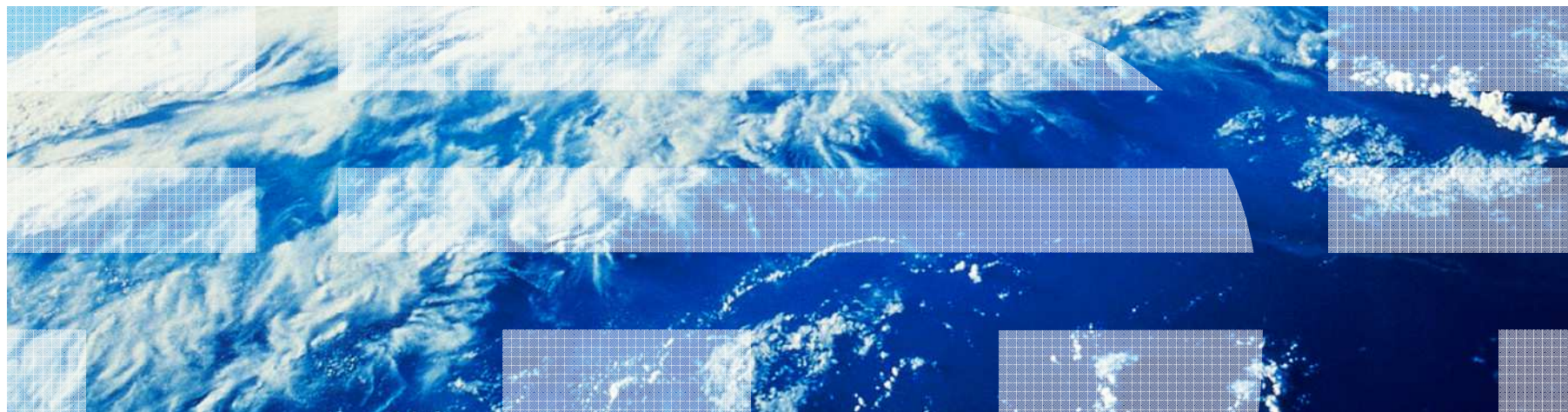


# z/VM HiperDispatch Deep Dive

Revision 2014-04-14.1, BKW

Brian K. Wade, Ph.D.  
bkw@us.ibm.com



## Agenda

- Objectives of the z/VM HiperDispatch enhancement
- A little about System z hardware and the PR/SM hypervisor
  - Machine structure
  - Behavior and features available in the hypervisor
- Key features of z/VM HiperDispatch
  - Use of vertical mode partitions
  - Running as widely as available power suggests
  - Reducing MP level when it appears z/VM overhead is a problem
  - Dispatching guests in a manner aware of physical and virtual topologies
  - Knobs you can twist or set
- Planning for z/VM HiperDispatch
- Workloads
  - Those that will benefit
  - Those that won't
- CP Monitor and z/VM Performance Toolkit
- Summary

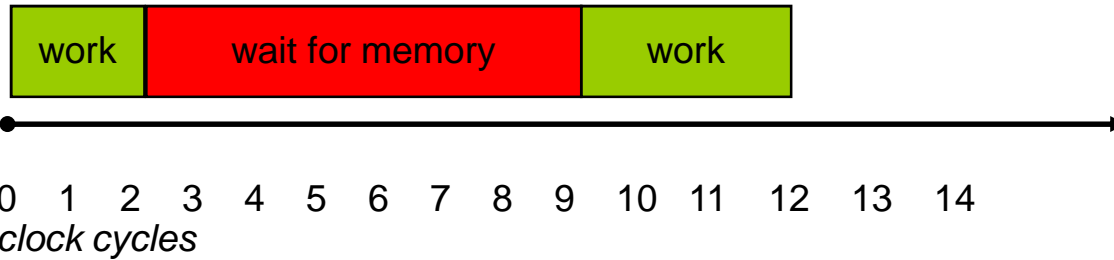
# Objectives and Strategies

## Objectives and Strategies of z/VM HiperDispatch

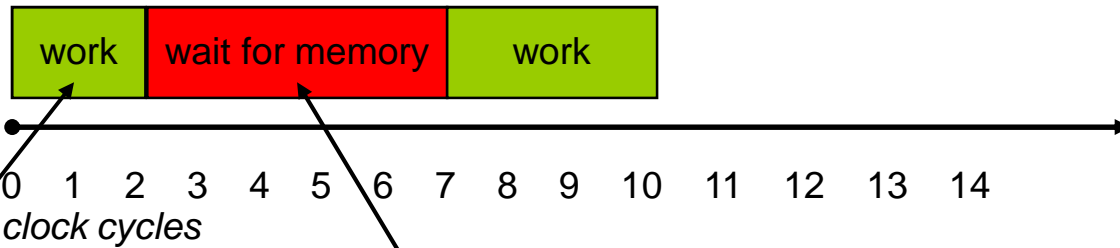
- Improve performance of your workloads, by ...
  - Reducing CPU time needed per unit of work done, by ...
  - Reducing the time needed for each instruction to run, by ...
  - Reducing the time the CPU waits for memory contents to be brought to it.
- Improve performance of your workloads, by...
  - Sensing situations where z/VM Control Program overhead is a problem, and...
  - Changing the LPU configuration to try to reduce the overhead.
- Strategies:
  - Exploit PR/SM hypervisor features meant to help instruction speed
  - Be smarter about what the right MP-level is for the partition at the moment
  - Be smarter about the dispatching of guest virtual CPUs

# What It Means to Reduce CPU Wait Time

A R3, MEMWORD



A R3, MEMWORD

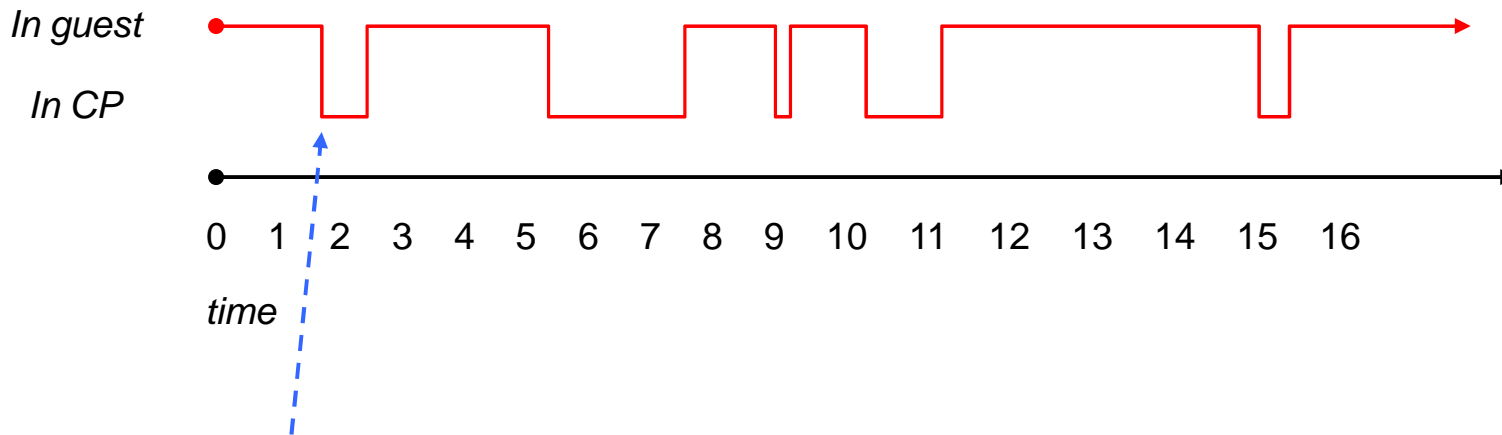


*Instruction complexity CPI aka Infinite CPI*

*Cache miss CPI aka Finite CPI*

# What It Means to Reduce z/VM Overhead

CPU Consumption Timeline of a Virtual CPU



**Some reasons guests go into CP:**

- Issue a Diagnose
- Perform I/O
- Issue some other priv op
- Incur a page fault

$$T/V \text{ ratio} = \frac{(\text{CP time} + \text{guest time})}{\text{guest time}}$$

**Things CP often does "down there":**

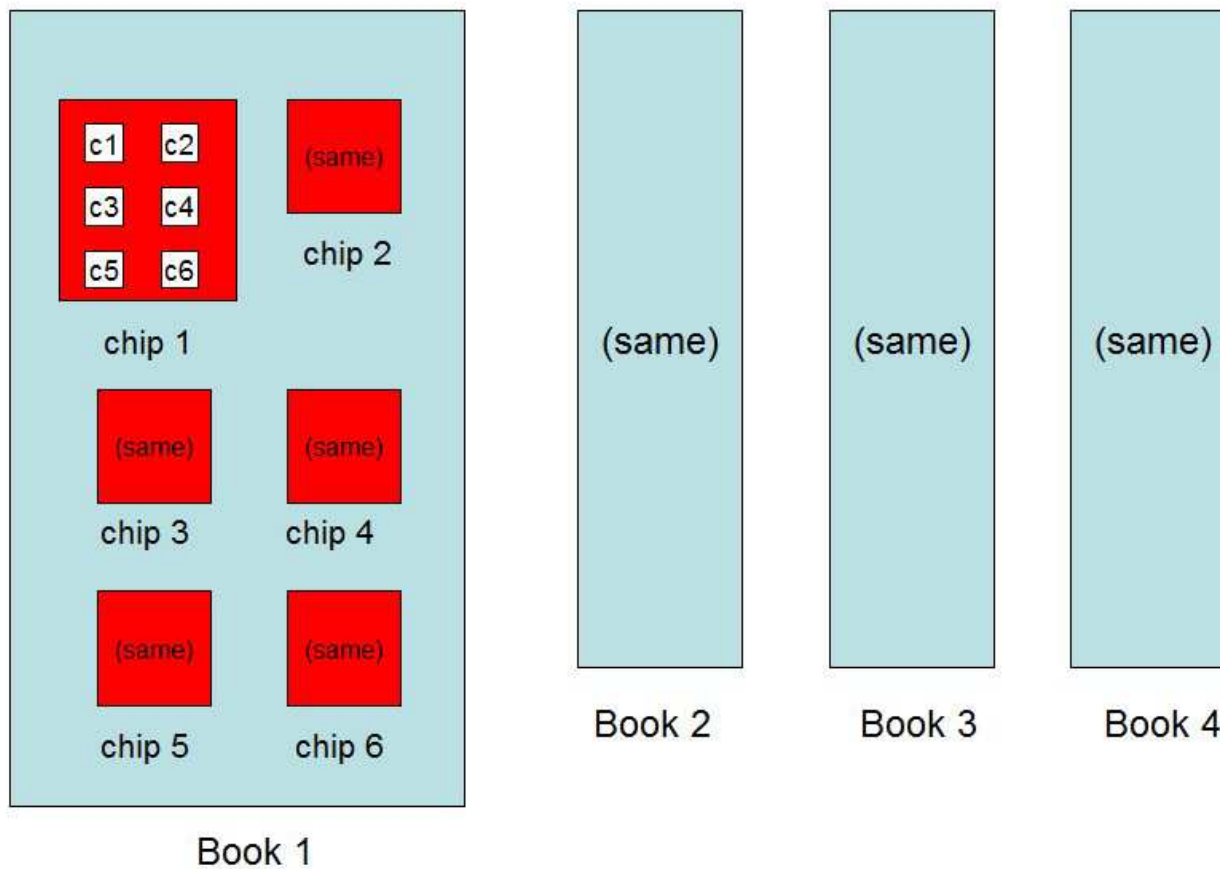
- Acquire a lock, for serialization
- Do some processing
- Release the lock
- Eventually, run the guest again

← Time spent spinning on locks is wasted time.  
 We can reduce it by reducing the partition's MP level.

# A Few Things About System z and PR/SM

# IBM System z: Cores, Chips, and Books

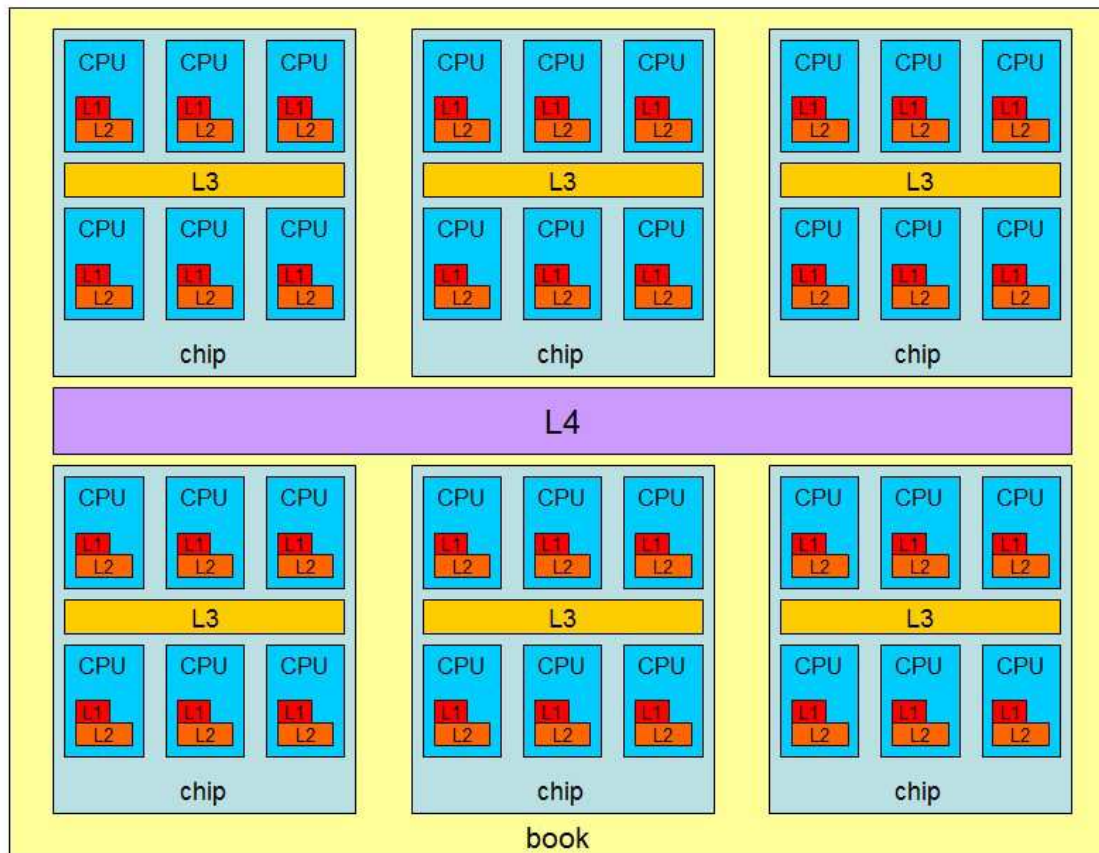
## IBM System z CPU–Chip–Book Relationship





# IBM System z: Layered Cache Structure

## IBM System z Cache Layering



### Cache mantra:

- Closer, smaller, faster.
- Farther, larger, slower.
- Try to run a context in the same place over and over.
- Try to run related contexts near to one another.
- Try to run unrelated contexts apart from one another.

