UNILOGIC scalable heterogeneous HPC architecture

Iakovos Mavroidis

Telecommunication Systems Institute
iakovosmavro@gmail.com

Workshop on Reconfigurable Computing (WRC'2019)
Valencia, January 21st, 2019

ECOSCALE has received funding from the EU Union's Horizon 2020 research and innovation programme under the grant agreement No 671632

Outline

- Introduction : HPC Systems
  - Exascale challenges
  - Approach: Accelerators (GPUs vs FPGAs)
- FPGA in HPC
  - Programming Model
  - Limitations & Solutions (ECOSCALE)
    - UNIMEM/UNILOGIC architecture
    - Reconfiguration toolset
    - Runtime System
    - Execution Environment
- Conclusion

Top HPC Servers Today

- **Summit & Sierra** (IBM Power9, NVIDIA GV100)
  - 2M cores, 143 PFLOPS, 9 MW
    - Performance: 70 GFLOPS/core
    - Efficiency: **15.8 GFLOPS/W**, 4.5 W/core

Summit

- **SunWay TaihuLight** (Chinese)
  - 10M cores, 93 PFLOPS, 15 MW
    - Performance: 10 GFLOPS/core
    - Efficiency: **6.2 GFLOPS/W**, 1.5 W/core

SunWay TaihuLight

- **Tianhe-2 (Xeon E5, Phi)**
  - 3M cores, 33 PFLOPS, 17MW
    - Performance: 10 GFLOPS/core
    - Efficiency: **1.96 GFLOPS/W**, 5.1W/core

Tianhe-2

Problem: Just scaling not a viable solution to reach ExaFlop

Energy Efficiency
- 16 GFLOPS/W ⇨ **1EFLOPS/63 MW** (should go down to ~20MW)

Cost & Space
- 16 GFLOP/ core ⇨ 1 EFLOP / 14 Mcores

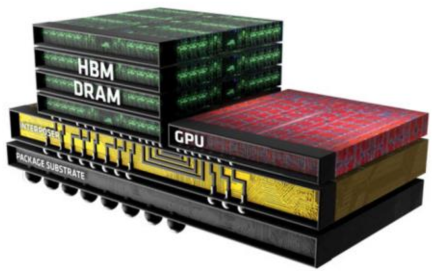
Scalability & Management
- Manage 14 Mcores

Resiliency

HPC Approach: Improve Technology, Architecture, Software

✓ **Technology**

- Transistor shrinking (TSMC 7 nm)
- Bring transistors closer: 3D technology



✓ **Architecture**

- Reduce data movements
  - Multi-level memory: scratchpad, cache, Flash
- Increase parallelism
- Von Neumann vs Dataflow Engines

✓ **Software**

- Programming Languages
- System Software/Runtime Systems
- Tune applications

Common Architecture Approach: Use of Accelerators

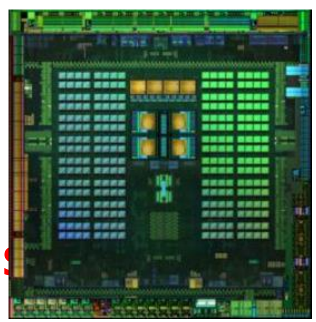
- Accelerators are based on
  - ✓ parallel processing
  - ✓ synchronized accesses/processing
  - ✓ locality

