

Logical Partitioning (LPAR) Performance on Power Systems with POWER4, POWER5 and POWER6



Ron Young, and Vernon Pahl
POWER Systems Virtualization Performance
May 2008

LPAR Performance on Power Systems with POWER4™, POWER5™ and POWER6™
©Copyright 2008 IBM. All rights reserved.

Table of Contents

1	Overview	4
2	Systems under Test	4
2.1	Hardware Overview and Memory Allocation.....	5
2.2	Scenarios / Workloads	7
3	Whole Processor Partitioning	8
3.1	Results Overview	8
3.2	Model 890 and Model 595 32-way Scenarios	8
3.2.1	Model 890 and Model 595 Concurrent Partitions: Performance Summary	9
3.3	Model 890 and Model 595 Shared Processor Scenarios.....	10
3.3.1	Model 890 and Model 595 Dedicated/Shared Partitions: Performance Summary	10
3.4	Model 825 6-way Scenarios.....	11
3.4.1	Model 825 Dedicated Partitions: Performance Summary	12
3.5	Model 570 4-way Scenarios.....	12
3.5.1	Model 570 Dedicated/Shared Partitions: Performance Summary	13
3.6	Tips	14
3.7	Estimating Whole Processor Partition Capacity.....	14
4	Micro-Partitioning.....	15
4.1	Results Overview	16
4.2	POWER6 1-way Micro-partition Scenarios	16
4.2.1	POWER6 1-way Micro-partition Performance Summary	17
4.3	POWER6 N-way Micro-partition Scenarios	17
4.3.1	POWER6 N-way Micro-partition Performance Summary	18
4.4	POWER5 1-way Micro-partition Scenarios	18
4.4.1	POWER5 1-way Micro-partition Performance Summary	19
4.5	POWER5 N-way Micro-partition Scenarios	20
4.5.1	POWER5 N-way Micro-partition Performance Summary	20
4.6	POWER4 Model 825 6-way Micro-partition Scenarios.....	21
4.6.1	POWER4 Model 825 Micro-Partition Performance Summary	21
4.7	Micro-partition Performance Tips	22
4.8	Estimating Micro-partition Capacity	23
5	Uncapped Processing and Virtual Processor Configurations	24
5.1	Uncapped configurations	24
5.1.1	POWER5 Model 595 Uncapped Processing Scenario.....	24
5.1.2	POWER5 Model 570 Entitlement Study	25
5.2	Virtual Processors	27
5.2.1	Results Overview	28
5.2.2	POWER5 Model 570 Virtual Processor Scenario	29
5.2.3	POWER5 Model 595 Virtual Processor Scenario	30
5.2.4	POWER6 Virtual Processor Scenarios	31
5.2.5	POWER5 Model 570 Virtual Processor Study	32
5.3	Performance Tips for Uncapped Processing and Virtual Processor	33
6	Mixed OS	35
6.1	Dedicated Processor Partitions	35
6.2	Whole, Shared Processor Partitions.....	38
6.3	Partial Processor Partitions	41

6.4	Conclusions.....	44
7	Memory Allocation and Performance.....	45
8	Trademarks and Disclaimers.....	46

1 Overview

The goals of this whitepaper are to characterize performance on System iTM, System pTM and PowerTM Systems in the following ways:

- Examine processor partitioning (LPAR) performance from two perspectives, whole processor partitioning and partial processor partitioning.
- Provide guidelines and tips to help optimize LPAR performance where appropriate.
- Provide some rules of thumb for sizing LPAR Capacity.

Over time, this document will be updated to include additional Logical Partitioning (LPAR) performance information as it becomes available.

This whitepaper is not a tutorial on Logical Partitioning (LPAR) and assumes that the reader has a basic understanding of partitioning concepts and configuration.

2 Systems under Test

The hardware used to perform LPAR performance testing consisted of System iTM models. With POWER5TM and POWER6TM the architecture, hardware, and firmware are common with System pTM models. The performance behavior documented here would be equivalent if measured on System pTM. The following System i models were used:

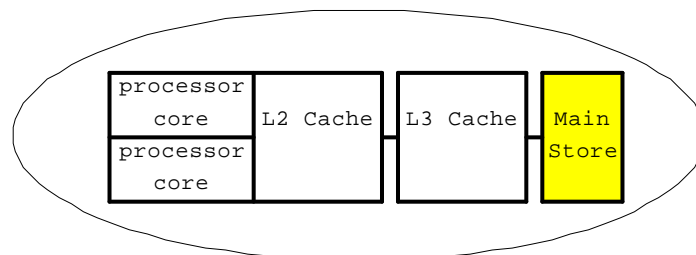
- 890-2498 32-way (37,400 published CPW) , POWER4TM architecture
 - 1.41MB shared L2 cache between 2 processors
 - Memory total: 256GB (balanced), DASD drives: 1440
 - OS/400® Release: V5R2
- 825-2473 6-way (6,600 published CPW), POWER4TM architecture
 - 1.41MB shared L2 cache between 2 processors
 - Memory total: 48GB (balanced), DASD drives: 465
 - OS/400® Release: V5R2
- 595-0947 32-way (85,000 published CPW)
 - POWER5TM architecture with Simultaneous Multi-threading (SMT)
 - 1.9MB/36MB L2/L3 cache shared between 2 processors
 - Memory total: 256 GB (balanced), DASD drives: 810
 - i5/OS® Release: V5R3
- 570-0922 8-way (23,500 published CPW)
 - POWER5TM architecture with Simultaneous Multi-threading (SMT)
 - 1.9MB/36MB L2/L3 cache shared between 2 processors
 - Memory total: 128GB (balanced), DASD drives: 900
 - i5/OS® Release: V5R3
- 570-0921 4-way (12,000 published CPW)
 - POWER5TM architecture with Simultaneous Multi-threading (SMT)
 - 1.9MB/36MB L2/L3 cache shared between 2 processors

- Memory total: 64GB (balanced), DASD drives: 465
 - i5/OS® Release: V5R3
- 520-0904 1-way (3,300 published CPW)
 - POWER5™ architecture with Simultaneous Multi-threading (SMT)
 - 1.9MB/36MB L2/L3 cache shared between 2 processors
 - Memory total: 16GB (balanced), DASD drives: 180
 - i5/OS® Release: V5R3
- 520-0903 1-way (2,400 published CPW)
 - POWER5™ architecture with Simultaneous Multi-threading (SMT)
 - 1.9MB L2 cache only, no L3 cache on this model
 - Memory total: 16GB (balanced), DASD drives: 180
 - i5/OS® Release: V5R3
- 570-9117-MMA (7380) 4-way POWER System (21,200 published CPW)
 - POWER6™ (4.7GHz) architecture with Simultaneous Multi-threading (SMT)
 - L2/L3 cache per CPU is 2x4MB / 32MB
 - Memory total: 64GB (balanced), DASD drives: 540
 - IBM i operating system 5.4.5 (also known as i5/OS® Release: V5R4M5)
- 570-9117-MMA (7380) 8-way POWER System (40,100 published CPW)
 - POWER6™ (4.7GHz) architecture with Simultaneous Multi-threading (SMT)
 - L2/L3 cache per CPU is 2x4MB / 32MB
 - Memory total: 128GB (balanced), DASD drives: 540
 - IBM i operating system 5.4.5 (also known as i5/OS® Release: V5R4M5)
- 520-9408 M25 (5634) 2-way (8,300 published CPW)
 - POWER6™ (4.2GHz) architecture with Simultaneous Multi-threading (SMT)
 - 2x4MB L2 cache only, no L3 cache on this model
 - Memory total: 32GB (balanced), DASD drives: 540
 - IBM i operating system 5.4.5 (also known as i5/OS® Release: V5R4M5)

The partitions under test operated as separate systems and there was no communication between partitions.

2.1 Hardware Overview and Memory Allocation

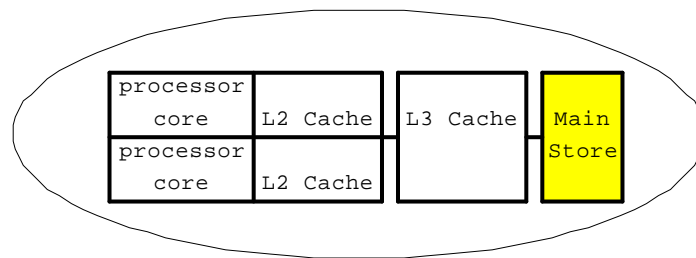
Before beginning to examine partitioning results, we will take a brief look at the hardware architecture used by the systems under test. One of the basic hardware building blocks of the POWER4™ processor and POWER5™ processor is the two-processor chip as shown in the diagram below.



The two processors on the chip each have their own Level 1 (L1) instruction and data caches. The two processors on the chip share a common Level 2 (L2) cache and where applicable share a common Level 3 (L3) cache.

- The Model 890-2498 32-way has 16 dual-core processor chips each with L3 cache.
- The Model 825-2473 6-way has 3 dual-core processor chips each with L3 cache.
- The Model 595-0947 32-way has 16 dual-core processor chips each with L3 cache.
- The Model 570-0922 8-way has 4 dual-core processor chips each with L3 cache.
- The Model 570-0921 4-way has 2 dual-core processor chips each with L3 cache.
- The Model 520-0904 1-way/2-way has 1 single- or dual-core processor chip with L3 cache.
- The Model 520-0903 1-way has 1 single-core processor chip and has no L3 cache.

The basic hardware building block of the POWER6™ processor is the two-processor chip as shown in the diagram below.



The two processors on the chip each have their own Level 1 (L1) instruction and data caches. Different from POWER4™ and POWER5™, on the POWER6™ chip, each processor on the chip has their own Level 2 (L2) cache and where applicable share a common Level 3 (L3) cache. The systems under test had the following ave modules defined as follows:

- 570-9117-MMA (7380) 4-way Power System has 2 dual-core processor chips each with L3 cache.
- 570-9117-MMA (7380) 8-way Power System has 4 dual-core processor chips each with L3 cache.
- i550-9408-M25 (5634) 1/2-way has 2 dual-core processor chips, each without L3 cache.

These two-processor chips are grouped together in sets of one or more chips to form a module. The notion of a module is important because memory and cache references within a module cost considerable less in terms of latency than references to memory or cache located within a different module.

On POWER4™ (Models 890 and 825) architecture, partition memory allocations are “striped” across all modules. On POWER5™ architecture (e.g. Models 520, 570, 595) and POWER6™ architecture (e.g. Models MMA, M25), partition memory allocations are first attempted on modules where the partition’s processors are allocated (nodal affinity) and tasks are attempted to be dispatched on the same processor that they last executed, thus reducing remote memory references.

For a more detailed look at memory affinity and memory access latencies, see the Enhanced eServer iSeries Performance Memory Affinity whitepaper at the following web site:
<http://www.ibm.com/servers/eserver/series/perfmgmt/pdf/memaffin.pdf>

2.2 Scenarios / Workloads

One of the workloads we used to evaluate LPAR performance was the CPW workload. The CPW workload represents a commercial computing environment. CPW is also the performance metric we use to depict relative performance between system models. The CPW workload performs a significant amount of database and journal operations and as such, this workload heavily utilizes the hardware L2 cache and memory subsystem. The CPW workload was used to evaluate performance of whole processor partitioning and micro-processor partitioning.

For a more detailed description of the CPW workload see the latest **IBM Power™ Systems Performance Capabilities Reference IBM i operating system™ Version 6, Release 1 appendix A** on the following IBM Performance Management Resource Library web site:
<http://www.ibm.com/servers/eserver/series/perfmgmt/resource.html>

Two other workloads were also used to evaluate micro-partitioning.

- WorkloadV is a highly threaded, CPU intensive Java™ workload that simulates sending/receiving short communication messages across TCP/IP socket interfaces.
- WorkloadS is a transaction based, compute intensive Java™ application that simulates a warehouse distribution environment without utilizing any database IO.

3 Whole Processor Partitioning

Whole processor partitions by definition contain no fractional processor allocations and thus contain an integer number of processors. Whole processor partition allocations may either be assigned as dedicated to a partition or part of a shared processor pool. In most cases, we have chosen to utilize dedicated processors as this provided the best performance for our workload under test. However, we will also present results utilizing a shared processor pool to demonstrate performance of shared processor partitions. The scenarios that utilize the shared pool have “capped processing”. Capped processing means that after a partition reaches its allotted processing capability, it will not be allowed to utilize unused capacity in other partitions. “Uncapped processing” allows a partition to utilize unused processing capacity in other partitions and thus may provide better overall utilization of the total system and its partitions.

As the following sections will show, running with multiple partitions instead of one large partition can provide better system capacity (results vary for workloads). Dedicated processor partitions are slightly better than shared partitions, but both show improvement over a single large partition. The choice between dedicated and shared partitions therefore falls more into the IT needs of a business instead of the difference in capacity. This provides the flexibility of having shared processor pools with whole processor and/or partial processor partitions and the ability to use uncapped processing to help make better use of system resources.

For whole processor partitioning, we will:

- evaluate system performance when using full processor partitions
- provide information relative to the cost of using shared processors
- provide performance tips to help tune for best performance
- provide a method for sizing the capacity for whole processor partitions

3.1 Results Overview

The results section contains the following information relative to LPAR scenario results:

- a brief description of the scenarios run
- performance results for stand-alone (baselines) and concurrent partition measurements
- a performance summary

For concurrent partition measurements, we defined the “overall” CPW metric as the sum of the CPW value achieved by each partition. This “overall” CPW metric can then be compared to baseline measurement of a single partitioned system containing the same number of processors as the sum of all the processors in the partitions under the measurement. The baseline measurements allowed us to gauge the relative performance when partitioning.

