15th ECMWF Workshop

FUJITSU
shaping tomorrow with you

Findings from real petascale computer systems with meteorological applications

Toshiyuki Shimizu
Next Generation Technical Computing Unit
FUJITSU LIMITED

October 2nd, 2012

Outline

- Introduction of Fujitsu petascale supercomputer architecture and software environment

- K computer and FX10 performance evaluations
  - New features and their performance efficiency improvements
  - Performance evaluation of hybrid execution using VISIMPACT

- Challenges towards Exascale computing

- Summary

Introduction of Fujitsu petascale supercomputer architecture and software environment

Massively parallel computing product

- Single CPU / node architecture for multicore CPU
  - FX1(4 cores) → K(8 cores) → FX10(16 cores)
  - High memory bandwidth balanced with CPU performance (8 DIMMs/node)
- Essential technologies for massively parallel computing environment
  - For high performance & multicore CPU
    - HPC-ACE ISA, Automatic hybrid parallelization
  - For higher scalability
    - HW support of collective communication, Direct network “Tofu”
- Inherit application software assets

Automatic parallelization
HW support of collective comm.

Direct network Tofu
HPC-ACE: SIMD etc.

FX1

CY2008～
40GF, 4-core CPU

K computer

CY2010～
128GF, 8-core CPU

PRIMEHPC FX10

CY2012～
236.5GF, 16-core CPU

HPC-ACE architecture

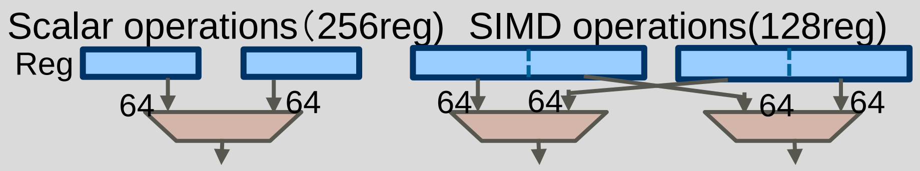
FUJITSU

SIMD and # of register extension

# of floating point registers

- 64bit double precision regs from 32 to 256
- can be used as 128bit x 128 sets of SIMD regs

Floating point operations



SIMD register specifications

HPC-ACE	Intel (AVX)	Intel (KNC)	IBM (P7)
128bit x 128	256bit x 16	512bit x 32	128bit x 64

Memory access instructions

Scalar inst (256reg): 64bit load/store

SIMD inst (128reg): 128bit load/store

Support for listed data access

HPC-ACE	Intel (AVX)	Intel (KNC)	P7
Scalar inst. can be used	Supported from AVX2	Supported	None

Conditional SIMD inst.

- Mask generation, Conditional move, and store

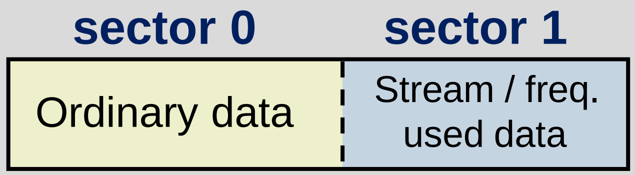
Sector cache

Function

Cache ways are arbitrary divided into two sectors, 0 and 1. Special load inst. loads the data for sector 1.

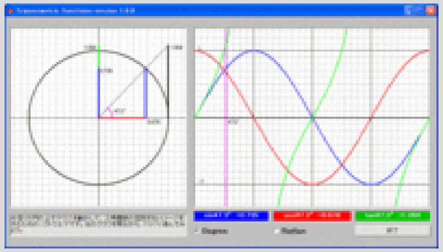
Sector 1 usage examples:

- Stream data: Avoids replacement of ordinary data
- Frequently used data: Assure high hit rate for the data



Mathematical calc. acceleration

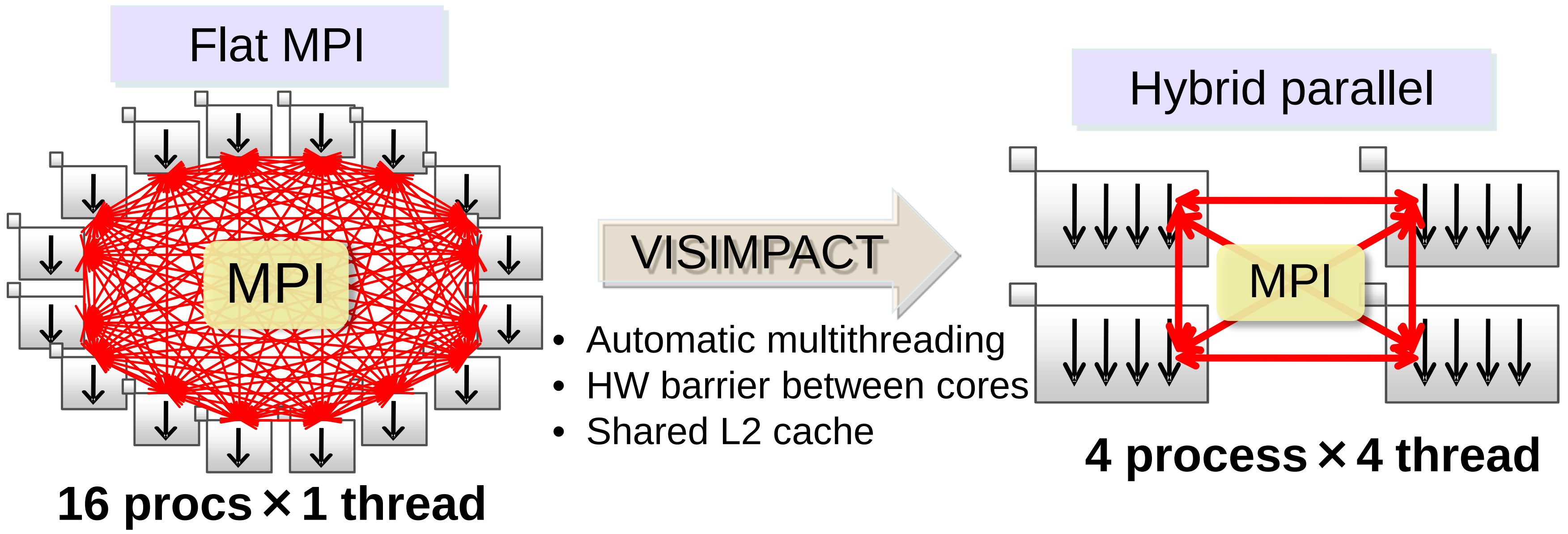
Floating-point reciprocal approximations and trigonometric func. instructions