- **Many-core**
  - Intel Xeon-Phi
  - KALRAY MPPA

- **GPUs**
  - NVIDIA
  - AMD
  - ARM

- **Dataflow Engines**
  - FPGAs
    - Altera
    - Xilinx

Xeon-Phi

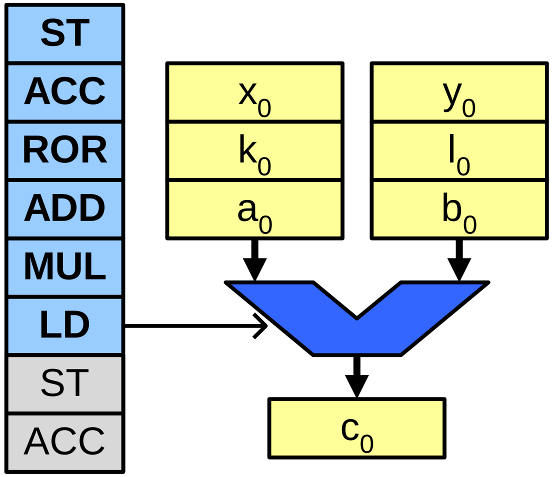


Tegra X1

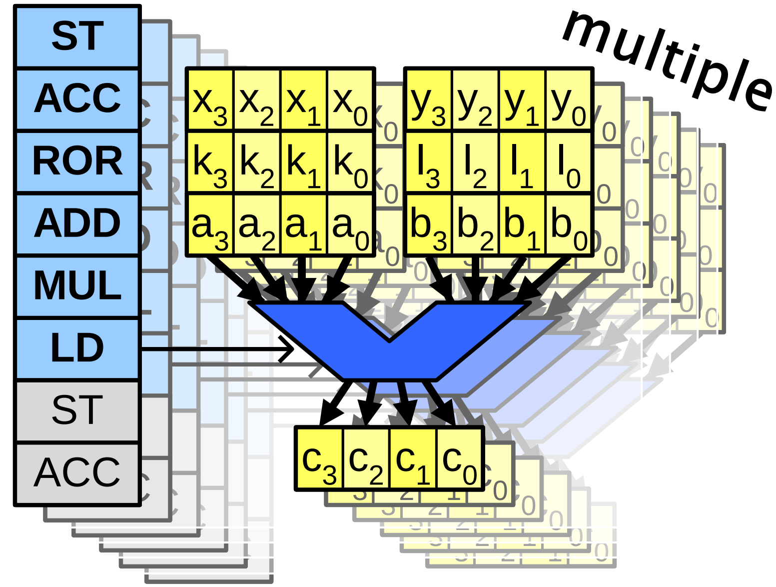
Ultrascale Board

CPUs vs. GPUs vs. Dataflow/FPGA

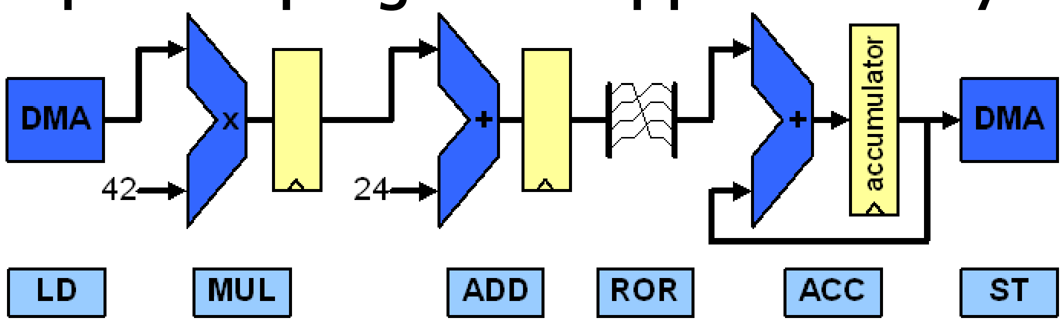
Scalar processing



SIMD processing (e.g. GPU)



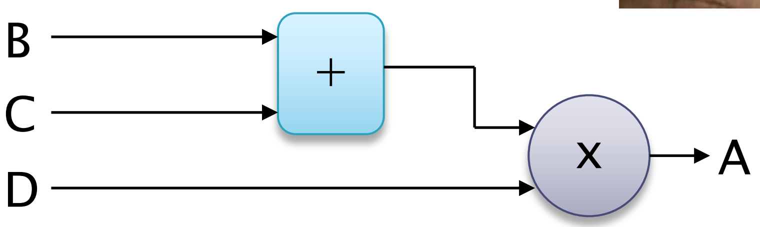
Dataflow graph of a program mapped directly in HW



Why are FPGA's so energy-efficient?

- Customized to the needs of the application
  - ✓ Instructions are part of the FPGA logic ☞ no IF, ID stages
  - ✓ Custom EX stage
  - ✓ Optimized data movements
  - ✓ Optimized control logic
  - ✓ Loop unrolling in HW

$$A = B + C$$
$$A = A * D$$



Dataflow Architectures

- Intel Configurable Spatial Accelerator (CSA)
  - Static HW, many CSA configurations
- Intel “Two Orders of Magnitude Faster than GPGPU by 2020”:
  - Deep Learn. Inference Accel. (DLIA) with Altera Arria 10
  - Broadwell Xeon with Arria 10 GX
- EC2 F1 instance with Xilinx
  - Up to 8xUltraScale+ VU9P per instance
- IBM SuperVessel OpenPOWER development cloud using Xilinx SDAccel
- Microsoft Bing with Altera Stratix V
- FPGA networking for Azure cloud
- Xilinx SDAccell on Nimbix cloud
- Xilinx OpenStack support
  - Libraries: DNN, GEMM, HEVC Decoder & Encoder, etc.

Still not used extensively – Why? Low Programmability

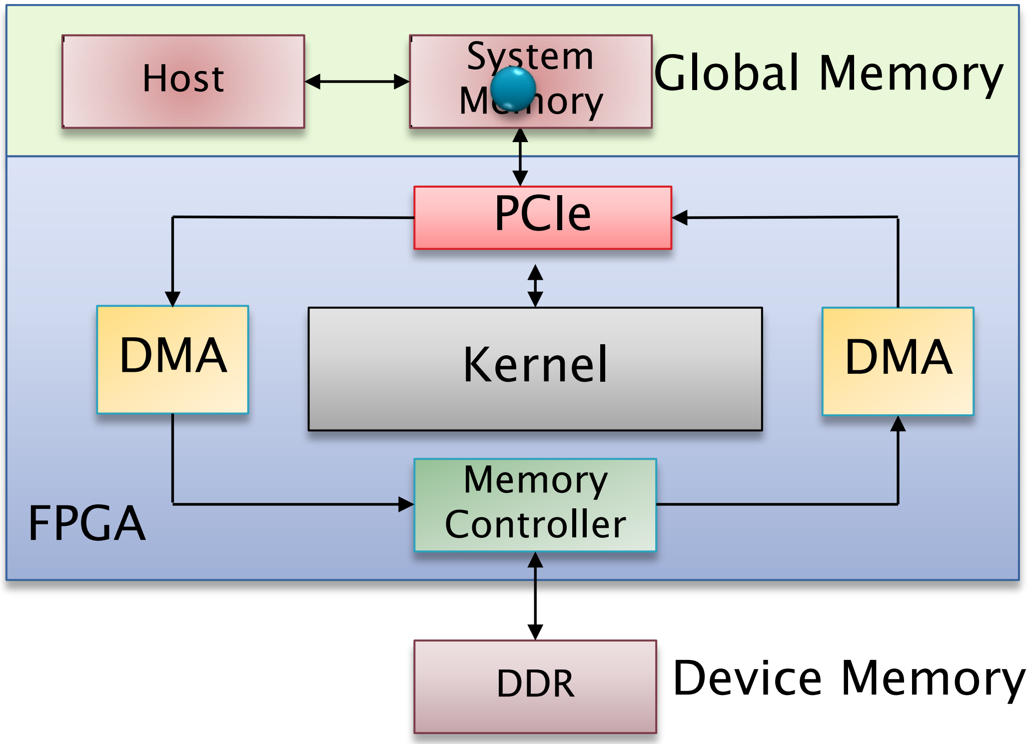
- Programmability
  - Huge Design Space ☞ hard to optimize application
    - Many FPGAs and choices
    - When/Where/What/How to accelerate
  - SW and HW skills required
- Compile/Synthesis time
  - Several hours to synthesize/PnR
  - Long time to program
- FPGA Size
  - FPGA Size directly affects speed and number of accelerated tasks

If your system is hard to program it,
it really doesn't matter how fast it is.

What should we do?
Help the programmer!

