Development of the *K computer* and toward EXA scale machines

17th October, 2012

Motoi Okuda
Technical Computing Solutions Unit
Agenda

- History of the *K computer* project
- Design concept of the *K computer* and its achievement
  - CPU, interconnect and reliability
  - Application performance
- Targeting Exascale computing
  - Lessons learned from *K computer* Project
  - Does co-design scheme work well?
  - Challenge to Exascale computing
- Conclusion
**Time-line of the K computer project**

- In 2001, High-end computing WG was established and investigation activities started
- Following Gird project, Elementally studies had started in 2005
- **K computer** project started in mid-2006 with two application projects
- System installation started in Oct. 2010
- Full system installation was finished in August 2011 and official operation has started 28th Sep. 2012

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre projects</td>
<td>High end computing WG</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>System</td>
<td>NAREGI: National Grid Project</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Elementally Studies for Next Gen. System</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Application</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Conceptual design</td>
<td>Detailed design</td>
<td>Prototype, evaluation</td>
<td>Production, installation, and adjustment</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Next-Generation Integrated Nano-science Simulation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Next-Generation Integrated Simulation of Living Matter</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>HPCI Strategic Applications</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
History of the *K computer* project

Design concept of the *K computer* and its achievement
- CPU, interconnect and reliability
- Application performance

Targeting Exascale computing
- Lessons learned from *K computer* Project
- Does co-design scheme work well?
- Challenge to Exascale computing

Conclusion
Design targets of K computer

- High Performance
  - High peak performance and high performance efficiency
  - *100 times more powerful* than the fastest supercomputer in 2005

- High operability
  - Low power consumption
  - High reliability
  - Easy to operate

- Highly parallel application performance and productivity
  - Easy to extract high performance from the highly paralleled programs without inordinate burden to programmers
  - Performance target of each strategic applications

- Time line
  - Development of the system by *the end of March 2012*
Fujitsu development strategy and achievement

- Trade off in system design and development phases
  - Combination of assured & mature technologies and advanced & challenging technologies
  - CPU & Interconnect architecture
  - CPU & Interconnect chip development process
  - Packaging & cooling
  - Implementation
  - Software

- No.1 in 37\textsuperscript{th} TOP500 in June 2011
  - 8.162 PFLOPS, 9.9MW and 93\% efficiency in LINPACK BMT

- No.1 in 38\textsuperscript{th} TOP500 in Nov. 2011
  - 10.51 PFlops, 12.66MW and 93.17\% efficiency in LINPACK BMT

- 2011 Gordon Bell Award, Peak Performance
  - Sustained performance of 3.08 PFLOPS (Running on 7.08PFlops system)
  - Efficiency of 43.6\%
High Performance
- Fujitsu designed SPARC64 VIIIfx CPU with HPC enhancement
- High Memory BW: 64 GB/s
- Newly designed interconnect, Tofu

High reliability, stability and operability
- Reliable designed SPARC64 VIIIfx CPU
- Newly designed interconnect, Tofu
- Direct water cooling

Greenness
- SPARC64 VIIIfx CPU
- Direct water cooling
CPU & Tofu interconnect features

- **SPARC64™ VIIIfx**: SPARC V9 extension for HPC (**HPC-ACE**)
  - Increase computing power ( = increase # of computing unit)
    - 8 cores and **SIMD extension** (with MASK operation)
  - Improve performance efficiency
    - **hardware barrier** between cores and **shared 2nd cache**
    - **software controllable cache**
    - **FP resister # extension** (32 → 256)
    - **mathematical function hardware implementation**

- **Interconnect**: New innovative interconnect — **Tofu** —
  - Increase # of connectable computing node
    - 3D mesh & Torus topology
  - Improve performance
    - 5GB/sec. /link x 2 high band width and 100GB/sec high throughput (10 links/node)
  - Improve performance efficiency
    - **hardware implementation** of MPI_Allgather, MPI_Alltoall and barrier function
Example of mask operation effect on SIMD (2 M&A pipeline x 2 /core) acceleration in *Computational chemistry program*

- Due to the branch operation “if” in the loop, SIMD operation scheduling doesn’t promote. This cause increase of floating point operation wait penalty.
- By using mask operation (-Ksimd=2), compiler can apply software pipelining and achieve high efficiency

