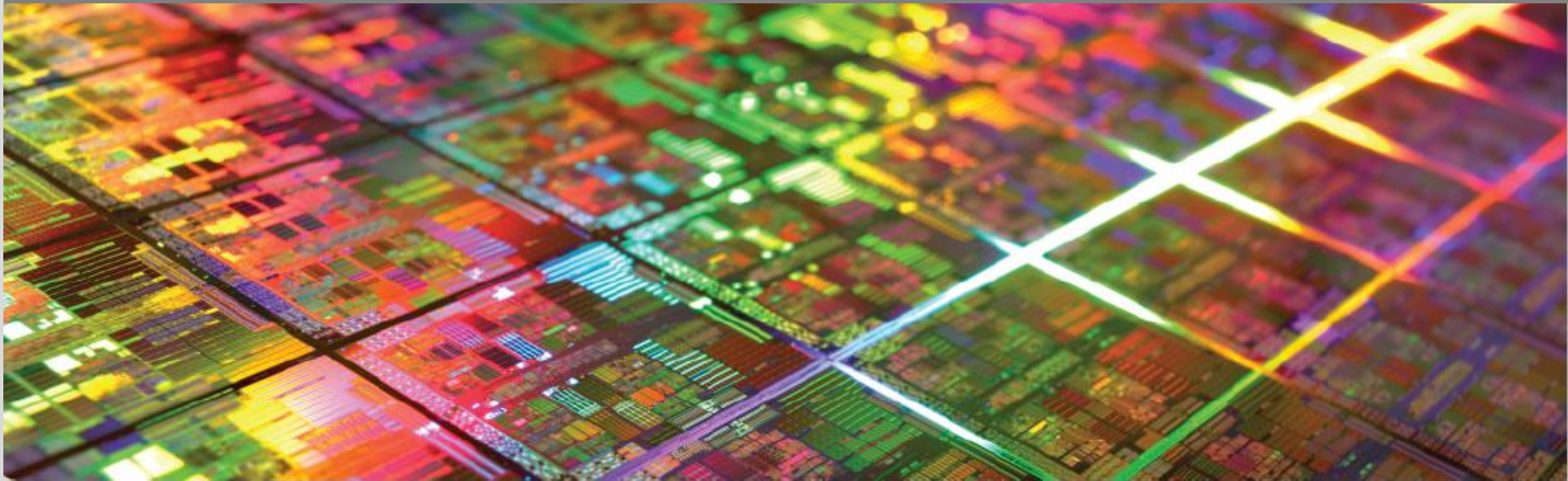


Rechnerstrukturen

Vorlesung im Sommersemester 2016

Prof. Dr. Wolfgang Karl

Institut für Technische Informatik (ITEC), Lehrstuhl für Rechnerarchitektur und Parallelverarbeitung



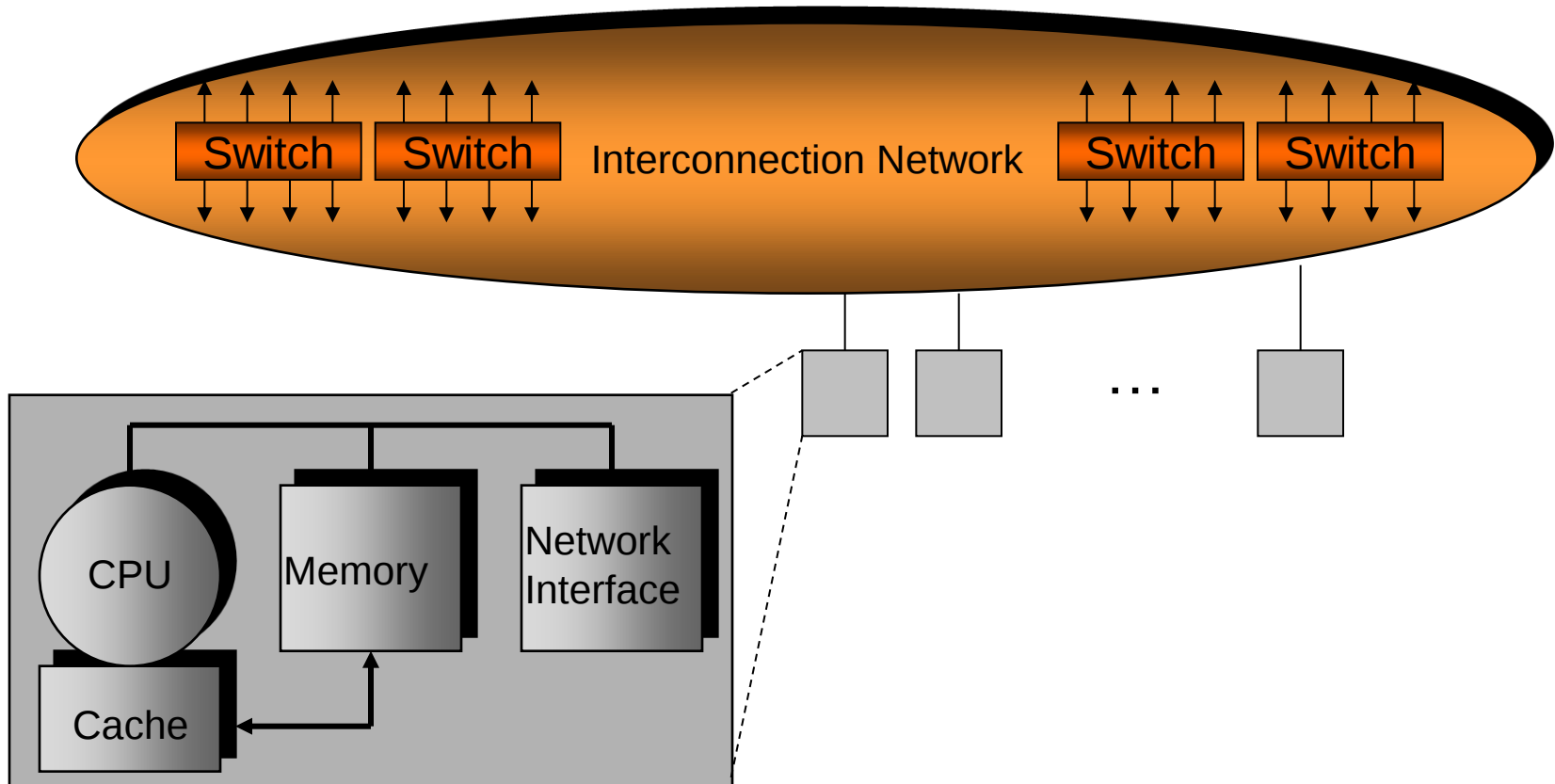
Vorlesung Rechnerstrukturen

Kapitel 3: Multiprozessoren – Parallelismus auf Prozess-/Blockebene

- 3.1 Motivation
- 3.2 Allgemeine Grundlagen
- 3.3 Parallele Programmierung
- 3.4 Quantitative Maßzahlen
- 3.5 Verbindungsstrukturen
- 3.6 Multiprozessoren mit gemeinsamem Speicher
- 3.7 Multiprozessoren mit verteiltem Speicher

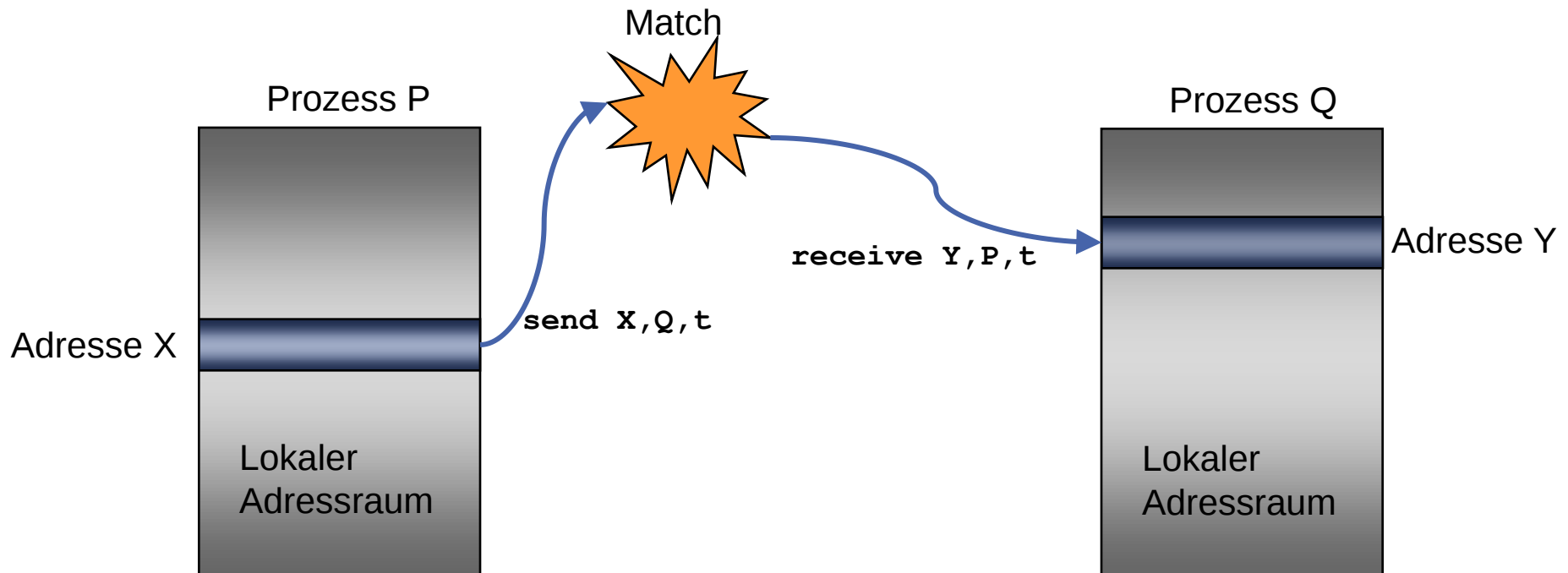
Multiprozessor mit verteiltem Speicher

■ Allgemeine Rechnerorganisation



Multiprozessor mit verteiltem Speicher

- Programmiermodell
- Nachrichtenorientiertes Programmiermodell (Message Passing)



Multiprozessor mit verteiltem Speicher

Nachrichtenorientiertes Programmiermodell (Message Passing)

■ Message-passing Primitive

■ Synchrones Message-Passing

Code für Prozess P1

```
A=10;  
SEND (&A, sizeof(A), P2, SEND_A);  
A=A+1;  
RECV (&C, sizeof(C), P2, SEND_B);  
printf(C)
```

Code für Prozess P2

```
B=5;  
RECV (&B, sizeof(B), P2, SEND_A);  
B=B+1;  
SEND (&B, sizeof(B), P1, SEND_B);
```

- Sender blockiert, bis die Nachricht beim Empfänger angekommen ist. Ebenso muss der Empfänger blockieren, bis die Nachricht empfangen worden ist und in den gewünschten lokalen Speicher kopiert worden ist
 - A=10 wird an den Empfänger P2 geschickt
 - P2 muss blockieren, bis der Inhalt der Nachricht (10) in die lokale Variable B kopiert worden ist.
 - Welchen Wert druckt P1 aus?

Multiprozessor mit verteiltem Speicher

Nachrichtenorientiertes Programmiermodell (Message Passing)

■ Message-passing Primitive

■ Synchrones Message-Passing: Deadlock-Gefahr!!!!

Code für Prozess P1

```
A=10;  
SEND (&A, sizeof(A), P2, SEND_A);  
RECV (&B, sizeof(B), P2, SEND_B);
```

Code für Prozess P2

```
B=5;  
SEND (&B, sizeof(B), P1, SEND_B);  
RECV (&A, sizeof(B), P1, SEND_A);
```

■ Synchroner Nachrichtentransfer:

- Der Sender und der Empfänger blockieren, bis die Nachrichtenübertragung beendet ist, d.h. bis ein korrespondierendes SEND/RECV Paar ausgeführt ist
- P1 und P2 sind blockiert bei ihren jeweiligen SEND-Primitiven, da beide auf ihr korrespondierendes RECV warten.