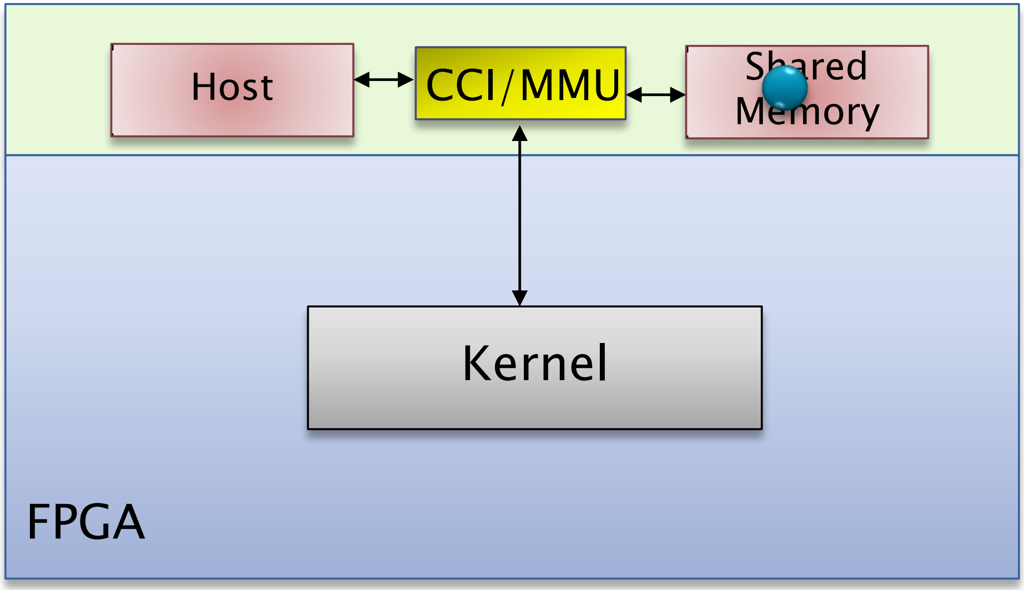
- **Improve Architecture**
  - Multi-FPGA support
  - Multi-user support/shared resources
  - OS support (device drivers)
  - Runtime System
    - Dynamic Scheduling
    - Dynamic reconfiguration
- **Improve Programming Language**
  - Support of OpenCL 2.0
    - Shared virtual memory
    - Pipes
    - Dynamic Parallelism, Atomics, etc.
- **Improve CAD Tools**
  - Intel FPGA SDK for OpenCL (supports part of OpenCL 2.0)
  - Xilinx Vivado HLS (supports OpenCL 1.1)

Traditional OpenCL Data Movement



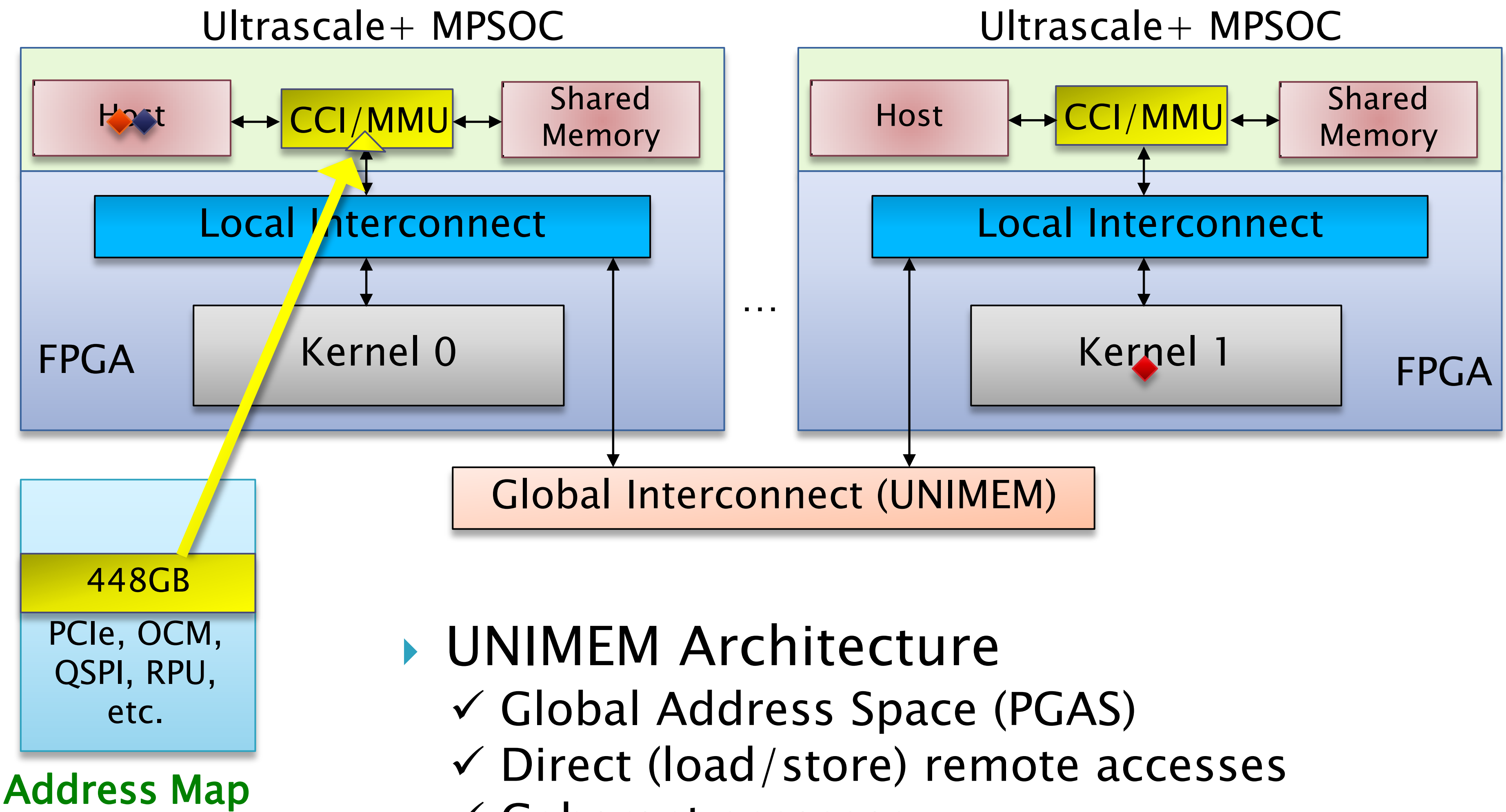
- Global Memory of Accelerator is Independent from Host/System Memory
- High-Latency PCIe

