The zEC12 Racehorse – a Linux Performance Update

Mario Held, System Performance Analyst
IBM Germany Lab Boeblingen

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

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

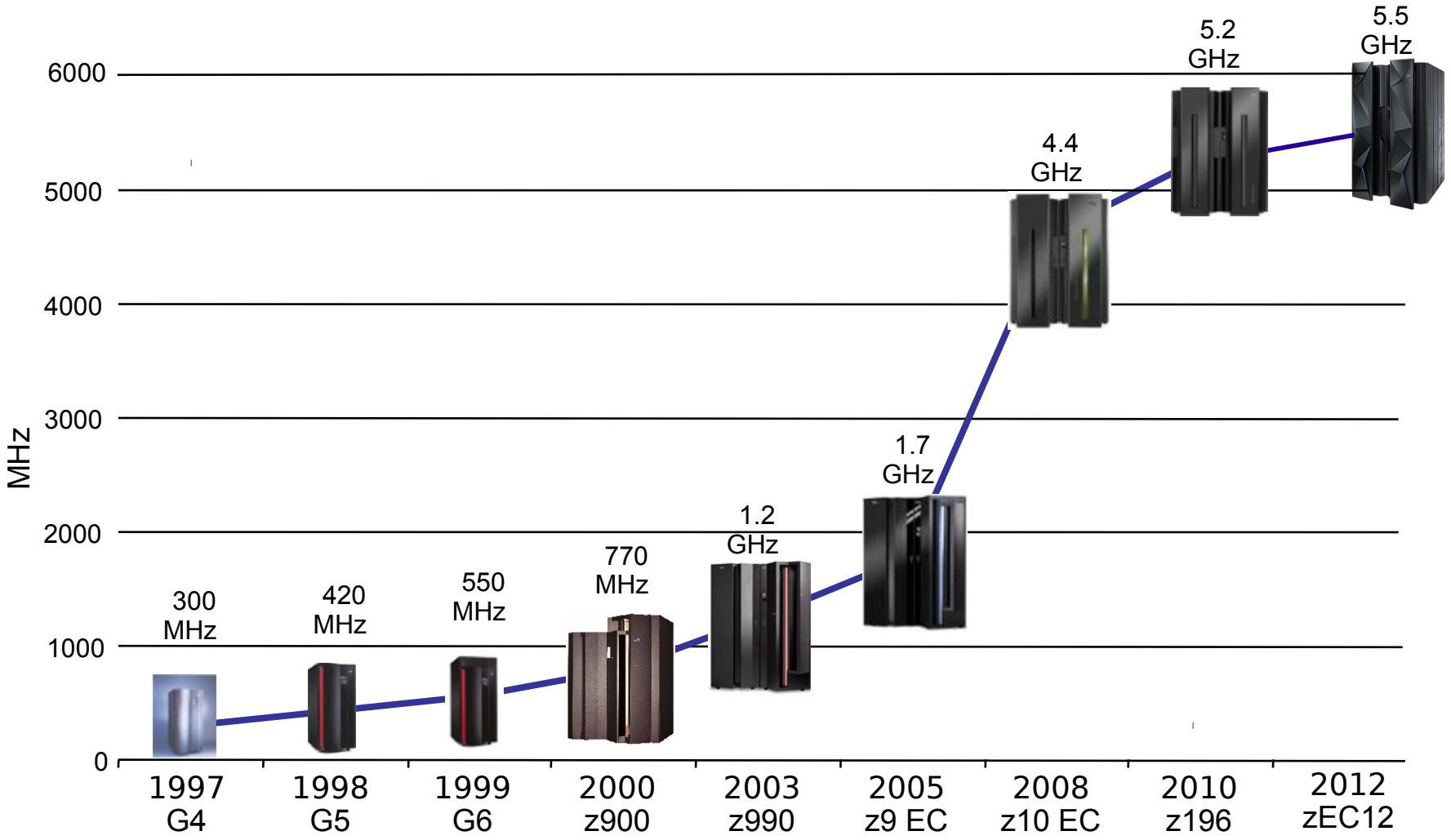
Java is a registered trademarks of Oracle and/or its affiliates.

Other product and service names might be trademarks of IBM or other companies.