Multiprozessor mit verteiltem Speicher

Nachrichtenorientiertes Programmiermodell (Message Passing)

■ Message-passing Primitive

- Synchrones Message-Passing:
 - Kombination von Synchronisation und Kommunikation in einer Primitive
 - Kann zu Leistungsverlust führen!

Multiprozessor mit verteiltem Speicher

Nachrichtenorientiertes Programmiermodell (Message Passing)

■ Message-passing Primitive

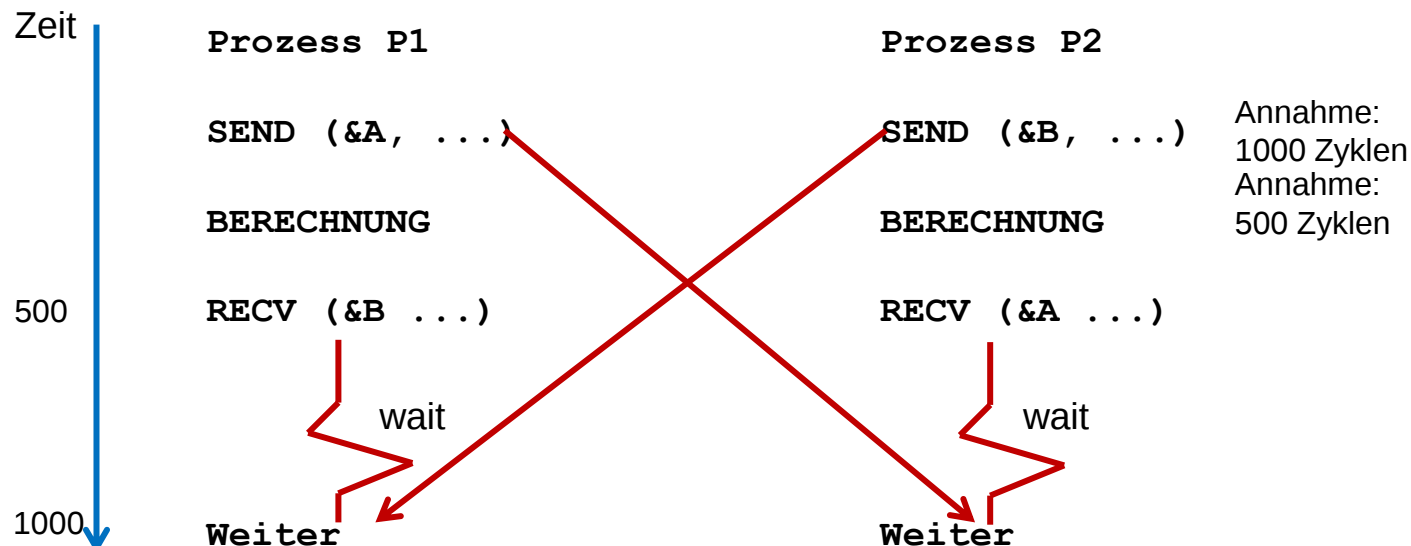
- Asynchrones Message-Passing: Überlappung von Kommunikation und Berechnung

Code für Prozess P1

```
A=10;
SEND (&A, sizeof(A), P2, SEND_A);
...BERECHNUNG
RCV (&B, sizeof(B), P2, SEND_B);
```

Code für Prozess P2

```
B=5;
SEND (&B, sizeof(B), P1, SEND_B);
...BERECHNUNG
RCV (&A, sizeof(B), P1, SEND_A);
```



Multiprozessor mit verteiltem Speicher

Nachrichtenorientiertes Programmiermodell (Message Passing)

■ Message-passing Primitive

■ Asynchrones Message-Passing:

■ Blockierendes asynchrones SEND:

- Gibt die Kontrolle an den Sende-Prozess zurück, wenn die zu versendenden Daten in einem Puffer kopiert worden sind und nicht mehr durch die Berechnung verändert werden können

■ Blockierendes asynchrones RECV:

- Gibt ebenso die Kontrolle nicht an den Empfangsprozess zurück, bis die Nachricht in einen lokalen Adressraum gespeichert worden ist

Multiprozessor mit verteiltem Speicher

Nachrichtenorientiertes Programmiermodell (Message Passing)

■ Message-passing Primitive

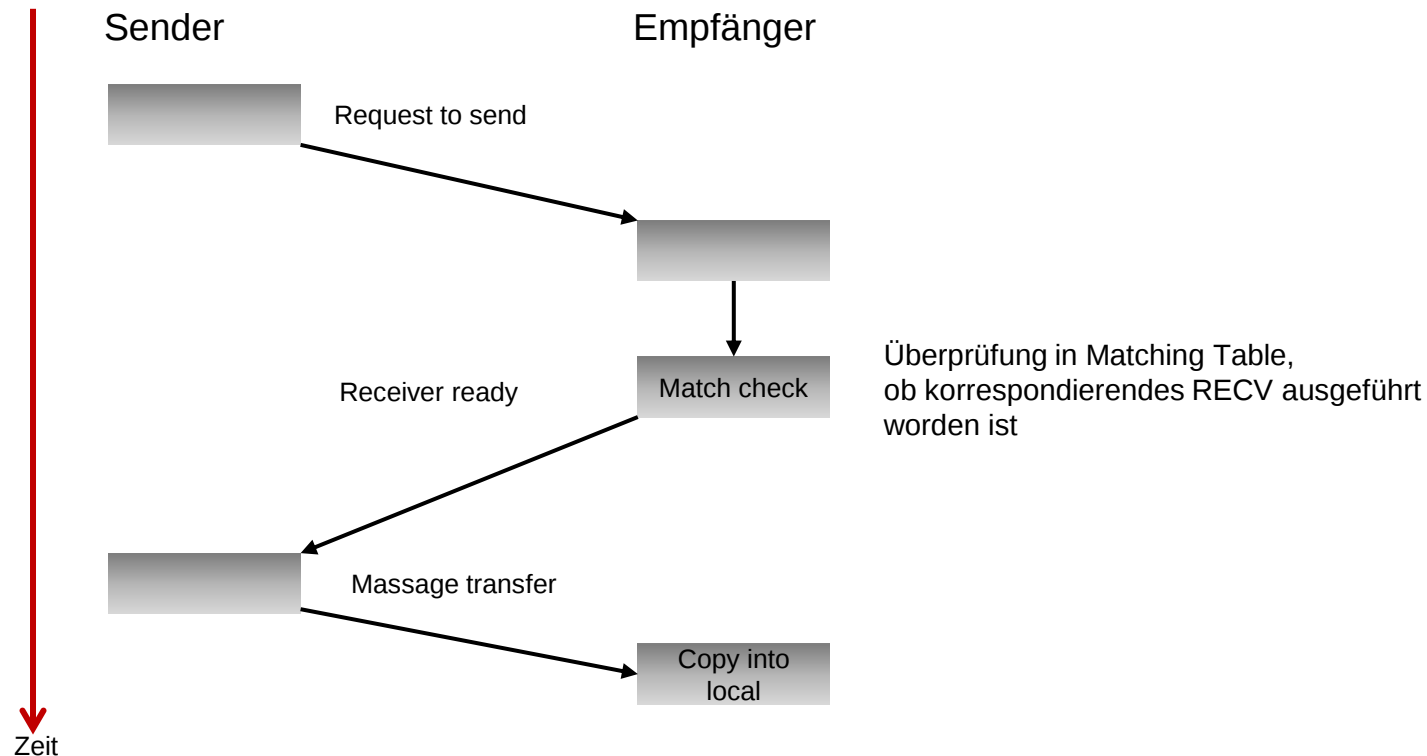
- Asynchrones Message-Passing:
 - Nichtblockierende asynchrones Kommunikationsprimitive:
 - Die Kontrolle wird sofort an den Sende- bzw. Empfangsprozess zurückgegeben und der Datentransfer läuft im Hintergrund
 - Probe-Funktionen:
 - Prüfen ob Daten vom lokalen Adressraum des Senders in einen Puffer kopiert worden sind, bzw. ob der Empfänger die Daten kopiert hat

Multiprozessor mit verteiltem Speicher

Nachrichtenorientiertes Programmiermodell (Message Passing)

■ Message-passing Protokolle

- Sender-initiiertes Message-Passing Protokoll (synchron): Drei-Phasen-Protokoll

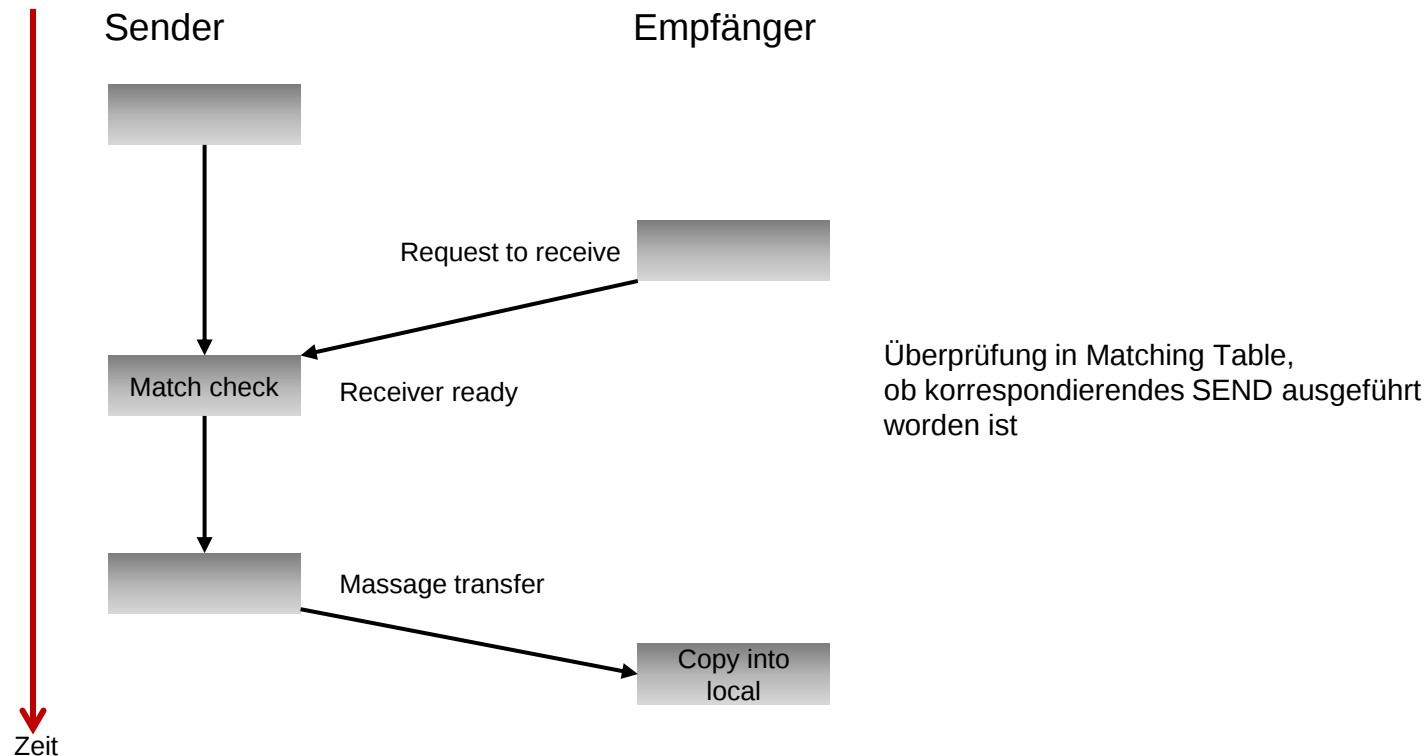


Multiprozessor mit verteiltem Speicher

Nachrichtenorientiertes Programmiermodell (Message Passing)

■ Message-passing Protokolle

- Empfänger-initiiertes Message-Passing Protokoll (synchron):



Multiprozessor mit verteiltem Speicher

Nachrichtenorientiertes Programmiermodell (Message Passing)

■ Hardware-Unterstützung für Message-passing Protokolle

- Im Allgemeinen bietet das Verbindungsnetzwerk rudimentäre Übertragungsprimitive an
- Zusätzliche Hardware-Unterstützung soll die Latenz für das Versenden einer Nachricht von einem Sende- zu einem Empfangsknoten reduzieren oder den Prozessor von der Verarbeitung des Message-Passing-Protokolls entlasten

Multiprozessor mit verteiltem Speicher

Nachrichtenorientiertes Programmiermodell (Message Passing)

■ Hardware-Unterstützung für Message-passing Protokolle

■ Software-Overhead

Erzeugerprozess:

`send(proci, processi, @sbuffer, num_bytes)`

Sender

Systemaufruf
Prüfe Schutzbed.
DMA Init.

