IBM Technical Forum & Executive Briefing

17 al 21 Octubre 2011

Imagine PODER Imagine CAPACIDAD

PowerVM Processor Virtualization

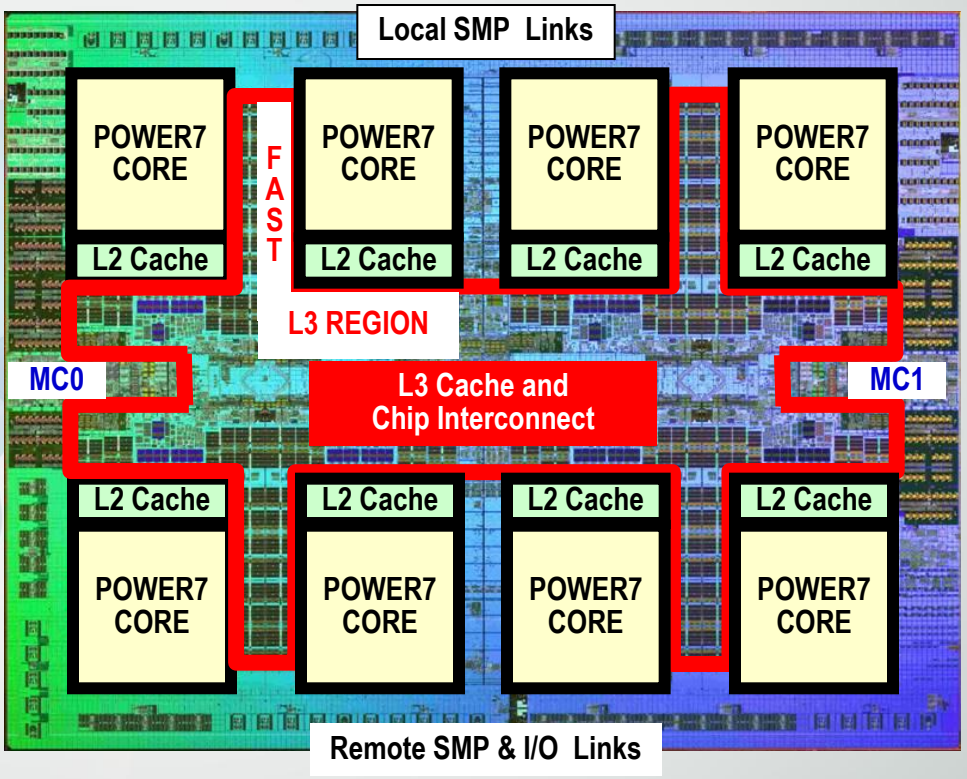
Conceptos y capacity planning

Cesar Diniz Maciel
Executive IT Specialist – IBM Power Systems
Global Techline – Latin America
cmaciel@us.ibm.com

- **Virtualization**
  - POWER7 & SMT
  - Micro-partitioning
  - Virtual processors
  - shared pools
  - Tools
  - Tuning

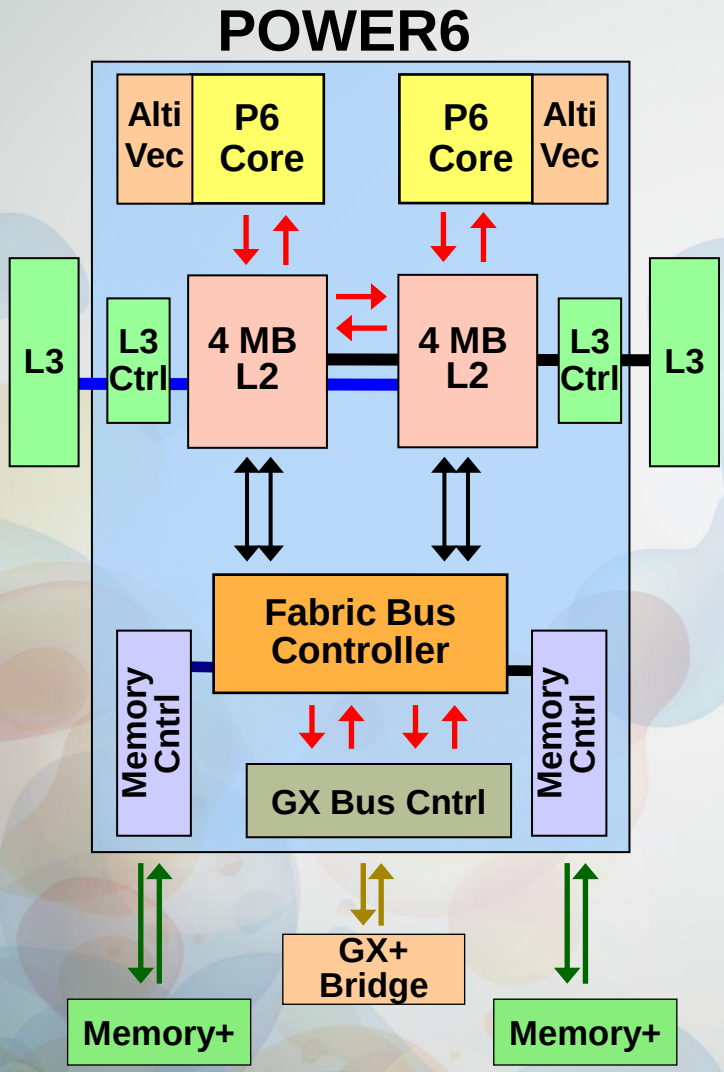
POWER7 Processor Chip

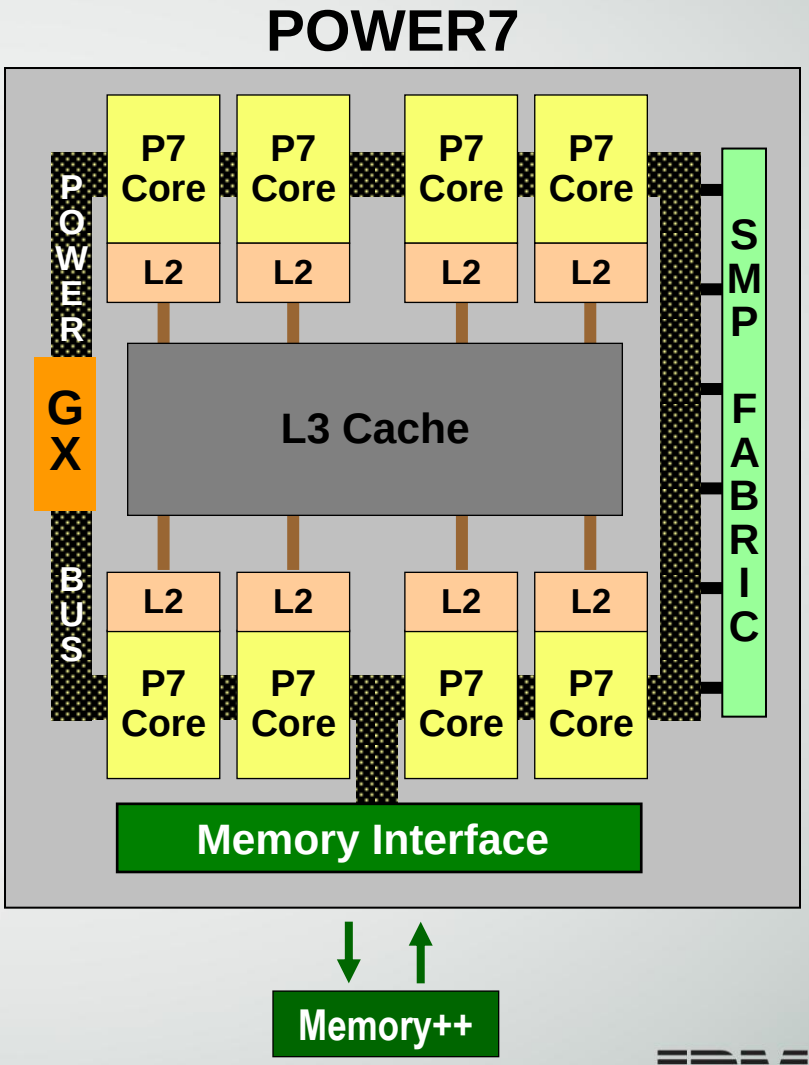
- **567mm² Technology: 45nm lithography, Cu, SOI, eDRAM**
- **2.7B equivalent transistors**
  - Actual count 1.2B
  - eDRAM leads to energy efficiency
- **Eight processor cores**
  - 12 execution units per core
  - 4 Way SMT per Core
  - 32 Threads Chip
  - 256KB L2 per Core
- **32MB on chip eDRAM shared L3**
- **Dual DDR3 Memory Controllers**
  - 100GB/s Memory Bandwidth per Chip sustained
- **Scalability up to 32 Sockets**
  - 360GB/s SMP Bandwidth/Chip
  - 20,000 operations in flight
- **Hardware instruction and data pre-fetch**
- **Binary Compatibility with POWER6**



* Statements regarding SMP servers do not imply that IBM will introduce a system with this capability.

