Next Generation Interconnection for Accelerated Computing

Taisuke Boku

Deputy Director, Center for Computational Sciences

University of Tsukuba

FPGA as “glue” for Acceleration and Communication

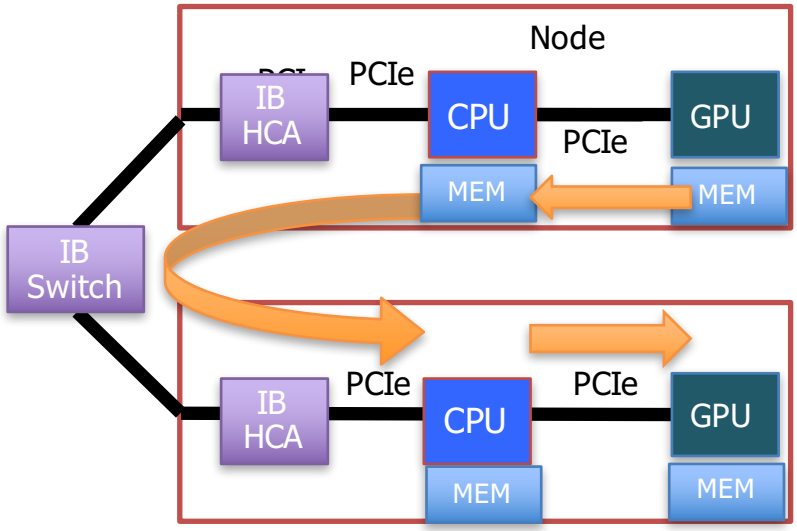

Issues on accelerated (GPU) computing

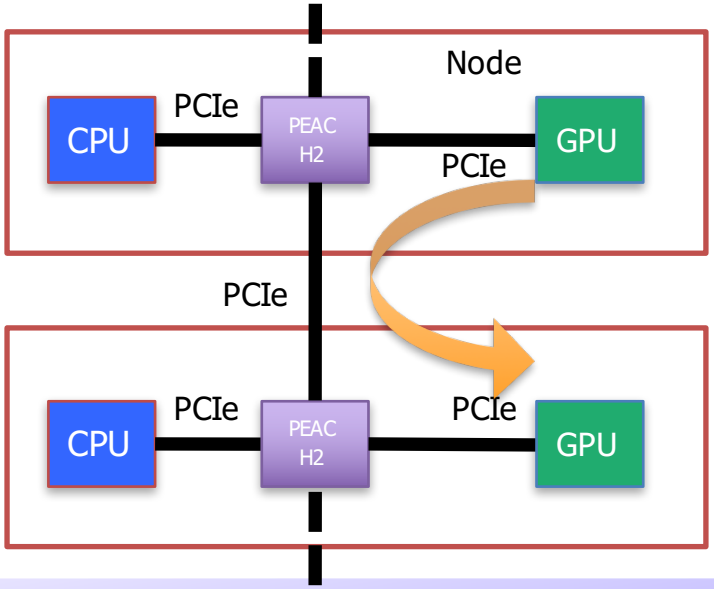
- Trading-off: Power vs Dynamism (Flexibility)
  - fine grain individual cores consume much power, then ultra-wide SIMD feature to exploit maximum Flops is needed
- Interconnection is a serious issue
  - current accelerators are hosted by general CPU, and the system is not stand-alone
  - current accelerators are connected by some interface bus with CPU then interconnection
  - current accelerators are connected through network interface attached to the host CPU
- Latency is essential (not just bandwidth)
  - with the problem of memory capacity, "strong scaling" is required to solve the problems
  - "weak scaling" doesn't work in some case because of time to solution limit
  - in many algorithms, reduction of just a scalar value over millions of node is required

Accelerators must be tightly coupled with each other, meaning "**They should be equipped with communication facility of their own**"

TCA (Tightly Coupled Accelerators) Architecture

- **True GPU-direct**
  - current GPU clusters require 3-hop communication (3-5 times memory copy)
  - For strong scaling, inter-GPU direct communication protocol is needed for lower latency and higher throughput
- ***PEACH2*** : a prototype implementation of TCA on PCIe
  - x4 lanes -> x8 lanes
  - hardwired on main data path and PCIe interface fabric

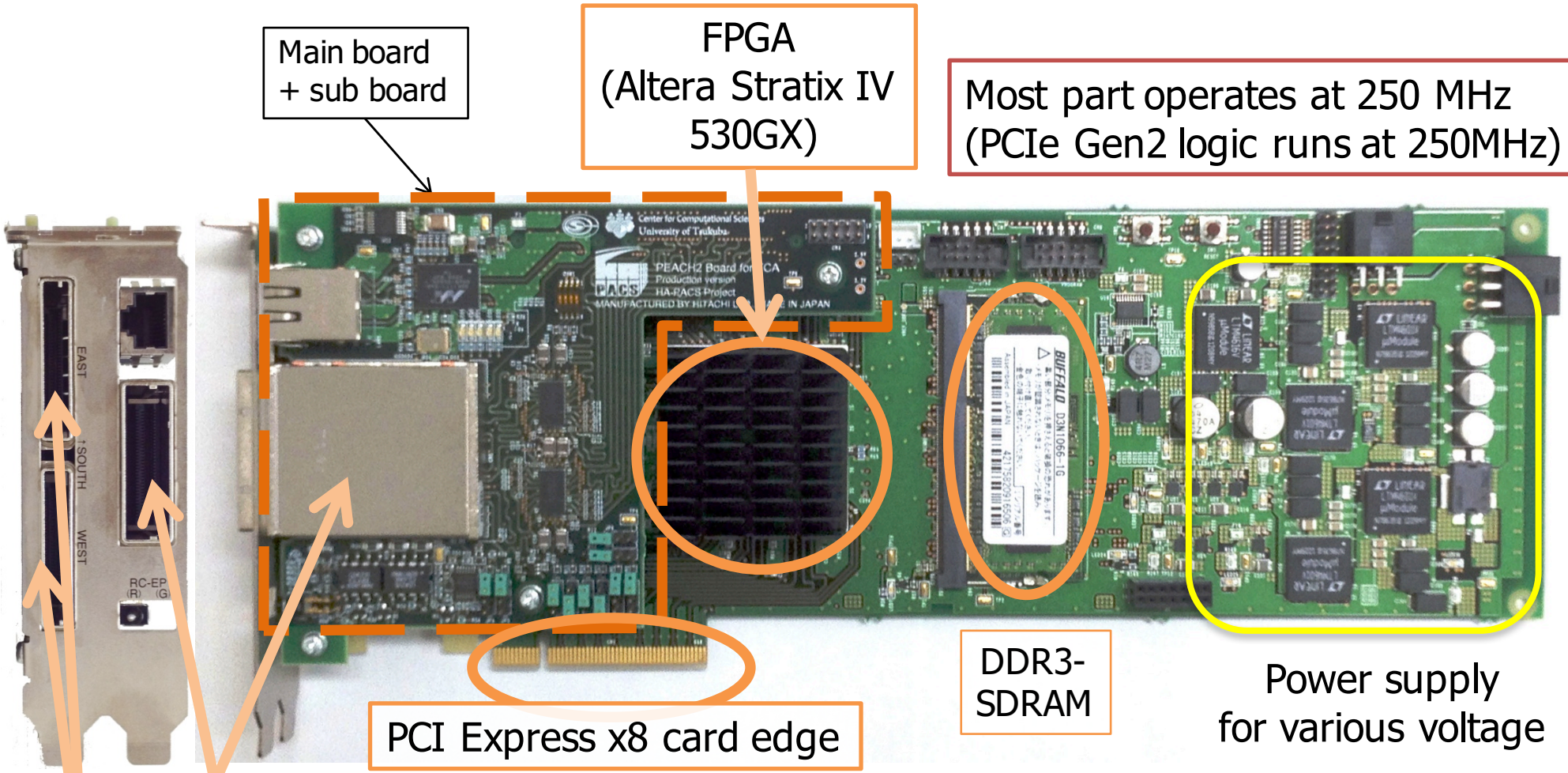




AC-CREST: Acceleration & Communication

- JST-CREST research projects
  - research area "Development of System Software Technologies for post-Peta Scale High Performance Computing" (RS: Dr. M. Sato, RIKEN)
  - Research theme: "Research and Development on Unified Environment of Accelerated Computing and Interconnection for Post-Petascale Era" (PI: T. Boku, U. Tsukuba), Oct. 2012 - Mar. 2018, US$3M+ (total)
- My team
  - Basic system software and performance evaluation
    (by T. Boku, U. Tsukuba)
  - TCA hardware enhancement
    (by H. Amano, Keio U.)
  - Language: XcalableMP/TCA with OpenACC – XcalbleACC
    (by H. Murai, AICS RIKEN)
  - Application software
    (by M. Umemura, U. Tsukuba)

