Theoretical Physics on Supercomputers

Hubert Simma

Università Milano Bicocca

Plan:

- ❑ Introduction
- ❑ Solution Steps
- ❑ Examples of Applications and Methods
- ❑ Lattice QCD
- ❑ Machines

Computational Tasks of LQCD

Run-time Profile:

useful (if not yet known from theoretical analysis) to determine

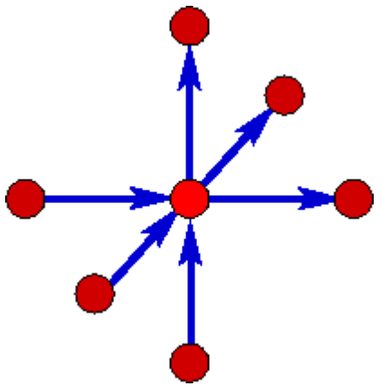
- algorithmic cost (how often is each computational tasks needed)
- CPU cost (how much time is spent for each computational tasks)

1 HMC trajectory, $24^3 \times 32$ lattice

routine	calls	time
Dirac operator (3 variants)	80844	58 %
Linear algebra (3 routines)	60736	26 %
Gauge forces + update	320	8 %
Global sum ($4 \times 8 \times 8$ nodes, 128 bit)	83554	0.4 %
Others ($\approx$ 70 routines)		7 %

➜ Dominant Task: Wilson-Dirac Operator

$$\phi' \equiv [D\phi]_x = \sum_{\mu=1}^{4} \{U(x,\mu)(1-\gamma_\mu)\phi(x+\hat{\mu}) + \cdots\}$$



LQCD on Parallel Computers

Data Storage:

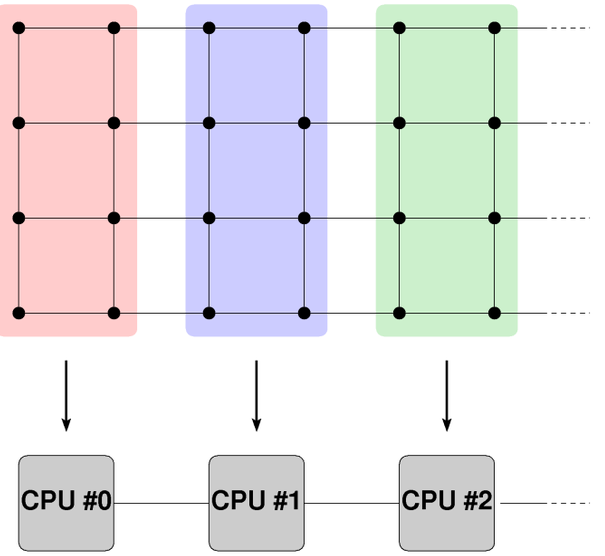
$\psi(x)$: 12 complex/site

$U(x, \mu)$: 9 complex/link

Lattice size $V = L^3 \times T$

e.g. $64^3 \times 128 \Rightarrow 3 \cdot 10^7$ sites (10 GB for gauge field U)

Processor grid: $P_0 \times P_1 \times P_2 \times P_3 = P$



➜ Trivial parallelisation by data distribution (uniform, static)

Communications:

- mainly nearest neighbour
- bandwidth requirements depends on implementation, algorithm, and physics choices

Number of remote neighbour sites (i.e. on different processor):

$$A^+ = \frac{V}{P} \sum_i \frac{1}{L_i} \qquad (P_i > 1)$$

Analysis of Computational Tasks

Total Computing Cost

$$N_{ops} \equiv \#\ \text{FP operations} \qquad = 1320$$

Computing vs. Memory Access

$$R_{ops} \equiv \frac{\#\ \text{arithmetic operations}}{\#\ \text{memory accessess}} \qquad \approx 7$$

Communication Requirement

$$R_{rem} \equiv \frac{\#\ \text{remote accesses}}{\#\ \text{memory accessess}} \qquad \sim A^{+}/V$$

Hardware Characteristics

Memory System

$$\rho_{mem} \equiv \frac{\text{flops}}{\text{bandwidth}} \quad [flop/byte]$$

Communication Network

$$\rho_{net} \equiv \frac{\text{network bandwidth}}{\text{memory bandwidth}}$$

Balance: Application vs. Hardware

$$R_{ops} \approx \rho_{mem} \qquad \text{and} \qquad R_{rem} \approx \rho_{net}$$

N.B.: Depending on (and to be taken into account in) various steps of the solution process

Example: APE Machines

History:

- mid 80's: APE1 idea for "Array Processor Experiment"
  by theoretical physicists at INFN
  $\longrightarrow$ 1 GFlops APE1 prototypes

- 1989–1994: APE100 full-custom development by INFN
  $\longrightarrow$ $O(200)$ GFlops installations (Quadrics)

- 1995–2000: APEmille development by INFN (+DESY)
  $\longrightarrow$ O(2) Tflops installations (Eurotech)

  CP-PACS/Hitachi

  QCDSP/Columbia

- 2001–2006: apeNEXT collaboration by INFN, DESY, Orsay
  $\longrightarrow$ O(15) Tflops installations (Eurotech)

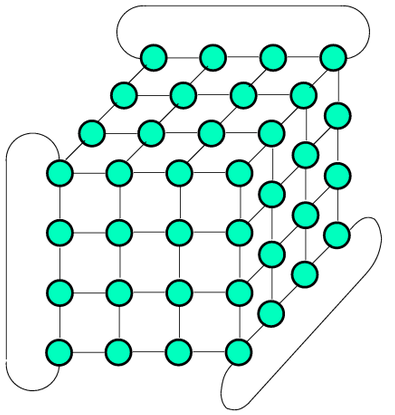
  QCDOC/Columbia

  BlueGene/IBM

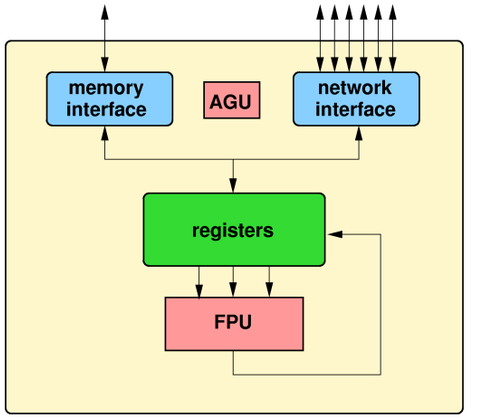
APE has provided major computing resources for LQCD in Europe