Transition to POWER7



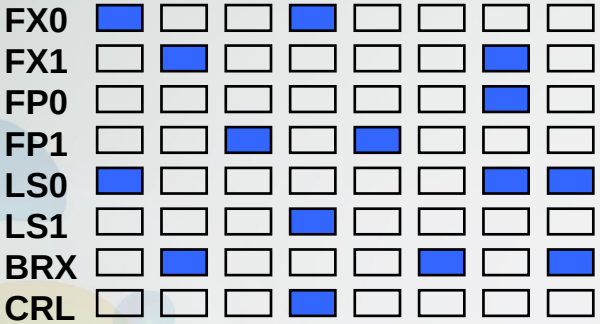


Simultaneous Multi-Threading

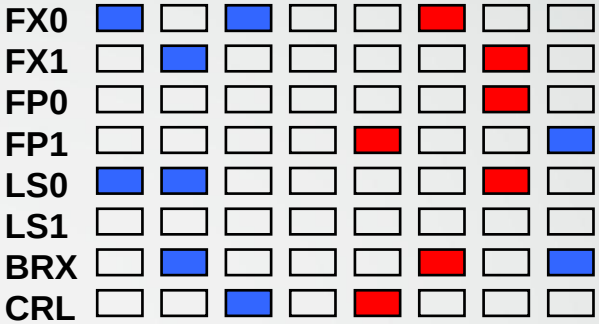
- **Simultaneous Multi-threading refers to the ability of a single core to support multiple hardware execution threads**
  - This technology allows for a core to execution more instructions per cycle
  - The number of threads can be controlled dynamically via the smtctl command
- **Dedicated Processor Partitions switch from simultaneous multi-threaded mode (SMT) to single-threaded mode (ST) automatically at low multi-programming levels**
  - On POWER5, Micro Partitions do not switch SMT/ST modes automatically
- **POWER6 had key technical improvements over POWER5 in multi-threading which dramatically reduce SMT effects in Micro partitions**
  - On POWER6 Micro partitions do switch SMT/ST modes automatically
  - On POWER6, on each cycle the hardware core may dispatch instructions for both hardware threads
- **POWER7 extends this capability from two to four simultaneous threads**

3rd Generation Multi-threading Capabilities: SMT4

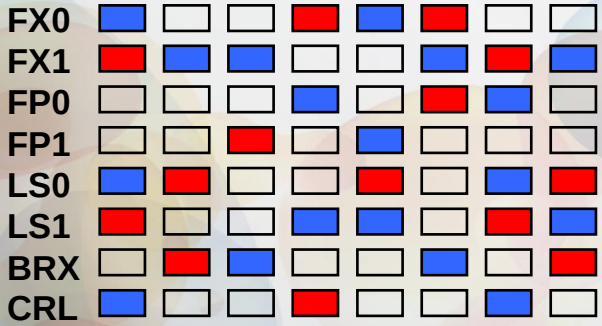
Single thread Out of Order



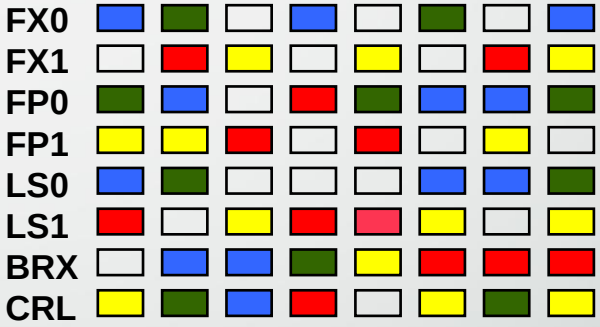
S80 Hardware Muti-thread



POWER5 2 Way SMT



POWER7 4 Way SMT



No Thread Executing

Thread 0 Executing

Thread 1 Executing

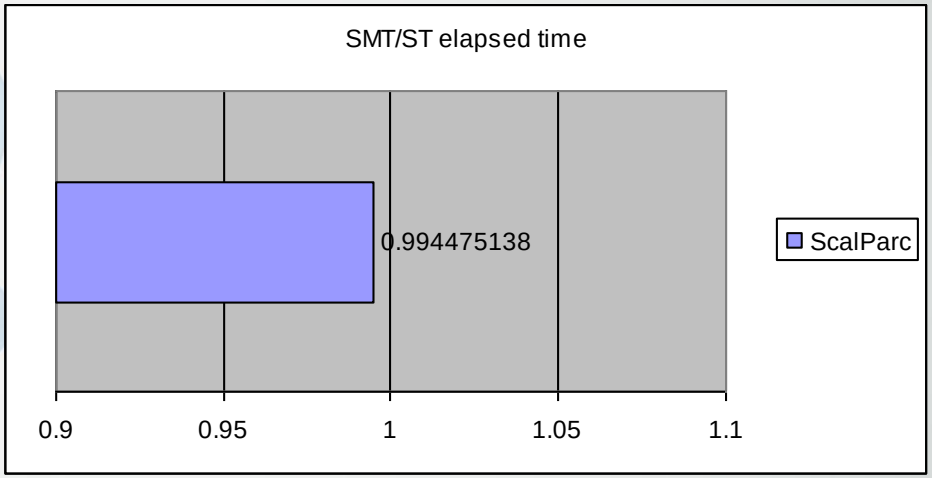
Thread 2 Executing

Thread 3 Executing

IBM

ST vs SMT in Micro partitions – POWER6 example

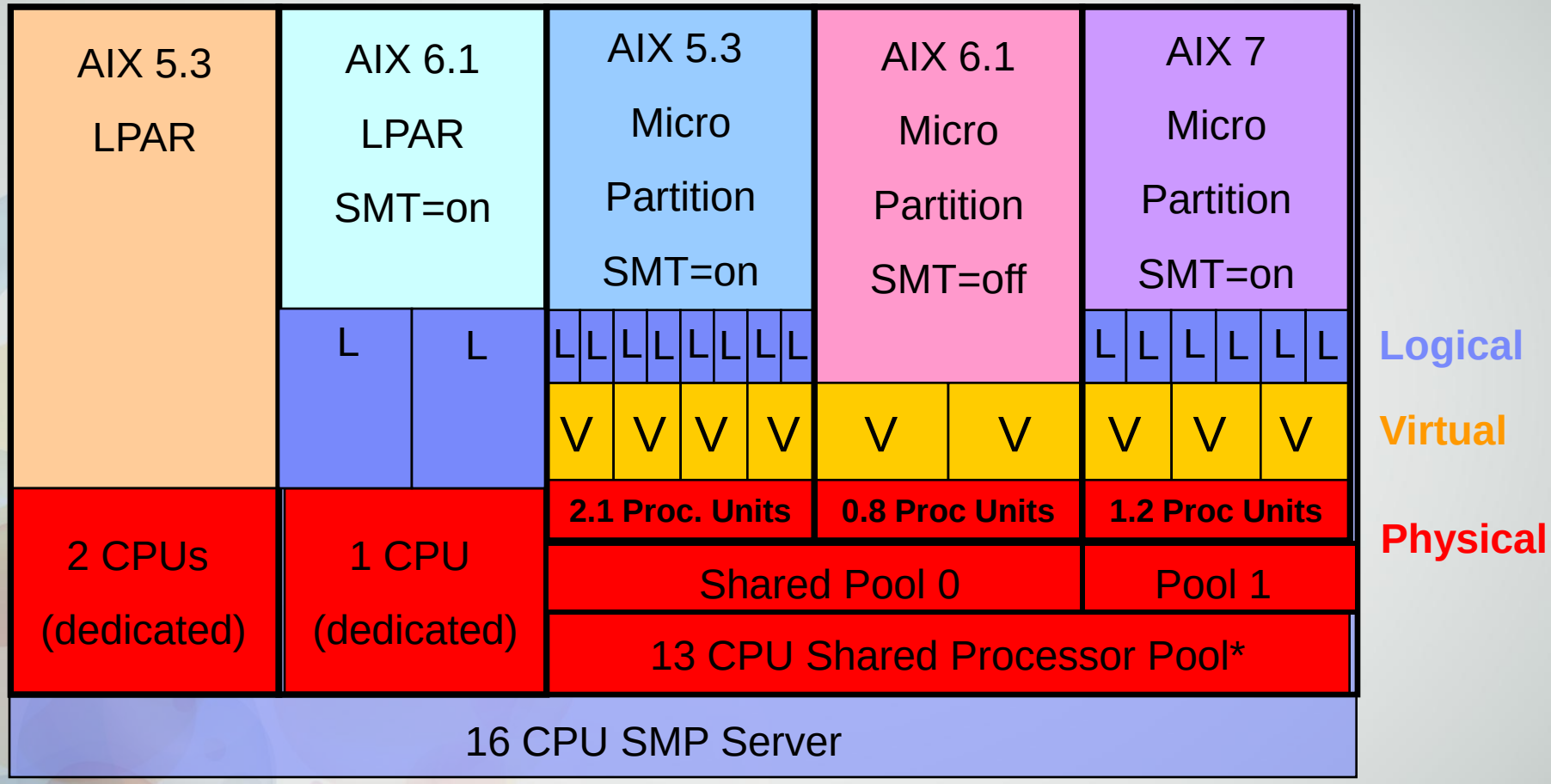
- **Generally, see perhaps 1% impact from running in SMT mode in Micro partitions on POWER6**
- **Example code from Northwestern University Minebench 1.0, a single-threaded "CPU hog"**
- **Shows the ratio of the test running in a Micro partition in SMT mode / ST mode**
- **It just works – no tuning required**



Shared Processor LPARs (Micro-partitions) - Definitions

- **LPARs are defined to be dedicated or shared**
  - Dedicated partitions use whole number of CPUs
  - Shared partitions use whole or fractions of CPUs (smallest increment is 0.1, can be greater than 1.0)
- **Shared processor pools - subset (or all) of physical CPUs in a system**
  - Desire is to have all of the installed processors in the shared pool and no dedicated CPU LPARs.
- **Entitled capacity expressed in the form of number of 10% CPU units**
  - Desired: Size of partition at boot time
  - Minimum: Partition will start will less than desired, but won't start if Minimum capacity not available
  - Maximum: DLPAR changes to desired cannot exceed this capacity
  - Divided among all of the LPARs within a shared processor pool
  - Uncapped capacity cannot exceed number of virtual processors for an LPAR
- **Capped vs uncapped**
  - Capped: CPU Capacity limited to desired setting.
  - Uncapped: CPU Capacity limited by unused capacity in 'pool' and cannot exceed number of virtual processors (not related to maximum processing units)
- **Shared Pool LPARs run in 'virtual' processors**
  - Time slicing of CPUs between partitions
- **Priority weighting to determine preference for spare cycles**
  - Automatic Load Balancing (default is 128, 0 implies no use of spare cycles, 255 is max priority)

