Empowered by Innovation

NEC

NEC SX-6
User Training HLRS
16.3.2004

Dr. Th. Schoenemeyer, NEC HPCE
Application Specialist
tschoenemeyer@hpce.nec.com
What you should learn from this course

- How to evaluate the efficiency of your application
- If efficiency is low → How to tune your code
- How can the compiler help you?
- Solve Potential Porting Problems

Terminology on Performance

**PEAK PERFORMANCE**
Theoretical value which can be reached by a computer assuming all CPU functional units are active at the same time. This is hypothetical, and will never be achieved.

**SUSTAINED PERFORMANCE**
This is the computational speed that really shows up in the system for a given code and a given problem size. It clearly relates to the architecture, because they have a different → efficiency.

**EFFICIENCY**
Value in percent of theoretical peak performance which a system is able to deliver over any time.
Common values on PCC and MPP systems tend to be between 2 and 10%.
On Vector computers the efficiency is typically in the range of 30 to 50%.
Typical Weather Code

SX-6(8GF/CPu), 30% efficiency
Aladin: 30.8%

Assume: Run 3 day forecast within 1000 sec's elapsed time corresponding to 32 GFLOPs (sust.)

Now you want to run the same within half of the time. Take double of the hardware (sust. 64 GFLOPs).
The difference in sustained between scalar and vector grows and grows

What you should learn from this course

- How to evaluate the efficiency of your application
It's all about this table

<table>
<thead>
<tr>
<th></th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real Time (sec)</td>
<td>142.373845</td>
</tr>
<tr>
<td>User Time (sec)</td>
<td>122.693250</td>
</tr>
<tr>
<td>Sys Time (sec)</td>
<td>1.331073</td>
</tr>
<tr>
<td>Vector Time (sec)</td>
<td>65.89359</td>
</tr>
<tr>
<td>Inst. Count</td>
<td>19621582641.</td>
</tr>
<tr>
<td>V. Inst. Count</td>
<td>2791820381.</td>
</tr>
<tr>
<td>V. Element Count</td>
<td>216956074637.</td>
</tr>
<tr>
<td>FLOP Count</td>
<td>97760445434.</td>
</tr>
<tr>
<td>MOPS</td>
<td>1905.449870</td>
</tr>
<tr>
<td>MFLOPS</td>
<td>796.787481</td>
</tr>
<tr>
<td>VLEN</td>
<td>77.711330</td>
</tr>
<tr>
<td>V. Op. Ratio (%)</td>
<td>92.801205</td>
</tr>
<tr>
<td>Memory Size (MB)</td>
<td>496.031250</td>
</tr>
<tr>
<td>MIPS</td>
<td>159.923897</td>
</tr>
<tr>
<td>I-Cache (sec)</td>
<td>1.852328</td>
</tr>
<tr>
<td>O-Cache (sec)</td>
<td>15.546471</td>
</tr>
<tr>
<td>Bank (sec)</td>
<td>3.069632</td>
</tr>
</tbody>
</table>

Start Time (date) : 2002/06/24 11:32:19
End Time (date)   : 2002/06/24 11:34:42

What you should learn from this course

- If your ported code crashes → How to solve
- If efficiency is low → How to tune your code
First principle on how to get performance

- Writing fast code is writing parallel code
- Writing parallel code on SX does not start with MPI or OpenMP!
- Single thread performance has to be improved first
- Your goal is not scalability, but time to solution
- Learn how to exploit lower levels of parallelism
- Make your code visible - the compiler will do the (most) of the rest for you, by the right directives/options

---

**Agenda**

<table>
<thead>
<tr>
<th>Time</th>
<th>Session</th>
</tr>
</thead>
<tbody>
<tr>
<td>09:00 - 10:00</td>
<td>Introduction to the installed system at HLRS</td>
</tr>
<tr>
<td>10:15 - 11:15</td>
<td>Theory I: Differences SX-5/SX-6, Basics on vectorization</td>
</tr>
<tr>
<td>coffee break</td>
<td></td>
</tr>
<tr>
<td>11:30 - 12:30</td>
<td>Theory II: Vectorization and optimization examples</td>
</tr>
<tr>
<td>lunch break</td>
<td></td>
</tr>
<tr>
<td>13:30 - 15:00</td>
<td>Theory III: Indirect addressing, General Strategy for code tuning</td>
</tr>
<tr>
<td>coffee break</td>
<td></td>
</tr>
<tr>
<td>15:15 - 16:30</td>
<td>Theory IV: Most important compiler switches and directives</td>
</tr>
</tbody>
</table>
THEORY I

