POWER5 and HPS Programming Strategies: Resources and Special Effects

Charles Grassl
IBM
July, 2004

© 2004 IBM

Agenda

- Part 1: Macro Architecture and Resources
  - Architecture Trends
  - Impact on Programming
  - POWER4 Processors
  - System Resources
- Part 2: Special Effects
  - Large pages
  - Memory affinity
  - Process binding
  - Simultaneous Multi-Threading (SMT)

© 2004 IBM Corporation

Part 1: Macro Architecture and Resources

- Architecture Trends
  - Impact on Programming
- POWER4 Processors
  - System Resources

© 2004 IBM Corporation

Underlying Technology

- Transistor construction
- Lithography
- Material Science
- Electronic structures
- Circuit density
- Fabrication

© 2004 IBM Corporation

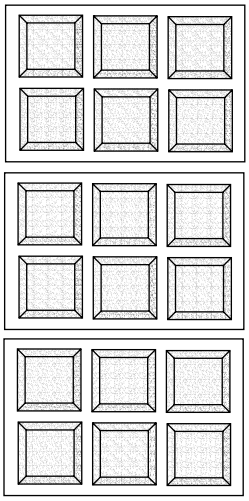
Integrated Circuit Trends

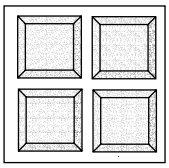
- Smaller line widths
  - Nearly 0.1 micron
  - Lower voltages
- Higher transistor count per chip
  - ~200 millions transistors per chip
- Area not growing fast
  - ~400 sq. mm
- High signal speed
  - SiO2
  - Copper
  - Low - k dielectric

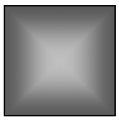
© 2004 IBM Corporation

Increasing Transistor Density

Processor, 1990



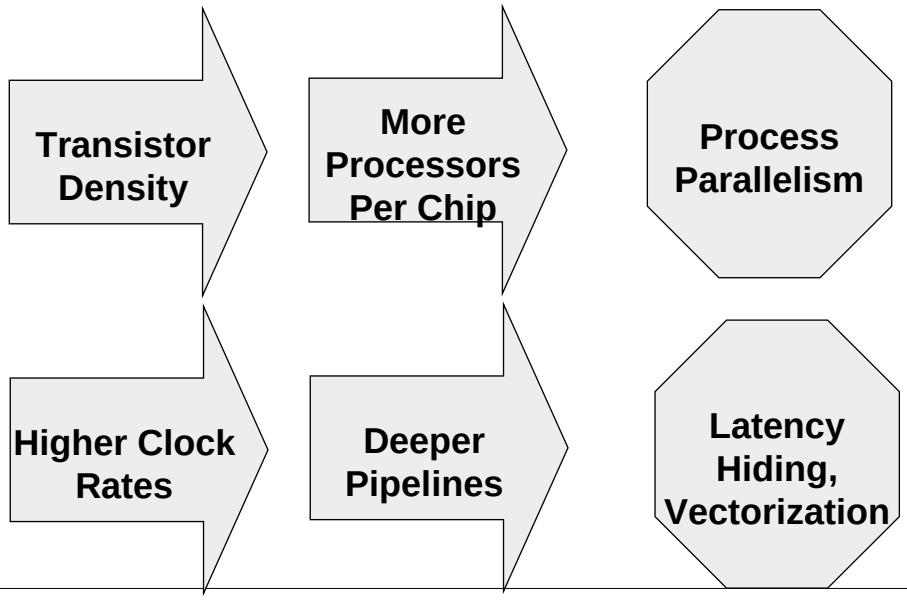




Processors, 2002

© 2004 IBM Corporation

Effect of Transistor Technology



© 2004 IBM Corporation

Effects on Software

- More Processors
  - Shared Memory Parallelism
    - Detection
    - Explicit
- Compilers
  - Automatic parallelism
  - OpenMP
- Scalability
- Communication
  - Barriers
- Deep Pipelines
- More registers
- Vectorization techniques
- Latency hiding
- Instruction Level Parallelism

© 2004 IBM Corporation

Microprocessor Architecture Trends

- Multithread
  - Cray MTA
  - Intel
    - Pentium
    - Xeon
  - Sun
  - IBM
    - P660
    - POWER5
- Multiple processors per chip
  - POWER4
  - POWER5
  - IBM Blue Gene
  - HP PA
  - Future Itanium