---

**Example of mask operation effect**

```c
40 1 do iv=1, natv
41 1 local unroll(4)
42 1 !$omp parallel do default(none)
43 1 !$omp private(eaarg,work)
44 1 !$omp shared(tuv,tuvres)

<<<<< Loops=Information Start>>>>
<<<<< [OPTIMIZATION]
<<<<< SIMD
<<<<< SOFTWARE PIPELINING
<<<<< Loops=Information End>>>>

45 2 p 4v  do iv=1, ngr
46 2 p 4v  eaarg = -uuu(g,iv) + tvu(g,iv,1)
47 3 p 4v  if (eaarg>0) then
48 3 p 4v  work = 1.0d0 + eaarg
49 3 p 4v  else
50 3 p 4v  work = exp(eaarg)
51 3 p 4v  endif
52 2 p 4v  tvures(g,iv,1) = work
53 2 p 4v  enddo
54 1 enddo
55 !$omp parallel
```

---

**Graphical Representation**

- **One operation commit**
- **Four operations commit**
- **Floating point operation wait**
- **Apply -Ksimd=2 option**

---

**Improvement**

- **x 2.5 improvement**
Example of the register size extension (from 32 to 256) effect on **NPB3.3-LU** high cost loop (340 lines)

- By using large size register, compiler can generate more efficient operation scheduling order and also eliminate unnecessary load operations

On SPARC64™ VIIIfx

![Graph showing performance improvement](image)
How does register extension (from 32 to 256) work in real 142 application program kernels?

<table>
<thead>
<tr>
<th>Program No.</th>
<th>Improved ratio</th>
<th>Performance improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>Average</strong> 120%</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Max.</strong> 252%</td>
</tr>
</tbody>
</table>

- Performance improvement by register extension
SIMD and Register Size Extension Effect

- Performance improvement of FP register # extension and SIMD extension in real 138 application program
- Clock normalized one core performance comparison between SPARC64™ VIIIfx (w/o SIMD & register size extension) and VIIIfx.

<table>
<thead>
<tr>
<th>Program No.</th>
<th>Improved ratio</th>
<th>Performance improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0%</td>
<td>100%</td>
</tr>
<tr>
<td></td>
<td>20%</td>
<td>101%</td>
</tr>
<tr>
<td></td>
<td>40%</td>
<td>102%</td>
</tr>
<tr>
<td></td>
<td>60%</td>
<td>103%</td>
</tr>
<tr>
<td></td>
<td>80%</td>
<td>104%</td>
</tr>
<tr>
<td></td>
<td>100%</td>
<td>105%</td>
</tr>
<tr>
<td></td>
<td>120%</td>
<td>106%</td>
</tr>
<tr>
<td></td>
<td>140%</td>
<td>107%</td>
</tr>
</tbody>
</table>

Performance improvement by HPC-ACE (SIMD + register extension)

- Performance improvement of FP register # extension and SIMD extension in real 138 application program
- Clock normalized one core performance comparison between SPARC64™ VIIIfx (w/o SIMD & register size extension) and VIIIfx.
Application Controllable Cache

- Application can access to cache management feature
  - L2$ can be divided by two sectors, Normal cache and Pseudo Local Memory
  - Use case of Sector Cache on NPB3.3-CG
    - By putting array P on sector 1, floating point data $ access wait is reduced

---

<table>
<thead>
<tr>
<th>Optimized code</th>
</tr>
</thead>
<tbody>
<tr>
<td>111</td>
</tr>
<tr>
<td>112</td>
</tr>
<tr>
<td>120</td>
</tr>
<tr>
<td>121</td>
</tr>
<tr>
<td>122</td>
</tr>
<tr>
<td>122</td>
</tr>
<tr>
<td>122</td>
</tr>
<tr>
<td>122</td>
</tr>
<tr>
<td>122</td>
</tr>
<tr>
<td>122</td>
</tr>
<tr>
<td>122</td>
</tr>
<tr>
<td>122</td>
</tr>
<tr>
<td>122</td>
</tr>
<tr>
<td>122</td>
</tr>
<tr>
<td>123</td>
</tr>
<tr>
<td>124</td>
</tr>
<tr>
<td>125</td>
</tr>
<tr>
<td>126</td>
</tr>
<tr>
<td>127</td>
</tr>
<tr>
<td>128</td>
</tr>
<tr>
<td>133</td>
</tr>
<tr>
<td>135</td>
</tr>
</tbody>
</table>