VISIMPACT (Virtual Single Processor by Integrated Multi-core Parallel Architecture)

FUJITSU

- Pros. & cons. of hybrid parallel execution
  - Pros.
    - Good scalability of processes (reduce communications by reducing the # of processes)
    - Larger usable memory per process
    - Efficient use of memory (smaller work area)
  - Cons.
    - Programing is harder due to two level of parallelization
- VISIMPACT: automatic parallelization of MPI programs
  - Sophisticated automatic parallelization originated from automatic vectorisation
  - CPU inter-core barrier assists multithreaded parallel execution



- Automatic multithreading
- HW barrier between cores
- Shared L2 cache

Tofu interconnect

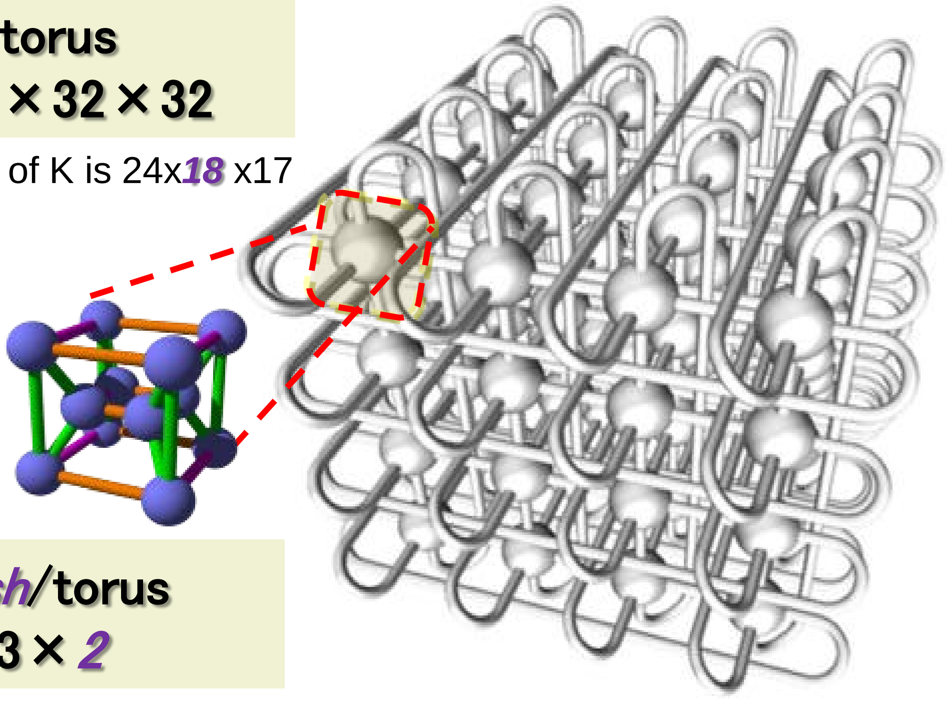
Direct 6D mesh/torus topology (Max. 32x32x32x*2*x3x*2*)

Mesh topology indicated by iatric

- Good scalability beyond 3D torus
- High reliability and operability
- Shorter average hop counts and higher bisection bandwidth
- HW support of collective communication

3D torus
Max. 32×32×32

Implementation of K is 24x*18* x17



3D *mesh*/torus
2*×3×*2

System software stack

FUJITSU

- All components are maintained by Fujitsu (including OSS)
- Common user interface as x86 clusters and hybrid systems

Applications(R&D for highly parallel application, algorithms)

HPC Portal / System Management Portal

Technical Computing Suite

System management

- Management, control, monitoring, installation support
- Automatic recovery from the failure enables 24x 365 operation

Job operations

- High speed job invocation
- Efficient scheduler
- Rich center operation funcs

High-performance file system (FEFS)

- Lustre-based distributed file system
- High scalability (Thousands of IO server)
- 1TB/s class IO access performance

Collaborate w/ Whamcloud

Automatic parallelizing compiler

- Fortran, C, C++
- High level SMID parallel, multicore parallel executions

Math. Lib and tools

- Rich tools
- High performance libraries (SSL II/BLAS etc.)

Parallel lang. & comm. Lib.

