Challenges and Potentials of Emerging Multicore Architectures

U. Rüde (LSS Erlangen, ruede@cs.fau.de)

joint work with

J. Götz, M. Stürmer, K. Iglberger, S. Donath, C. Feichtinger,

T. Gradl, C. Freundl, H. Köstler, T. Pohl

G. Wellein, G. Hager, T. Zeiser (RRZE)

N. Thürey (ETH Zürich)

Lehrstuhl für Informatik 10 (Systemsimulation)

Universität Erlangen-Nürnberg

www10.informatik.uni-erlangen.de

LRZ Garching, 4.12.2007

Example OMP-parallel Flow Animation

- Resolution: 880*880*336; 260M cells, 6.5M active on average

Overview

- Introduction:

  Riding on Multi Core towards PetaScale and Beyond
- Example multi core architectures
- Case study: Lattice Boltzmann Methods for Computational Haemodynamics on the Play Station
- Conclusions

Part I

Towards PetaScale and Beyond

How much is a PetaFlops?

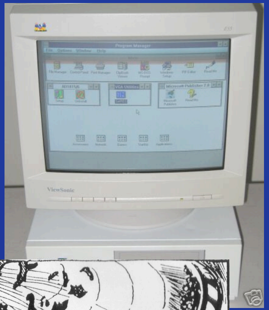
- 10^6 = 1 MegaFlops: Intel 486

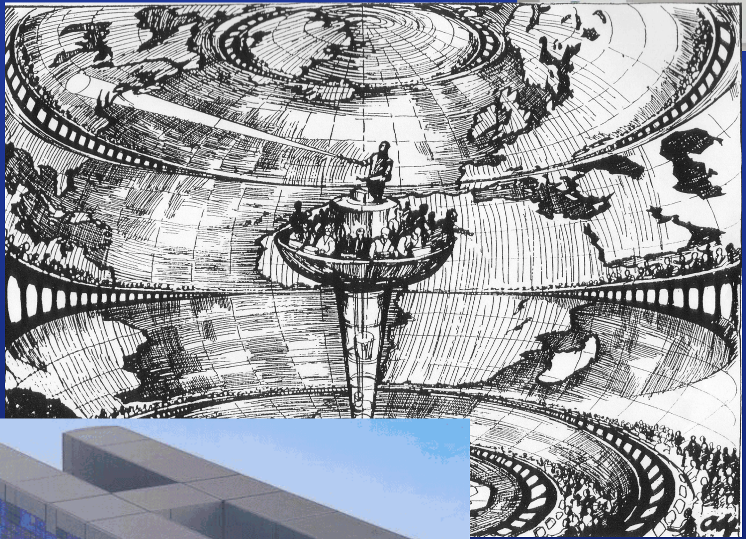
  33MHz PC (~1989)
- 10^9 = 1 GigaFlops: Intel Pentium III

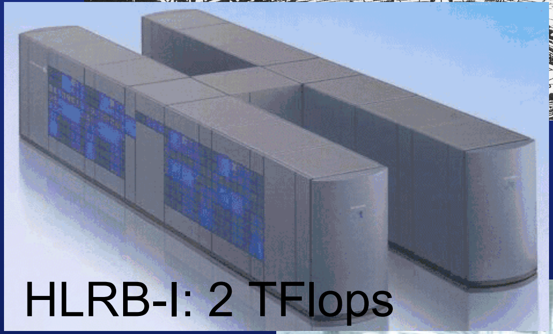
  1GHz (~2000)
  - If every person on earth does one operation every 6 seconds, all humans together have 1 GigaFlops performance (less than a current laptop from Aldi)
- 10^{12}= 1 TeraFlops: HLRB-I

  1344 Proc., ~ 2000
- 10^{15}= 1 PetaFlops

  >250 000 Proc. Cores?, ~2008?
  - If every person on earth runs a 486 PC, we all together have an aggregate Performance of 6 PetaFlops.






HLRB-I: 2 TFlops

HLRB-II: 63 TFlops

So What?

Example application:

- Flow with moving objects on the nano-scale (suspensions)
- 1μm lattice resolution, object size 5-10 μm (example: blood cells)

We can simulate (with this resolution)

- on a laptop: 2 Million lattice cells = 0.002 mm^3 of liquid.
- on a TOP-10 supercomputer: 80 Billion lattice cells = 80 mm^3= 0.08 ml of liquid
    - for blood additionally required: 400 Million blood cells
- on a (peak) peta scale system: 2 Trillion lattice cells = 2 cm^3 = 2 ml of liquid
    - ~ 10 Billion blood cells
- on an ten peta flop system: 20 Trillion lattice cells = 20 ml of liquid

Is this good for anything?

Evolution of Semiconductor Technolgy

ITRS International Technology Roadmap for Semiconductors

- Collects trends of semiconductor technology
- Not easy to read, but excellent source of information
- See http://www.itrs.net/reports.html

Evolution of Chip Technology

- From the semiconductor roadmap (old version, around 2002)

	2001	2003	2005	2008	2011	2014
nm	130	110	90	60	40	30
SRAM: M Tr/cm^2	70	128	234	577	1432	3510
Logic: M Tr/cm^2	13	24	44	109	265	664
M Tr/ Chip	122	244	488	1381	3907	11052
On Chip/local GHz	2.1	2.9	4.1	7.1	11.1	14.1
Chip-Board GHz	1.4	1.7	2	2.6	3.2	3.8
Number IO/chip	3000	3000	3200	3400	3800	4000

- From the semiconductor roadmap (2005 version/ 2006 update)

	2008	2011	2014	2020
nm	57	40	28	14
M Tr/ Chip	1106	2212	4424	17696
On Chip/local GHz	10	17	28	73
Chip-Board GHz	6	11	23	72
Number pins/chip	4400	5094	5896	7902

Evolution of Chip Technology

- From the semiconductor roadmap (2005 version/ 2006 update)

	2008	2011	2014	2020
nm	57	40	28	14
M Tr/ Chip	1106	2212	4424	17696
On Chip/local GHz	10	17	28	73
Chip-Board GHz	6	11	23	72
Number pins/chip	4400	5094	5896	7902

Where does Computer Architecture Go?