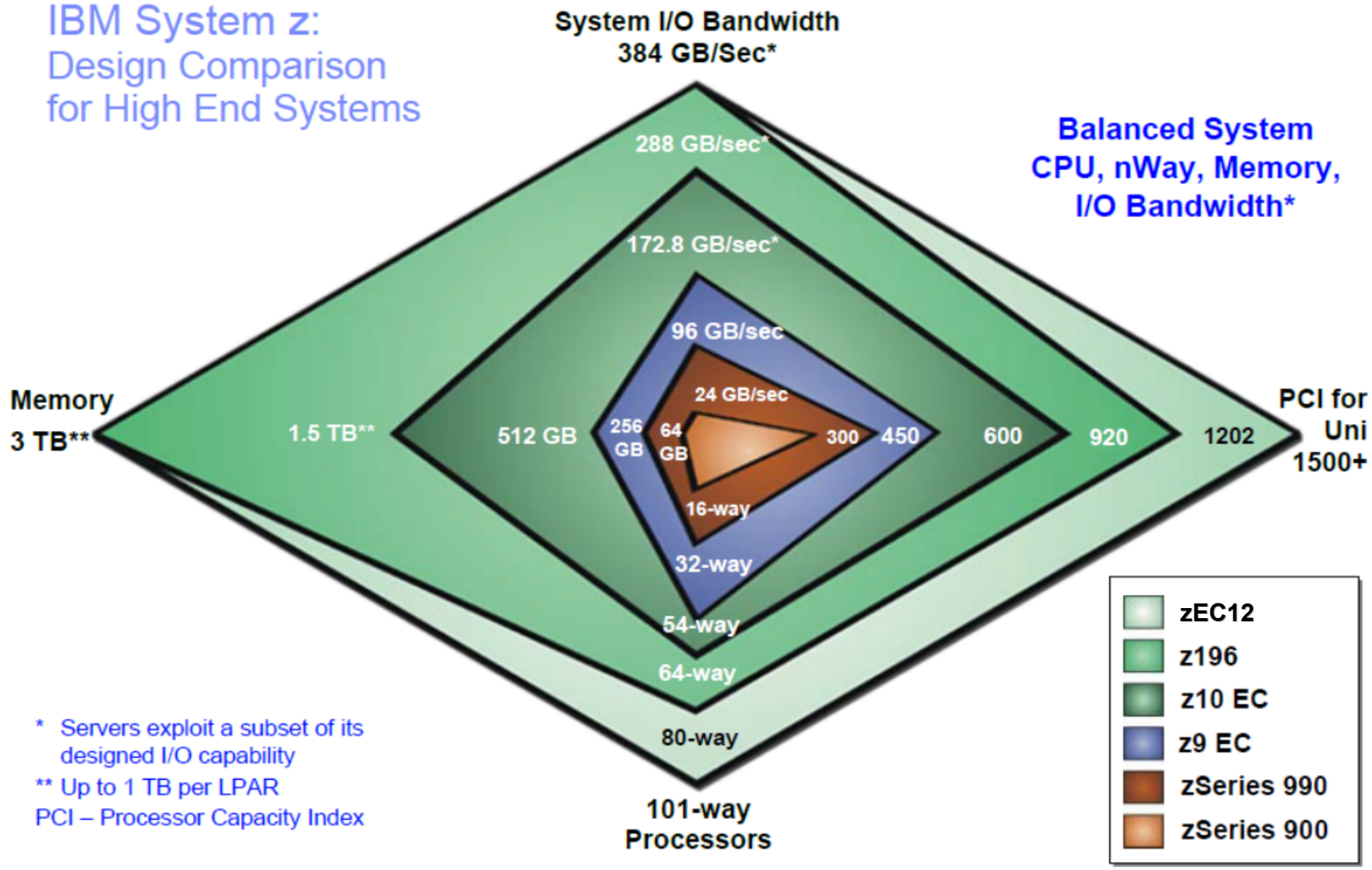
Agenda

- zEnterprise EC12 design
- Linux performance comparison zEC12 and z196

zEC12 Continues the Mainframe Heritage



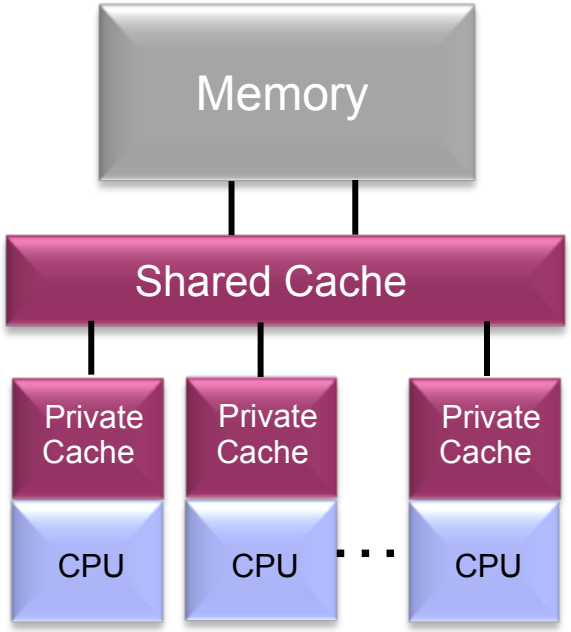
The Evolution of Mainframe Generations



Processor Design Basics

- CPU (core)
  - Cycle time
  - Pipeline, execution order
  - Branch prediction
  - Hardware versus millicode
- Memory subsystem
  - High speed buffers (caches)
  - On chip, on book
  - Private, shared
  - Coherency required
- Buses
  - Bandwidth
- Design Limited by distance, speed of light, and available space