© 2004 IBM Corporation

Other Trends

- Smaller processors
  - Blue Gene
- Tighter coupling
- Hardware synchronization
- Clustered processors
- Virtualization

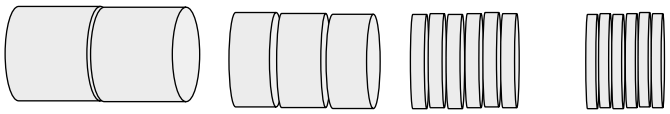
© 2004 IBM Corporation

POWER4 Performance Features

- Deep pipelines
- Registers
  - 32 ISA registers
  - Rename registers
- Caches

© 2004 IBM Corporation

POWER Architecture FMA Trend



	POWER2	POWER3	POWER4	POWER45
Clocks periods	2	3	6	6
Clock Rate	125 MHz	375 MHz	1.3 GHz	2 GHz
Transit Time	16 nanosec	8 nanosec	4.4 nanosec	3 nanosec
Registers	32	32	32	32

© 2004 IBM Corporation

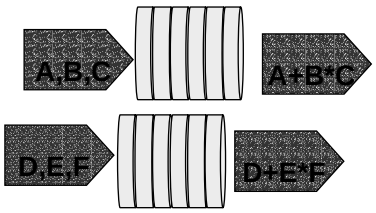
POWER4 Floating Point Functional Units

- Two symmetric floating point units
  - Divide and square-root sub-units
- Double precision (64-bit) data path

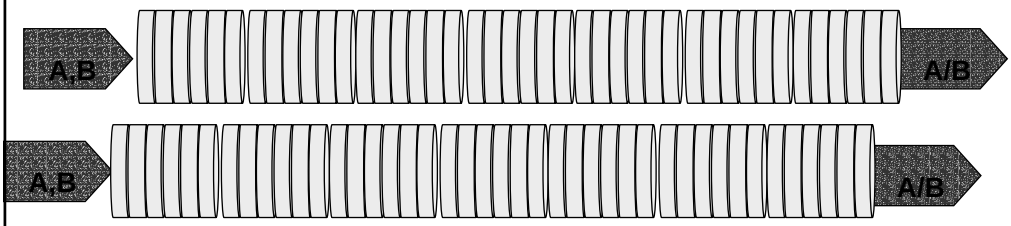
Instruction	Single	Double	Pipelined
Fma	6	6	YES
Fdiv	32	32	No
Fsqrt	38	38	No

© 2004 IBM Corporation

Pipelined Functions



	Results per clock period
Fma	12/6
Divide	2/32
Sqrt	2/38



© 2004 IBM Corporation

Deep Pipelines

- Operations limited by functional unit transit time:
  - Divide
  - Square root
  - Intrinsic functions
  - Recursion

© 2004 IBM Corporation

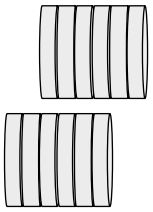
Attaining Peak Speed (Demonstration Kernel)

- Independent scalar triads
- Test code:

```
do i=1,n
   y1 = x1 + s*x1
   y2 = x2 + s*x2
  ....
   x1 = y1 + s*y1
   x2 = y2 + s*y2
  ....
end do
```


© 2004 IBM Corporation

Fma Functional Unit



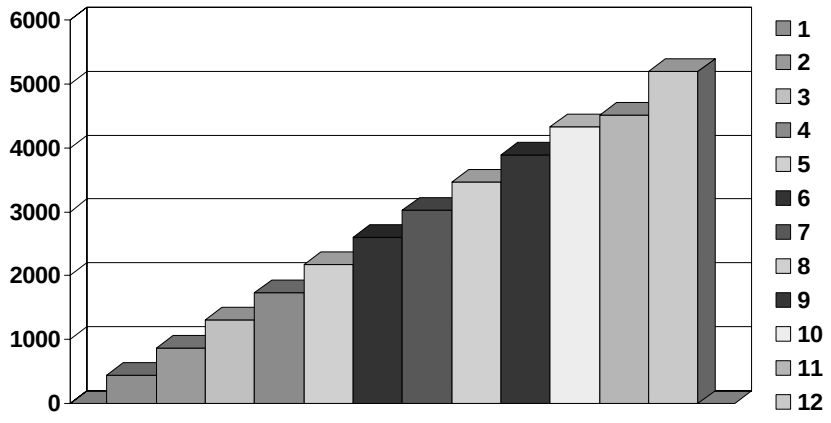
Clock Periods	6
Results per Clock	4
Results in Transit	24