DMA nach NI

Verbraucherprozess:

`receive(@rbuffer, max_bytes)`

Empfänger

DMA vom Netzwerk in den Puffer
BS Interrupt und Dekodierung der Nachricht
BS kopiert Systempuffer in Userpuffer
Reschedule Benutzerprozess
Lesen der Nachricht

Zeit

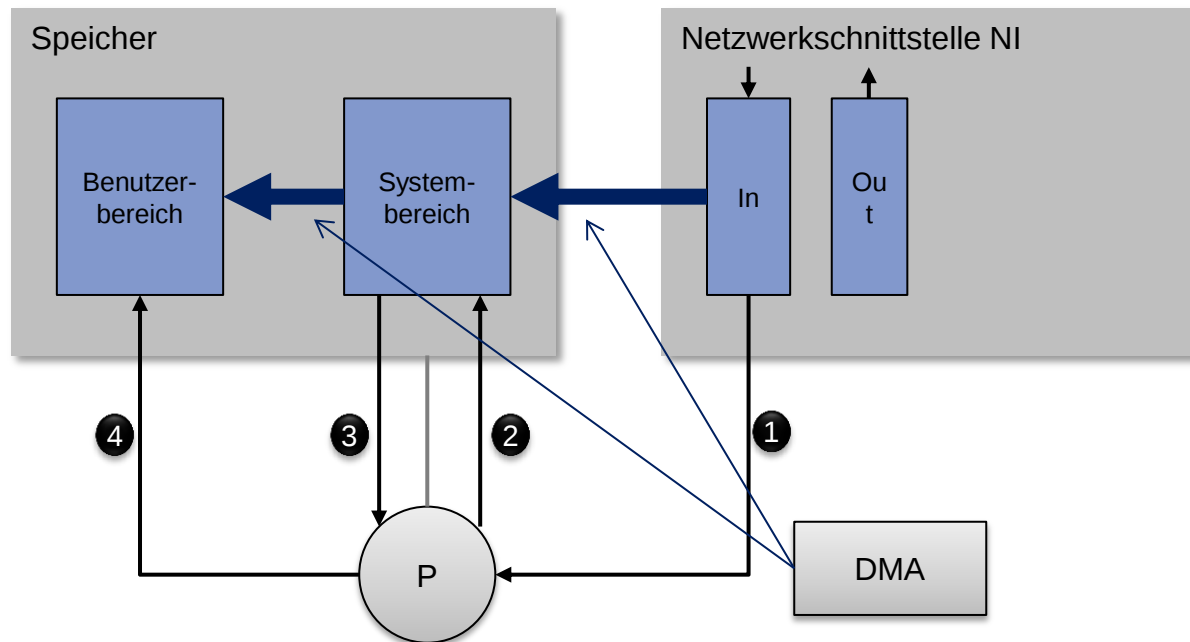
NI: Netzwerkschnittstelle
DMA: Direktspeicherzugriff

Multiprozessor mit verteiltem Speicher

Nachrichtenorientiertes Programmiermodell (Message Passing)

■ Hardware-Unterstützung für Message-passing Protokolle

- Hardware-Support für Message-Passing: DMA

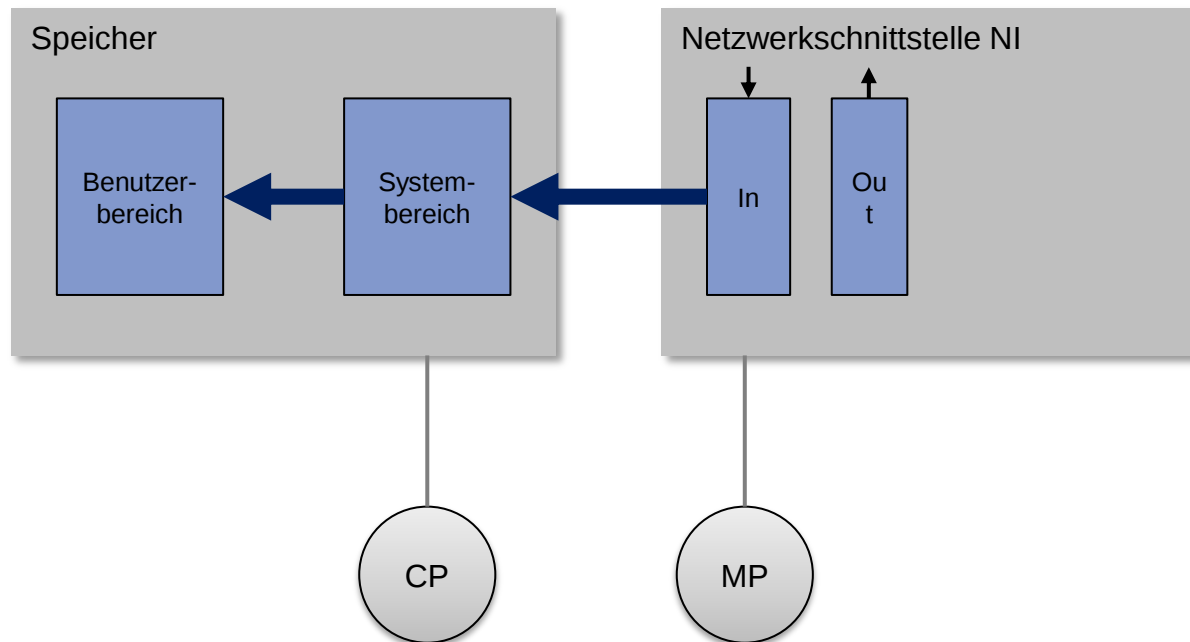


Multiprozessor mit verteiltem Speicher

Nachrichtenorientiertes Programmiermodell (Message Passing)

■ Hardware-Unterstützung für Message-passing Protokolle

- Hardware-Support für Message-Passing: Kommunikationsprozessor



Multiprozessor mit verteiltem Speicher

■ Fallstudie IBM Blue Gene/L

■ Systemkomponenten

■ 65536 Knoten

- in bis zu 64 Racks, die auch so organisiert werden können, als wären es verschiedene Systeme, wobei auf jedem ein eigenes Single Software Image läuft

■ Knoten

■ 2 BG/L Compute ASIC (BLC)

■ Dual Processor SoC ASIC

- 9 Double data rate synchronous dynamic random access memory chips (DDR SDRAM chips) pro ASIC

■ Knoten über 5 Netzwerke verbunden

- 64 x 32 x 32 3-D Torus
- Global Collective Network
- Global Barrier and Interrupt Network
- I/O Network (Gigabit Ethernet)
- Service Network

Multiprozessor mit verteiltem Speicher

■ Fallstudie JUGENE - Juelicher BlueGene/P

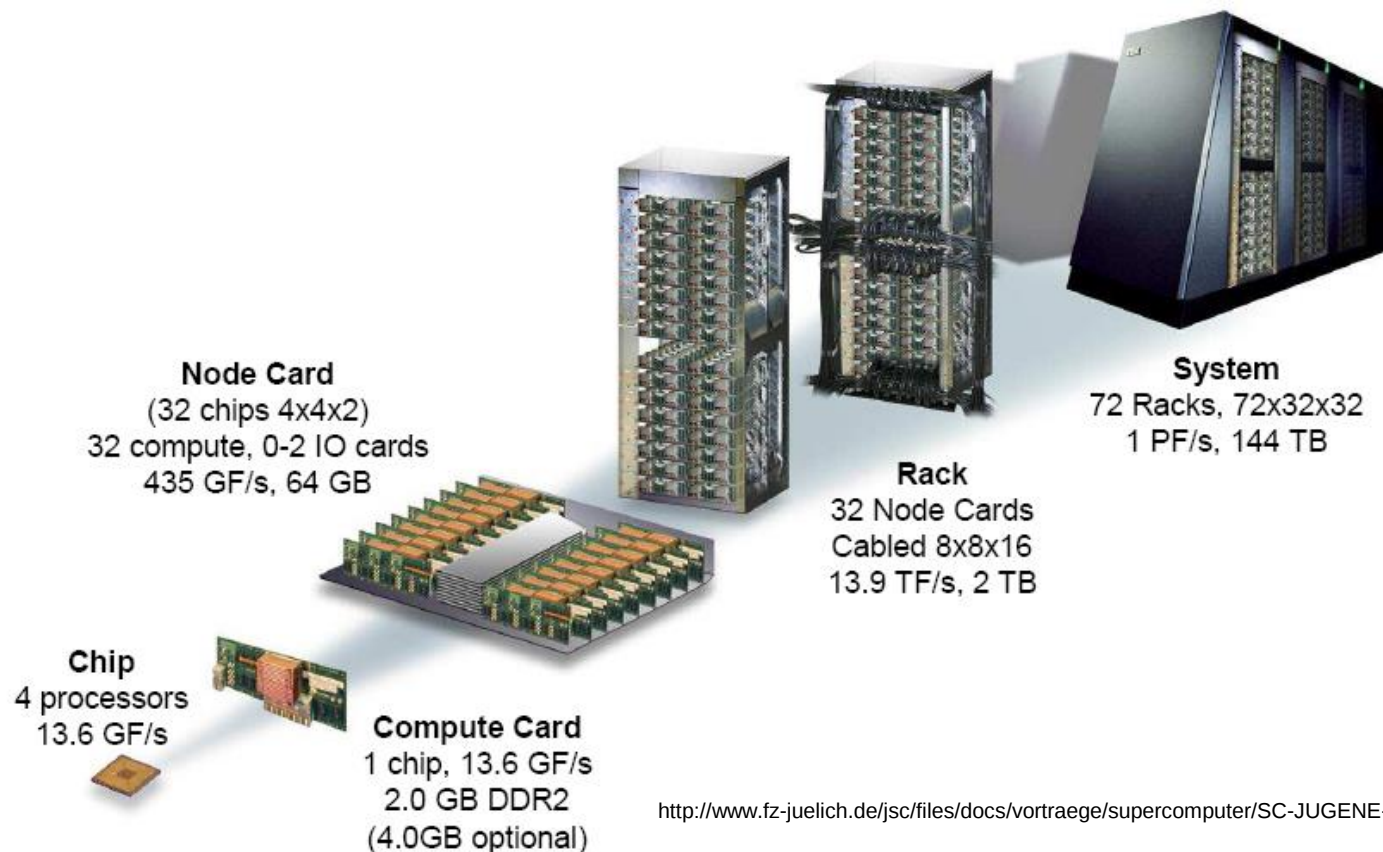
- Top500 – Juni 2010: Nr. 5
 - 825,5 TFLOPS
 - Cores: 294912



<http://www.fz-juelich.de/jsc/files/docs/vortraege/supercomputer/SC-JUGENE-Introduction.pdf>

