FUJITSU

shaping tomorrow with you

Programming on K computer

Koh Hotta

The Next Generation Technical Computing

Fujitsu Limited

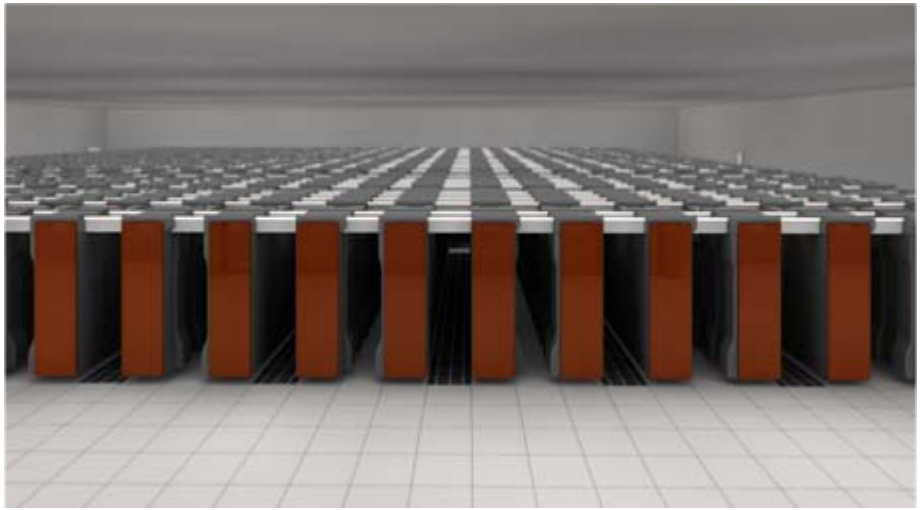
System Overview of “K computer”

FUJITSU

- Target Performance : 10PF
- over 80,000 processors
  - Over 640K cores
  - Over 1 Peta Bytes Memory
- Cutting-edge technologies
  - CPU : SPARC64 VIIIfx

    8 cores, 128GFlops

    Extension of SPARC V9
  - Interconnect, “*Tofu*” : 6-D mesh/torus
  - Parallel programming environment.



