Linux on z Systems

IBM z/VM 6.3 HiperDispatch - Polarization Modes and Middleware Performance

Before using this information and the product it supports, read the information in “Notices” on page 23.

Edition notices

Contents

About this publication

This white paper provides information about IBM® z/VM® 6.3 HiperDispatch polarization modes and their affect on middleware performance.

Authors

Linux end-to-end Performance team: Michael Johanssen, Dr. Juergen Doelle

Remarks

The web-links referred in this paper are up-to-date as of April, 2015.

Chapter 1. Introduction

General information is provided in this topic about the z/VM 6.3 *HiperDispatch* feature and the testing results described in this White Paper.

The HiperDispatch feature and APAR VM65586

z/VM 6.3 introduced the HiperDispatch feature that takes CPU topology into account when scheduling virtual CPUs. Our testing processes produced unexpectedly high values for the percentage of state samples that showed a virtual CPU to be waiting for a logical CPU, reported in the z/VM Performance Toolkit report USTAT.

By applying the z/VM 6.3 z13 SPE (APAR VM65586 and related) we did achieve a noticeable performance improvement. However, it did not improve the observed behaviour. We found that the CPU dispatch polarization mode a had significant influence on dispatch delay times and CPU usage.

This paper analyzes the performance impact of both the above items when applied to a typical customer mixed-workload running on an IBM zEnterprise 196.

Notational conventions

Table 1 shows the notation conventions used in this paper, in accordance to IEC 60027-2 Amendment 2.

Table 1. Notation conventions.

Symbol	Full name	Derivation
KiB	kibibyte	2 ** 10 byte == 1024 byte
MiB	mebibyte	2 ** 20 byte == 1048576 byte
GiB	gibibyte	2 ** 30 byte == 1073741824 byte

Chapter 2. Summary

This White Paper describes the z/VM 6.3 HiperDispatch *polarization modes* and their affect on middleware performance.

The first step in this analysis was to update z/VM 6.3 to the z13 exploitation SPE (APAR VM65586 and related). While the main focus of this APAR is the support of the new Simultaneous Multithreading (SMT) feature for z/VM 6.3 on IBM z13, the APAR also provides improvements on non-SMT z Systems machines. It contains changes in CPU and scheduler management that also benefit zEnterprise 196 systems. The workload throughput increased by 8.6% and the CPU cost was reduced by by 3.5%, which in this environment represents almost a full IFL (0.92 IFL).

The HiperDispatch feature provided by z/VM 6.3 introduced a new approach for dispatching virtual CPUs on physical CPUs that is controlled by the SRM (System Resource Manager) polarization parameter. The new dispatch behavior is called *vertical polarization*, which is also the default. The previous dispatch behavior is called *horizontal polarization*.

The use of the HiperDispatch feature in combination with vertical polarization did achieve good cache performance due to reduced motion of logical CPUs, the grouping of logical CPUs, and so on. However, this was achieved at the cost of longer dispatch waits. As a result, the USTAT report showed a higher percentages of samples in which the user waited for a CPU. However, the throughput of our workloads increased by 3% and the CPU load was reduced by 2.5 IFLs (from 28.8 IFLs by horizontal to 26.2 with vertical). We could also show that this improvement is caused by a better cache utilization of the z196 processors.

Therefore, the observed increase of dispatch delay times with our workload pattern did not indicate a decrease in performance. On the contrary, the system's overall performance was significantly improved!

Chapter 3. Hardware and software configurations

The hardware and software configurations (guests, virtual configurations, and physical configuration) that were used for testing how z/VM 6.3 HiperDispatch *polarization modes* affect middleware performance, are described in this topic.

Host system setup

The host system used in the test scenario was an IBM zEnterprise 196 (z196), Model 2817-M66.

On this host system, an LPAR was configured with processor and memory resources, as described in Table 2:

Table 2. LPAR processor and memory configuration

Resource	
Shared Processors (CPs)	32
Memory	200 GiB

The use of shared processors was chosen because this is the typical configuration for customer environments.

Note: Shared IFLs (Integrated Facility for Linux type processors) could have been used instead.

Software system setup

The operating systems, middleware, and workloads that were used in the test scenario, are described in this topic.