- Computer architects have capitulated: It may not be possible anymore to exploit progress in semiconductor technology for *automatic* performance improvements
  - Even today a single core CPU is a highly parallel system
    - using superscalar execution
    - using complex pipelines
    - ... and additional tricks
  - Internal parallelism is a major reason for the performance increases until now:
    - Parallelism in the architecture and transparent to the user, but:
    - for further improvements, a conventionally written program does (usually) not contain enough additional parallelism
    - There is a limited amount of parallelism that can be exploited automatically
- Multi-core systems concede the architects´ defeat:
  - We cannot exploit additional transistors for better performance without application programmer help
  - Architects fail to build faster single core CPUs given more transistors
  - Clock rate increases slow down due to power considerations
- Therefore architects have started to put several cores on a chip for
  - programmers to use them directly

What are the consequences for us all?

- For the application developers “*the free lunch is over*”
  - we have used (internally parallel) CPUs, often without being aware of the parallelism
  - From now on, it is up to us, whether we can use additional performance through parallel execution - even on a single chip
  - Without explicitly parallel algorithms, the performance potential cannot be used any more
  - Architects and compiler writers are (of course) still needed to help with better languages and systems to make parallel computing less painful
- For HPC
  - CPUs will have 2, 4, 8, 16, ..., 128, ..., ??? cores - maybe sooner than we are ready for this
  - Large scale systems will be built from 1000s of such CPUs
  - We will have to deal with systems with millions of cores
  - How many of us have used more than a few hundred at this time?

Part II

Example multicore architectures

UltraSPARC T2: Server on a Chip – Alpha hardware

8 SPARC V9 cores @ 1.2–1.4GHz

- 8 threads per core
- 2 execution pipelines per core
- 1 instruction/cycle per pipeline
- 1 FPU per core
- 1 SPU (crypto) per core
- 4 MB, 16-way, 8-bank L2$

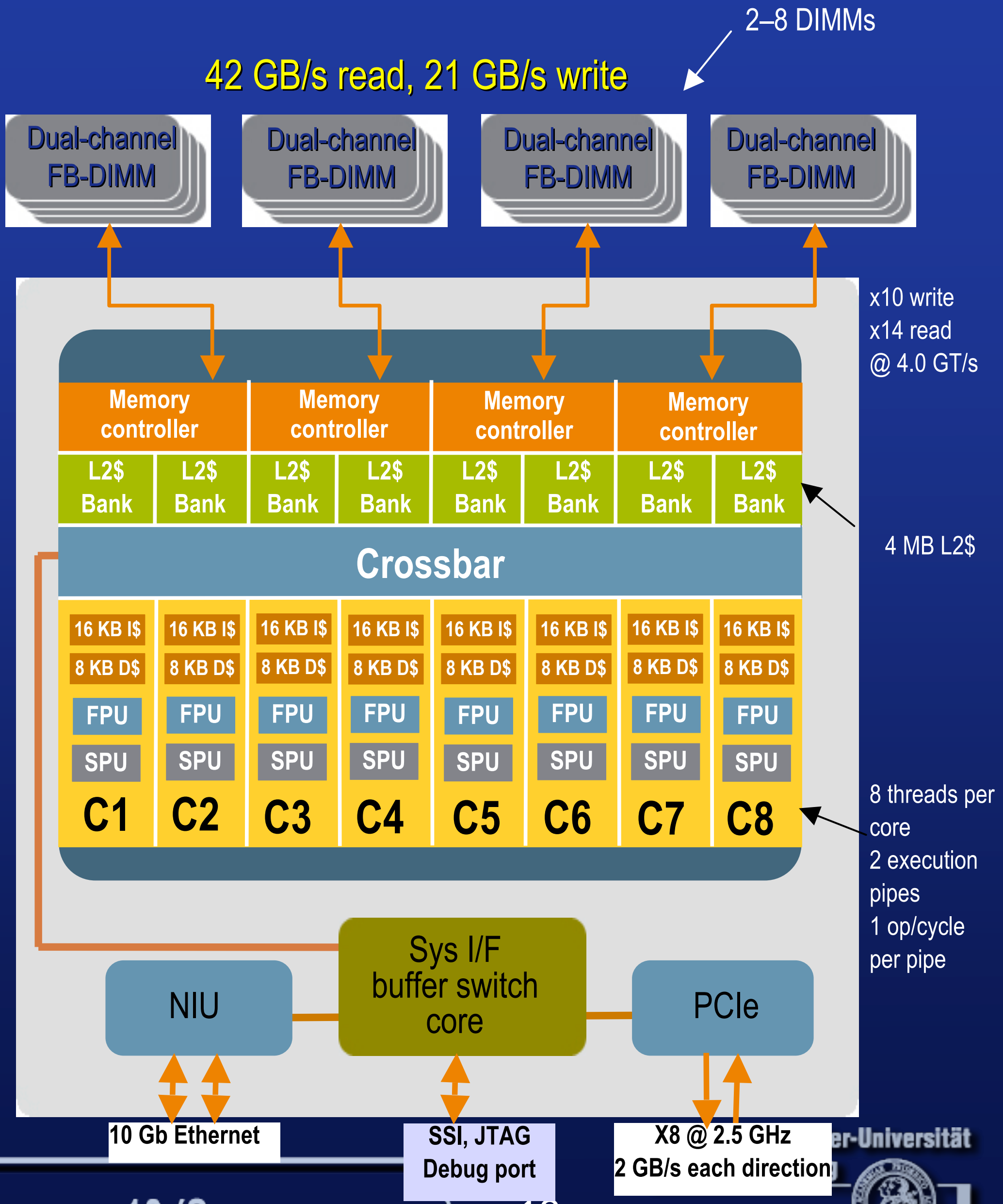
4 FB-DIMM DRAM controllers

2.5 GHz x 8 PCI-Express interface

2 x 10 Gb on-chip Ethernet

Die size: 342mm^2 Power: < 95 W

LBM: *Single chip can achieve the same performance level as a 4-socket DC Opteron system.*