- OpenMP, MPI(Open MPI), XPFortran
- Multicore and SIMD support

Linux based OS (Enhanced for FX10) : low OS jitter

K computer / PRIMEHPC FX10 / X86 clusters

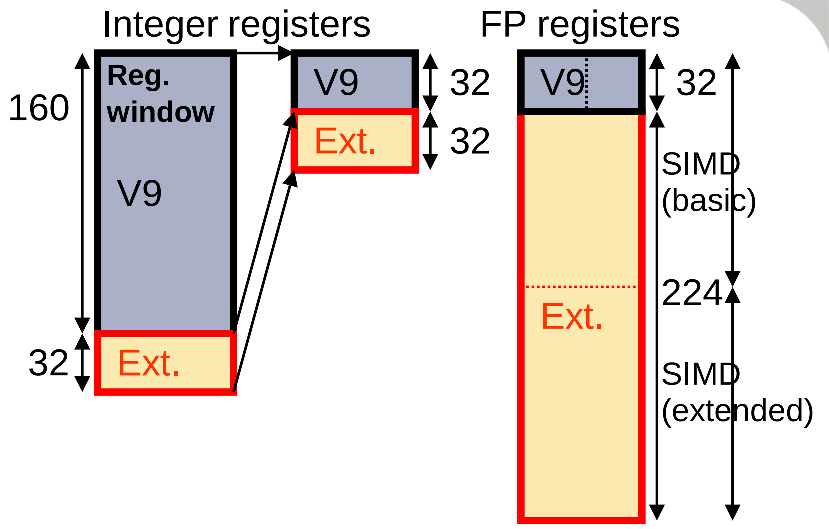
New features and their performance efficiency improvements on K computer and PRIMEHPC FX10

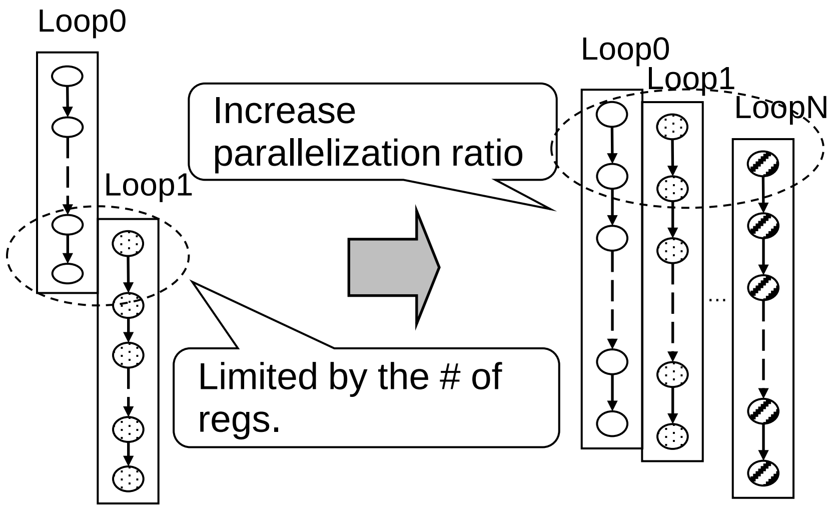
New features and evaluations

- Selected new features for this evaluations
  - # of register extension
  - Conditional SIMD
  - Floating-point reciprocal approximations
- Evaluations
  - Contributions of performance efficiency improvements
  - Performance efficiency of large applications using K computer & FX10

# of register extension

- Enhancement of SPARC V9 specification
  - # of integer regs from 32 to 64
  - # of DP FP regs from 32 to 256
    - Lower 32 registers work as SPARC-V9
    - Extended registers can be access from non-SIMD inst.
- Register extension reliefs the limitation for parallelization due to the shortage of registers
  - Large loops are efficiently parallelized and reduce spill/fill overhead





Conditional SIMD instructions

- Conditional branch (if clause) limits the performance
  - Limits instruction scheduling of compiler
  - Cancels instructions in the instruction pipelines
- Conditional SIMD instruction removes the conditional branches and software pipelining is widely applied

```
      subroutine sub(a, b, c, x, n)
       real*8  a(n), b(n), c(n), x(n)
       do i = 1 , n
         if ( x(i) .gt. 0.0 ) then
           a(i) = b(i) * c(i)
         else
           a(i) = b(i) - c(i)
         endif
       enddo
       end
```

Conditional SIMD

```
   :
.L100:
   :  calculations of b(i)*c(i) and b(i)-c(i)
fcmpgted,s      %f32,%f34,%f32         Mask generation
fselmovd,s      %f42,%f40,%f32,%f42    Use of mask
std,s           %f42,[%o5+%xg1]
add             %xg1,16,%xg1
bne,pt          %icc, .L100
nop
   :
```

Floating-point reciprocal approximations

- Divide and Sqrt reciprocal approximation instructions
  - Error of approx. < 1/256 calc. reciprocal approximations
  - Divide and Sqrt are pipelined

```
REAL(KIND=8),DIMENSION(1024) :: A, B, C
DO I=1,1024
  A(I) = B(I) / SQRT(C(I))
END DO
```