Table 3. Software elements used in the test scenario

Operating systems	
Hypervisor z Systems	IBM z/VM Version 6 Release 3.0, service level 1401 (64-bit) (Product number 5741-A07) • installed in z Systems LPAR
Guest operating system z Systems	Linux on z Systems SUSE Linux Enterprise Server 11 (390x), SLES 11, SP3 • installed in z/VM virtual systems
Operating system client system	Linux for System x SUSE Linux Enterprise Server 11 SP2 (x86_64) • installed on System x (x86_64) server
Middleware	
Oracle Database	Oracle Database 11*g* Enterprise Edition Release 11.2.0.3.0 – 64bit • installed under Linux on z Systems

Table 3. Software elements used in the test scenario (continued)

WebSphere Application Server	IBM WebSphere Application Server Network Deployment, 8.5.5.1 Build Number: cf011341.03 Build Date: 10/18/13 • installed under Linux on z Systems
WebServer	IBM HTTP Server (IHS) Server Server version: IBM_HTTP_Server/8.5.5.1 (Unix) Apache version: 2.2.8 (with additional fixes) Server built: Jul 11 2013 18:01:24 Build level: IHS85/webIHS1327.01 • installed under Linux on z Systems
DB/2	DB2 v10.5.0.2, s131001, IP23538 and Fix Pack 2. • installed under Linux on z Systems
Test suites	
Fio	Fio 2.1.7 Committed 2014-03-31.
Daytrader	IBM WebSphere Application Server Samples DayTrader 3.0.7 (No Caching) Full EE6 Spec Compliant for WebSphere 8.5.5.0 Date: 20130925
SwingBench	Version 2.5.0.909
Uperf	Version 1.0.4

For a complete description of the setup and configuration of the hardware and software stack see http://www.ibm.com/developerworks/linux/linux390/perf/tuning_vm.html#reoc.

Guest sizing

Table 4 shows the virtual system configuration that was used in the test scenario.

Table 4. Virtual system configuration used in the test scenario

Workload					
#	**Type**	**Level**	**Component or variant**	**# CPUs**	**Memory**
1	Database		Database server	4	300 GiB
2	WebSphere Transactional Workload	medium	IHS	1	700 MiB
3			WAS	2	4 GiB
4			DB2	1	2 GiB
5		high	IHS	1	700 MiB
6			WAS	4	4 GiB
7			DB2	2	2 GiB

Table 4. Virtual system configuration used in the test scenario (continued)

8	File system	medium	page cached	2	16 GiB
9			direct I/O	2	16 GiB
10		high	page cached	4	16 GiB
11			direct I/O	4	16 GiB
12	Java	high		2	4 GiB
13	Transactional Workload	medium		1	4 GiB
14	Network	medium	client	1	1 GiB
15			server	1	1 GiB
16		high	client	2	1 GiB

For a complete description of the setup and configuration of the workloads, see: `http://www.ibm.com/developerworks/linux/linux390/perf/tuning_vm.html#reoc`

Overview of the test environment

An overview of the environment that was used for testing how polarization modes affect middleware performance is provided in this topic.

Figure 1 shows the testing environment in regard to guests, virtual configurations, and physical configuration.