OpenCL 2.0: Virtual Shared Memory



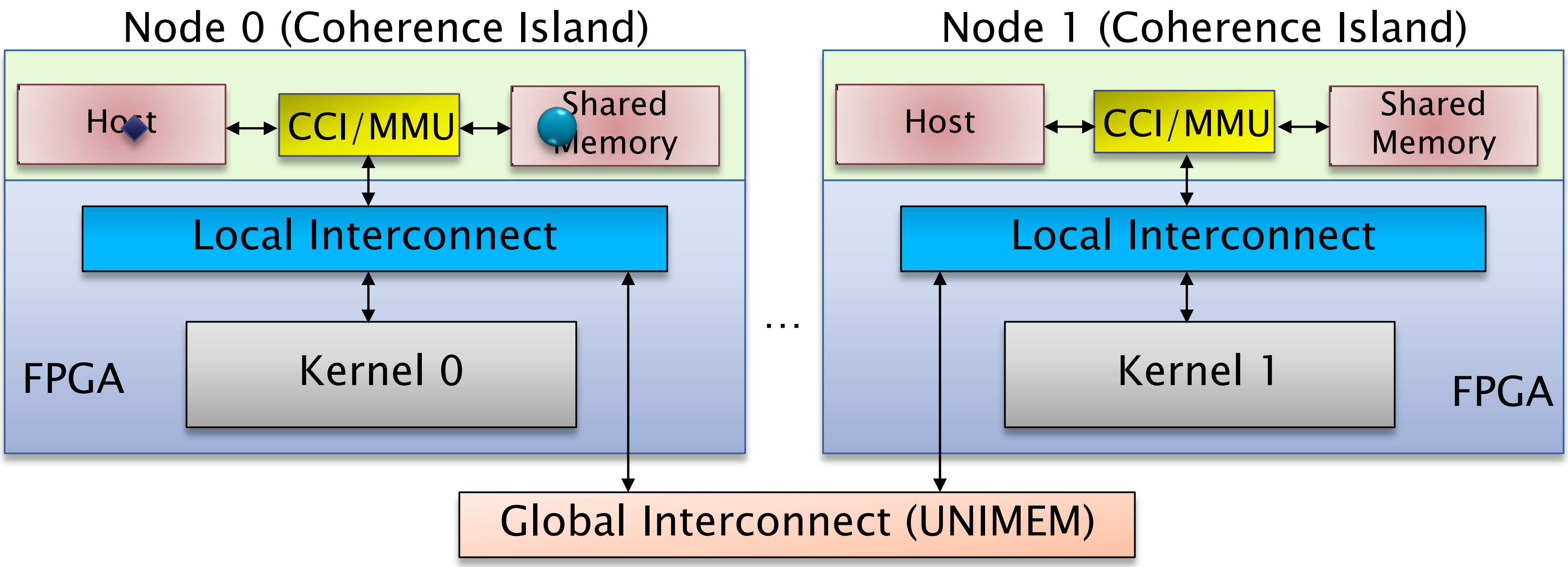
- **Cache Coherency**
  - ✓ ARM CCI (Ultrascale+)
  - Xilinx CCIX (next generation of Ultrascale+)
  - IBM CAPI (Intel QPI/CAPI)
- **IO MMU**
  - ✓ ARM SMMU (Ultrascale+)
  - ☞ Coherent Cache on the FPGA (Ultrascale+)?

UNIMEM: Remote Coherent Accesses



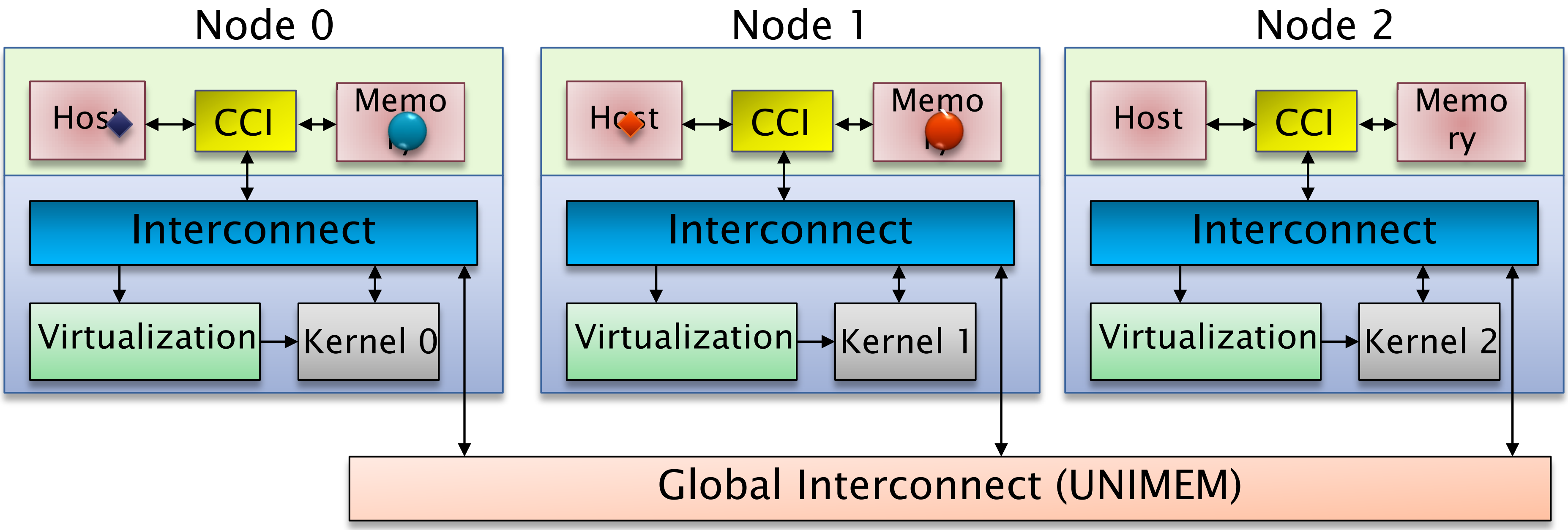
- UNIMEM Architecture
  - ✓ Global Address Space (PGAS)
  - ✓ Direct (load/store) remote accesses
  - ✓ Coherent accesses