Optimization of APE Architectures for LQCD

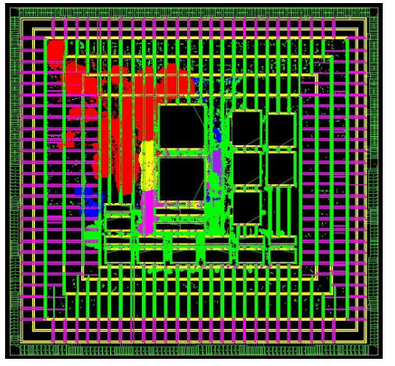
- 3D torus network

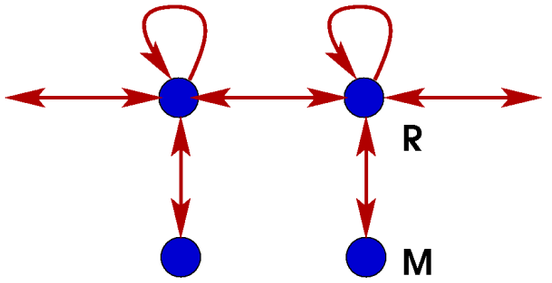


- slow clock, but many FP operations (complex) per cycle
  ⇒ low power consumption
- integrated memory and communication interface
  ⇒ compact design
- Very Long Instruction Word (VLIW) architecture
  ⇒ optimized scheduling at compile-time

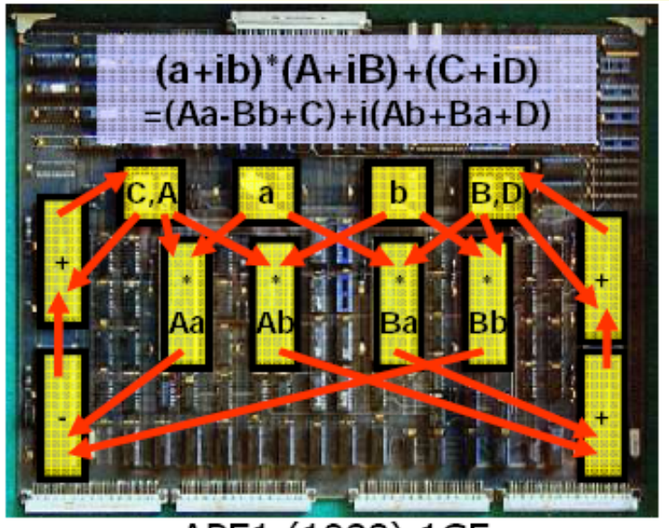


- large register file instead of cache
  ⇒ predictable and synchronous execution
- global syncronisation (mechanisms)
- RAS (ECC, status registers, . . .)
  ⇒ duration of single program execution $O(days)$
- SIMD programming model + communications by address mapping