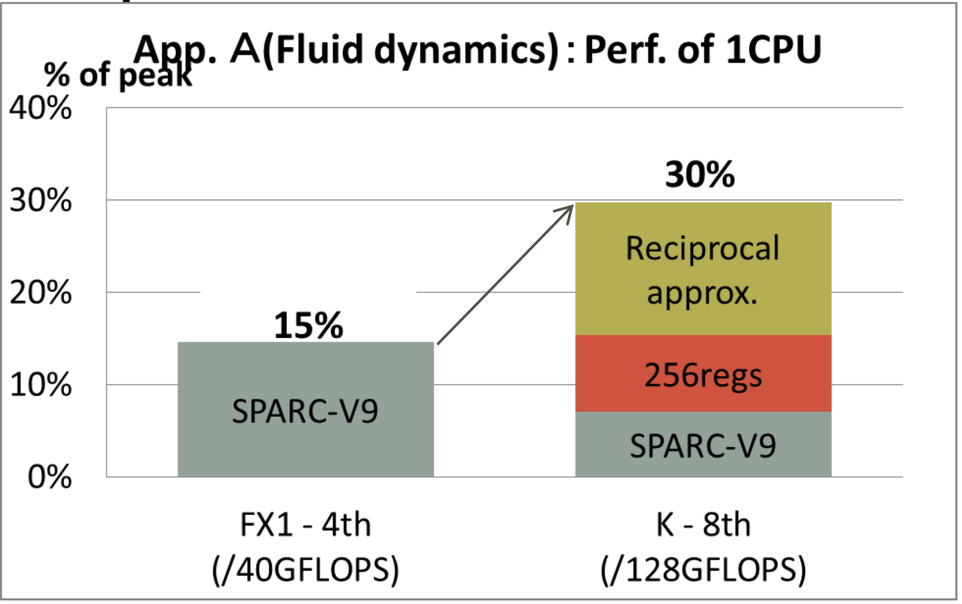
No reciprocal approximations

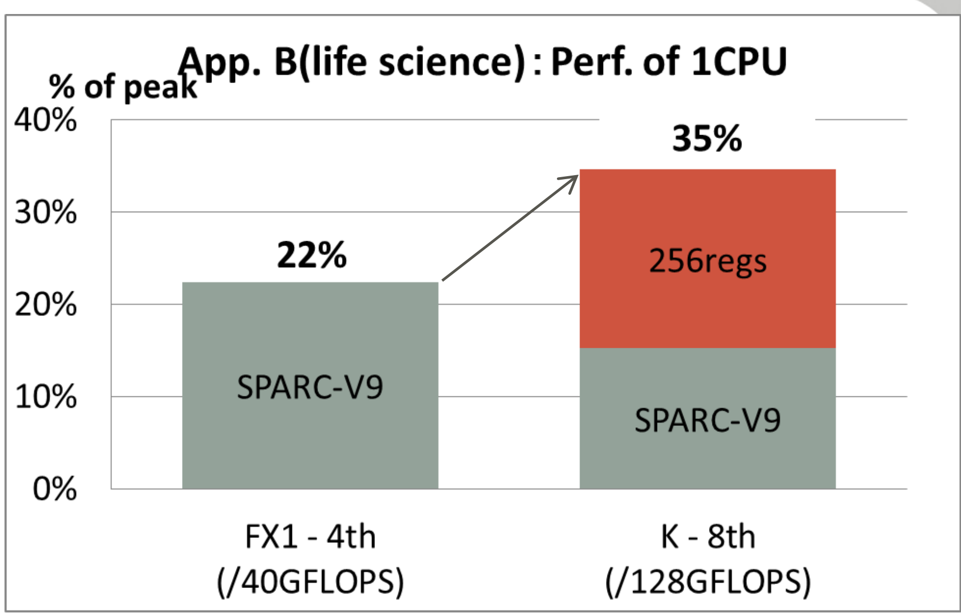
```
.LL1:
    ldd        [%o2 + %g2], %f2
    ldd        [%o1 + %g2], %f0
    subcc      %g3, 1, %g3
    add        %g2, 8, %g1
    fsqrtd     %f2, %f2
    fdivd      %f0, %f2, %f0
    std        %f0, [%o0 + %g2]
    mov        %g1, %g2
    bne,pt     %icc, .LL1
    nop
```

