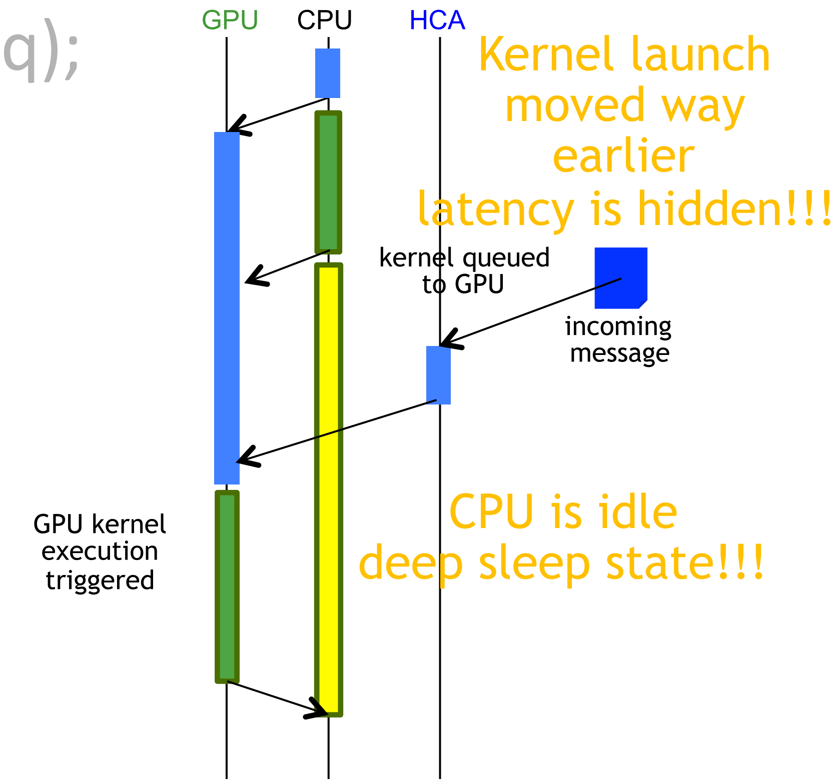
Receive+Kernel Normal flow

```
while (!done) ibv_poll_cq();

a_kernel<<<…,stream>>>(buf);

cuStreamSynchronize(stream);
```



Receive+Kernel GPUDirect Async

```
gds_stream_wait_cq(stream,rx_cq);
a_kernel<<<…,stream>>>(buf);
cuStreamSynchronize(stream);
```



Kernel launch moved way earlier latency is hidden!!!

CPU is idle deep sleep state!!!

Use case scenarios

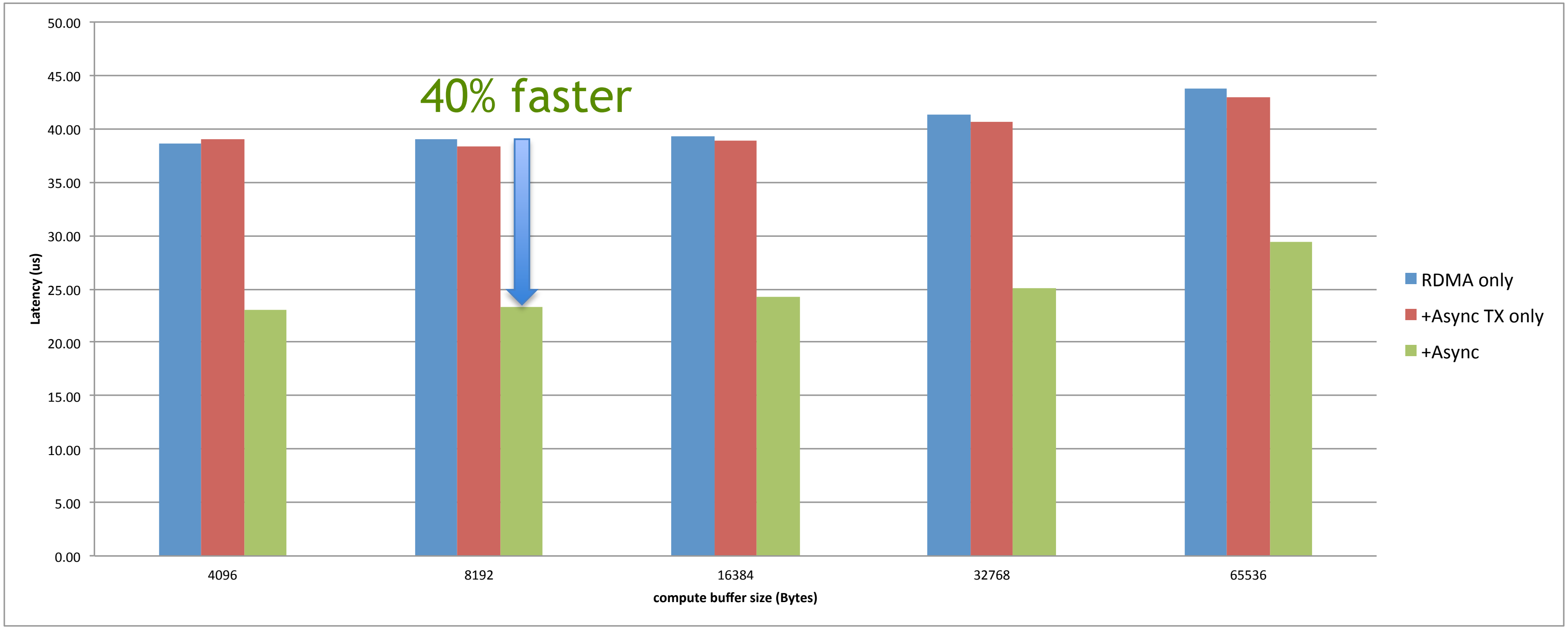
Performance mode (~ Top500)