Multiprozessor mit verteiltem Speicher

■ Fallstudie JUGENE - Juelicher BlueGene/P Systemkomponenten

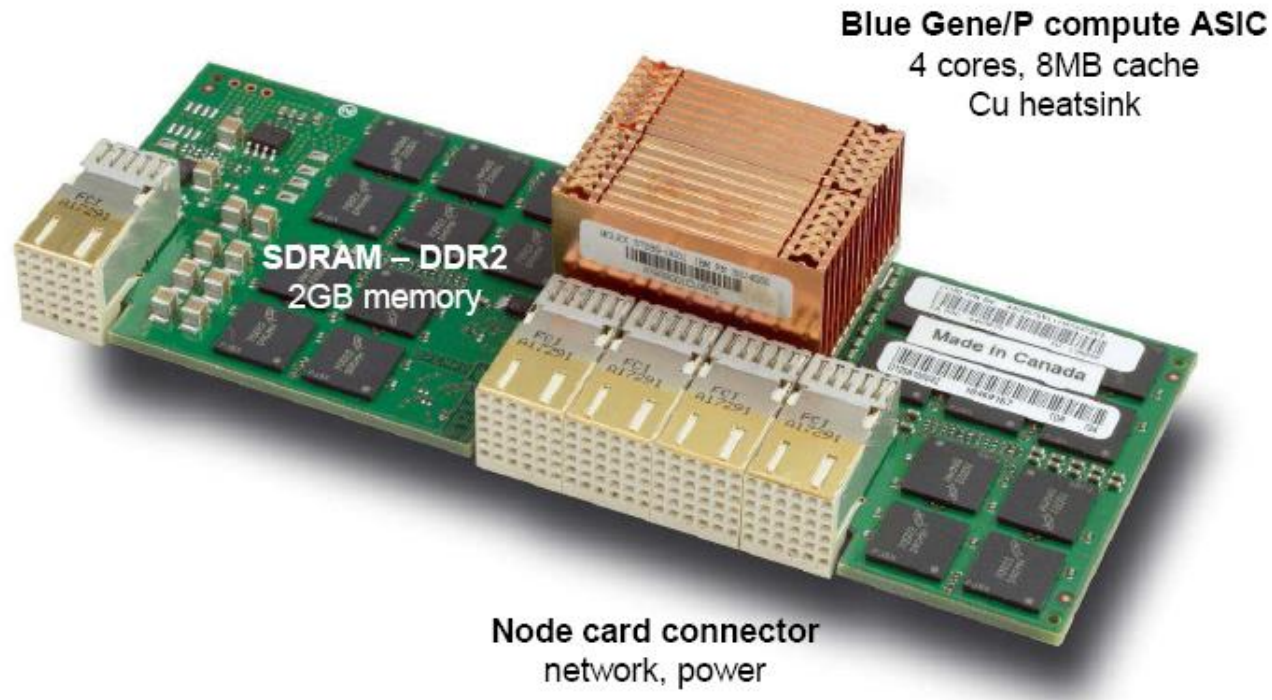


<http://www.fz-juelich.de/jsc/files/docs/vortraege/supercomputer/SC-JUGENE-Introduction.pdf>

Multiprozessor mit verteiltem Speicher

■ JUGENE - Juelicher BlueGene/P

■ Compute Card



<http://www.fz-juelich.de/jsc/files/docs/vortraege/supercomputer/SC-JUGENE-Introduction.pdf>

Multiprozessor mit verteiltem Speicher

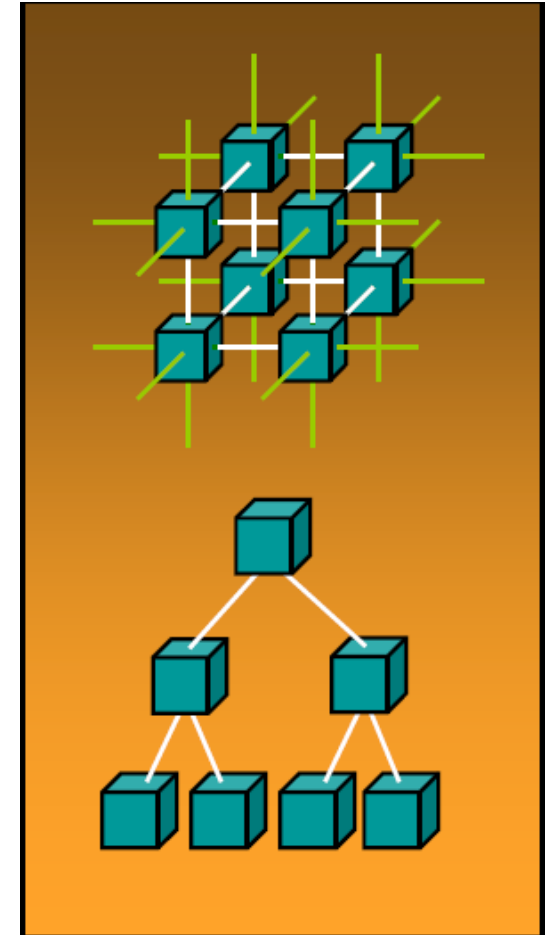
- **JUGENE - Juelicher BlueGene/P**
 - Node Card



<http://www.fz-juelich.de/jsc/files/docs/vortraege/supercomputer/SC-JUGENE-Introduction.pdf>

Multiprozessor mit verteiltem Speicher

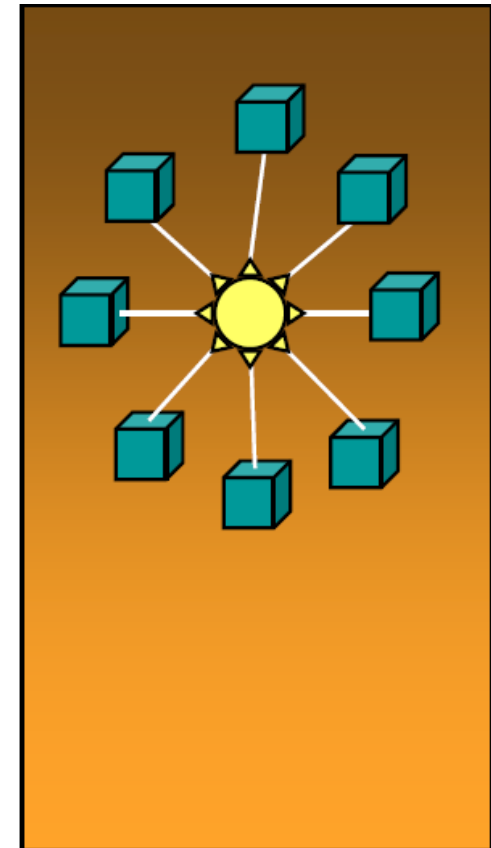
- **JUGENE - Juelicher BlueGene/P**
- **Verbindungsnetze**
- **3 Dimensional Torus**
 - Interconnects all compute nodes (73,728)
 - Virtual cut-through hardware routing
 - 425 MB/s on all 12 node links (5.1 GB/s per node)
 - Communications backbone for computations
 - 188TB/s total bandwidth
- **Collective Network**
 - One-to-all broadcast functionality
 - Reduction operations functionality
 - 850 MB/s of bandwidth per link
 - Interconnects all compute and I/O nodes



<http://www.fz-juelich.de/jsc/files/docs/vortraege/supercomputer/SC-JUGENE-Introduction.pdf>

Multiprozessor mit verteiltem Speicher

- **JUGENE - Juelicher BlueGene/P**
- **Verbindungsnetze**
- Low Latency Global Barrier and Interrupt
 - Latency of one way to reach all 72K nodes 0.65 μ s,
 - MPI 1.6 μ s
 - External I/O Network
- 10 GBit Ethernet
 - Active in the I/O nodes
 - All external comm. (file I/O, control, user interaction)
- Control Network
 - 1 GBit Ethernet
 - Boot, monitoring and diagnostics



<http://www.fz-juelich.de/jsc/files/docs/vortraege/supercomputer/SC-JUGENE-Introduction.pdf>

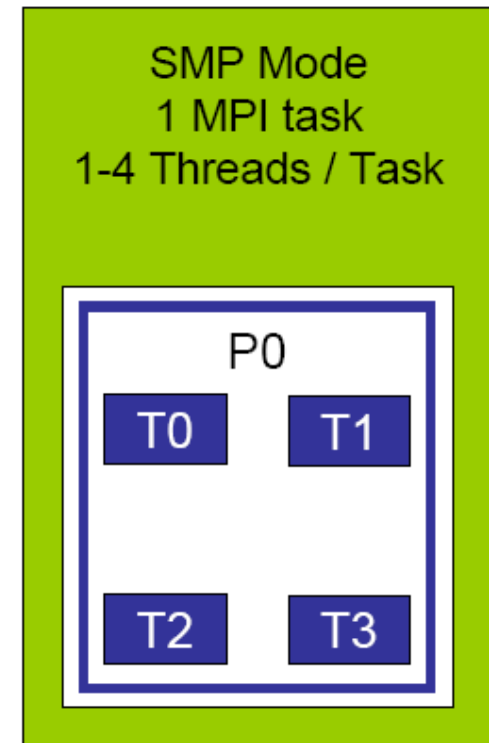
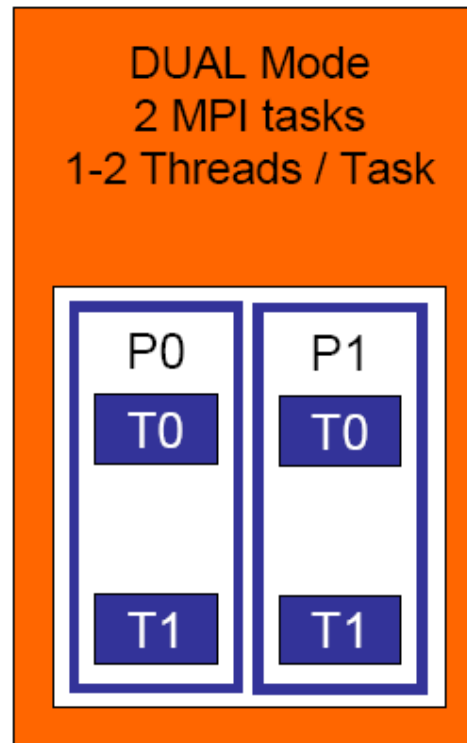
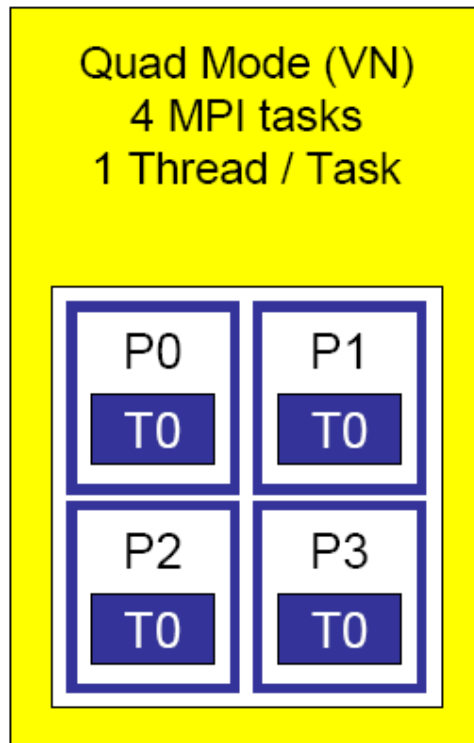
Multiprozessor mit verteiltem Speicher