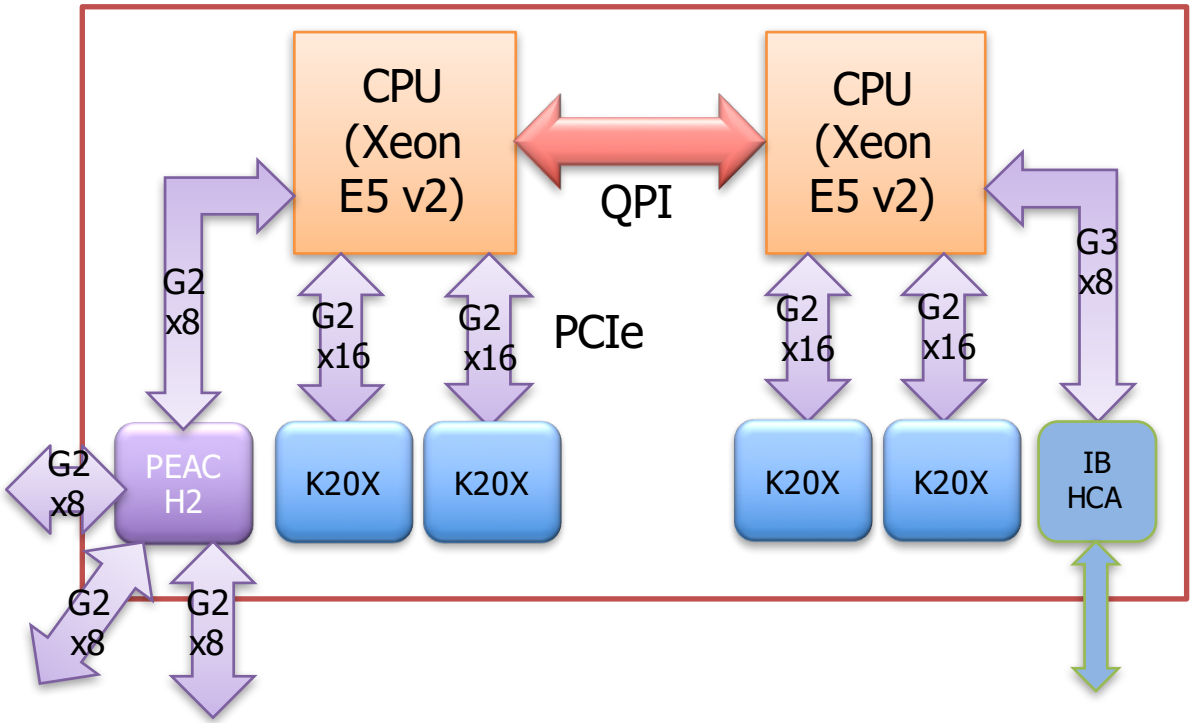

PEACH2 board



HA-PACS/TCA test-bed node structure

- CPU can uniformly access to GPUs.
- PEACH2 can access every GPUs
  - Kepler architecture + CUDA 5.0 "GPUDirect Support for RDMA"
  - Performance over QPI is quite bad.
    => support only for two GPUs on the same socket
- Connect among 3 nodes

- This configuration is similar to HA-PACS base cluster except PEACH2.
  - All the PCIe lanes (80 lanes) embedded in CPUs are used.



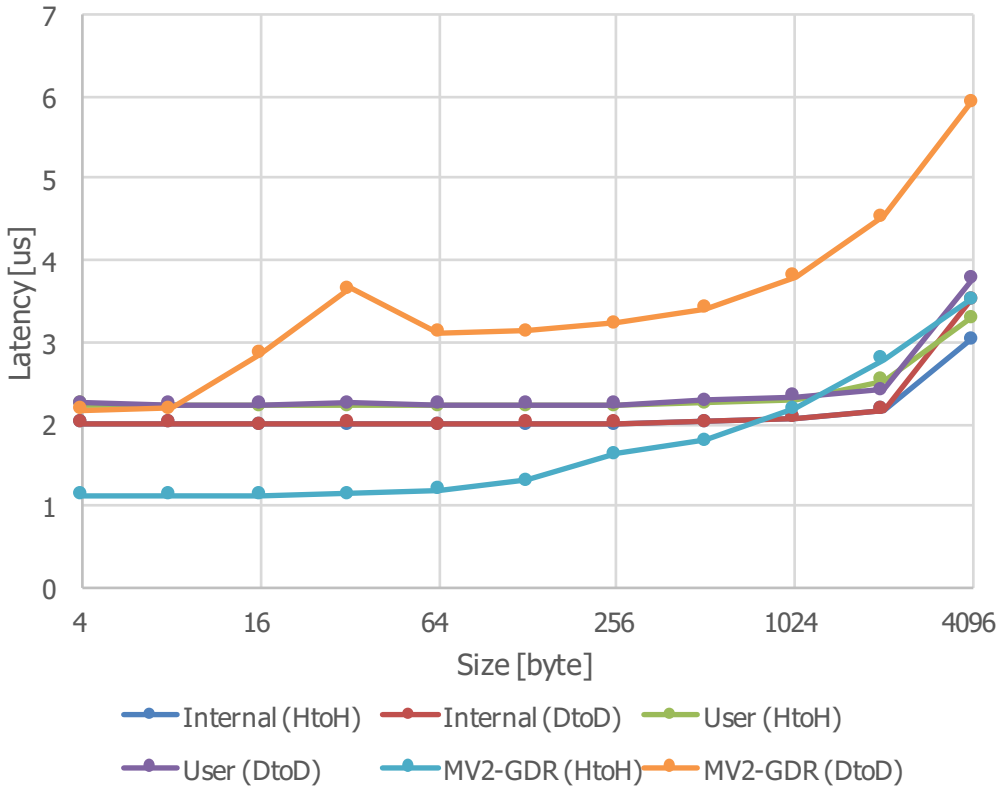
HA-PACS Base Cluster + TCA (TCA part starts operation on Nov. 1st 2013)

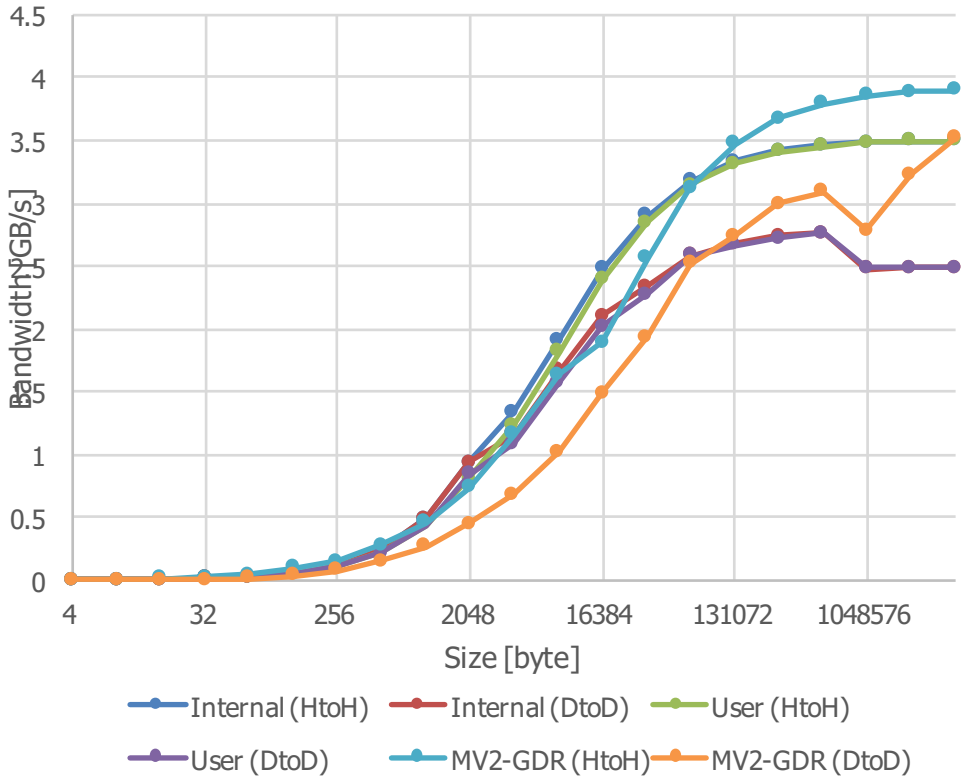
- HA-PACS Base Cluster = 2.99 TFlops x 268 node = 802 TFlops
- HA-PACS/TCA = 5.69 TFlops x 64 node = 364 TFlops
- TOTAL: 1.166 PFlops
- TCA part (individually) ranked as #3 in Green500, Nov. 2013

Ping-pong performance on TCA/PEACH2 (as in 2013)