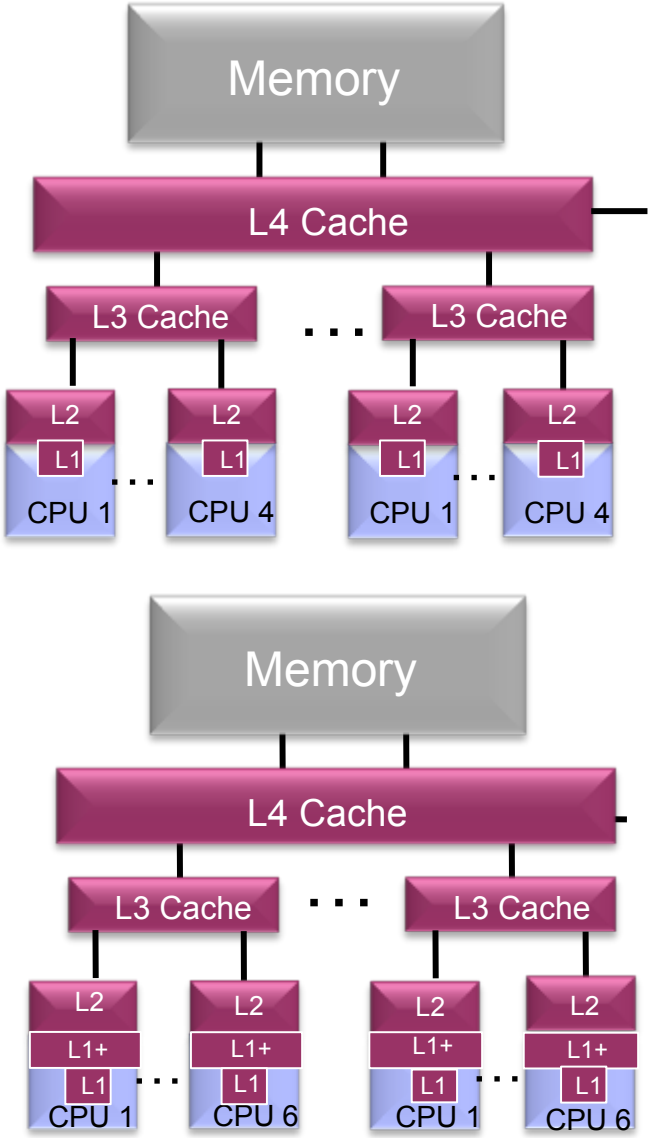
Generic Hierarchy example



zEC12 versus z196 – Memory and Caches

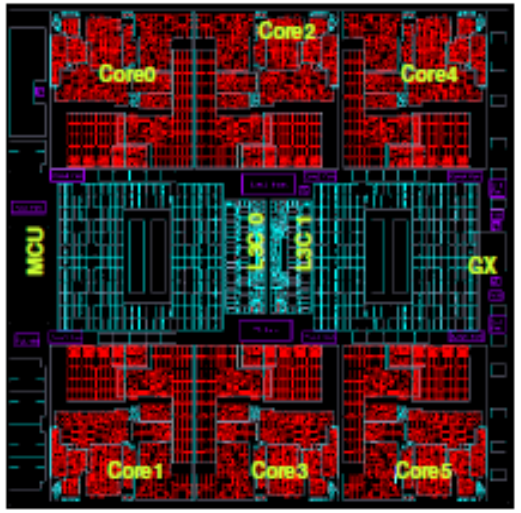
- z196
  - Caches
    - L1 private 64k instr, 128k data
    - L2 private 1.5 MiB
    - L3 shared 24 MiB per chip
    - L4 shared 192 MiB per book

- zEC12
  - Caches
    - L1 private 64k instr, 96k data
    - L1+ 1 MiB (acts as second level data cache)
    - L2 private 1 MiB (acts as second instruction cache)
    - L3 shared 48 MiB per chip
    - L4 shared 2 x 192 MiB => 384 MiB per book