■ Blue Gene/L ↔ Blue Gene/P

Property		Blue Gene/L	Blue Gene/P
Node Properties	Node Processors Processor Frequency Coherency L3 Cache size (shared) Main Store Main Store Bandwidth (1:2 pclk) Peak Performance	2* 440 PowerPC® 0.7GHz Software managed 4MB 512MB/1GB 5.6GB/s 5.6GF/node	4* 450 PowerPC® 0.85GHz SMP 8MB 2GB/4GB 13.6 GB/s 13.9 GF/node
Torus Network	Bandwidth Hardware Latency (Nearest Neighbour) Hardware Latency (Worst Case)	6*2*175MB/s=2.1GB/s 200ns (32B packet) 1.6µs (256B packet) 6.4µs (64 hops)	6*2*425MB/s=5.1GB/s 100ns (32B packet) 800ns (256B packet) 3.2µs(64 hops)
Tree Network	Bandwidth Hardware Latency (worst case)	2*350MB/s=700MB/s 5.0µs	2*0.85GB/s=1.7GB/s 3.5µs
System Properties	Area (72k nodes) Peak Performance (72k nodes) Total Power	114m ² 410TF 1.7MW	160m ² ~ 1PF ~2.3MW

Multiprozessor mit verteiltem Speicher

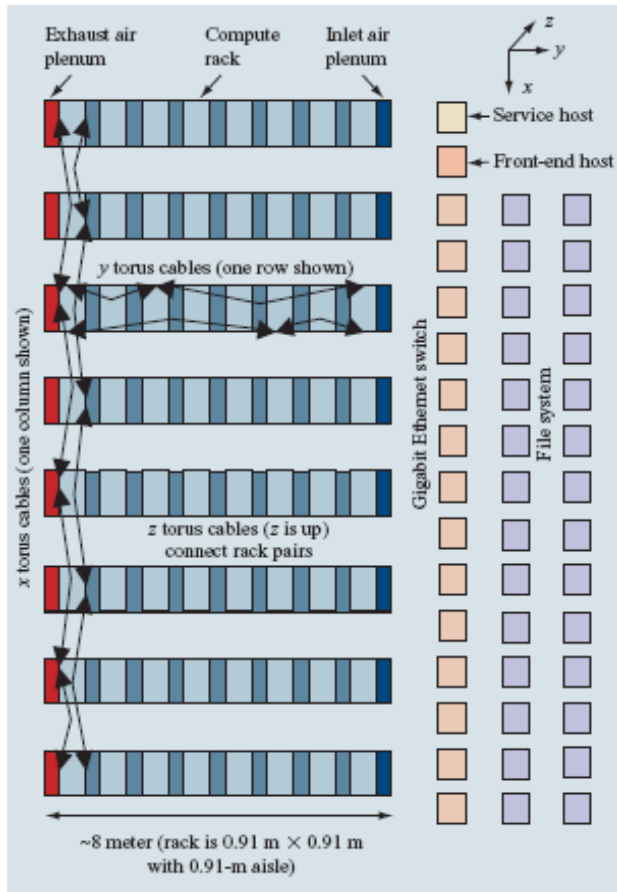
- **JUGENE - Juelicher BlueGene/P**
- Ausführungsmodi



<http://www.fz-juelich.de/jsc/files/docs/vortraege/supercomputer/SC-JUGENE-Introduction.pdf>

Multiprozessor mit verteiltem Speicher

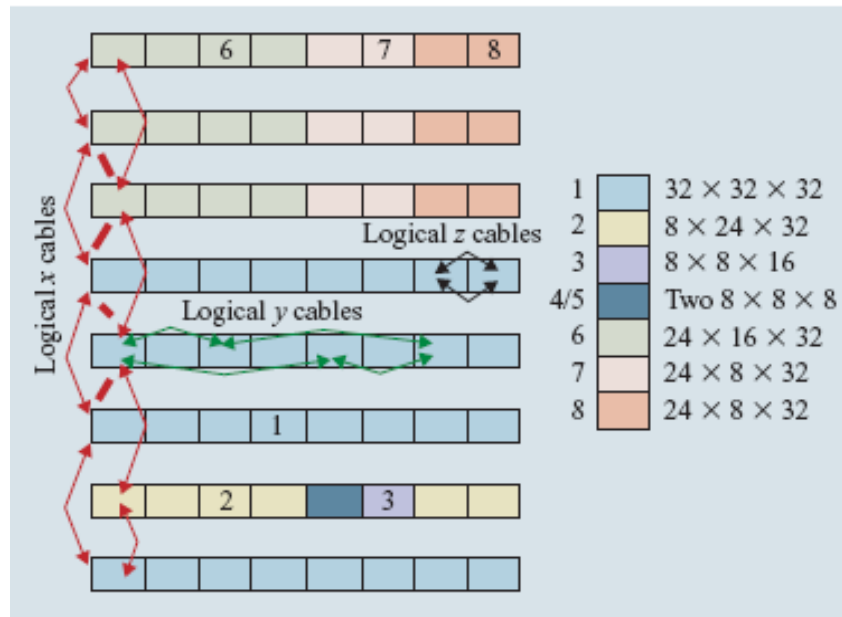
- Fallstudie Blue Gene/L
- Konfiguration



Gara et.al.: Overview of the Blue Gene/L system architecture. In: IBM Journal of Research and Development, Vol. 49, No. 2/3, 2005, pp.195-212

Multiprozessor mit verteiltem Speicher

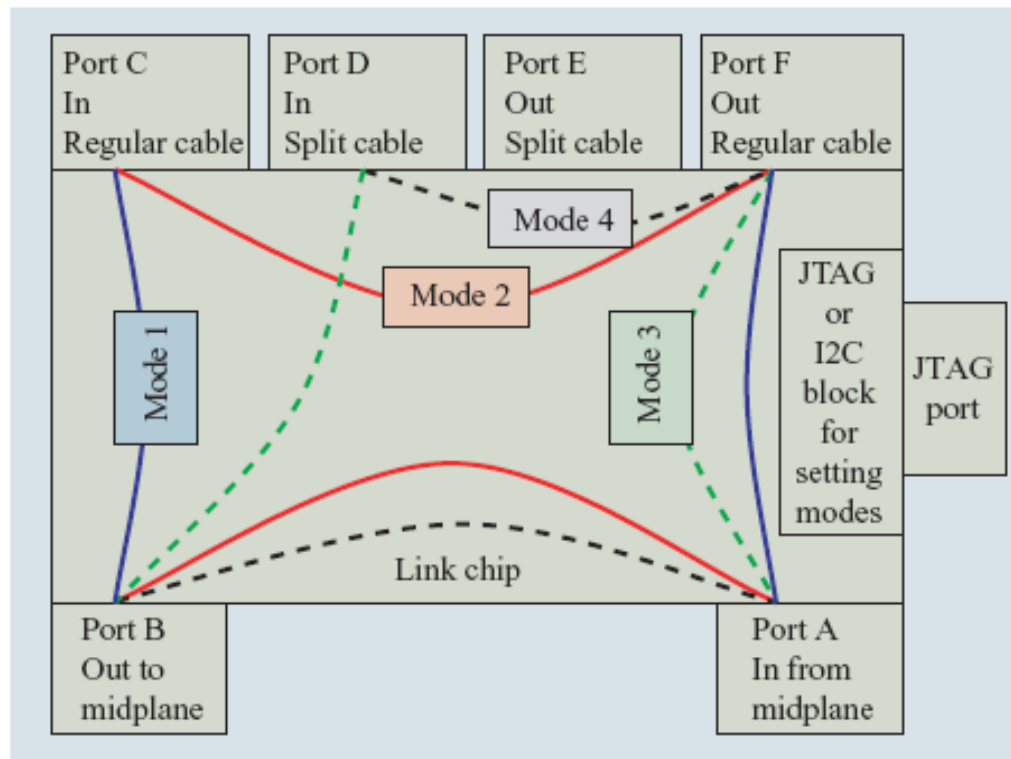
- Fallstudie Blue Gene/L
- Partitionierung
 - Beispiel: 8 Nutzer



Gara et.al.: Overview of the Blue Gene/L system architecture. In: IBM Journal of Research and Development, Vol. 49, No. 2/3, 2005, pp.195-212

Multiprozessor mit verteiltem Speicher

- Fallstudie Blue Gene/L
- BG/L Link Chip



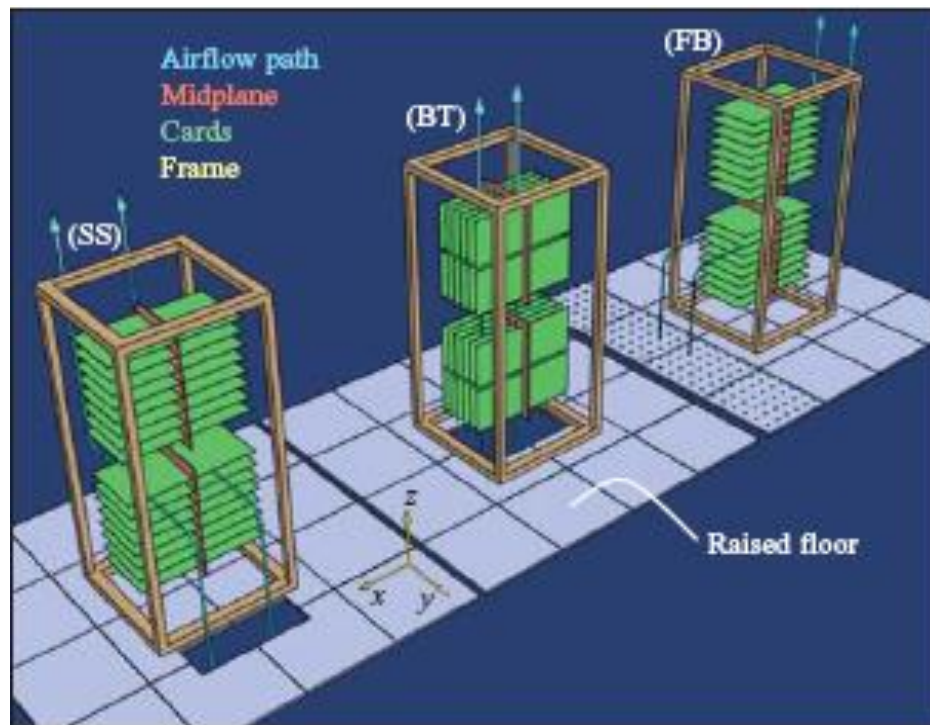
Gara et.al.: Overview of the Blue Gene/L system architecture. In: IBM Journal of Research and Development, Vol. 49, No. 2/3, 2005, pp.195-212