I have a dream

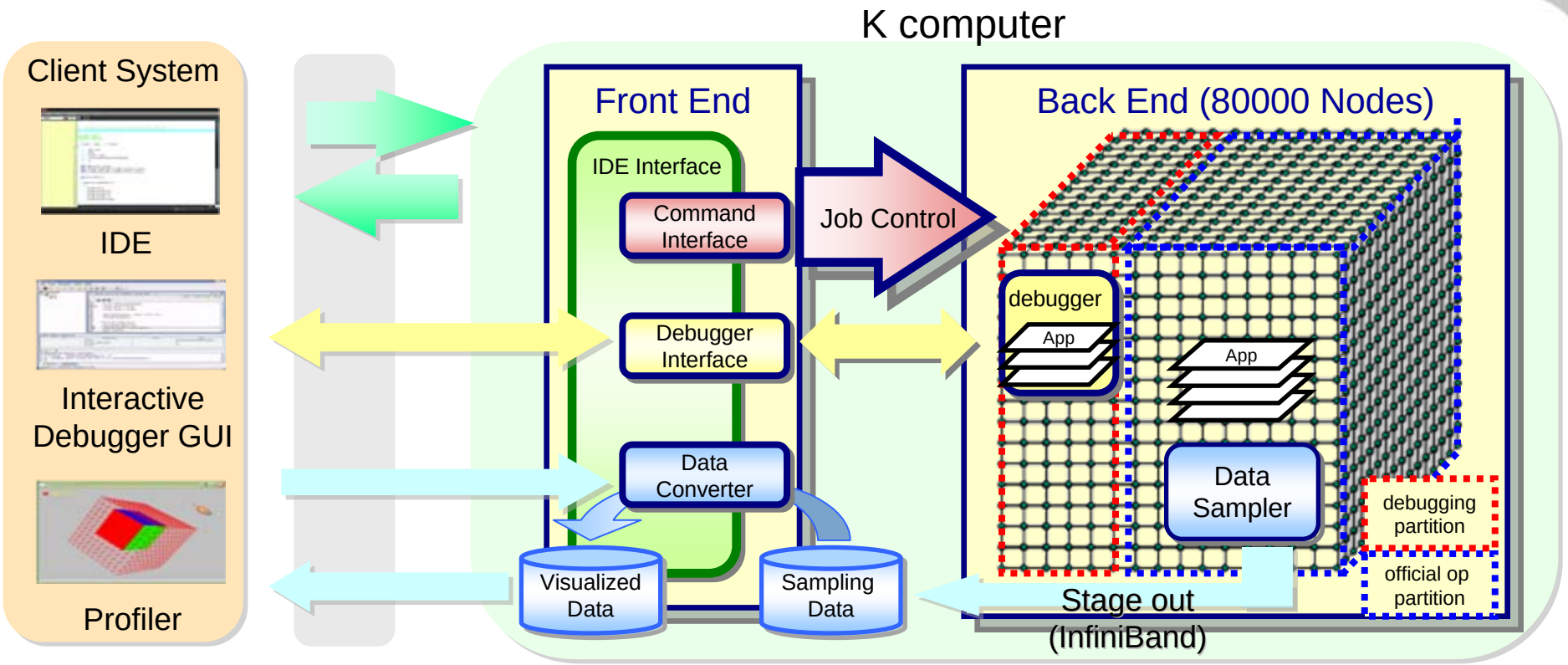
FUJITSU

*that one day you **just compile** your programs and enjoy high performance on your high-end supercomputer.*

- So, we must provide easy hybrid parallel programming method including compiler and run-time system support.

User I/F for Programming for K computer

FUJITSU



FUJITSU

Parallel Programming

Hybrid Parallelism on over-640K cores

FUJITSU

- Too large # of processes to manipulate
  - **To reduce number of processes, hybrid thread-process programming is required**
  - **But** ***Hybrid parallel programming is annoying for programmers***
- Even for multi-threading, procedure level or outer loop parallelism was desired
  - **Little opportunity for such coarse grain parallelism**
  - **System support for “fine grain” parallelism is required**

Targeting inner-most loop parallelization

FUJITSU

- Automatic vectorization technology has become mature, and vector-tuning is easy for programmers.
- Inner-most loop parallelism, which is fine-grain, should be an important *portion* for peta-scale parallelization.

Inner-most loop acceleration by multi-threading technology

FUJITSU

Inner-most loop acceleration by multi-threading technology

- CPU architecture is designed to reuse vectorization methodology efficiently.
- Targeting the inner-most loop automatic parallelization for multi-core processor.

VISIMPACT™ : you need not think about multi-cores

FUJITSU

- Efficient multi-thread execution on multiple cores tightly coupled with each other
  - Collaboration between hardware architecture and compiler optimization makes high efficiency
    - Shared L2 cache on a chip
    - High speed hardware barrier on a chip
    - Automatic parallelization

➔ Automatic parallelization facility makes multi-cores like a single high-speed core