LBM Benchmark kernel Performance – Basics

- Performance unit: Mega Lattice Site Updates per Second (MLUPS)
- Per Lattice Site update we need:
  - ~ 170-250 Floating Point Ops. -> 5 MLUPS ~ 1 GFlop/s
  - ~ 40 double (8 byte) transfers between CPU & Memory
- Estimate max. performance (200 Flops per Lattice Site)

Performance Comparison

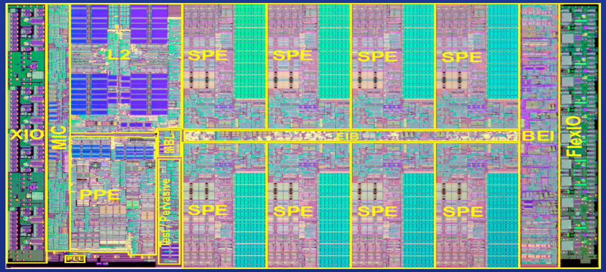
- LBM test kernel
- Same code run on Woodcrest, Opteron & Niagra2
- 3D model (D3Q19) & double precision accuracy
  - Niagara: 15 – 25 MLUPs
- Additional testing on Nigara2 for
  - optimal OMP scheduling strategy &
  - optimal number of OMP threads (8,…,64)

Kernel Benchmark

	Peak Perf. [GFlop/s]	Max. MLUPS	Bandwidth [GByte/s]	Max. MLUPS	Measure 128^3
Intel Xeon	**6.8**	34	**5.3**	11.8	5.1 **(43%)**
AMD Opteron	**4.4**	22	**6.4**	14.0	5.1 **(36%)**
Intel Itanium2	**5.6**	28	**6.4**	14.0	8.5 **(61%)**
CRAY X1	**12.8**	64	**34.1**	112	34.9 **(55 %)**
NEC SX6+	**9.0**	45	**36.0**	118	41.3 **(92 %)**

Examples of even less-standard hardware: IBM Cell Processor & nVIDIA Graphics Card

IBM Cell processor (1 chip)



nVIDIA GTX 8800

D3Q19 & single precision: 98 MLUPs

<u>Double precision: ~45-50 MLUPs</u>

D2Q9 & single precision: 330 MLUPs

<u>Double precision: ~60 MLUPs</u>

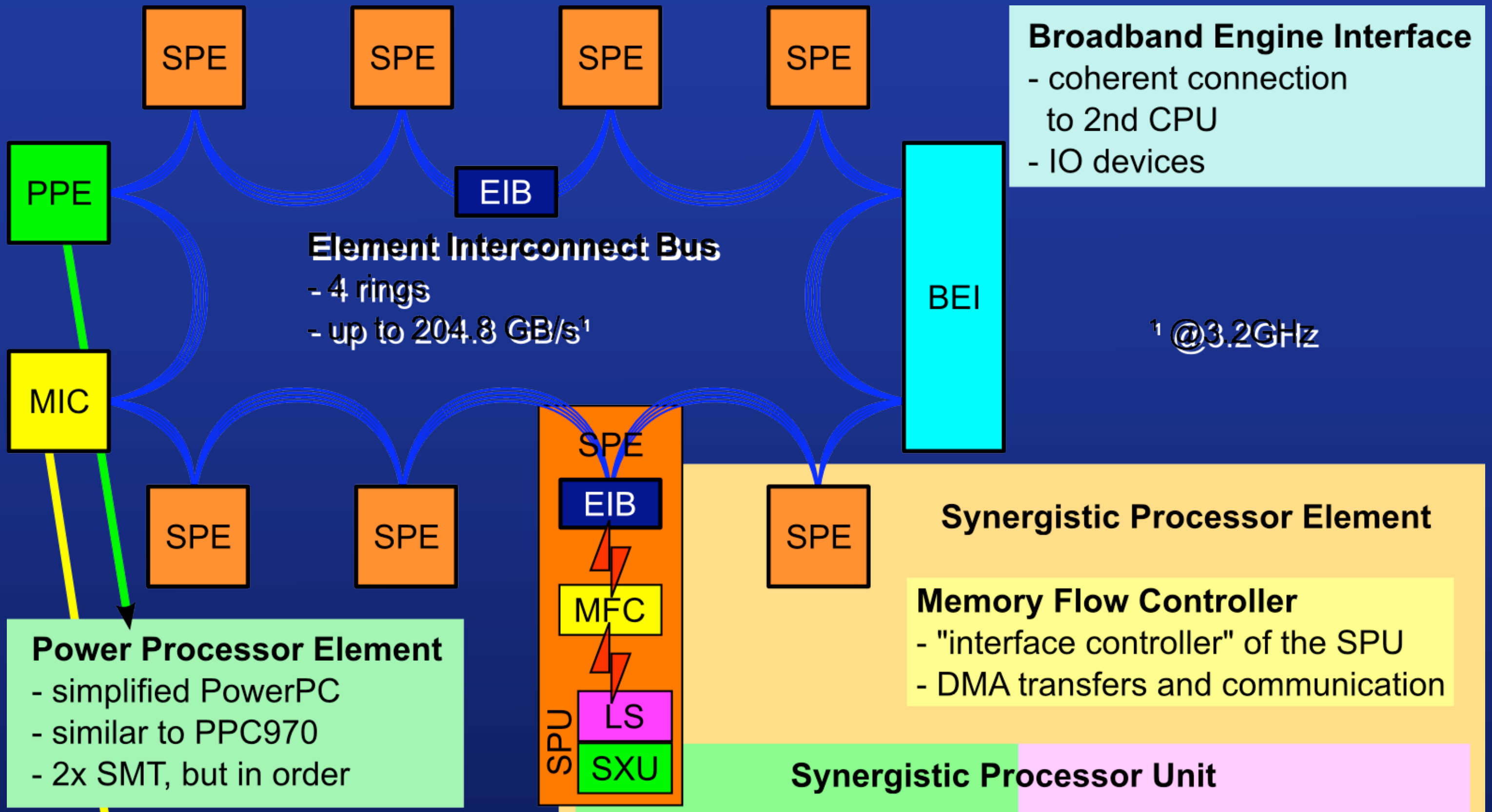
By courtesy of Jonas Tölke, Manfred Krafczyk
Comp. Modeling in Civil Eng., TU Braunschweig