zEC12 PU Core

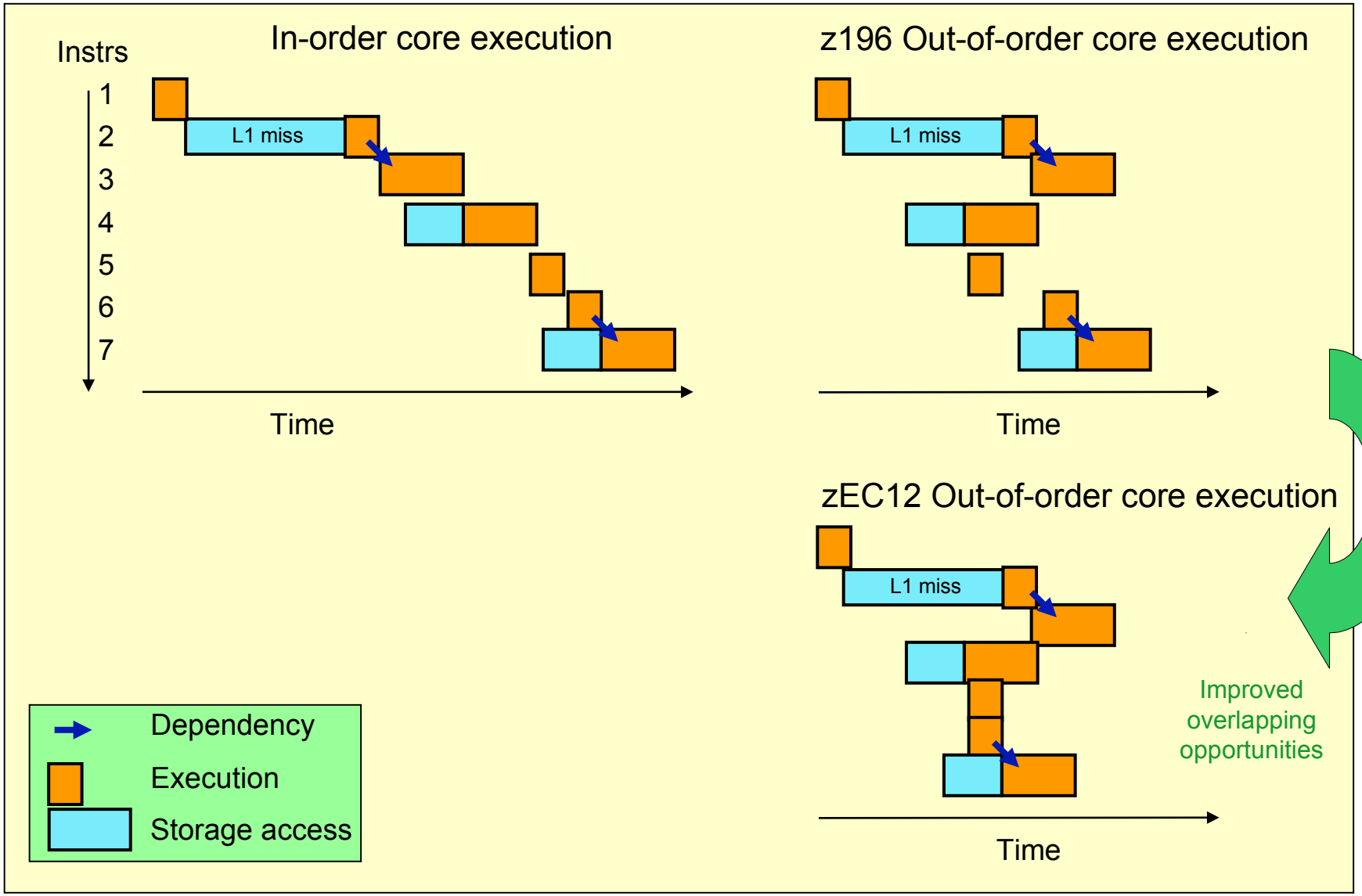
- Each core is a super-scalar processor with these characteristics:
  - Six execution units
    - 2 fixed point (integer), 2 load/store, 1 binary floating point, 1 decimal floating point
  - Up to three instructions decoded / completed per cycle
  - Up to seven instructions issued per cycle
  - New instructions
  - Better branch prediction
    - Virtual branch unit and queue
  - Enhanced out-of-(program)-order (OOO+) capabilities
    - More out of order groups