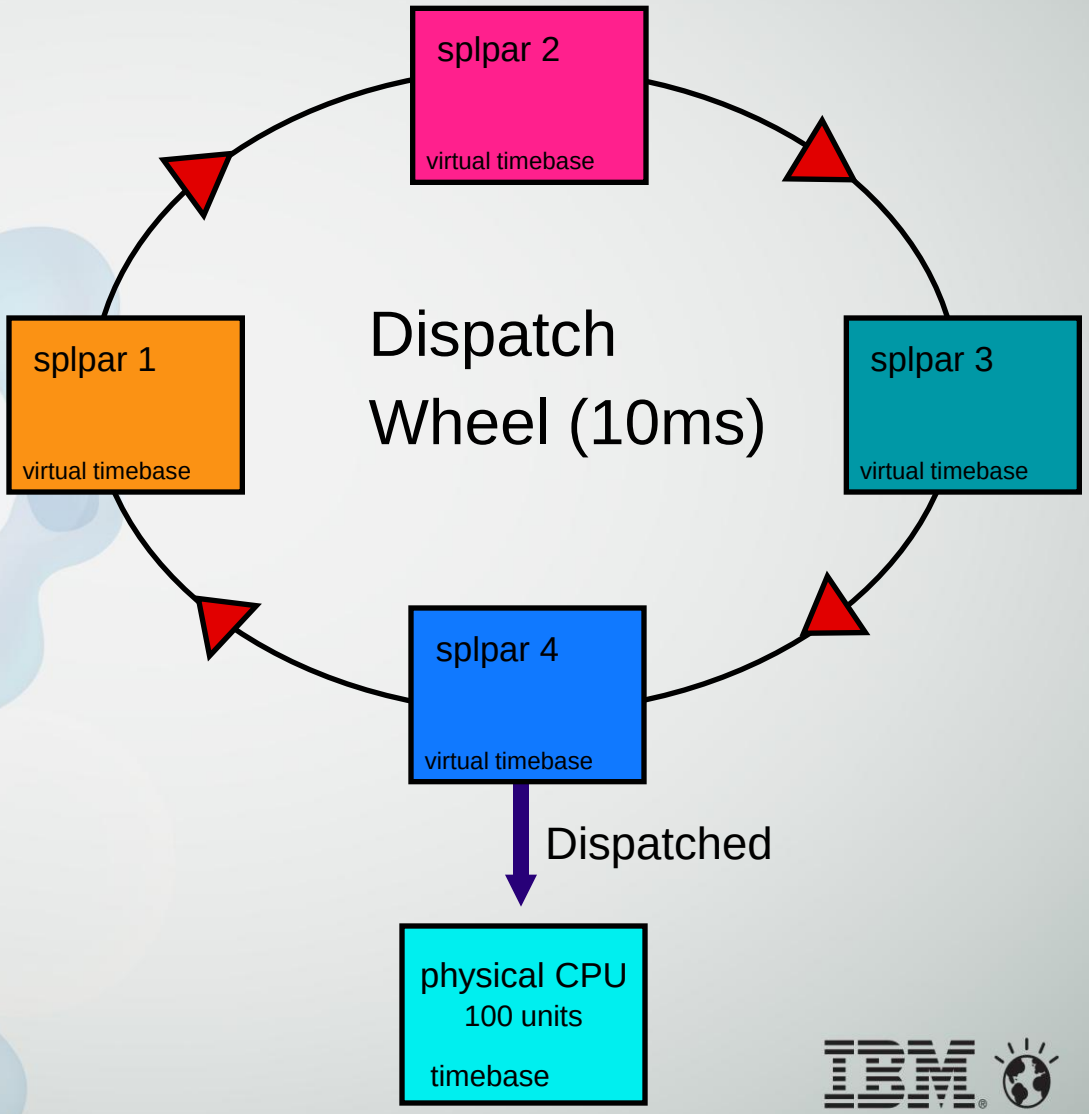
Physical, Logical, Virtual Layers



Think "PVL " **P=Physical** **V=Virtual** **L=Logical (SMT)**

Micropartitions Summary

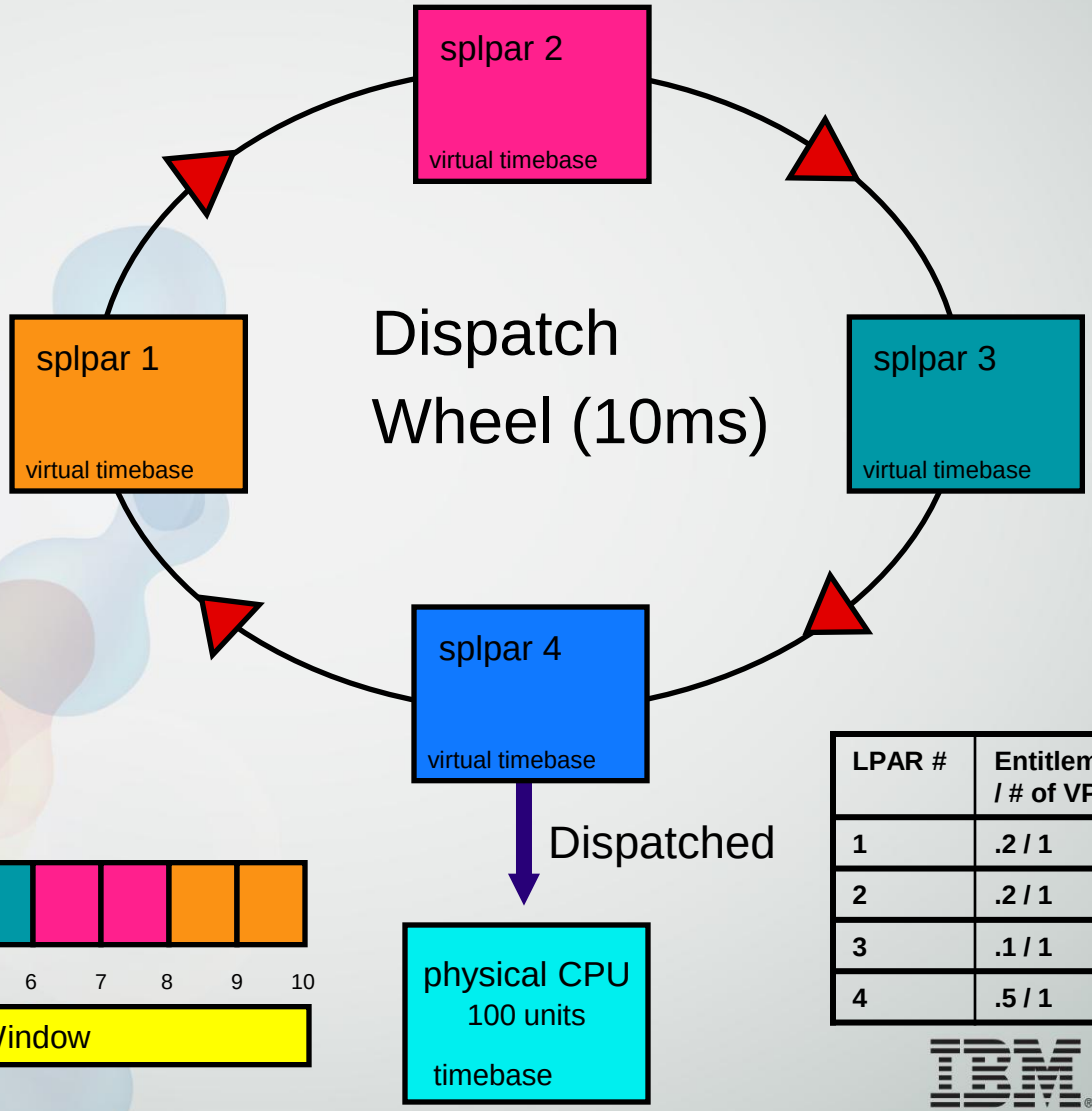
- Shared Processor concepts
  - Partitions run on a Virtual Processor (VP).
  - VP runs on Physical Processors (PP) only part of the time.
  - A VP has one or two logical processor depending on the SMT state.
  - Minimum size of a partition is .1 with increments of 1/100th of a processing unit.
  - A partition's capacity is defined by the entitlement and for uncapped partition by the number of VPs.
- Phyp (hypervisor) is responsible for scheduling & dispatching VPs on PPs.
  - Using a 10msec dispatch wheel.
  - Partition's time become "virtual", which is maintained by the phyp in the partition's PURR.

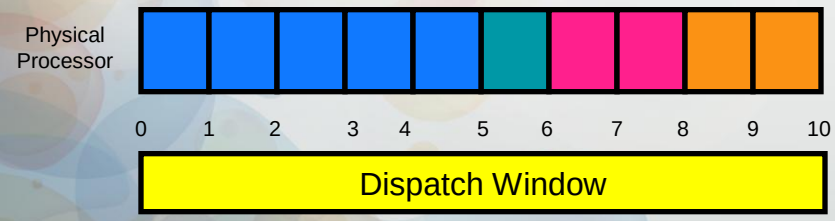


Hypervisor Dispatch Algorithm

The diagram illustrates the hypervisor dispatch algorithm, which can be viewed using the metaphor of a “wheel” with a fixed rotation period of 10 ms to guarantee that each VP will receive it’s share of entitlement in a timely fashion.

- At time period 0 a new 10 ms dispatch window and splpar 4’s VP is dispatched to a physical processor, and will run of 5 msecs.
- At time period 5, splpar 3’s VP is dispatched for 1 msecs
- At time period 6, splpar 2’s VP is dispatched for 2 msecs
- Finally, at the end of the 10 ms dispatch window, splpar 1’s VP is dispatched for 2 msecs





LPAR #	Entitlement / # of VP
1	.2 / 1
2	.2 / 1
3	.1 / 1
4	.5 / 1

Virtual Processors and Processing Unit Relationship

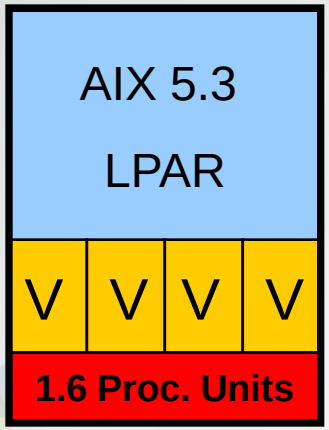
Virtual Processors Assigned to LPAR	Range Of Processing Units that the LPAR can utilize
1	0.1 - 1
2	0.2 - 2
3	0.3 - 3
4	0.4 - 4
…	…10x range

Example: An LPAR has 2 virtual processors. This means that it's minimum must be 0.2 or higher (0.1 per virtual processor). The max proc. units that it can utilize is 2.0.

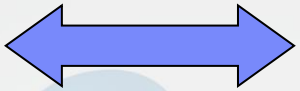
If we want this LPAR to use more than 2.0 physical CPUs worth of cycles, we need to dynamically add more virtual processors, perhaps 2 more. This would make its new minimum 0.4 and it max utilization 4.0.