Platform: HA-PACS/TCA
IB: Connect-X3 QDR (x 2rail)
GPU: K20X

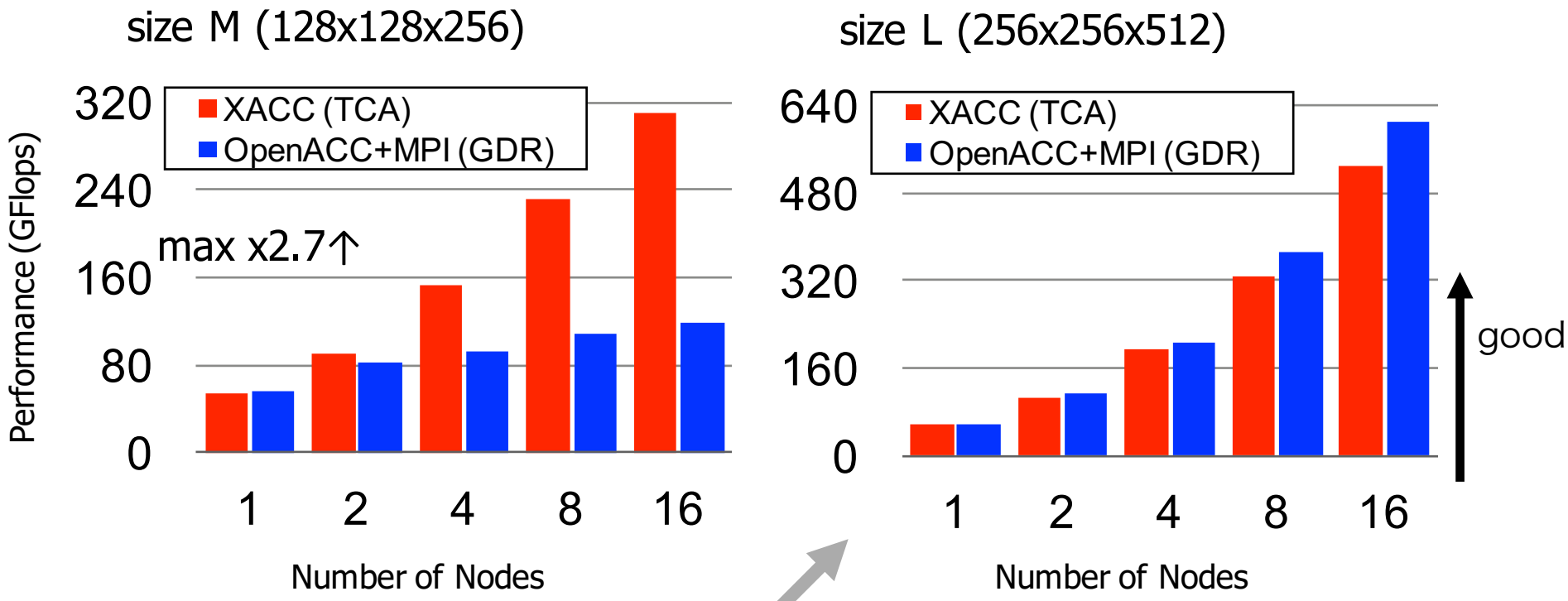
Data is at 2013, not today
Now IB latency for GPU-GPU comm. is ~2us.





Performance on Himeno benchmark

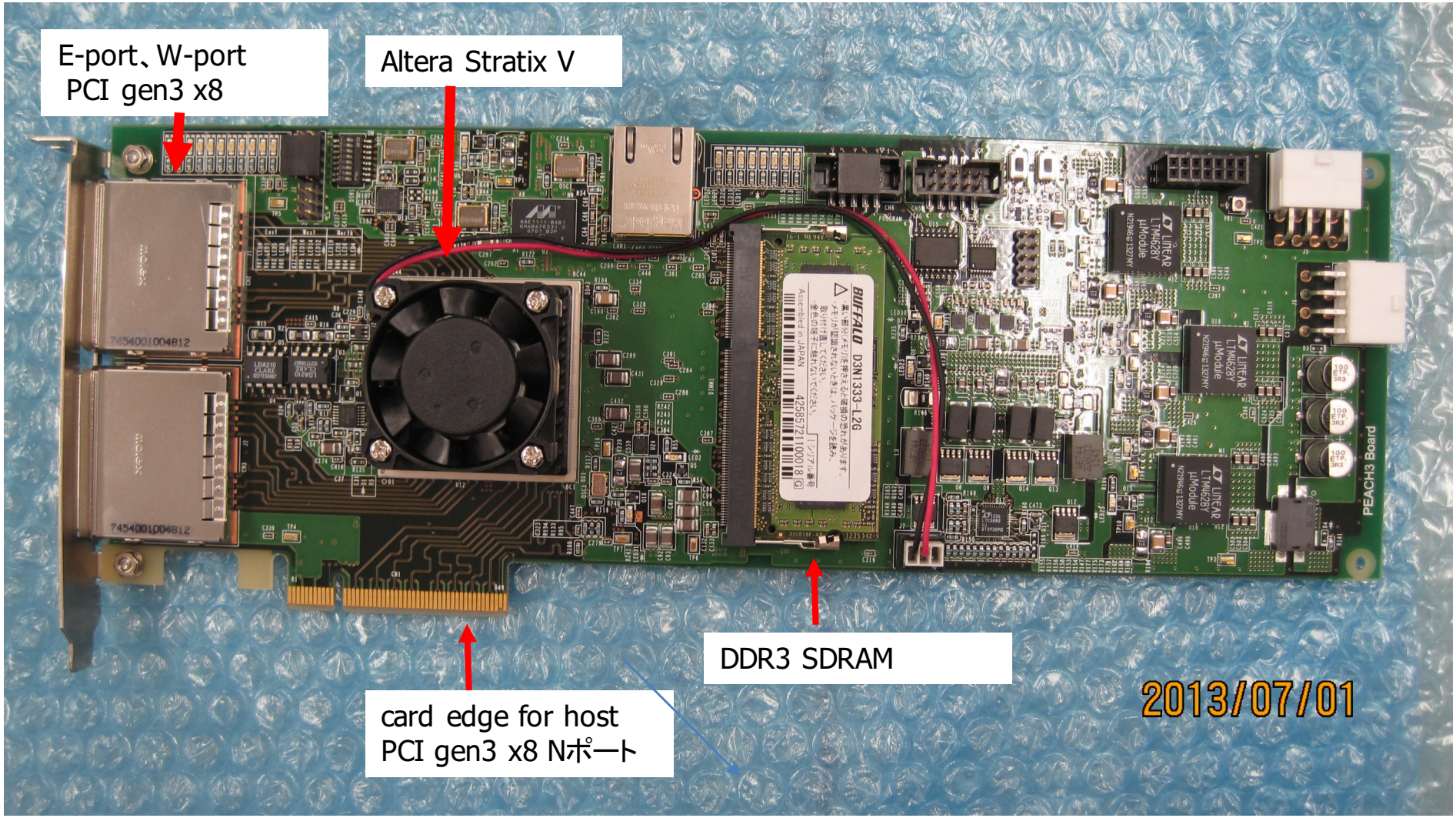
2-D stencil computing for fluid dynamics



For size L, size of sleeve area is approximately 520KB, so TCA's advantage is small compared to MVAPICH2-GDR.

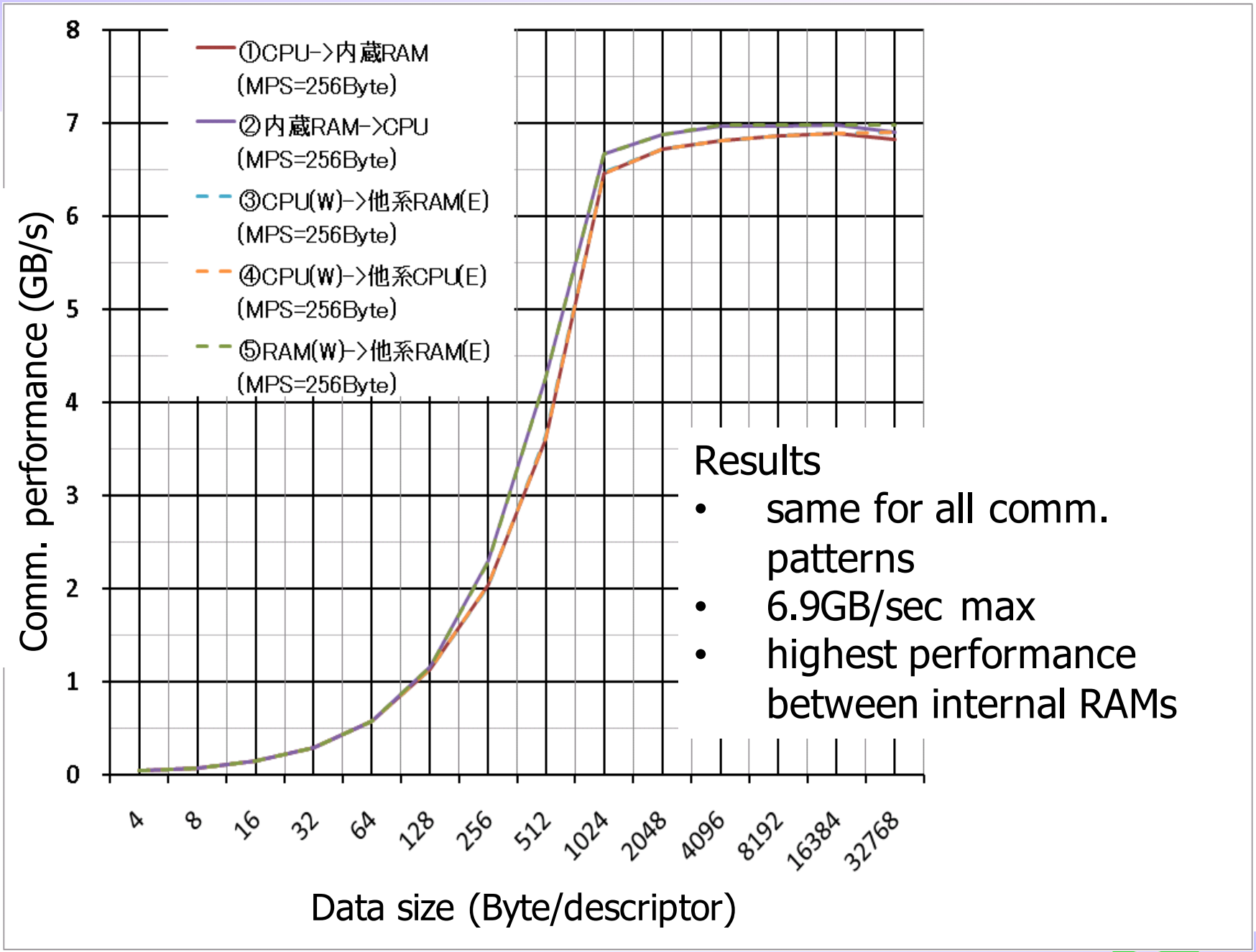
Additionally, TCA requires a barrier synch. after DMA transfer to cause additional overhead

Bandwidth enhancement: PEACH2 -> PEACH3



Bandwidth enhancement: PEACH2 -> PEACH3

	PEACH2	PEACH3
FPGA family	Stratic IV GX	Stratix V GX
FPGA type	EP4SGX530NF45C2	ES5GXA7N3F45C2
Process Technology	40nm	28nm
Max LE counts	531K	622K
PCI port	PCIe Gen2 x 8	PCIe Gen3 x 8
Max data transfer rate per port	4Gbyte/sec	7.9Gbyte/sec
Max frequency	250MHz	250MHz
Internal bus width	128bit	256bit
LE consuming ratio	22%	38%
DRAM on-board	DDR 512Mbyte	DDR 512Mbyte

Results

- same for all comm. patterns
- 6.9GB/sec max
- highest performance between internal RAMs

Accelerator in Switch


Is GPU enough for everything ?