- enable batching
- increase performance
- CPU available, additional GFlops

Economy mode (~ Green500)

- enable GPU IRQ waiting mode
- free more CPU cycles
- Optionally slimmer CPU

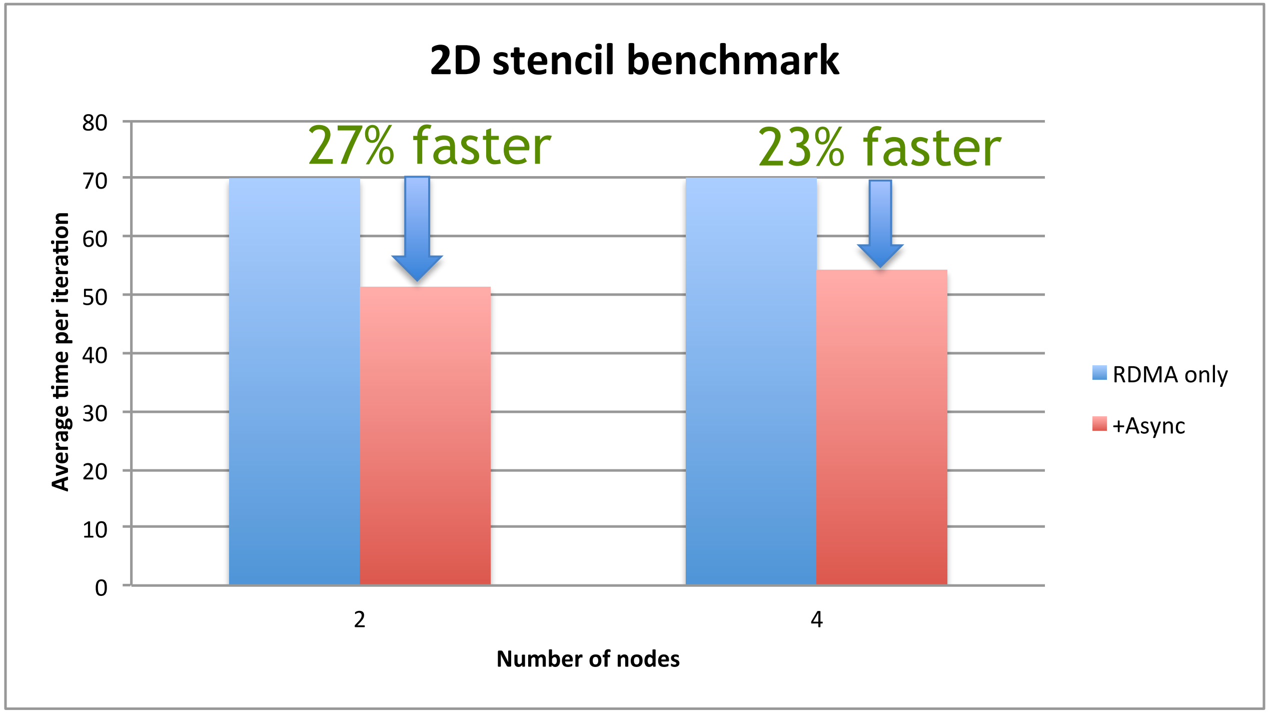
Performance Mode



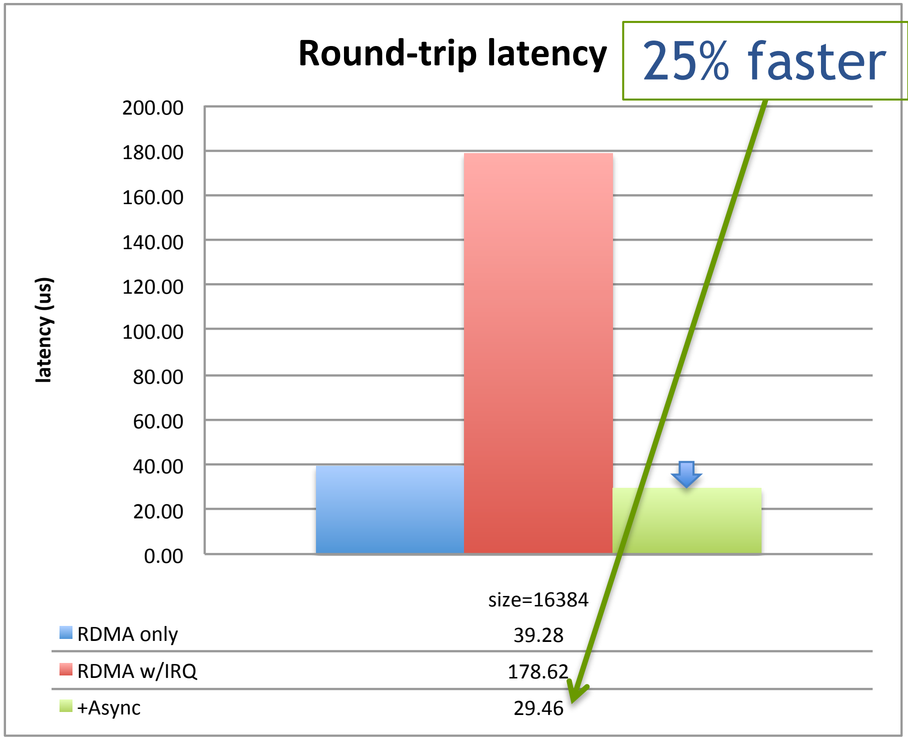
[*] modified ud_pingpong test: recv+GPU kernel+send on each side.
2 nodes: Ivy Bridge Xeon + K40 + Connect-IB + MLNX switch, 10000 iterations, message size: 128B, batch size: 20