Triads	Rate (Gflop/s)
1	433
2	866
4	1733
6	2600
8	3466
10	4333
12	5200

1.3GHz POWER4

© 2004 IBM Corporation

Independent Triad Performance



1.3GHz POWER4

© 2004 IBM Corporation

Effect of Registers

Processor	DGEMM Speed (Mflop/s	% Peak
POWER2	600	98
POWER3	1480	98
POWER4	3500	70
POWER5	~6000	~90

- POWER4 registers
  - Architecture: 32
  - Rename: 72
- POWER5 registers
  - Architecture: 32
  - Rename: 110

© 2004 IBM Corporation

Cache Architecture

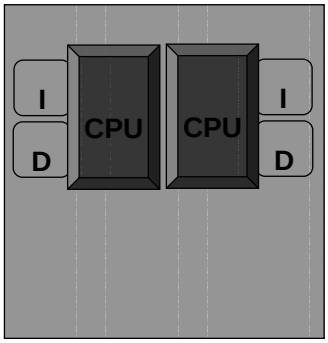
	POWER4	POWER5	Benefit
L1 Cache	2-way Associative FIFO	4-way Associative LRU	Improved L1 Cache performance
L2 cache	8-way Associative 1.44MB	10-way Associative 1.9MB	Fewer L2 Cache misses Better performance
L3 Cache	32MB 8-way Associative 118 Clock Cycles	36MB 12-way Associative ~80 Clock Cycles	Better Cache performance
Memory Bandwidth	4GBbyte / sec / Chip	~16Gbyte / sec / Chip	4X improvement Faster memory access

© 2004 IBM Corporation

* All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

L1 Cache

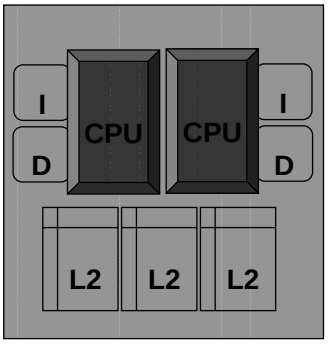
- **Size:**
  - 32 kbyte data
- **Bandwidth:**
  - 2 words x 8 bytes x clock_rate (Same as POWER3)
  - 20 Gbyte/s
- **Latency:**
  - 1 clock period



© 2004 IBM Corporation

L2 Cache

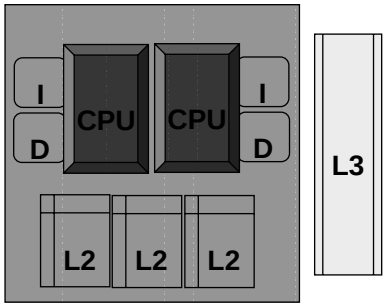
- **Size:**
  - POWER4: 1.45 Mbyte
  - POWER5: 1.9 Mbyte
- **Shared by pairs of processors**
- **Three slices**
- **Bandwidth:**
  - 100 - 200 Gbyte/s
    - 32 bytes * 3 slices / clock
- **Latency:**
  - 8 - 10 clock periods



© 2004 IBM Corporation

L3 Cache

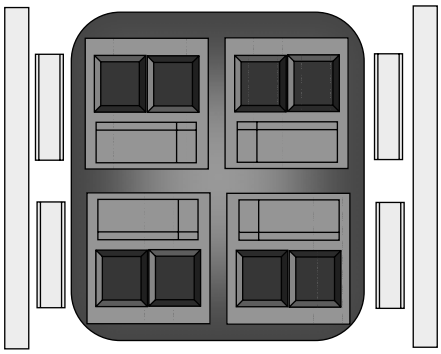
- **Size:**
  - POWER4: 32 Mbyte
  - POWER5: 36 Mbyte
- **Bandwidth:**
  - 11 Gbyte/s
- **Latency:**
  - POWER4: ~150 clock periods
  - POWER5: ~100 clock periods
- **POWER4: shared across system**
- **POWER5: Shared by pairs of processors**