zEC12 Out of Order – significant performance benefit

- Re-ordering instruction execution
  - Instructions stall in a pipeline because they are waiting for results from a previous instruction or the execution resource they require is busy
  - In an in-order core, this stalled instruction stalls all later instructions in the code stream
  - In an out-of-order core, later instructions are allowed to execute ahead of the stalled instruction
- Re-ordering storage accesses
  - Instructions which access storage can stall because they are waiting on results needed to compute storage address
  - In an in-order core, later instructions are stalled
  - In an out-of-order core, later storage-accessing instructions which can compute their storage address are allowed to execute
- Hiding storage access latency
  - Many instructions access data from storage
  - Storage accesses can miss the L1 and require 10 to 500 additional cycles to retrieve the storage data
  - In an in-order core, later instructions in the code stream are stalled
  - In an out-of-order core, later instructions which are not dependent on this storage data are allowed to execute

Out of Order Execution – z196 versus zEC12



Set the expectations

- Performance relevant changes in these areas
  - Increased clock speed (5.2 GHz to 5.5GHz)
  - Changes in cache
    - Level 1 data cache smaller (128 k → 96k)
    - Level 1+ and Level 2 cache bigger (1.5 MiB → 2MiB)
    - Level 3 and Level 4 now of double size
  - Modern processor with
    - Improved Pipelining and Decoding
    - Better branch prediction
    - Out of Order of the second generation
- The expectation was to see 25 percent performance improvement overall

Agenda

- zEnterprise EC12 design
- Linux performance comparison zEC12 and z196

zEC12 versus z196 Comparison Environment

- Hardware
  - zEC12
    - 2827-789 H89
    - OSA Express 4S 10 GiB and 1 GiB
    - Connected to a DS8870 via FICON Express 8S
  - z196
    - 2817-766 M66
    - OSA Express 4S 10 Gib and 1 Gib
    - Connected to a DS8800 via FICON Express 8S