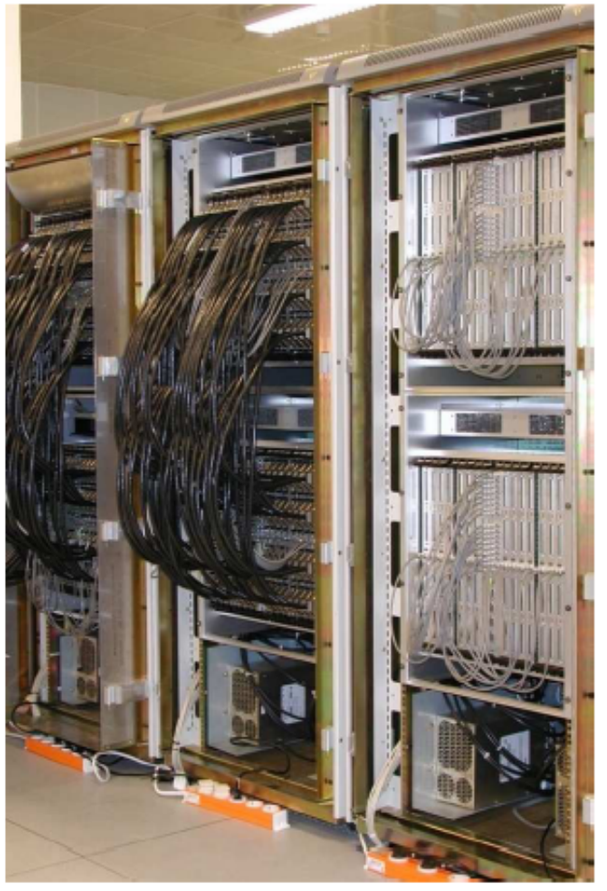


APE1 — APE100 — APEmille — apeNEXT

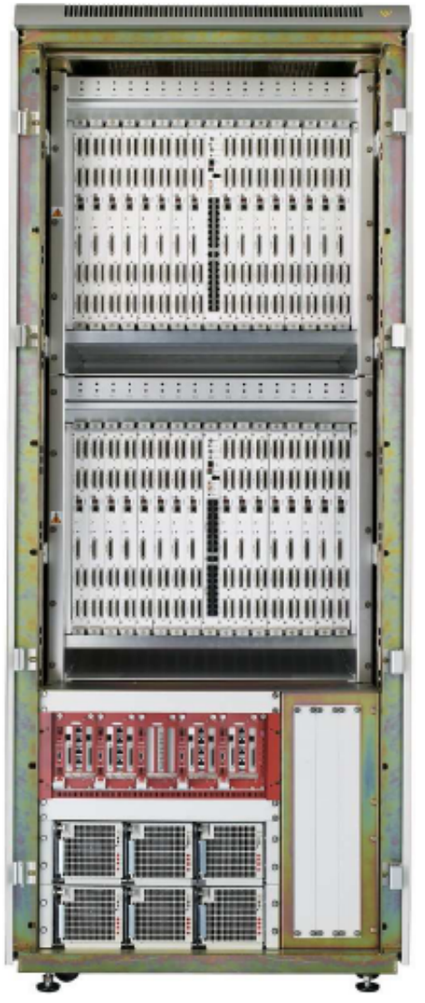


APE1 (1988) 1GF

APE100 (1992) 25GF, SP, REAL



APEmille (1999) 128GF, SP, Complex



apeNEXT (2004) 800GF, DP, Complex

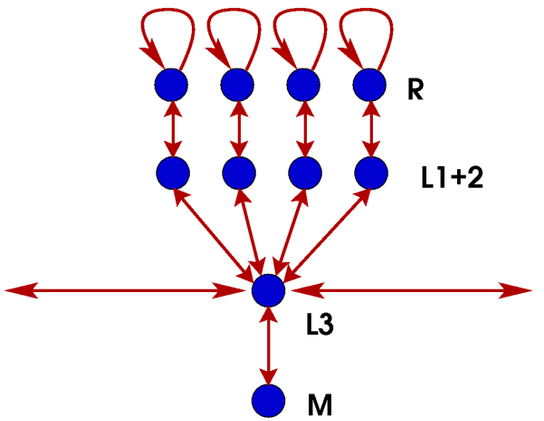
Example: BlueGene/L and P

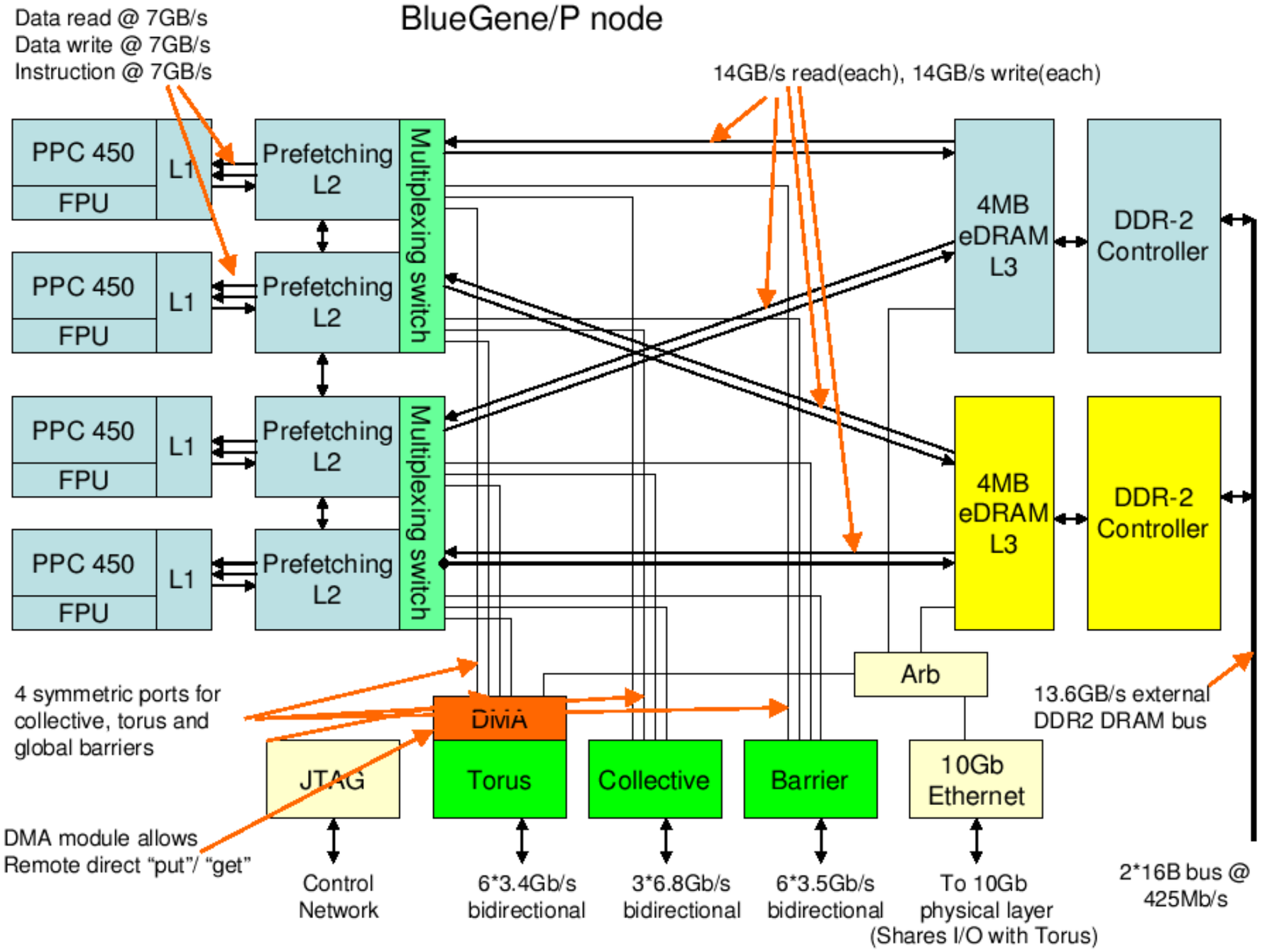
Characteristics: [QCDOC]