3.2 Model 890 and Model 595 32-way Scenarios

On the 32-way models we ran several whole processor partitions containing either 16, 8, 4, 2 or 1 dedicated processors. The partitions were run in combination with other identical partitions and

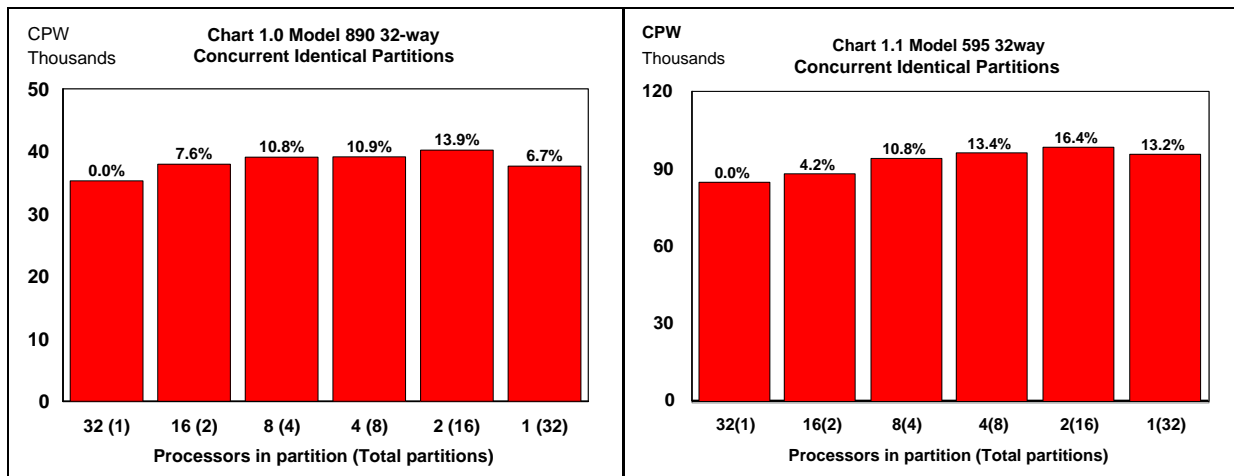
then compared to a baseline which was a single partition containing all the processors on the system. Identical partitions mean that the processor, memory and DASD allocations were the same.

For the concurrent measurements, we ran the following scenarios:

- one 32-way partition (baseline)
- two 16-way partitions
- four 8-way partitions
- eight 4-way partitions
- eight 2-way partitions (scaled to 16 partitions)
- eight 1-way partitions (scaled to 32 partitions)

Since we only ran a maximum of eight partitions concurrently, the eight 2-ways and eight 1-way scenarios results were scaled upward to allow a direct comparison to other partition scenarios which utilized all 32 processors.

In the chart that follows, the percentages that are over the bar graphs represent the delta CPW performance of a scenario when compared against the 32-way baseline CPW. As you can see from the following chart, the overall CPW metric generally improved as the number of concurrently running partitions increased.



3.2.1 Model 890 and Model 595 Concurrent Partitions: Performance Summary

When running concurrent partitions on the Model 890 32-way, the overall CPW throughput generally increased from 7% to 14% as the number of partitions increased. On the Model 595 32-way, the overall CPW throughput increased from 4% to 16% as the number of partitions increased. This performance gain was due in part to relieving both software and hardware constraints brought about by running smaller CPW workloads on the various partitions rather than one large CPW workload. For most scenarios, the partition processor allocation was set as a multiple of 2 processors (with their shared L2 cache). This processor allocation proved to be optimal and minimized the L2 cache interference that might occur when another partition executes. As you may recall, the model 890 and model 595 have two processors on a chip and

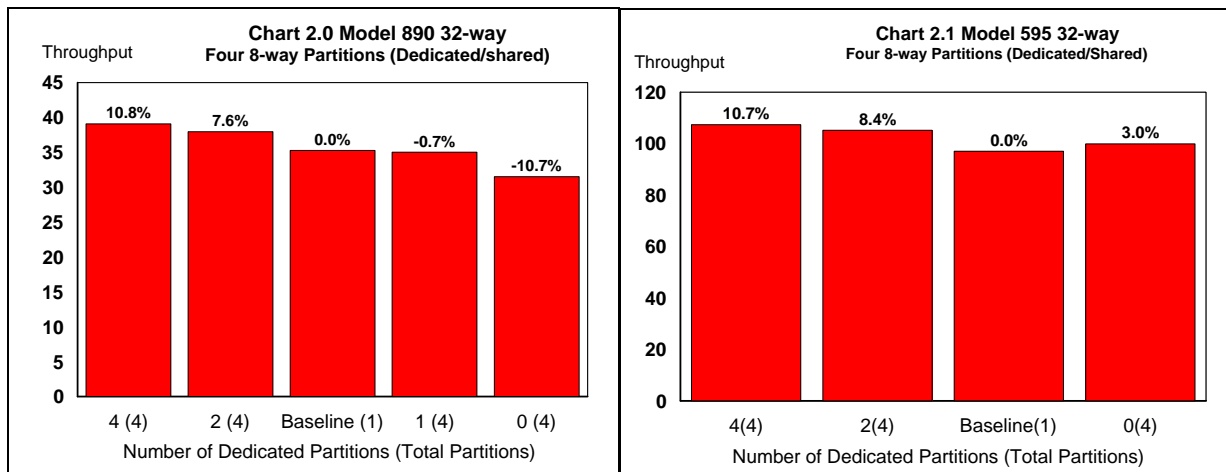
these two processors share a common L2 cache. When we allocated processors in multiples of 1 processor, each partition was essentially sharing the L2 cache with another partition and performance dropped off. Of course, this is only a concern if your workload is cache sensitive. Most workloads are cache sensitive to varying extents.

Another performance consideration was the fact that the installed memory was optimally balanced on this system. This means that we had an equal amount of memory plugged into each MCM and the memory DIMMs were all the same size. As the partition memory allocation is striped across all MCMs on a Model 890, this configuration ensured that we equalized the number of remote memory references (and their longer latency times) between partitions. Since we were running the same identical workload in all the partitions, we ensured that each partition had the same amount of “local” memory. For these scenarios, partitioning not only provided consolidation of systems but also improved CPW throughput. Obviously, not all workloads would benefit from this technique.

3.3 Model 890 and Model 595 Shared Processor Scenarios

In the next partitioning experiments, we will examine the performance impact of using a shared processor pool to allocate some number of 8-way partitions. The following partition scenarios will be compared to a single partition of 32 dedicated processors (baseline):

- Four 8-way partitions run currently utilizing all dedicated processors
- Two 8-way partitions utilizing all dedicated processors, and two 8-way partitions utilizing shared processors (shared pool size: 16 processors)
- One 8-way partition utilizing all dedicated processors and three 8-way partitions utilizing shared processors (shared pool size: 24 processors)
- Four 8-way partitions utilizing shared processors (shared pool size: 32 processors)



3.3.1 Model 890 and Model 595 Dedicated/Shared Partitions: Performance Summary

The percentages above the bar graphs represent the delta throughput of the test case when compared to the 32-way baseline. As you can see from the chart above, the partition test cases that utilized the most dedicated processors outperformed the shared processor test cases. On the

model 890 (chart 2.0) the performance throughput differences of the shared processor scenarios ranged from a low of 3% to a high of 21% when comparing the four dedicated 8-way LPAR throughput to the four shared 8-way processor LPARs. There have been some improvements made to i5/OS® V5R3M5 and beyond that are visible in the model 595 chart (chart 2.1). The 8-way shared LPARs correlate more closely to dedicated processor LPARs than in previous OS releases. With this improvement, shared and dedicated processor LPARs configured with similar capacity will show similar CPU utilization when running equivalent workloads. The above model 595 graph (chart 2.1) shows that four 8-way shared LPARs could have about 7% decreased throughput compared to four 8-way dedicated LPARs due to sharing of processors by 4 partitions. It also shows that four shared 8-way LPARs could see 3% more throughput than one 32-way partition.

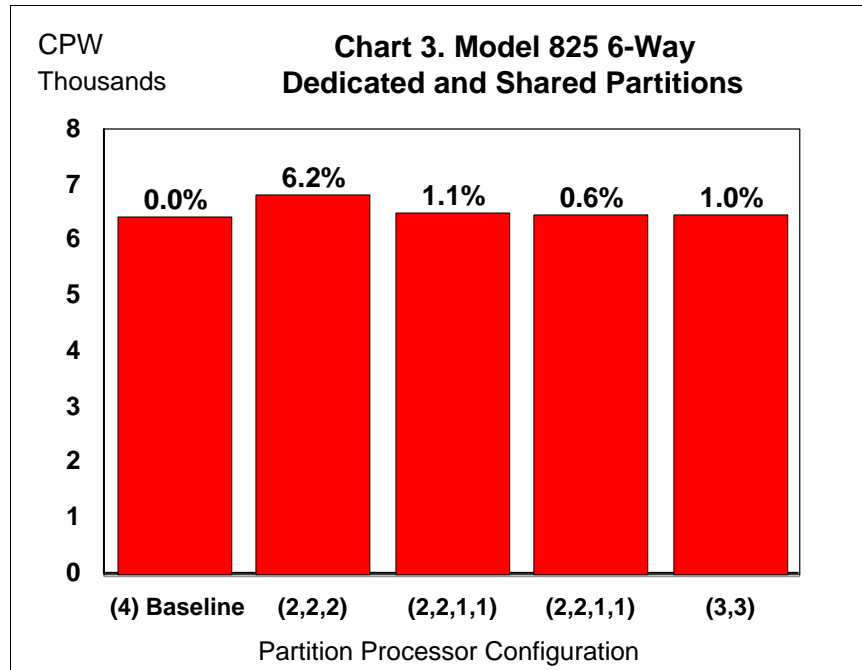
The results also show on the model 890, that the larger the shared processor pool, the greater the impact was to the overall throughput. Because jobs move from processor to processor within the shared pool, there is a higher likelihood that L2 cache contents may be lost.

3.4 Model 825 6-way Scenarios

On the 6-way Model 825 with its 3 DCMs, we ran several whole processor partitions containing either 1, 2, or 3 processors. The partitions were run in combination with other partitions and then compared to a single partition containing 6 dedicated processors (baseline). The memory and DASD allocations for each partition were proportional to the number of processors allocated for the partition. For the concurrent measurements, we ran the following LPAR scenarios:

- three 2-way partitions (dedicated processors)
- two 2-way partitions and two 1-way partitions (dedicated processors)
- two 2-way partitions (dedicated) and two 1-way partitions (shared pool size 2)
- two 3-way partitions (dedicated)

In the chart that follows, the percentages that are over the bar graphs represent the delta CPW performance of a scenario when compared against the 6-way baseline CPW.



3.4.1 Model 825 Dedicated Partitions: Performance Summary

As you can see from Chart 3, the three 2-way partitions provided the best overall CPW metric. This is due to the fact that configuring LPARs processors in multiples of two minimizes L2 cache contention (gets entire L2 cache) and LPARs also fit nicely within the boundaries of an SCM which helps further reduce remote memory references. For the Model 825, memory allocations are striped across the SCMs.

The comparisons between the (2,2,1,1) scenarios, and the (3,3) scenario showed very little difference in performance. This was due to the fact that there was extensive sharing of the L2 caches in these scenarios which was responsible for the majority of the performance change when compared to the three 2-way scenario.

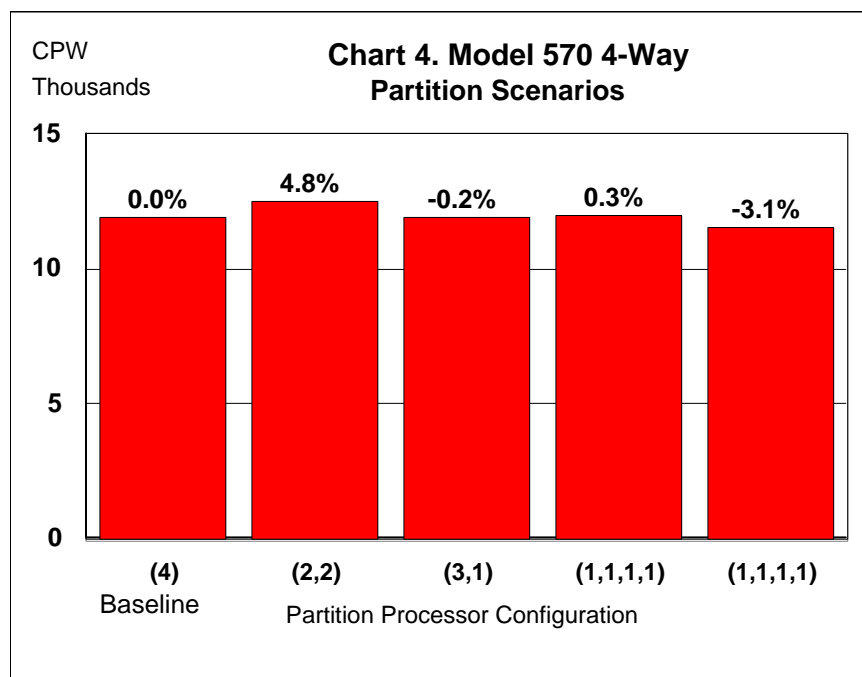
3.5 Model 570 4-way Scenarios

The POWER5™ architecture brings a couple of significant performance improvements to the table. First, the hardware has Simultaneous Multi-threading (SMT) which better utilizes the hardware by executing a second processor thread concurrently. For more information on SMT, see the **[Simultaneous Multi-Threading on eServer iSeries POWER5™ Processors](http://www.ibm.com/servers/eserver/series/perfmgmt/pdf/SMT.pdf)** whitepaper at <http://www.ibm.com/servers/eserver/series/perfmgmt/pdf/SMT.pdf>. Secondly, partition memory allocations are first attempted on modules where the workload executes and the system attempts to keep the workload jobs running on the same processors that they were dispatched on before (nodal affinity). Therefore, memory references have a much better chance of being “local” when compared to the memory striping technique prevalent on POWER4™ systems. Also, any data left in the L2 cache from the previous executions may be reused. This change can provide a significant boost to some workloads that are run in partitions.

On the Model 570 we ran several whole processor partitions containing either 1, 2 or 3 processors. The partitions were in combination with other partitions and then compared to a baseline run which was a single partition containing 4 dedicated processors. The memory and DASD allocations for each partition were proportional to the number of processors allocated for the partition.

For the concurrent measurements, we ran the following scenarios and compared the scenarios to a single 4-way partition with dedicated processors:

- two 2-way partitions (dedicated processors)
- one 3-way partition with one 1-way partitions (all dedicated processors)
- Four 1-way partitions (dedicated processors)
- Four 1-way partitions (shared processors, pool size was 4 processors)



3.5.1 Model 570 Dedicated/Shared Partitions: Performance Summary

The percentages that are over the bar graphs represent the delta CPW performance of a scenario when compared against the 4-way baseline CPW.