© 2004 IBM Corporation

Memory: Single MCM

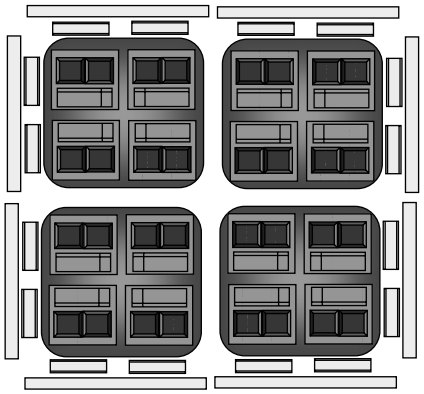
- **Size:**
  - Up to 128 Mbyte per MCM
- **Bandwidth:**
  - 11 Gbyte/s per physical L3
- **Latency:**
  - ~300 clock periods



© 2004 IBM Corporation

Memory: Multiple MCMs

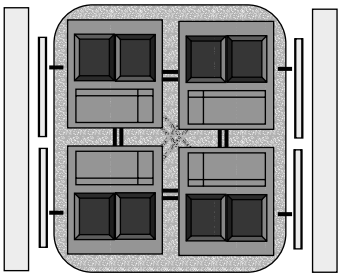
- **Size:**
  - **Up to 512 Mbyte**
- **Bandwidth:**
  - **16 x 11 Gbyte/s**
  - **50 Gbyte/s attainable**
- **Latency:**
  - **~300 clock periods**

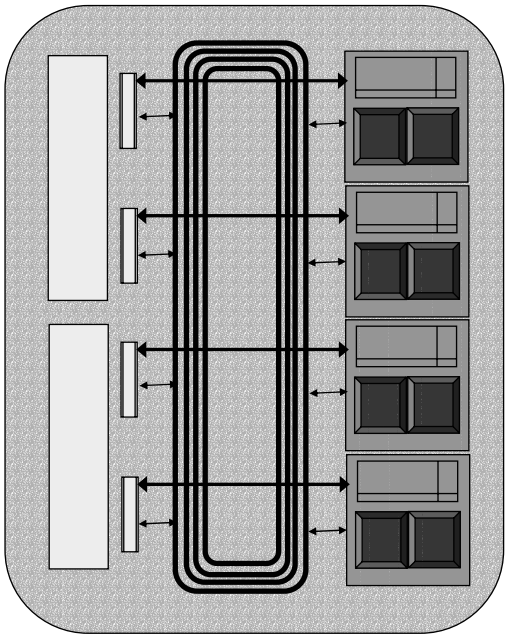


© 2004 IBM Corporation

Intra-module Communications

- **4 busses**
- **16 bytes**
- **2:1 clock rate**
- **41 Gbyte/s @ 1.3 GHz**

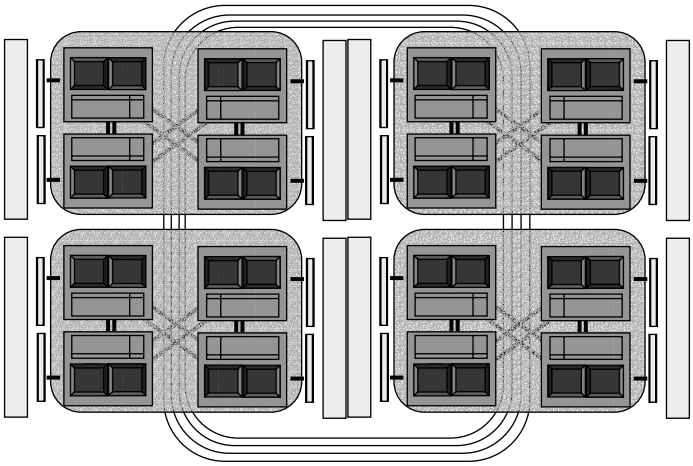




© 2004 IBM Corporation

Inter-module Communications

- 4 busses
- 8 bytes
- 2:1
- 21 Gbyte/s @ 1.3 GHz



© 2004 IBM Corporation

Part 2: Special Effects

- Large Pages
- Memory Affinity
- Process Binding
- Simultaneous Multi-Threading

© 2004 IBM Corporation

Large Page Effects

- Additional bandwidth
- Additional Translation Lookaside Buffer (TLB) coverage
- Poor interactive performance
  - Scripts
  - Compile
  - Forks