- PowerPC 440 core @ 850 MHz [500 MHz]
- MIMD distributed memory
- on-chip L1, L2, L3 caches (8 MB) [4 MB]

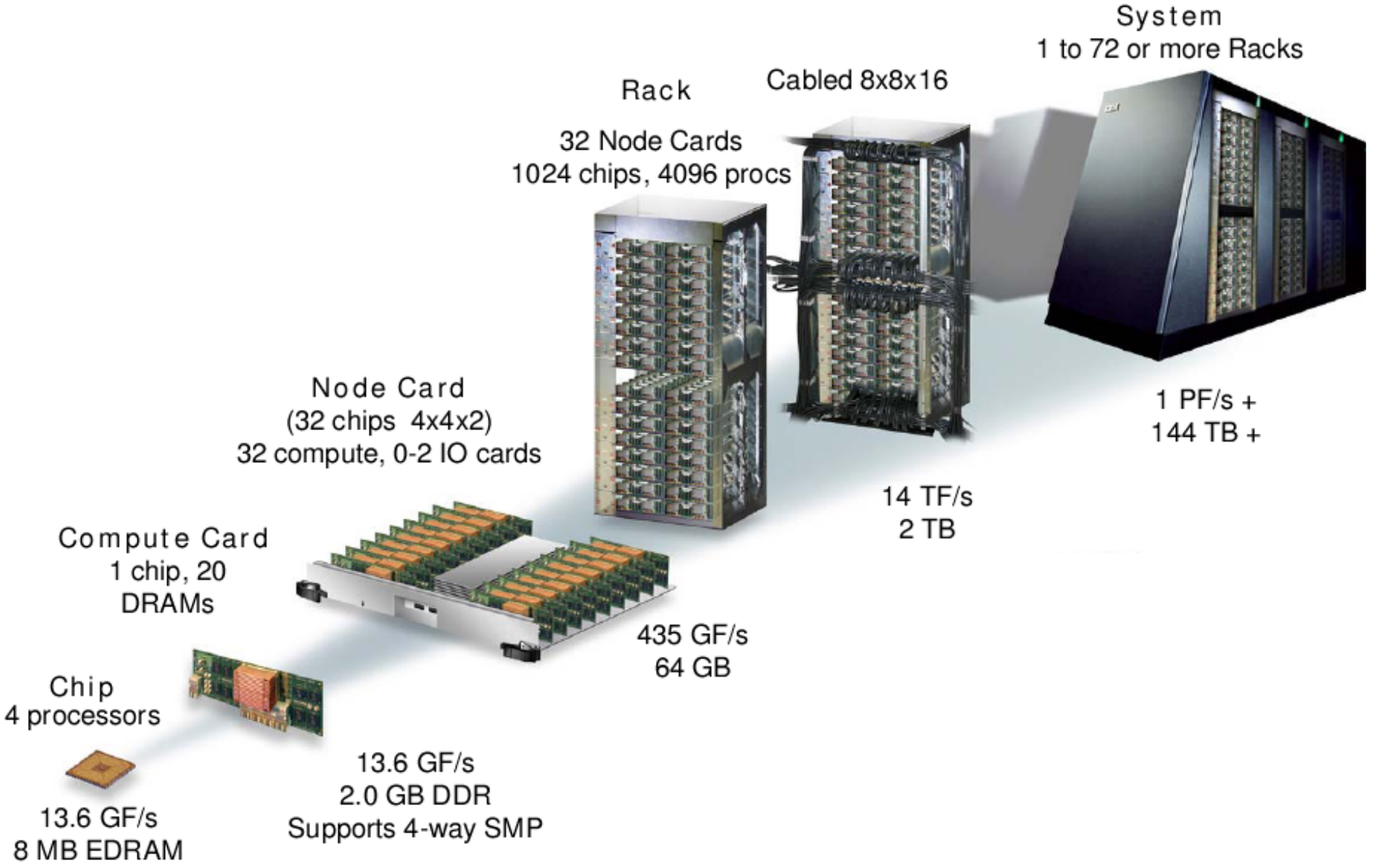
Node:

- 4 cores [1 core]
- 2 GB DDR
- 3-d torus network (6 links at 425 MB/s = 0.5 B/clock) [6-d at 65 MB/s]
- tree network (850 MB/s)




BlueGene/P node
Data read @ 7GB/s
Data write @ 7GB/s
Instruction @ 7GB/s
14GB/s read(each), 14GB/s write(each)
PPC 450
FPU
L1
Prefetching L2
Multiplexing switch
4MB eDRAM L3
DDR-2 Controller
Arb
DMA
JTAG
Torus
Collective
Barrier
10Gb Ethernet
4 symmetric ports for collective, torus and global barriers
DMA module allows Remote direct "put"/ "get"
13.6GB/s external DDR2 DRAM bus
Control Network
6*3.4Gb/s bidirectional
3*6.8Gb/s bidirectional
6*3.5Gb/s bidirectional
To 10Gb physical layer (Shares I/O with Torus)
2*16B bus @ 425Mb/s