## IBM System z: Partition Entitlement vs. Logical CPU Count

Suppose we have 12 physical IFLs: 2 dedicated, 10 shared.

| Partition                      | Weight | Weight Sum | Weight Fraction | Physical Capacity | Entitlement Calculation | Entitlement | Maximum Achievable Utilization |
|--------------------------------|--------|------------|-----------------|-------------------|-------------------------|-------------|--------------------------------|
| FRED,<br>a logical 10-<br>way  | 63     | 100        | 63/100          | 1000%             | 1000% x<br>(63/100)     | 630%        | 1000%                          |
| BARNEY,<br>a logical 8-<br>way | 37     | 100        | 37/100          | 1000%             | 1000% x<br>(37/100)     | 370%        | 800%                           |

FRED can always run up to 630% busy. That's what *entitlement* means.

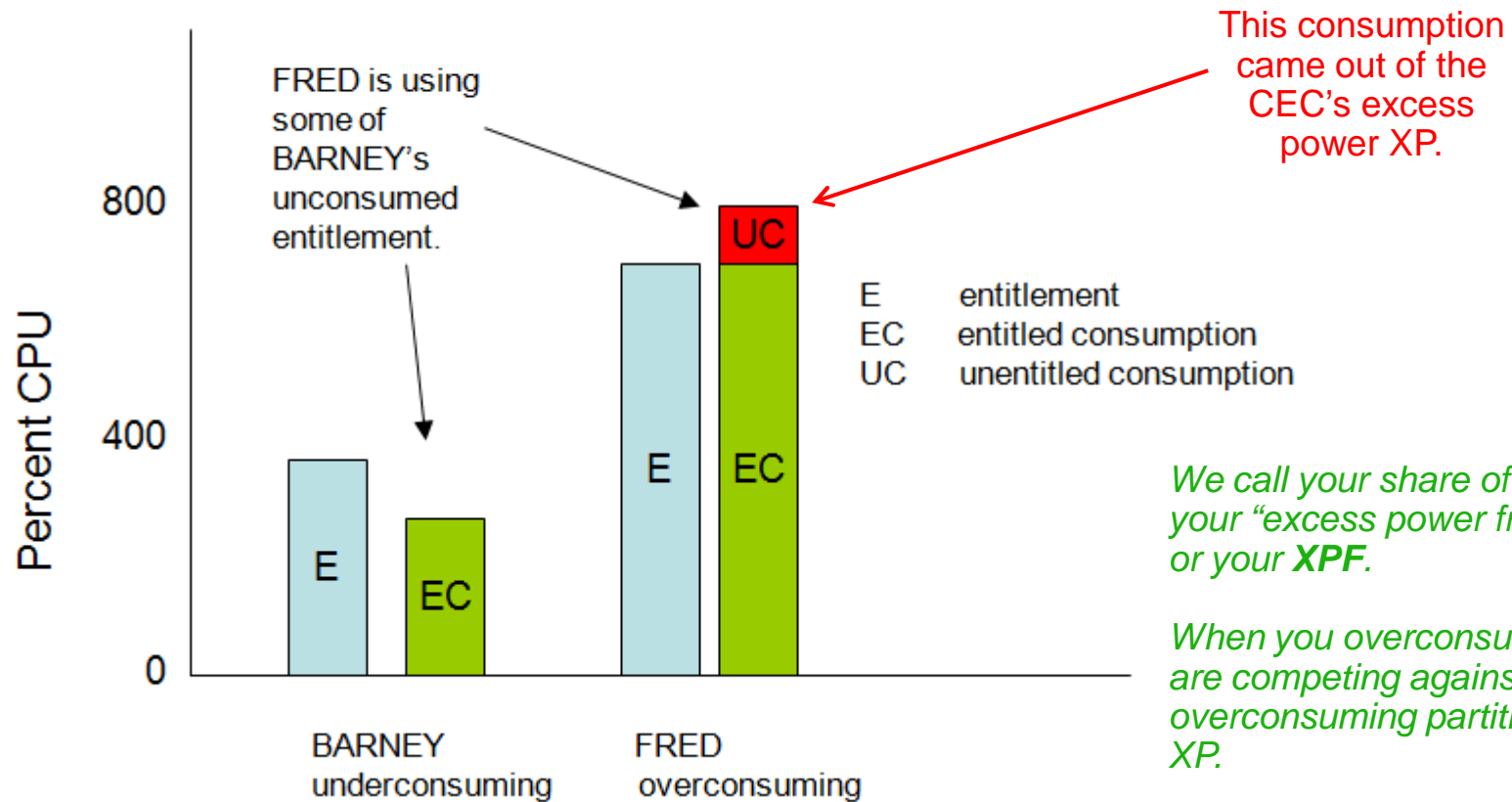
But for FRED to run *beyond* 630% busy, BARNEY has to leave some of its entitlement *unconsumed*.

Keep this in mind: (CEC's excess power XP) = (total power TP) - (consumed entitled power EP).

Excess power XP will become very important later.

# IBM System z: Entitlement and Consumption

## Entitlement and Consumption



## IBM System z: A Little More About XP and XPF

Suppose there is 180% left after all entitled consumptions are satisfied.  $XP=180\%$ .  
 Suppose P1, P2, and P3 (me), all equal weights, are competing for it.  
 Their first-pass weight fractions of XP are therefore each 60%.

### Case 1:

- P1 is overconsuming 15%
- P2 is overconsuming 25%

P3 can have  $(180-(15+25)) = 140\%$   
 if it wants it.  $XPF=140$

### Case 2:

- P1 is overconsuming 90%
- P2 is overconsuming 90%

P3 can have 60% if it wants it.  $XPF=60$

### Case 3:

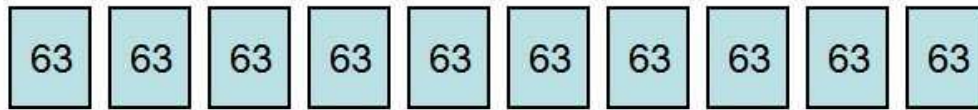
- P1 is overconsuming 135%
- P2 is overconsuming 10%

Round 1:  $P1+=60$ ,  $P2+=10$ ,  $P3+=60$ ,  $s=130$ ,  $r=50$   
 Round 2:  $P1+=25$ ,  $P3+=25$ ,  $s=50$ ,  $r=0$   
 P3 can have 85% if it wants it.  $XPF=85$

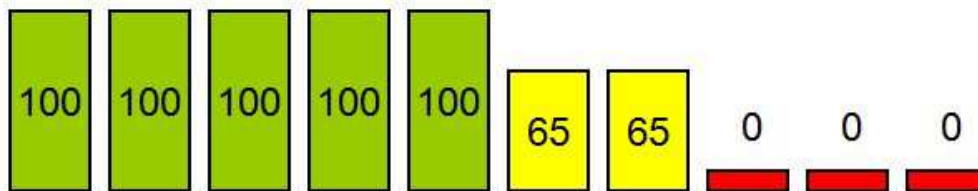
# IBM System z: Horizontal and Vertical Partitions

## Two Ways To Get 630% Entitlement

Horizontally: 10 each @ 63%



Vertically: 5 Vh @ 100%, 2 Vm @ 65%, 3 VI @ 0%

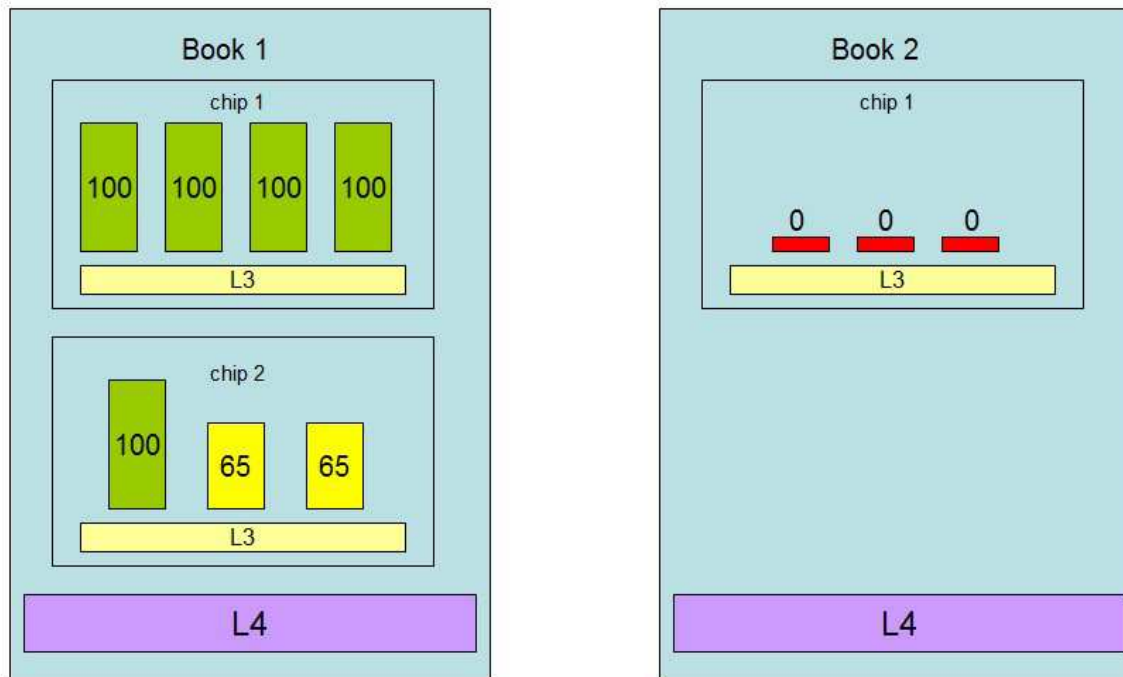


### In vertical partitions:

- Entitlement is distributed unequally among LPUs.
- The unentitled LPUs are useful only when other partitions are not using their entitlements.
- PR/SM tries very hard not to move Vh LPUs.
- PR/SM tries very hard to put the Vh LPUs close to one another.
- Partition consumes its XPF on its Vm and VI LPUs.

# IBM System z: The Partition Knows Its Placement

## Partition Topology



### In vertical partitions:

- Sense your placement
- Run work smartly in light of your placement
- Sense unentitled power
- Use LPUs smartly in light of unentitled power

*Notice PR/SM has given this partition a “quiet place” to do its work, provided the partition runs its work on its Vh LPUs.*

# What z/VM HiperDispatch Does With All This

## z/VM HiperDispatch: Use of Vertical Mode

**indicate load**

```
AVGPROC-000% 24
XSTORE-000000/SEC MIGRATE-0000/SEC
MDC READS-000000/SEC WRITES-000000/SEC HIT RATIO-000%
PAGING-0/SEC
Q0-00000(00000)                                DORMANT-00000
Q1-00000(00000)                                E1-00000(00000)
Q2-00000(00000) EXPAN-000 E2-00000(00000)
Q3-00000(00000) EXPAN-000 E3-00000(00000)
```

```
PROC 0000-000% CP   VH   PROC 0001-000% CP   VH
PROC 0002-000% CP   VH   PROC 0003-000% CP   VH
PROC 0004-000% CP   VH   PROC 0005-000% CP   VH
PROC 0006-000% CP   VH   PROC 0007-000% CP   VH
PROC 0008-000% CP   VH   PROC 0009-000% CP   VH
PROC 000A-000% CP   VH   PROC 000B-000% CP   VH
PROC 000C-000% CP   VH   PROC 000D-000% CP   VH
PROC 000E-000% CP   VH   PROC 000F-000% CP   VH
PROC 0010-000% CP   VH   PROC 0011-000% CP   VH
PROC 0012-000% CP   VH   PROC 0013-000% CP   VH
PROC 0014-000% CP   VM   PROC 0015-000% CP   VM
PROC 0016-000% CP   VL   PROC 0017-000% CP   VL
```

```
LIMITED-00000
Ready; T=0.01/0.01 13:13:39
```

Here we see an assortment of LPU's:

- 20 Vh
- 2 Vm
- 2 Vl

If I recall correctly this was a 24-way with 2130% entitlement.