The "desired" number of virtual processors establishes the maximum number of processing units that an LPAR can access.

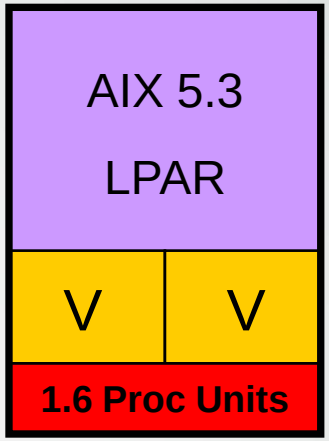
Virtual Processors and Processing Unit Relationship



Different number of virtual processors



Same amount of processing units



Each virtual processor will receive 0.4 processing units

Max processing units accessible to handle peak workload is 4

Individual processes/threads may run slower

Workloads with a lot of processes/threads may run faster

Each virtual processor will receive 0.8 processing units

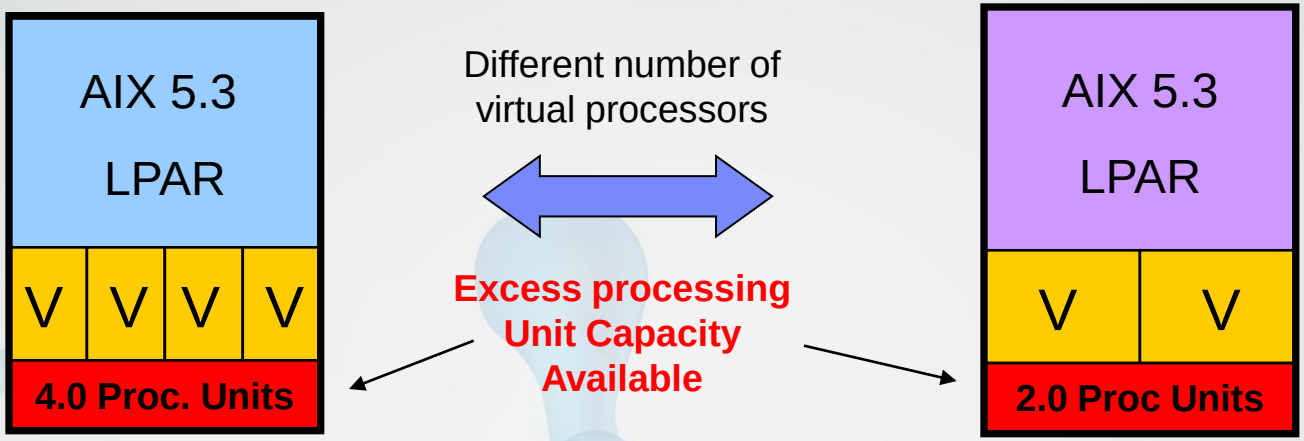
Max processing unit accessible to handle peak workload is 2

Individual processes/threads may run faster

Workloads with a lot of processes/threads may run slower

Consider the peak processing requirements when setting the desired number of virtual processors. In addition, the quantity of virtual processors can be adjusted to match the number of processes/threads present in the workload.

Virtual Processors and Processing Unit Relationship



Each virtual processor will receive 1.0 processing units

Max processing units accessible to handle peak workload is 4

Virtual processors receive 1 full CPU worth of processing units.

Workloads with a lot of processes/threads may run faster due to larger number of virtual processors.

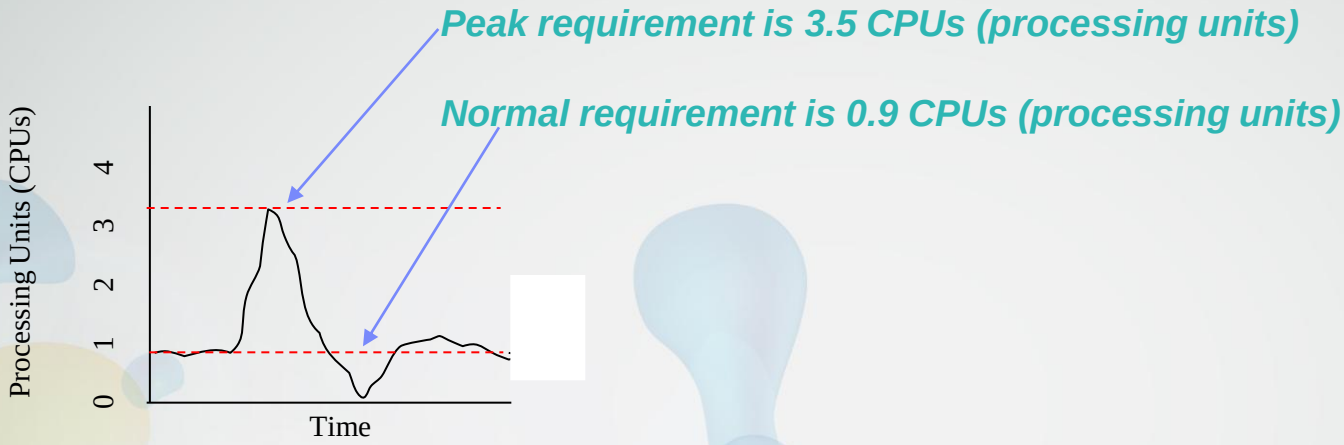
Each virtual processor will receive 1.0 processing units

Max processing unit accessible to handle peak workload is 2

Virtual processors receive 1 full CPU worth of processing units.

Workloads with a lot of processes/threads may run slower due to lower number of virtual processors.

Sizing Processing Units and Virtual Processors



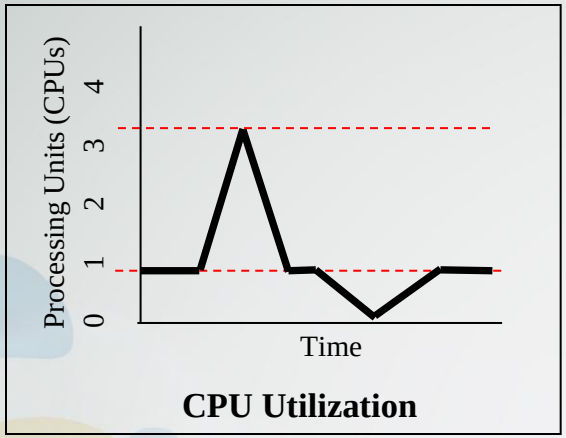
Processing Units Sizing:

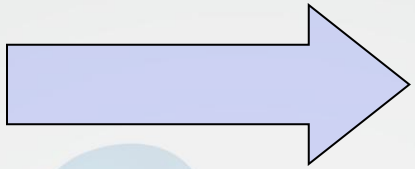
Need to size <u>desired</u> processing units to address non-peak, normal workload.
Desired = 0.9 (set to match 0.9 processing units, *normal requirement*)
Minimum = Starting point might be 0.5, or approx. ½ of Desired.
Maximum = 4+
Set as uncapped

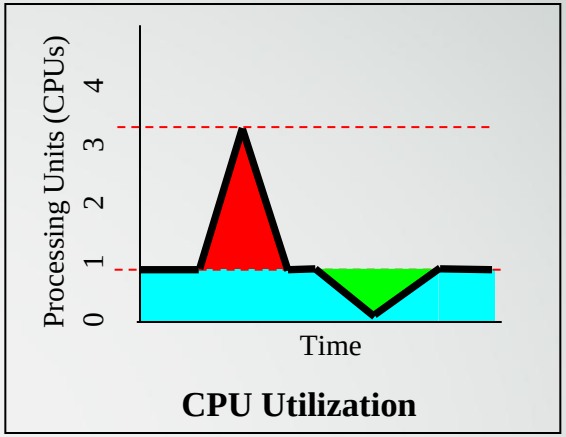
Virtual Processor Sizing:

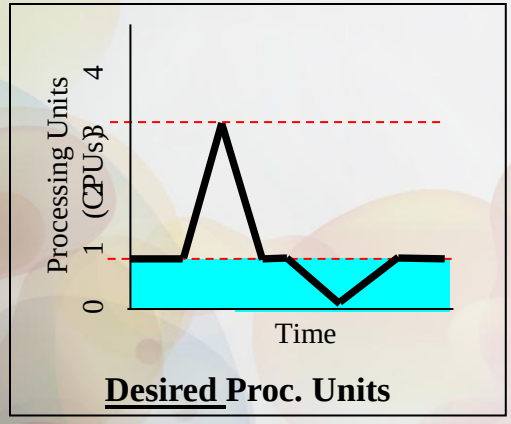
We need to size <u>desired</u> number of virtual processors to be able to handle peak load.
Desired = 4 (round 3.5, *peak requirement* up to next whole number)
Minimum = minimum as required by the capacity (in this case, 1 VP).
Maximum = 4+

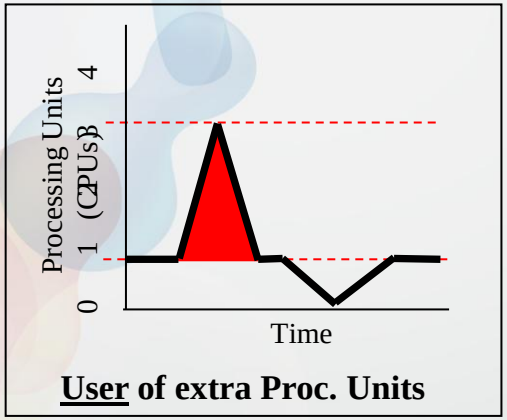
Operating within the Shared Processor Pool