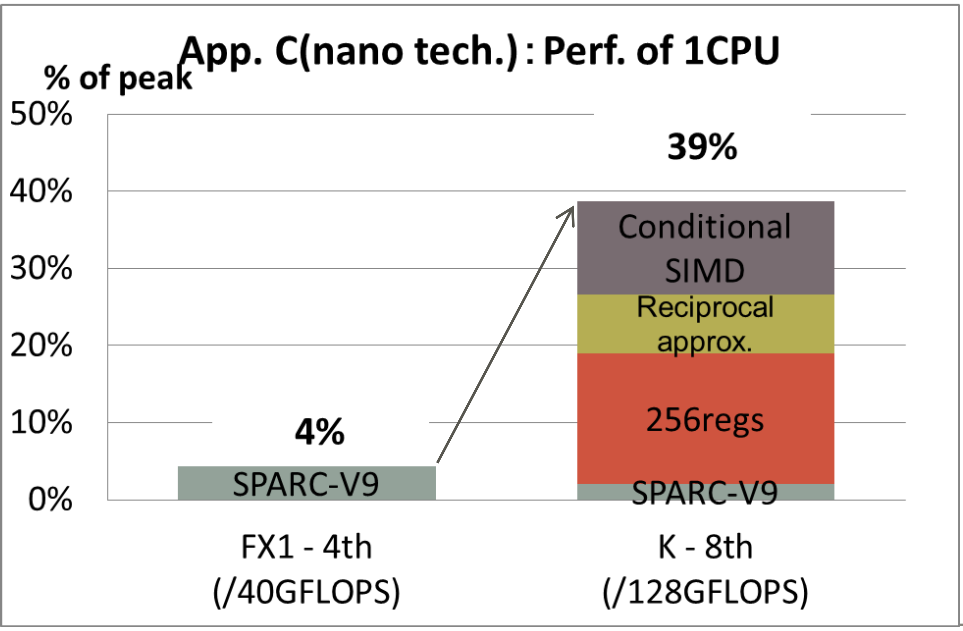
Reciprocal approximations

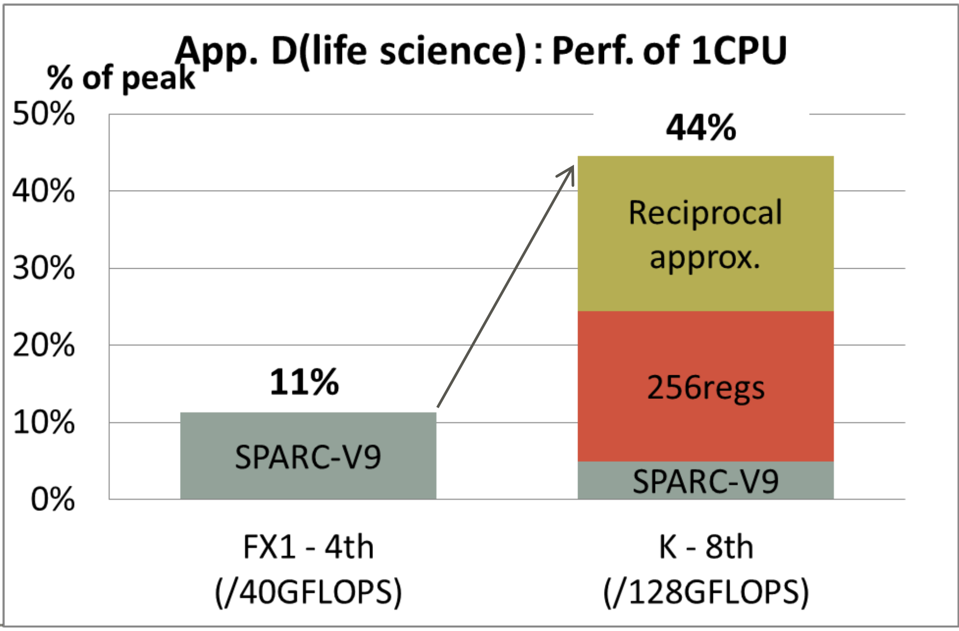
```
.LL1:
    ldd        [%o2 + %g2], %f2
    ldd        [%o1 + %g2], %f0
    subcc      %g3, 1, %g3
    add        %g2, 8, %g1
    fmuld      %f8, %f2, %f10
    frsqrtad   %f2, %f2
    fmuld      %f10, %f2, %f14
    fnmsubd    %f2, %f14, %f8, %f14
    fmaddd     %f2, %f14, %f2, %f2
    fmuld      %f10, %f2, %f12
    fnmsubd    %f2, %f12, %f8, %f12
    fmaddd     %f2, %f12, %f2, %f2
    fmuld      %f10, %f2, %f10
    fnmsubd    %f2, %f10, %f8, %f6
    fcmpeqd    %f10, %f16, %f18
    fmaddd     %f2, %f6, %f2, %f2
    fselmovd   %f4, %f2, %f18, %f2
    fmuld      %f0, %f2, %f0
    std        %f0, [%o0 + %g2]
    mov        %g1, %g2
    bne,pt     %icc, .LL1
    nop
```

Contributions of new features for efficiency improvements









Performance efficiency of real apps.

- Measured by petascale K computer and PRIMEHPC FX10
  - 6 out of 11 applications attain over 30% efficiency

Applications	# of Cores	Performance		System (Peak)	Efficiency of FX1
		Efficiency	IPC		
A*	98,304	30%	1.7	K(1.57PF)	8%
B*	98,304	41%	1.5		18%
C*	98,304	32%	1.3		5%
D*	98,304	27%	1.0		
E*	98,304	12%	0.6		
F*	98,304	52%	1.5		
G*	98,304	9%	0.7		
H*	20,480	3%	0.6	K(0.33PF)	
I*	32,768	23%	0.9	K(0.52PF)	
J*	98,304	40%	1.5	K(1.57PF)	
K	6,144	30%	1.6	FX10(0.1PF)	15%

* Measured by K computer: Results are from trial use & not the final.

Performance evaluation of hybrid execution using VISIMPACT on K computer and PRIMEHPC FX10

Survey of hybrid execution on FX10

- The fastest combination (✔) varies on applications
- Hybrid parallelization reduces load imbalance & comm. time

Applications	Programing model	# of processes x # of threads					Notes
		16P1T	8P2T	4P4T	2P8T	1P16T	
1	**MPI ***				✔		Load imbalance
2	**MPI+OpenMP**					✔	Load imbalance
3	**MPI**	✔					Thread size is small
NPB.CG	**MPI**					✔	Communication time, shared cache
NPB.LU	**MPI**	✔					Thread size is small, cache line conf.
6	**MPI+OpenMP**				✔		Communication time
7	**MPI+OpenMP**		✔				Load imbalance, communication time
8	**MPI ***			✔			Load imbalance, communication time
9	**MPI+OpenMP**			✔			Load imbalance, thread ratio
10	**MPI ***					✔	Communication time
11	**MPI+OpenMP**		✔				Thread ratio, communication time

Reduce load imbalance

Reduce comm. time

Reduce comm. time

MPI * : Hint directives for thread parallelization are used

Evaluations with practical meteorological apps.