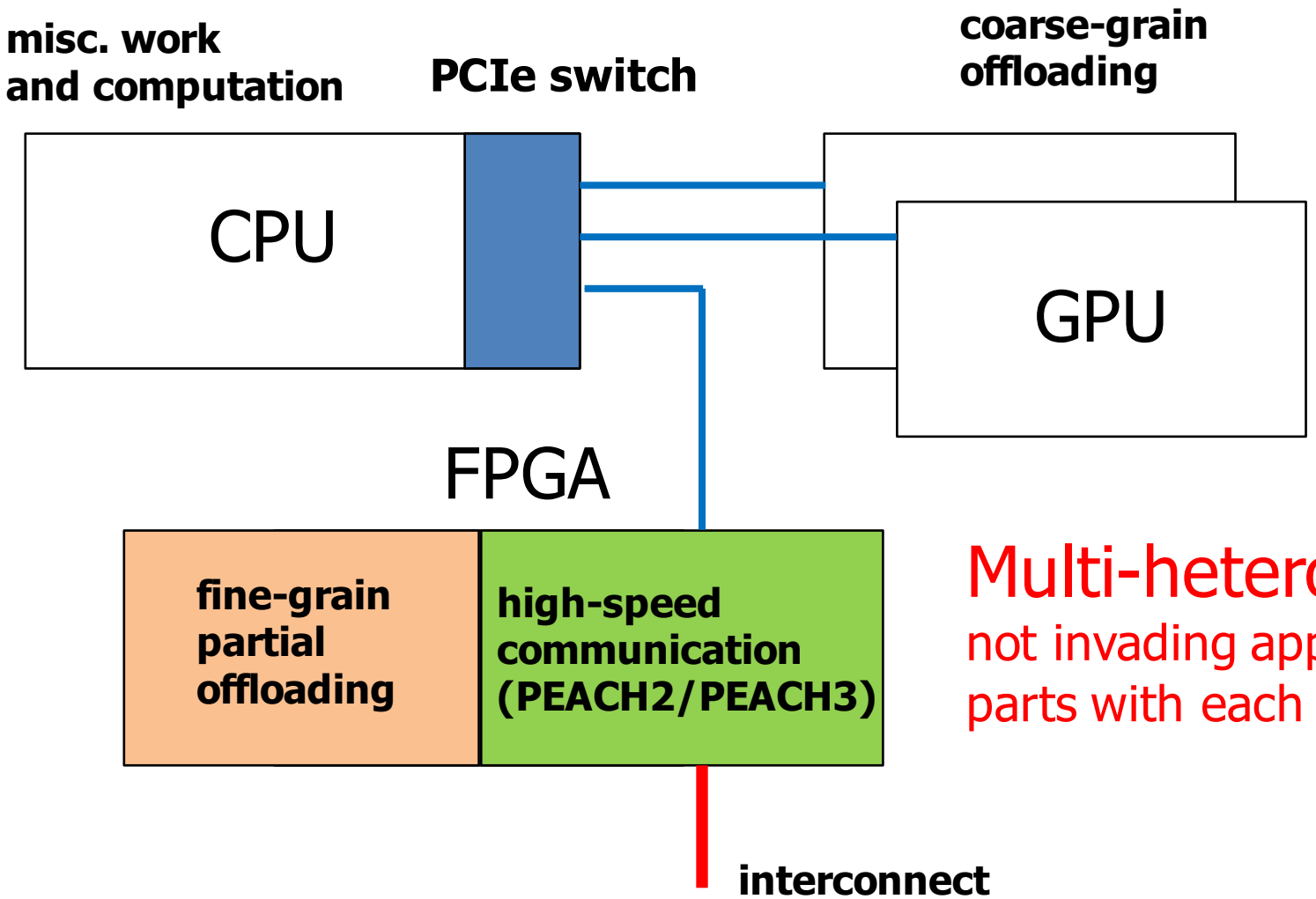
- GPU is good for:
  - coarse grained parallel applications
  - regular pattern of parallel computation without exception
  - applications relying on high memory bandwidth
- In recent HPC applications:
  - various *precision* of data is introduced (double, single, half...)
  - some computation is strongly related to communication
    ➟ fine ~ mid grained computation not suitable for GPU and too slow by CPU
  - complicated algorithm with exception


PEACH2/PEACH3 is based on FPGA

- FPGA for parallel platform for HPC
  - in general and regular computation, GPU is better
  - for something "weird/special" type of computation
  - (relatively) non bandwidth-aware computation
- PEACH solution on FPGA provides communication and computation on a chip
  - PEACH2/PEACH3 consumes less than half of LE on FPGA
  - "partial offloading" of computation in parallel processing can be implemented on rest of FPGA

➟ Accelerator in Switch (Network)


Schematic of Accelerator in Switch



Multi-hetero

not invading appropriate parts with each other

Example of Accelerator in Switch

- Astrophysics
  - Gravity calculation in domain decomposition
  - Tree search is efficient
  - LET (Locally Essential Tree) is introduced to reduce the search space in tree structure
    ➟ too complicated to handle in GPU
  - CPU is too slow
    ➟ implementing the function on FPGA and combining with PEACH2 communication part


LET (Locally Essential Tree)