**Note:** these percent-busies are now *percent of a physical CPU*, not percent-not-deliberately-waiting as they used to be:  
 - Older releases: if the logical CPU never loaded a wait PSW, it showed 100% busy no matter what it was truly using.  
 - New release: these percentages are the *fraction of the capacity of a physical CPU* being used by the logical CPU.



## z/VM HiperDispatch: Awareness of Topology

### q proc topology

```

13:14:59 TOPOLOGY
13:14:59   NESTING LEVEL: 02  ID: 01
13:14:59     NESTING LEVEL: 01  ID: 01
13:14:59       PROCESSOR 00  PARKED    CP   VH  0000
13:14:59       PROCESSOR 01  PARKED    CP   VH  0001
13:14:59       PROCESSOR 12  PARKED    CP   VH  0018
13:14:59     NESTING LEVEL: 01  ID: 02
13:14:59       PROCESSOR 0E  MASTER     CP   VH  0014
13:14:59       PROCESSOR 0F  ALTERNATE  CP   VH  0015
13:14:59       PROCESSOR 10  PARKED    CP   VH  0016
13:14:59       PROCESSOR 11  PARKED    CP   VH  0017
13:14:59     NESTING LEVEL: 01  ID: 03
13:14:59       PROCESSOR 02  PARKED    CP   VH  0002
13:14:59       PROCESSOR 03  PARKED    CP   VH  0003
13:14:59       PROCESSOR 04  PARKED    CP   VH  0004
13:14:59     NESTING LEVEL: 01  ID: 04
13:14:59       PROCESSOR 05  PARKED    CP   VH  0005
13:14:59       PROCESSOR 06  PARKED    CP   VH  0006
13:14:59       PROCESSOR 07  PARKED    CP   VH  0007
13:14:59     NESTING LEVEL: 01  ID: 05
13:14:59       PROCESSOR 08  PARKED    CP   VH  0008
13:14:59       PROCESSOR 09  PARKED    CP   VH  0009
13:14:59       PROCESSOR 0A  PARKED    CP   VH  0010
13:14:59     NESTING LEVEL: 01  ID: 06
13:14:59       PROCESSOR 0D  PARKED    CP   VH  0013
13:14:59   NESTING LEVEL: 02  ID: 02
13:14:59     NESTING LEVEL: 01  ID: 02
13:14:59       PROCESSOR 14  PARKED    CP   VM  0020
13:14:59     NESTING LEVEL: 01  ID: 04
13:14:59       PROCESSOR 15  PARKED    CP   VM  0021
13:14:59       PROCESSOR 16  PARKED    CP   VL  0022
13:14:59       PROCESSOR 17  PARKED    CP   VL  0023
13:14:59     NESTING LEVEL: 01  ID: 05
13:14:59       PROCESSOR 0B  PARKED    CP   VH  0011
13:14:59       PROCESSOR 0C  PARKED    CP   VH  0012
13:14:59       PROCESSOR 13  PARKED    CP   VH  0019
Ready; T=0.01/0.01 13:14:59

```

Here we see the placements of our LPUUs on the physical topology.

For example,

- LPU 00: Vh, book 1, chip 1
- LPU 15: Vm, book 2, chip 4

*Nesting level* just refers to book, chip, etc. They are numbered from smallest to largest:

- z10: book=1
- z196, zEC12: chip=1, book=2

*CP Monitor has been updated to log out logical CPU polarity.*

## z/VM HiperDispatch: What Does “Parked” Mean?

- A *parked* logical CPU is simply not participating in the running of the system’s work.
- It is still varied-on
- It is still a configured logical CPU as far as PR/SM is concerned
- It still counts as far as software licensing is concerned
- It is sitting in a barely-enabled wait-state PSW waiting for somebody to wake it up
- It might sit there in a wait for a really long time
- When we need it, we will signal it aka *unpark* it.
- Unparking requires a SIGP and some wakeup processing. Much faster than VARY ON.

## z/VM HiperDispatch: Running According to Available Power

Your available power  $A$  = your entitled power  $E$  + your excess power fraction  $XPF$ .

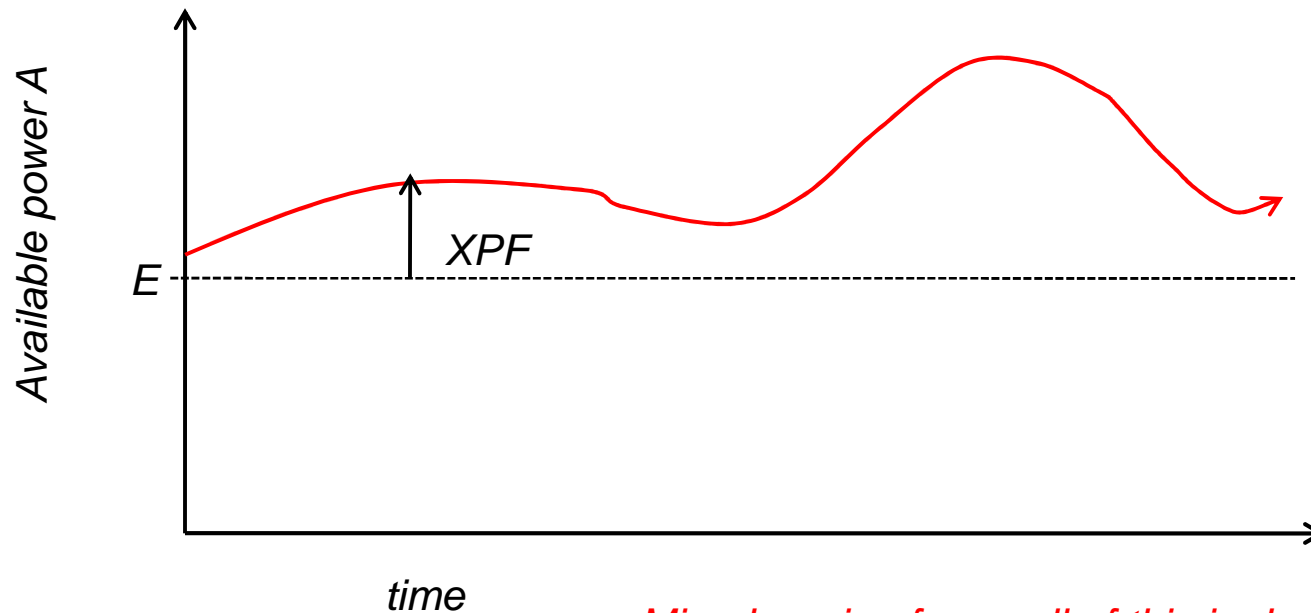
In other words, you can use your  $E$  plus what PR/SM will let you use from the excess power  $XP$ .

-- You can have all of the  $XP$  no one else wants, or your weight-fraction among your competitors.

You want to run with just the right number of CPUs to be able to consume  $E + XPF$  if you need it.

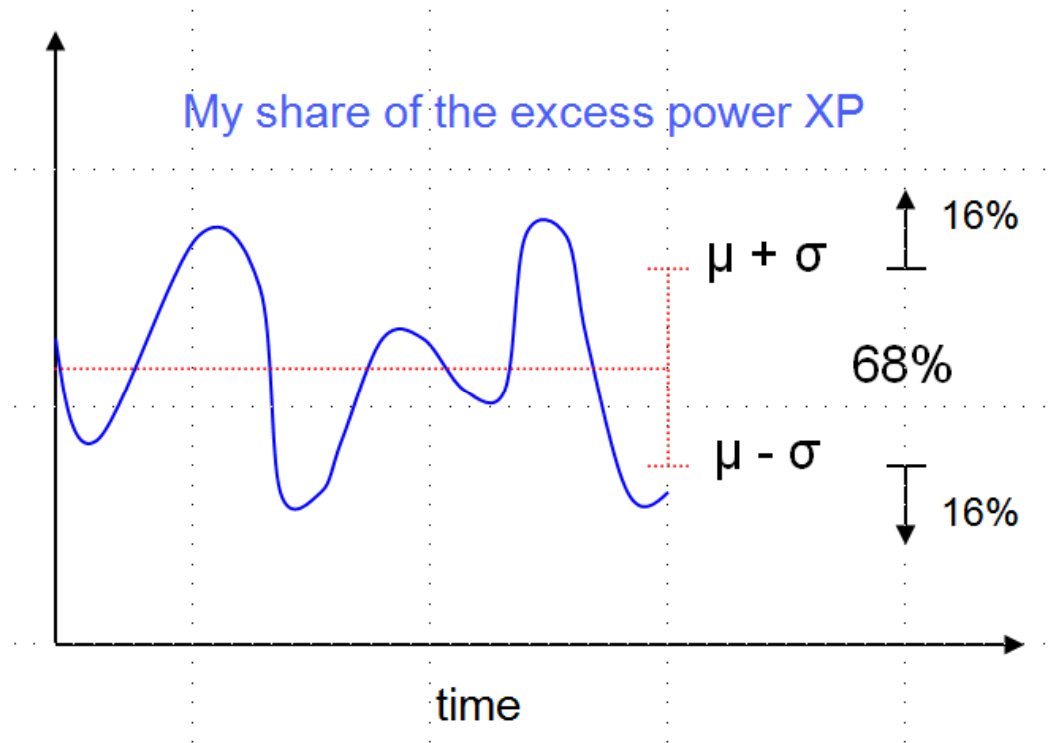
-- For example, if  $E+XPF = 1458\%$ , you need 15 CPUs unparked to consume it.

The trick in selecting the number of CPUs to use is to guess well about how much  $XPF$  you are likely to have for the next little bit of time.



*Mixed-engine fans: all of this is done by CPU-type-pool.*

## z/VM HiperDispatch: How We Calculate XPF'



*CP Monitor has been updated to log out all of the observations and all of the predictions.*

### Every two seconds, we:

- Query all partitions' weights and consumptions, so we can...
- Figure out how much excess power is available to compete for, and...
- Who our competitors for it are...
- And this tells us what our XPF is.

We keep a history of our last 10 observations of XPF.

Using the observation history we **statistically project a floor** for XPF, called XPF', for the next two seconds.

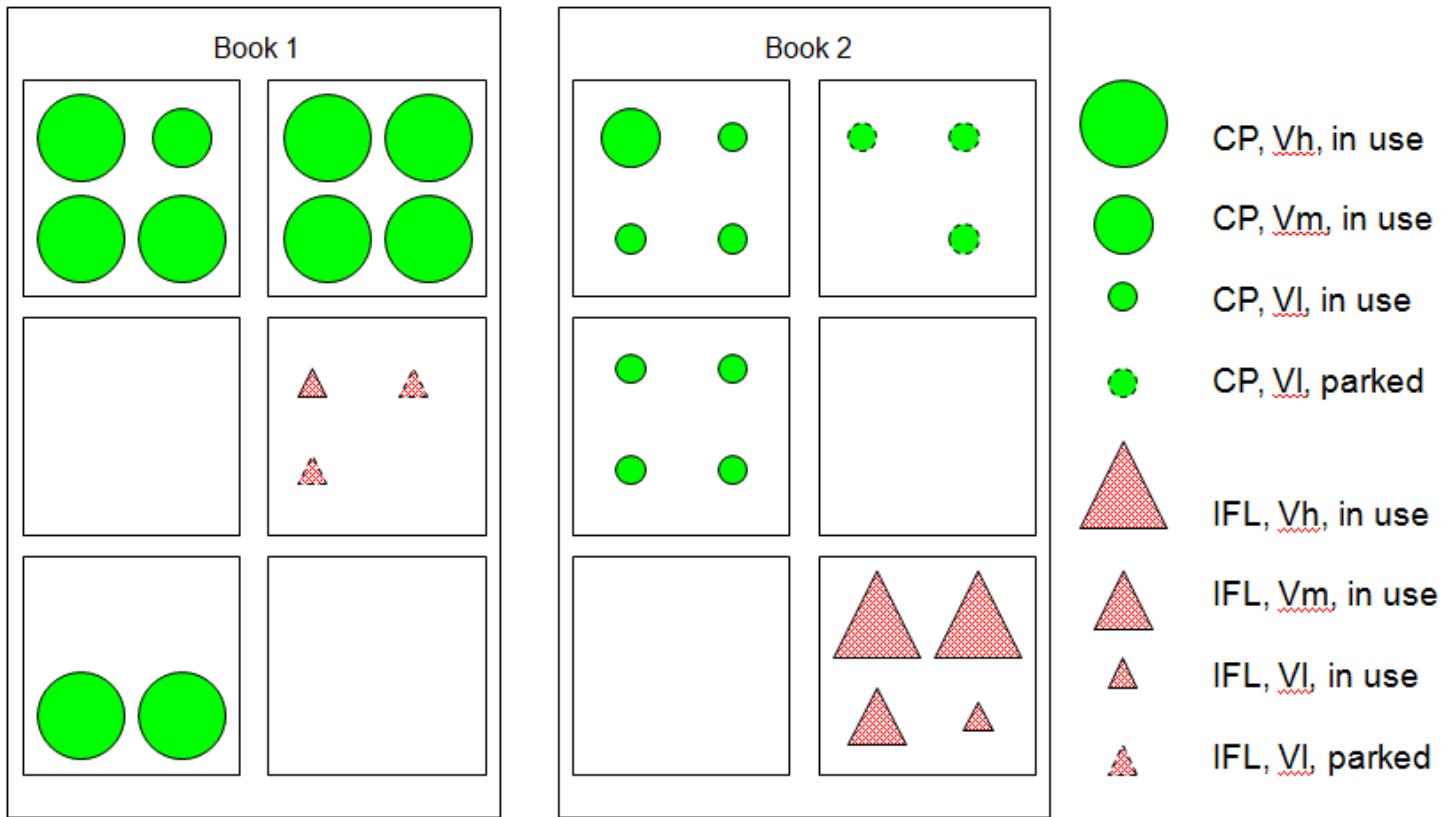
And we then *park* or *unpark* according to the engines needed to consume predicted  $A' = E + XPF'$ .