As you can see from Chart 4, the two 2-way partitions scenario provided the best overall performance (4.8% improvement over baseline). Once again, this was due to the fact that a two processor dedicated partition minimizes L2 cache contention and remote memory references. As was previously stated, breaking up the CPW workload into smaller workloads and then running CPW workloads on multiple partitions relieved some software and hardware constraints.

The last two scenarios on the chart compare dedicated versus shared performance. The performance throughput of the shared processor scenario was 3.4% lower than the four dedicated 1-way scenario.

3.6 Tips

When creating whole processor partitions, here are some tips to consider:

1. Configure the system properly. Balance the memory DIMMs and IO across the MCMs, SCMs or DCMs. Utilize the same size memory DIMMs on the modules whenever possible. This will help reduce latencies caused by remote references and avoid memory “hot spots”.
2. Generally avoid using a shared processors pool when you have whole processor LPARs on POWER4TM systems. In addition, you may want to consider rounding up some partitions to the nearest whole processor and then use dedicated processors instead of shared processors. On POWER5TM and POWER6TM architecture you may want to use shared processors with uncapped processing to better utilize partitions that have extra cycles available. Our experience with uncapped processing has shown that the uncapped function works as advertised, providing significant improvement to fully utilizing all the processor capacity in partitions that have idle processor cycles available.
3. On POWER4TM and POWER5TM, size partitions in multiples of two processors if feasible. This will minimize some of the interference to shared hardware on a chip such as the L2 cache. Optimal performance on POWER6TM does not depend on sizing partitions in multiple of two processors since on POWER6TM processors have private L2 caches.
4. Size partitions to fit within MCM, SCM and DCM boundaries if feasible. Partitions that fit well within these boundaries will have fewer remote memory references and less L2 cache interference. Also, make sure that your allocated memory doesn't exceed the memory on the module if possible.
5. On POWER5TM and POWER6TM systems, the hypervisor attempts to optimize memory allocations at full system startup. If after the system has started, you change a partition's memory allocation on a multi-module system (DCM or MCM), you may introduce more remote memory references as memory is "reallocated" from its initial optimal allocation to another module. If you suspect that this has happened, another full system startup will re-optimized the memory allocations.

3.7 Estimating Whole Processor Partition Capacity

Processor efficiency with Whole Processor partitioning varies between system models, workloads, and partition sizes. It is beyond the scope of this document to describe the algorithms needed to accurately size Whole Processor partitions. The reader should not attempt sizing from the results discussed in this document. For accurate estimates, you should use the IBM Systems Workload Estimator (WLE) site to size various partitions. WLE may be accessed via the web at <http://www.ibm.com/systems/support/tools/estimator/index.html>.

4 Micro-Partitioning

Micro-Partitioning™ support provides flexible and efficient use of system hardware resources by allowing physical processors to be shared (time-sliced) between micro-partitions. Among the key customer environments where micro-processor LPARs provide benefit are ones in which multiple system images exist due to isolation requirements (security, software fault tolerance), and generally have low processor requirements but occasionally requiring more. Historically, each of these system images runs on a separate hardware system with sufficient capacity to handle spikes in processor requirements, but in general each hardware system goes underutilized. Micro-Partitioning™ support can allow these system images to be consolidated on a single set of physical processors, allowing more efficient use of hardware resources, and provide a reduction in the physical footprint required by multiple systems. Micro-Partitioning™ with uncapped processing, provides more granularity for CPU resource balancing and allows idle CPU cycles to be recovered and used by other partitions.

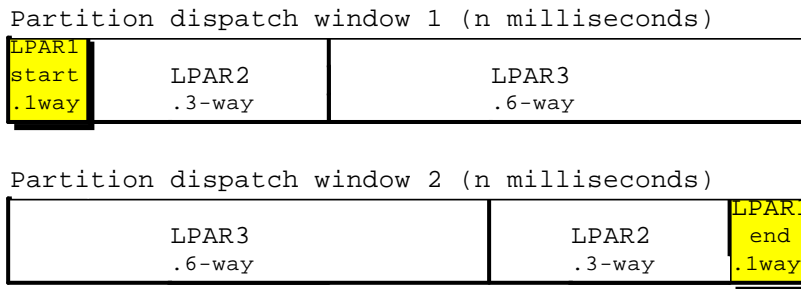
Micro-partitions contain a fractional processor allocation (e.g. 0.3, 1.3, 3.5, etc.) and the processor allocation will come from the shared processor pool. Micro-partition processor allocations may either be capped or uncapped on POWER5™ or POWER6™ architecture.

For Micro-Partitioning™, we will:

- Evaluate system performance when using micro-partitions
- Provide performance tips to help lead to best performance
- Provide methodology to estimate LPAR capacity for micro-partitions

Care must be taken when you are configuring the micro-partition size, especially if the workload that you will be running in the partition is CPU intensive and/or has some response time criteria. It is not unusual for some CPU intensive workloads to require 10-20 milliseconds of CPU to complete a transaction. When a workload with similar processing requirements is placed in too small of a micro-partition, you might see increases in response time.

The following example shows what can happen if the micro-partition size too small. This example has three micro-partitions (LPAR1, LPAR2 and LPAR3) running concurrently. LPAR2 and LPAR3 have been correctly configured. Micro-partition LPAR1 was allocated 10% of a physical processor (capped processing) from a shared processor pool (1 processor) but in actuality requires closer to 20% of a processor. Micro-partitions will be given a portion of a partition time-slice directly proportional to the “size” of the partition. Since LPAR1 was assigned 10% of a physical processor, it will get up to 10% of each available time-slice. Because LPAR1 actually requires 20% of a physical processor to complete its work, it will not finish until sometime during the second time-slice. See the diagram below.



For this example, if the dispatch window were 10 milliseconds, then the LPAR1 portion of the dispatch window would be 1 millisecond (10%). An LPAR1 transaction which requires 2 milliseconds of processing to complete would finish somewhere between 11 and 20 milliseconds. That represents a 5 - 10 times longer response time when compared to the average 2 milliseconds transaction CPU service time which might occur if LPAR1 were configured as 0.2 processor units!

4.1 Results Overview

The results section provides the following information relative to Micro-Partitioning™:

- a brief description of the scenarios run
- a performance summary of results for stand-alone (baselines) and concurrent partition measurements

For concurrent partition measurements, a throughput metric for each partition is summed up with all the other active partitions to produce an “overall” throughput metric. This “overall” throughput metric can then be compared to baseline measurement of a single partitioned system containing the same number of processors as all the partitions under the measurement.

4.2 POWER6 1-way Micro-partition Scenarios

In the POWER6™ 1-way scenarios, we defined a shared pool containing a single processor and then ran several micro-processor partitions which attempted to fully utilize the shared pool. The partitions were run in combination with other identical partitions and compared to a single partition containing 1 dedicated processor. For the concurrent measurements, we ran the following scenarios:

- One 1-way dedicated processor
- Two 0.5 processor unit partitions
- Three 0.33 processor unit partitions
- Four 0.25 processor unit partitions
- Five 0.2 processor unit partitions

- Six 0.16 processor unit partitions
- Eight 0.12 processor unit partitions
- Ten 0.1 processor unit partitions

Two different workloads were used to evaluate Micro-Partitioning™ performance: WorkloadV and WorkloadS. The following summary represents a composite average of the throughput of the two workloads compared against the baseline of a 1-way dedicated partition.

Note that measurements were run on three operating systems (IBM i operating system 5.4.5, AIX 5.3L, and Linux SUSE SLES 10 SP1), and results are composite of all three OSs and workloads.

Measurements were run on a System i model 9408-M25 system that has no L3 cache and on a Power™ 570 model 9117-MMA system that has an L3 cache.

4.2.1 POWER6 1-way Micro-partition Performance Summary

For any POWER6™ model, processor efficiency is impacted as more and more partitions share a processor. The more partitions sharing a processor reduce the combined capacity of that processor.

POWER6™ models with L3 cache have significantly higher system capacity than POWER6 models without L3 cache.

For POWER6™ models with L3 cache, as the number of partitions sharing a processor increase, the burden on the L3 cache increases, reducing the L3 cache effectiveness at large numbers of partitions.

Although the capacity of the system model without L3 cache is less that of the capacity of the system model with L3 cache, the rate at which processor efficiency is impacted from increasing numbers of partitions is smaller (better). Since there is no L3 cache, there is no processor efficiency impact from L3 burden as the number of partitions increase.

The relative processing efficiency reduction as more and more partitions share the processor is not as great on the model without the L3 cache as on models with the L3 cache.

4.3 POWER6 N-way Micro-partition Scenarios

The POWER6™ n-way scenarios are very similar to the scenarios defined for the POWER6™ 1-way scenarios. In these scenarios, we defined a shared pool containing N processors and then ran several micro-processor partitions which attempted to fully utilize the shared pool. The micro-partitions are sized identically but there “n” times as many of the micro-partitions. N is defined as the number of processors on the system and will be the size of the shared processor pool. For example, for N equal to 4 processors, we ran eight 0.5 processor unit partitions, sixteen

0.25 processor unit partitions, etc. All concurrent scenarios were compared to a single partition containing all the N processors dedicated, which totaled the number of processors in the shared pool as well.

The partitions were run in combination with other identical partitions and compared to a single partition containing all the processors dedicated. For the concurrent measurements, we ran the following scenarios:

- One N-way dedicated processor
- N/4 4.00 processor unit partitions
- N/2 2.00 processor unit partitions
- N/2*three 0.66 processor unit partitions
- N*two 0.5 processor unit partitions
- N/2*five 0.4 processor unit partitions
- N* three 0.33 processor unit partitions
- N*four 0.25 processor unit partitions
- N*five 0.2 processor unit partitions
- N*ten 0.1 processor unit partitions

For these results we used N values of 2, 4, 6 and 8.

Two different workloads were used to evaluate Micro-Partitioning™ performance: WorkloadV and WorkloadS. The following summary represents a composite average of the throughput of the two workloads compared against the baseline of a partition with all the processors dedicated.

Note that measurements were run on three operating systems (IBM i operating system 5.4.5, AIX 5.3L, and Linux SUSE SLES 10 SP1), and results are composite of all three OSs and workloads.

Measurements were run on a Power™ 570 model 9117-MMA 4-way system that has L3 cache and on a Power™ 570 model 9117-MMA 8-way system that has L3 cache.

4.3.1 POWER6 N-way Micro-partition Performance Summary

The processing efficiency reduction for running smaller partitions/more partitions on n-way POWER6™ systems is greater than on a 1-way system. This is due to the fact that the 1-way system has the entire L3 cache and is not sharing it with another processor running on the same chip. With n-way systems, there are two processors on a chip that share L3 cache. Therefore, even before we start Micro-Partitioning™ the system, there may be considerable cache contention on n-way systems. Essentially, on an n-way system, the L3 caches are effectively cut in half by the sharing of common hardware. For these workloads, Micro-Partitioning™ performance is better on POWER6™ systems than on POWER5™ systems.

4.4 POWER5 1-way Micro-partition Scenarios

In the POWER5™ 1-way scenarios, we defined a shared pool containing a single processor and then ran several micro-processor partitions which attempted to fully utilize the shared pool. The partitions were run in combination with other identical partitions and compared to a single partition containing 1 dedicated processor. For the concurrent measurements, we ran the following scenarios:

- One 1-way dedicated processor
- Two 0.5 processor unit partitions
- Three 0.33 processor unit partitions
- Four 0.25 processor unit partitions
- Five 0.2 processor unit partitions
- Six 0.16 processor unit partitions
- Eight 0.12 processor unit partitions
- Ten 0.1 processor unit partitions

Three different workloads were used to evaluate Micro-Partitioning™ performance: CPW, WorkloadV and WorkloadS. The following summary represents a composite average of the throughput of the three workloads compared against the baseline of a 1-way dedicated partition.

Note: measurements were run on IBM i operating system 5.3. Results were checked with measurements on AIX 5.3 and Linux SUSE. Results are composite of all three workloads.

Measurements were run on a System i model 520-0903 1-way system that has no L3 cache and on a System i model 520-904 1-way system that has an L3 cache.

4.4.1 POWER5 1-way Micro-partition Performance Summary

All workloads see a general trend where dividing up a processor more and more by increasing the number of partitions sharing it reduces the combined capacity of the partitions sharing that processor.

Performance on POWER5™ 1-way models benefits from having the entire L2 cache on the chip for 1 processor's use.

Performance of Micro-Partitioning™ on POWER5™ 1-way models sees additional benefit from not having to share the L2 cache with a second processor. For equal number of partitions per processor (e.g. 10 partitions on 1-way vs. 20 partitions on 2-way), the impact on processor efficiency as the number of partitions increase will be smaller (better) on the 1-way model than on models with more than 1 processor. Capacity for multiple partitions on a 1-way model will be closer to the best case result of 1 partition.

Performance on POWER5™ models also benefit significantly from having L3 cache. However, as the number of partitions sharing a processor increase, the burden on the L3 cache increases, reducing the L3 cache effectiveness at large numbers of partitions.

There is a twist on performance of Micro-Partitioning™ on POWER5™ 1-way models without L3 cache. Although the capacity of the system is less than the capacity of the 1-way model with L3 cache, the rate at which processor efficiency is impacted from increasing numbers of partitions is smaller (better). Since there is no L3 cache, there are no processor efficiency impacts from L3 burden as the number of partitions increase.

The relative processing efficiency reduction as more and more partitions share the processor is not as great on the model without the L3 cache as on models with the L3 cache.

4.5 POWER5 N-way Micro-partition Scenarios

The POWER5™ n-way scenarios are very similar to the scenarios defined for the POWER5™ 1-way. We defined a shared pool containing N processors and then ran several micro-processor partitions which attempted to fully utilize the shared pool. The micro-partitions are sized identically but there “n” times as many of the micro-partitions. N is defined as the number of processors on the system and will be the size of the shared processor pool. For example, for N equal to 8 processors, we ran sixteen 0.5 processor unit partitions, thirty-two 0.25 processor unit partitions, etc. All concurrent scenarios were compared to a single partition containing all the processors dedicated which totaled the number of processors in the shared pool.