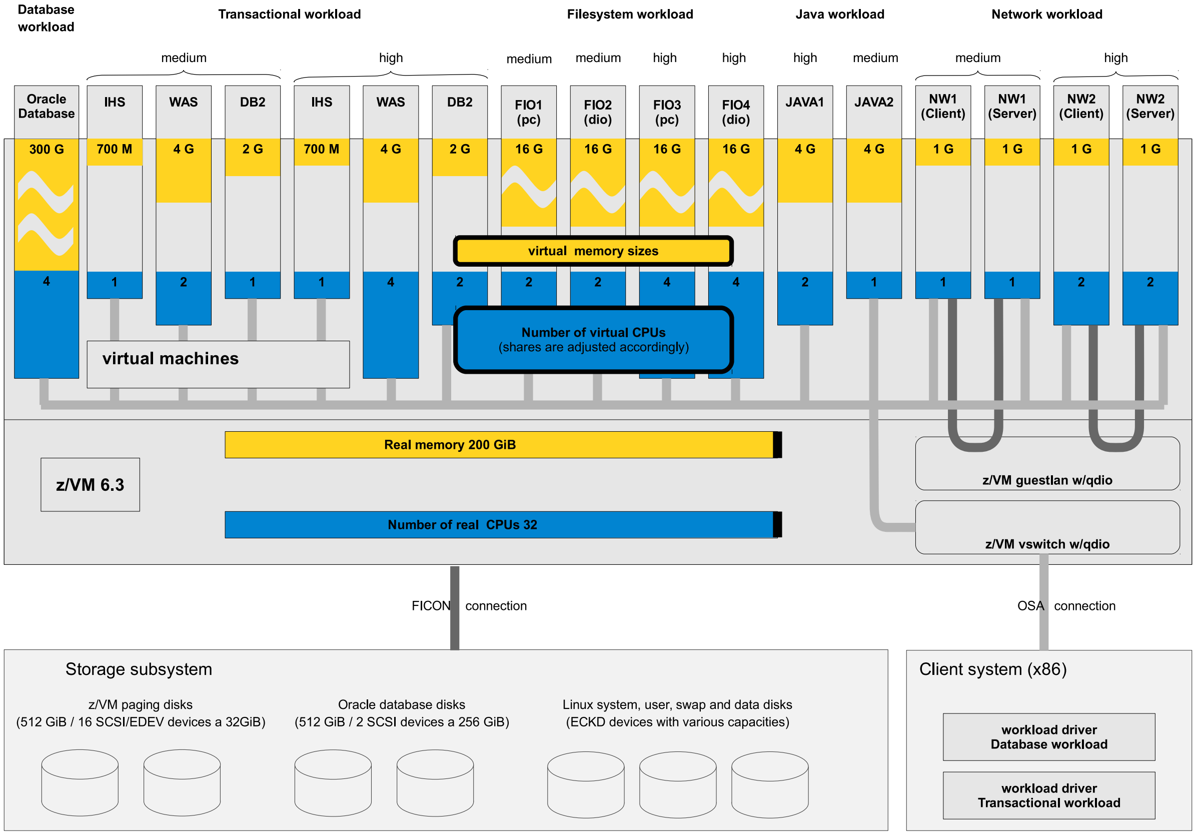

Figure 1. Overview of z/VM 6.3 HiperDispatch test environment

For a complete description of the setup and configuration of the hardware and software stack, see http://www.ibm.com/developerworks/linux/linux390/perf/tuning_vm.html#reoc.

Chapter 4. Test Methodology

The methodology that was used for testing how z/VM 6.3 HiperDispatch polarization modes affect middleware performance is described in this topic.

Test workloads

To ensure a customer-like scenario, a workload mix of very different workloads was chosen in which each represented a typical workload pattern. All workloads were driven in parallel.

Figure 1 on page 8 describes how the software stack for these workloads was implemented on the guests.

- **Database BI workload within a large virtual system**

  Swingbench was used for the database BI workload. This is a heavy disk, memory and CPU intensive workload. This was the largest guest in the sample. For further details about Swingbench workload, see:

  `http://www.dominicgiles.com/swingbench.html`
- **Transactional WAS workload**

  For the transactional WAS workload, an adaptation of the DayTrader workload to the WebSphere Application Server (WAS) was used. This workload was run in two load levels. For further details about the DayTrader workload see:
- **Java workload**

  For the Java workload, a Java program was used that executed a number of parallel threads, with each thread performing the same kind of transactions. This workload was run in two load levels. This was an IBM internal benchmark.
- **File system I/O workload**

  For the file system I/O workload, the fio tool was used. fio is an I/O tool intended to be used both for benchmarking, and for stress/hardware verification. It was run with direct I/O and with page cache usage. Each of these two variants were run additionally at two load levels. For further details about, fio see:

  `https://github.com/axboe/fio`
- **Network workload**

  For the network workload, the uperf network performance tool was used. This is a client/sever application, both components run on separate guests. Client and server part were spread into multiple processes, with each pair of processes exchanging messages through separate TCP/IP network connections. This workload was run in two load levels. For further details about uperf, see:

  `http://www.uperf.org/`

How the z/VM Performance Toolkit was used

A description of the *z/VM Performance Toolkit* that was used for testing how polarization modes affect middleware performance is provided in this topic.

Overview of z/VM HiperDispatch

In z/VM 6.3, IBM introduced a new virtual server dispatching technology called *HiperDispatch*. The prime objective of z/VM HiperDispatch is to help virtual servers to get good performance from the z Systems memory subsystem.

z/VM HiperDispatch works toward this objective by managing the partition, and dispatching virtual CPUs in a way that takes into account the physical machine's organization, but especially its memory caches. Therefore, depending upon the type of workload, this z/VM dispatching method can help to achieve good performance on z Systems hardware.

The processors of a z Systems machine are physically placed in hardware in a hierarchical, layered fashion:

- CPU cores are fabricated together on chips, perhaps six or eight cores to a chip, depending upon the model.
- Chips are assembled onto nodes, perhaps three to six chips per node, again, depending upon model.
- The nodes are then fitted into the machine's frame.