Implementation & Optimization may cost weeks!

The STI Cell Processor

- hybrid multicore processor based on IBM Power architecture
- (simplified) PowerPC core
  - runs operating system
  - controls execution of programs
- multiple co-processors (8, on Sony PS3 only 6 available)
  - operate on fast, private on-chip memory
  - optimized for computation
  - DMA controller copies data from/to main memory
    - multi-buffering can hide main memory latencies completely for streaming-like applications
    - loading local copies has low and known latencies
- memory with multiple channels and links can be exploited if many memory transactions are in-flight

The STI Cell Processor



Broadband Engine Interface
- coherent connection to 2nd CPU
- IO devices

Element Interconnect Bus
- 4 rings
- up to 204.8 GB/s[1]

[1] @3.2GHz

Synergistic Processor Element

Memory Flow Controller
- "interface controller" of the SPU
- DMA transfers and communication

Power Processor Element
- simplified PowerPC
- similar to PPC970
- 2x SMT, but in order

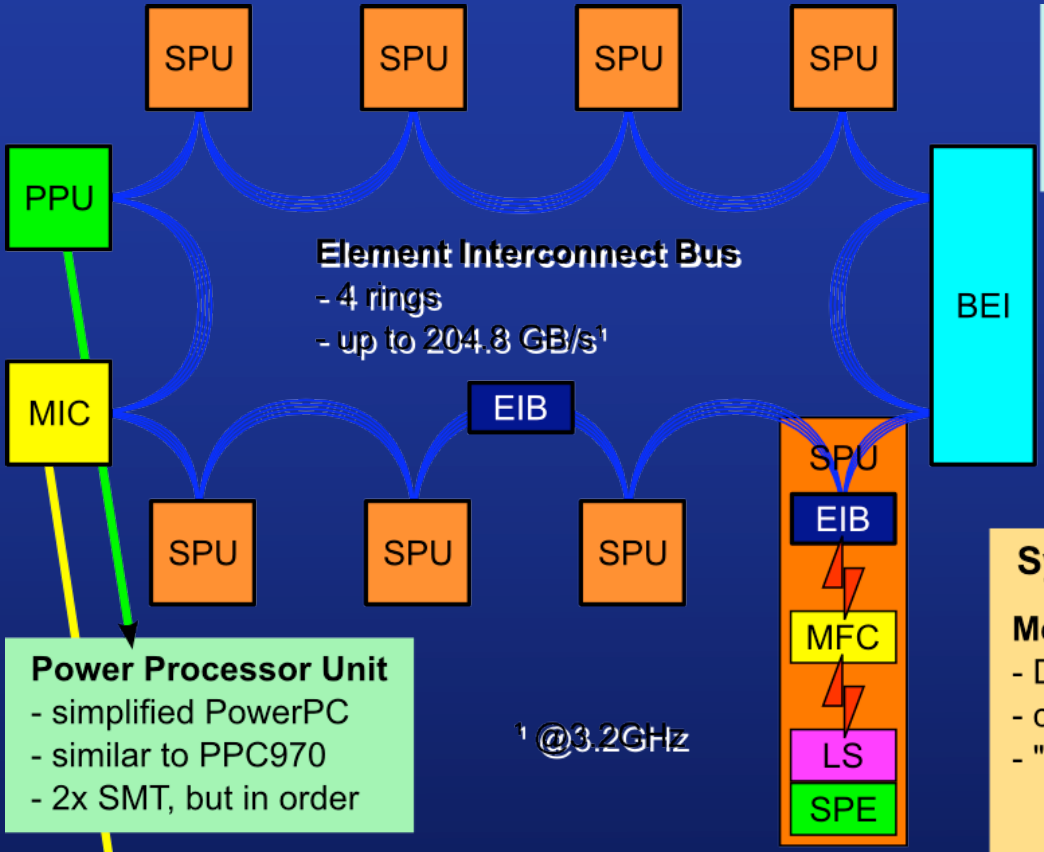
Synergistic Processor Unit

Synergistic eXecution Unit
- SIMD vector processor
- 25.6 GFlop/s
- no direct connection to EIB
- two pipelines

Local Store
- 256 kB "SXU main memory"
- 16 B/s bandwidth
- 128 B/s if all 8 banks accessed concurrently

Memory Interface Controller
- Rambus eXtreme Data Rate
- 2 channels, 8 links each
- 25.6 GB/s[1] bandwidth

Cell Architecture: 9 cores on a chip



Broadband Engine Interface
- coherent connection to 2nd CPU
- IO devices

Power Processor Unit
- simplified PowerPC
- similar to PPC970
- 2x SMT, but in order

Memory Interface Controller
- Rambus eXtreme Data Rate
- 2 channels
- 25.6 GB/s[1] bandwidth