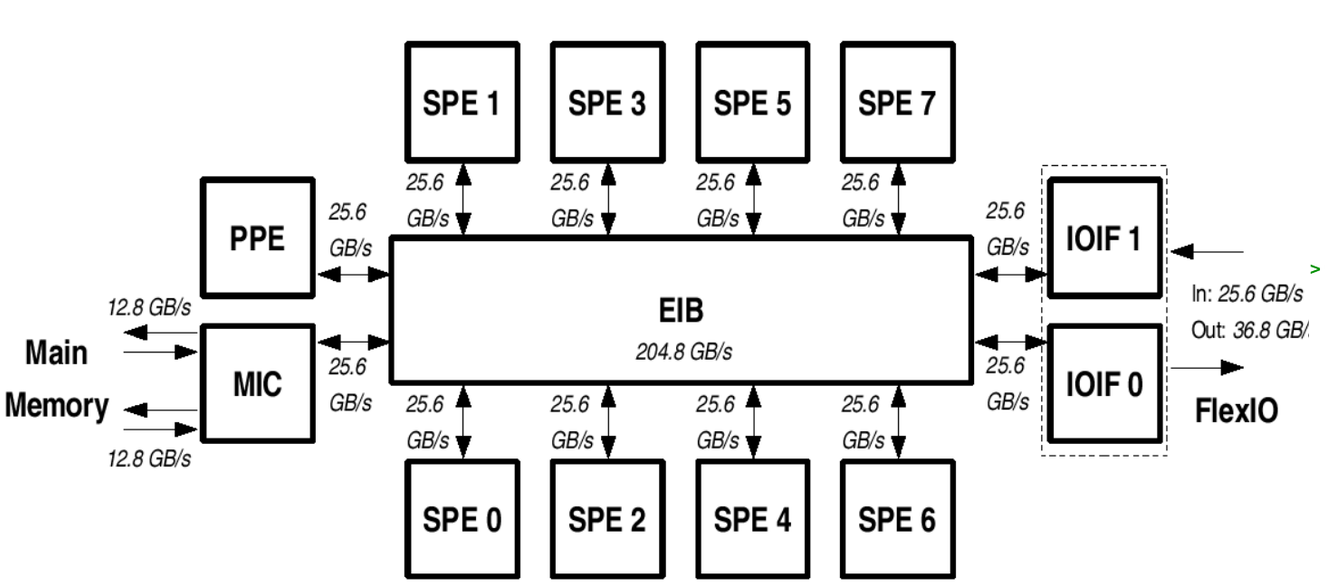
System

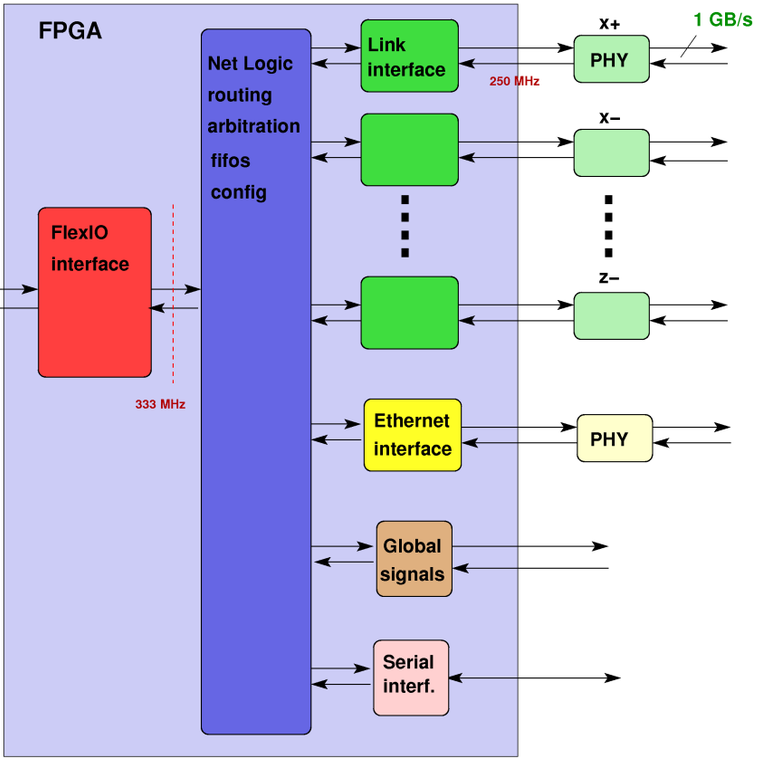

System
1 to 72 or more Racks
Rack
Cabled 8x8x16
32 Node Cards
1024 chips, 4096 procs
Node Card
(32 chips 4x4x2)
32 compute, 0-2 IO cards
Compute Card
1 chip, 20
DRAMs
Chip
4 processors
13.6 GF/s
8 MB EDRAM
13.6 GF/s
2.0 GB DDR
Supports 4-way SMP
435 GF/s
64 GB
14 TF/s
2 TB
1 PF/s +
144 TB +

Example: Cell / QPACE

- Idea: Combine enhanced Cell BE (PowerXCell 8i) with APE-like torus network
  - 200 GFlop/s single precision (peak)
  - 100 GFlop/s double precision (peak)
  - DDR2 memory controller
  - $\sim 25\%$ sustained performance (performance model)
- Collaboration between academic partners (Uni Regensburg, Jülich, DESY, Milano, Ferrara) and IBM Research Lab Böblingen
- Funded as part of SFB/TR-55 by Deutsche Forschungsgemeinschaft (DFG)
- Design work started $\approx$ Jan 2008
- Running Board Jul 2008
- Production of 2048-node machine early 2009