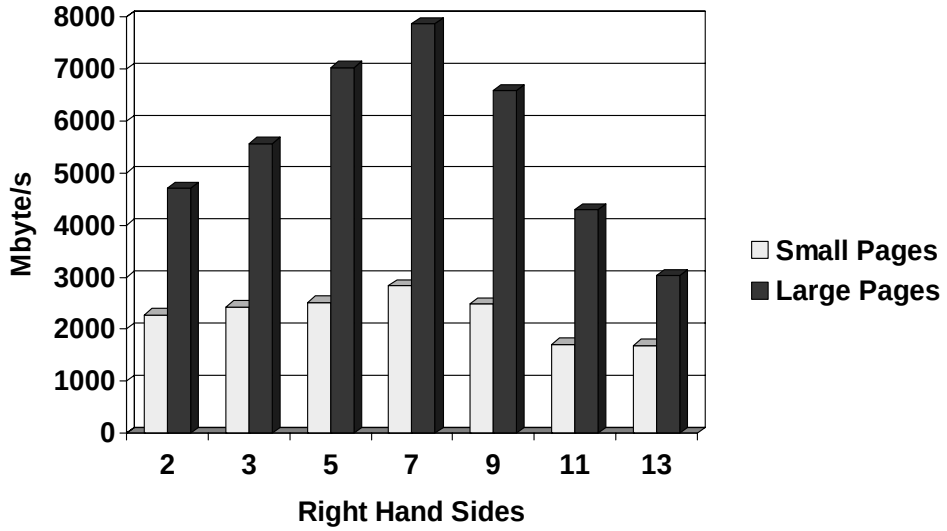
© 2004 IBM Corporation

Large Page Motivation

- Enhance memory bandwidth
  - Prefetch performance is limited by page size
- Bandwidth increase of up to 3x
- Enhance Translation Lookaside Buffer (TLB) coverage
  - POWER3: 128x2 TLB entries
  - POWER4: 1024 TLB entries

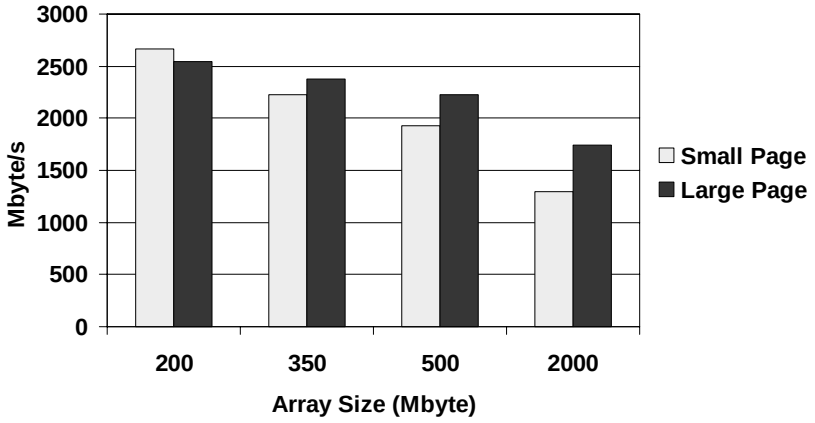
© 2004 IBM Corporation

Memory Bandwidth



© 2004 IBM Corporation

TLB Span



A(:) = A(:) + B(index(:))

© 2004 IBM Corporation

Large Page Usage

- Detection
  - Old:
    - $ vmtune
    - ….. –lp …..
  - AIX 5.2:
    - $ vmstat –l # Small L
    - …….. alp flp
    - 50 650

© 2004 IBM Corporation

Large Page Usage

- Loader:
  - $ xlf .... -blpdata -o a.out
- ldedit:
  - $ /usr/ccs/bin/ldedit -blpdata a.out
- Environment variable:
  - LDR_CNTRL=LARGE_PAGE_DATA={Y,N,M}
  - Y: Yes ("advisory" or "maybe") mode
    - Use large pages if available
    - Mode used by loader and ldedit
  - N: No large pages
  - M: Mandatory
    - Do not run if large pages not available