Synergistic Processor Unit

Memory Flow Controller
- DMA transfer into and out of LS
- channel interface
- "interface controller" of the SPU

Synergistic Processor Element
- 128 SIMD registers (128 bit)
- two SIMD pipelines
- 204.8 GFlop/s[1] max. for 8 SPEs

Local Store
- "main memory" of SPE
- 256 kB only
- 16 B/s bandwidth
- 128 B/s if all 8 banks accessed concurrently

Part III

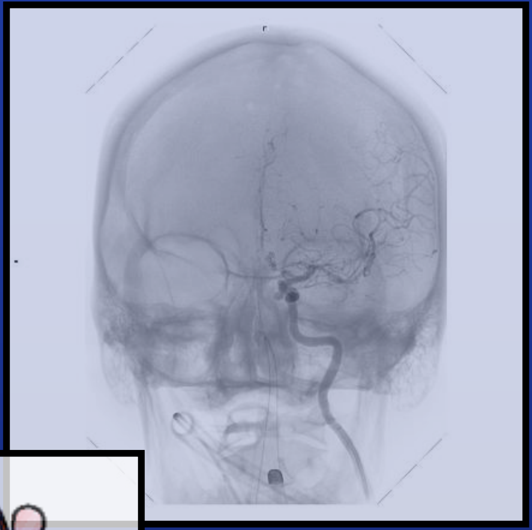
Case study: Lattice Boltzmann Methods for Computational Haemodynamics on the Play Station

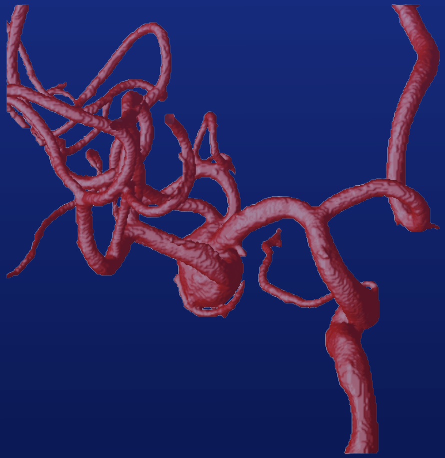
Aneurysms

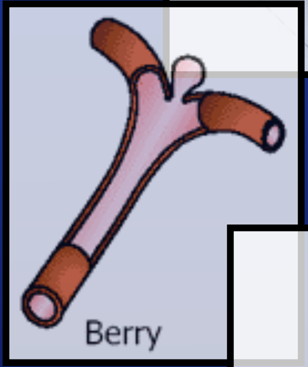
- Motivation
  - Unruptured aneurysms are a major public health issue in every developed nation
  - The flow situation could be crucial for further treatment of the patient
- Goals
  - Help to understand the development of aneurysms
  - To support therapy planning
- Challenges
  - Current imaging techniques result in data sets of 512^3 and more
  - Long runtimes on standard PCs and workstations
  - For intra-surgery planning the algorithm should perform quasi real-time

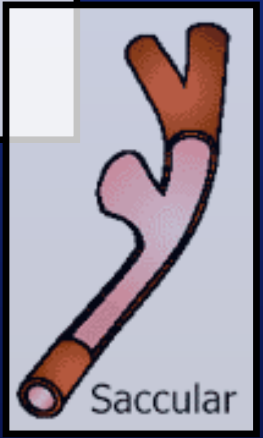
Aneurysms

- Aneurysm are local dilatations of the blood vessels
- Localized mostly at large arteries in soft tissue (e.g. aorta, brain vessels)
- Can be diagnosed by modern imaging techniques (e.g. MRT,DSA)
- Can be treated e.g by clipping or coiling



