21st Summer School of PARALLEL COMPUTING

July 2 - 13, 2012 (Italian)
September 3 - 14, 2012 (English)

BGQ PowerA2 processor

Carlo Cavazzoni

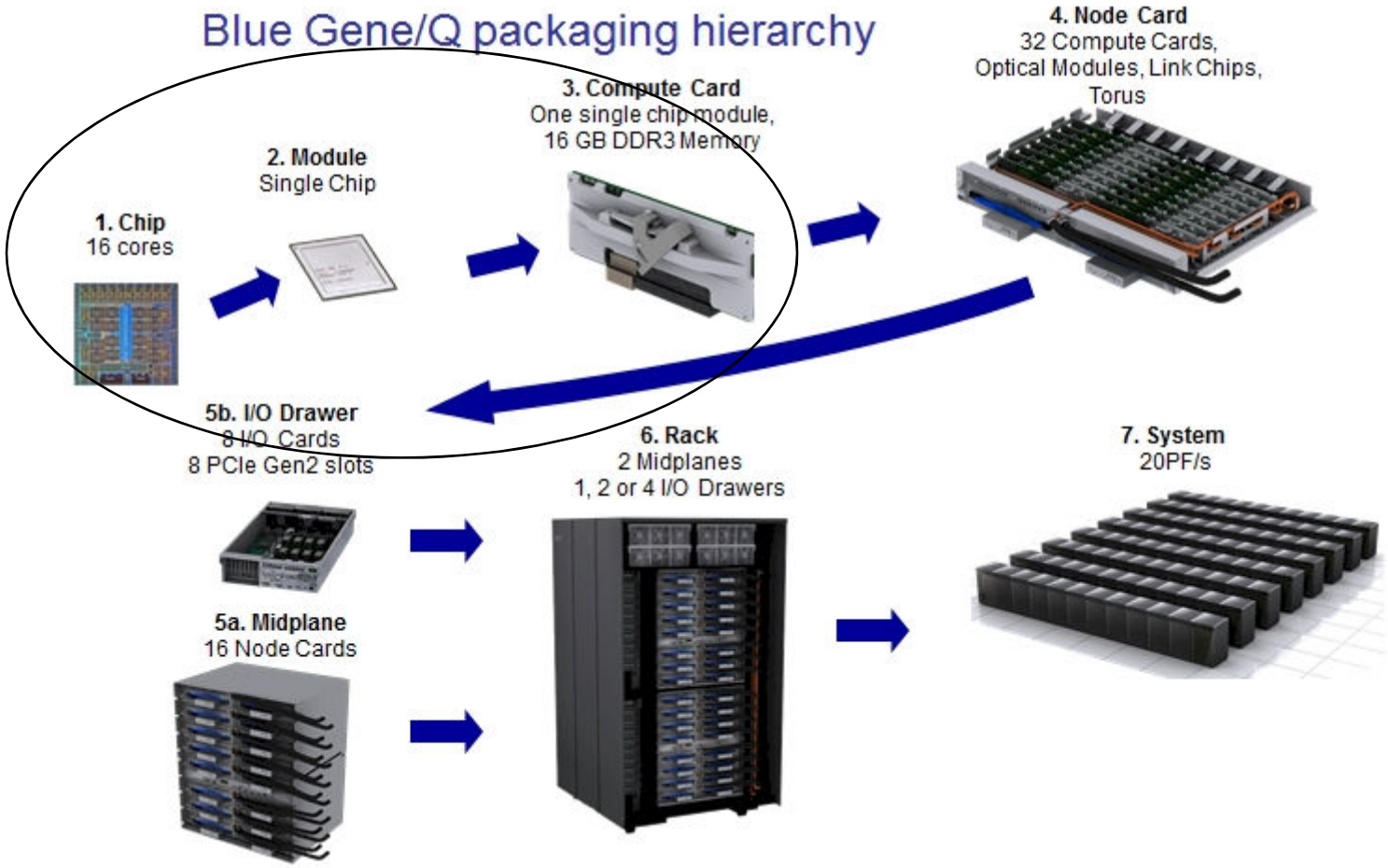
CINECA

Power processors family

Power1…Power7
PowerPC
PowerA2

RISC processors
Superscalar
Multiple Floating Point units
SMT
Multicore

BGQ


Blue Gene/Q packaging hierarchy
1. Chip
16 cores
2. Module
Single Chip
3. Compute Card
One single chip module,
16 GB DDR3 Memory
4. Node Card
32 Compute Cards,
Optical Modules, Link Chips,
Torus
5a. Midplane
16 Node Cards
5b. I/O Drawer
8 I/O Cards
8 PCIe Gen2 slots
6. Rack
2 Midplanes
1, 2 or 4 I/O Drawers
7. System
20PF/s