## z/VM HiperDispatch: Importance of Global Performance Data

- “Global Performance Data” is a setting in the partition’s activation profile, “Security” category
  - Look for the checkbox labelled “Performance Data Control”
  - Also you can use the SE’s “Change LPAR Security” function to change it while the partition is up
  - z/VM can handle changes in GPD without a re-IPL
  
- GPD is on by default (in DR scenario, ask your partition provider about it)
  
- When it is on, the partition can see performance data about all partitions
  - Their weights
  - How much CPU they are consuming
  
- That performance data lets the z/VM system do all of these things:
  - Determine every partition’s entitlement
  - Determine how much entitled power is being consumed
  - Determine how much excess power is available ( $XP = TP - EP$ )
  - Determine which partitions are overconsuming
  - Calculate the z/VM system’s XPF
  
- z/VM HiperDispatch is substantially crippled if you fail to enable GPD for the partition
  - You might see HCP1052I, “Global performance data is disabled. This may degrade system performance.”
  - You can always use CP QUERY SRM to find out whether GPD is on for your partition

# z/VM HiperDispatch: Which LPUUs Do We Park?

## Sample Partition Snapshot

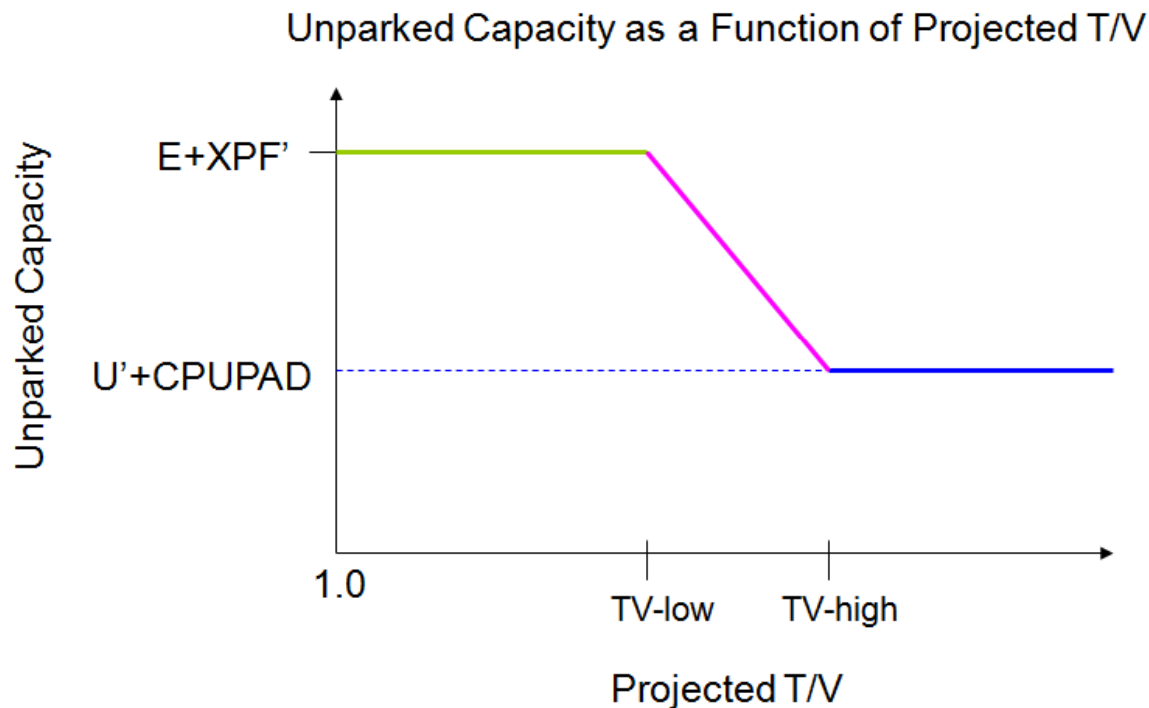


We park topological outliers.

*CP Monitor has been updated to log out the park/unpark state every two seconds.*

## z/VM HiperDispatch: Reducing MP Level to Avoid Overhead

Sometimes, less is more.



Just as we project a floor on XPF, we also project:

- A *ceiling*  $U'$  on partition's CPU utilization.
- A *ceiling*  $T'$  on partition's T/V ratio.

Then, if  $U'$  is small enough and  $T'$  is large enough, we *park* LPUs to try to get rid of overhead.

Severity of parking below  $E+XPF'$  can be controlled by setting a safety margin or CPUPAD value that we add to  $U'$ .

*CP Monitor has been updated to log out all of the observations and all of the predictions.*

**We do not park below  $E+XPF'$  on low  $T'$  because being wide is not hurting us and the parallelism is apparently there for us to use.**

## z/VM HiperDispatch: Guest Dispatch Objectives and Strategy

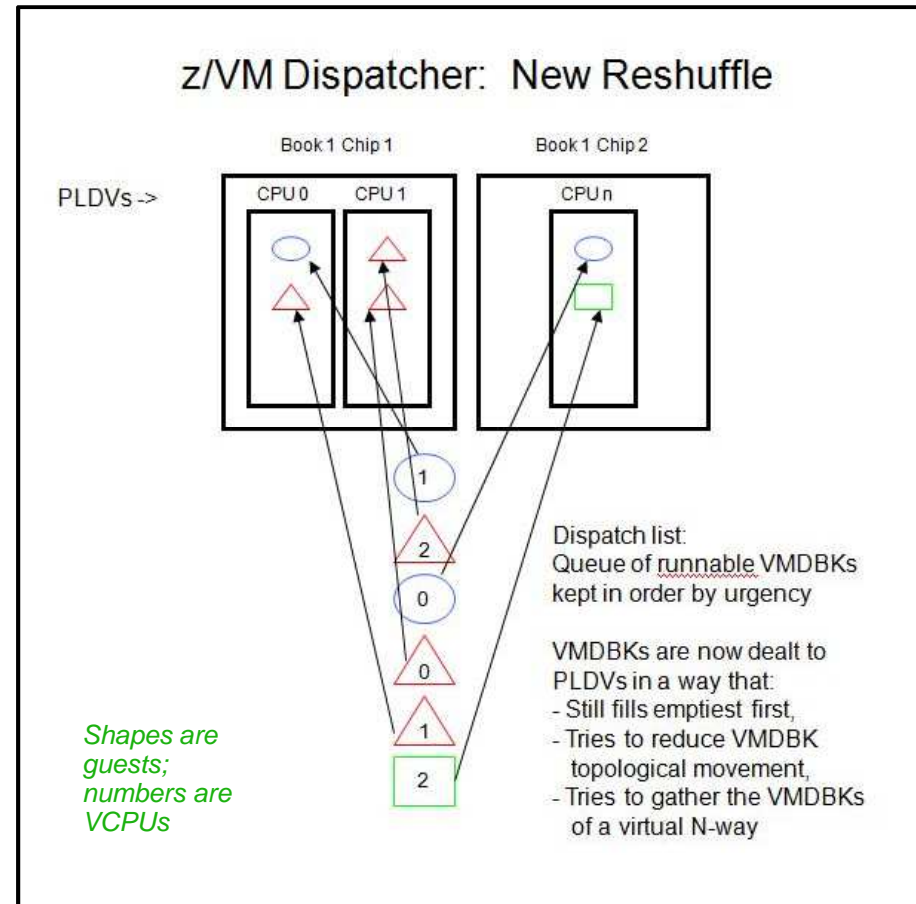
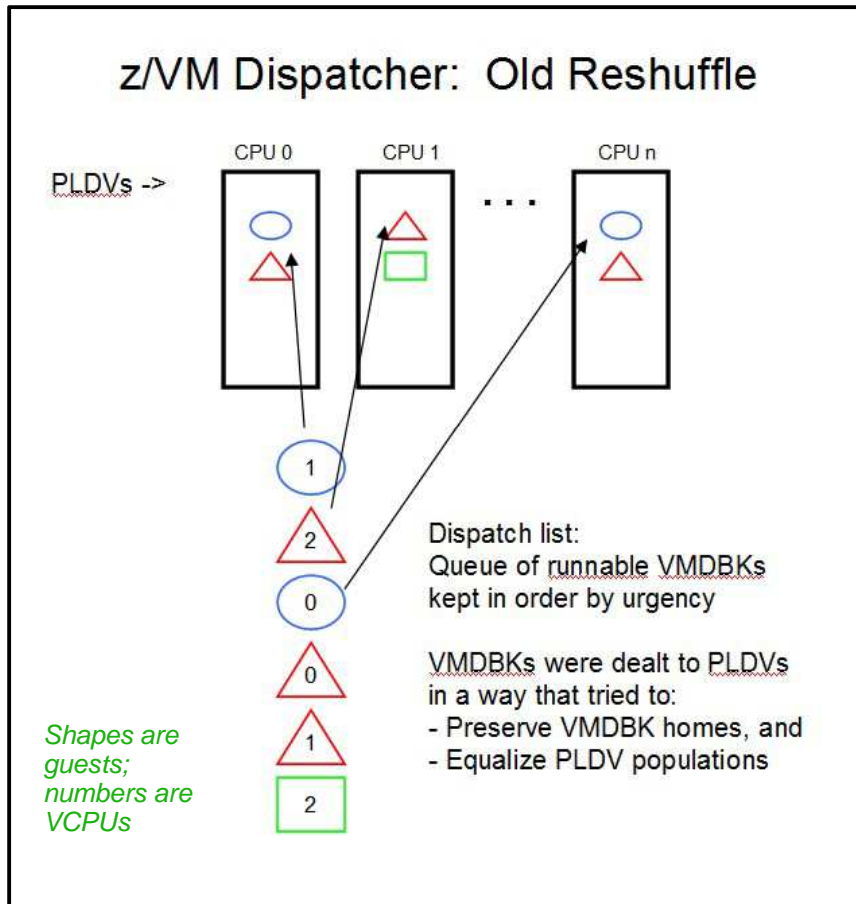
- Objectives: compared to earlier z/VM releases,
  - Reduce movement of virtual CPUs
  - Try to place the virtual CPUs of an N-way guest close to one another
  
- Strategies:
  - We made several small changes or additions:
    - Reshuffle
    - VMDBK steal
    - Work stacking wakeup
    - Needs help
  
  - We added a new work distribution algorithm:
    - Rebalance



# z/VM HiperDispatch: Reshuffle Changes

## Horizontal mode

## Vertical mode



- Balances PLDV populations.
- If not home, then anywhere.
- No awareness of virtual N-ways.

- Still balances PLDV populations.
- If not home, then hunt outward topologically.
- Collects virtual N-ways.

## z/VM HiperDispatch: VMDBK Steal

### OLD WAY

0 → 1 → 2 → 3 → 4 ... → 19 → 0

Steal from neighbor by CPU number.

Work your way around the ring.

This is not topologically informed.

### NEW WAY

(Easy) Steal within your chip.

(Harder) Steal within your book.

(Still harder) Steal across books.

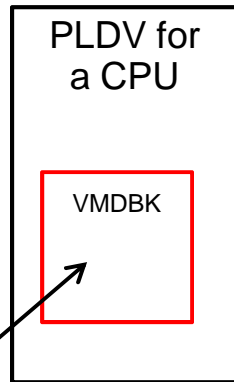
This is topologically informed.

**Barriers are for  
vertical mode only.**

*CP Monitor has been updated to  
log out steal behavior as a function  
of topology drag distance.*

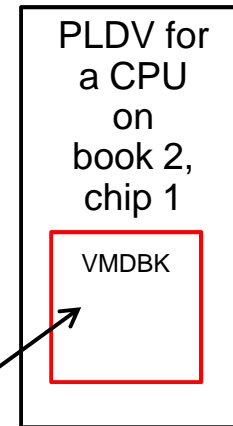
# z/VM HiperDispatch: Work Stacking CPU Wakeup

## Horizontal mode



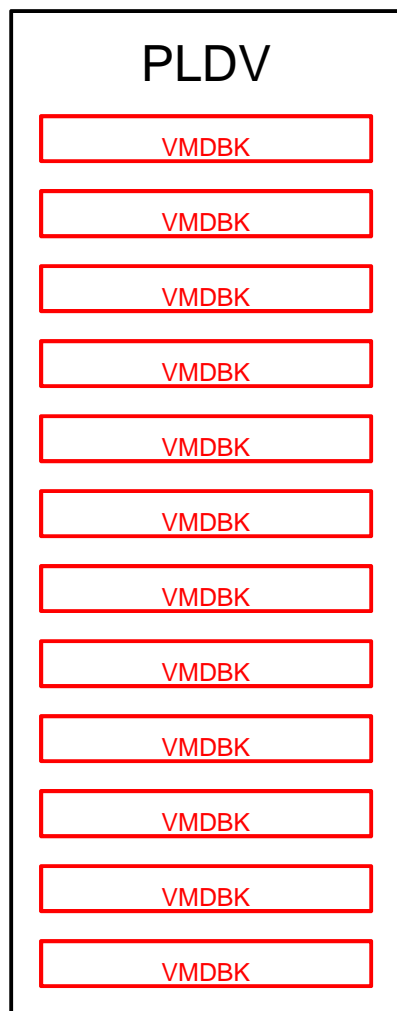
- Stack work on PLDV
- If target CPU is busy,
- Find first wait-state CPU right of stack target (CPU 0, 1, 2, 3, ...)
- Wake up the found CPU to prowl for steal