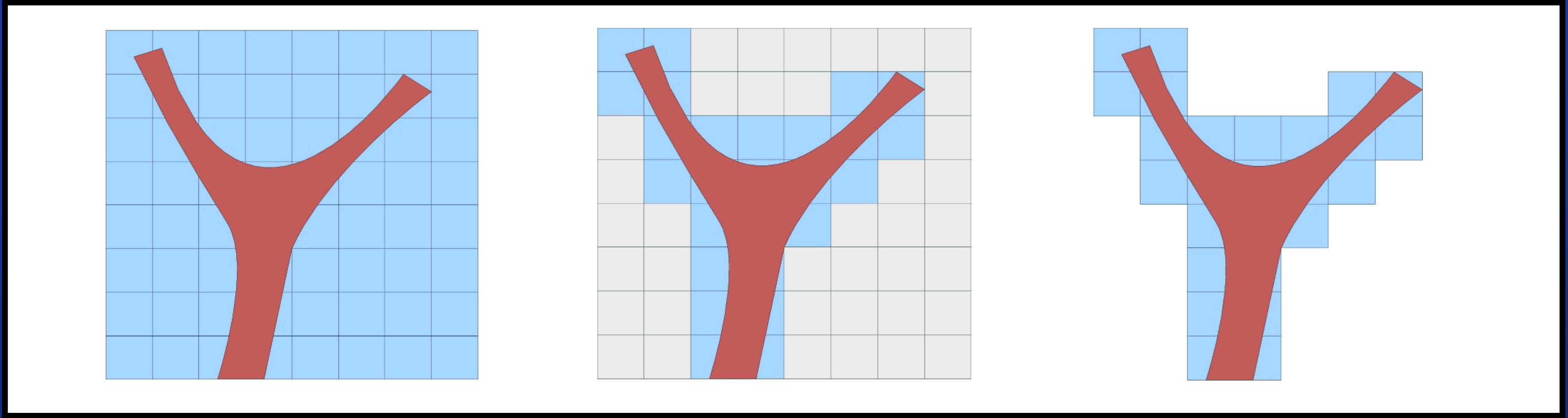






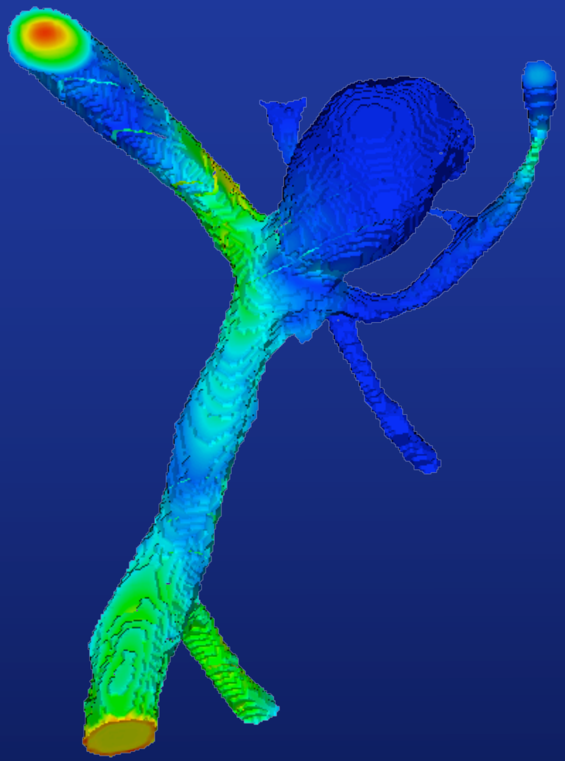
A data structure for simulating flow in blood vessels

- In a brain geometry only about 3-10% of the nodes are fluidal

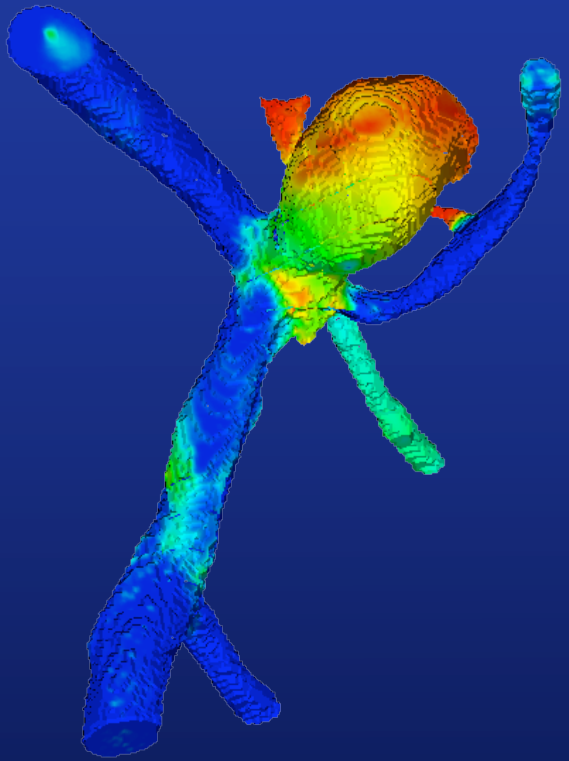


- We use a domain decoupling in equally sized blocks, so-called patches and only allocate patches containing fluid cells
- The memory requirements and the computational time could be reduced significantly
- For the Cell processor we use patches of size 8x8x8, fitting into the SPU local memory

Results

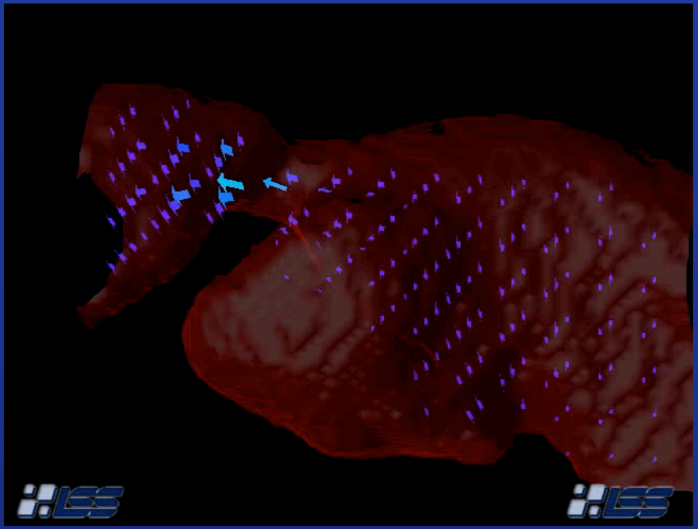


Velocity near the wall in an aneurysm



Oscillatory shear stress near the wall in an aneurysm

Pulsating Blood Flow in an Aneurysm



Collaboration between:

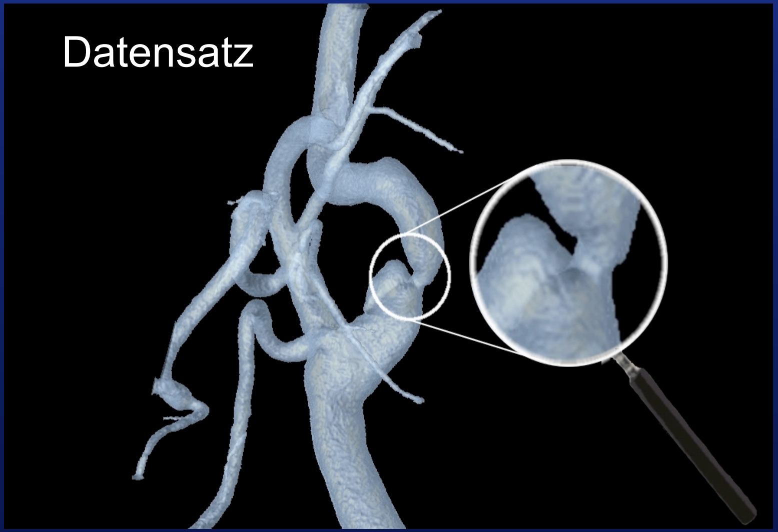
Neuro-Radiology (Prof. Dörfler, Dr. Richter)