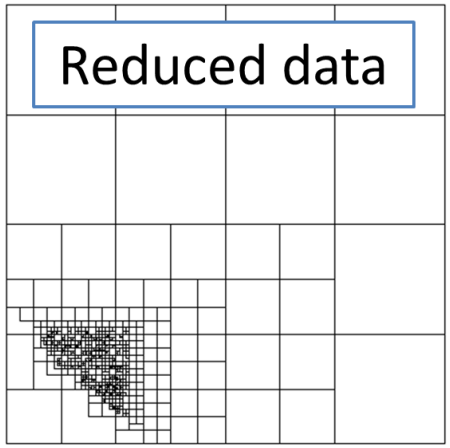
mask[id] array

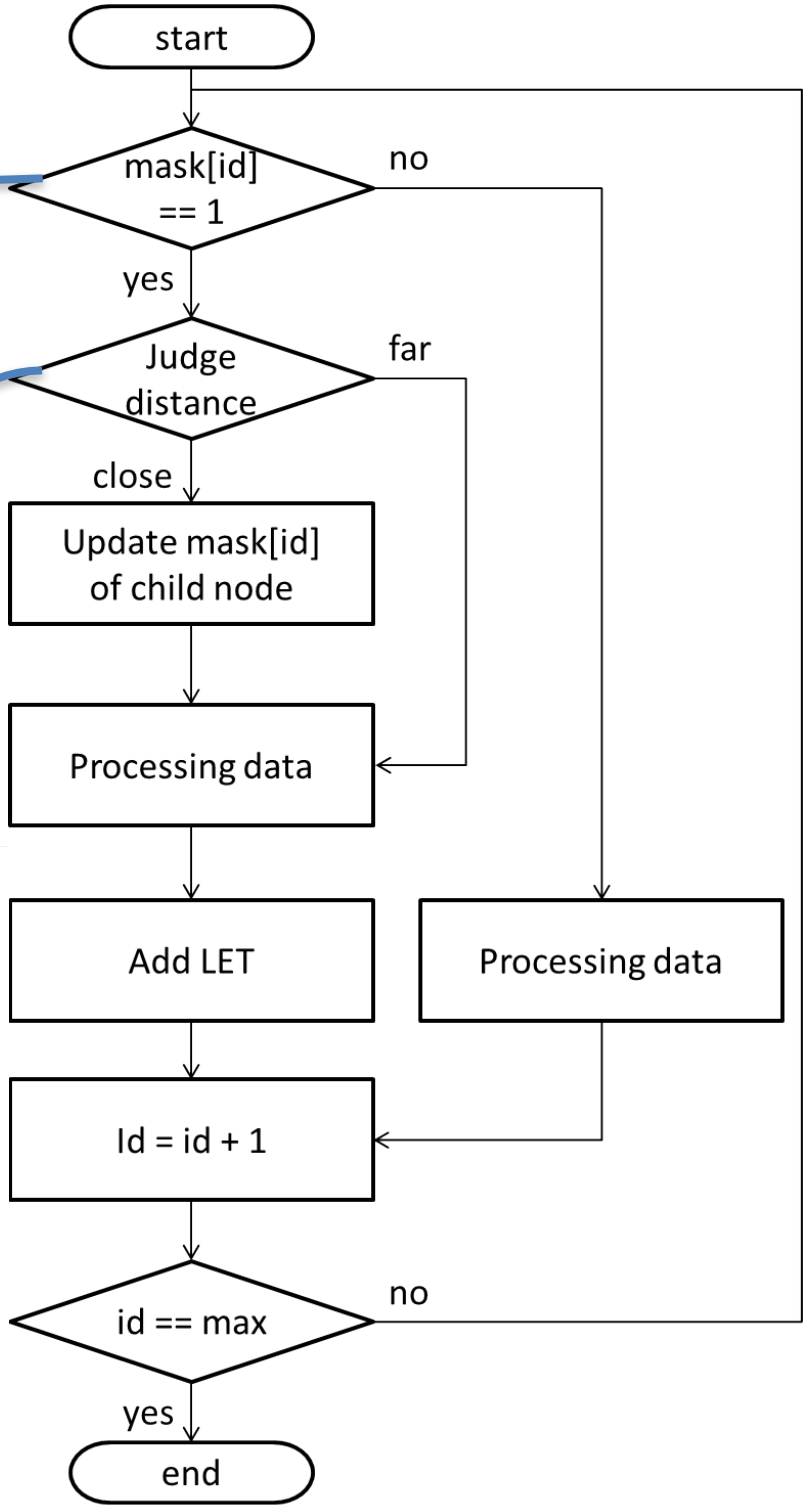
mask[id] == 0 skip

mask[id] == 1 add to LET

distance judgement

partial regional data in receiver side and each cell in sender side

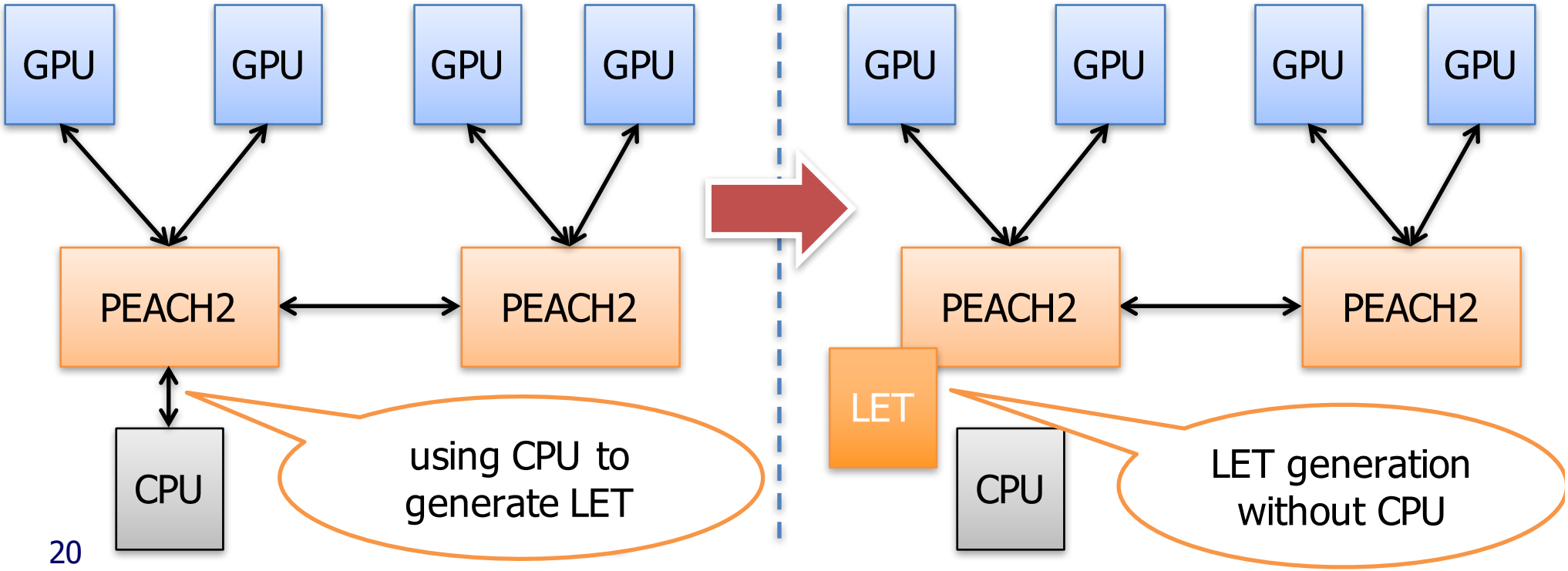




Problem of Tree-method on TCA

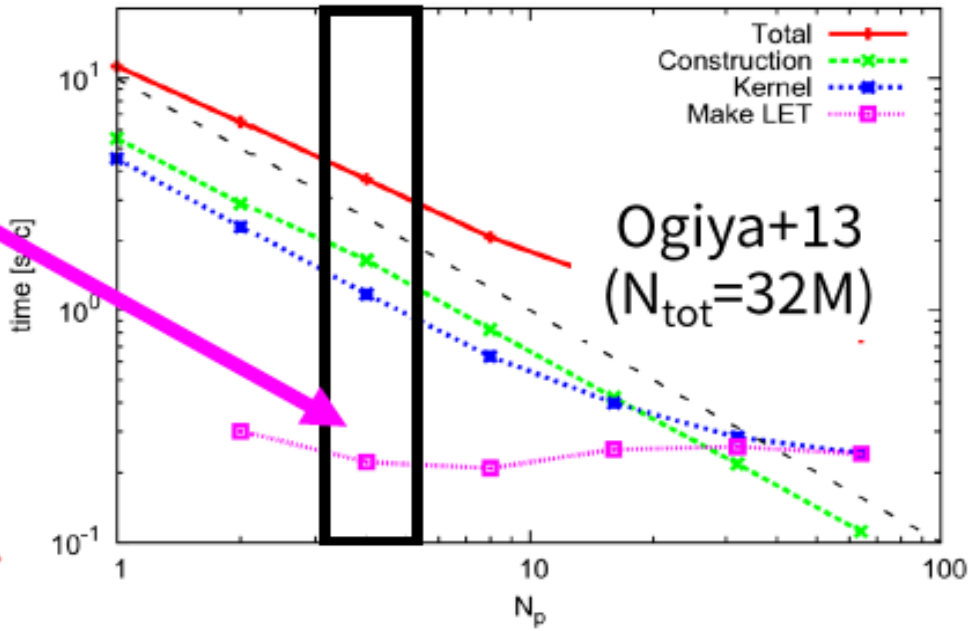
LET generation is not efficient on GPU, so it's done on CPU
→ communication with CPU is required

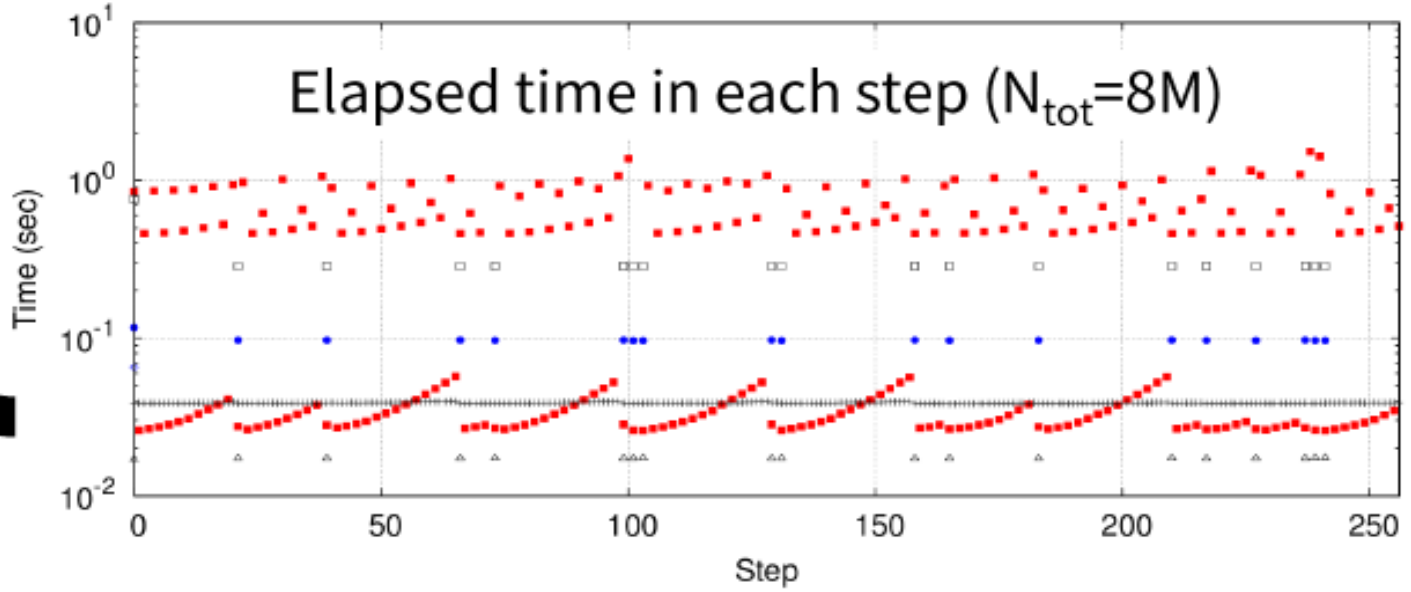
implementing LET generation on PEACH2 FPGA



Cost of using LET

- ～0.2 sec. (Ogiya+13)
- Must be smaller than ～0.04 sec. to scale.
  - At least ～5 times acceleration is required.