Fallstudie

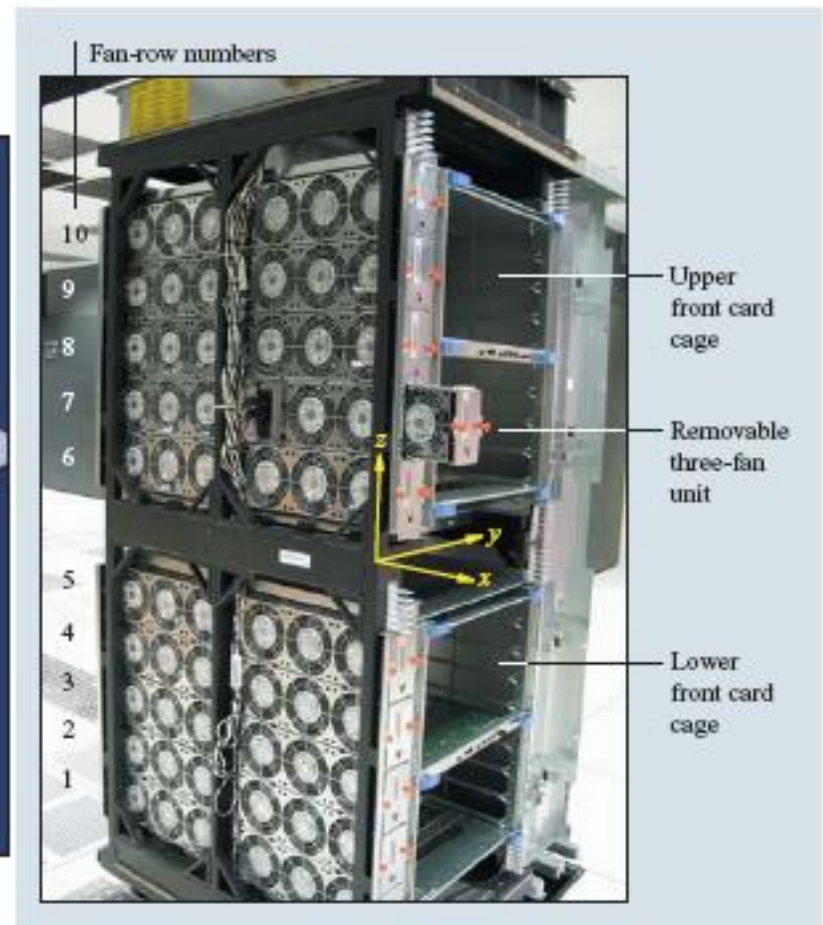
IBM Blue Gene/L Überblick

■ Systemkomponenten

■ Kühlung



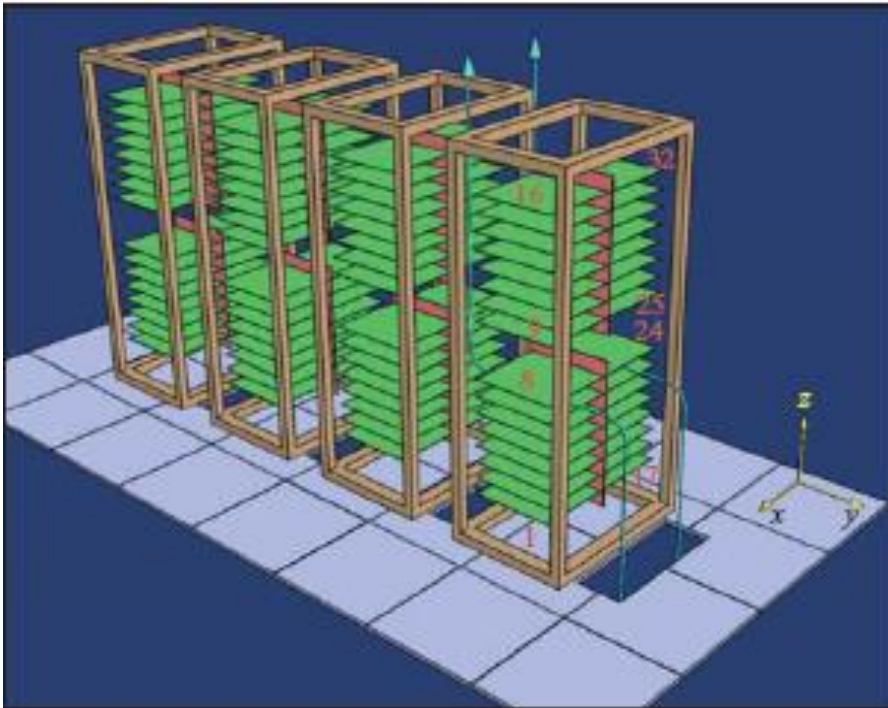
Quelle: P. Coteus, et.al.: Packaging the Blue Gene/L supercomputer. In: IBM Journal of Research and Development, Vol. 49, No. 2/3, 2005, pp.195-212



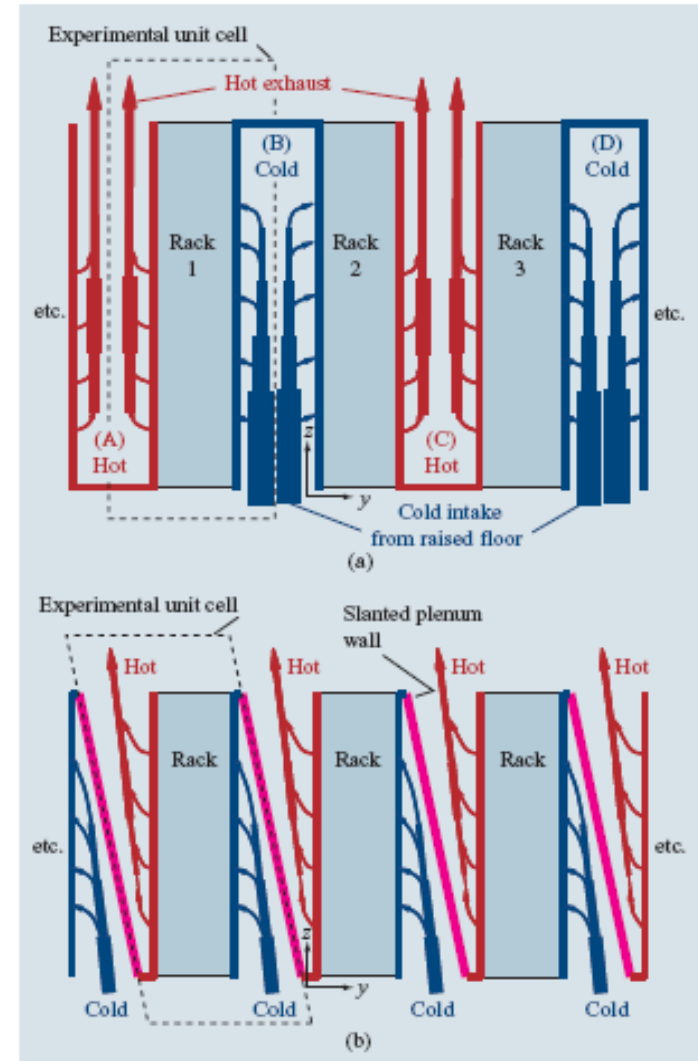
Fallstudie

IBM Blue Gene/L Überblick

■ Systemaufbau



Quelle: P. Coteus, et.al.: Packaging the Blue Gene/L supercomputer. In: IBM Journal of Research and Development, Vol. 49, No. 2/3, 2005, pp.195-212



Fallstudie

■ Literatur:

- IBM Journal of Research and Development, Vol. 49, No. 2/3, 2005, Special Issue

Fallstudie SuperMUC

■ Leibniz Rechenzentrum Garching bei München



■ [Movie on YouTube](#)

Quelle: Prof. Dr. Michael Gerndt, TUM

Fallstudie SuperMUC

- Distributed Memory Architecture
 - 18 partitions called islands with 512 nodes
 - Node is a shared memory system with 2 processors
 - Sandy Bridge-EP Intel Xeon E5-2680 8C
 - 2.7 GHz (Turbo 3.5 GHz)
 - 32 GByte memory
 - Infiniband network interface
- Processor has 8 cores
 - 2-way hyperthreading
 - 21.6 GFlops @ 2.7 GHz per core
 - 172.8 GFlops per processor



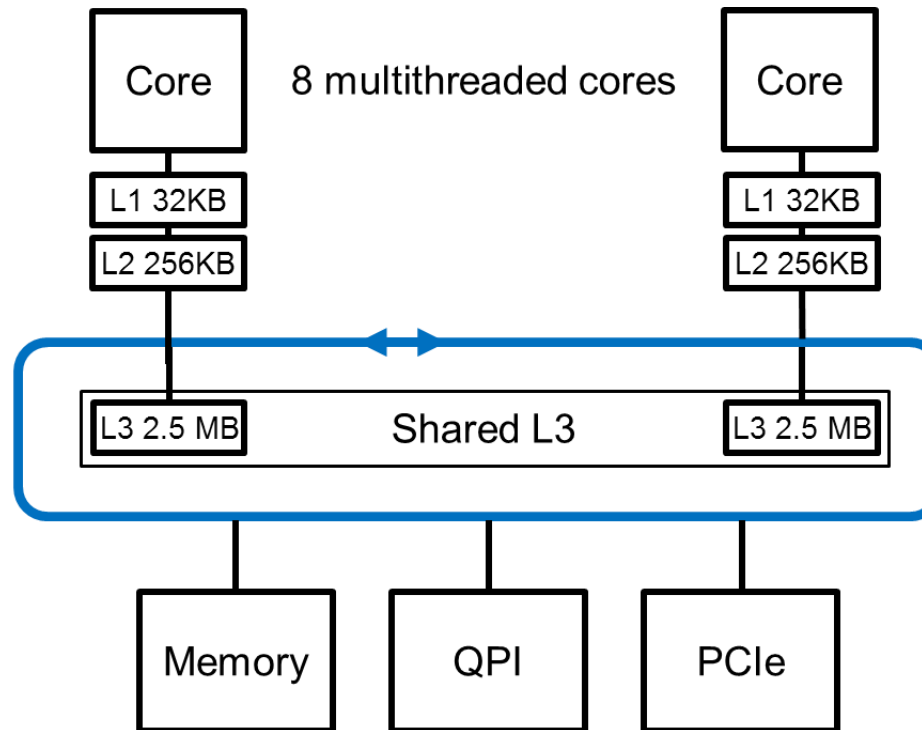
Quelle: Prof. Dr. Michael Gerndt, TUM

Fallstudie SuperMUC

■ Sandy Bridge Processor

Latency:

- 4 cycles
- 12 cycles
- 31 cycles



Bandwidth:

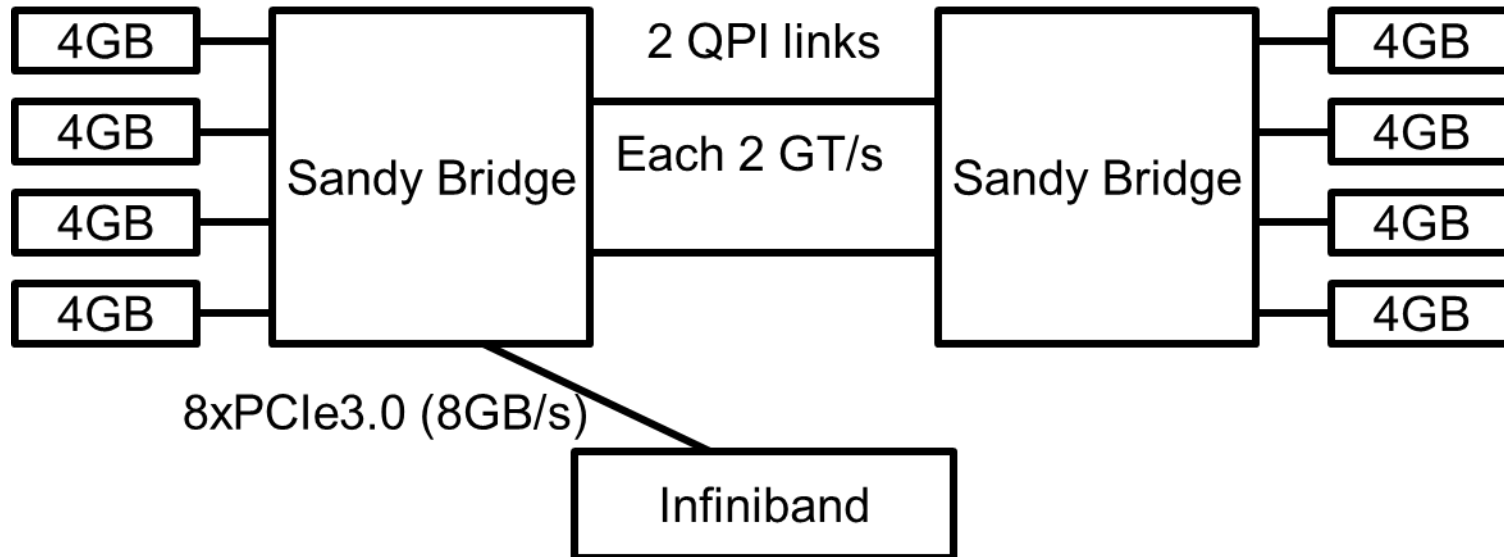
- $2 \times 16 / \text{cycle}$
- 32 / cycle
- 32 / cycle