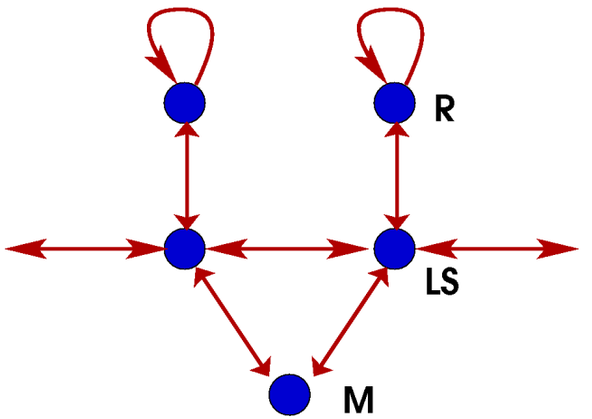
Cell/QPACE



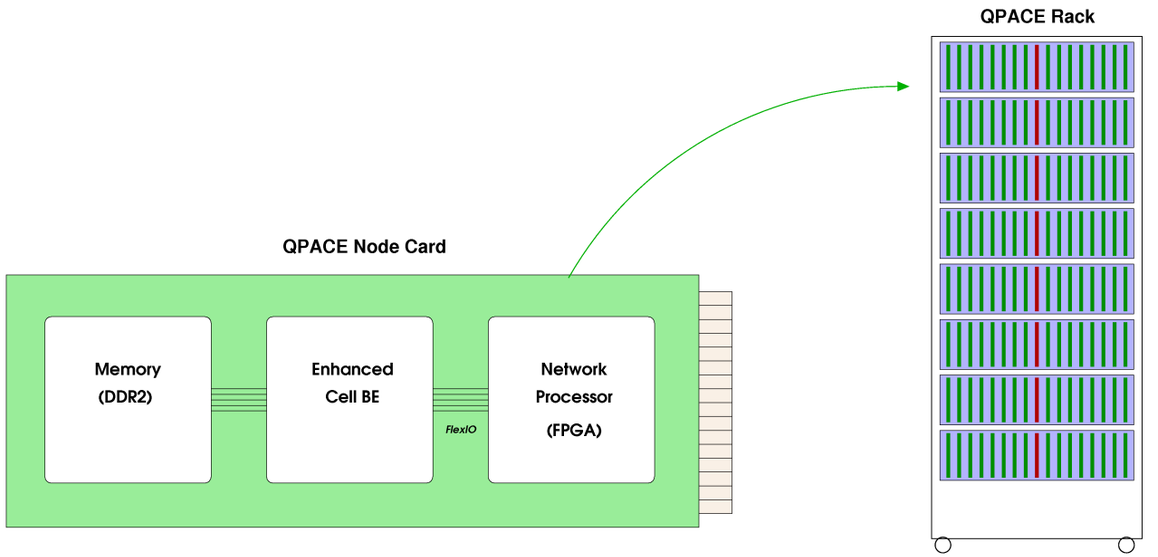


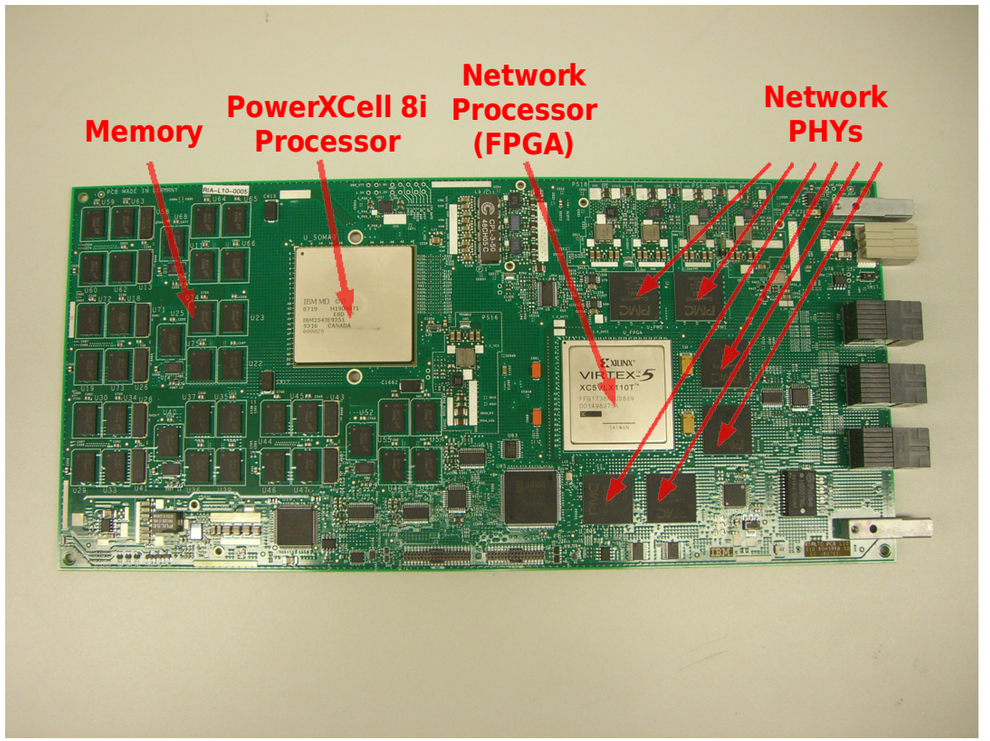
SPE:

- In-order execution
- 4 MulAdd/clk = 25.6 Gflops (SP)
- Local Store (LS) 256 k
- Data transport memory ↔ LS can (and must) be controlled by SW



Cell/QPACE

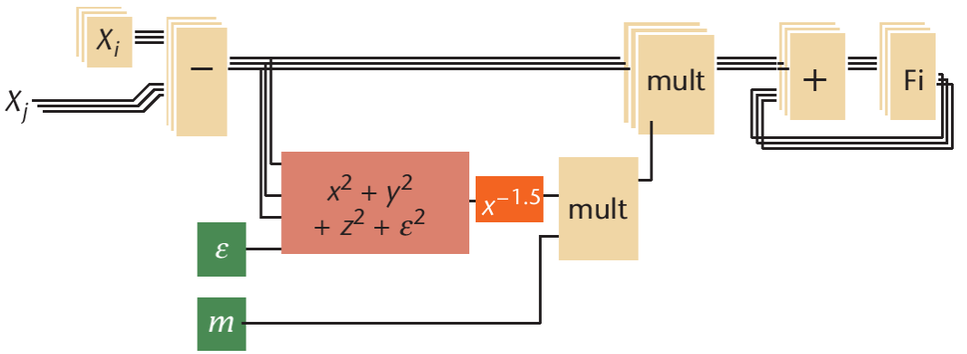

QPACE Rack
QPACE Node Card
Memory
(DDR2)
Enhanced
Cell BE
FlexIO
Network
Processor
(FPGA)