Multiple FPGAs



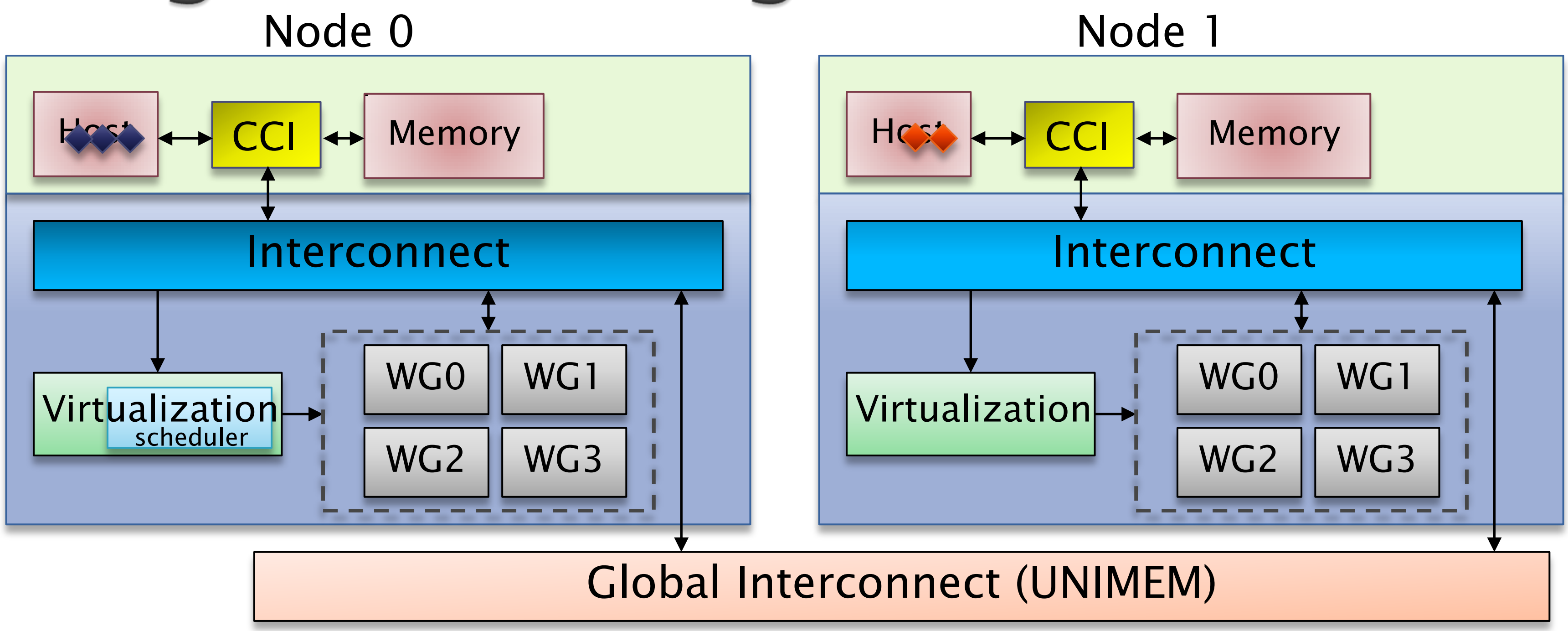
- Access any FPGA in the System
  - ✓ Remote Kernel calls
  - ✓ Remote Memory Accesses

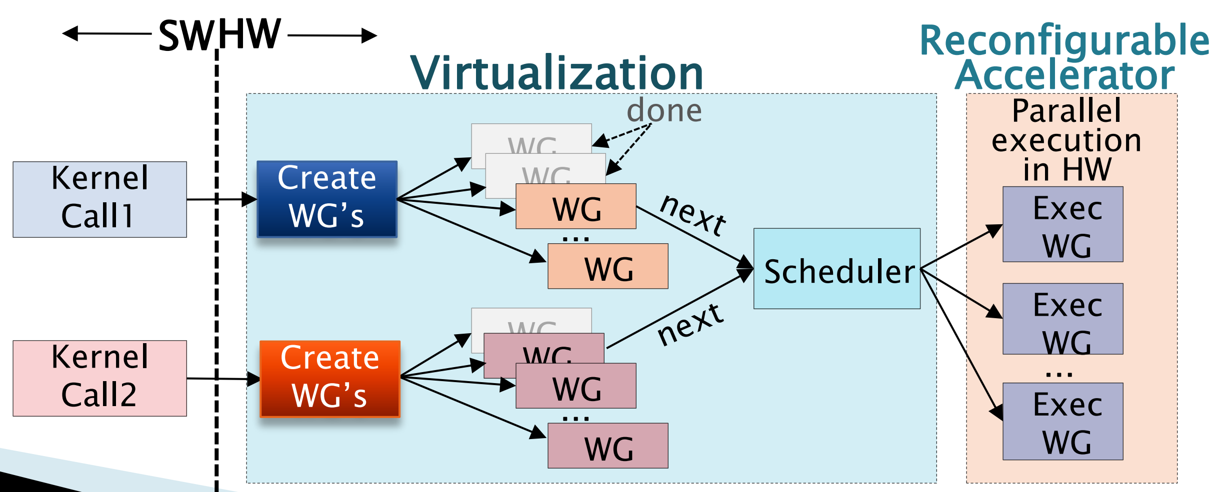
Resource sharing



- Virtualization Block
  - Receives Kernel Execution command from any Node
  - Schedules commands and executes them locally