Network frequency
equal to core frequency

Quelle: Prof. Dr. Michael Gerndt, TUM

Fallstudie SuperMUC

■ Node



- 2 processors with 32 GB of memory
- Aggregate memory bandwidth per node 102.4 GB/s Latency
 - local ~50ns (~135 cycles @2.7 GHz)
 - remote ~90ns (~240 cycles)

Quelle: Prof. Dr. Michael Gerndt, TUM

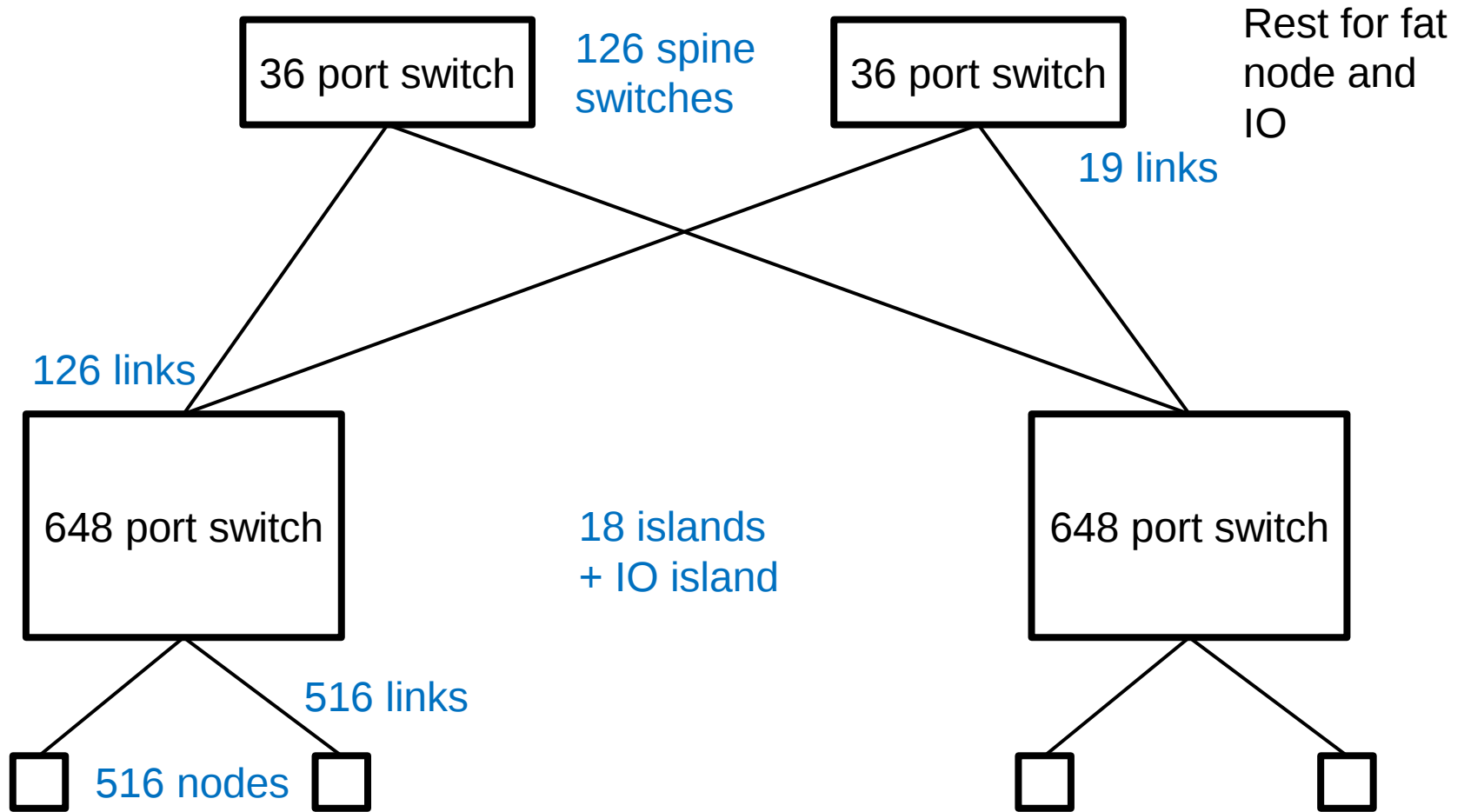
Fallstudie SuperMUC

- Interconnection Network
 - Infiniband FDR-10
 - FDR means fourteen data rate
 - FDR-10 has an effective data rate of 41.25 Gb/s
 - Latency: 100 nsec per switch, 1usec MPI
 - Vendor: Mellanox
 - Intra-Island Topology: non-blocking tree
 - 256 communication pairs can talk in parallel.
 - Inter-Island Topology: Pruned Tree 4:1
 - 128 links per island to next level

Quelle: Prof. Dr. Michael Gerndt, TUM

Fallstudie SuperMUC

■ Interconnection Network



Quelle: Prof. Dr. Michael Gerndt, TUM

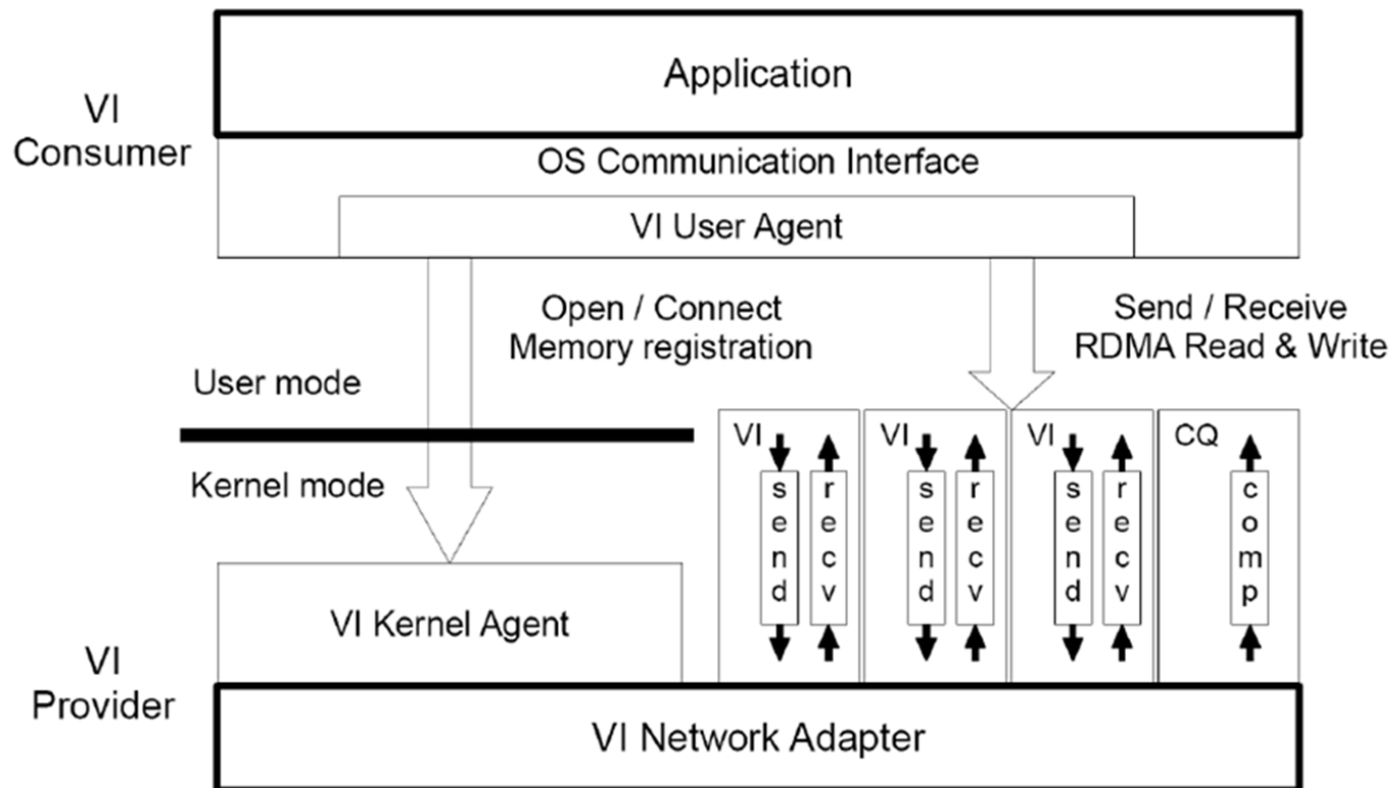
Fallstudie SuperMUC

- VIA and Infiniband
- VIA (Virtual Interface Architecture)
 - Standardized user-level network interface
 - Specification of the software interface not the NIC implementation
 - Can be fully implemented in the hardware NIC or major parts of the protocol processing can be on-loaded on the host processor.
 - Allows to bypass the OS on the data path
 - Consumers acquire one or more virtual interfaces via the kernel agent (control path)
- Efficient sharing of NICs
 - Getting more important in multicore processors

Quelle: Prof. Dr. Michael Gerndt, TUM

Fallstudie SuperMUC

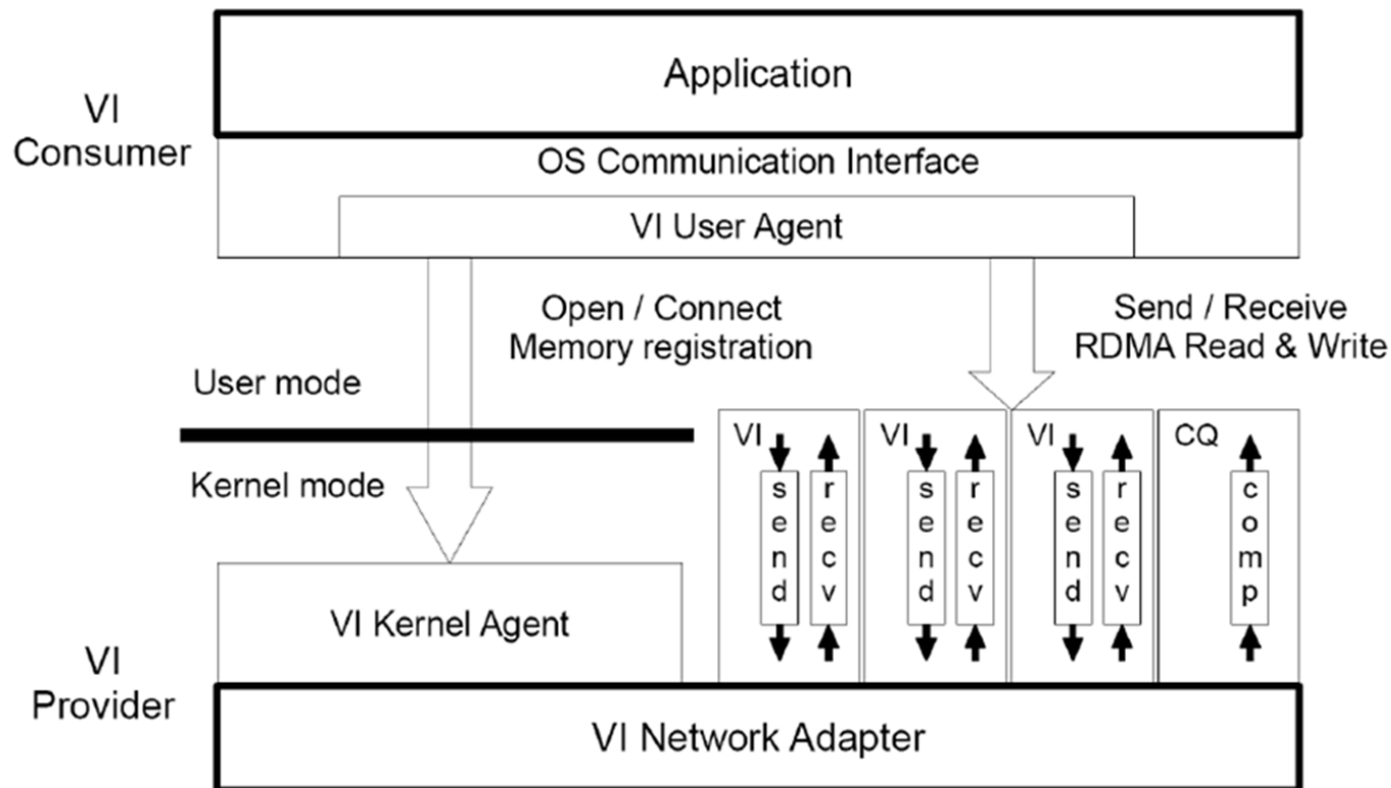
- VIA and Infiniband
- Virtual Interface Stack