- Hardware Overview
- Basics on Vectorization

NEC Productline

- NEC TX7 based on IA-64 Technology
- NEC-Cluster based on IA-32
- NEC SX-Series based on Vector CPU
The TX-7 Series Server family uses the Intel® Itanium®2 Processor

• up to 1.5 GHz clock speed
• up to 32 CPU’s
• up to 256 GB Main Memory
• 6.4 GB/s system bus bandwidth
• 3-6 MB integrated L3 cache
• 64-bit memory addressing
• Peak performance up to 6 GFLOPS per processor
• Linpack: 171.61 GFLOPs with full node
  (15th July 2003)
• HP-UX, Windows, NEC Linux

Overview NEC TX-7

NEC IA-32 Cluster

• NEC Cluster solution for crash simulation
  – 4 fully equipped Racks
  – 256 Xeon CPUs
• Intel Xeon based Server
• Available: Gigabit-Eth., Myrinet, Quadrics, Infiniband
• System Support from NEC
NEC’s SX-Series: Innovation since 1983

1983
SX 1/2 Series
The first computer in the world

1989
SX-3 Series
Shared memory, multi-function processor

1994
SX-4 Series
High sustained performance

2001
SX-5 Series
High sustained performance

International Recognition
• Eckart-Mauchly Award of IEEE for CPU architecture
• IEEE and ACM Award for innovation in packaging technology

NEC’s Goals: The Vector Architecture

• Highest single CPU performance
• High sustained performance
• Extremely high bandwidth to Memory
• High performance with comparatively small number of CPUs
• Same manufacturing technology like conventional CPUs
• Provide Software to use it effectively
**SX-5 The Specifications**

- **Single Node**
  - Up to 128 GFLOPS
    - With 16 x 8 GFLOPS Processors
  - Up to 128 GBytes Shared Main Memory
- **Multi Node**
  - Up to 4 TFLOPS
  - Up to 32 Nodes Using SX-5 IXS
  - Up to 512 Processors
  - Up to 4 TBytes Main Memory

**SX-5 HLRS Installation**

- **2 Nodes**
  - 128 GFLOPS
    - With 32 x 4 GFLOPS Processors
  - 80 GBytes Shared Main Memory
Our current High-End product SX-6+

- High Performance Vector Supercomputer
- 9 GF / CPU
- 72 GF/ Node
- Maximum 128 GB Main Memory per node
- Maximum 128 nodes
- Maximum 1024 CPUs, max 9.2 TFLOPS Peak Perf.
- Ultra-High bandwith shared memory subsystem
- Internode crossbar Switch
- 1 TB/s maximum interconnect bandwidth per system

SX-6+ Technology Breakthrough

- single chip vector processor
- unprecedented memory bandwidth per chip: up to 36 Gbytes/s
- low latency between execution control and vector pipes
NEC Vector CPU Evolution

**SX-4**
- 457 x 386 mm
- 2 GFLOPS at 8.0 ns
- 0.35µ CMOS
- 37 Chips

**SX-5**
- 225 x 225 mm
- 8 GFLOPS at 4.0 ns
- 0.25µ CMOS
- 32 Chips

**SX-6+**
- 11,0 x 11,5 mm
- 9 GFLOPS at 1.77 ns
- 0.15µ CMOS
- Single Chip Processor

---

NEC Memory Evolution

**SX-4**
- 457 x 386 mm
- up to 32 Mb SDRAM
- max. 8 Modules/Card
- 256MB / Card
- 1 Card/CPU

**SX-5**
- 457 x 386 mm
- 64 - 128 Mb SDRAM
- max. 16 Modules/Card
- up to 8 GB / Card
- 1 Card/CPU

**SX-6+**
- 105 x 176 mm
- 4GB / Card
- 512 Mb DDR-SDRAM
- 4 Cards/CPU
- = 16 GB/CPU
CPU-Memory Connection full node SX-6+

8 CPU (72 GFLOPS)

RCA: Memory Interface for IOP & IXS

32 MMU Cards (128 GB)

Single Node Small Configuration SX-6

Basic Cabinet 1000 mm

CPU CPU CPU
CPU CPU CPU
CPU CPU CPU
CPU RCA CPU

Inter-node network

4PCIBx3PCIx1slots (133MB/s PCI)