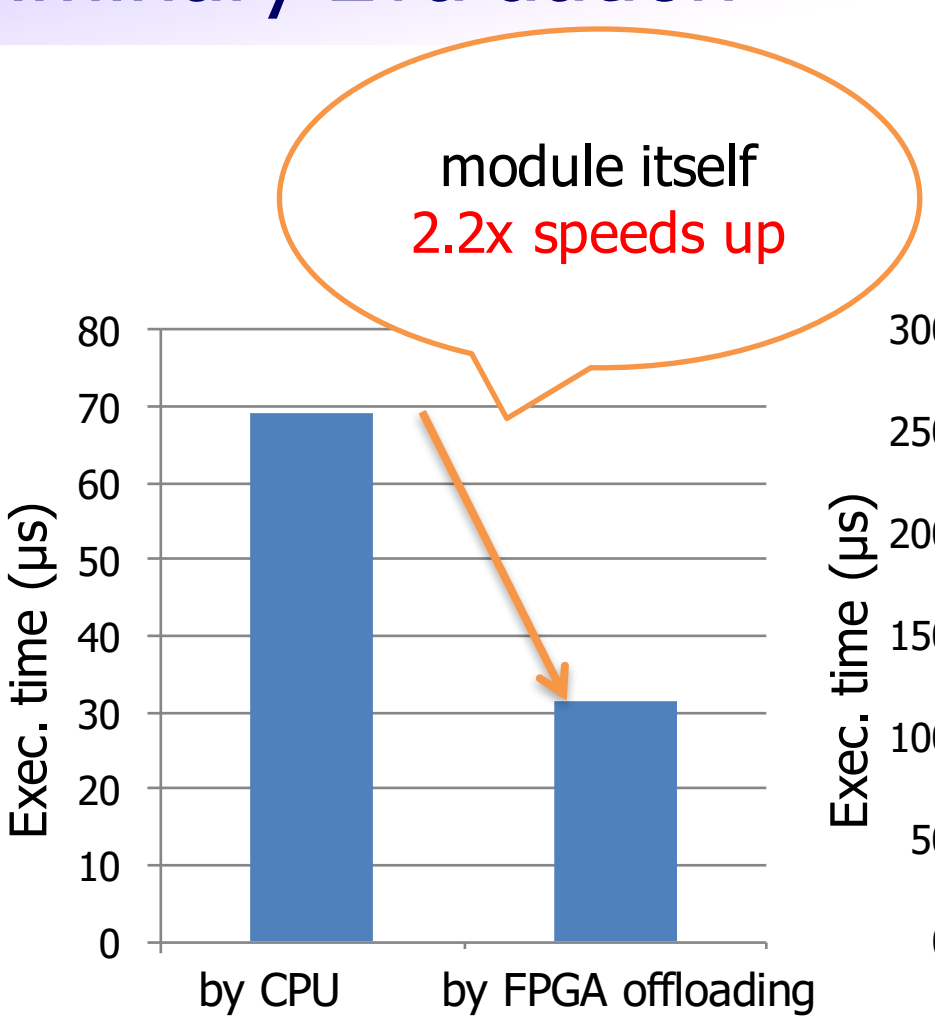


FPGA Gate Usage

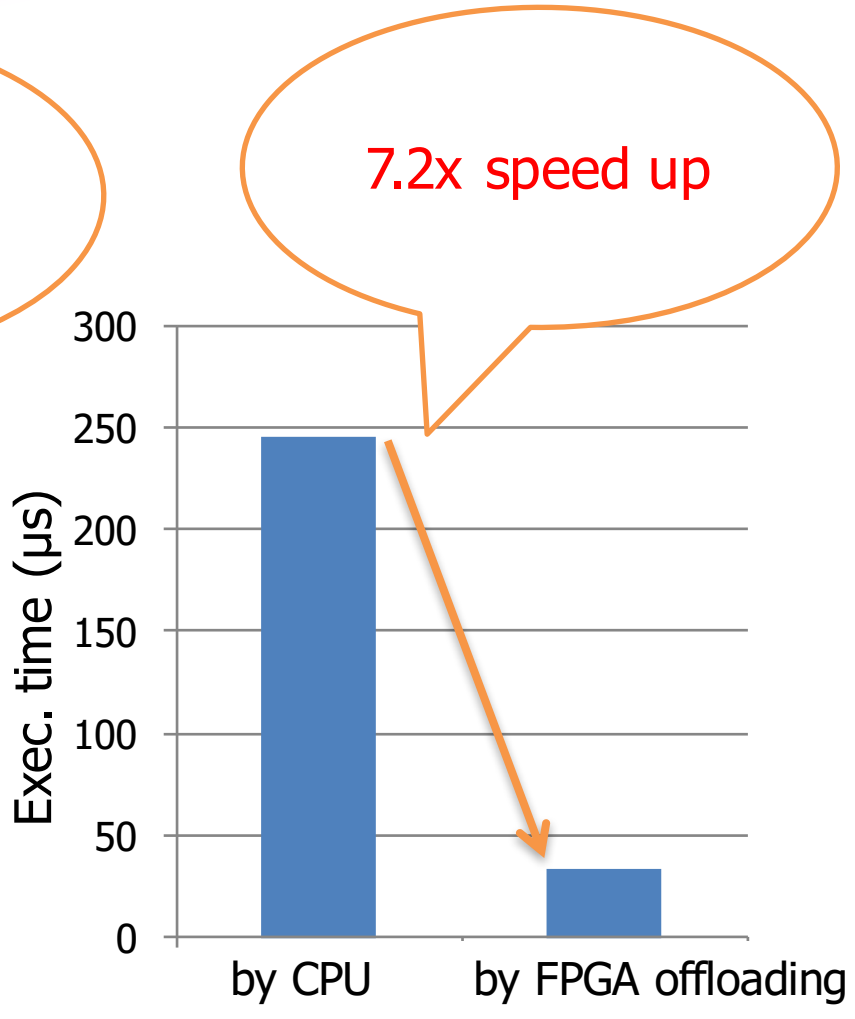
	PEACH2 Used (%)	LET generator & PEACH2 Used (%)
Logic utilization	46 %	67 %
Combinational ALUTs	65665 (28 %)	74561 / 232960 (32 %)
Dedicated logic registers	83690 (36 %)	122714 / 232960 (53 %)
Total block memory bits	2964560 (21 %)	2744448 / 13934592 (20 %)
DSP block elements	4 (<1 %)	36 / 832 (4 %)

※ PEACH2 has enough empty logic elements for LET generator.

Preliminary Evaluation



Exec. time for making LET



from LET making to GPU data transfer

Open Issues

- How to program ?
  - OpenCL may be a key for FPGA computing
  - How to combine the module on FPGA computation and other framwork (CPU and GPU)
    → calling special function to invoke FPGA computation from CPU
  - XcalableACC -> OpenACC -> OpenCL ?

- How to reconfigure FPGA ?
  - Partial reconfiguration on FPGA (we have done it)
  - How to combine multiple modules on FPGA ?
  - We need to prepare the partially-reconfigurable modules of PEACH2/3 communication part