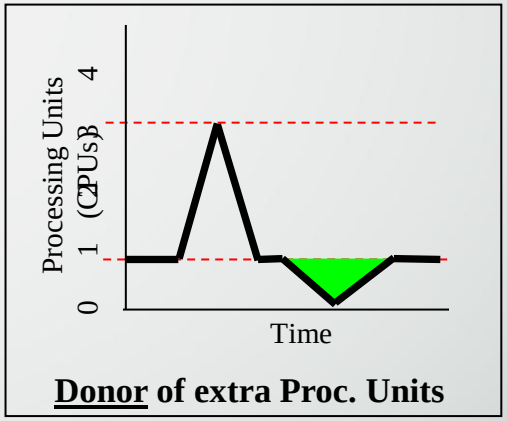










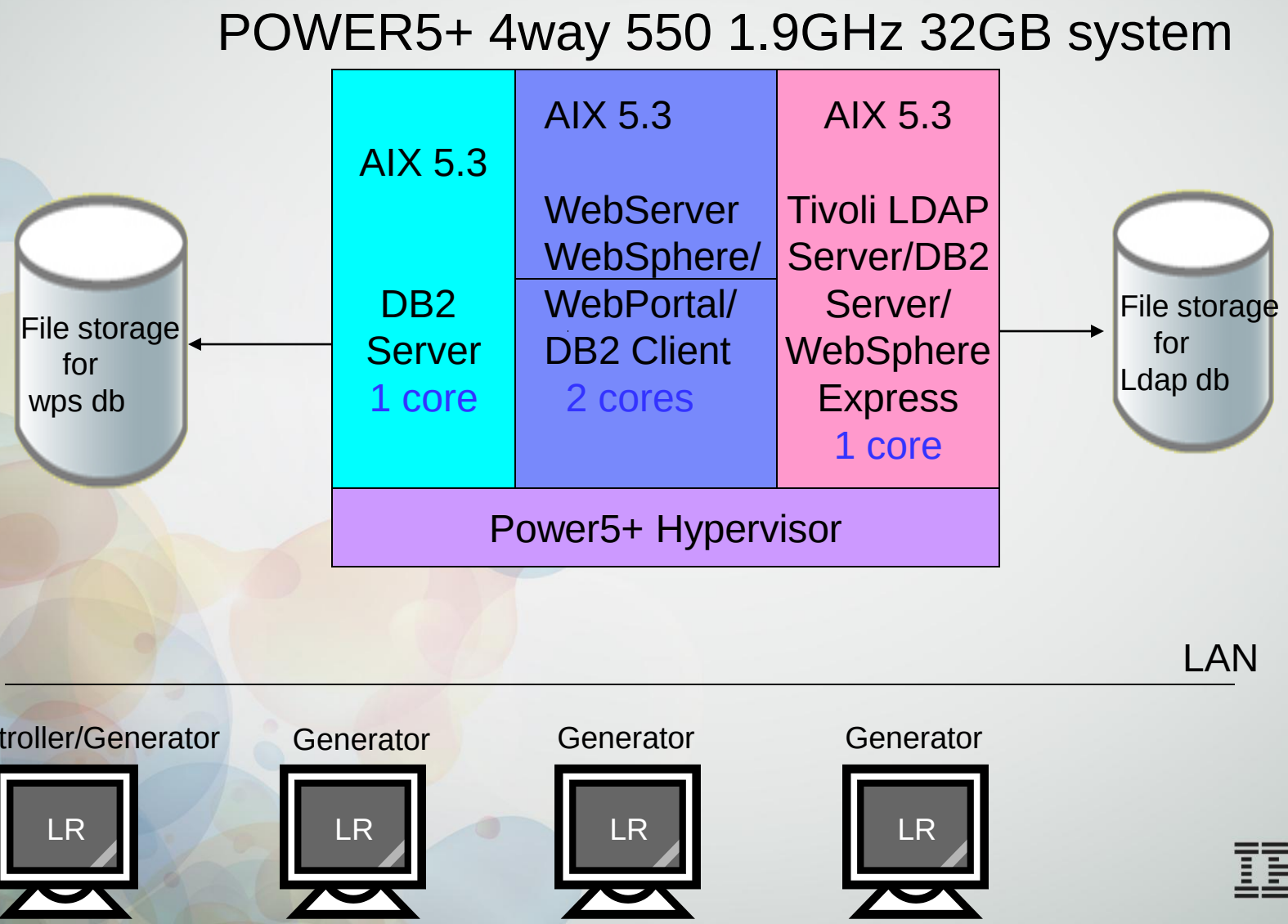


The goal is to match Users and Donors so that the *planned* overall shared processing pool CPU utilization does not exceed 100%.

A virtualization Study – Websphere Portal Server

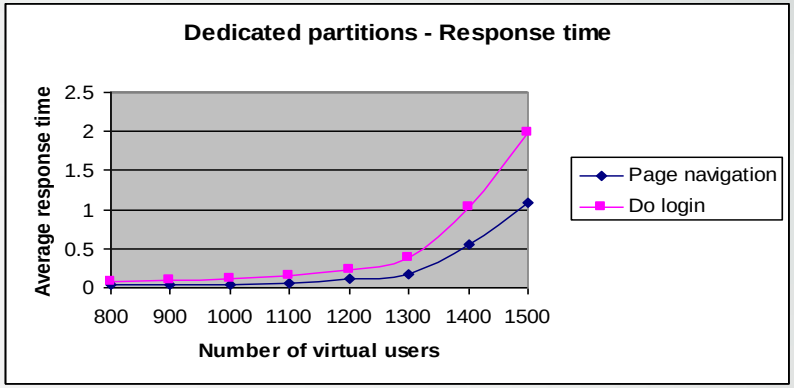
- **Websphere Portal provide a single access point to Web provided content and applications, and can be personalized to individual user preferences.**
- **A typical Websphere Portal implementation consists of at least three components**
  - Websphere Portal Server
    - Can be split HTTP server into another partition, but we combine them for this exercise
  - A database instance
    - DB2 for these tests
  - An identity management (LDAP) instance
    - IBM Directory Server for these tests
- **Because a Websphere Portal implementation combines multiple components, it is an ideal candidate for virtualization**
  - We can take a single system and partition it to provide a complete portal environment

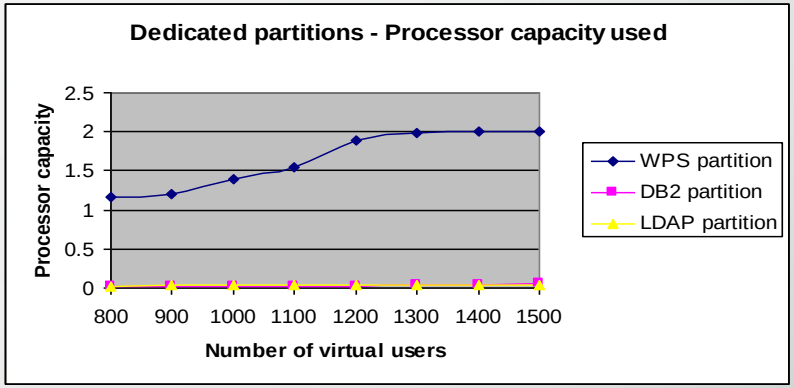
Websphere Portal Server – Dedicated partitions



First Test – Dedicated Partitions

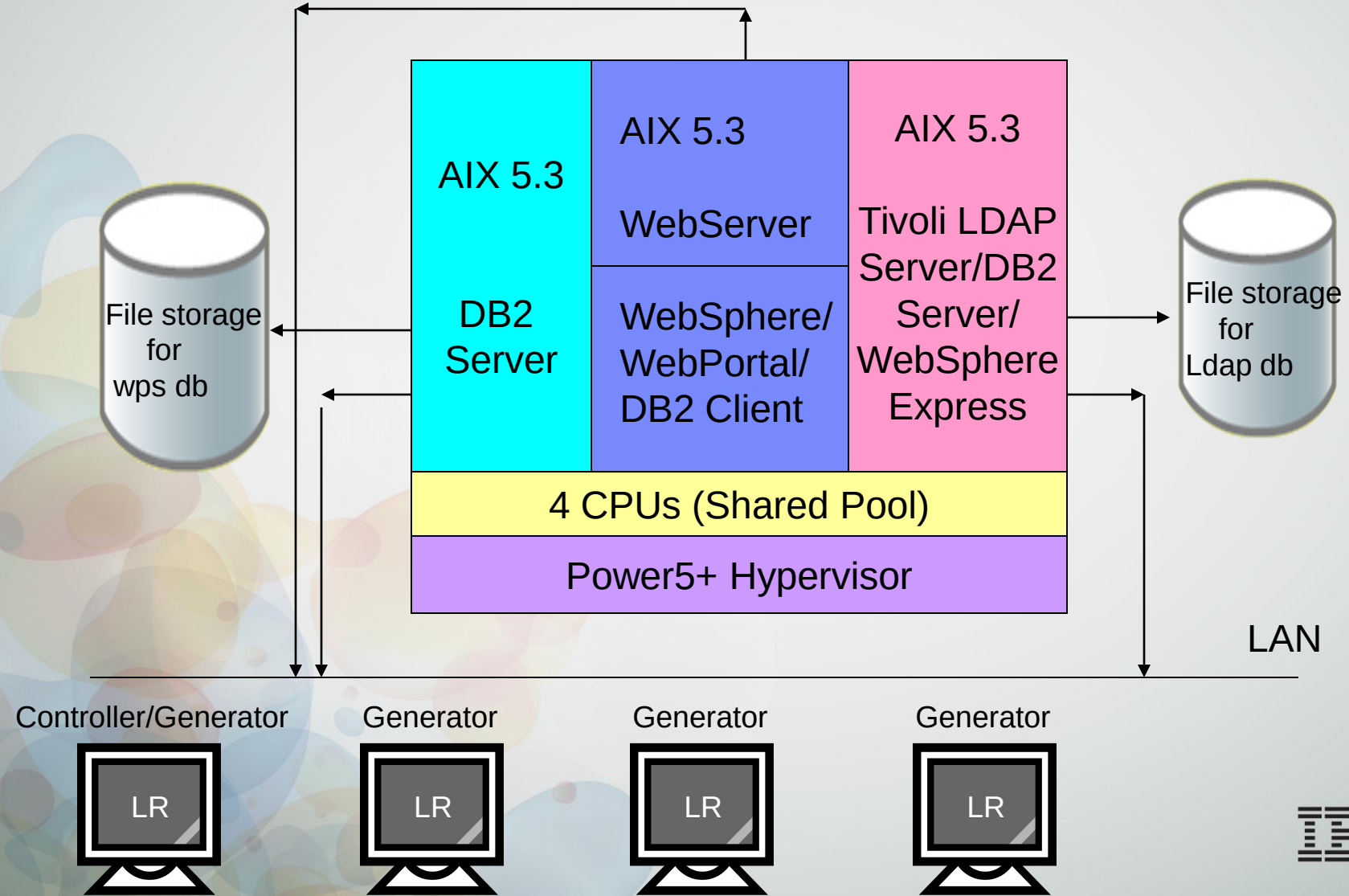
- **3 dedicated partitions**
  - Portal – 2 processors, 16GB of memory
  - DB2 – 1 processor, 8GB of memory
  - LDAP – 1 processor, 8GB of memory
  - Interpartition communications via physical Ethernet adapters
- **The peak number of vusers supported in this test is 1500**
  - Peak is maximum throughput with acceptable response time
- **At peak virtual users, the Portal partition is essentially 100% CPU busy.**
- **At peak, the DB2 and LDAP partitions are 5% and 3% CPU busy respectively**
  - Most of the CPU capacity in these two partitions is idle