PowerA2 chip, basic info

16 cores + 1 + 1
1.6GHz
32MByte cache
system-on-a-chip design
16GByte of RAM at 1.33GHz
Peak Perf 204.8 gigaflops
power draw of 55 watts
45 nanometer copper/SOI process (same as Power7)
Water Cooled

PowerA2 chip, more info

Originally created for networking devices and experimentation

800MHz crossbar switch

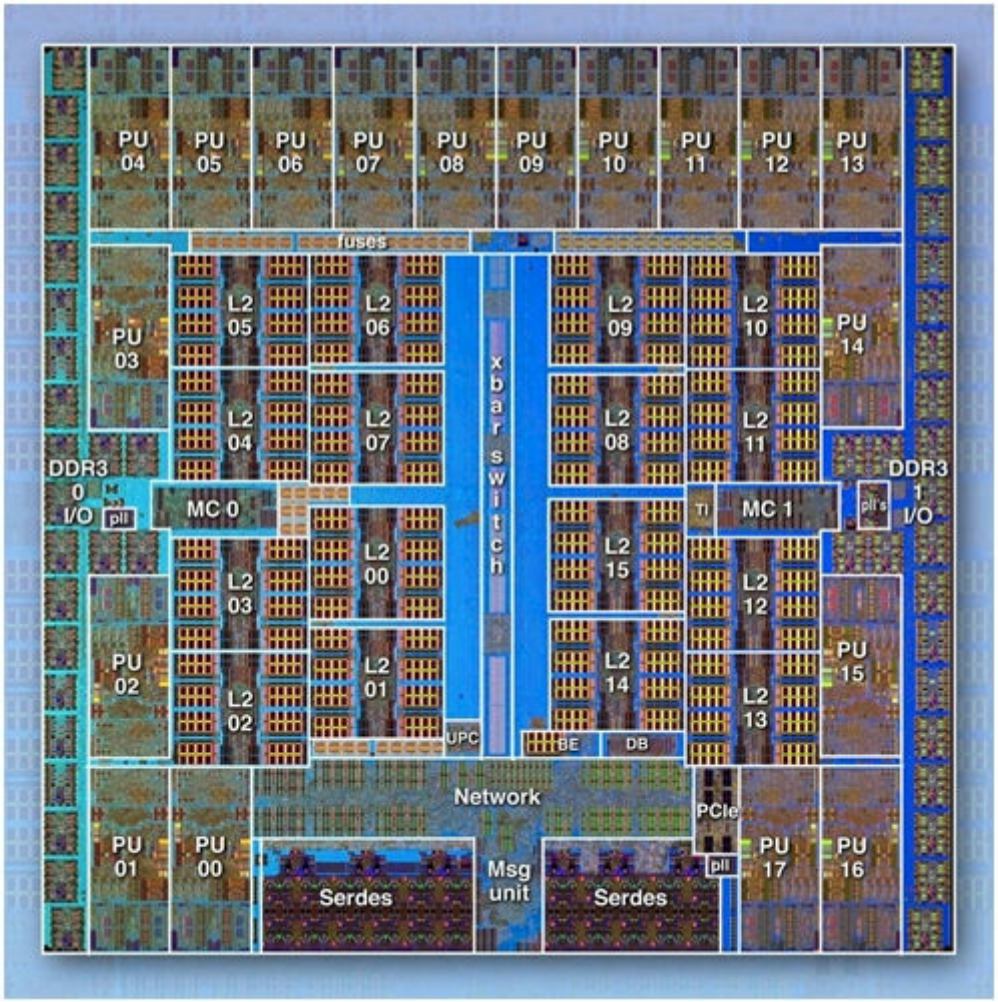
- links the cores and L2 cache memory together
- peak bisection bandwidth of 563GB/sec
- connects the processors, the L2, the networking, and other parts of the chip

5D torus interconnect is also embedded on the chips, with 11 links running at 2GB/sec

Two of these can be used for PCI-Express 2.0 x8 peripheral slots.

The 14-port crossbar switch/router at the center of the chip supports point-to-point, collective, and barrier messages and also implements direct memory access between nodes.