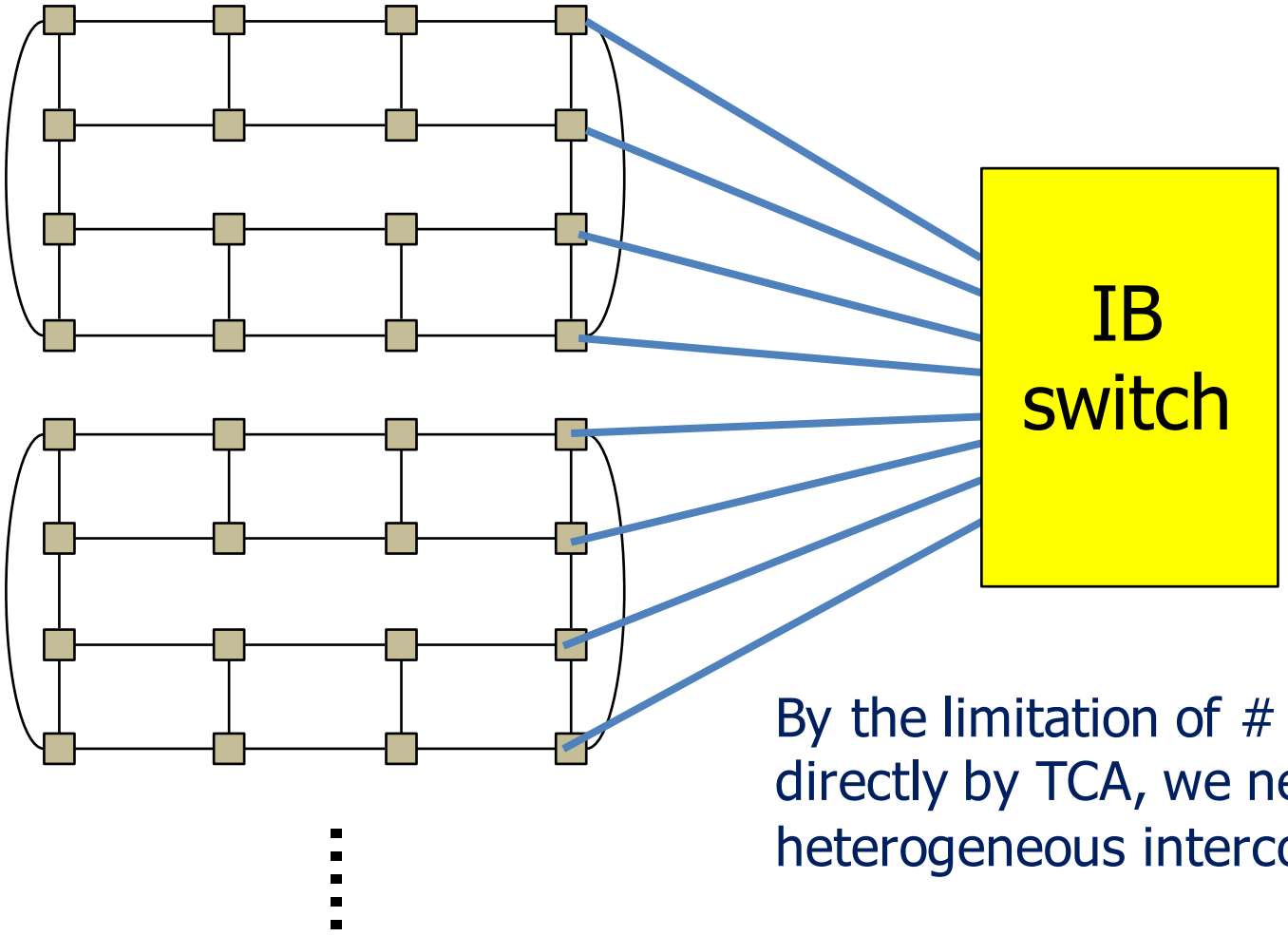

Inter-FPGA communication link

Performance growth from 2010 to 2016: PCIe on FPGA

2010	2016
Virtex 7 (XILINX) Stratix IV (Altera)	UltraScale+ (XILINX) Stratix V (Altera/Intel)
PCI Express Gen2	PCI Express Gen4 ?
8 lanes	8 lanes when PCIe Gen4 16 lanes when PCIe Gen3
4 PCI ports	6 PCI ports
256 Gb/s per FPGA	1,536 Gb/s per FPGA

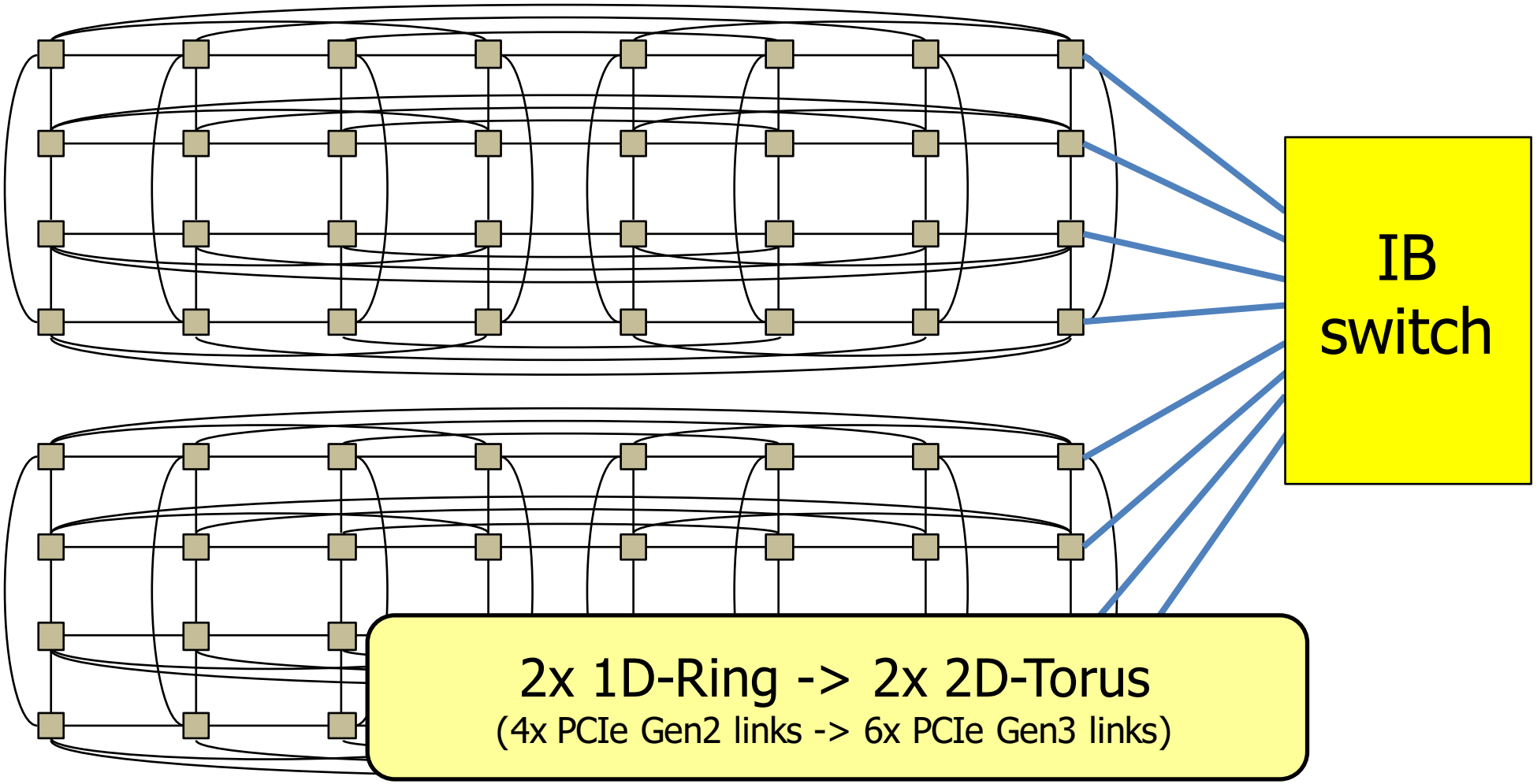
What happens by this performance improvement?

HA-PACS/TCA (connected by PCIe Gen2 8 lanes)

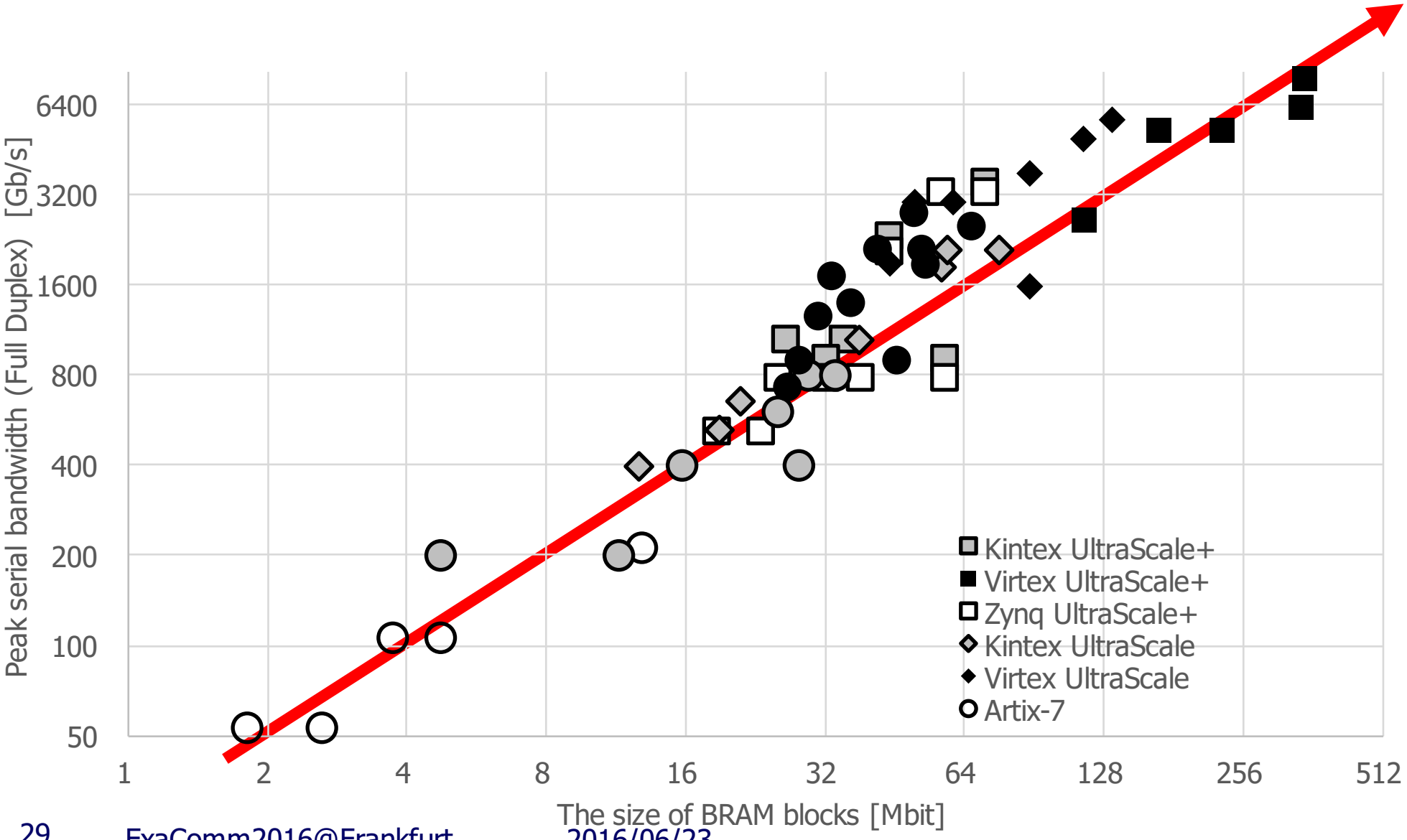


By the limitation of # of nodes connected directly by TCA, we need hierarchical and heterogeneous interconnect (with IB)