- WRF and COSMO were evaluated as operational weather forecasting codes

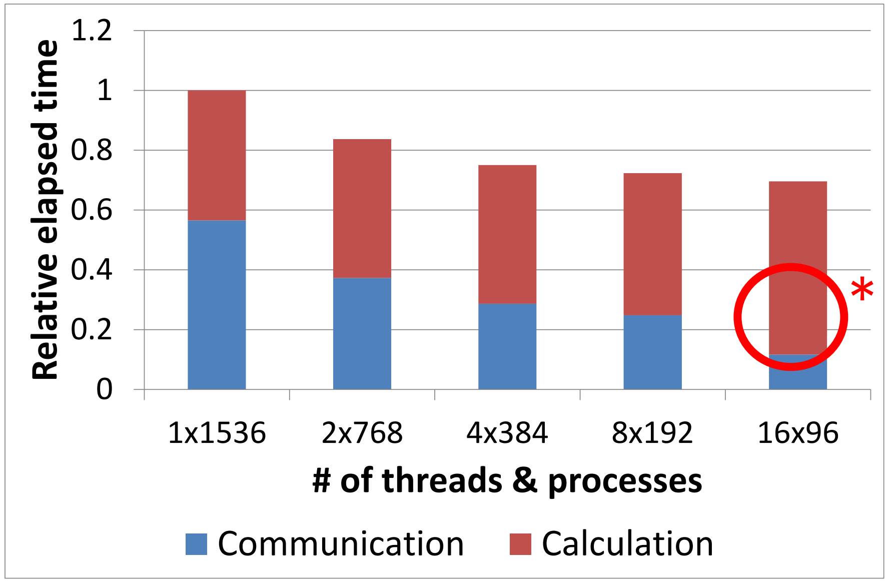
- Both are regional models, but have different computational characteristics
  - "IO Quilting" mechanism is used for WRF.
  - COSMO is memory intensive.
  - WRF is thread-parallelized by OpenMP, but COSMO is not.

- Key topics
  - Hybrid parallel execution
  - Load imbalance between processes & threads

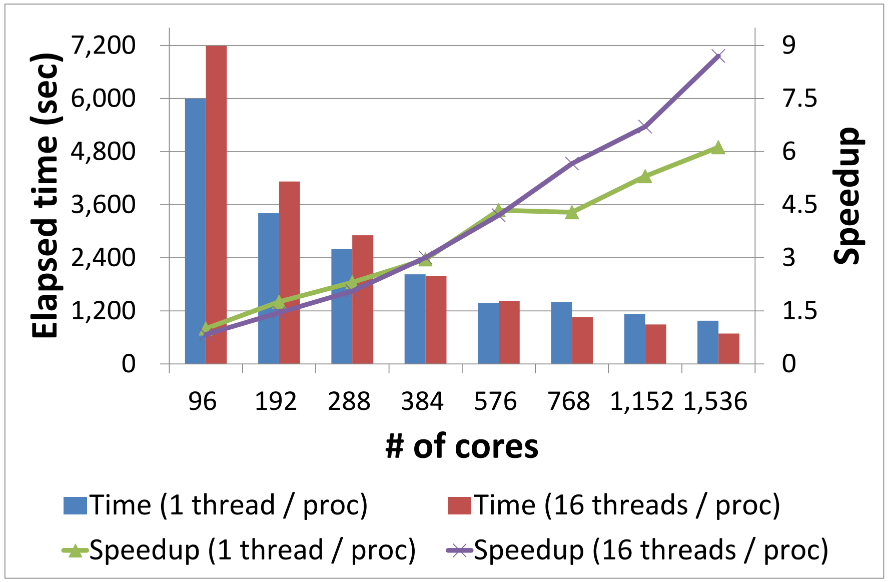
Advantage of hybrid parallelization on FX10

WRF V3.3.1

Reduction of Communication Cost



Difference of Node Scalability



(WRF V3.3.1, 1200x700x45 grids, by courtesy of CWB)

- Single thread spends much cost for communication
- The cost includes load imbalance between processes
  - Load imbalance ratio† of 1x1536 configuration reached to 21%, while 9.8% for 16x96
  - *Load imbalance between threads in a process is included in the calculation time.
- Reducing processes saves the cost and contributes much to node scalability

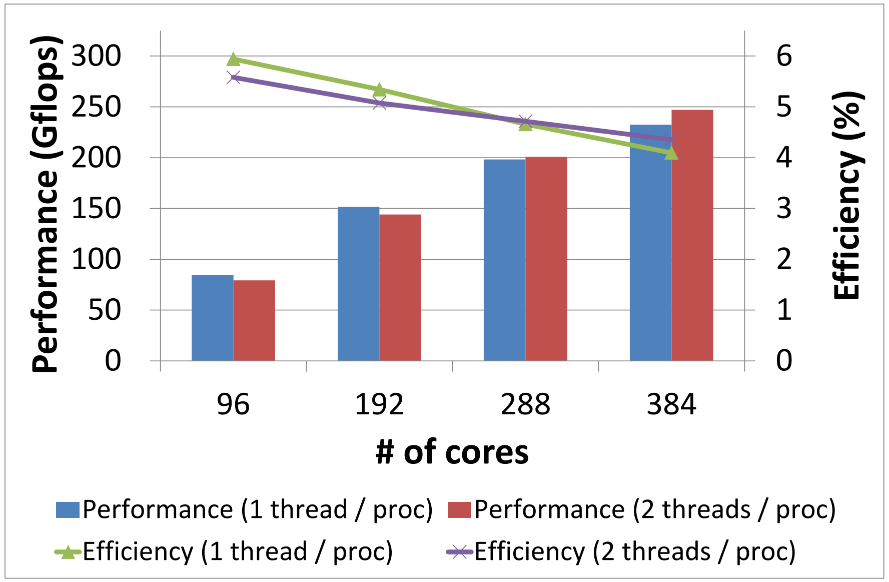
†:*LoadImbalanceRatio*＝(*MaxCalculationTime*÷*AvgCalculationTime*)－1