To help improve data access times, a z Systems machine uses high-speed memory caches at important points in the CPU placement hierarchy:

- Each core has its own L1 and L2.
- Each chip has its own L3.
- Each node has its own L4.
- Beyond L4 lies memory.

One way z/VM HiperDispatch tries to achieve its objective is by requesting that the PR/SM hypervisor provisions the LPAR in "vertical mode". A vertical mode partition has the property that the PR/SM hypervisor will repeatedly attempt to run the partition's logical CPUs on the same physical cores, and to run other partitions' logical CPUs elsewhere. In this way, the partition's workload benefits from having its memory references build up context in the caches. Therefore, the overall system behavior is more efficient.

Another way z/VM HiperDispatch tries to achieve its objectives is to repeatedly run the guests' virtual CPUs on the same logical CPUs. This strategy helps guests to experience the benefit of having their memory references build up context in the caches. This should also enable the individual workloads to run more efficiently.

For details of the other features of z/VM HiperDispatch, see:

`http://www.vm.ibm.com/perf/tips/zvmhd.html`

CPU MF (Measurement Facility)

The CPU MF (Measurement Facility) was used in order to analyze L1 cache misses and the cache levels where they were resolved.

The following CPU MF parameters were used:

CPUMF L1MP

L1 miss percentage. This is the percent of instructions that incur an L1 miss.

CPUMF L2P

Percent of L1 misses sourced from cache level 2.

CPUMF L3P
Percent of L1 misses sourced from cache level 3.

CPUMF L4LP
Percent of L1 misses sourced from cache level 4 from the local book .

CPUMF L4RP
Percent of L1 misses sourced from cache level 4 of a remote book.

CPUMF MEMP
Percent of L1 misses sourced from memory.

For further information about the CPU MF, see:

`www.vm.ibm.com/perf/tips/cpumf.html`

Guest dispatching metrics

The z/VM Performance Toolkit report USTAT ("FCX114, Wait State Analysis by User") can be used as a first approach for identifying guest wait states. However, the interpretation of this report requires some caution.

One aspect of report USTAT is that the reported percentages are percent of samples collected, not the percent of time spent in the various states. For more accuracy we therefore used the new *dispatch delay time accumulators* provided by the z13 exploitation SPE. These report the dispatch wait time - the amount of time (microseconds µs) that passes between the instant the virtual CPU becomes ready to use a logical CPU, and the instant the z/VM Control Program dispatches it.

The new dispatch wait delay time accumulators can be found in the CP Monitor D4 R3 record, MRUSEACT, fields USEACT_CALDWDCT and so on, which is documented at:

`http://www.vm.ibm.com/pubs/mon630/MRUSEACT.HTML`

This seems to be much more appropriate for determining the effects we wish to show in this study.

Workload performance metrics

All workload performance metrics are not presented as absolute values, but instead as the relative values with respect to the reference result.

The test scenario used z/VM 6.3 RSU (recommended service upgrade) 1401 with all SRM settings set to the default. Therefore, vertical polarization was selected.

- Four runs were carried out for each scenario.
- For each individual workload, the average of the four runs was taken as 100% value for this workload type.
- The overall throughput was then determined as the average of all the averaged individual workloads.

Chapter 5. Test results

The performance impact of the APAR VM65586 and the HiperDispatch polarization modes on middleware performance are described in this topic.

z/VM 6.3 and the z13 exploitation SPE APAR VM65586

When starting analyzing the various waits in relation with dispatching virtual CPUs, we found that a beneficial side effect of the APAR VM65586 - which is intended to enable the Simultaneous Multithreading (SMT) feature for z/VM 6.3 on IBM z13 - is also of interest for non-SMT z Systems machines.

Figure 2 compares the total workload throughput of the z/VM 6.3 RSU (recommended service upgrade) 1401 release with the SPE APAR VM65586.