The partitions were run in combination with other identical partitions and compared to a single partition containing all the processors dedicated. For the concurrent measurements, we ran the following scenarios:

- One N-way dedicated processor
- N/2*three 0.66 processor unit partitions
- N*two 0.5 processor unit partitions
- N/2*five 0.4 processor unit partitions
- N*Five 0.2 processor unit partitions
- N*Ten 0.1 processor unit partitions

For these results we used N values of 2, 4, 6 and 8.

Three different workloads were used to evaluate micro-partitioning performance: CPW, WorkloadV and WorkloadS. The following summary represents a composite average of the throughput of the three workloads compared against the baseline of a 1-way dedicated partition.

Note: measurements were run on IBM i operating system 5.3. Results were checked with measurements on AIX 5.3 and Linux SUSE. Results are composite of all three workloads.

Measurements were run on a System i model 570-0921 4-way system that has L3 cache and on a System i model 570-0922 8-way system that has L3 cache.

4.5.1 POWER5 N-way Micro-partition Performance Summary

The processing efficiency reduction for running smaller partitions/more partitions on n-way POWER5™ systems is greater than on a 1-way system. This is due to the fact that the 1-way system has the entire caches and is not sharing them with another processor running on the same chip. With n-way systems, there are two processors on a chip that share caches. Therefore, even before we start micro-partitioning the system, there may be considerable cache contention on n-way systems. Essentially, on an n-way system, the L2 caches are effectively cut in half by the sharing of common hardware. Overall, Micro-Partitioning™ performance is better on POWER5™ systems than on POWER4™ systems because of the hardware Simultaneous Multi-threading (SMT) function.

4.6 POWER4 Model 825 6-way Micro-partition Scenarios

On the POWER4™ Model 825 we defined a shared pool containing a single processor and then ran several micro-processor partitions which attempted to fully utilize the shared pool. The partitions were run stand-alone to get a baseline and in combination with other identical partitions. The memory and DASD allocations were proportional to the fractional number of processors allocated for each partition.

We ran the following scenarios:

- One 1-way dedicated processor
- Two 0.5 processor unit partitions
- Three 0.33 processor unit partitions
- Four 0.25 processor unit partitions
- Five 0.2 processor unit partitions
- Six 0.16 processor unit partitions

Three different workloads were used to evaluate micro-partitioning performance: CPW, WorkloadV and WorkloadS.

Note that results are only for i5/OS V5R3.

The result summary that follows compared identical micro-partitions to a baseline 1-way dedicated LPAR measurement. The percentages represent the relative throughput of the scenario compared to the baseline throughput. There were some scenarios where all the processor allocations didn't total one processor (.33 and .16). We scaled these results upward to allow a direct comparison to the baseline. The following summary uses all three workloads compared to the baseline.

4.6.1 POWER4 Model 825 Micro-Partition Performance Summary

Performance with Micro-Partitioning™ can vary greatly depending upon the workload characteristics. Capacity reduction stems from Processor Efficiency reductions due to impacts from more partitions sharing the caches. Some workloads will see more of an impact from micro-partitioning than other workloads. All workloads see a general trend where dividing up a

processor more and more by increasing the number of partitions sharing it reduces the combined throughput of the partitions sharing that processor.

The “average” capacity reduction in these measurements was somewhere in the range of 1% to 6% per partition when compared to the baseline.

These results were for identically defined micro-partitions. We also ran a few experiments with two different sized micro-partitions (.9 with .1, and .8 with .2). In general, the smaller sized partitions were impacted much more severely than the larger partitions but overall throughput was higher in both cases (.9 with .1, and .8 with .2) than the 0.5 processor unit partitions scenario.

4.7 Micro-partition Performance Tips

Note that tips are for any of three OSs (IBM i operating system, AIX, and Linux SUSE).

1. Correctly determine the micro-partition processor allocation. Sizing the partition too small may significantly increase response times. In addition, the processor efficiency is affected more with smaller partitions/more partitions.
2. On POWER5™ systems, consider using uncapped processing to better utilize idle processor cycles in other partitions in the shared pool.
3. Limit the number of micro-partitions that are active at any one time. Workloads that are cache sensitive or have response time criteria may suffer with the increased contention that micro-partitioning places on shared resources.
4. Balance the memory DIMMs and IO across the modules. Utilize the same size memory DIMMs on the modules whenever possible. This will help reduce latencies caused by remote references and avoids “hot spots”.
5. If you use a shared processor pool on POWER4™ systems, keep the size of the shared pool as small as possible. The larger the shared pool, the more likely jobs will be dispatched on a different processor than their last execution and lose the previous L2 cache contents.
6. For POWER4™ systems, size the shared processor pool in multiples of two processors if feasible. This will help minimize some of the interference with dedicated partitions. On POWER5™ and POWER6™ systems, you can not directly size the shared pool. The size of the shared pool will be comprised of all the processors not currently in use by dedicated processor partitions.
7. For optimal system capacity, keep the number of virtual processors as small as possible. For example, it is better to run a 0.8-way partition as 1 Virtual Processor versus 2 or more virtual processors. See the section on Virtual Processors for more details.
8. On POWER5™ and POWER6™ systems, the hypervisor attempts to optimize memory allocations at full system startup. If after the system has started, you change a partition’s memory allocation on a multi-module system, you may introduce more remote memory references as memory is “reallocated” from its initial optimal allocation to another module. If you suspect that this has happened, another full system startup will re-optimized the memory allocations.

4.8 Estimating Micro-partition Capacity

Processor efficiency with Micro-Partitioning[™] varies between system models, workloads, and partition sizes. It is beyond the scope of this document to describe the complex algorithms needed to size Micro-partitions. The reader should not attempt sizing from the results discussed in this document. For accurate estimates, you should use the IBM Systems Workload Estimator (WLE) to size various micro-partitions. WLE may be accessed via the web at <http://www.ibm.com/systems/support/tools/estimator/index.html>.

5 Uncapped Processing and Virtual Processor Configurations

In this section, we will take a look at uncapped processing and Virtual Processor configurations. As was noted earlier uncapped processing, together with a shared processor pool, allows a partition to utilize idle cycles in other partitions that run out of the shared processor pool. Thus, uncapped processing may provide a method for better overall utilization of a system and all its resources. Uncapped processing is only available on POWER5™ and POWER6™ systems.

There are two performance knobs that can be used to control uncapped processing:

- Processing capacity as defined by Virtual Processors (entitled capacity, maximum capacity). The partition will be provided the entitled capacity and may grow to the maximum capacity if idle cycles are available in other partitions (see section 5.2 for further information on Virtual Processors).
- Partition weights are used as a priority scheme to divide up available cycles between uncapped partitions. If the partition weight is set to zero, the partition will not be able to utilize idle cycles in other partitions.

5.1 Uncapped configurations

Uncapped partitioning allows the idle machine cycles in the shared processing pool to be used by any partition within the shared pool that needs extra processing. So a 0.1 processor partition with 1 virtual processor would have a base entitlement of 0.10 of a processor with a maximum capacity of 1.0 processor.

In this sub-section we will examine 2 Scenarios:

- POWER5™ Model 595 running larger partitions (multiple processor units)
 - 4 partitions running in a shared pool of 16 processors
- POWER5™ Model 570 running small and micro partitions
 - to 5 partitions running in a shared pool of 1 processor
 - to 5 partitions running in a shared pool of 4 processors

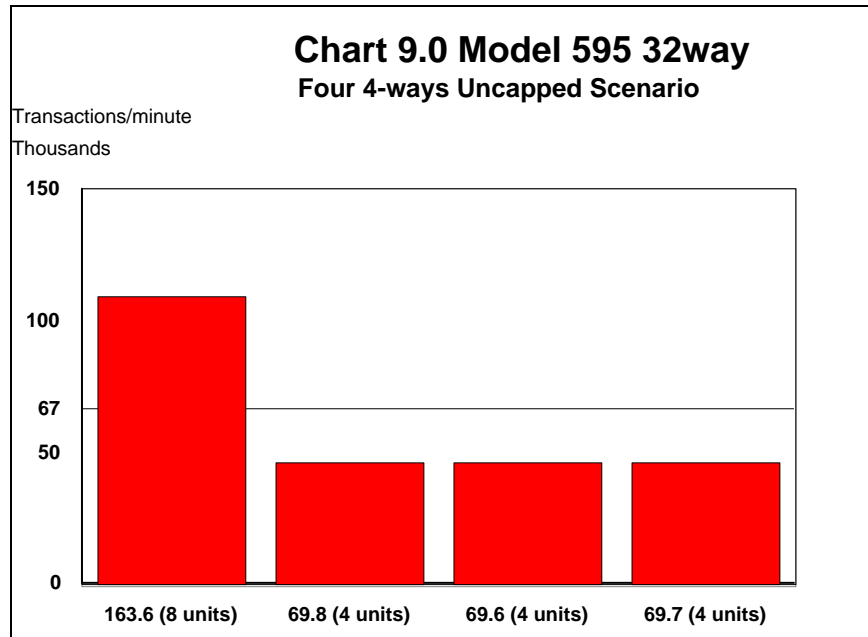
5.1.1 POWER5 Model 595 Uncapped Processing Scenario

On the POWER5™ Model 595 32-way system we defined the following partitions:

- One uncapped 4.0 processor unit partition (entitled capacity = 4.0 processor units, virtual processors = 8), weight = 128
- Three capped 4 processor unit partitions (entitled capacity = 4.0 units, capped processing)
- One dedicated partition (idle) containing the rest of the processors (16-way)

The objective of the scenario was to determine how well the uncapped partition used available cycles from the other partitions. All partitions were running the CPW workload. The capped partitions ran around 70% CPU utilization. We continued to add work to the uncapped partition

as long as the throughput improved and response times were acceptable. Chart 9.0 shows the results of this scenario.



5.1.1.1 POWER5 Model 595 Uncapped Partition: Performance Summary

In uncapped partitions, processing can exceed 100% CPU utilization (163.6 % in this scenario). This is due to the fact that the CPU utilization is based upon the “entitled capacity”) or in this case, 4.0 processor units. The uncapped partition had approximately 3.6 additional processor units worth of capacity available to it based upon the total idle processor units of the other three partitions running in the shared pool. The other three 4.0 processor unit partitions had approximately 30% idle CPU utilization or 1.2 processor units available (4*.3). For this scenario, we set the maximum capacity for the uncapped partition as close as possible to the available idle cycles (4+3.6 = 8 virtual processors). This is a good practice as there is some increased processing associated with the number of virtual processors defined for a partition.

In the chart above, the line at 67,000 represents the maximum throughput if partition 1 processing were capped. As you can see, uncapped processing provided a very significant advantage to this scenario. With uncapped processing, we were able to better utilize the total system. The overall system CPU utilization (all partitions) was equal to 93.2% (sum of all utilizations/400%) or $(163.6+69.8+69.6+69.7)/400 = 93.2\%$. Without uncapped processing, the overall system CPU utilization would have been a maximum of 77.3% $((100+69.8+69.6+69.7)/400 = 77.3\%)$ and we would have processed 42,500 transactions/minute fewer in the partition that was previously uncapped.

5.1.2 POWER5 Model 570 Entitlement Study

On the POWER5™ Model 570 4-way system we defined the following environments:

LPAR Performance on Power Systems with POWER4™, POWER5™ and POWER6™

- One processor shared pool with 2 to 5 equal sized partitions. The remaining processors were assigned to a non-active dedicated partition to eliminate those processors from the shared processor pools
- Four processor shared pool with 2 to 5 equal sized partitions.

The objective of these experiments was to illustrate the performance differences as more processing capacity is distributed as uncapped capacity. These experiments were run using all uncapped partitions. Multiple configurations and workloads were run for each environment. Each configuration variation was run twice, once with WorkloadS and again with WorkloadV. For simplicity the performance of both workloads were combined.

The charts below represent the set of experiments run.

There were three variables:

- Total number of processors in the shared pool, (title of each chart)
- The number of partitions in the shared pool (rows in the table and each bar grouping in the charts)
- The total assigned entitlement (Columns in the table and each bar in the groups in each chart)

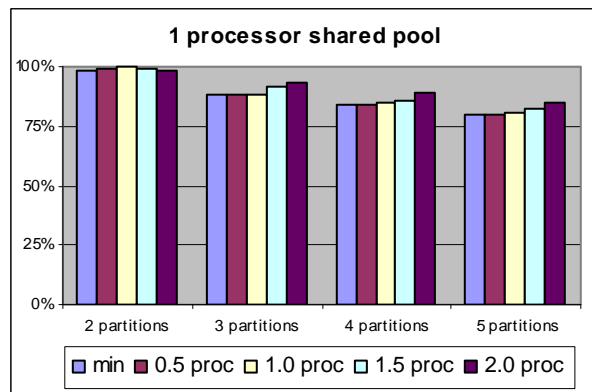
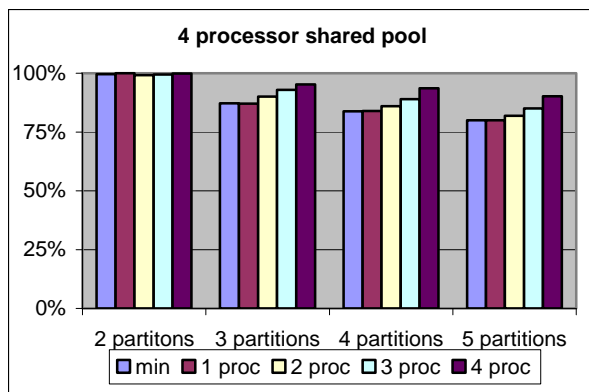
The cells in the table at the intersection of each row/column are the assigned processor units (entitlement) for each partition.

Note: In order to allow use of all processors in the shared pool, we set the partitions to use 2 virtual processors per partition for all 4-processor shared pool runs. The 1-processor shared pool runs were all 1 virtual processor per partition.

The experiments were defined like so:

	shared pool = 4 processors entitlement					shared pool = 1 processors entitlement		
	min	1 processor	2 processor	3 processor	4 processor	min	.5 processor	1 processor
2 partitions	0.2	0.5	1	1.5	2	0.1	0.25	0.5
3 partitions	0.2	0.33	0.66	1	1.33	0.1	0.16	0.33
4 partitions	0.2	0.25	0.5	0.75	1	0.1	0.12	0.25
5 partitions	0.2	0.2	0.4	0.6	0.8	0.1	0.1	0.2

The following charts show the results of these experiments:



5.1.2.1 POWER5 Model 570 Entitlement Study: Performance Summary

There is a correlation between the amount of uncapped processing available in the shared pool (i.e. processing power not entitled to any partition in the shared pool) and total performance of the partitions.