## Vertical mode



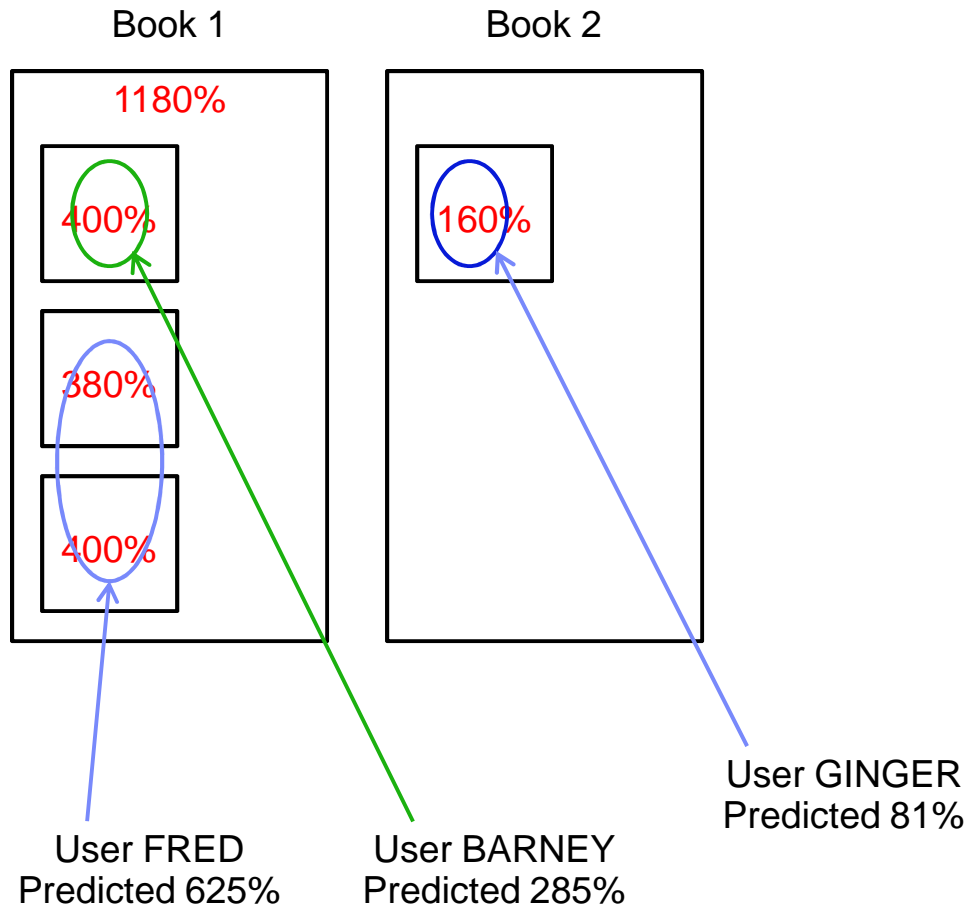
- Stack work on PLDV
- If target CPU is busy,
- Is there a wait-state CPU in this chip?
- Is there a wait-state CPU in this book?
- Is there a wait-state CPU anywhere?

## z/VM HiperDispatch: Needs Help



- Come out of wait
- Start working off my PLDV's VMDBKs
- About every minor time slice, calculate, "How long since I woke up?"
- If greater than a **very long time**, wake up the topologically closest waiter **anywhere** in the system so as to start him prowling to steal
- If greater than **only a moderate time**, wake up the topologically closest waiter **in my book** so as to start him prowling to steal

# z/VM HiperDispatch: Rebalance



*CP Monitor has been updated to log out the decisions of rebalance.*

## Rebalance highlights:

- Periodic rework of the assignments of all guests to the topological containers
- Reassigns every guest every pass, not just the VMDREADY, dispatch-list-resident VMDBKs as reshuffle does
- Predicts all guests' near-future utilizations
- Assigns guests to containers like this:
  - Predicted heaviest guests first
  - Spreads load over all containers
  - Tries not to split guests
- Good for situations where:
  - Guests' utilizations are easily distinguished from one another
  - A few heavy guests need not to move around
  - Movement of light users is OK
  - VCPU:LCPU ratio not too big

## z/VM HiperDispatch: Knobs

| Concept                                                                   | Knob                                                   |
|---------------------------------------------------------------------------|--------------------------------------------------------|
| Horizontal or vertical                                                    | SET SRM POLARIZATION { HORIZONTAL   VERTICAL }         |
| How optimistically to predict XPF floors                                  | SET SRM [TYPE cpu_type] EXCESSUSE { HIGH   MED   LOW } |
| How much CPUPAD safety margin to allow when we park below available power | SET SRM [TYPE cpu_type] CPUPAD nnnn%                   |
| Reshuffle or rebalance                                                    | SET SRM DSPWDMETHOD { RESHUFFLE   REBALANCE }          |

### Defaults:

- Vertical mode
- EXCESSUSE MEDIUM (70%-confident floor)
- CPUPAD 100%
- Reshuffle

*CP Monitor has been updated to log out the changes to these new SRM settings.*

## z/VM HiperDispatch: Horizontal Mode vs. Vertical Mode

- Horizontal mode
  - All unparked all the time
  - Reshuffle, but old-style
    - Not topologically aware
    - Does not gather virtual N-ways
  - Steal prowls topologically outward
  - Barrier-free steal
  - Work-stack wakeup is not topologically aware
  - Needs-help is in effect
  - LPU dedicate to guest is OK
- It's very much like z/VM 6.2
- Vertical mode
  - Unparks according to  $A' = E + XPF'$
  - Parks below  $A'$  if  $U'$  seems low and  $T/V'$  seems high
  - Reshuffle is new-style
    - Knows system topology
    - Knows about virtual N-ways
  - Steal prowls topologically outward
  - Difficulty barriers in steal
  - Work-stack wakeup is topologically aware
  - Needs-help is in effect
  - *Cannot dedicate an LPU to a guest*
- More topological awareness

## z/VM HiperDispatch: Aspects of Dedicated Partitions

- The physical PUs backing the partition are not part of the shared physical CPU pool
- If it is a mixed-engine partition, all CPU types are dedicated
- There's no such thing as "weight"
- Its entitlement E is  $N * 100\%$
- A dedicated partition never consumes from XP. XPF=0 always.
- If you run a dedicated partition in vertical mode,
  - All of the logical PUs are vertical highs (Vh)
  - z/VM will park a logical PU only because of high T/V projections



# Planning for z/VM HiperDispatch

## z/VM HiperDispatch: Planning for It

- Normal best practices for migrating from an earlier release certainly apply
- While you are still on the earlier release, collect measurement data:
  - Know what your key success metrics are and what their success thresholds are
  - Transaction rates – *only you* know where these are on your workloads
  - MONWRITE files – some tips:
    - When: Daily peaks? Month-end processing? Quarter-end processing?
    - Collection tips: <http://www.vm.ibm.com/devpages/bkw/monwrite.html>
    - CPU MF tips: <http://www.vm.ibm.com/perf/reports/zvm/html/620con.html>
    - CPU MF reduction: <http://www.vm.ibm.com/perf/tips/cpumf.html>
- Remember to turn on Global Performance Data for your z/VM partition
- Then go ahead and try z/VM 6.3
  - Remember the default for z/VM 6.3 is vertical mode
  - Consider asking IBM whether your workload is amenable to using rebalance
- When you start running on z/VM 6.3, collect the very same measurement data
- Compare z/VM 6.3 back to z/VM 6.2 to see what the effect is on your workload
- If you like, you can revert to horizontal mode with these means:
  - CP SET SRM POLARIZATION HORIZONTAL
  - SRM statement in the system configuration file

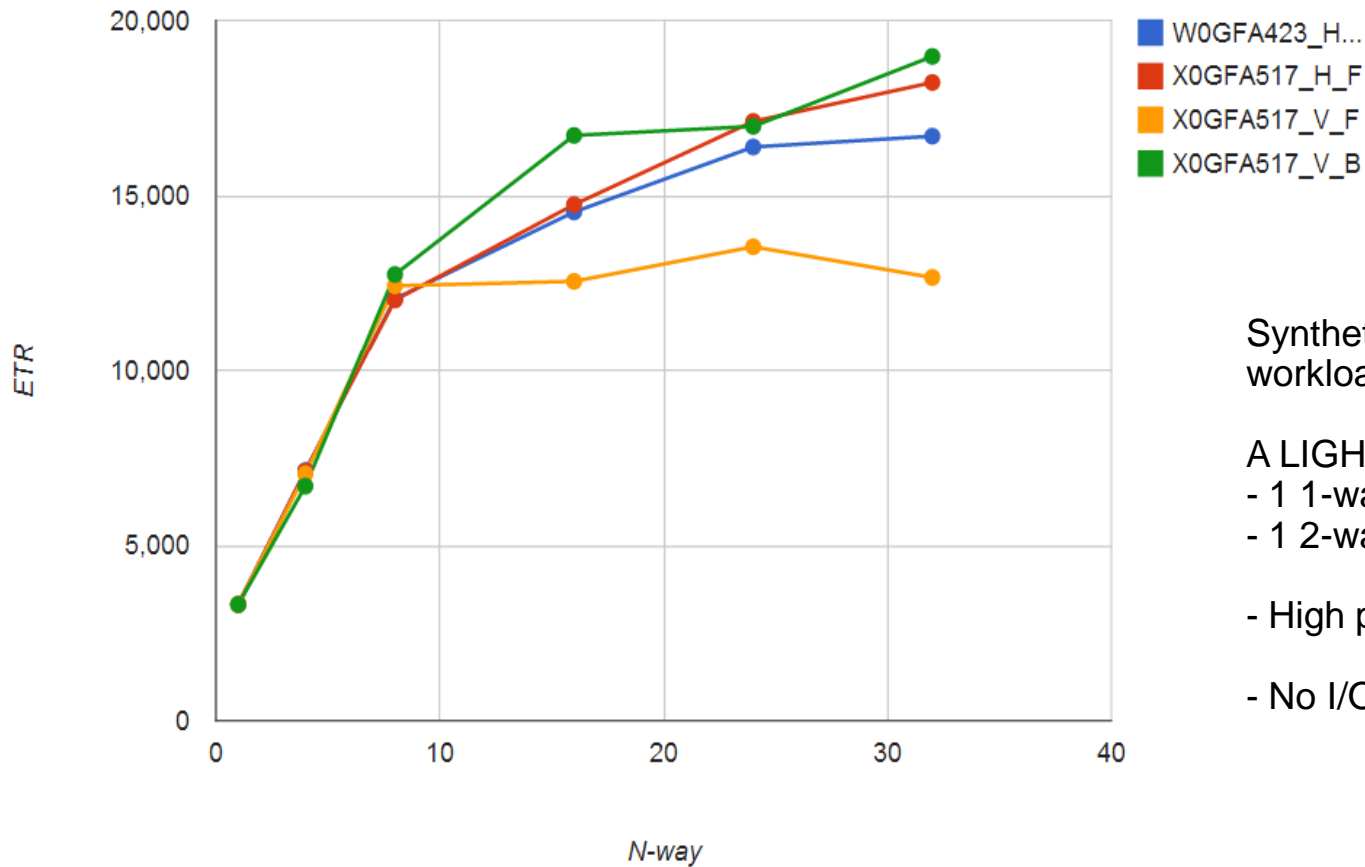
# Comments on Workloads

## z/VM HiperDispatch: Traits of Workloads

- Amenable workloads for z/VM HiperDispatch:
  - High-CPU, CPU-constrained workloads
    - Improving cache behavior stands to improve performance
  - Active VCPU:LCPU ratio isn't too large
    - High ratio has too much context switching to feel much effect
  - Runs in a partition having multiple topology containers
    - Gives z/VM an opportunity to separate guests from one another
  
- Compare those statements to IBM's statements about PR/SM and partitions
  
- Indifferent workloads for z/VM HiperDispatch
  - Constrained by something else, such as I/O
  - Memory-overcommitted
  - High VCPU:LCPU ratio with every virtual CPU active just a little bit
  - Workloads with bad memory access habits
  
- Remember that vertical mode also keeps your partition away from the other partitions

# z/VM HiperDispatch: Various Numbers of LIGHT Tiles

zEC12 ETR as f(N-way) for 8 LIGHT, Low T/V



Synthetic, memory-touching workload

A LIGHT tile is 81% busy:

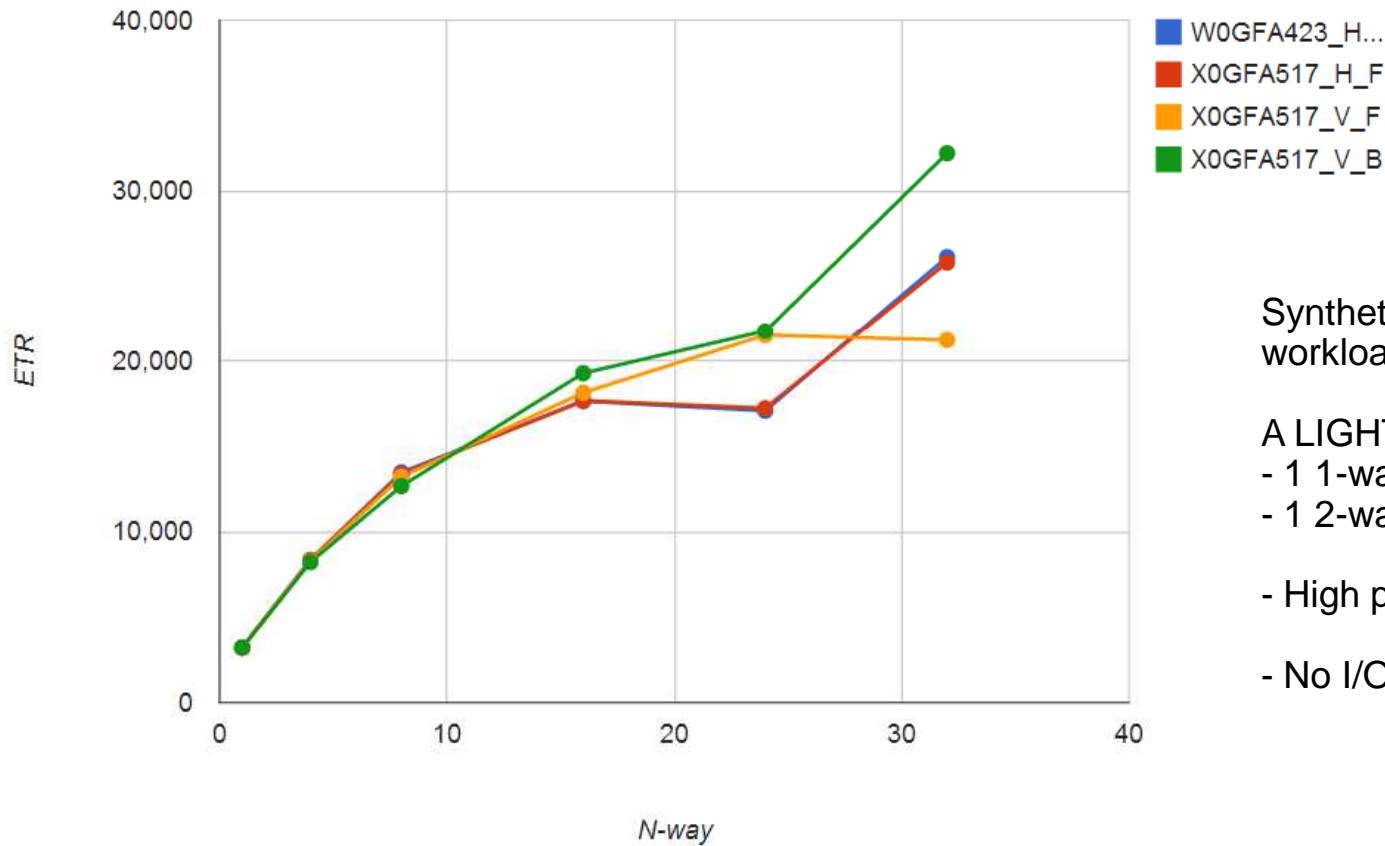
- 1 1-way @ 15% each
- 1 2-way @ 33% each

- High pressure on nest

- No I/O, paging, etc.

# z/VM HiperDispatch: Various Numbers of LIGHT Tiles

zEC12 ETR as f(N-way) for 16 LIGHT, Low T/V



Synthetic, memory-touching workload

A LIGHT tile is 81% busy:

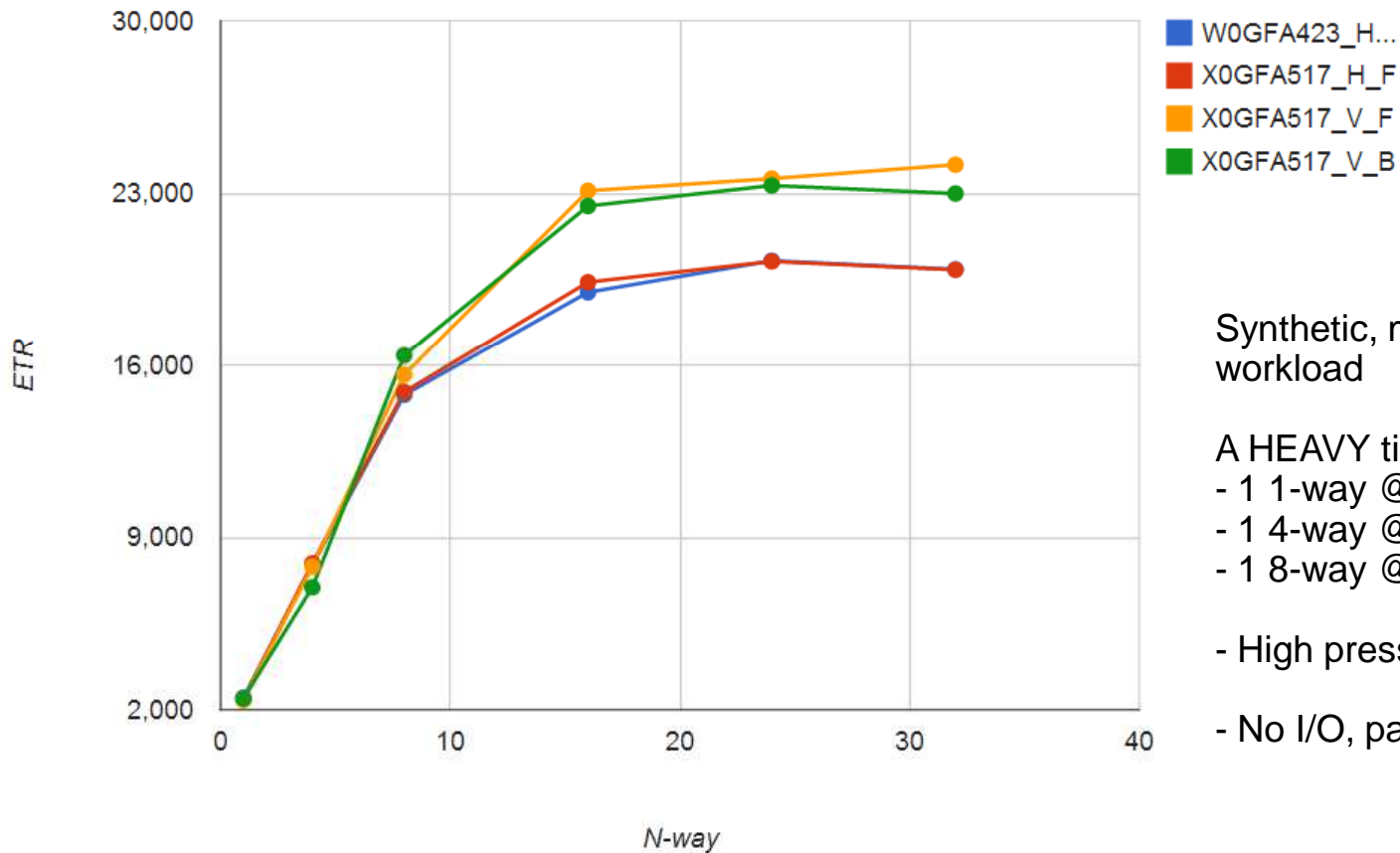
- 1 1-way @ 15% each
- 1 2-way @ 33% each

- High pressure on nest

- No I/O, paging, etc.

# z/VM HiperDispatch: Various Numbers of HEAVY Tiles

zEC12 ETR as f(N-way) for 2 HEAVY, High T/V



Synthetic, memory-touching workload

A HEAVY tile is 540% busy:

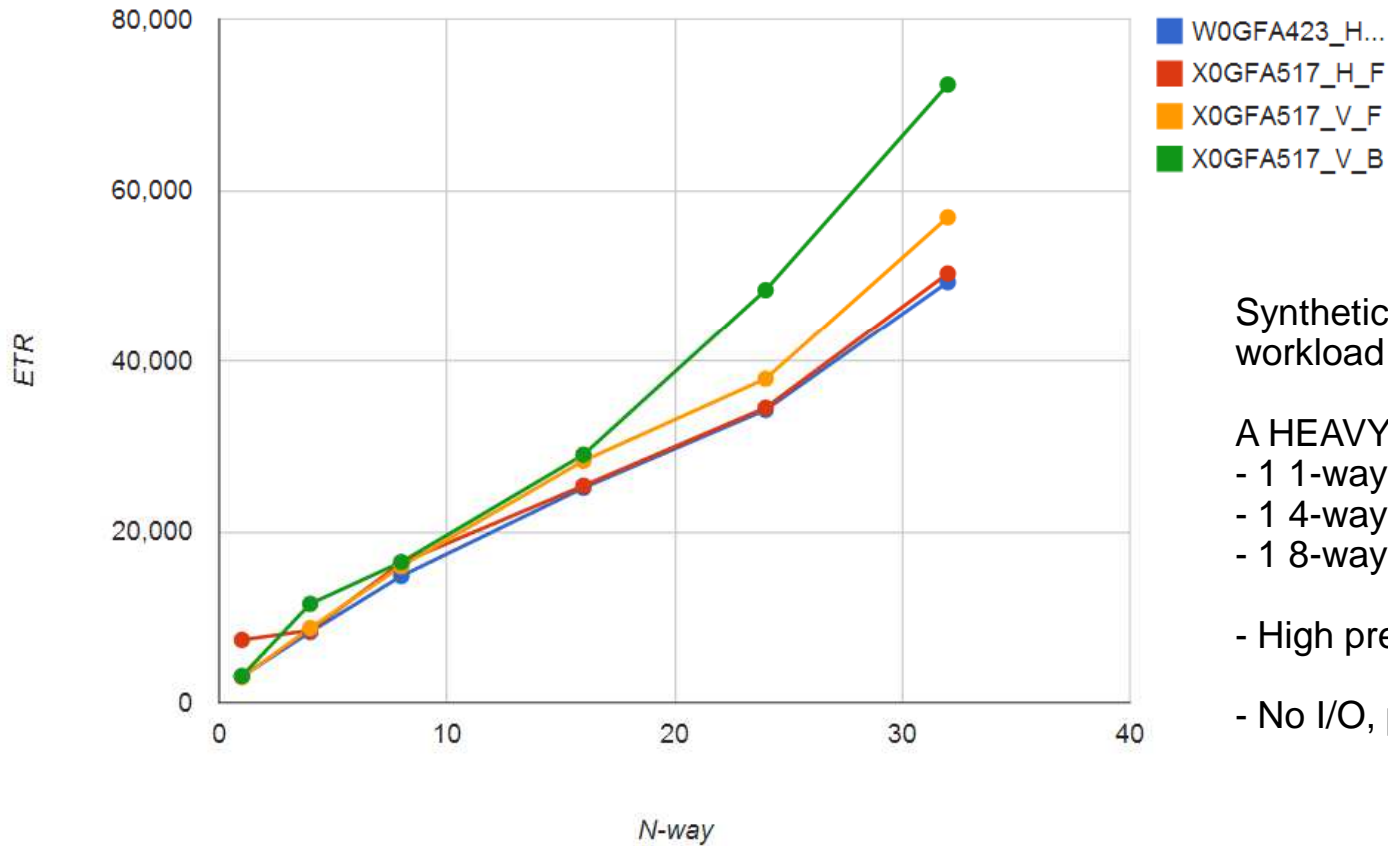
- 1 1-way @ 15% each
- 1 4-way @ 31% each
- 1 8-way @ 50% each

- High pressure on nest

- No I/O, paging, etc.

# z/VM HiperDispatch: Various Numbers of HEAVY Tiles

zEC12 ETR as f(N-way) for 6 HEAVY, Low T/V



Synthetic, memory-touching workload

A HEAVY tile is 540% busy:

- 1 1-way @ 15% each
- 1 4-way @ 31% each
- 1 8-way @ 50% each

- High pressure on nest

- No I/O, paging, etc.



# CP Monitor and z/VM Performance Toolkit

## z/VM HiperDispatch: CP Monitor Records

| Domain | Record    | Name     | Type   | Description of Change                                                 |
|--------|-----------|----------|--------|-----------------------------------------------------------------------|
| D0     | R2        | MRSYTPRP | sample | Added polarity, entitlement, and park-time accumulator                |
| D0     | R16       | MRSYTCUP | sample | Added partition current weight                                        |
| D0     | R23       | MRSYTLCK | sample | Added the HCPDSVTL topology lock                                      |
| D1     | R4        | MRMTRSYS | config | Added bit indicating whether system is horizontal or vertical         |
| D1     | R5        | MRMTRPRP | config | Added park state, polarization, entitlement, and topological location |
| D1     | R16       | MRMTRSCH | config | Added h/v bit, CPUPAD settings, and EXCESSUSE settings                |
| D2     | R7        | MRSCLSRM | event  | Added h/v bit, CPUPAD settings, and EXCESSUSE settings                |
| D4     | R2        | MRUSELOF | event  | Added rebalance results and steal results                             |
| D4     | R3        | MRUSEACT | sample | Added rebalance results and steal results                             |
| D5     | R2        | MRPRCVOF | event  | Added park/unpark failure as reason varied off                        |
| D5     | R3        | MRPRCVON | event  | Added parked as a state; use iff neither D5 R17 nor D5 R18 are seen   |
| D5     | R15 (new) | MRPRCDSV | event  | Records assignment of processors to dispatch vectors                  |
| D5     | R16 (new) | MRPRCPUP | event  | Records park/unpark decision                                          |
| D5     | R17 (new) | MRPRCRCD | sample | Records processor's VMDBK steal behavior                              |
| D5     | R18 (new) | MRPRCDHF | sample | Records PLDV population trends                                        |

As usual, the monitor records will be on [www.vm.ibm.com](http://www.vm.ibm.com) at GA.