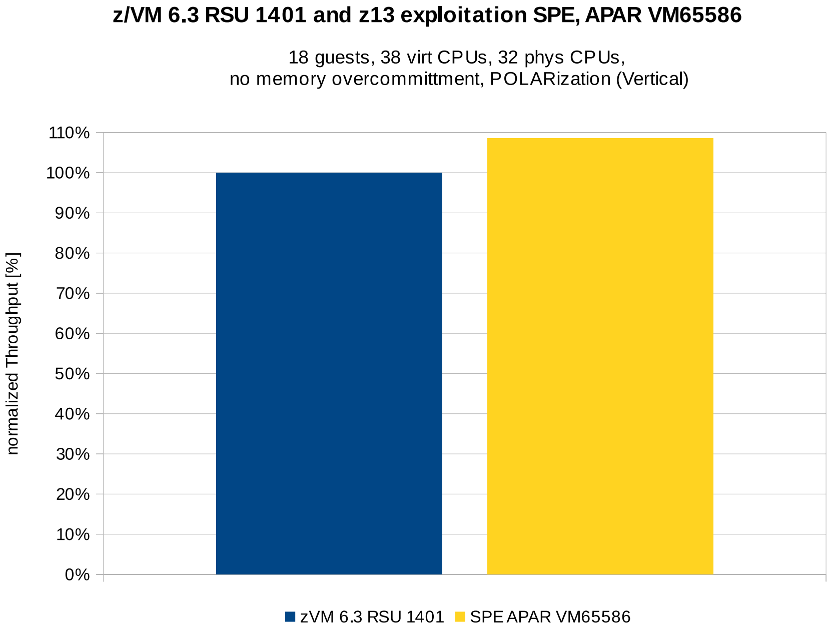

Figure 2. Comparison of total workload throughput for z/VM 6.3 RSU 1401 release with SPE APAR VM65586

The use of the SPE APAR VM65586 improved the throughput of the total workload mix by plus 8.6%.

Figure 3 on page 14 compares the LPAR CPU load of the z/VM 6.3 RSU 1401 release with the SPE APAR VM65586.

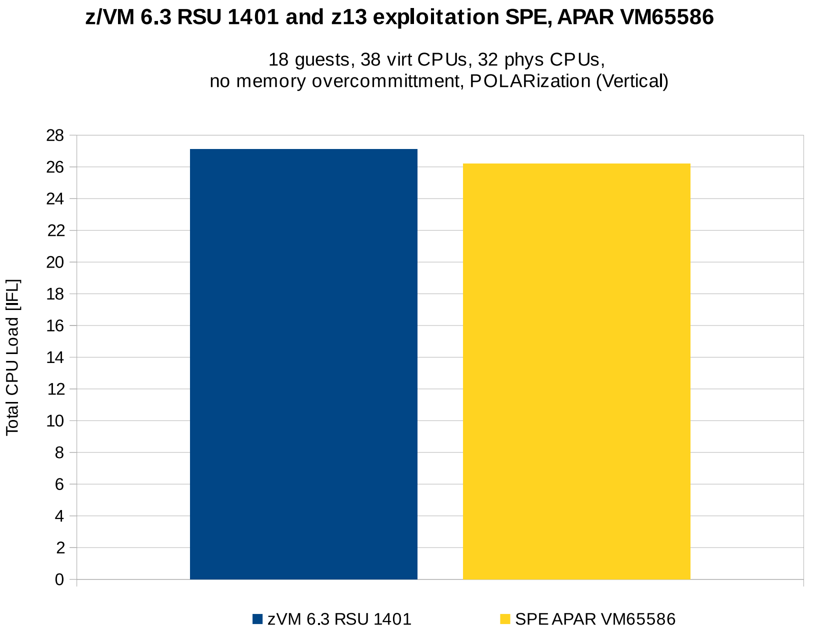

Figure 3. Comparison of LPAR CPU load for z/VM 6.3 RSU 1401 release with SPE APAR VM65586

The use of the SPE APAR VM65586 reduced the CPU load of the total workload mix by 3.5%, which in this case was almost the equivalent of an entire IFL (0.92 IFL).

Conclusion:

The APAR VM65586 provides changes in CPU and scheduler management that are also advantageous for the zEnterprise 196 system, which does not support SMT. We observed a significant increase in throughput and at reduced CPU cost.

SRM polarization and guest CPU waits

The initial reason for that study was the observation in regard to CPU waits reported from the USTAT report for the used guests.

The CPU wait values for the guests that were used were added together, and the test was repeated four times to ensure the statistical relevance of the results. Figure 4 on page 15 shows the CPU waits reported from the USTAT report for the used guests.