The two charts point out that as total base entitlement increases so does the total performance of the partitions. The less the amount of uncapped resources available to dispense to uncapped partitions, the less processing required to dispense the resources and the more efficient the execution.

The only exception in the carts is the 2 partition 2 virtual processors per partition on a 4 processor shared pool. This is a case where the total virtual processors equals the number of physical processors in the shared pool and is further addressed later in section 5.2.5 POWER5 Model 570 Virtual Processor Study.

5.2 Virtual Processors

In a shared processor partition, the operating system's processors are virtualized. In this context, virtualized have the following concepts:

- Virtual processors are used to manage physical processors in the shared pool.
- Virtual processors are used to manage work assigned to shared partitions to make sure that all partitions get their assigned processing capacity.
- Virtual processors are used to distribute excess cycles from the shared pool to uncapped partitions.

Using virtual processors to estimate performance of a micro partition is not a valid approach. There is a difference between running multiple partitions per physical processors (Micro-Partitioning™) and running 1 partition with multiple virtual processors. Section 4.0 of this document describes micro partitions while this section describes multiple virtual processors.

The numbers of virtual processors assigned to a partition are the number of concurrent units of execution that are possible in a partition. The partition uses the desired number of virtual processors to consume the desired processing units (for a capped partition). If the partition does not consume all of the CPU time, the excess time will go back into the shared pool for uncapped partitions to utilize.

For example, if the system administrator assigns 6.0 processing units and 8 virtual processors, the operating system can dispatch work to the 8 virtual processors to consume 6.0 processing units worth of CPU time. In this example, the CPU capacity of a virtual processor of the partition is $6.0/8=0.75$ of a physical processor.

As stated earlier in the 4.5 Tips section (number 6), the number of processors in the shared pool on POWER5™ and POWER6™ is different than POWER4™ in that you can not specify the number of processors that are in the shared pool on POWER5™ and POWER6™. For POWER5™ and POWER6™ the number of available processors in the shared pool equals the

total system processors minus the unavailable processors, where unavailable processors are any of the following:

- Dedicated processors
 - There is a subset of processors assigned for dedicated processor partitions that can be available to the shared pool. When creating a dedicated partition profile using the HMC, there are 2 check boxes under the heading “Processor Sharing”:
 - “Allow when partition is inactive” - is a check box that if checked, will allow the processors assigned for the dedicated processor partition to be included in the shared processor pool when the partition is powered off.
 - “Allow when partition is active” - is a check box that if checked, will allow the processors assigned for the dedicated processor partition to be included in the shared processor pool when the partition is active, but not making full use of the processors. For example when the partition utilization is low.
- CUoD processors that are not purchased

For example, if the system has a total of 8 available physical processors and the system administrator creates 3 different partitions with the following configuration.

- Partition A is a dedicated partition (“Allow when partition is inactive” is not checked) with 3 processors.
- Partition B is a shared partition with 2.5 processing units.
- Partition C is a shared partition with 2.0 processing units.

The shared pool will contain 5 processors shared between partition B & partition C. If partition A would have been created with “Allow when partition is inactive” checked and it was powered off, the shared pool (as long as partition A was off) would have 8 processors shared between partitions B & C each with their respective processing capacity (partition B has 2.5 processing units & partition C has 2.0 processing units).

In this sub-section we will examine 3 Scenarios:

- POWER5™ Model 570 running a partition of 6 processor units
 - Vary the VPs from 6 to 8
- POWER5™ Model 595 running larger partitions (multiple processor units)
 - 4 partitions of 8 processor units running in a shared pool of 32 processors
 - Vary the VPs from 8 to 32 for each partition
- POWER5™ Model 570 running small and micro partitions
 - to 5 partitions running in a shared pool of 4 processors
 - Vary the VPs

5.2.1 Results Overview

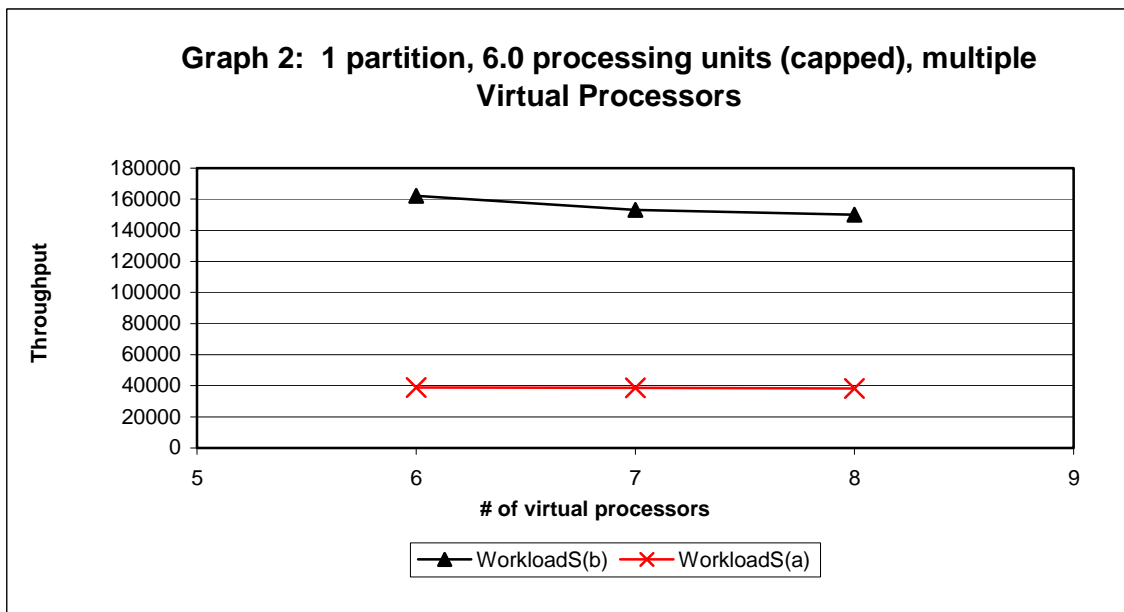
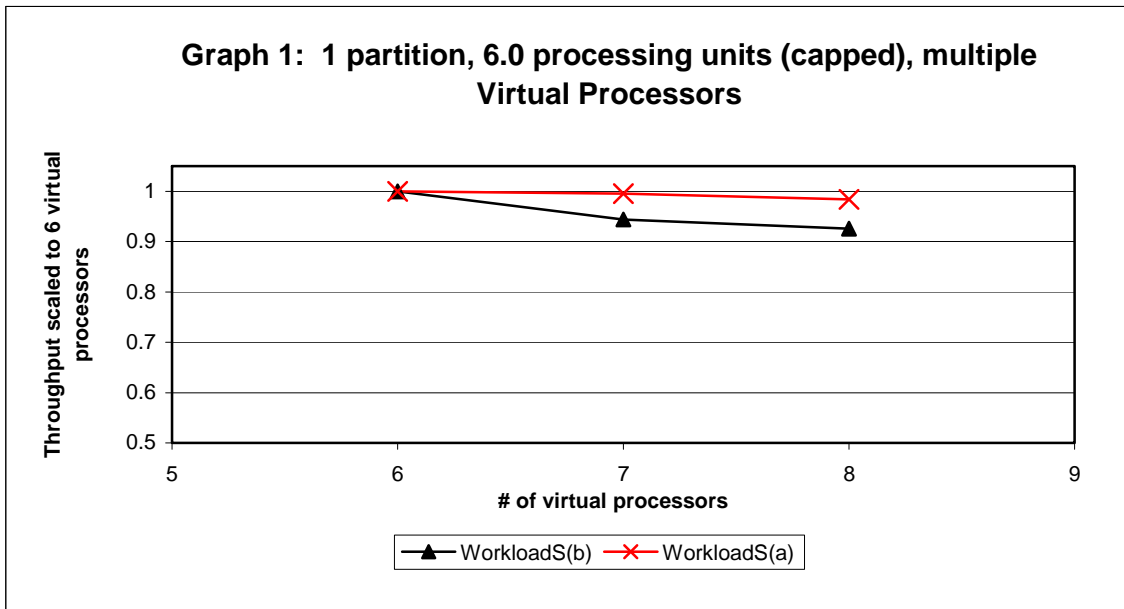
The results section will provide the following information relative to virtual processors:

- A brief description of the scenarios run
- Performance results for the measurements
- A performance summary & tips section

5.2.2 POWER5 Model 570 Virtual Processor Scenario

For this scenario, we defined a shared pool of 8 processors and created a single shared partition with 6.0 processing units. We started with the number of virtual processors defined at 6 and increased the number up to 8. The workloads we used for this experiment were as follows:

- WorkloadS is a transaction based, compute intensive Java application that simulates a warehouse distribution environment without utilizing any database IO. There are a few variations of WorkloadS:
 - WorkloadS(a) is running a few Java threads (low CPU utilization)
 - WorkloadS(b) is running many Java threads (high CPU utilization)



5.2.2.1 POWER5 Model 570 Virtual Processor Scenario: Performance Summary

Having more virtual processors than what is actually needed increases IBM i Operating System (formerly know as i5/OS) processing to manage virtual processors. On graph 1 above, WorkloadS(a) is constant as you increase the number of virtual processors. On this example, we only had a few Java threads running and we were not using all of the entitled capacity for the partition. WorkloadS(b) has many Java threads running and uses up all of the entitled capacity, so having more virtual processors without having excess capacity available has an impact on the partitions performance.

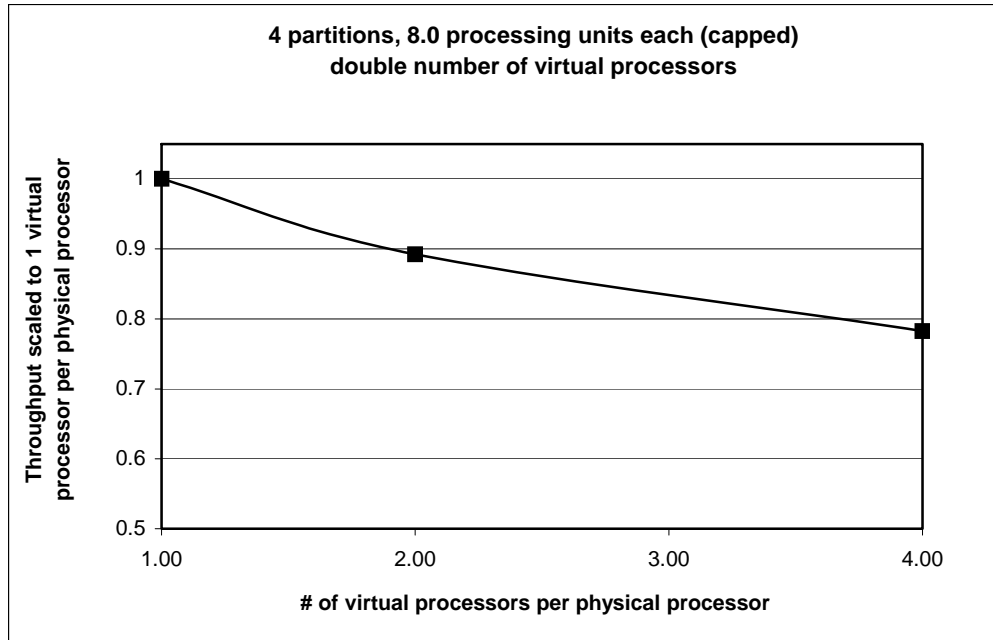
Graph 2 shows WorkloadS(a) has less overall throughput than WorkloadS(b) since in WorkloadS(a) the number of executing threads is lower & the entitled capacity is not fully utilized as it is with WorkloadS(b).

5.2.3 POWER5 Model 595 Virtual Processor Scenario

For this scenario, we defined a shared pool of 32 processors and created 4 identical shared partitions with 8.0 processing units each. The three points on the graph are as follows:

- Point 1 was 4 partitions running concurrently with 8.0 processing units & 8 virtual processors specified for each partition.
- Point 2 was 4 partitions running concurrently with 8.0 processing units & 16 virtual processors specified for each partition.
- Point 3 was 4 partitions running concurrently with 8.0 processing units & 32 virtual processors specified for each partition.

The workload that was used for this experiment was the CPW workload which represents a commercial computing environment.



5.2.3.1 POWER5 Model 595 Virtual Processor Scenario: Performance Summary

The results depicted in the above graph shows that as the number of virtual processors per physical processor is doubled, the throughput for the partition is reduced between 1% and 10%. By increasing the number of virtual processors at this rate, you are reducing the total throughput of the partitions. The throughput reduction is due to increased operating system and firmware processing and reduced processor efficiency from increased switching between the increased numbers of virtual processors. For this case, the CPU entitlement for a virtual processor of the partition is $8.0/8=1.00$ of a physical processor, $8.0/16=0.50$ of a physical processor, or $8.0/32=0.25$ of a physical processor.

5.2.4 POWER6 Virtual Processor Scenarios

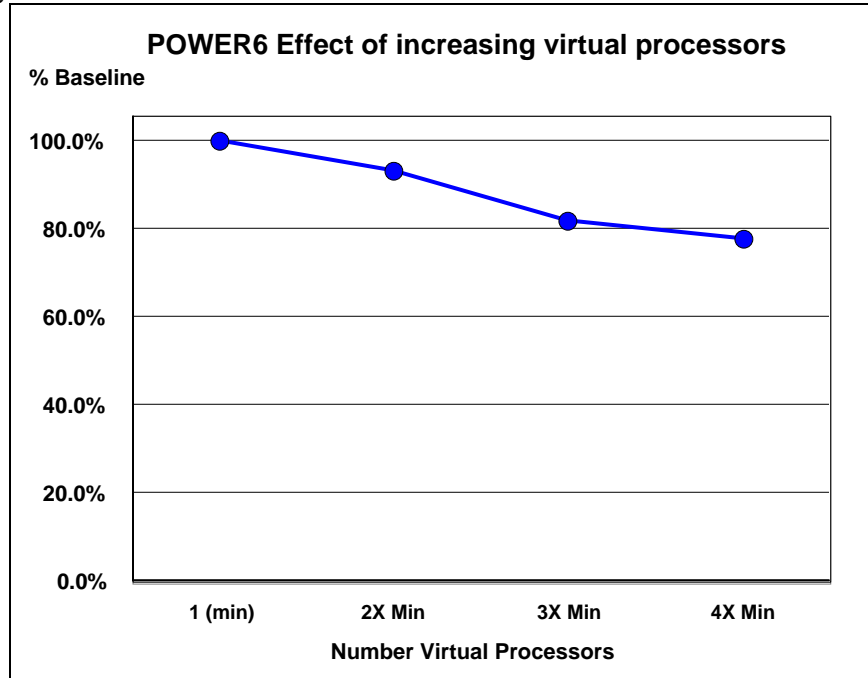
In the POWER6™ Virtual Processor scenarios we defined a shared pool containing a single processor and then ran Shared processor partitions with different settings for the number of virtual processors. The numbers of virtual processors per partition were varied from 1 (minimum number of virtual processors allowable) to 4 (4 times the number required). Results were compared against running the same Shared processor partition scenario at only 1 virtual processor. We ran the following scenarios:

- One 1.00 processor unit Shared processor partition
- Three 0.33 processor unit Shared processor partitions

Two different workloads were used to evaluate micro-partitioning performance: WorkloadV and WorkloadS. The results represent a composite average of the throughput of the two workloads.