Fine-grain sharing

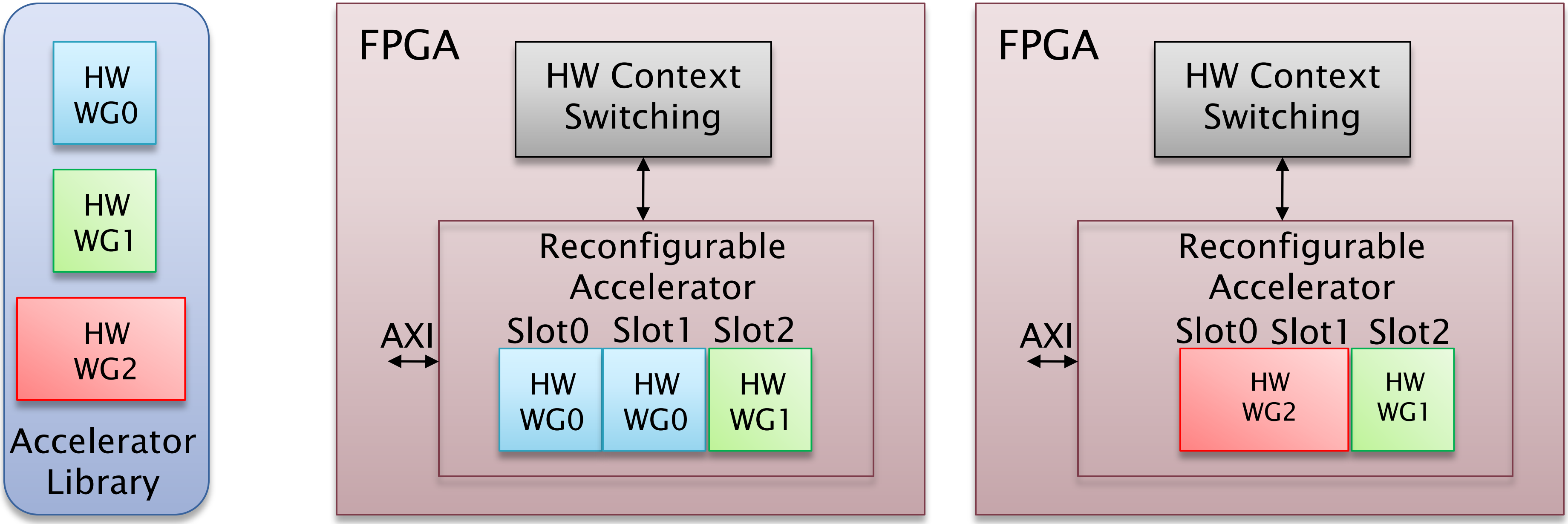




Dynamic Reconfiguration: Challenges and Potential Solutions

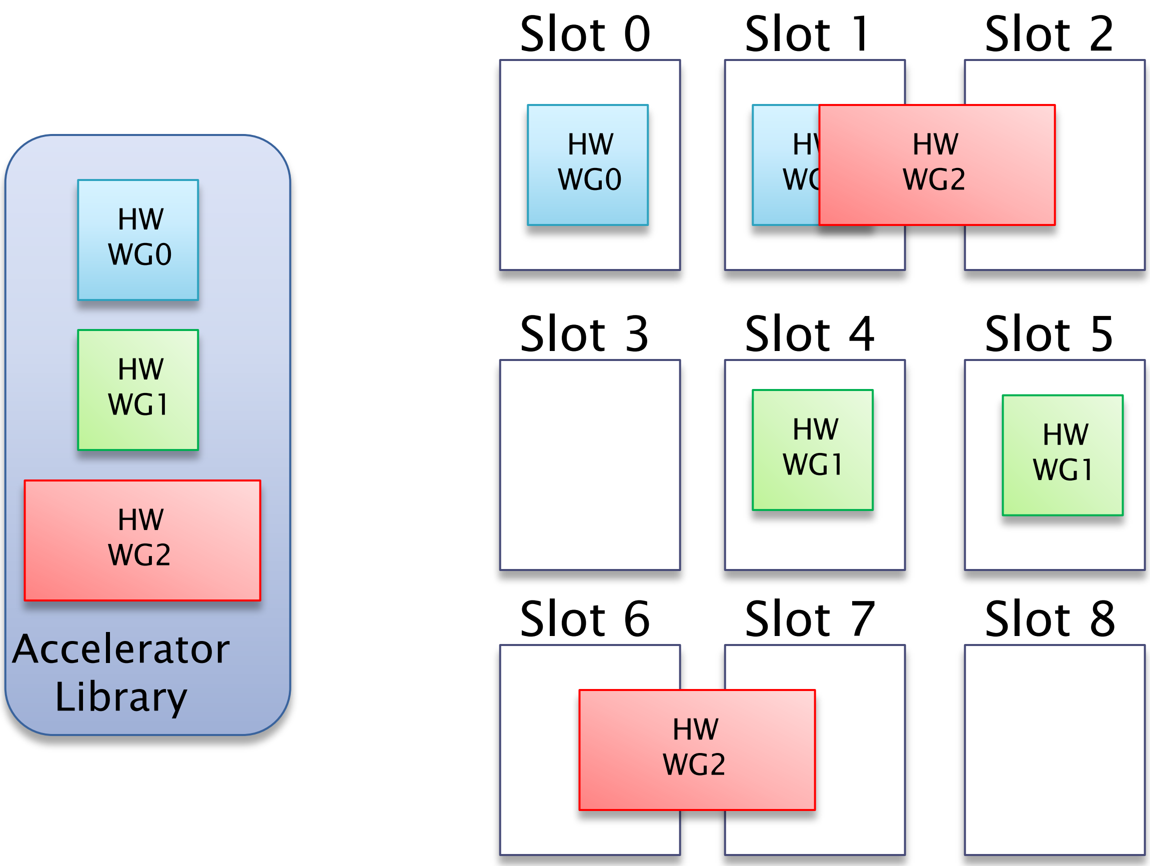
- Synthesis/PnR too slow → Synthesis at compile time
- Bitstream per FPGA per location → Improve Placement
- Reconfiguration is slow → Prefetching
  GPU/CPU as alternative
- Limited FPGA size → Improve Architecture (UNIMEM)
- Limited tool support → Standardization/ Improve Tools
- When/what/where to reconfigure → Runtime System