Computer Science

Simulation

Imaging

CFD

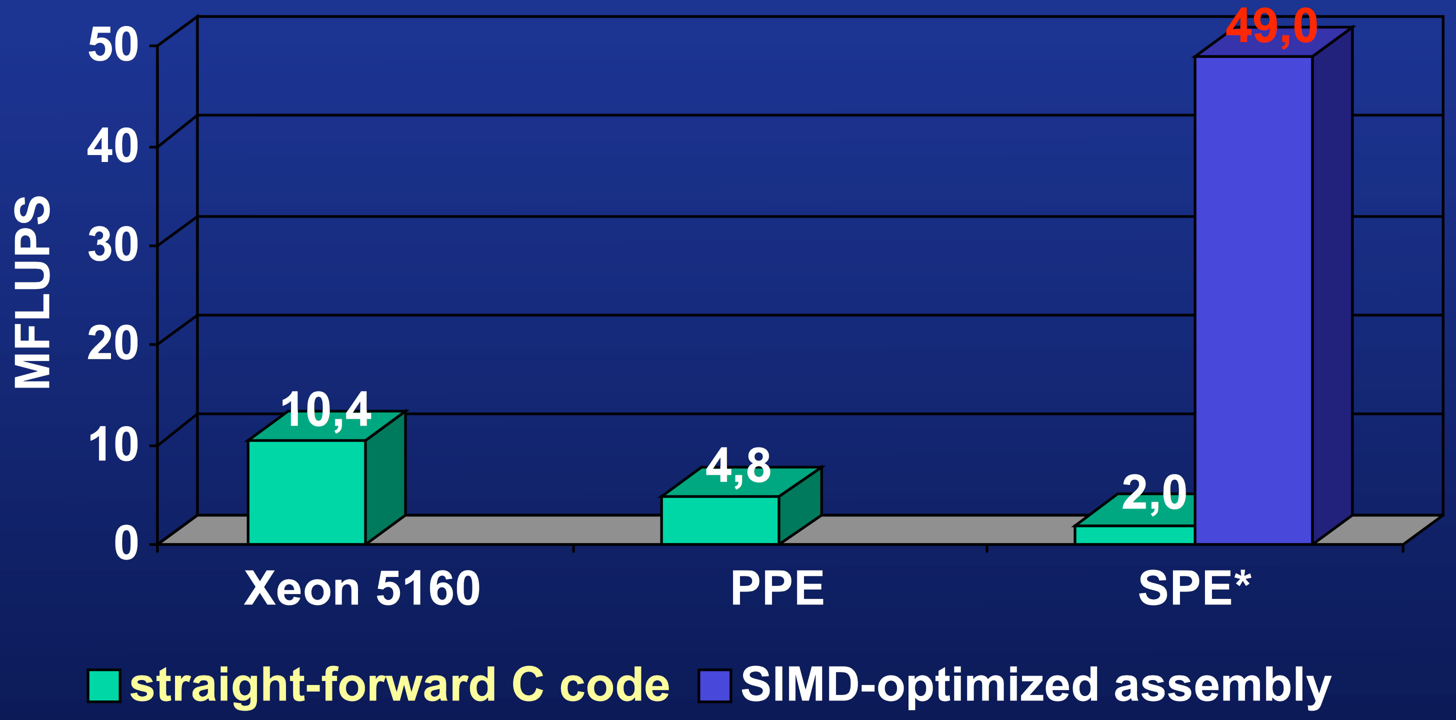


LBM Optimized for Cell

- memory layout
  - optimized for DMA transfers
  - information propagating between patches is reordered on the SPE and stored sequentially in memory for simple and fast exchange
- code optimization
  - kernels hand-optimized in assembly code
  - SIMD-vectorized streaming and collision
  - branch-free handling of bounce-back boundary conditions

Performance Results

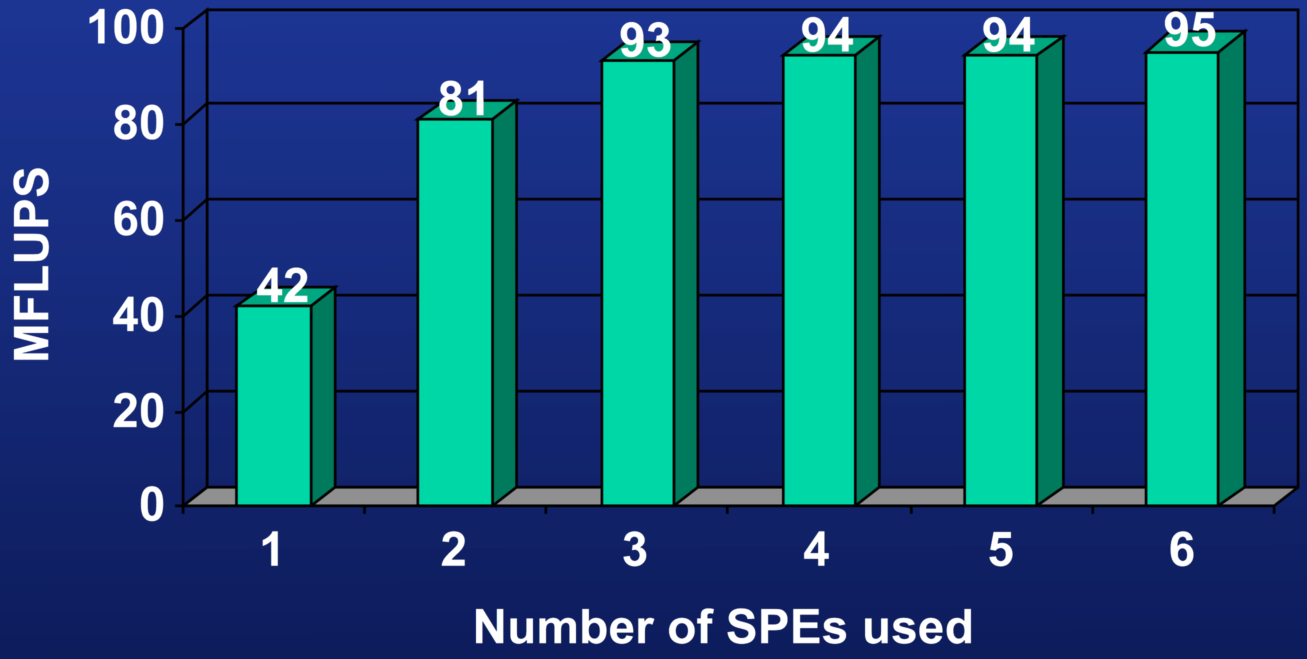
LBM performance on a single core
(8x8x8 channel flow)