Measurements were run on three operating systems (IBM i operating system 5.4.5, AIX 5.3L, and Linux SUSE SLES 10 SP1), and results are composite of all three OSs and workloads.

Measurements were run on a model 9408-M25 system that has no L3 cache and on a model 9117-MMA system that has an L3 cache.



5.2.4.1 POWER6 Virtual Processor Scenarios: Performance Summary

There were a total of 4 sets of measurement results {the results for both system models (9408-M25 and 9117-MMA) and both scenarios (the three 0.33 processor partitions scenario and the one 1.00 processor partition scenario)}. Each set of measurements yielded similar results, so for simplicity the results depicted in the above graph are the composite average of 4 sets of measurement results.

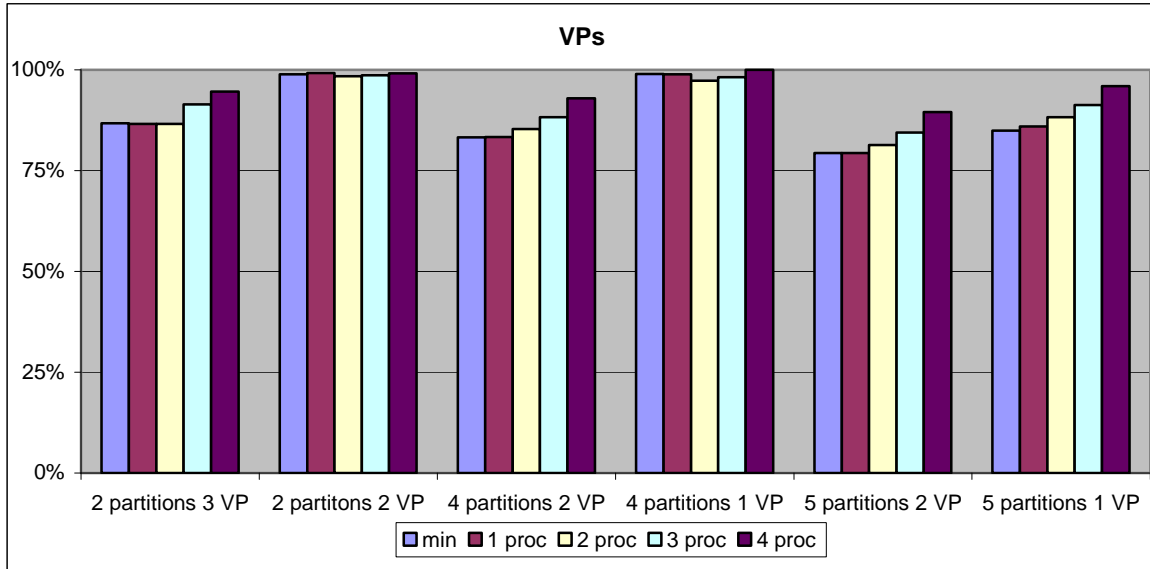
As was true for POWER5™, for POWER6™ as the number of virtual processors is increased the total throughput of the partitions declines. With POWER6™ as the number of virtual processors per physical processor is doubled, the throughput for the partition is reduced between 6% and 10%. By increasing the number of virtual processors at this rate, you are reducing the total throughput of the partitions. The throughput reduction is due to increased operating system and firmware processing and reduced processor efficiency from increased switching between the increased numbers of virtual processors.

5.2.5 POWER5 Model 570 Virtual Processor Study

During the POWER5™ Model 570 Entitlement Study, we noticed that performance improved greatly when the total number of virtual processors for all partitions equals the number of

processors in the shared pool. In the results (section 5.1.2 POWER5 Model 570 Entitlement Study) this could be seen in the 2 partition (with 2 virtual processors per partition) on a 4 processor shared pool.

For this analysis, we ran tests on the 4-way 570 where the number of virtual processors per partition was changed. For the 2 partition configuration, we increased the virtual processors to 3 per partition (6 VPs total in the shared pool of 4 physical processors). For the 4 and 5 partition runs, we decreased the virtual processors from 2 per partition to only one (respectively 4 and 5 VPs total in the shared pool of 4 physical processors). The results:



5.2.5.1 POWER5 Model 570 Virtual Processor Study: Performance Summary

The results from this study showed in all cases when the number of virtual processors is decreased the throughput total of all partitions increases. Also, the performance is greatly increased when the total number of virtual processors of all partitions is equal to the number of processors in the shared pool.

5.3 Performance Tips for Uncapped Processing and Virtual Processor

Here are a few tips that should be considered for Virtual Processor and Uncapped Processing when creating shared partition environments:

1. When creating a capped partition, for maximum processor efficiency and partition CPU capacity, the number of desired virtual processors should be the minimum that can consume the desired entitled capacity. For example,
 - If the desired entitled capacity is 3.6 processing units, the number of desired virtual processors should be 4.
 - If the desired entitled capacity is 0.75 processing units, the number of desired virtual processors should be 1.

2. When creating an uncapped partition, for maximum processor efficiency and partition CPU capacity,
 - a) Do not make the number of virtual processors for a partition greater than processors in the shared processor pool. The number of processors in the shared pool is the maximum number of physical processors a partition could use concurrently. For example:
 - If you specify 4.0 processing units and there are 8.0 processors in the shared processing pool, the partition can not make use of more than 8 processors at a time, so the recommendation is to specify no more than 8 virtual processors.
 - b) Do not set the number of virtual processors for a partition greater than the number of available processing units. Other shared partitions will be utilizing their entitled processing units so in many cases the entire shared pool size is not available for a single partition to use.
 - c) Where possible, set the partition's entitled processing units as close to the anticipated CPU processing requirements as possible. The more CPU processing partitions use as uncapped (i.e. beyond their entitlement) the greater the processor efficiency impacts caused by increased virtual processor switching.
 - d) When attempting to take advantage of unused shared pool resources, set the number of virtual processors close to the expected capacity you are trying to achieve for that partition. For example: If you specify 4.0 processing units, you think that you will only use 2.0 more processing units during peaks and there will be unused capacity in the shared pool, make the number of virtual processors equal to 6.
3. Setting Virtual processors to higher values usually results in reduced processor efficiency and can result in decreased performance from increased contention.

6 Mixed OS

A system running different operating systems concurrently can be called a “mixed OS” system. IBM’s Power™ hardware supports the running of four different operating systems: IBM i (previously known as i5/OS), AIX, SUSE Linux, and RedHat Linux. This section describes what kind of performance effects you may encounter when mixing operating systems on POWER5™ hardware.

The short answer is that there is typically no effect from mixing operating systems. The Power™ hypervisor keeps the partitions from directly interacting with each other. At the same time, the hypervisor does not treat the operating systems differently (i.e. no operating system gets any special treatment when it comes to allocating resources).

Two workloads were used to evaluate mixed OS.

- WorkloadV is a highly threaded, CPU intensive Java workload that simulates sending/receiving short communication messages across TCP/IP socket interfaces.
- WorkloadS is a transaction based, compute intensive Java application that simulates a warehouse distribution environment without utilizing any database IO.

The measurements were done on an IBM iSeries 570 8-way POWER5™ system. Results compare baseline measurements (where all partitions ran the same operating system) against mixed OS measurements (where different operating systems were run together). Various combinations were measured to determine if the POWER5™ Hypervisor behaved differently based on the operating system. All measurements were made running three partitions. The combinations consisted of:

- i5/OS + AIX + SUSE Linux
- i5/OS + AIX + RedHat Linux
- i5/OS + SUSE Linux + RedHat Linux
- AIX + SUSE Linux + RedHat Linux

These different operating system combinations were run in three configurations:

- three 2-way dedicated processor partitions with 30 GB memory each
- three 2-way shared processor partitions with 30 GB memory each
- three 0.33 shared processor partitions with 5 GB memory each

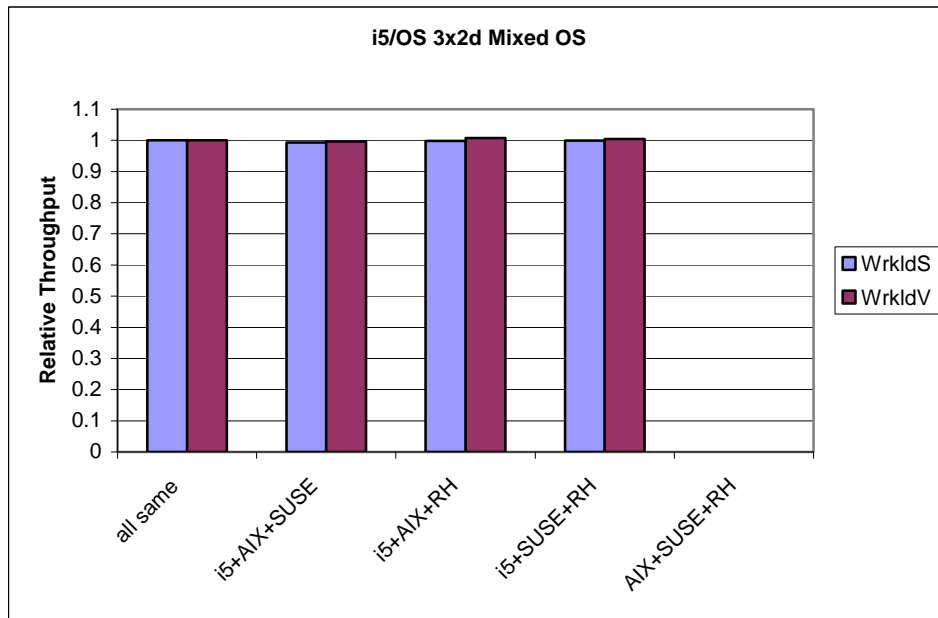
6.1 Dedicated Processor Partitions

The results are shown in a series of charts. Each chart shows the results of four runs, and each of those runs has been normalized to a non-mixed OS baseline result. For example, the following chart shows the results of running dedicated processor partitions in a mixed OS environment where i5/OS is always present. The bars on the far left are the baseline results with three i5/OS partitions (the average throughput per i5/OS partition). The remaining pairs of bars show the throughput of the i5/OS partition when run in combination with other operating systems. The

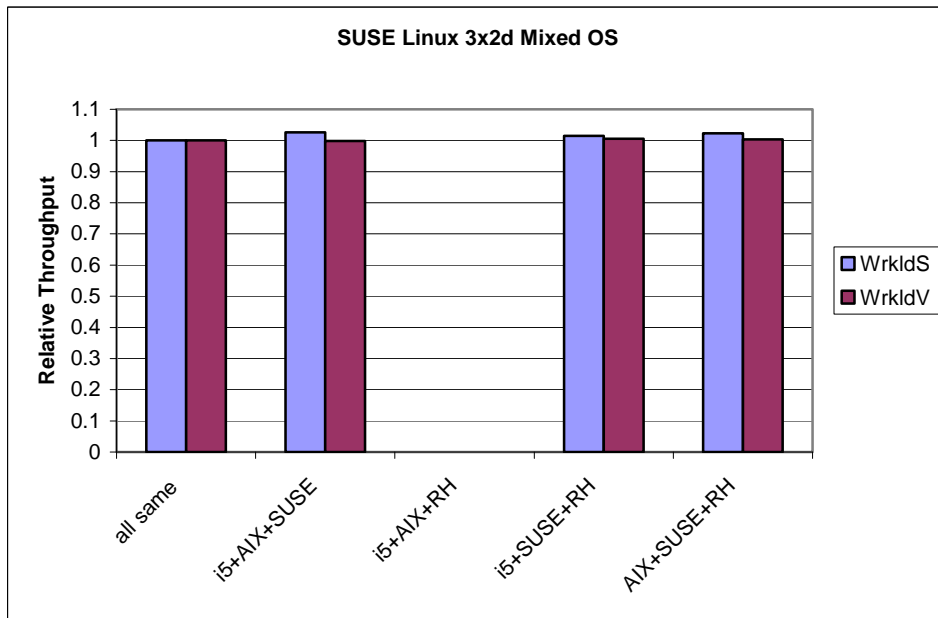
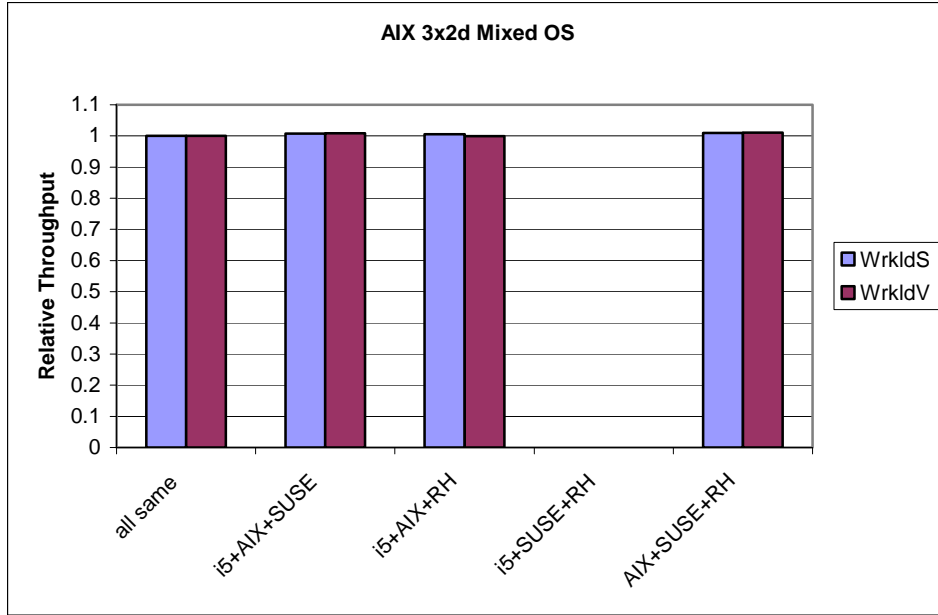
empty space at the right of the chart is the OS combination where i5/OS is not present, and is left in to provide continuity between charts.