450mm 200mm 450mm

1.8m 1.1m 1.1m
Vector Unit Architecture SX-5

- Multiple Vector Parallel Pipelines
- 64 (32) Pipelines
  - Add-Shift x 16 (8)
  - Multiply  x 16 (8)
  - Logical  x 16 (8)
  - Divide   x 16 (8)
- Each Instruction Uses 16 Pipelines
  - Automatic Hardware Parallelism
- Concurrent Pipeline Set Operation

SX-5 Vector Unit Block Diagram

cycle = 4 ns
1 Func. U. = 8*250 MHZ= 2 GFLOP's
2 Func. U. conc. = 4 GFLOP's
SX-6+ CPU

One Functional Unit: 565 Mhz x 8 = 4.5 GFLOPs

8 way pipes @ 565 MHZ

SX-6+ Processor Architecture

One Functional Unit: 565 Mhz x 8 = 4.5 GFLOPs
Scalar CPU: SX-5/6

- is more important than one would assume!
- features
  - 2 x 64 kByte Cache (Instructions, Data)
  - 8 kByte Instruction Buffer
  - 128 64-bit registers
  - 4 instructions per cycle
  - instruction reordering (data flow control)
  - branch prediction
  - 2 fmult/fadd/fdiv pipes, 2 integer pipes
  - double word load (SX-4: single)
  - 565 MHZ -> 1,13 GFLOPs Peak
To transfer data at high speed between main memory and vector registers, main memory is divided into a maximum of 4096 independent modules (called banks);

Parallel reading or writing is enabled for each different bank (called interleaving or interlacing). Since only one read or write process is performed at a time in each bank, if two or more read or write requests are made to one bank, the later requests must wait for completion of the preceding request. The fall-off in speed is due to bank conflict.

Main Memory - SX-6

A SX-6 node can be used as a SMP-System or distributed memory system

Several SX-6 nodes can be used to run a MPI-job, using 1-8 MPI-processes per node

Current OS Version is SUPER-UX 13.1
Memory Bandwidth I

Stream Triad

<table>
<thead>
<tr>
<th>Processor</th>
<th>MB/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Itanium 800 MHZ</td>
<td>1402</td>
</tr>
<tr>
<td>Pentium 4 (1.4 GHz)</td>
<td>1575</td>
</tr>
<tr>
<td>p690 (16 CPUs)</td>
<td>2005</td>
</tr>
<tr>
<td>p690 turbo (32 CPUs)</td>
<td>2550</td>
</tr>
<tr>
<td>SX-6i</td>
<td>27339</td>
</tr>
<tr>
<td>SX-6 (1 CPU)</td>
<td>31982</td>
</tr>
</tbody>
</table>

Peak = 32 GB/s

Memory Bandwidth II

Stream Triad

<table>
<thead>
<tr>
<th>Processor</th>
<th>MB/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>p690 (16 CPUs)</td>
<td>2005</td>
</tr>
<tr>
<td>p690 turbo (32 CPUs)</td>
<td>25501</td>
</tr>
<tr>
<td>SX-6i</td>
<td>27339</td>
</tr>
<tr>
<td>SX-6 (1 CPU)</td>
<td>31982</td>
</tr>
<tr>
<td>SX-6 (8 CPUs)</td>
<td>213024</td>
</tr>
</tbody>
</table>
Basics on Vectorization

IEEE Format (float0) !!!

- **4 Byte:**
  - about 7 digits
  - $10^{-38} - 10^{38}$

- **8 Byte:**
  - about 16 digits
  - $10^{-308} - 10^{308}$

- **16 Byte:**
  - about 32 digits
  - $10^{-308} - 10^{308}$
  - **not vectorizable!**
### Word-Length

Please have always in mind

<table>
<thead>
<tr>
<th></th>
<th>Real</th>
<th>Real*4</th>
<th>Integer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cray</td>
<td>8</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>SX-5/6</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>SX-5/6 (-ew)</td>
<td>8</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>SX-5/6 (dbl4)</td>
<td>8</td>
<td>8</td>
<td>4</td>
</tr>
<tr>
<td>Workstation</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>WS (-r8)</td>
<td>8</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>SX-5/6 (idbl4)</td>
<td>8</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

### Compatibility

<table>
<thead>
<tr>
<th></th>
<th>SX-4</th>
<th>SX-5</th>
<th>SX-6</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ok</td>
<td>ok</td>
<td>! -sx5</td>
</tr>
<tr>
<td></td>
<td>! -sx4</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

NEC – Theory I
I–19
Basics on vectorization

First very important principle: segmentation of operations and pipelining

Segmentation & Pipelining

- Operations are decomposed into segments
- Example: add Floating point number

- compare exponent
- Shift mantissa
- Add mantissa
- Select exponent and normalize

1.14e9  -  2.78e8
1.14e9  -  0.278e9
0.862e9
8.62e8
Pipelining (cont.)