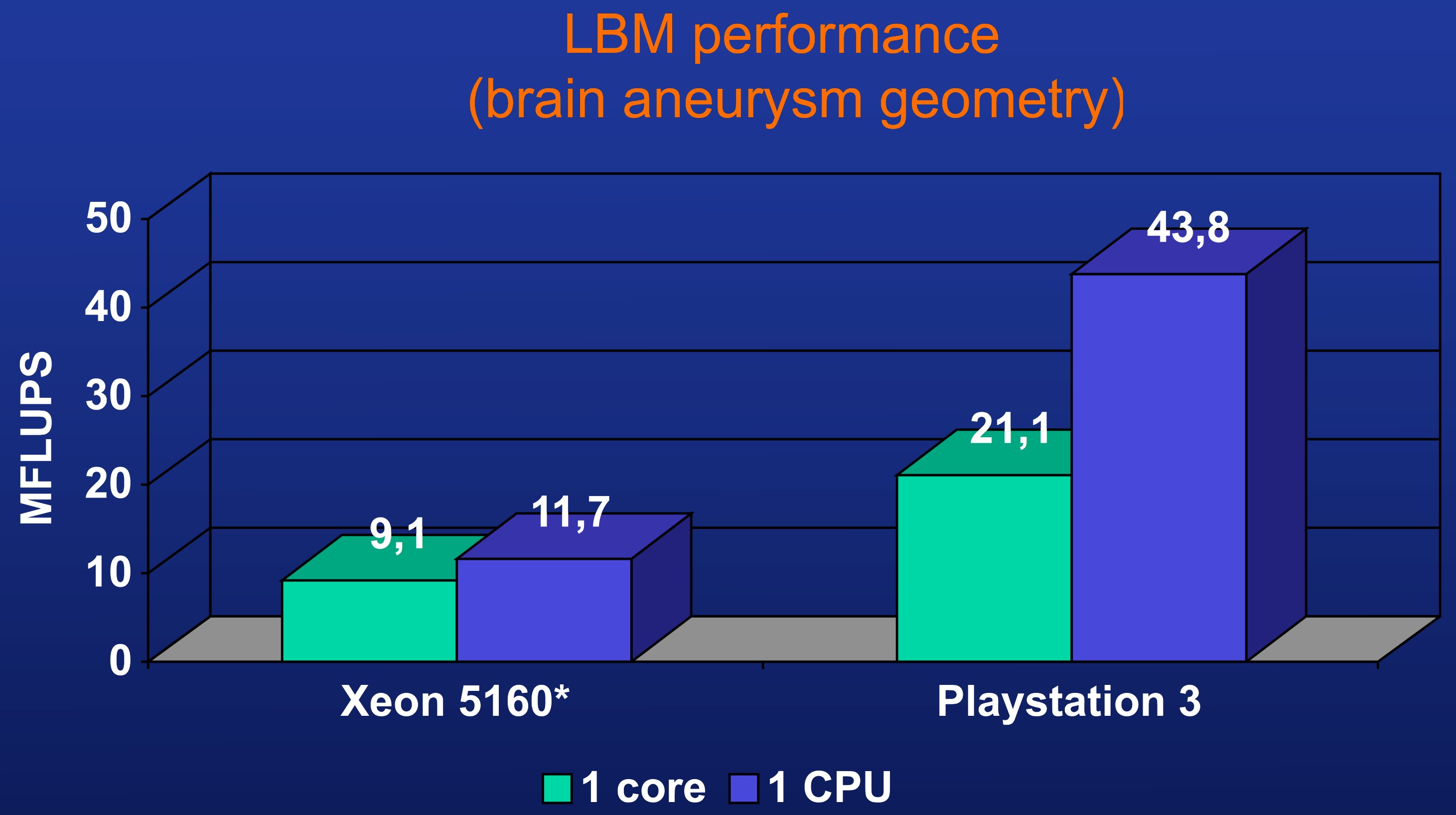
*on Local Store without DMA transfers

Performance Results

Playstation 3 LBM performance
(94x94x94 channel flow)



Performance Results



*performance optimized code by LB-DC

Part IV

Conclusions

What have we learned?

- The future is parallel on multi core CPUs
- Memory bandwidth per core will be a severe bottleneck
  - “inverse Moore’s law”
- Programming current leading edge multi-core architectures to exploit their performance potential requires expert knowledge of the architecture
  - better tool and system support needed
  - complexity of the architecture

Related Talks

- 11:00 - 11:30 K. Iglberger:

  Large Scale Lattice Boltzmann Simulations: The need for supercomputers

- 14:45 - 15:15 T. Gradl:

  Scalable Parallel Multigrid

  „how to solve FE systems with 300 000 000 000 unknowns on 9000 processos of HLRB-II“

  see also the article in the current inSiDE

Talk is Over

Questions?