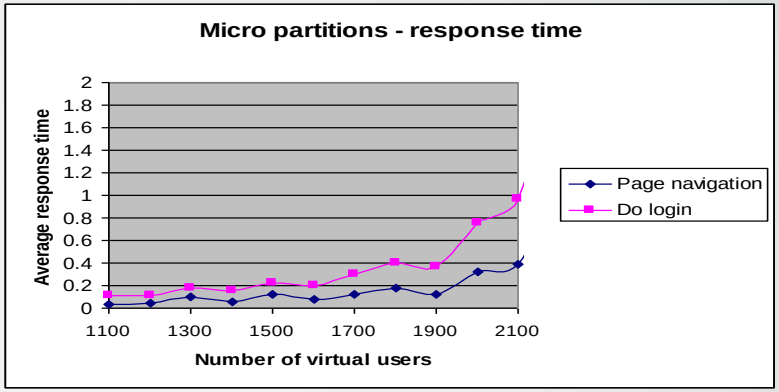
Websphere Portal Server - Micropartitions

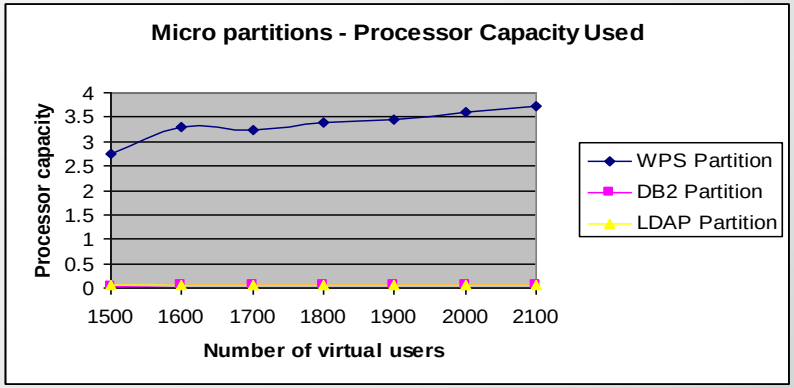
POWER5+ 4way 550 1.9GHz 32GB system



Micro Partitions – Response times and user levels

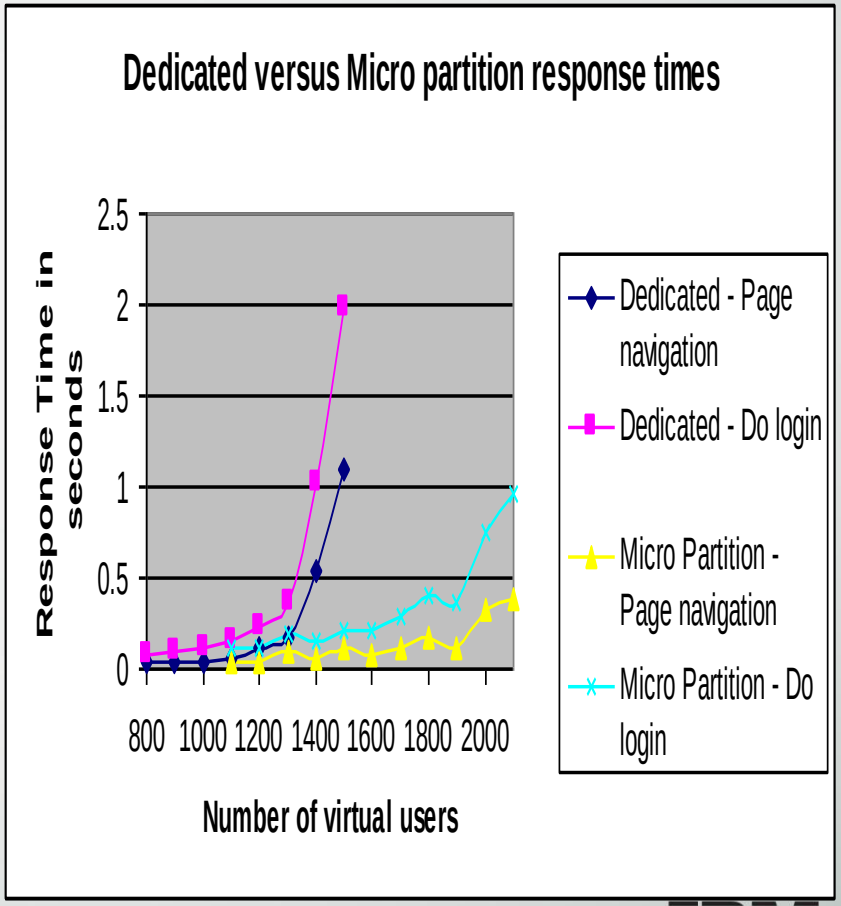
- **3 Micro partitions**
  - Portal – 3.0 processors entitlement, 4 vcpu's, uncapped
  - DB2 – 0.5 processor entitlement, 1 vcpu, uncapped
  - LDAP – 0.5 processor entitlement, 1 vcpu, uncapped
  - Same memory/communications as dedicated partitions case
- **The peak number of vusers supported in this test is 2100**
- **At peak virtual users, the Portal partition is consuming 3.68 processors CPU capacity on average**
- **At peak, the DB2 and LDAP partitions consume 0.053 and 0.063 processors respectively on the average**
- **Workload peaks in some cases consume the full CPU capacity of the system**





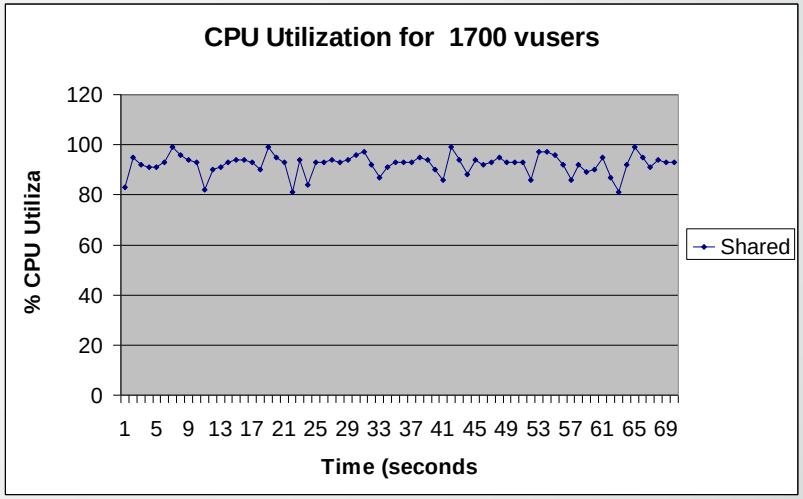
Comparing Response Times of dedicated processor partitions and shared processor partitions

- **Because the Micro partition test can supply more processing power to the CPU-intensive portal partition, it is difficult to do a direct comparison**
- **However, the response times of the Micro partitioned test are relatively equivalent or better than those in the dedicated processor partitions**



Micro Partitioning Best Practices

- **The CPU time used by a partition is spread over virtual CPU's**
  - The operating system running in the partition dispatches work to virtual CPUs
  - For performance, it never makes sense to have more virtual CPUs in a partition than there are physical CPUs in the shared pool
  - AIX will reduce the number of virtual CPUs it is dispatching work toward at lower utilization points (processor folding)
  - Even with processor folding, reducing the number of virtual CPUs to match the partition entitlement can help wring the most performance from the server
- **Capped or uncapped?**
  - Most workloads have higher "peak" utilizations than average utilizations (see graph at right)
  - Using uncapped partitions allows the Hypervisor to use available processor resources to address short spikes in partition utilization