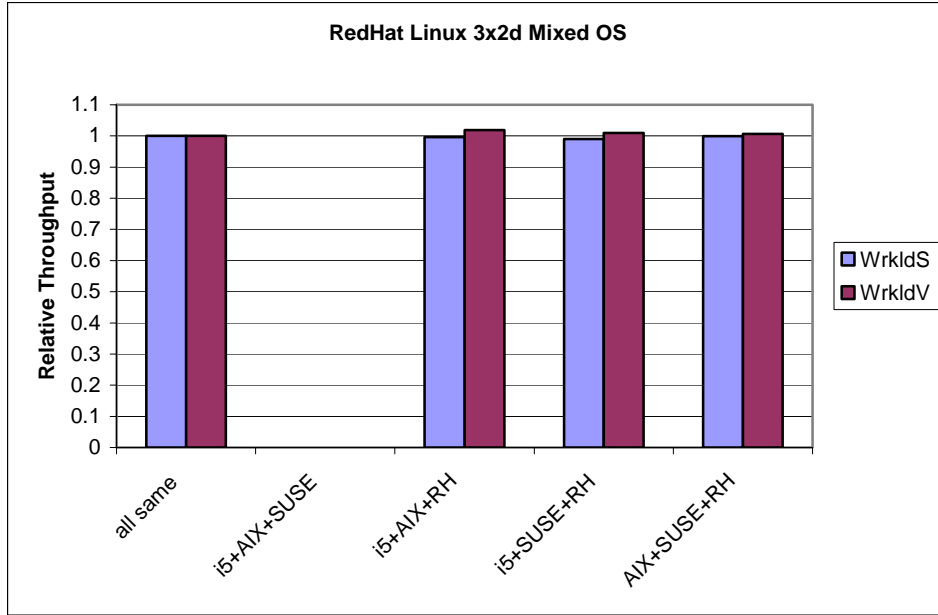
The charts demonstrate that there is no measurable difference when mixing operating systems in this environment. The amount of difference between the results is within measurement variation for each specific workload.

These dedicated processor runs were purposely done using 2-processor partitions to eliminate sharing of L2 caches. 30 GB of memory was used to ensure that all of the memory for the partition would be allocated locally based on the processors. These settings allow us to focus on any hypervisor behavior while reducing any hardware interactions between partitions.

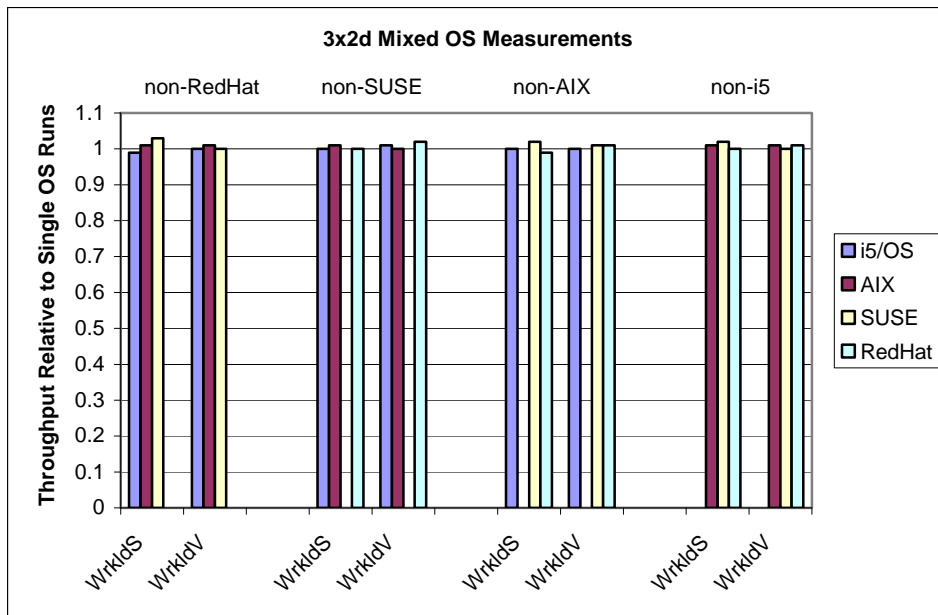


The following three charts are for each of the other operating systems when running in a 2-way dedicated processor partition environment.





The following chart is another way of looking at all the same data in the four charts above, combined into one chart. In this chart the data from four runs are shown, again normalized to runs done using a single operating system type. On the far left are the normalized results when running all but the RedHat Linux partition. Each bar is still normalized to the throughput that particular operating system achieved when running in a homogeneous operating system environment.



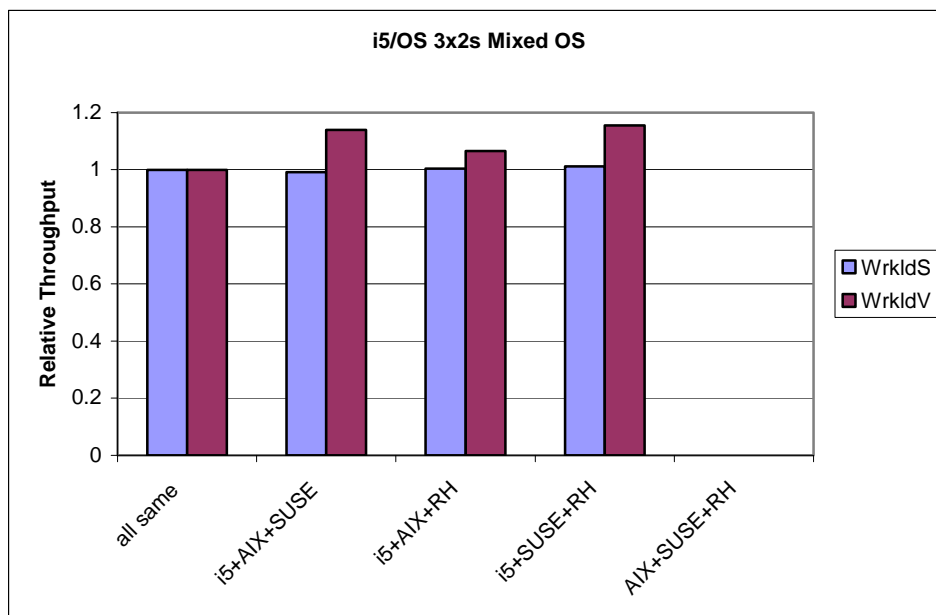
6.2 Whole, Shared Processor Partitions

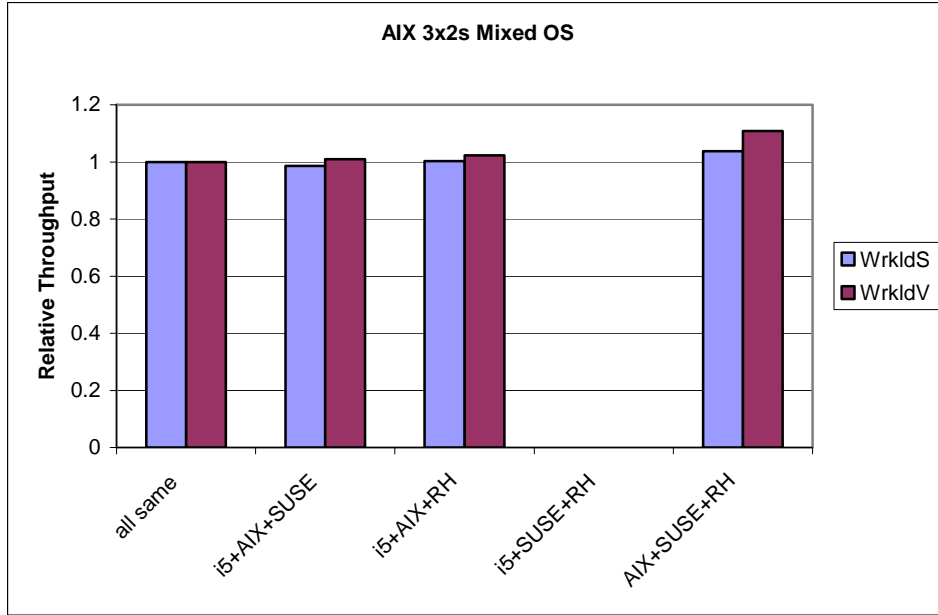
As mentioned before, the dedicated processor partition runs were done in such a way as to reduce or eliminate hardware interactions between partitions. When running shared processor partitions, the partitions can be scheduled on any processor in the shared processor pool. So while a 2-way dedicated partition can avoid cache conflicts from other partitions, 2-way shared processor partitions will have some amount of cache sharing with other partitions. The hypervisor's scheduling algorithms do reduce this by giving partitions affinity to certain processors, but the sharing cannot be eliminated as it can in dedicated environments.

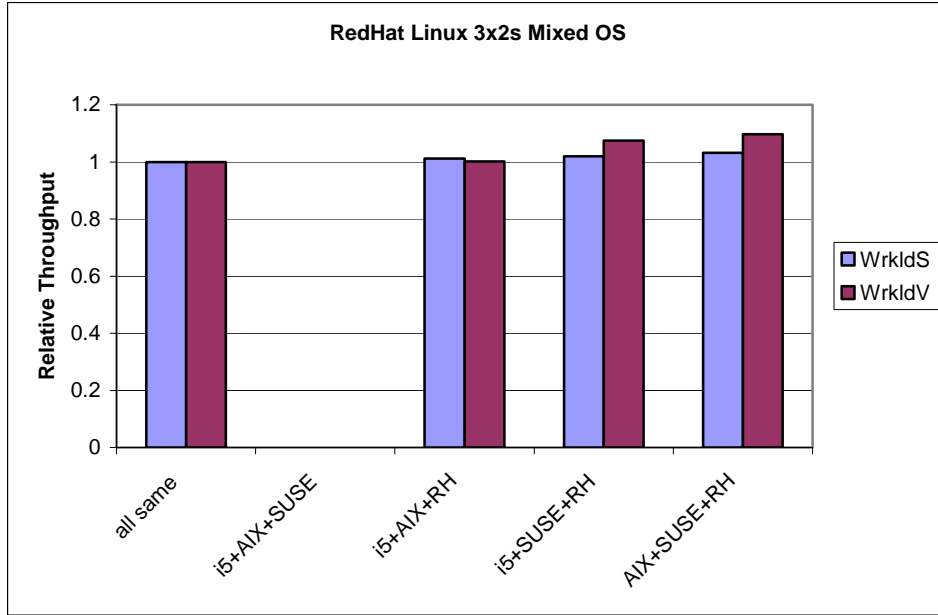
The one area where some effect could be seen from mixing operating systems was where the memory usage of one operating system indirectly caused some performance impact due to L2 cache sharing. One of the two workloads used, WorkloadV, is highly dependent on the L2 cache. In addition, this workload is even more cache sensitive when run on i5/OS partitions. This results in two effects. The first effect is that an i5/OS partition running WorkloadV concurrently with non-i5 partitions gets better performance than the baseline, where three i5/OS partitions are sharing L2 cache capacity. The other effect is that non-i5/OS partitions get slightly better WorkloadV throughput when they are not competing with an i5/OS partition for cache resources.

This effect is not considered significant for several reasons. WorkloadV is a highly specialized workload that is more sensitive to L2 cache than a typical customer workload. In most runs the cache effect was less than 10% and at most 15%, and was measured when the partitions were running at levels close to 100% CPU utilization.

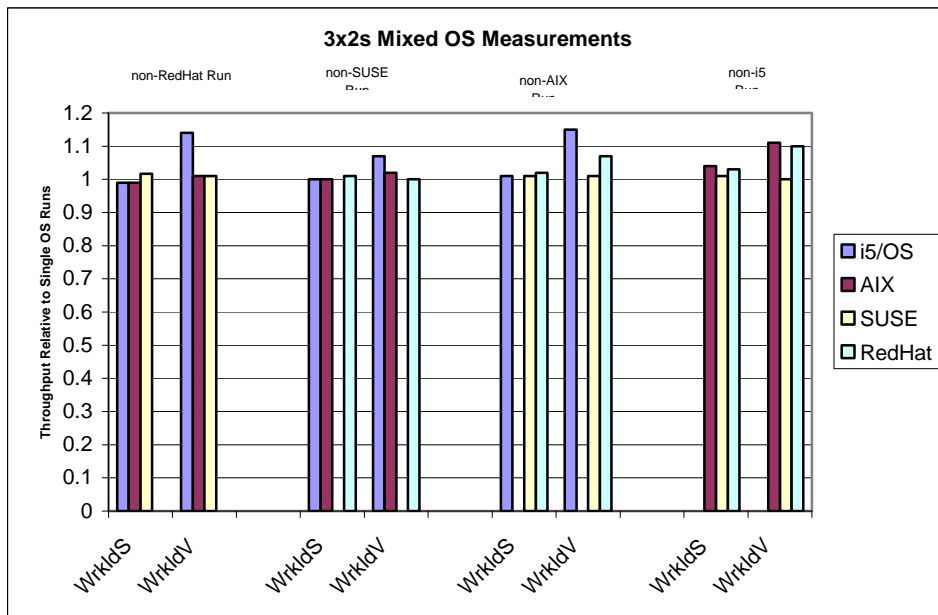
Once again the results are shown grouped by operating system.