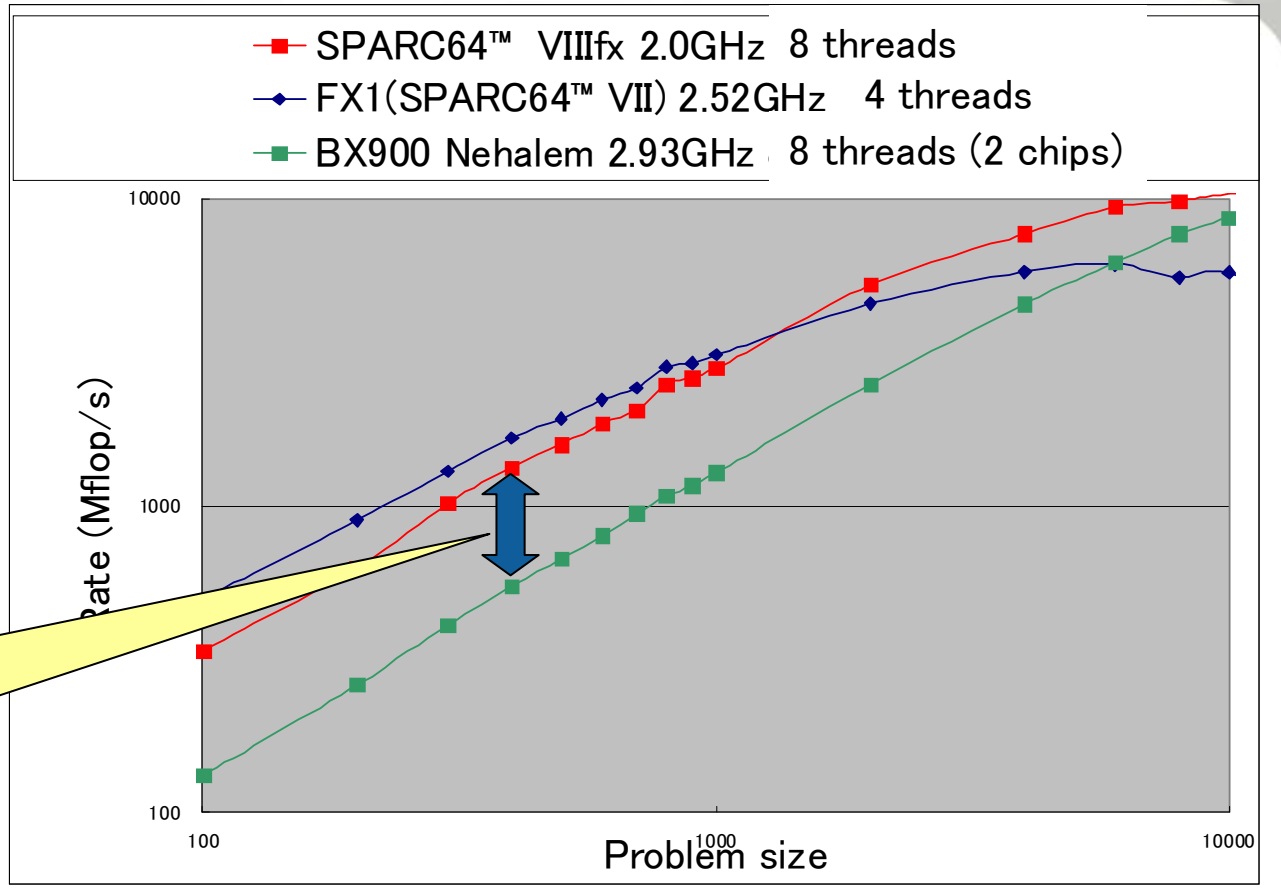
  - You need not think about cores in a CPU chip.

VISIMPACT™ Performance on DAXPY

FUJITSU

- Euroben 8 (DAXPY)

```
Do i = 1, n
  y(i,jsw) = y(i,jsw)
     + c0*x1(i)
End Do
```



Shared cache provides twice performance than Nehalem 2.93GHz.

FUJITSU

MPI

- Open MPI based
- Tuned to “Tofu” interconnect

MPI Approach for the K computer

FUJITSU

- Open MPI based
  - Open Standard, Open Source, Multi-Platform including PC Cluster
  - Adding extension to Open MPI for "*Tofu*" interconnect
- High Performance
  - Short-cut message path for low latency communication
  - Torus oriented protocol: Message Size, Location, Hop Sensitive
  - Trunking Communication utilizing multi-dimensional network links by Tofu selective routing.