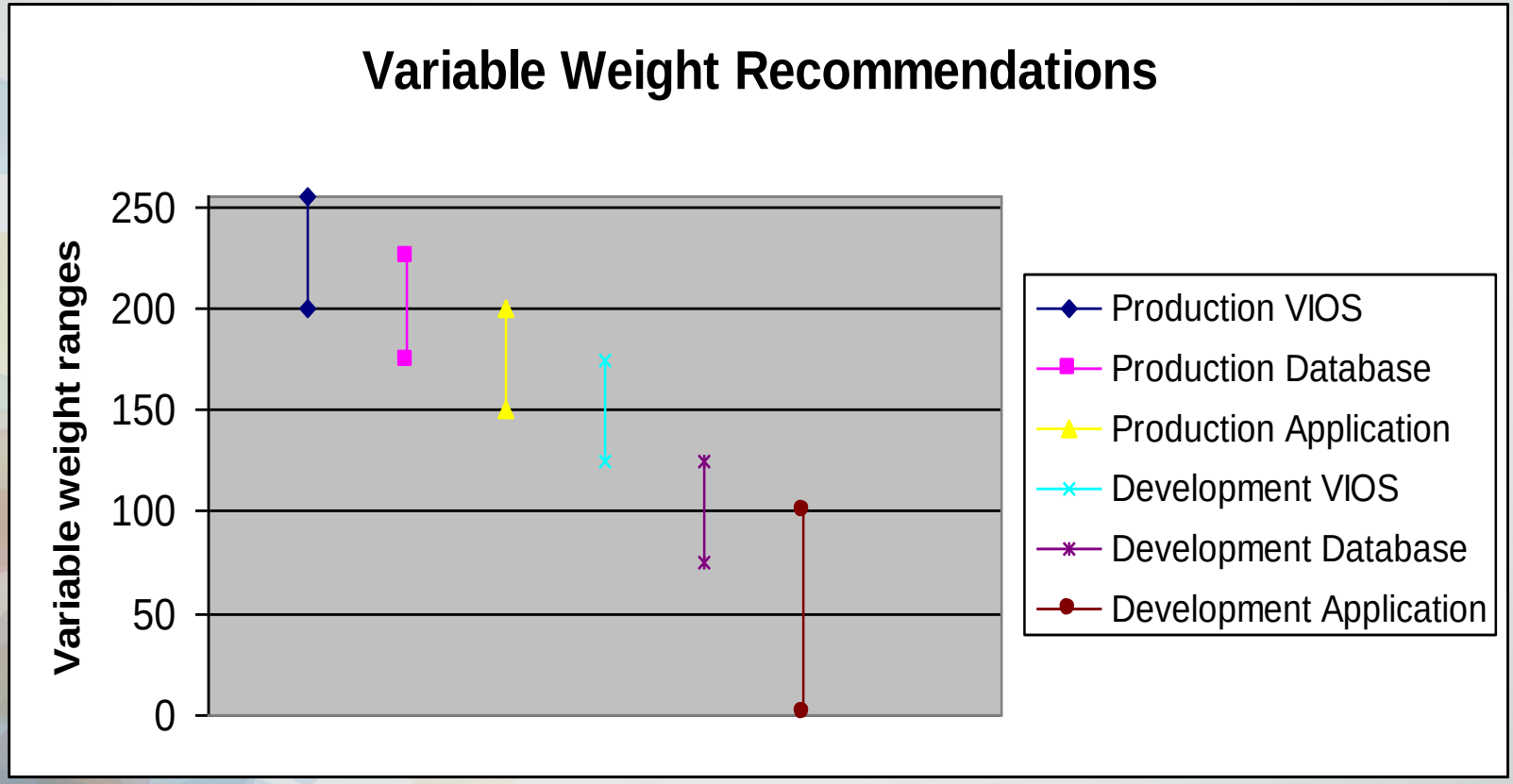
Virtual Processors - Folding

- **Dynamically adjusting active Virtual Processors**
  - System consolidates loads onto a minimal number of VPs
    - Scheduler computes utilization of VPs every second
      - If VPs needed to host physical utilization is less than the current active VP count, a VP is put to sleep
      - If VPs needed are greater than the current active VPs, more are enabled
  - On by default in AIX 5.3 ML3 and later
    - `vpm_xvcpus` tunable
- **Increases processor utilization and affinity**
  - Inactive VPs don't get dispatched and waste physical CPU cycles
  - Fewer VPs can be more accurately dispatched to physical resources by Hypervisor
- **When to adjust**
  - Burst/Batch workloads with short response-time requirements may need sub-second dispatch latency
    - Disable or manually tune the number of VPs
      - `# schedo -o vpm_xvcpus=[-1 | N]`
      - Where `N` specifies the number of VPs to enable in addition to the number of VPs needed to consume physical CPU utilization

Virtual Processors - Sizing

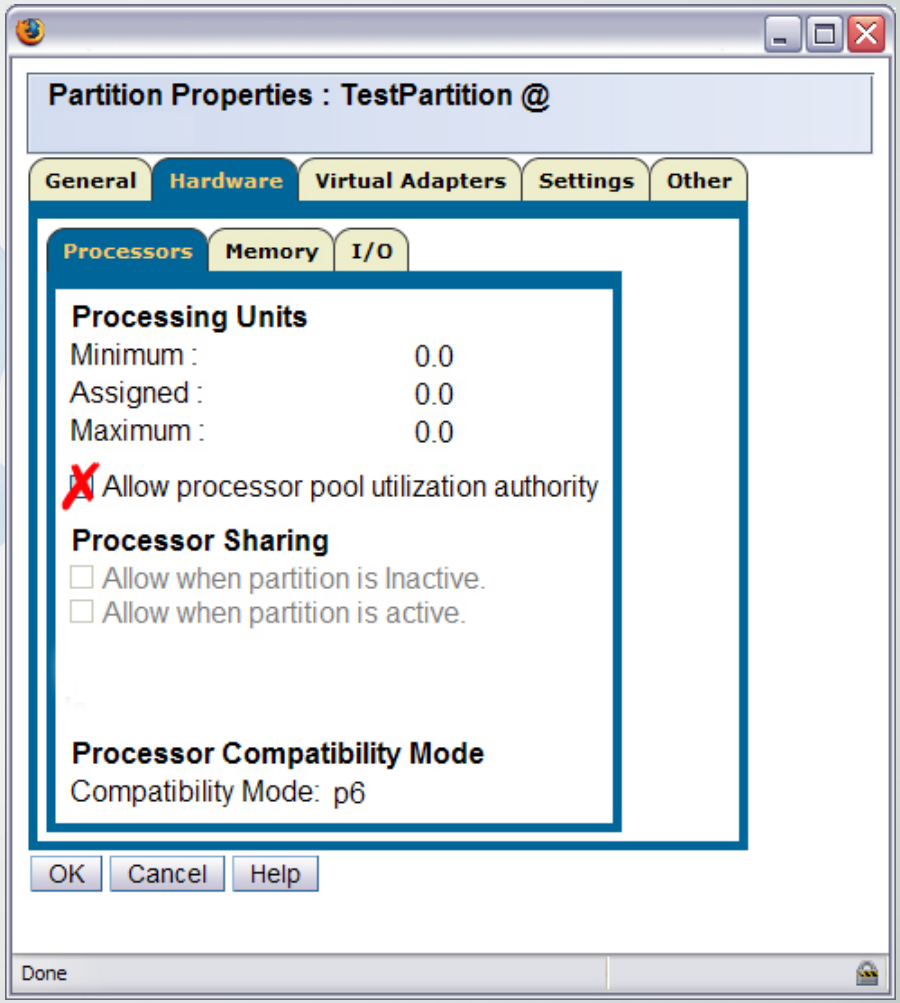
- **Capped**
  - Start with minimum number of virtual processors
  - Monitor application behavior, adjust as needed
- **Uncapped**
  - Unless you know the workloads peaks very well, don't make the aggregate of partitions VP > 1.5 the total number of physical processors in the shared pool
  - Good: workloads from partitions sharing pool do not peak at same times
  - Bad: workloads from partitions sharing pool peak simultaneously
    - Partitions that have dependency, such as DB/App Server tend to peak simultaneously
  - Uncapped Weights
    - Hypervisor-based weighting of available cycles to uncapped partitions
    - Set within HMC
    - Use default of 128 for new partitions
    - Adjust carefully based on consistent over utilization and application priority needs
    - See next chart for suggestions
- **Try to minimize the number of VPs for anticipated loads**
  - Too many VPs can cause high context switching, cache misses, lock contention
    - Try disabling SMT to see if lock contention can be reduced
    - Reduce the number of VPs
  - Too few VPs on uncapped partitions cannot use all of available resources
  - VP folding can help, but it isn't perfect

Variable Weight Example



Enable Monitoring of the shared pool usage

- **Make sure at least one partition on the CEC can do pool monitoring!**
  - lparstat not displaying the available pool processor “app” value
  - Required for lparstat to see free pool resources, but topas gets around this because it can collect data from remote agents and calculate itself
  - nmon recording will only see app value if this is enabled



Topas LPAR & CEC View

- **topas –C**
  - Upper section displays aggregated CEC information
  - Lower section displays shared/dedicated data – closely mimics lparstat

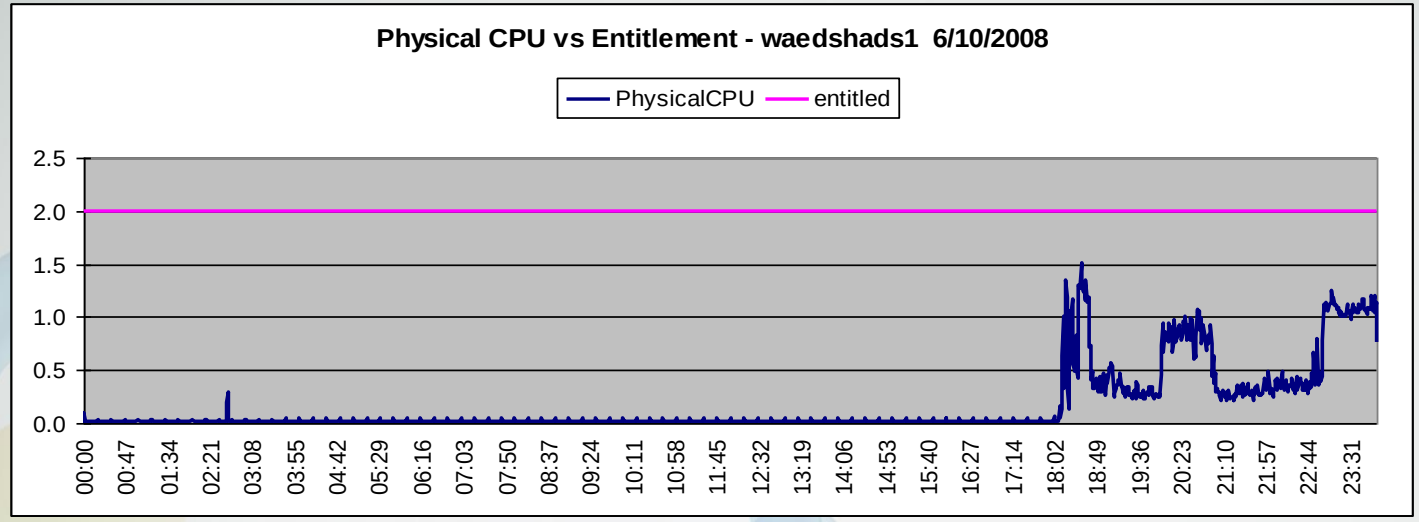
```
Topas CEC Monitor            Interval:  10             Thu Jul 28 17:04:57 2006
Partition Info    Memory (GB)         Processor
Monitored  :  6   Monitored  :24.6   Monitored  :1.2   Shr Physical Busy:  0.30
UnMonitored:  -   UnMonitored:   -   UnMonitored:  -   Ded Physical Busy:  2.40
Shared     :  3   Available  :24.6   Available  :  -
Dedicated  :  3   UnAllocated:   0   UnAllocated:  -   Hypervisor
Capped     :  1   Consumed   : 2.7   Shared     :1.5   Virt. Context Switch: 632
Uncapped   :  1                       Dedicated  :  5   Phantom Interrupts  :   7
                                      Pool Size  :  3
                                      Avail Pool :2.7

Host          OS  M Mem InU Lp  Us Sy Wa Id  PhysB  Ent  %EntC Vcsw PhI
-------------------------------------shared-------------------------------
ptoolsl3      A53 c 4.1 0.4  2  14  1  0 84   0.08  0.50  15.0  208   0
ptoolsl2      A53 C 4.1 0.4  4  20 13  5 62   0.18  0.50  36.5  219   5
ptoolsl5      A53 U 4.1 0.4  4   5  0  0 95   0.04  0.50   7.6  205   2

-----------------------------------dedicated----------------------------
ptoolsl1      A53 S 4.1 0.5  4  20 10  0 70   0.30
ptoolsl4      A53   4.1 0.5  2 100  0  0  0   2.00
ptoolsl6      A52   4.1 0.5  1   5  5 12 88   0.10
```