Quelle: Prof. Dr. Michael Gerndt, TUM

Fallstudie SuperMUC

- VIA and Infiniband
- Virtual Interface Stack



Quelle: Prof. Dr. Michael Gerndt, TUM

Fallstudie SuperMUC

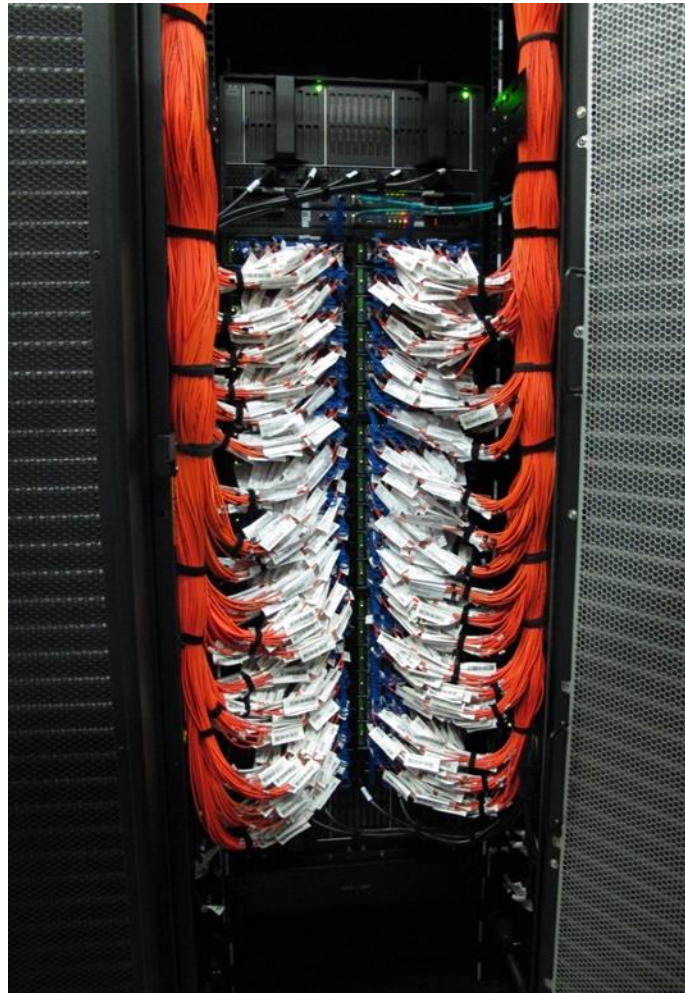
■ 9288 Compute Nodes



Quelle: Prof. Dr. Michael Gerndt, TUM

Fallstudie SuperMUC

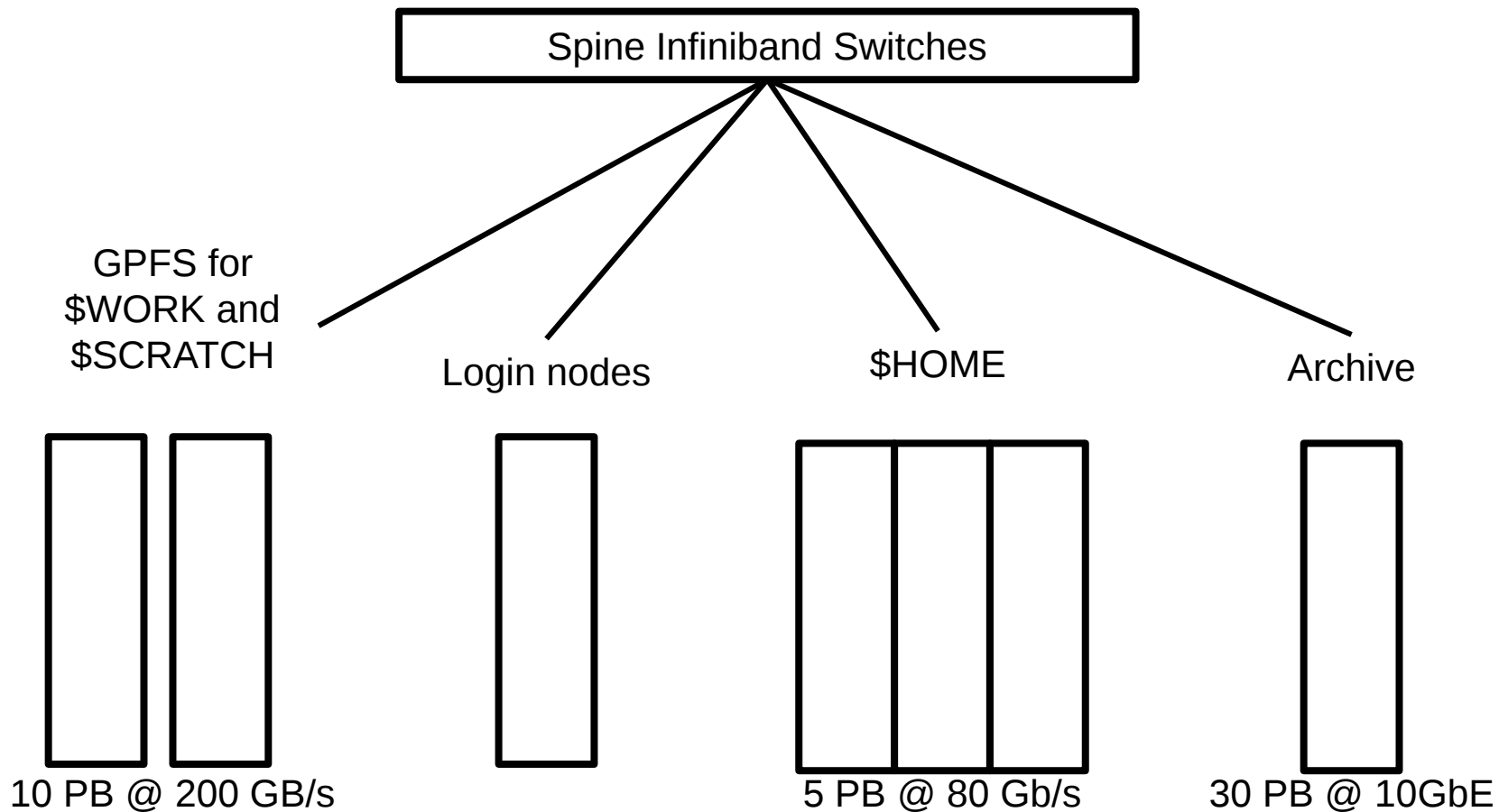
- Infiniband Interconnect
 - 11900 Infiniband Cables



Quelle: Prof. Dr. Michael Gerndt, TUM

Fallstudie SuperMUC

■ I/O System



Quelle: Prof. Dr. Michael Gerndt, TUM

Fallstudie SuperMUC

- I/O System: Parallel File System GPFS
 - 10 PByte, 200 GigaByte/s I/O Bandwidth



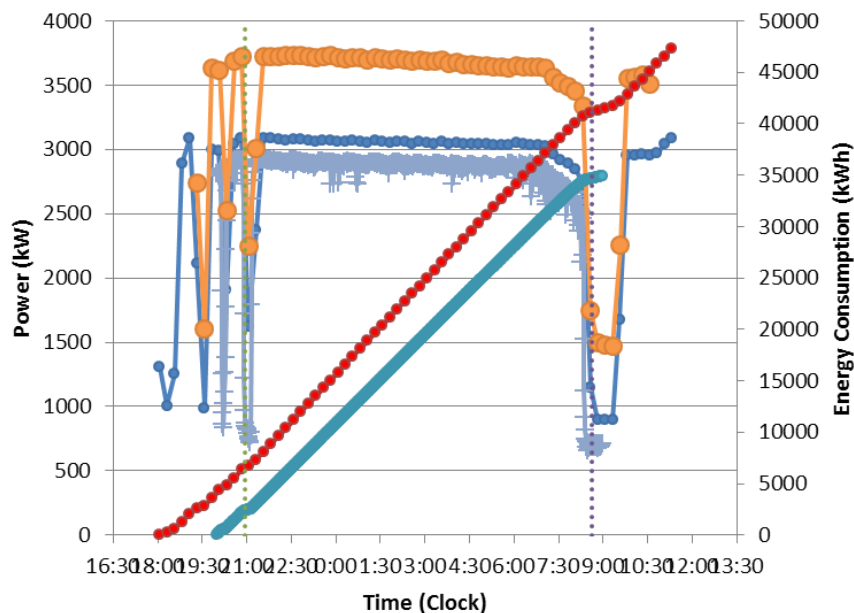
9 DDN SFA 12k Controller
5040 3 TByte SATA Disks

Quelle: Prof. Dr. Michael Gerndt, TUM

Fallstudie SuperMUC

■ SuperMUC: Power Consumption

SuperMUC HPL Power Consumption (Infrastructure, Machine Room & PDU Measurements)



Linpack HPL run May 17, 2012 – 2.582 PF

Run Start: 17.05.2012 20:56, 965,40 kW

Run End: 18.05.2012 08:37, 711,02 kW

Duration: 42045s or 11.68 hours

Avg. power: 2758.87 kW

Energy: 32308.68 kWh

- Power (Machine Room, kW)
- +— Power (PDU, kW)
- Power (infrastructure, kW)
- Energy (Machine Room, kWh)

Subsystems included in PDU measurements:

- Computational Nodes
- Interconnect Network

Quelle: Prof. Dr. Michael Gerndt, TUM

Fallstudie SuperMUC

■ SuperMUC: Kosten:

	2010-2014 Phase 1	2014-2016 Phase 2
High End System		
Investment Costs (Hardware and Software)	53 Mio €	~ 19 Mio €
Operating Costs (Electricity costs and maintenance for hardware und software, some additional personnel)	32 Mio €	~ 29 Mio €
SUM	85 Mio €	~48 Mio €
Extension Buildings (construction and Infrastructure)	49 Mio €	

Quelle: Prof. Dr. Michael Gerndt, TUM

Fallstudie SuperMUC

■ SuperMUC: Kosten:

	2010-2014 Phase 1	2014-2016 Phase 2
High End System		
Investment Costs (Hardware and Software)	53 Mio €	~ 19 Mio €
Operating Costs (Electricity costs and maintenance for hardware und software, some additional personnel)	32 Mio €	~ 29 Mio €
SUM	85 Mio €	~48 Mio €
Extension Buildings (construction and Infrastructure)	49 Mio €	

Quelle: Prof. Dr. Michael Gerndt, TUM

Fallstudie SuperMUC

- SuperMUC: Kosten:
 - New funding scheme: energy included
 - Contract includes energy cost for 5 years
 - Power consumption of the system varies between 1 and 3 MW depending on the usage by the applications
 - The contract is based on the energy consumed in a benchmark suite agreed between IBM and LRZ

Quelle: Prof. Dr. Michael Gerndt, TUM

Fallstudie SuperMUC

■ Leibniz Rechenzentrum Garching bei München



■ Movie on YouTube