---

L2 cache miss ratio reduce from 4.38% to 3.16%

x 1.23 improvement

- Floating point data load $ access wait
- Integer data load $ access wait

---

Copyright 2012 FUJITSU LIMITED
One CPU performance analysis on real applications shows contribution of implemented features

**App. A (Fluid dynamics)**

- 30% Reciprocal approx.
- 256regs
- SPARC-V9

K - 8th (/128GFLOPS)

**App. B (Life science)**

- 35% 256regs
- SPARC-V9

K - 8th (/128GFLOPS)

**App. C (Nano tech.)**

- 39% Masked operator
- Reciprocal approx.
- 256regs
- SPARC-V9

K - 8th (/128GFLOPS)

**App. D (Life science)**

- 44% Reciprocal approx.
- 256regs
- SPARC-V9

K - 8th (/128GFLOPS)
Collective communications

- High performance MPI_Barrier, MPI_Allreduce and MPI_Bcast used Tofu barrier facility

Scalable MPI_Allgather and MPI_Alltoall for Tofu interconnect

- 256-nodes All-to-all performance
- Tofu bandwidth is better than InfiniBand QDR
New *allreduce* algorithm for Tofu interconnect contributed to achieve 2011 Gordon Bell Award*

Original

- 32%
- Tuning for computation
- Tuning for communication

Modified for K computer

- 19%

- Tuning for computation
- Tuning for communication

- Computation
- Communication
  (8,000 atoms, 256 nodes)

- Computation
- Communication
  (107,292 atoms, 55,296 nodes)

- New high speed *allreduce* algorithm: Communication throughput 3.2GB/s.
- Application performance: 3.08Pflops (43.6% efficiency)

* : SC ’11 Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis, Article No. 1, First-principles calculations of electron states of a silicon nanowire with 100,000 atoms on the K computer

Courtesy of RIKEN
Several application programs already achieved very high efficiency on large scale K computer system.

Practical application outcomes is expected in early stage of HPCI program.

Performance of application programs on large scale K computer:

- RSDFT: Real Space DFT program
- ZZ-EFSI: Combination of CFD and structure analysis program simulating blood flow in vessel
- UT-Heart: Multi-scale and multi-physics simulating system for coronary artery circulation with capillary and metabolism in cell
- ccpmd: Molecular dynamics program designed for K computer

Graph showing PFlops for full system and 2/3 system for each application program:

- RSDFT: Peak 43.6%, Effective 42.7%
- ZZ-EFSI: Peak 42.7%, Effective 27.7%
- UT-Heart: Peak 38.0%, Effective 27.7%
- ccpmd: Peak 38.0%

Courtesy of ISLiM
History of the *K computer* project

Design concept of the *K computer* and its achievement
- CPU, interconnect and reliability
- Application performance

Targeting Exascale computing
- Lessons learned from *K computer* Project
- Does co-design scheme work well?
- Challenge to Exascale computing
  - Technologies trend
  - Japanese initiative

Conclusion
Lessons learned from *K computer* Project

- Challenges to leading edge project will bring us:
  - *Strong mind and challenging spirit*
  - End to End system development: LSI, OS, MW, system, & interconnect
  - Application software: momentum of application software asset

- Importance of the feedback from application evaluation and optimization process
  - Accumulate more expertise required for effective 1PFlops applications
  - *Transfer of the expertise to the next generation systems* development

- Project management
  - Consensus building for the *national roadmap* and securing *sustainable budget*
  - The *speed of decision-making* is really the key
  - Understanding of the nature of technology development
    - LSI development becomes very risky business
    - Combination of *assured & mature technologies* and *advanced & challenging technologies* brought the success
Co-design and Co-development work scheme

**Key phrase:** “Co-design & co-development will be the key issue for aiming at Exascale computing”

**Benefits to application team**
- Influence on the next-generation architecture and compiler
- Early access and closer look at new supercomputer architecture

**Benefits to computer development team**
- Evaluation of new supercomputer architectures
- Better understanding of the next-generation applications
Time-line of the *K computer* project and Co-design