## z/VM HiperDispatch: z/VM Performance Toolkit

- Themes in the changes in existing Perfkit screens
  - CPU entitlement appears in sensible places, e.g. FCX100 CPU
  - Percent-parked appears in sensible places, e.g. FCX100 CPU
  - Parked time is correctly accounted for, e.g. FCX126 LPAR %Susp
  - SRM settings are reported where they ought to be, e.g. FCX154 SYSSET
  - Interesting events are reported in FCX180 SYSCONF as they should
  - Number of unparked CPUs appears in sensible places, e.g. FCX225 SYSSUMLG
  - Counts of new monitor records appear in FCX155 MONDATA as they should
  - Obsolete data is compatibly deleted in certain places, e.g. FCX144 PROCLOG
- New reports sure to attract interest:
  - FCX287 TOPOLOG shows a log of partition topology, container-major
  - FCX298 PUORGLOG shows a log of partition topology, CPU-major
  - FCX299 PUCFGLOG shows a log of the park/unpark state
  - FCX301 DSVBKACT replaces the PLDV emptiness columns on FCX144 PROCLOG
  - FCX302 PHYSLOG shows a physical CPU utilization log of the CEC by type pool
  - FCX303 DSVSLOG replaces the PLDV steal columns on FCX144 PROCLOG
  - FCX304 PRCLOG is where you should now look instead of FCX144 PROCLOG
  - FCX306 LSHARACT reports the partitions' entitlements vs. logical CPU counts
- Obsolete reports
  - FCX144 PROCLOG is still there for now, but start using FCX304 PRCLOG instead

## z/VM HiperDispatch: Interesting New Report PUORGLOG

1FCX298 Run 2013/05/20 10:39:48

PUORGLOG  
Processor Unit organization log

From 2013/05/19 03:39:31

To 2013/05/19 03:41:31

For 120 Secs 00:02:00

Result of GF003855 Run

Logical PU organization for Partition PPRF1 (GDLBOFVM)

| Date              | Time     | CPU | Type | PPD | Ent. | Location |
|-------------------|----------|-----|------|-----|------|----------|
| 05/19             | 03:39:31 | 0   | CP   | Vhd | 100  | 1:6      |
| 05/19             | 03:39:31 | 1   | CP   | Vhd | 100  | 1:6      |
| 05/19             | 03:39:31 | 2   | CP   | Vhd | 100  | 1:5      |
| 05/19             | 03:39:31 | 3   | CP   | Vhd | 100  | 1:5      |
| 05/19             | 03:39:31 | 4   | CP   | Vhd | 100  | 1:5      |
| 05/19             | 03:39:31 | 5   | CP   | Vhd | 100  | 1:5      |
| 05/19             | 03:39:31 | 6   | CP   | Vhd | 100  | 1:5      |
| 05/19             | 03:39:31 | 7   | CP   | Vhd | 100  | 1:4      |
| 05/19             | 03:39:31 | 8   | CP   | Vhd | 100  | 1:4      |
| 05/19             | 03:39:31 | 9   | CP   | Vhd | 100  | 1:4      |
| 05/19             | 03:39:31 | 10  | CP   | Vhd | 100  | 1:4      |
| 05/19             | 03:39:31 | 11  | CP   | Vhd | 100  | 1:2      |
| 05/19             | 03:39:31 | 12  | CP   | Vhd | 100  | 1:2      |
| 05/19             | 03:39:31 | 13  | CP   | Vhd | 100  | 1:2      |
| 05/19             | 03:39:31 | 14  | CP   | Vhd | 100  | 1:2      |
| ... truncated ... |          |     |      |     |      |          |

### Notes:

Vh: vertical high

Vm: vertical medium

VI: vertical low

VhD: vertical high, dedicated partition

Ent: entitlement wrt a physical CPU

Location: book:chip (z10: book)

# z/VM HiperDispatch: Interesting New Report LSHARACT

```
1FCX306 Run 2013/06/24 09:36:54 LSHARACT
Logical Partition Share
From 2013/02/19 11:49:58
To 2013/02/19 11:56:10
For 372 Secs 00:06:12 Result of GFCM0107 Run
```

LPAR Data, Collected in Partition RPRF2

```
Physical PUs, Shared: CP- 40 ZAAP- 2 IFL- 16 ICF- 1 ZIIP- 3
Dedicated: CP- 4 ZAAP- 0 IFL- 0 ICF- 0 ZIIP- 0
```

| Proc Type | Partition Name | LPU Count | LPAR Weight | Entlment | TypeCap | <LPU Total,%> Busy | Excess | LPU Conf |
|-----------|----------------|-----------|-------------|----------|---------|--------------------|--------|----------|
| CP        | RCPX4          | 10        | 10          | 59.3     | ...     | 3.0                | .0     | o        |
| CP        | RCTS1          | 5         | 10          | 59.3     | ...     | 311.9              | 252.6  | o        |
| CP        | RCTS2          | 5         | 30          | 177.8    | ...     | 1.0                | .0     | o        |
| CP        | RCT1           | 20        | 30          | 177.8    | ...     | 111.3              | .0     | o        |
| CP        | RCT2           | 10        | 10          | 59.3     | ...     | 11.2               | .0     | o        |
| CP        | REXT1          | 5         | 10          | 59.3     | ...     | .0                 | .0     | o        |
| CP        | REXT2          | 4         | 10          | 59.3     | ...     | .0                 | .0     | o        |
| CP        | RINS           | 10        | 10          | 59.3     | ...     | .0                 | .0     | o        |
| CP        | RPRF1          | 4         | DED         | ...      | ...     | ...                | ...    | .        |
| CP        | RPRF2          | 24        | 335         | 1985.2   | ...     | 1548.4             | .0     | o        |
| CP        | RSPX1          | 6         | 40          | 237.0    | ...     | 481.3              | 244.3  | o        |
| CP        | RSPX2          | 6         | 40          | 237.0    | ...     | 499.7              | 262.7  | o        |
| CP        | RSPX5          | 6         | 40          | 237.0    | ...     | 126.5              | .0     | o        |
| CP        | RST1           | 10        | 10          | 59.3     | ...     | 16.2               | .0     | o        |
| CP        | RST1X          | 6         | 10          | 59.3     | ...     | 102.5              | 43.2   | o        |
| CP        | RST2           | 6         | 50          | 296.3    | ...     | .9                 | .0     | o        |
| CP        | RST3           | 3         | 30          | 177.8    | ...     | 1.2                | .0     | o        |
| ICF       | RCTS2          | 1         | 10          | 25.0     | ...     | .0                 | .0     | -        |
| ICF       | RCT1           | 1         | 30          | 75.0     | ...     | .0                 | .0     | -        |
| IFL       | RCTS2          | 2         | 10          | 188.2    | ...     | .0                 | .0     | -        |
| IFL       | RCT1           | 2         | 30          | 564.7    | ...     | .0                 | .0     | u        |
| IFL       | RSTL1          | 16        | 45          | 847.1    | ...     | 449.2              | .0     | o        |
| ZAAP      | RCPX4          | 1         | 10          | 40.0     | ...     | .1                 | .0     | -        |
| ZAAP      | RCTS2          | 1         | 10          | 40.0     | ...     | .0                 | .0     | -        |
| ZAAP      | RCT1           | 1         | 30          | 120.0    | ...     | .0                 | .0     | u        |
| ZIIP      | RCPX4          | 1         | 10          | 60.0     | ...     | .3                 | .0     | -        |
| ZIIP      | RCTS2          | 1         | 10          | 60.0     | ...     | .0                 | .0     | -        |
| ZIIP      | RCT1           | 1         | 30          | 180.0    | ...     | .0                 | .0     | u        |

You now have an easy way to see the entitlements of your partitions.

## Features:

- Reports by partition and CPU type
- Reports entitlement in percent
- Reports percent-busy of the partition's CPUs of that type
- Reports whether the partition is consuming beyond its entitlement ("Excess")
- Reports LPU configuration wrt entitlement:
  - "o" – overconfigured
  - "u" – underconfigured
  - "-" – apparently just right

# z/VM HiperDispatch: Interesting New Report PUCFGLOG

1FCX299 Run 2013/06/24 09:36:54

PUCFGLOG  
Processor Unit Configuration log

Page 6

From 2013/02/19 11:49:52  
To 2013/02/19 11:56:10  
For 378 Secs 00:06:18

Result of GFCM0107 Run

GFCM0107  
CPU 2817-744 SN B6D85  
z/VM V.6.3.0 SLU 0000

| Date  | Time     | Type  | OnL | Entitl      | Type  | Cap   | CPUPAD | EX | Load   | XP     | XPF   | T/V   | LCei   | XPF   | T/V   | N  | NotVh | UpCap  | LPU               | Unparked | mask |
|-------|----------|-------|-----|-------------|-------|-------|--------|----|--------|--------|-------|-------|--------|-------|-------|----|-------|--------|-------------------|----------|------|
| 02/19 | 11:49:54 | CP    | 24  | 1985.2      | ...   | 100.0 | 70     |    | 2.2    | 1159.4 | 892.8 | 3.519 | 3.9    | 885.9 | 200.5 | 2  | .0    | 200.0  | 00300000_00000000 |          |      |
| 02/19 | 11:49:56 | CP    | 24  | 1985.2      | ...   | 100.0 | 70     |    | .5     | 1153.3 | 888.1 | 256.0 | 1.7    | 883.4 | 201.3 | 2  | .0    | 200.0  | 00300000_00000000 |          |      |
| 02/19 | 11:49:58 | CP    | 24  | 1985.2      | ...   | 100.0 | 70     |    | .5     | 1159.7 | 893.1 | 122.3 | 1.7    | 885.2 | 204.2 | 2  | .0    | 200.0  | 00300000_00000000 |          |      |
| 02/19 | 11:50:00 | CP    | 24  | 1985.2      | ...   | 100.0 | 70     |    | .7     | 1136.7 | 875.4 | 53.45 | 1.7    | 857.7 | 172.5 | 2  | .0    | 200.0  | 00300000_00000000 |          |      |
| 02/19 | 11:50:02 | CP    | 24  | 1985.2      | ...   | 100.0 | 70     |    | .9     | 1128.6 | 869.2 | 4.531 | 1.7    | 863.0 | 172.5 | 2  | .0    | 200.0  | 00300000_00000000 |          |      |
| 02/19 | 11:50:04 | CP    | 24  | 1985.2      | ...   | 100.0 | 70     |    | 1.3    | 1034.5 | 778.8 | 1.822 | 1.8    | 688.3 | 172.4 | 2  | .0    | 200.0  | 00300000_00000000 |          |      |
| 02/19 | 11:50:06 | CP    | 24  | 1985.2      | ...   | 100.0 | 70     |    | .6     | 1157.1 | 891.1 | 38.57 | 1.8    | 856.4 | 168.5 | 2  | .0    | 200.0  | 00300000_00000000 |          |      |
| 02/19 | 11:50:08 | CP    | 24  | 1985.2      | ...   | 100.0 | 70     |    | .5     | 1162.9 | 895.5 | 250.8 | 1.7    | 856.9 | 211.1 | 2  | .0    | 200.0  | 00300000_00000000 |          |      |
| 02/19 | 11:50:10 | CP    | 24  | 1985.2      | ...   | 100.0 | 70     |    | 44.8   | 1161.8 | 894.7 | 2.214 | 89.1   | 858.9 | 211.1 | 2  | .0    | 200.0  | 00300000_00000000 |          |      |
| 02/19 | 11:50:12 | * CPU |     | Park/Unpark | State |       |        |    |        |        |       |       |        |       |       |    |       |        |                   |          |      |
| 02/19 | 11:50:12 | CP    | 24  | 1985.2      | ...   | 100.0 | 70     |    | 199.7  | 1145.1 | 881.9 | 1.517 | 354.6  | 858.5 | 197.6 | 5  | .0    | 500.0  | 00300000_00000000 |          |      |
| 02/19 | 11:50:14 | * CPU |     | Park/Unpark | State |       |        |    |        |        |       |       |        |       |       |    |       |        |                   |          |      |
| 02/19 | 11:50:14 | CP    | 24  | 1985.2      | ...   | 100.0 | 70     |    | 501.6  | 1155.6 | 890.0 | 1.009 | 803.5  | 858.3 | 197.5 | 10 | .0    | 1000.0 | 013C0000_00000000 |          |      |
| 02/19 | 11:50:16 | * CPU |     | Park/Unpark | State |       |        |    |        |        |       |       |        |       |       |    |       |        |                   |          |      |
| 02/19 | 11:50:16 | CP    | 24  | 1985.2      | ...   | 100.0 | 70     |    | 999.6  | 1147.4 | 883.6 | 1.001 | 1497.6 | 857.9 | 146.5 | 16 | .0    | 1600.0 | 0FFC0000_00000000 |          |      |
| 02/19 | 11:50:18 | * CPU |     | Park/Unpark | State |       |        |    |        |        |       |       |        |       |       |    |       |        |                   |          |      |
| 02/19 | 11:50:18 | CP    | 24  | 1985.2      | ...   | 100.0 | 70     |    | 1599.3 | 1155.1 | 889.6 | 1.001 | 2199.1 | 857.7 | 130.3 | 23 | 100.0 | 2300.0 | FFFF0000_00000000 |          |      |
| 02/19 | 11:50:20 | * CPU |     | Park/Unpark | State |       |        |    |        |        |       |       |        |       |       |    |       |        |                   |          |      |
| 02/19 | 11:50:20 | CP    | 24  | 1985.2      | ...   | 100.0 | 70     |    | 2297.6 | 1179.7 | 908.5 | 1.001 | 2995.8 | 860.2 | 125.6 | 24 | 100.0 | 2400.0 | FFFFFFE0_00000000 |          |      |
| 02/19 | 11:50:22 | * CPU |     | Park/Unpark | State |       |        |    |        |        |       |       |        |       |       |    |       |        |                   |          |      |
| 02/19 | 11:50:22 | CP    | 24  | 1985.2      | ...   | 100.0 | 70     |    | 2397.1 | 1144.5 | 881.4 | 1.005 | 2496.6 | 854.3 | 125.4 | 24 | 100.0 | 2400.0 | FFFFFFF0_00000000 |          |      |
| 02/19 | 11:50:24 | CP    | 24  | 1985.2      | ...   | 100.0 | 70     |    | 2080.5 | 1181.8 | 910.1 | 1.002 | 2569.2 | 887.6 | 125.3 | 24 | 100.0 | 2400.0 | FFFFFFF0_00000000 |          |      |
| 02/19 | 11:50:26 | CP    | 24  | 1985.2      | ...   | 100.0 | 70     |    | 1681.3 | 1140.0 | 878.0 | 1.002 | 2660.9 | 845.8 | 122.1 | 24 | 100.0 | 2400.0 | FFFFFFF0_00000000 |          |      |
| 02/19 | 11:50:28 | CP    | 24  | 1985.2      | ...   | 100.0 | 70     |    | 1632.4 | 1169.6 | 900.7 | 1.002 | 2684.7 | 886.2 | 1.660 | 24 | 100.0 | 2400.0 | FFFFFFF0_00000000 |          |      |
| 02/19 | 11:50:30 | CP    | 24  | 1985.2      | ...   | 100.0 | 70     |    | 1587.7 | 1149.4 | 885.2 | 1.002 | 2635.4 | 869.6 | 1.252 | 24 | 100.0 | 2400.0 | FFFFFFF0_00000000 |          |      |
| 02/19 | 11:50:32 | CP    | 24  | 1985.2      | ...   | 100.0 | 70     |    | 1878.3 | 1129.6 | 869.9 | 1.011 | 2560.8 | 854.7 | 1.008 | 24 | 100.0 | 2400.0 | FFFFFFF0_00000000 |          |      |
| 02/19 | 11:50:34 | CP    | 24  | 1985.2      | ...   | 100.0 | 70     |    | 1824.3 | 1176.2 | 905.8 | 1.002 | 2425.8 | 884.3 | 1.007 | 24 | 100.0 | 2400.0 | FFFFFFF0_00000000 |          |      |