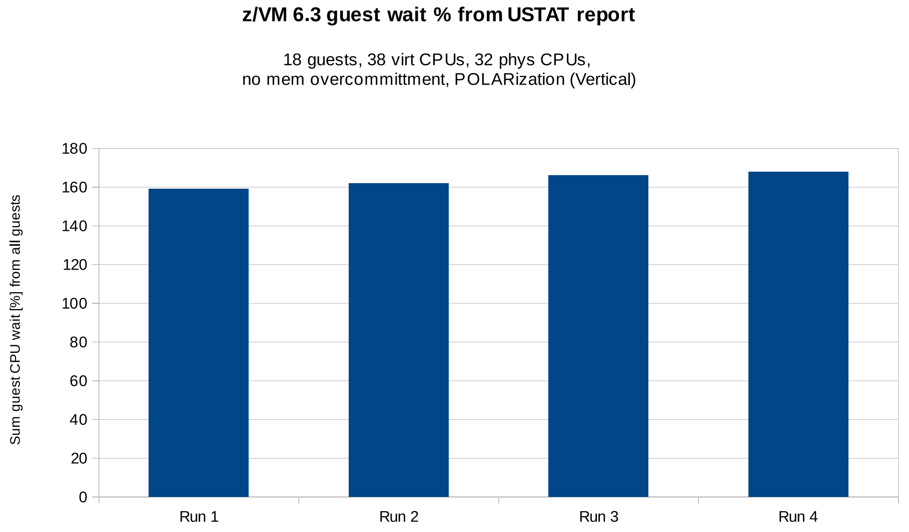

Figure 4. Sum of CPU wait values from perfKit report USTAT for four identical runs without CPU shortage.

Observation:

z/VM reports relatively high values for CPU waits for some guests (Database 10%, File System I/O 19% - 27%, others 1% - 9%). There are no parked CPUs. There is no CPU shortage, the capacity of 2.5 IFLs is still available.

Conclusion:

The initial assumption was that an environment in which the 32 real CPUs are utilized by 80% – 90% (which means that 3 - 6 CPUs are unused) should not show significant waits for CPUs. The occurrence of double-digit percentages for CPU waits in the USTAT report might be an indicator of a problem.

For a more detailed analysis, some of the settings of the z/VM System Resource Manager (SRM) were varied. We found that the polarization parameter was the only parameter that significantly influenced the behavior in regard to waits, and that the observed changes are very significant.

In addition, we changed the VM release to z13 exploitation SPE, APAR VM65586. This change did not change the general behavior, but provided the improvements described in “z/VM 6.3 and the z13 exploitation SPE APAR VM65586” on page 13. Also, it provided the new parameter *guest dispatch wait* (as described in “Guest dispatching metrics” on page 11) for determining the delay time during the dispatching of virtual CPUs to real CPUs. The results for the throughput are shown in Figure 5 on page 16.

The throughput values are normalized to the average throughput values with z/VM 6.3 RSU 1401 and vertical polarization (=100%). For each scenario, four runs were carried out. The vertical error bars show the variation of the values.

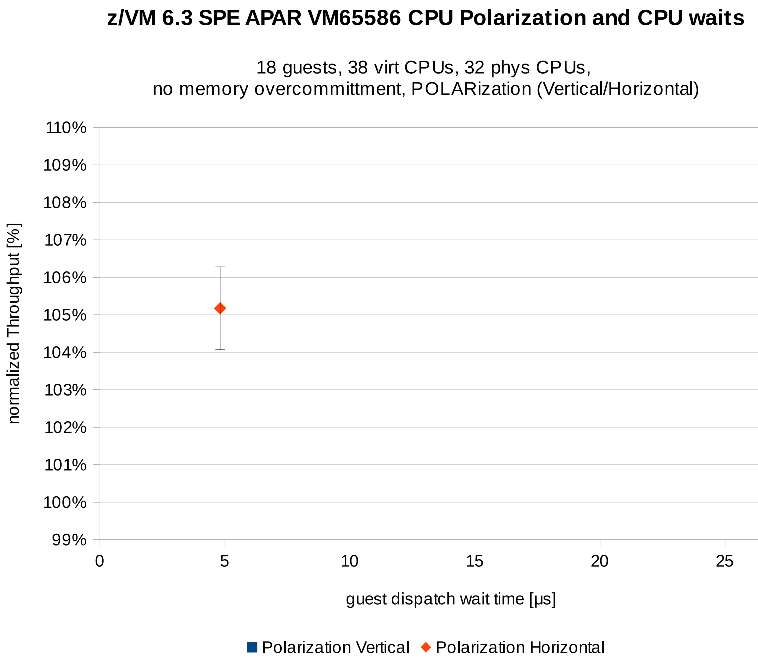

Figure 5. Performance impact on total throughput of vertical (z/VM 6.3 HiperDispatch mode) and horizontal polarization mode (z/VM 6.2 mode)

Observation:

With horizontal polarization the reported guest dispatch wait times decreased significantly to about 5 µs (minus 80%), but at the same time the throughput decreased by 3%.