APP = Pool Size - Shared Physical Busy

- **TL-08 topas supports**
  - View of each shared pool, if AIX partitions
  - CPU cycle donations made by dedicated partitions

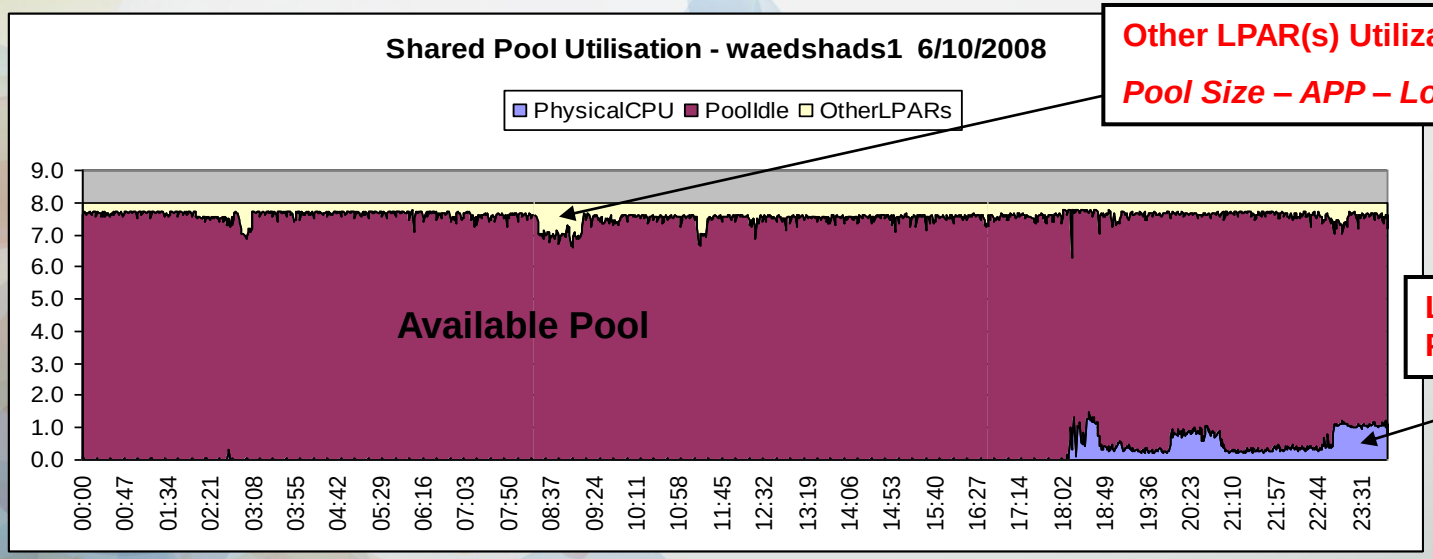
- **Dashes represent data not available at OS level**
  - Can be provided via command-line
  - Topas can be configured to collect via ssh to HMC
  - Topasrec or nmon can collect data

LPAR View - nmon Analyser



Analyser can only see:

- **Pool Size**
- **APP**
- **Local Utilization**

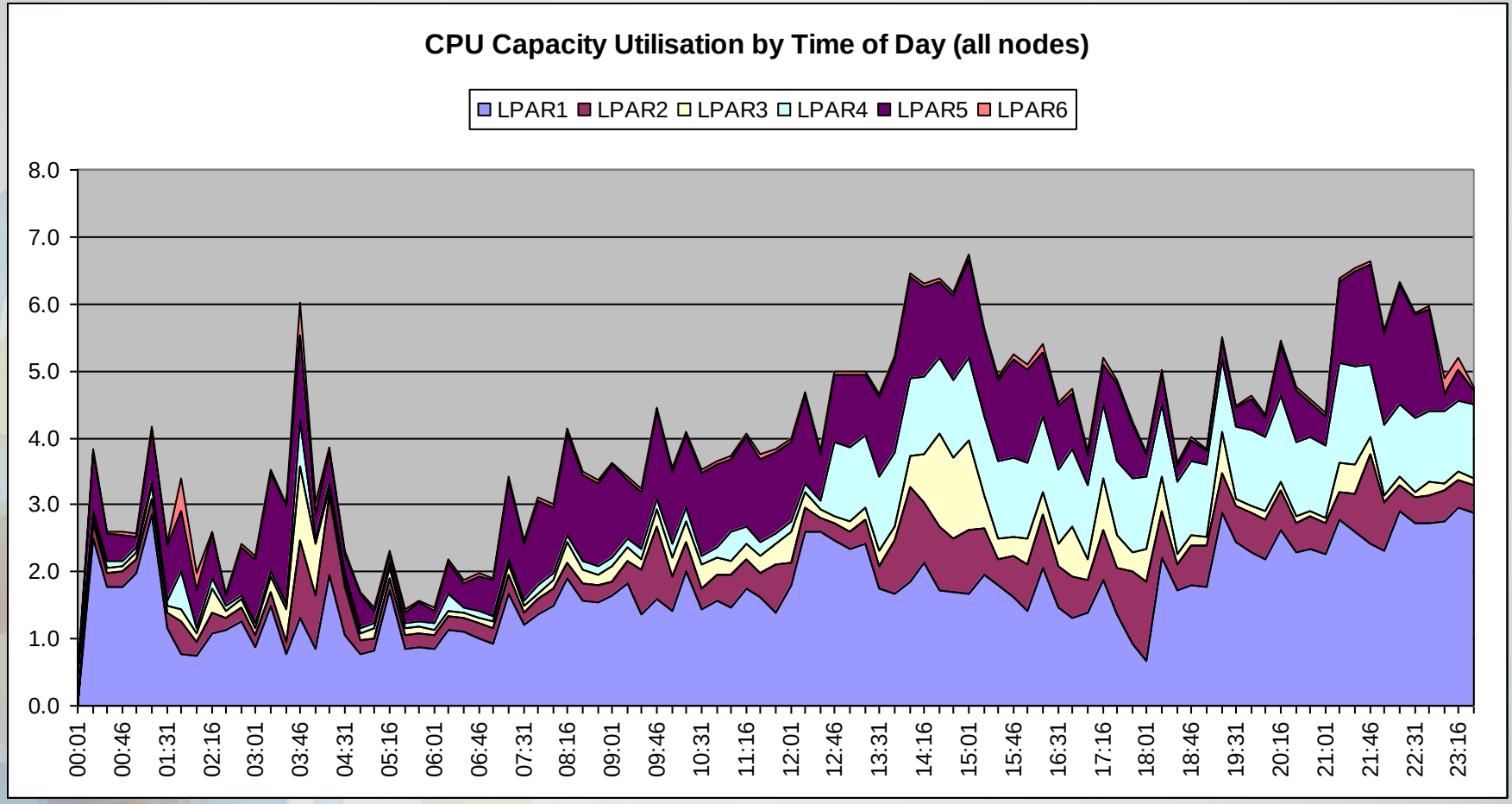


Other LPAR(s) Utilization of Pool =

Pool Size – APP – Local Utilization

Local LPAR Physical Used

LPAR(s) View - nmon Consolidator



View depends on partitions selected

Consolidator also provides graphs of estimated VP usage

Deployment Choices

- No information on application behavior and utilization of resources?
  - Size for peak processing requirements
    - Minimize risk, but excess capacity is unused
    - Collect performance data to determine suitability for reducing resources
  - Use shared processors
    - Allocate entitlement liberally, until resource behavior known
- Mixed applications, variable behavior
  - Size to known peaks
    - Enough application, benchmark or local performance information to model expected behavior
    - Size each to micro-partition, allocate extra shared pool and memory resources
  - Collect performance data to validate model, free shared pool and memory allocation to optimize
- Well-defined applications
  - Detailed application knowledge allowing for partitions to be individually over-committed (don't conflict for shared resources)
  - Ideal usage of resources

Links, References and Sources

Links

- AIX Wiki
  - http://www.ibm.com/developerworks/wikis/display/WikiPtype/Home
- AIX Performance Tools (nmon, nmon analyser/consolidator, etc)
  - http://www.ibm.com/developerworks/wikis/display/WikiPtype/nmon
- AIX DeveloperWorks
  - http://www.ibm.com/developerworks/aix

References

- AIX 6.1 Performance management and tuning http://publib.boulder.ibm.com/infocenter/aix/v6r1/index.jsp?topic=/com.ibm.aix.doc/doc/base/performance.htm
- Huang, W., Cheng, L., Accapadi, M., Keung, N., (2005, July). CPU monitoring and tuning. Retrieved Sept 2005, from http://www-128.ibm.com/developerworks/eserver/articles/aix5_cpu/
- Frankek, F., (2004). Memory as a Programming Concept in C and C++. Cambridge University Press ISBN 052181720X. Retrieved Oct 2005 from http://www.books24x7.com.
- Braden, B., (2005-2008). Disk Sizing, Data Layout and Tuning Presentation.
- Barker, R., (2005). VIO Server Guidelines Presentation.
- Smolders, L., (2005-2008). Performance Tools Presentations.
- Mathis, H.M., Mericas, A.E, McCalpin J.D., Eickemeyer R. J., and Kunkel S.R.. Characterization of simultaneous multithreading (SMT) efficiency in POWER5. IBM Journal of Research and Development Volume 49, Number 4/5 2005. Retrieved November 2005 from http://www.research.ibm.com/journal/rd/494/mathis.html

IBM

Technical Forum & Executive Briefing

17 al 21 Octubre 2011

Imagine **PODER** Imagine **CAPACIDAD**

Gracias!

cmaciel@us.ibm.com