Memory
PowerXCell 8i
Processor
Network
Processor
(FPGA)
Network
PHYs

Example: GRAPE

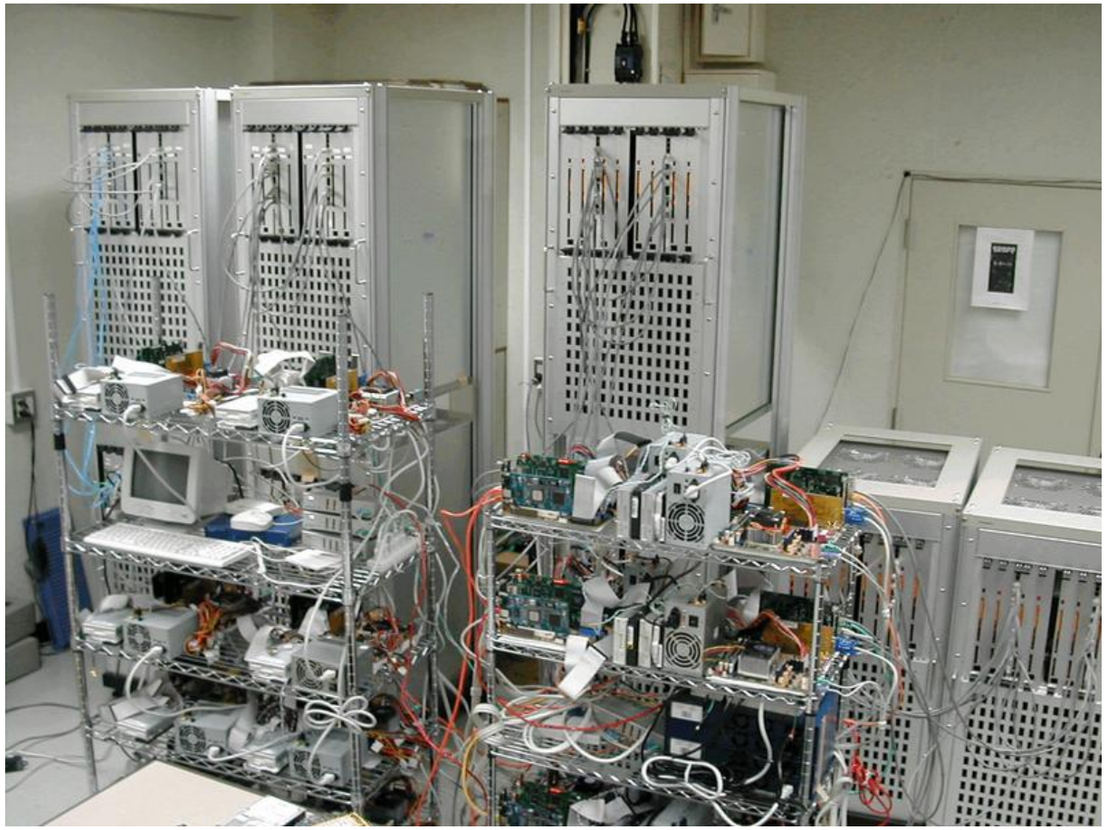
History:

1989	Idea by physicists at Tokyo University to implement N-body computation in HW
	GRAPE-1 prototype
1991	GRAPE-1A, GRAPE-2
	discrete commercial chips, 40 MFlops
1993	GRAPE-3
	first custom LSI chip
1996	MD-GRAPE
1997	GRAPE-4
	19 flop/3 $\times$ 32 MHz = 203 Mflops
2000	GRAPE-5
2001	Gordon Bell Prize for GRAPE-6 prototype
2003	GRAPE-6
	6 pipelines $\times$ 57 flop $\times$ 90 MHz = 30 Gflops
2005	MD-GRAPE-3
	20 pipelines $\times$ 36 flop $\times$ 250 MHz = 180 Gflops



N.B.: Odd series have low accuracy, even series have high accuracy

GRAPE-6



- Pipeline = 60 arithmetic units
- FPGA = 6 pipelines
- PCI module (123 Gflops) = 4 chips + FPGA + SRAM
- Mother board (1 Tflops) = 8 modules
- Full system (64 Tflops) = 64 mother boards

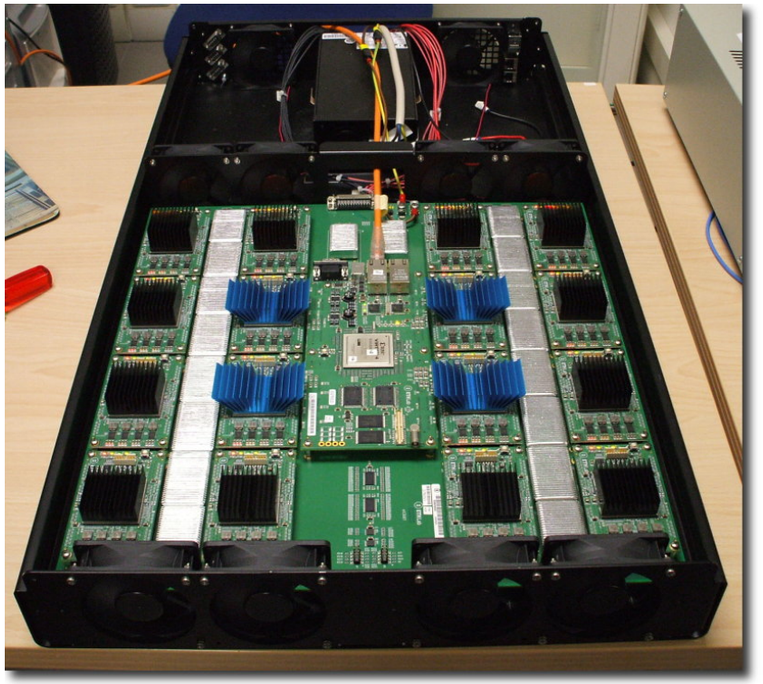
Example: IANUS

History:

- 1991: RTN with transputers (Zaragoza)
- 2000: SUE "Spin Update Engine" with FPGA (Zaragoza)
  $\longrightarrow$ 217 ps/update on full machine
- 2006: IANUS "Spin Update Engine" with FPGA (Zaragoza + Ferrara)
  $\longrightarrow$ 1 ps/update on one PB

Architecture:

- FPGA (62.5 MHz)
  - 512 update engines (RNG+LUT)
  - on-chip memory
  - links for 2-d network
- PB = 16 FPGA (4×4)



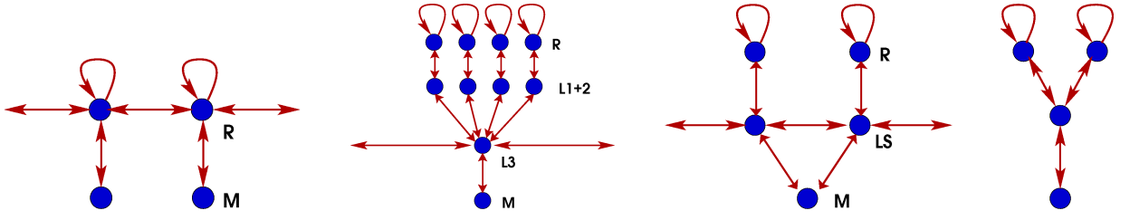
Simulations of Edward-Anderson Spin Glas

$$H = -\sum_{\langle i,j \rangle} J_{ij} \cdot x_i \cdot x_j \quad (J_{ij} = \pm 1 \text{ random})$$

allows 10^{11} MC steps on $L^3 = 80^3$ with $\Delta t = 10^{-12}$ sec $\rightarrow$ 0.1 sec

Comparison of Machines used for LQCD

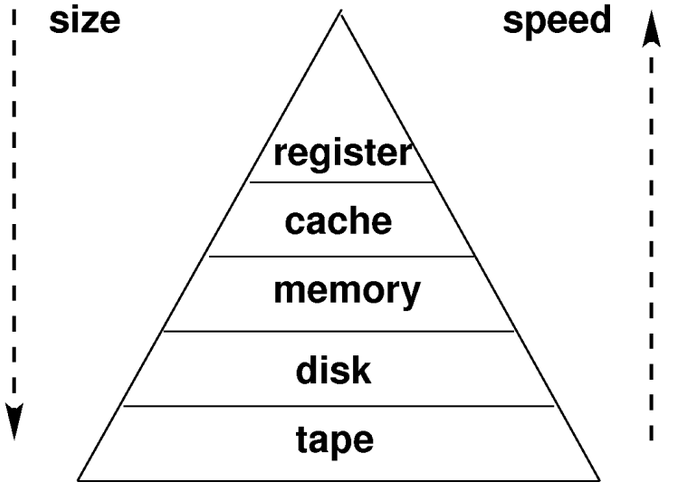
	unit	apeNEXT 2006	BG/P 2008	Cell 2009	PC 2009
Arithmetics					
f_{clk}	[GHz]	0.13	0.85	3.2	2.8
FP word	[bits]	64	64	64 (32)	64 (32)
FP/core	[flop/clk]	8	4	4	2×2
core/chip		1	4	8	4
FP/chip	[Gflops]	1	13.6	100	50
power (overall)	[W/Gflops]	9	3	1.5	2.5
Cache		—	L3	LS	L3
size	[word]	—	1 M	8×32 K	1 M
Memory					
bandwidth	[word/flop]	1/4	1/8	1/32	1/12.5
latency	[clk]	≈ 20	≥ 30	≥ 200	≥ 100
Network		3d	3d + tree	3d	—
bandwidth	[word/flop]	1/24	1/64	$< 1/16$	—
latency	[clk]	≈ 40	≈ 700	≈ 3000	—



Simplified Hardware Model

Distinction between devices/units for:

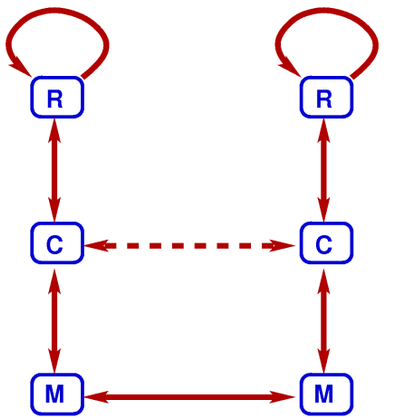
- control (of data and program flow)



- storage of data (and code)
  - memory
  - cache(s)
  - registers
  - internal buffers, fifos, flip-flops, ...

  σ_x = storage size

- processing/transport of data
  - arithmetic pipelines
  - storage access (hopefully pipelined)
  - combinatorical logics
  - buses?



β_{xy} = bandwidth (data throughput/time)
λ_{xy} = latency (delay between input and first output)
ISA = instruction set architecture

Implementation of Computational Tasks

Three inter-related **problems:**

(1) translation of the computational task into hardware operations

(2) allocation of the hardware resources for data storage and transport

(3) scheduling of the operations

N.B.: (2) and (3) are NP-hard

. . . need to be tackled at various abstraction **levels:**

- development or selection of algorithm
- development of a high-level code
- code generation by the compiler
  (code selection, register allocation, and instruction scheduling)
- out-of-order execution by hardware (micro-instructions)

DAG of computational task:		Model of hardware architecture:
• vertices: operations	$\leftrightarrow$	• edges: transport devices
• edges: variables	$\leftrightarrow$	• vertices: storage devices