Automatic thread-parallelization on FX10

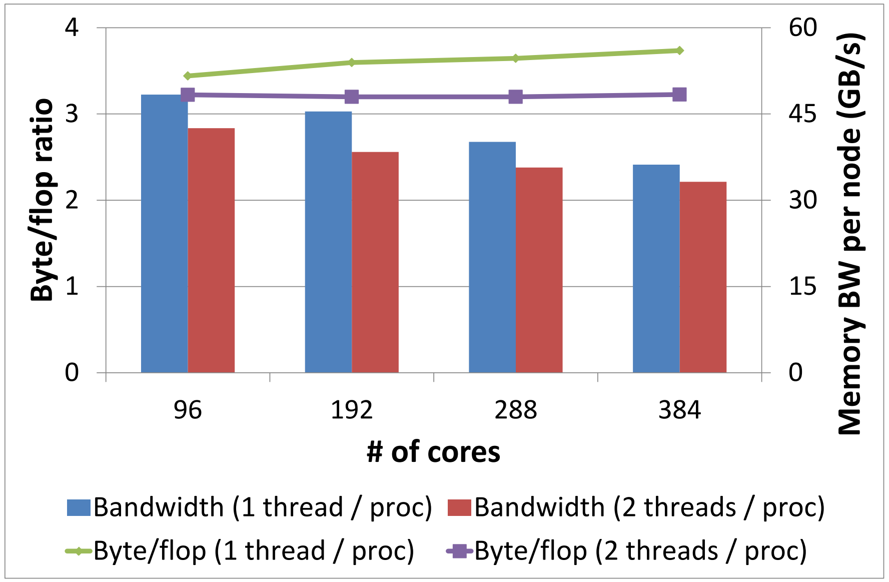
FUJITSU

COSMO-DE(RAPS 5.0)

Sustained Performance & Efficiency



Effective B/F Ratio & Memory BW



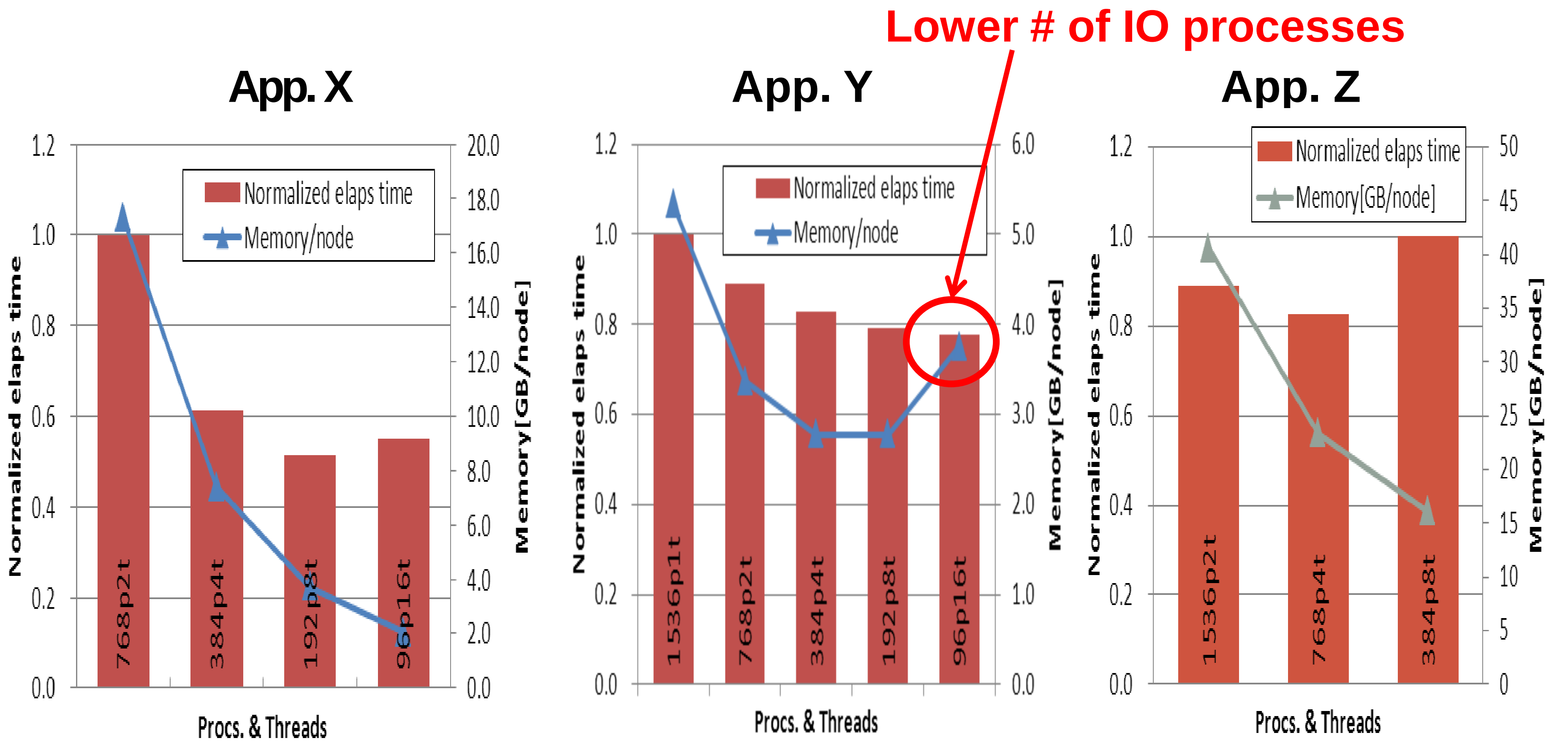
(COSMO RAPS 5.0, 421x461x50 grids)