Goal for MPI on K system

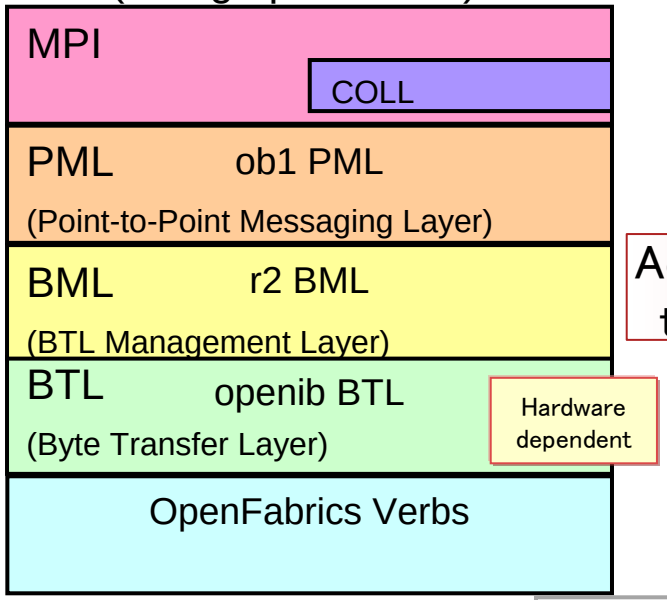
FUJITSU

- High Performance
  - Low Latency & High Bandwidth
- Highly Scalability
  - Collective Performance Optimized for Tofu interconnect
- High Availability, Flexibility and Easy to Use
  - Providing Logical 3D-Torus for each JOB with eliminating failure nodes.
  - Providing New up version of MPI Standard functions as soon as possible

MPI Software stack

FUJITSU

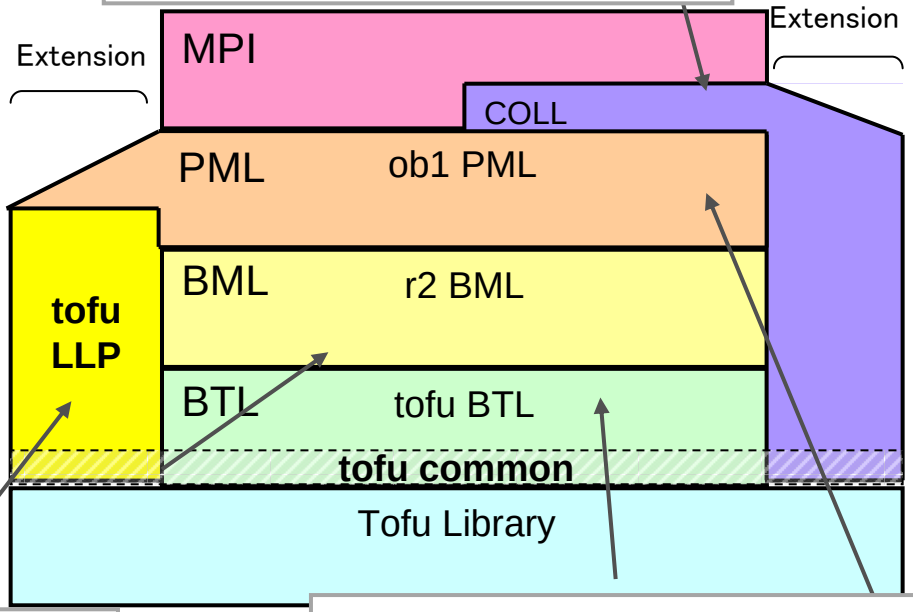
Original Open MPI Software Stack
(Using openib BTL)



Hardware dependent

Adapting to tofu

Supported special Bcast・Allgather・Alltoall・Allreduce for Tofu



LLP (Low Latency Path)

Special Hardware dependent layer
For Tofu Interconnect

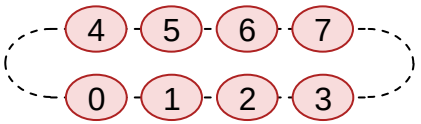
Providing Common Data processing and structures for BTL・LLP・COLL

Rendezvous Protocol Optimization etc

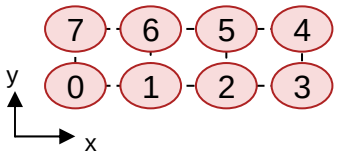
Flexible Process Mapping to Tofu environment

- You can allocate your processes as you like.
- Dimension Specification for each rank
  - 1D : (x)
  - 2D : (x,y)
  - 3D : (x,y,z)

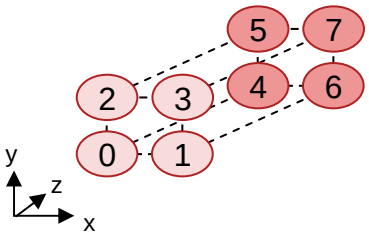
```
(0)
(1)
(2)
(3)
(7)
(6)
(5)
(4)
```




```
(0,0)
(1,0)
(2,0)
(3,0)
(3,1)
(2,1)
(1,1)
(0,1)
```




```
(0,0,0)
(1,0,0)
(0,1,0)
(1,1,0)
(0,0,1)
(0,1,1)
(1,0,1)
(1,1,1)
```