© 2004 IBM Corporation

Large Page Usage

Method	Usage	Comment
Linker	Xlf –blpdata …	Advisory
ldedit	Ldedit –blpdata …	Advisory
Env. Variable	LDR_CNTRL= LARGE_PAGE_DATA = [Y,M,N]	Y: Advisory M: Mandatory N: No

© 2004 IBM Corporation

Expected Application Performance

- Memory intensive applications
  - Single CPU performance increase: typically 5-20%
- Multiple CPU performance not as much
  - Nothing to gain if all memory bandwidth already used

© 2004 IBM Corporation

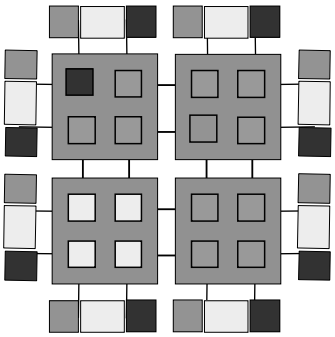
Memory Affinity

- Allocate memory pages to local module
- Access to local boards is robust
  - Little contention
  - Except for processor pairs on chip
  - Lower latency
- Access to remote modules uses buses
  - Contention
  - Slightly higher latency (~10%)

© 2004 IBM Corporation

Default Memory Allocation

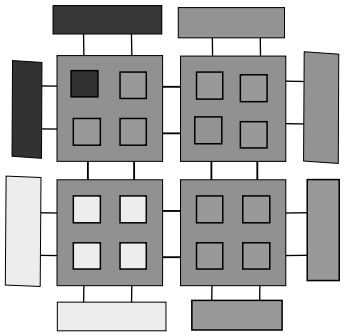
- Pages allocated to all modules
- Independent of process location (approximately)
- Periodic placement ("approximate round robin")



© 2004 IBM Corporation

Memory Affinity

- Pages allocated preferentially to local module



© 2004 IBM Corporation

Memory Affinity Concerns

- Less system-wide contention
  - Works well for MPI
  - Memory localization
- Difficulty with threads
  - New threads may require references to remote memory
  - "First touch" strategies

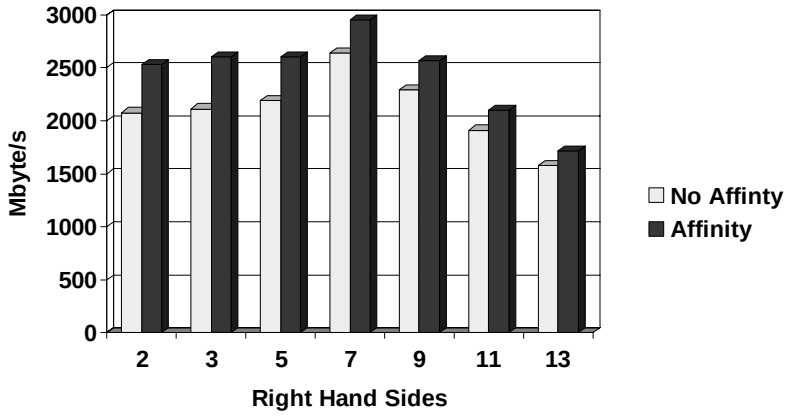
© 2004 IBM Corporation

Memory Affinity Implementations

- Export MEMORY_AFFINITY=MCM

© 2004 IBM Corporation

Memory Bandwidth: Affinity



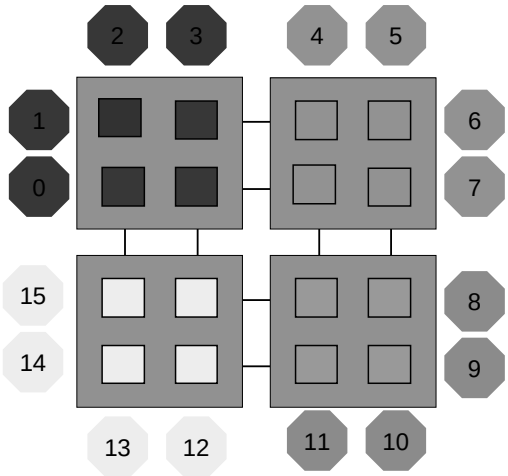
© 2004 IBM Corporation

Process Binding

- **Memory does not migrate**
  - **Processes migrate over time**
  - **Some processor affinity is in effect from AIX**
- **Process binding can be detrimental in shared usage**
  - **Bound process might share processor with other users processes**