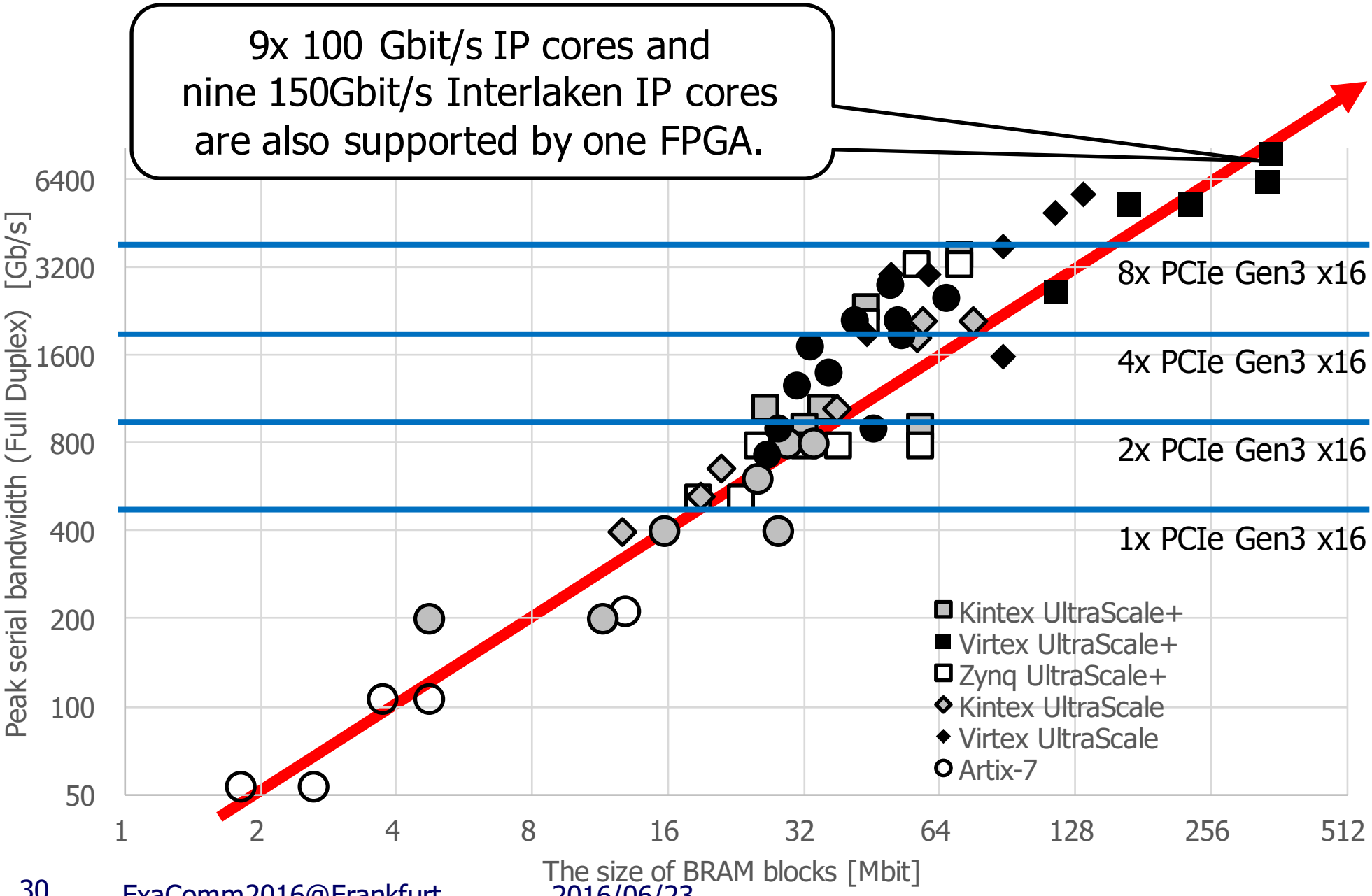
pre-PACS-X (connected by PCIe Gen3 16 lanes)



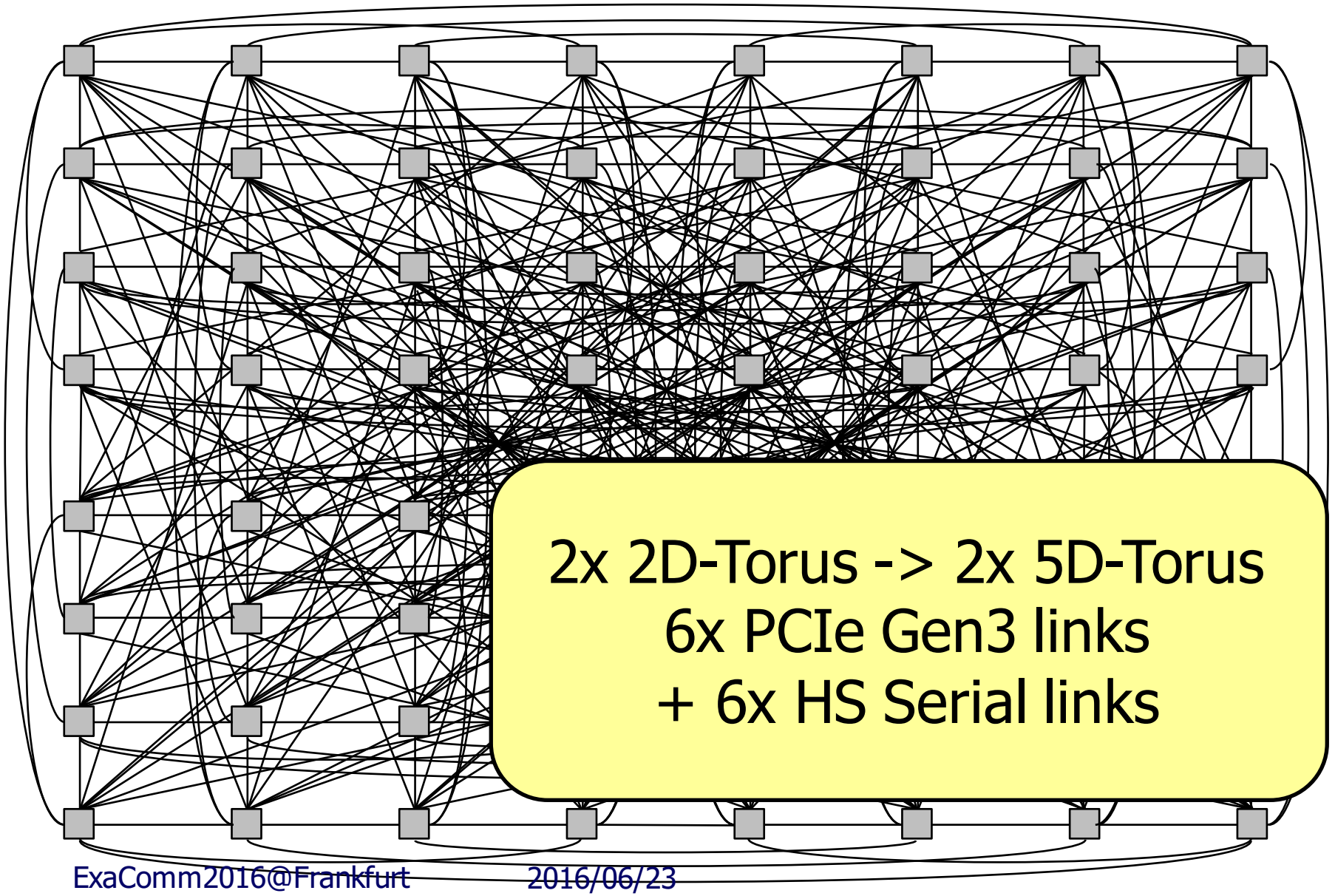
Performance growth from 2010 to 2016: Serial I/O IF on FPGA



Performance growth from 2010 to 2016: Serial I/O IF on FPGA



PACS-X (connected by PCIe & High Speed Serial IO)

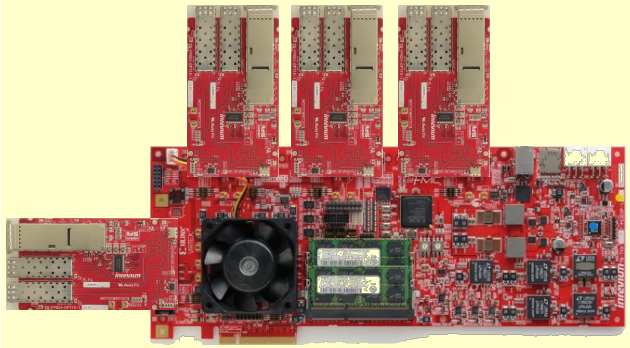


Toward Heterogeneous System w/ FPGA

XILINX Virtex 7

Altera Stratix V

XILINX Virtex 7



STEP 1 (PCIe Gen3)
intra-node
parallelism test

STEP 2 (HSSIO)
inter-node
parallelism test

We are here!!

STEP 3 (PCIe+HSSIO)
heterogeneous
computing cluster

node plan
- two CPUs
- four GPUs
- two FPGAs

Summary

- Next generation Exa-scale (Flops) system is a challenge to power consumption (Flops/W)
- Strong scaling is essential, and direct interconnection between accelerators is necessary, within chip and over chips
- TCA is a basic concept on the possibility on direct network between accelerators (GPUs) by current available technology
- FPGA is a very aggressive CoDesign solution, and TCA by PEACH2 on FPGA can handle both partial-offloading and communication based on GPU computation