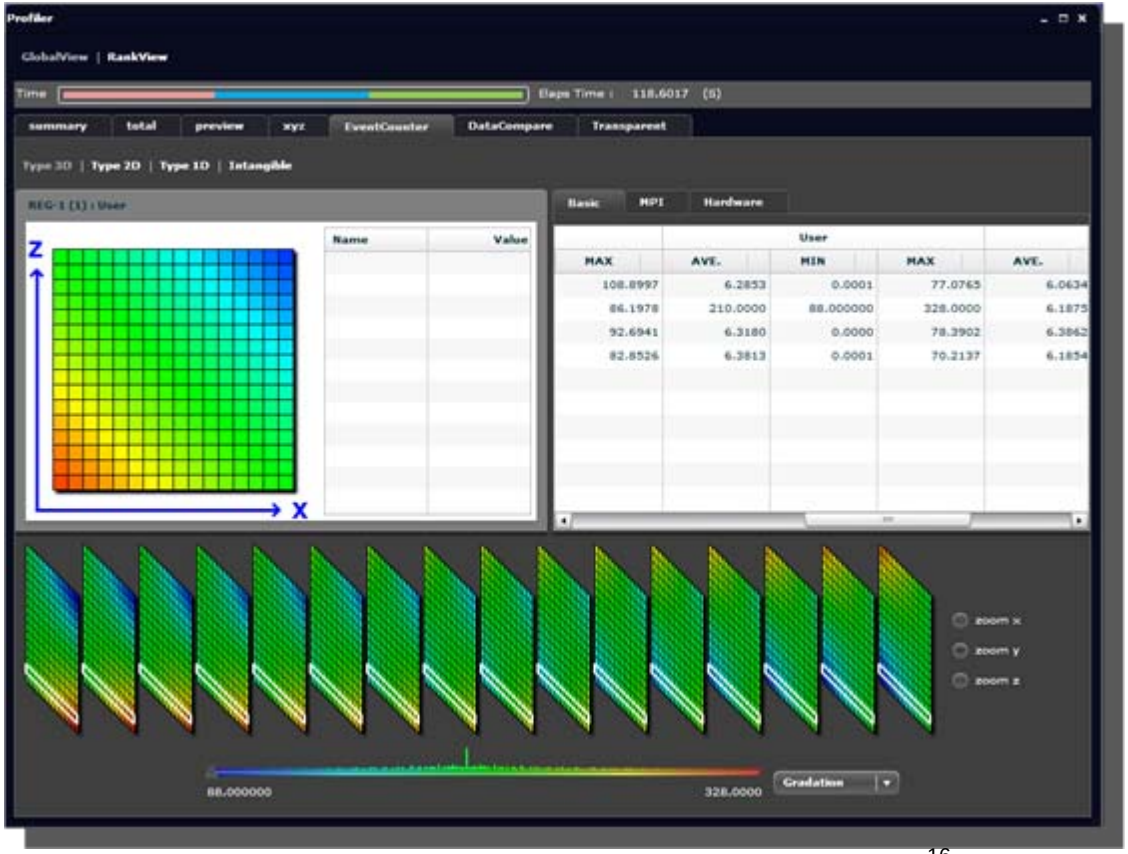


FUJITSU

Performance Tuning

- Not only by compiler optimization, but also you can manipulate performance
  - Compiler directives to tune programs.
- Tools to help your effort to tune your programs
  - ex. Watch your program using event counter

Performance Tuning (Event Counter Example)



- 3-D job example
  - Display 4096 procs in 16 x 16 x 16 cells
  - Cells painted in colors according to the proc status (e.g. CPU time)
  - Cut a slice of jobs along x-, y-, or z-axis to view

Conclusion:
Automatic and transparency of performance

- VISIMPACT™ lets you treat 8-cored CPU as a single high-speed core.
  - Collaboration by the CPU architecture and the compiler.

    High-speed hardware barrier to reduce the overhead of synchronization

    Shared L2 cache to improve memory access

    Automatic parallelization to recognize parallelism and accelerate your program
- Open MPI based MPI to utilize “Tofu” interconnect.
- Tuning facility shows the activity of parallel programs.

shaping tomorrow with you