- Shows what can happen to T/V when utilization is really low
- Shows parking because of high projected T/V
- Shows unpark when workload ramps up
- Shows how a varying U produces a high U'
- Shows XPF and XPF' in action
- Shows that the T/V projections level right out once the guest workload is well underway
- Shows what a non-Vh LPU will be "truly worth" during the next interval

## z/VM HiperDispatch: Interesting New Report DSVSLOG

```
1FCX303 Run 2013/05/20 10:32:38      DSVSLOG
From 2013/05/19 02:03:25             DSVBK Steals per logical CPU Log, by Time
To 2013/05/19 02:05:19
For 114 Secs 00:01:54                Result of GF003820 Run
```

| Interval | C  | P  | U  | Type | PPD  | Ent. | DVID   | Pct    | Park | Time | -----> | DSVBK  | Steal  | /s     | -----> |        |
|----------|----|----|----|------|------|------|--------|--------|------|------|--------|--------|--------|--------|--------|--------|
| End Time |    |    |    |      |      |      |        |        |      |      | Lvl-00 | Lvl-01 | Lvl-02 | Lvl-03 | Lvl-04 | Lvl-05 |
| >>Mean>> | 0  | CP | vh | 100  | 0000 | 0    | 4.404  | 4.088  | .000 | .... | ....   | ....   | ....   | ....   | ....   | ....   |
| >>Mean>> | 1  | CP | vh | 100  | 0001 | 0    | 2.456  | 2.561  | .000 | .... | ....   | ....   | ....   | ....   | ....   | ....   |
| >>Mean>> | 2  | CP | vh | 100  | 0002 | 0    | 6.877  | .921   | .000 | .... | ....   | ....   | ....   | ....   | ....   | ....   |
| >>Mean>> | 3  | CP | vh | 100  | 0003 | 0    | 7.596  | .930   | .000 | .... | ....   | ....   | ....   | ....   | ....   | ....   |
| >>Mean>> | 4  | CP | vh | 100  | 0004 | 0    | 4.500  | .482   | .000 | .... | ....   | ....   | ....   | ....   | ....   | ....   |
| >>Mean>> | 5  | CP | vh | 100  | 0005 | 0    | 3.614  | .228   | .000 | .... | ....   | ....   | ....   | ....   | ....   | ....   |
| >>Mean>> | 6  | CP | vh | 100  | 0006 | 0    | 4.518  | .482   | .000 | .... | ....   | ....   | ....   | ....   | ....   | ....   |
| >>Mean>> | 7  | CP | vh | 100  | 0007 | 0    | 2.912  | .386   | .000 | .... | ....   | ....   | ....   | ....   | ....   | ....   |
| >>Mean>> | 8  | CP | vh | 100  | 0008 | 0    | 1.412  | .421   | .000 | .... | ....   | ....   | ....   | ....   | ....   | ....   |
| >>Mean>> | 9  | CP | vh | 100  | 0009 | 0    | 1.386  | .184   | .000 | .... | ....   | ....   | ....   | ....   | ....   | ....   |
| >>Mean>> | 10 | CP | vh | 100  | 000A | 0    | 2.070  | .544   | .000 | .... | ....   | ....   | ....   | ....   | ....   | ....   |
| >>Mean>> | 11 | CP | vh | 100  | 000B | 0    | 2.114  | .149   | .000 | .... | ....   | ....   | ....   | ....   | ....   | ....   |
| >>Mean>> | 12 | CP | vh | 100  | 000C | 0    | 5.886  | 1.623  | .000 | .... | ....   | ....   | ....   | ....   | ....   | ....   |
| >>Mean>> | 13 | CP | vh | 100  | 000D | 0    | 3.772  | .702   | .000 | .... | ....   | ....   | ....   | ....   | ....   | ....   |
| >>Mean>> | 14 | CP | vh | 100  | 000E | 0    | 3.026  | .675   | .000 | .... | ....   | ....   | ....   | ....   | ....   | ....   |
| >>Mean>> | 15 | CP | vh | 100  | 000F | 0    | 2.658  | .360   | .000 | .... | ....   | ....   | ....   | ....   | ....   | ....   |
| >>Total> | 16 | CP | vh | 1600 | MIX  | 0    | 59.202 | 14.737 | .000 | .... | ....   | ....   | ....   | ....   | ....   | ....   |

Reports VCPU steal behavior by the distance the steal dragged the VCPU.

- Lvl-00: you stole it from a CPU in your chip (z10: ... in your book)
- Lvl-01: you stole it from a CPU in your book (z10: ... in another book)
- Lvl-02: you stole it from a CPU on another book (z10: ... not applicable)

# z/VM HiperDispatch: Interesting New Report PHYSLOG

1FCX302 Run 2013/06/24 09:36:54

PHYSLOG  
Real CPU Utilization Log

From 2013/02/19 11:49:58  
To 2013/02/19 11:56:10  
For 372 Secs 00:06:12

Result of GFCM0107 Run

| Interval | <PU Num> | Total |     |        |        |        |        |        |        |         |
|----------|----------|-------|-----|--------|--------|--------|--------|--------|--------|---------|
| End Time | Type     | Conf  | Ded | weight | %LgclP | %Ovrhd | LpuT/L | %LPmgt | %Total | TypeT/L |
| >>Mean>> | CP       | 44    | 4   | 675    | 3387.1 | 27.947 | 1.008  | 31.870 | 3446.9 | 1.018   |
| >>Mean>> | ZAAP     | 2     | 0   | 50     | .093   | .042   | 1.451  | .424   | .559   | 6.015   |
| >>Mean>> | IFL      | 16    | 0   | 85     | 448.16 | 1.017  | 1.002  | 2.108  | 451.28 | 1.007   |
| >>Mean>> | ICF      | 1     | 0   | 40     | .004   | .003   | 1.624  | 2.257  | 2.263  | 563.66  |
| >>Mean>> | ZIIP     | 3     | 0   | 50     | .193   | .090   | 1.465  | 1.204  | 1.487  | 7.694   |
| >>Mean>> | >Sum     | 66    | 4   | 900    | 3835.5 | 29.099 | 1.008  | 37.864 | 3902.5 | 1.017   |
|          |          |       |     |        |        |        |        |        |        |         |
| 11:50:04 | CP       | 44    | 4   | 675    | 1963.9 | 33.262 | 1.017  | 36.226 | 2033.4 | 1.035   |
| 11:50:04 | ZAAP     | 2     | 0   | 50     | .004   | .001   | 1.306  | .037   | .042   | 10.107  |
| 11:50:04 | IFL      | 16    | 0   | 85     | 501.44 | 1.087  | 1.002  | 2.372  | 504.90 | 1.007   |
| 11:50:04 | ICF      | 1     | 0   | 40     | .007   | .004   | 1.566  | 2.277  | 2.289  | 312.13  |
| 11:50:04 | ZIIP     | 3     | 0   | 50     | .005   | .002   | 1.334  | .093   | .100   | 19.003  |
| 11:50:04 | >Sum     | 66    | 4   | 900    | 2465.4 | 34.356 | 1.014  | 41.006 | 2540.7 | 1.031   |
|          |          |       |     |        |        |        |        |        |        |         |
| 11:50:10 | CP       | 44    | 4   | 675    | 2074.2 | 25.632 | 1.012  | 28.117 | 2127.9 | 1.026   |
| 11:50:10 | ZAAP     | 2     | 0   | 50     | .004   | .001   | 1.340  | .003   | .008   | 2.013   |
| 11:50:10 | IFL      | 16    | 0   | 85     | 502.09 | .993   | 1.002  | 2.130  | 505.21 | 1.006   |
| 11:50:10 | ICF      | 1     | 0   | 40     | .007   | .004   | 1.568  | 2.165  | 2.176  | 322.32  |
| 11:50:10 | ZIIP     | 3     | 0   | 50     | .004   | .001   | 1.354  | .096   | .102   | 24.829  |
| 11:50:10 | >Sum     | 66    | 4   | 900    | 2576.3 | 26.632 | 1.010  | 32.511 | 2635.4 | 1.023   |
|          |          |       |     |        |        |        |        |        |        |         |
| 11:50:16 | CP       | 44    | 4   | 675    | 2753.4 | 23.553 | 1.009  | 25.725 | 2802.7 | 1.018   |
| 11:50:16 | ZAAP     | 2     | 0   | 50     | .003   | .001   | 1.352  | .002   | .007   | 2.015   |
| 11:50:16 | IFL      | 16    | 0   | 85     | 502.84 | .728   | 1.001  | 1.603  | 505.17 | 1.005   |
| 11:50:16 | ICF      | 1     | 0   | 40     | .006   | .003   | 1.508  | 2.168  | 2.178  | 335.01  |
| 11:50:16 | ZIIP     | 3     | 0   | 50     | .004   | .001   | 1.317  | .093   | .098   | 27.041  |
| 11:50:16 | >Sum     | 66    | 4   | 900    | 3256.3 | 24.287 | 1.007  | 29.592 | 3310.1 | 1.017   |
|          |          |       |     |        |        |        |        |        |        |         |
| 11:50:22 | CP       | 44    | 4   | 675    | 3859.2 | 16.099 | 1.004  | 20.018 | 3895.4 | 1.009   |
| 11:50:22 | ZAAP     | 2     | 0   | 50     | .004   | .001   | 1.326  | .003   | .008   | 2.022   |
| 11:50:22 | IFL      | 16    | 0   | 85     | 500.49 | .947   | 1.002  | 2.817  | 504.25 | 1.008   |
| 11:50:22 | ICF      | 1     | 0   | 40     | .007   | .003   | 1.510  | 1.797  | 1.807  | 264.49  |
| 11:50:22 | ZIIP     | 3     | 0   | 50     | .043   | .032   | 1.742  | .126   | .201   | 4.685   |
| 11:50:22 | >Sum     | 66    | 4   | 900    | 4359.8 | 17.082 | 1.004  | 24.760 | 4401.6 | 1.010   |

You now have an easy way to see how busy your CEC is. (At last!)

## Features:

- Tallied by CPU type (CP, IFL, ...)
- One group of rows every sample interval
- Reports all three ways CPU gets used:
  - By logical CPUs
  - By PR/SM, chargeable
  - By PR/SM, unchargeable
- New concepts:
  - LPU T/L: like "guest T/V"
  - Type T/L: like "system T/V"



# Summary

## z/VM HiperDispatch: Summary

- Objective: try to help CPU performance
- Strategies: pay attention to topology and to z/VM system overhead
- z/VM can now run in vertical mode
  - Runs just widely enough to be able to consume available power
  - Runs more narrowly when it looks like system overhead is a problem
  - Guest dispatch pays more attention to recent run location and to virtual N-way
  - CPU wakeup tries to be topologically friendly
  - VCPU steal tries to be topologically friendly
- Planning: not too difficult, just remember to measure before and after
- Amenable workloads should see improvements
- CP Monitor conveys the new information
- z/VM Performance Toolkit has been updated
  
- Thank you

## z/VM HiperDispatch: References

- z/VM Planning and Administration – nice abstract writeup on HiperDispatch
- z/VM Performance – points to P&A
- z/VM CP Commands and Utilities – descriptions of the new commands
- z/VM Performance Report on [www.vm.ibm.com/perf/](http://www.vm.ibm.com/perf/)
- “Understanding z/VM HiperDispatch” article on [www.vm.ibm.com/perf/tips/](http://www.vm.ibm.com/perf/tips/)
- This presentation cites two [www.vm.ibm.com](http://www.vm.ibm.com) articles describing z/VM and the CPU Measurement Facility.