Physical Implementation



- **Resource-aware Reconfigurable Accelerator Floorplanning and Backend Tool**

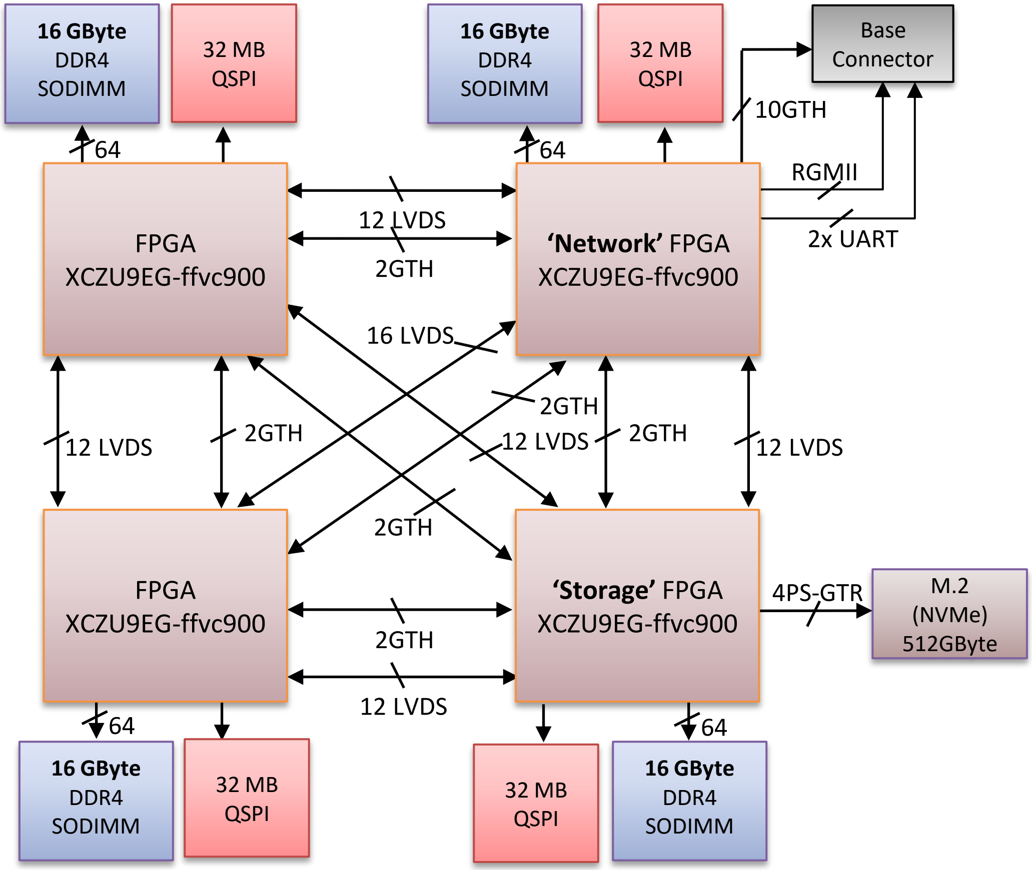
ECOSCALE Recofiguration



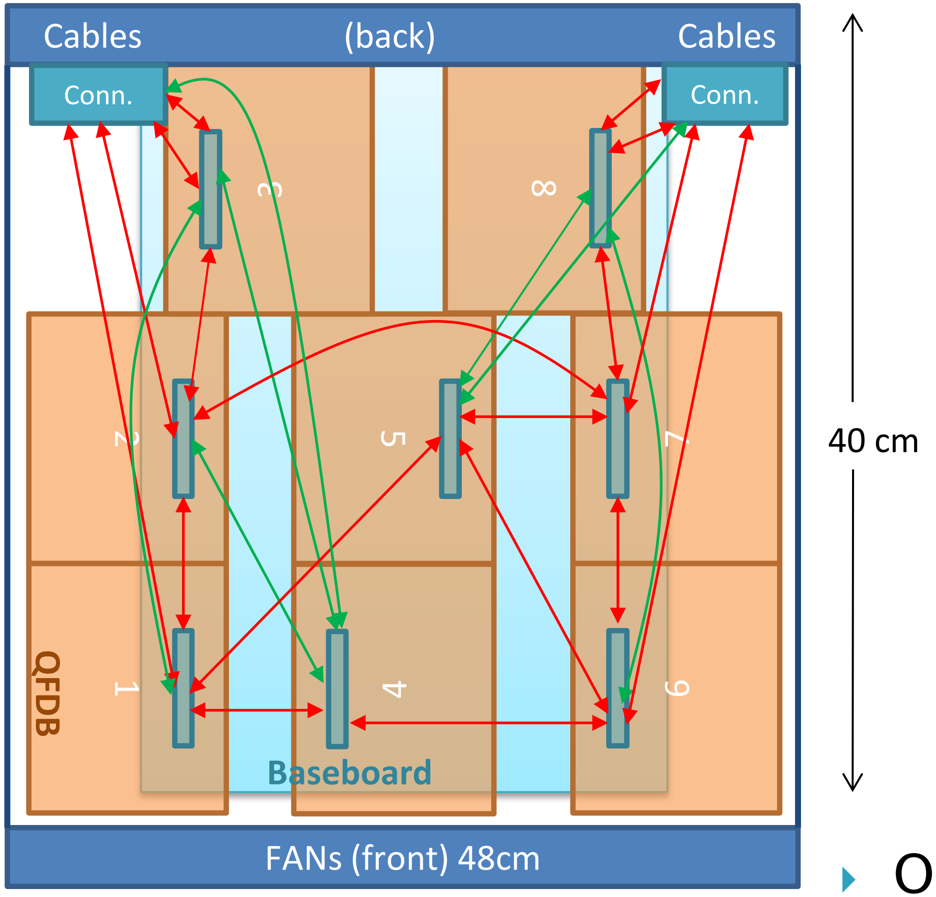
- Acceleration Library
- WG's of different size
- Relocation
  - Defregmentation
  - Move logic closer to data