Conclusion:

When running with vertical polarization, in order to get a better cache-hit ratio z/VM attempts to schedule virtual CPUs on the same real CPUs where they previously ran. It does so even it has to delay the dispatch until the corresponding CPU is free. Although this causes observable waits, overall throughput did increase. This proves that the vertical dispatch approach does pay off. When running with vertical polarization, the observed guest dispatch waits did not appear to indicate a performance degradation.

Figure 6 on page 17 shows the impact of vertical and horizontal polarization of the LPAR CPU load. For each scenario, four runs were carried out. The vertical error bars shows the variation in values.

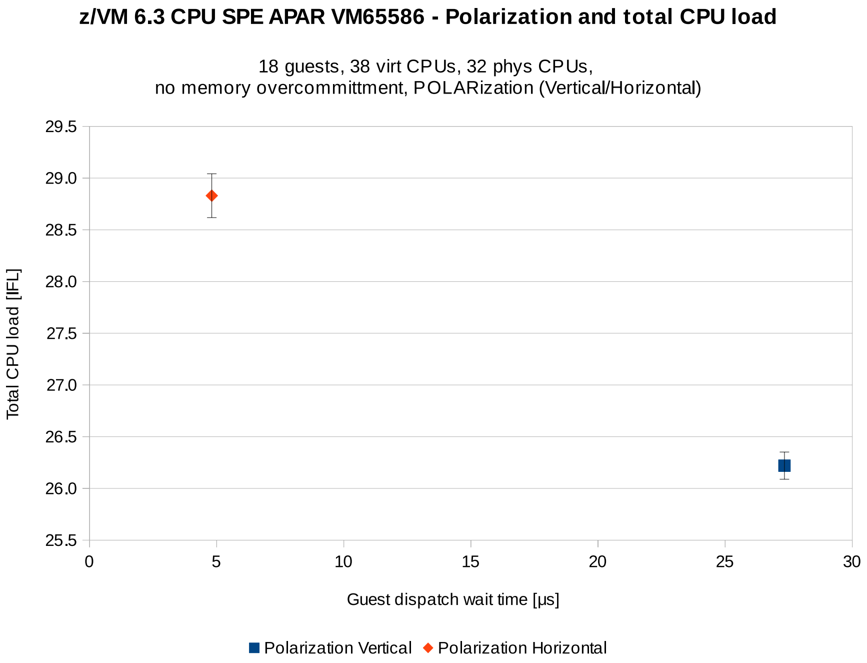

Figure 6. Impact on total CPU load of vertical (z/VM 6.3 HiperDispatch mode) and horizontal polarization mode (z/VM 6.2 mode)

Observation:

The horizontal polarization costs about 2.5 IFLs more at a reduced throughput, but provided a significant reduction in guest dispatch wait times.

Conclusion:

The vertical polarization, introduced with the new z/VM 6.3 HiperDispatch mode, provides real benefits. It provides higher throughput values at significantly lower CPU cost!

When running with vertical polarization, the guest dispatch waits - at least in the observed magnitude – again did not seem to indicate a performance degradation.

The guest dispatch wait values are less than than 30 microseconds. This must be taken into account when judging the observed percent CPU wait samples of up to 20% it the USTAT report.

In order to confirm that the caches are used much more efficiently when data is spread over less CPUs, we also analyzed the cache misses that were reported from CPU MF. Figure 7 on page 18 shows from which cache the L1 misses are resolved.

For a description of the z196 book and Cache structure, see “3.2 Book Design” at: `http://www.redbooks.ibm.com/abstracts/sg247833.html`

For a description of the abbreviations that were used, see “CPU MF (Measurement Facility)” on page 10.

The amount of L1 misses are *the same for both modes*, as shown in Figure 7:

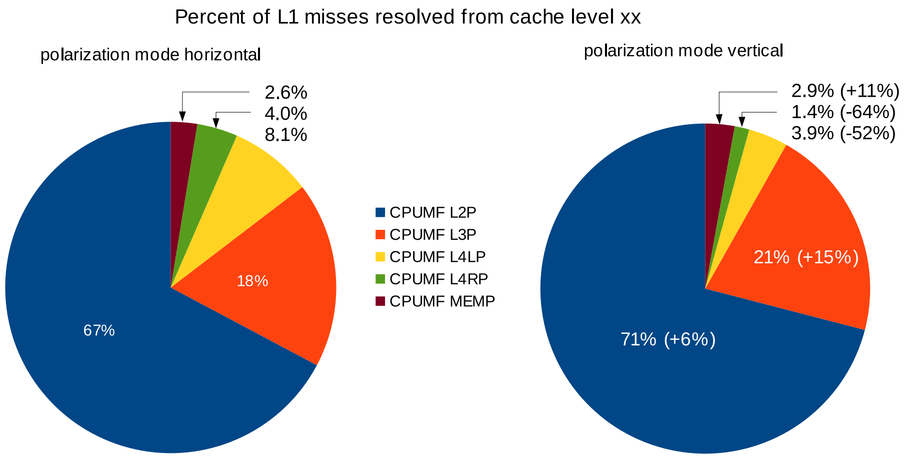

Figure 7. The percentage of the L1 misses resolved from cache level xx for horizontal and vertical polarization mode