© 2004 IBM Corporation

Process Binding



© 2004 IBM Corporation

Process Binding

- Binds or unbinds the kernel threads of a process to a processor
  - /usr/sbin/binprocessor
- Examples:

```
$ bindprocessor -q
The available processors are:  0 1 2 3 4 5 6 7
$ ps
PID    TTY  TIME CMD
35036  pts/6  2:02 cg.B
44268  pts/6  0:00 -ksh
$ bindprocessor 35036  4
```


© 2004 IBM Corporation

Task and Thread Binding

- Library call:
  - Bindprocessor(BINDTHREAD, (int) thread_number, (cpu_t) CPU_number);
  - Binds TID to specific processor

© 2004 IBM Corporation

bindUtils library

- NOT A PRODUCT
- Library utility which maps tasks (MPI) or threads (OpenMP) to specific processors
- Does not map memory (affinity)
- Usage:
  - $ Xl{fc} .. –l {OMP,MPI,MPIOMP}bin
  - $ export TARGET_CPU_LIST="0 2 4 6”
  - $ a.out

© 2004 IBM Corporation

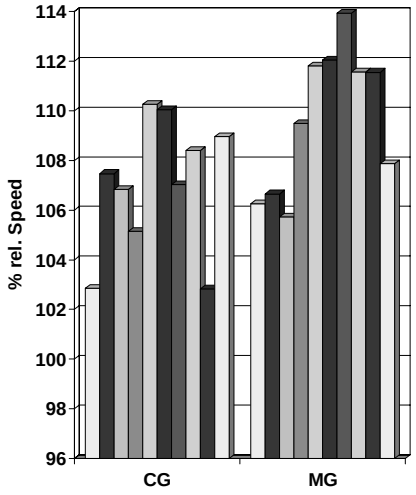
bindUtils: MPI and OpenMP

- MPI:
  - Uses MPI_Init interface
    - Builds list of hosts
    - Calls bindprocessor for each task
- OpenMP
  - Uses called bind(ierr) procedure
  - #pragma omp parallel region
    - Calls bindprocessor for each thread

© 2004 IBM Corporation

bindUtils: MPI Example

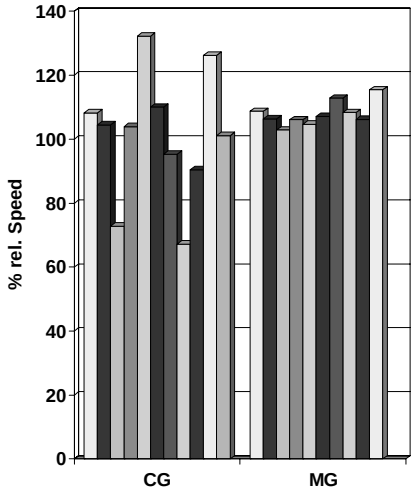
- $ mpxlf ... -l MPIbind
- Test cases:
- Program CG and MG
- 8 MPI tasks on 16 CPU node
- CG: 7% faster
- MG: 10% faster



© 2004 IBM Corporation

bindUtils: OpenMP Example

- Source code:
- CALL BIND (IER)
- $ xlf_r -lOMPbind.a ...
- Test cases:
- Program CG and MG
- 8 OMP threads on 16 CPU node
- CG: 1% faster
- Large variance
- MG: 8% faster



© 2004 IBM Corporation

Simultaneous Multi-Threading (SMT)

- **Operating system "sees" twice as many processors**
  - **Example: 64 processor POWER5**
    - **128 running processes**
- **Shared resources**
  - **Memory**
  - **Cache**
  - **Registers**
  - **Functional units**
- **Concurrent thread exectution**

© 2004 IBM Corporation

SMT: Control

- **System-wide feature**
  - **Usually turned on or off by system administrator**
- **Smtctl command:**
  - View: /usr/sbin/smtctl
  - Turn off: /usr/sbin/smtctl -m off now
  - Turn on: /usr/sbin/smtctl -m on now

© 2004 IBM Corporation