- COSMO-DE is a single-threaded, memory intensive application
- VISIMPACT parallelizes with threads and improves performance when cores are increased
  - Thread parallelization reduces effective byte/flop ratio and utilizes memory bandwidth
  - For 384 cores, load imbalance ratio† was mitigated from 15% to 7.9%.

†:*LoadImbalanceRatio*＝(*MaxCalculationTime*÷*AvgCalculationTime*)－1

Memory usage of hybrid parallelized meteorological apps.

- Varying hybrid execution conditions on 96,192-node FX10
  - Memory usage is reduced by increasing the # of threads



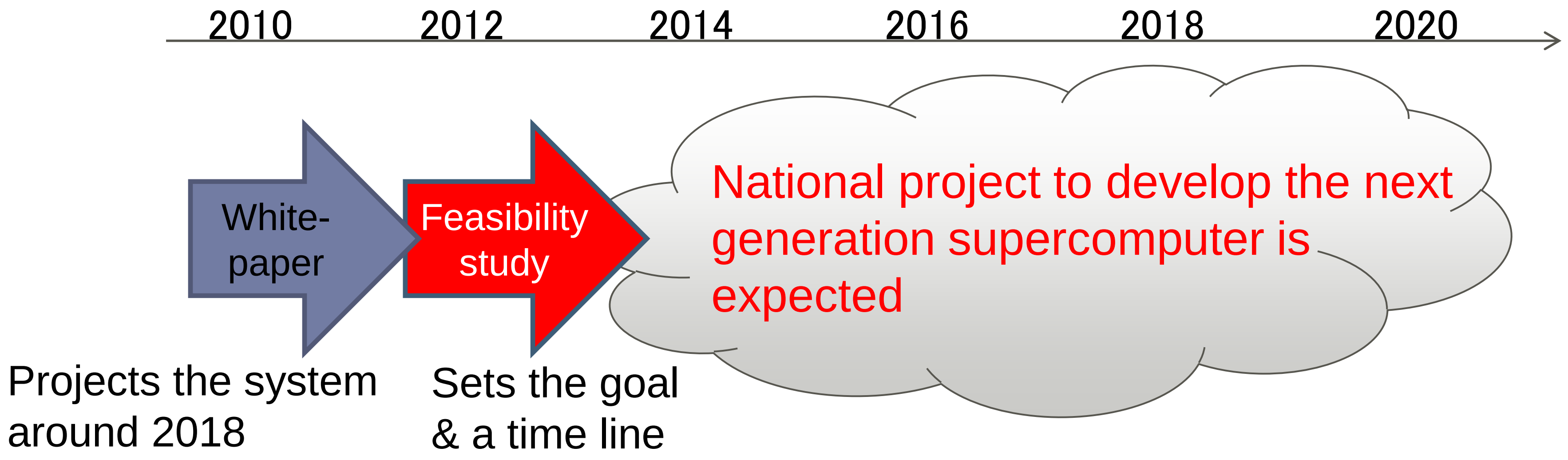
Summary of hybrid execution on VISIMPACT

- Hybrid parallelization with VISIMPACT leads good scalability by reducing load imbalance, communication cost, memory usage, and memory bandwidth requirement
- For further optimization
  - Load balancing between processes and between threads in a process
  - Controlling the effect of IO processes

Challenges towards Exascale computing

Approach toward exascale

- Fujitsu is developing a 100 Petaflops capable system as a midterm goal
- Participates two consecutive national projects for exascale
  - Fujitsu contributed to develop the whitepaper
    - “Report on Strategic Direction/Development of HPC in Japan”
  - Fujitsu has started two-year feasibility study to set the goal & schedule since July 2012


2010
2012
2014
2016
2018
2020
White-paper
Feasibility study
National project to develop the next generation supercomputer is expected
Projects the system around 2018
Sets the goal & a time line

Summary

- K computer and PRIMEHPC FX10 introduced new features
  - Original high performance CPU and a direct network interconnect Tofu
  - HPC-ACE & VISIMPACT for massively parallel environment
- Evaluations of real applications on K computer and FX10
  - High performance efficiency
  - Hybrid parallelization with VISIMPACT leads good scalability by reducing load imbalance, communication cost, memory usage, and memory bandwidth: all of these features are essential for the future
- Research and develop technologies for exascale step-by-step
  - 100PF capable supercomputer development is on the way
  - Feasibility study has started

FUJITSU

shaping tomorrow with you