- no pipelining:
  - comp. exponents
  - shift mantissa
  - add mantissa

- with pipelining:
  - comp. exponents
  - shift mantissa
  - add mantissa

Segmentation Pipelining

NEC – Theory I
I–21
Pipelining (cont.)

- Superscalar pipeline (RISC):
  - Issue
  - Latency
  - Vector register length

- Vector pipeline:
  - Comp. exponents
  - Shift mantissa
  - Add mantissa

Simple Example: Data Parallelism

- 'Vector Loop': Data Parallel
- Independent data

F77:
```
Real a(n), b(n), c(n)
do i = 1, n
  a(i) = b(i) + c(i)
end do
```

F90:
```
Real, dimension(n): a, b, c
a = b + c
```

N=256, Loop will be divided in 8 parts
Pipelining (cont.)

This is the basics of vectorization!!

8-fold parallel on SX-6

8 X

Optimizations means: Keep these pipelines busy

SX-6 CPU

8 way pipes @ 565 MHZ

Vector- Data 64 x 256

Vadd / Vshift
Vmull
Vlogical
Vdiv

Vregs 8 x 256

Memory

32 GB/s

NEC – Theory I
I–23
Let's start with a quick test

Real, dimension(n): a, b, c
do i = 1, n
  a(i) = b(i) + c(i)
extend do

Real, dimension(n): a, b, c
do i = 1, n
  a(i) = b(i) + a(i-1)
extend do

• Vector loop: Data Parallel
  
  each loop iteration can be executed in parallel

• Scalar loop: Recursion

  Current iteration depends on
  The result of the previous one
What is peak performance of a full SX-6 node?

8*9 = 72 GFLOPs

How can we calculate this?

8 (CPU) * 2 (FU) * 8 (PIP)*565 Mhz

What is the maximum vector length and why is it important to use it?

256 words

Vectorization

REAL, DIMENSION (51200) :: A, B, X

A = 5.25

B = 0.0

X = 10.4 + CONSTANT

B = A + X ** 2

All loops were vectorized.

note: Fortran 90 notation.
Vectorization

DO K = 2, KOUNT - 1
X(K+1) = X(K) + Y(K-1)
END DO

Is it vectorizable or not? Why?

Recurrence of X
Not vectorized

Vectorization

program quiz1
REAL, DIMENSION(100) :: A, B, C
DO I = 99, 1, -1
    B(I) = A(I + 1)
    A(I) = C(I)
END DO
end program

Is it vectorizable or not? Why?
Vectorization

1 program quiz1
2 REAL, DIMENSION(100) :: A, B, C
3 DO I = 99, 1, -1
 4       B(I) = A(I + 1)
 5       A(I) = C(I)
 6 END DO
 7 !CDIR NODEP
 8   do i = 1, 99
 9     a(100-i) = c(100-i)
10     b(100-i) = a(101-i)
11   end do
12 end program

This is what the compiler does

Program quiz2
real, dimension (1000) :: a,b
   DO I = 1, 240
50 CONTINUE
   A(I) = B(I) + 2.45
   TEST = TEST + INC
   IF( TEST == SOMETHING ) GO TO 50
END DO
end program

Does it vectorize or not? Why?
Vectorization

program quiz2
real, dimension (1000) :: a,b
  DO I = 1, 240
    50 CONTINUE
    A(I) = B(I) + 2.45
    TEST = TEST + INC
    IF( TEST == SOMETHING ) GO TO 50
  END DO
end program

No - Contains a backwards branch

Vectorization

program quiz3
REAL, DIMENSION(1000) :: A, B
  DO I = 2, 999
    B(I) = A(I - 1)
    A(I) = B(I-1)
  END DO
end program

Does it vectorize or not? Why?

No, recurrence of B
Vectorization

program quiz3
REAL, DIMENSION(1000) :: A, B
DO I = 2, 999
   B(I) = A(I - 1)
   A(I) = B(I+1)
END DO
end program

Does it vectorize or not? Why?
Yes, no recurrence of B