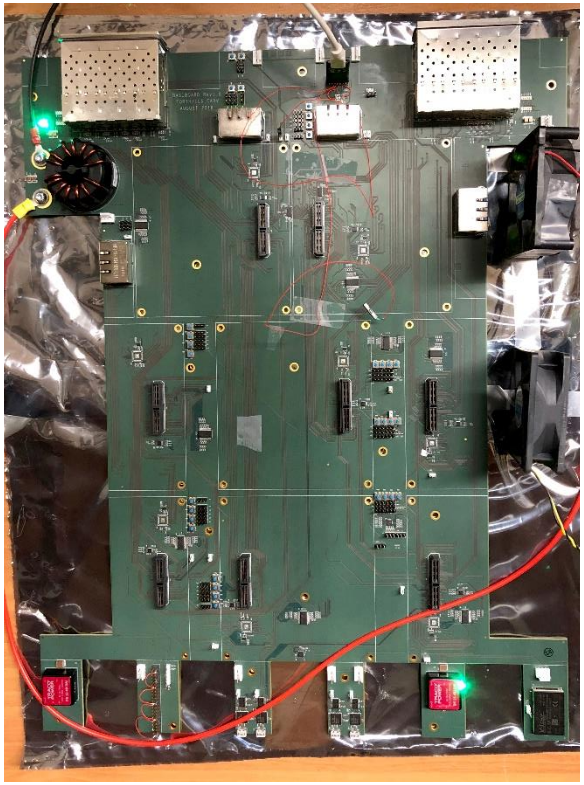
The ExaNeSt QFDB

- Quad-FPGA Daughter Board
  - Designed by sister project ExaNeSt
  - 4x Xilinx Zynq Ultrascale+ FPGAs
  - 4x16 = 64GB DDR
  - Extensive High Speed Connectivity
  - Complex & dense PCB
  - 16 layers, 120x130mm, 16 power sensors



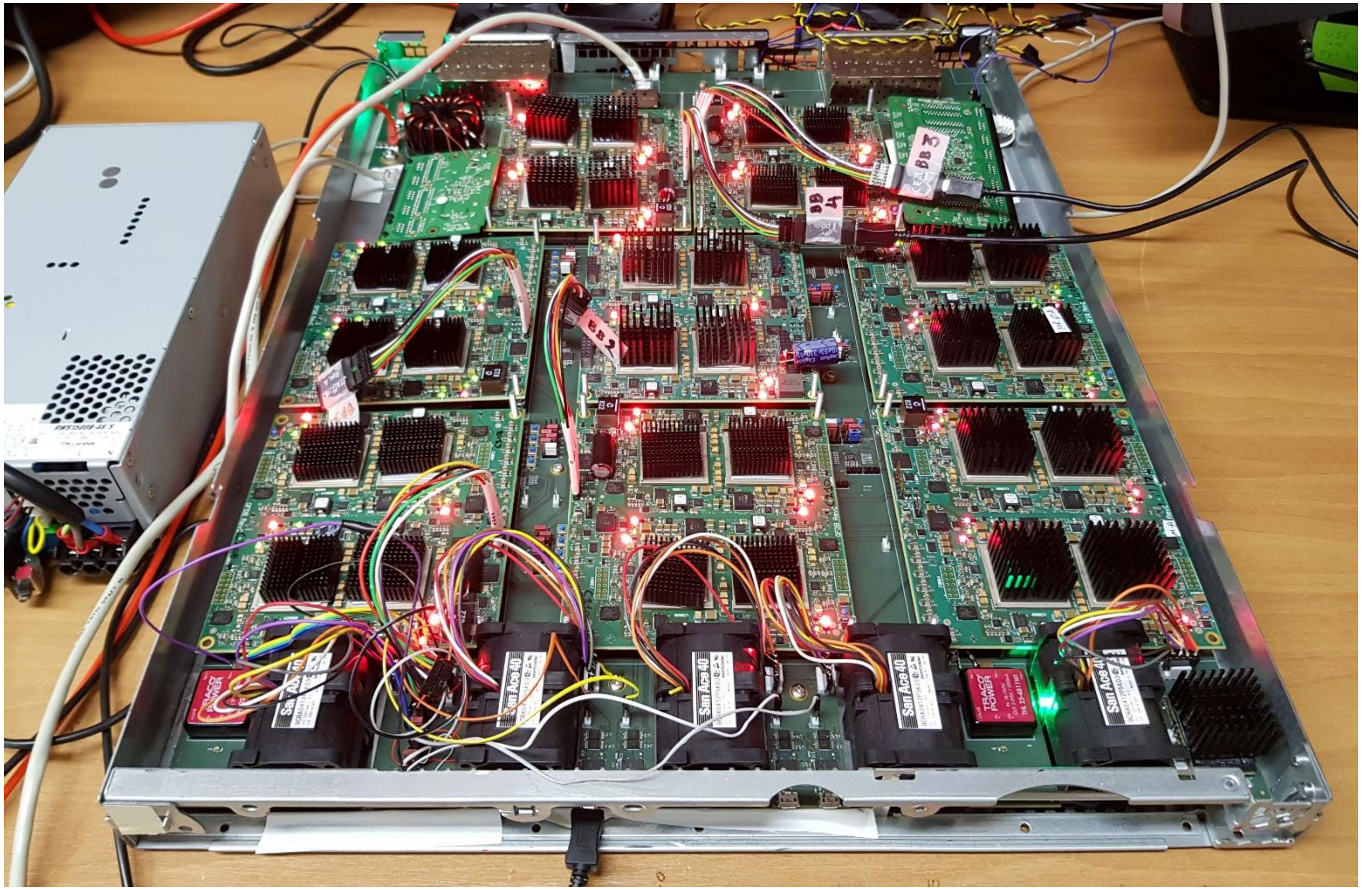
The ECOSCALE BaseBoard





- Offers a densely QFDB-populated prototype
- Provides many option for intra- and inter-board connectivity

Fully Populated Baseboard



Conclusion

- Common HPC Approach
  - ☞ Use accelerators: GPUs vs FPGAs
- FPGAs main advantage
  - 👍 **Energy Efficiency**
  - ☝ But hard to be used by programmers
    - Limited FPGA Size ☞ Sharing of resources
    - Reconfiguration ☞ Optimize tools/Standardization
    - Runtime system which automates/coordinates actions
- UNILOGIC architecture and ECOSCALE firmware
  - 👍 Improved Programmability

www.ecoscale.eu

@ecoscale_H2020

Stay Tuned!