Observation:

Using vertical polarization, 6% more misses are resolved from L2, 15% more from L3, and the amount of misses resolved from L4 is reduced more than the half, either local or remote, compared to horizontal polarization. Access to remote L4 caches occurs, because the 32 CPUs are spread over two books. The access to memory increases by 11%, but the percentage value itself is very small. Only 2.9% (or 2.6% for horizontal polarization) is sourced from main memory.

Conclusions:

In both cases 97% of the L1 misses are sourced from a cache. With vertical polarization the processor can source 92% of the L1 cache misses can be sourced from the L2 and L3 caches, whereas with horizontal polarization only 85% can be sourced from the L2 and L3. Also, with horizontal polarization more cache misses can only be sourced from L4, and, even worse, from the L4 caches at remote books.

Given that:

- the lower the cache number is, the faster are the access times,
- memory is the slowest resource, and
- the slower the access the more cycles are needed to serve the request and the slower is the execution,

these caching statistics comprehensively explain the advantage of the vertical polarization in regard to throughput and CPU utilization.

The increase in main memory access can be attributed, at least in part, to the increase in throughput. However, this factor accounts for only 3% approximately of the total result set.

Finally, the data illustrates the considerable advantage of vertical polarization that was introduced with z/VM 6.3 HiperDispatch, and how efficiently the z Systems architecture can handle very large workloads (200GiB, 32 IFLs), when 97% of the L1 misses can be sourced from a cache.

References

View a list of documents referenced in this white paper.

Michael Johanssen, Dr. Juergen Doelle, *z/VM 6.3 Resource Overcommitment* (2014):

`http://www.ibm.com/developerworks/linux/linux390/perf/tuning_vm.html#reoc`

Understanding z/VM HiperDispatch:

`http://www.vm.ibm.com/perf/tips/zvmhd.html`

Using CPU Measurement Facility Host Counters:

`http://www.vm.ibm.com/perf/tips/cpumf.html`

How to Collect CPU MF Information on z/VM:

`http://www.vm.ibm.com/perf/tips/cpumfhow.html`

IBM zEnterprise 196 Technical Guide:

`http://www.redbooks.ibm.com/abstracts/sg247833.html`

z/VM 6.3 Performance Toolkit Reference, SC24-6210-03, accessible from the IBM Library Server:

`http://publib.boulder.ibm.com/cgi-bin/bookmgr/download/HCSL7C20.pdf`

DayTrader - a more complex application, Apache Foundation, accessible from the Apache Geronimo Website:

`http://geronimo.apache.org/GMOxDOC22/daytrader-a-more-complex-application.html`

Jens Axboe, *fio readme*, accessible from the Git repository:

`https://github.com/axboe/fio`

Dominic Giles, *SwingBench 2.2 Reference and User Guide* (stable version 2.4.0.845 - Updated: 2011-12-08), accessible from:

`http://www.dominicgiles.com/swingbench.html`

uperf - A network performance tool, Performance Applications Engineering group at Sun Microsystems Documentation and download:

`http://www.uperf.org/`

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing 2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This 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 may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Software Interoperability Coordinator, Department 49XA
3605 Highway 52 N
Rochester, MN 55901
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of 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.

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

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information in softcopy, the photographs and color illustrations may not appear.

Trademarks

IBM, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at http://www.ibm.com/legal/copytrade.shtml

Adobe is either registered trademarks or trademark of Adobe Systems Incorporated in the United States, and/or other countries.

Java™ and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Linux is a registered 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.

Terms and conditions

Permissions for the use of these publications is granted subject to the following terms and conditions.

Personal Use: You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative works of these publications, or any portion thereof, without the express consent of the manufacturer.

Commercial Use: You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of the manufacturer.

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any data, software or other intellectual property contained therein.

The manufacturer reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by the manufacturer, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

THE MANUFACTURER MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THESE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

Printed in USA