<table>
<thead>
<tr>
<th>CY</th>
<th>2006</th>
<th>2007</th>
<th>2008</th>
<th>2009</th>
<th>2010</th>
<th>2011</th>
<th>2012</th>
</tr>
</thead>
<tbody>
<tr>
<td>System</td>
<td>Fixed System Conceptual design ▼</td>
<td>Start system test ▼</td>
<td>Start installation ▼</td>
<td>Test operation started ▼</td>
<td>End of installation ▼</td>
<td>Operation start ▼</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Conceptual design</td>
<td>Detailed design</td>
<td>Prototype, evaluation</td>
<td>Production, installation, and adjustment</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Target Applications ▲ System outline ▼ Application optimization by using lead off system (FX1) and other highly parallel systems

Feed back to compiler development ★ Application optimization on test system and test operation phase *K computer* Gordon Bell Award

Application

Target Applications ▲ System outline ▼ Application optimization by using lead off system (FX1) and other highly parallel systems

Feed back to compiler development ★ Application optimization on test system and test operation phase *K computer* Gordon Bell Award

Next-Generation Integrated Nano-science Simulation

Next-Generation Integrated Simulation of Living Matter

HPCI Strategic Applications
When we challenge to Exascale system, it may need to start the project around 4.5~6 years before the target operation start date.

First one year may be an only chance for application programs to influence to computer design.
Various type of architecture is trying to address Exascale computing.
Approach to Exascale computing

- Trade off between cost performance and burden of application implementation
Power consumption and reliability & resiliency may become key issue.

- Increase # of cores
- Increase core performance & functionality
- **Reduce power consumption**
- Improvement of Reliability & resiliency
- Interconnect interface integration
- Control function integration
- 3D stacking technologies
Japanese Approaches to Exa-Scale Computing

- In 2011, Japanese Government started the Exa-scale computing project
  - In 2011, application team reviewed the applications with architecture team and wrote up the application requirement report
  - In 2012, four two-year-Feasibility study (FS) themes were implemented

**Outline**

<table>
<thead>
<tr>
<th>Leading members and vendors</th>
<th>Outline</th>
</tr>
</thead>
<tbody>
<tr>
<td>RIKEN</td>
<td><strong>Assessing the following architectures in terms of applications</strong></td>
</tr>
<tr>
<td>The University of Tokyo</td>
<td><strong>Similar architecture to K Computer’s</strong> (for Advanced &amp; Efficient Latency Core-Based Architecture)</td>
</tr>
<tr>
<td>FUJITSU</td>
<td><strong>Arithmetic Accelerators</strong> (for compute oriented Applications)</td>
</tr>
<tr>
<td>University of Tsukuba</td>
<td><strong>Vector Supercomputers</strong> (for High-Bandwidth Applications.)</td>
</tr>
<tr>
<td>HITACHI</td>
<td></td>
</tr>
<tr>
<td>NEC</td>
<td></td>
</tr>
</tbody>
</table>

**Requirements from applications**

(from the Application review report)

- X : Each program requirement
- GP : General Purpose
- CB : Capacity -Bandwidth oriented
- RM : Reduced Memory
- CO : Compute oriented

---

25
Japanese Approaches to Exascale Computing (cont.)

- Two-year FSs expect to provide future direction of Exascale system development and main development project is expected to start around 2014
- Trans-Exa system, around 2014-1015, may be necessary to step up the R&D for Exascale system

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>National Projects</td>
<td>Operation of <em>K Computer</em></td>
<td>HPCI Strategic Applications Program</td>
<td>FS Projects</td>
<td>Exa-system Development Project</td>
<td>Exa system</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Fujitsu

**PRIMEHPC FX10**
- 1.85 x CPU Performance
- Better Operability

**Trans-Exa system**
- Improved CPU & Network Performances
- High-Density Packaging & Low Power Consumption
Agenda

- History of the *K computer* project
- Design concept of the *K computer* and its achievement
  - CPU, interconnect and reliability
  - Application performance
- Targeting Exascale computing
  - Lessons learned from *K computer* Project
  - Does co-design scheme work well?
  - Challenge to Exascale computing
- Conclusion
Success of the K computer project brought us several valuable and important expertise

- in Project management, system development and application development
- and reminded us of importance of co-design & co-development and its time line

The new challenges to Exascale computing has already started with reflecting the lessons and expertise we acquired through the **K computer** project

The key is *demonstration of Petascale computing power* with real & practical application programs
shaping tomorrow with you