PowerA2 chip, layout



PowerA2 core

4 FPU

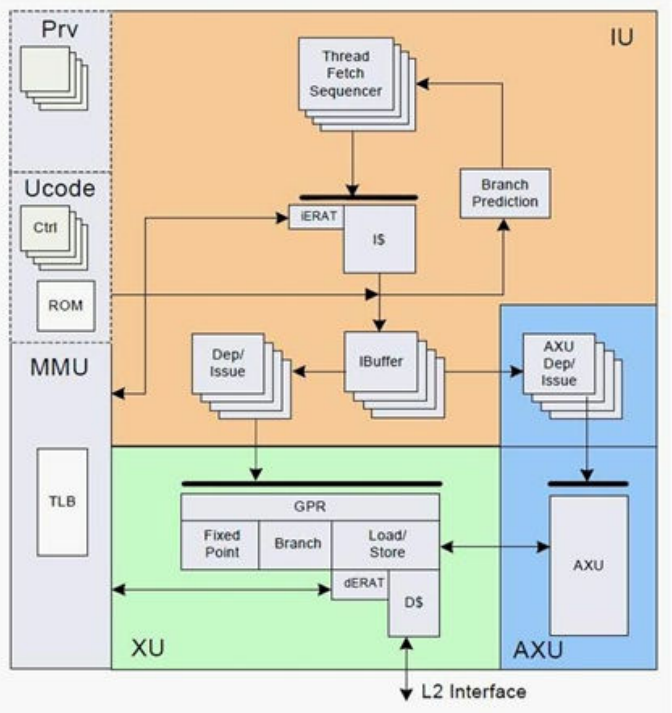
4 way SMT

64-bit instruction set

in-order dispatch, execution, and completion

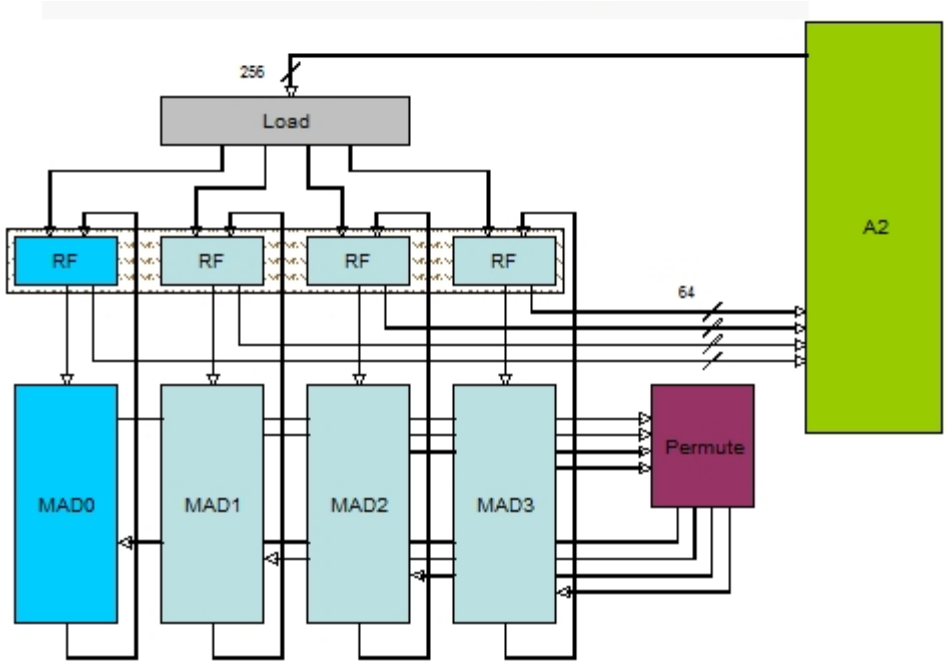
16KB of L1 data cache

16KB of L1 instructions cache



PowerA2 FPU

Each FPU on each core has four pipelines
execute scalar floating point instructions
Quad pumped
four-wide SIMD instructions
two-wide complex arithmetic SIMD instructions
six-stage pipeline
permute instructions
maximum of eight concurrent
floating point operations
per clock plus a load and a store.



PowerA2 transactional memory

Performance optimization for critical regions

1. Software threads enter “transactional memory” mode
   - Memory accesses are tracked.
   - Writes are not visible outside of the thread until committed.
2. Perform calculation without locking
3. Hardware automatically detects memory contention conflicts between threads
   - If conflict:
     - TM hardware detects conflict
     - Kernel decides whether to rollback transaction or let the thread continue
     - If rollback, the compiler runtime decides whether to serialize or retry
   - If no conflicts, threads can commit their memory

Threads can commit out-of-order.

XL Compiler only