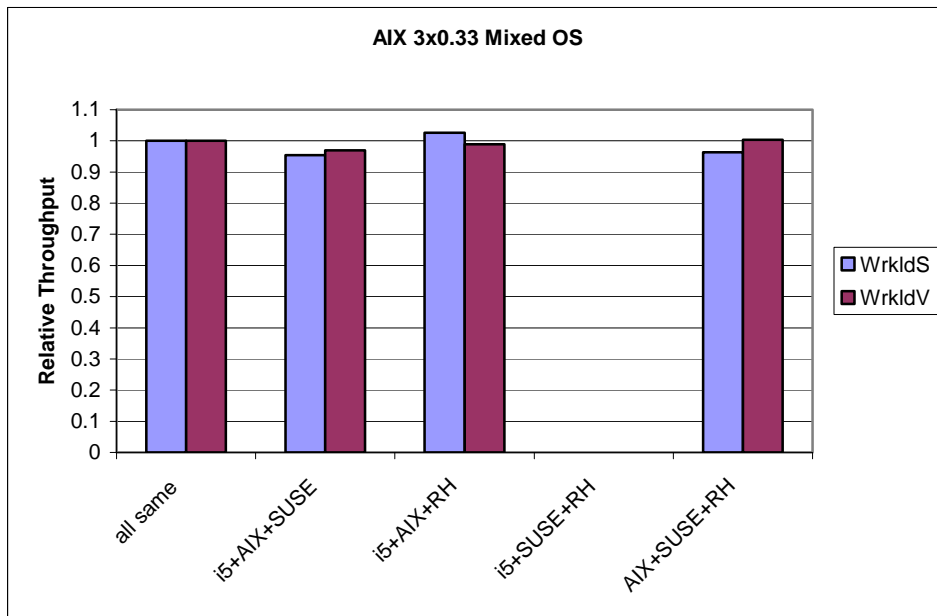
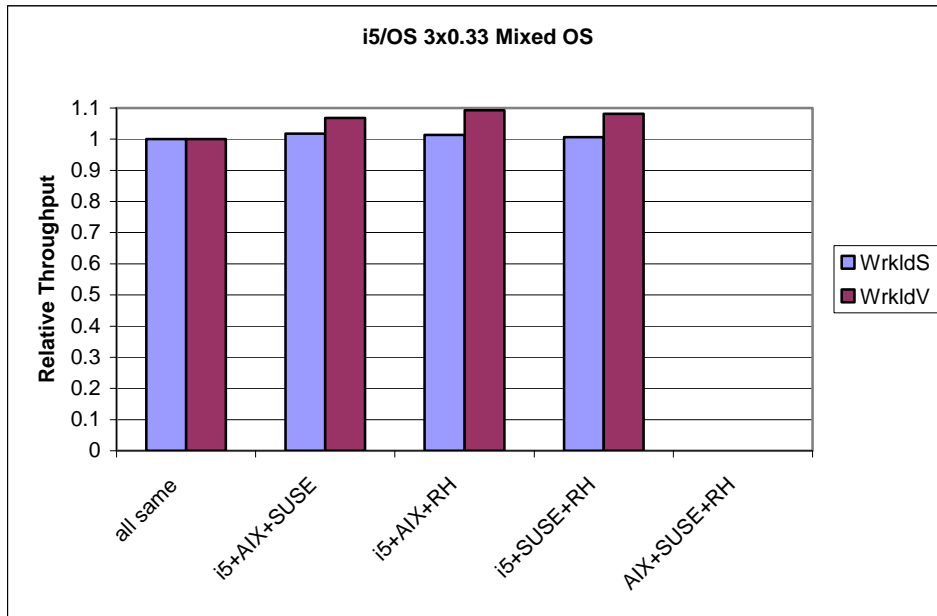
The following chart shows the results grouped by measurement.

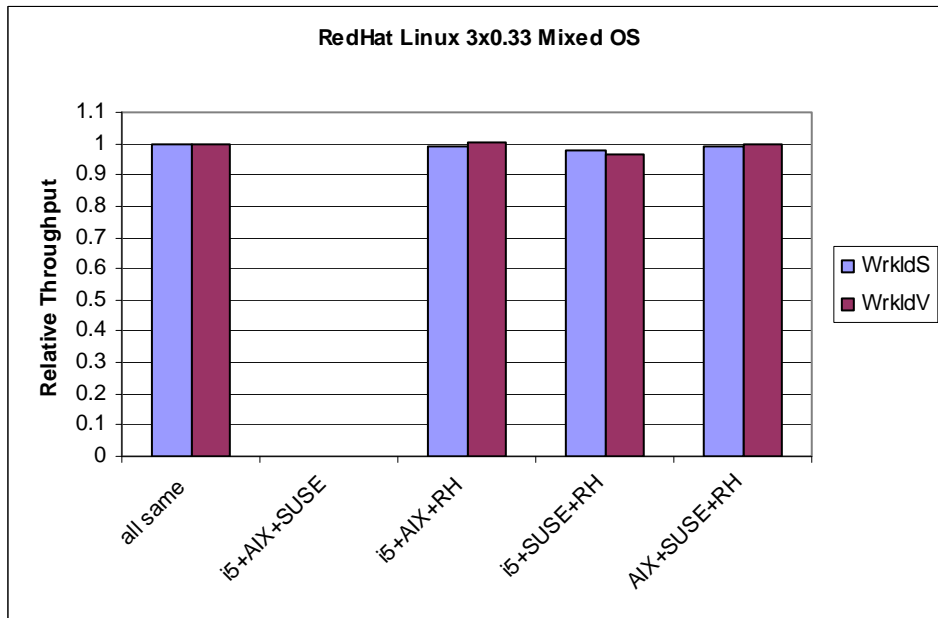
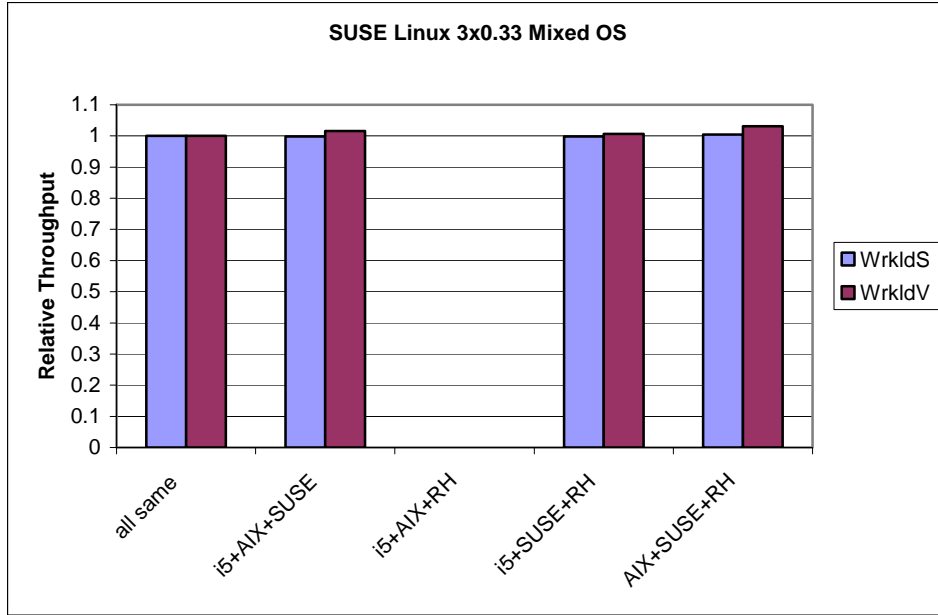


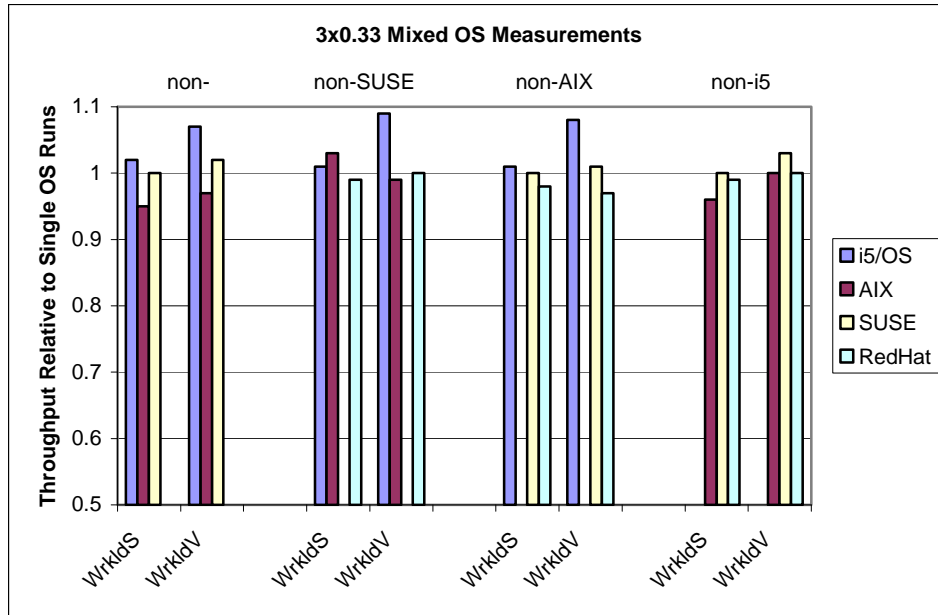
6.3 Partial Processor Partitions

The following set of mixed operating system measurements was done using partial processor partitions. All measurements were done with three partitions having 0.33 processing units each. The remaining processors were made unavailable such that all of the partitions were running on the same processor (and therefore sharing the same L2 cache). There is more run-to-run measurement variation in this environment, but the one characteristic that stands out once again

is the behavior of WorkloadV. i5/OS is able to run WorkloadV slightly better when it is running with non-i5/OS partitions.







6.4 Conclusions

Generally, mixing operating systems in partitioned environments should have minimal performance effect. The hypervisor does not give preferential treatment to any operating system at the expense of others. The only performance effect comes indirectly from the sharing of resources such as caches and memory subsystems. If there is any concern about this indirect effect when running multiple operating systems concurrently, using dedicated processor partitions in multiples of two processors ensures that the partitions do not share cache.

7 Memory Allocation and Performance

In this section, the term ‘node’ refers to the POWER5 or POWER6 packaging of processor chips. On High end systems (e.g. 590, 595) the node is 4 processor chips in a module. On low and midrange systems (e.g. 520, 570, etc) the node is a package of 1 processor chip with either 1 or 2 processors active.

When partitioning multi-node systems, how the memory of a partition is allocated across the nodes on the managed system can affect performance. When a managed system is first powered up, the processor and memory allocation is optimized for the partitions that were active just before the managed system was IPLed. When these partitions are activated, they will generally have optimal memory allocation. By optimal we mean that dedicated processor partitions will have memory that is local to their processors, and shared processor partitions will have their memory spread across the nodes used by the processors in the shared processor pool. When partitions that were not active before the managed system was IPLed are activated, their memory allocation will also generally be allocated as expected.

Unexpected, sub-optimal memory allocation situations can increase over time though, as previously used partitions are powered down or change their memory requirements, and when new partitions are activated. This occurs due to the HMC and hypervisor needing to take memory away from previously activated partitions in order to give it to newly activated partitions. This can result in partitions performing somewhat worse than if they had optimal memory allocation. This is generally a problem for dedicated processor partitions which end up not getting memory allocated on a node where the partition’s processors are. The result can be two identically configured partitions performing at different levels, due to differences in how those partitions’ memory is allocated. Unfortunately there is no way to tell what the current memory allocation for partitions across nodes is on customer machines. Partition memory allocation can be re-optimized by shutting down all partitions and doing a full managed system IPL. Work is underway to help alleviate this problem in future releases of the hypervisor.

Lab measurements on a four node POWER5 system using a memory, CPU, and I/O intensive workload saw up to a 6% reduction in throughput when a 4-way dedicated processor partition went from having all of its memory local to the nodes its processors were on, to having no local memory. This same workload saw a 2% reduction in throughput when its memory was spread across all four nodes on the system vs. having its entire memory local to the processors. It is more likely a partition would end up in a situation where its memory is partially spread, than for its memory to be totally remote. Other simple Java workloads that did not have high memory usage saw less than a one percent throughput reduction when running with all local vs. all remote memory.

8 Trademarks and Disclaimers

©IBM Corporation 1994-2008. All rights reserved.

References in this document to IBM products or services do not imply that IBM intends to make them available in every country.

The following are trademarks of the International Business Machines Corporation in the United States and/or other countries. For a complete list of IBM Trademarks, see www.ibm.com/legal/copytrade.shtml:

AS/400®	e-business on demand®	IBM i operating system®
AS/400e™	IBM™	i5/OS®
eServer™	IBM (logo)®	OS/400®
iSeries®	System i™	AIX®
System p™	WebSphere®	pSeries®
System i5™	Micro-Partitioning™	POWER™
POWER Hypervisor™	Power™ Systems	Power™ Systems Software
PowerVM™	Power Architecture®	POWER4™
POWER4+™	POWER5™	POWER5+™
POWER6™		

The following are trademarks or registered trademarks of other companies:

Java and all Java based trademarks and logos are trademarks of Sun Microsystems, Inc., in the United States and other countries or both

Microsoft, Windows, Windows NT and the Windows logo are registered trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

UNIX is a registered trademark of The Open Group in the United States and other countries or both.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

NOTES:

Any performance data contained in this document was determined in a controlled environment. Actual results may vary significantly and are dependent on many factors including system hardware configuration and software design and configuration. Some measurements quoted in this document may have been made on development-level systems. There is no guarantee these measurements will be the same on generally-available systems. Users of this document should verify the applicable data for their specific environment.

LPAR Performance on Power Systems with POWER4™, POWER5™ and POWER6™

©Copyright 2008 IBM. All rights reserved.

Page 46 of 48

IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.

Information is provided “AS IS” without warranty of any kind.

All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.

This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.

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

Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Prices are suggested US list prices and are subject to change without notice. Starting price may not include a hard drive, operating system or other features. Contact your IBM representative or Business Partner for the most current pricing in your geography.

Any proposed use of claims in this presentation outside of the United States must be reviewed by local IBM country counsel prior to such use.

The information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM makes no representation or warranty regarding third-party products or services including those designated as ServerProven, ClusterProven or BladeCenter Interoperability Program products. Support for these third-party (non-IBM) products is provided by non-IBM Manufacturers.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. Send license inquires, in writing, to IBM Director of Licensing, IBM Corporation, New Castle Drive, Armonk, NY 10504-1785 US