2D stencil benchmark

- weak scaling
- 256^2 local lattice
- 2x1, 2x2 node grids
- 1 GPU per node

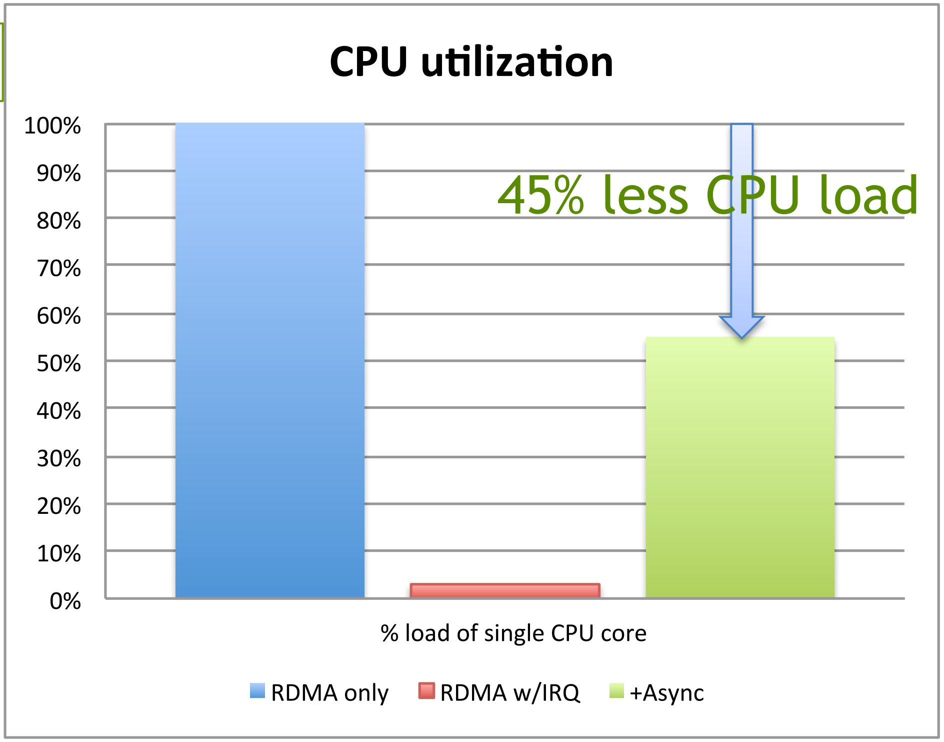


Economy Mode



25% faster

	size=16384
🟦 RDMA only	39.28
🟥 RDMA w/IRQ	178.62
🟩 +Async	29.46



45% less CPU load

[*] modified ud_pingpong test, HW same as in previous slide

Summary

- Meet Async, next generation of GPUDirect
- GPU orchestrates network operations
- CPU off the critical path
- **40% faster**, **45% less** CPU load

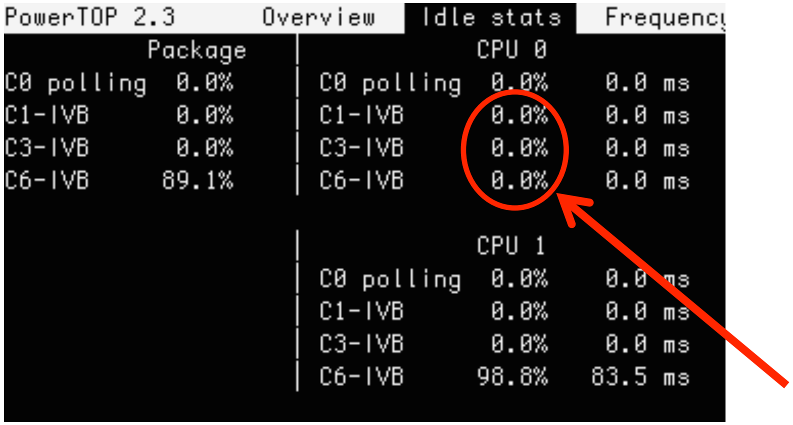
Thank You

OPENFABRICS ALLIANCE

10TH ANNUAL INTERNATIONAL DEVELOPER WORKSHOP

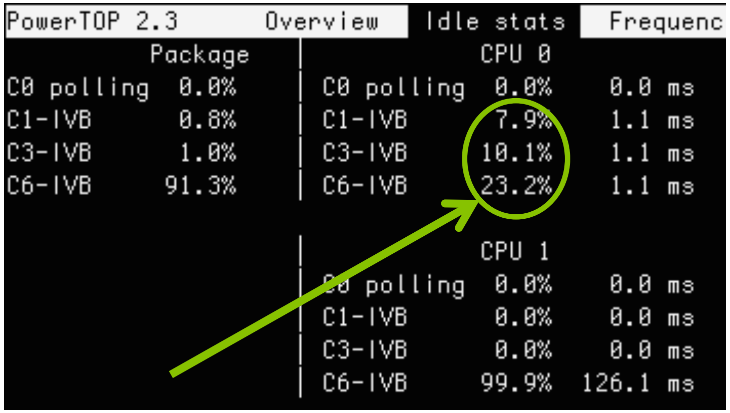
Performance vs Economy

Performance mode



```
PowerTOP 2.3            Overview   Idle stats   Frequency
            Package    |            CPU 0
C0 polling   0.0%      | C0 polling   0.0%      0.0 ms
C1-IVB       0.0%      | C1-IVB       0.0%      0.0 ms
C3-IVB       0.0%      | C3-IVB       0.0%      0.0 ms
C6-IVB      89.1%      | C6-IVB       0.0%      0.0 ms

                       |            CPU 1
                       | C0 polling   0.0%      0.0 ms
                       | C1-IVB       0.0%      0.0 ms
                       | C3-IVB       0.0%      0.0 ms
                       | C6-IVB      98.8%     83.5 ms
```

Economy mode



```
PowerTOP 2.3            Overview   Idle stats   Frequenc
            Package    |            CPU 0
C0 polling   0.0%      | C0 polling   0.0%      0.0 ms
C1-IVB       0.8%      | C1-IVB       7.9%      1.1 ms
C3-IVB       1.0%      | C3-IVB      10.1%      1.1 ms
C6-IVB      91.3%      | C6-IVB      23.2%      1.1 ms

                       |            CPU 1
                       | C0 polling   0.0%      0.0 ms
                       | C1-IVB       0.0%      0.0 ms
                       | C3-IVB       0.0%      0.0 ms
                       | C6-IVB      99.9%    126.1 ms
```

[*] modified ud_pingpong test, HW same as in previous slide, NUMA binding to socket0/core0, SBIOS power-saving profile