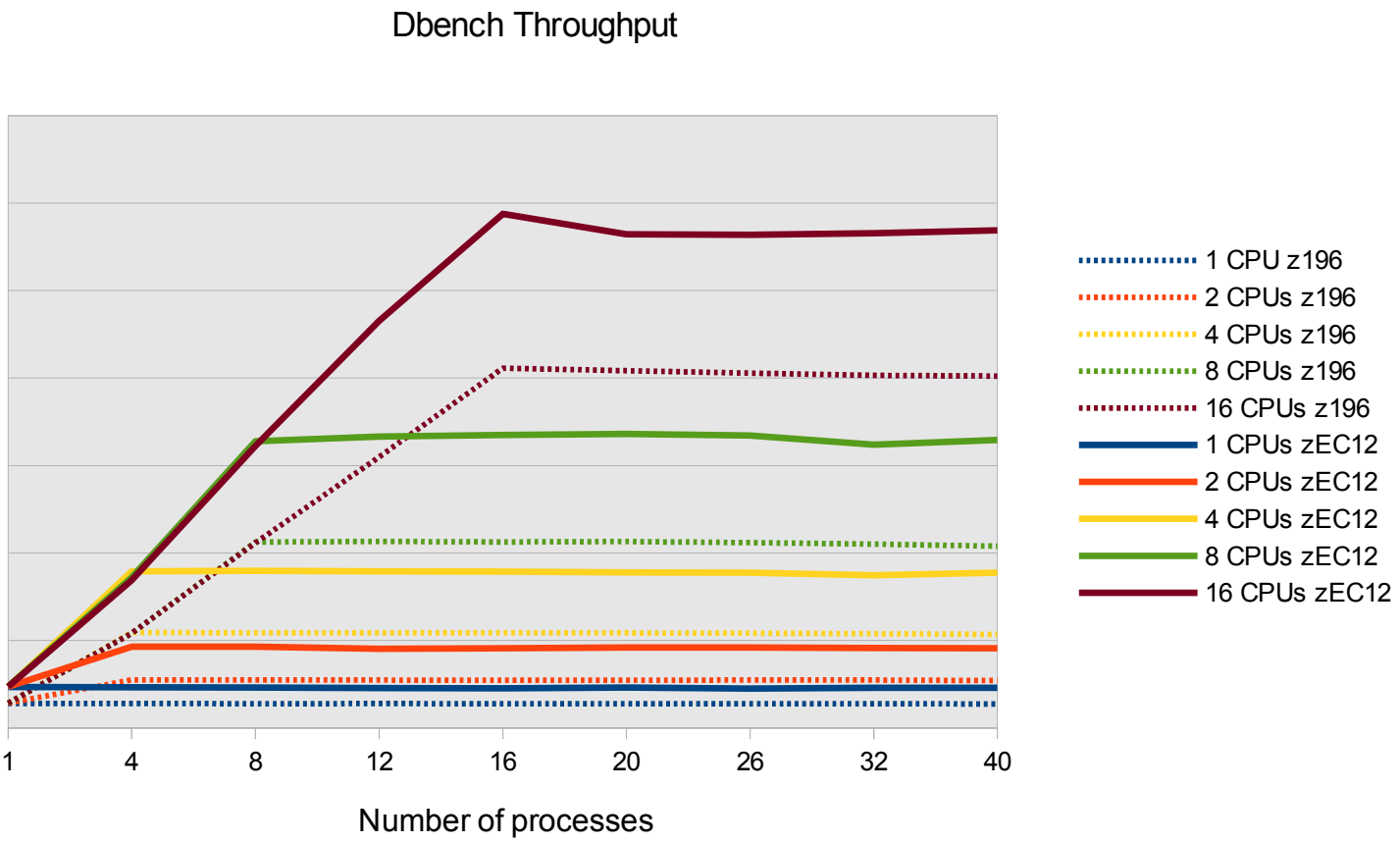
- Recent Linux distribution with kernel
  - SLES11 SP2: 3.0.13-0.27-default (if not stated otherwise)
  - Shared processors (if not stated otherwise)
  - Linux in LPAR exclusively measured
    - Other LPARs deactivated

File Server Benchmark Description

- Dbench 3
  - Emulation of Netbench benchmark
  - Generates file system load on the Linux VFS
  - Does the same I/O calls like the smbd server in Samba (without networking calls)
  - Mixed file operations workload for each process: create, write, read, append, delete
  - Measures throughput of transferred data
- Configuration
  - 2 GiB memory, mainly memory operations
  - Scaling processors 1, 2, 4, 8, 16
  - For each processor configuration scaling processes 1, 4, 8, 12, 16, 20, 26, 32, 40

Dbench3

- Throughput improves by 38 to 68 percent in this scaling experiment, comparing zEC12 to z196



Kernel Benchmark Description

- Lmbench 3
  - Suite of operating system micro-benchmarks
  - Focuses on interactions between the operating system and the hardware architecture
  - Latency measurements for process handling and communication
  - Latency measurements for basic system calls
  - Bandwidth measurements for memory and file access, operations and movement
- Configuration
  - 2 GB memory
  - 4 processors

Lmbench3

- Benefits seen in the very most operations

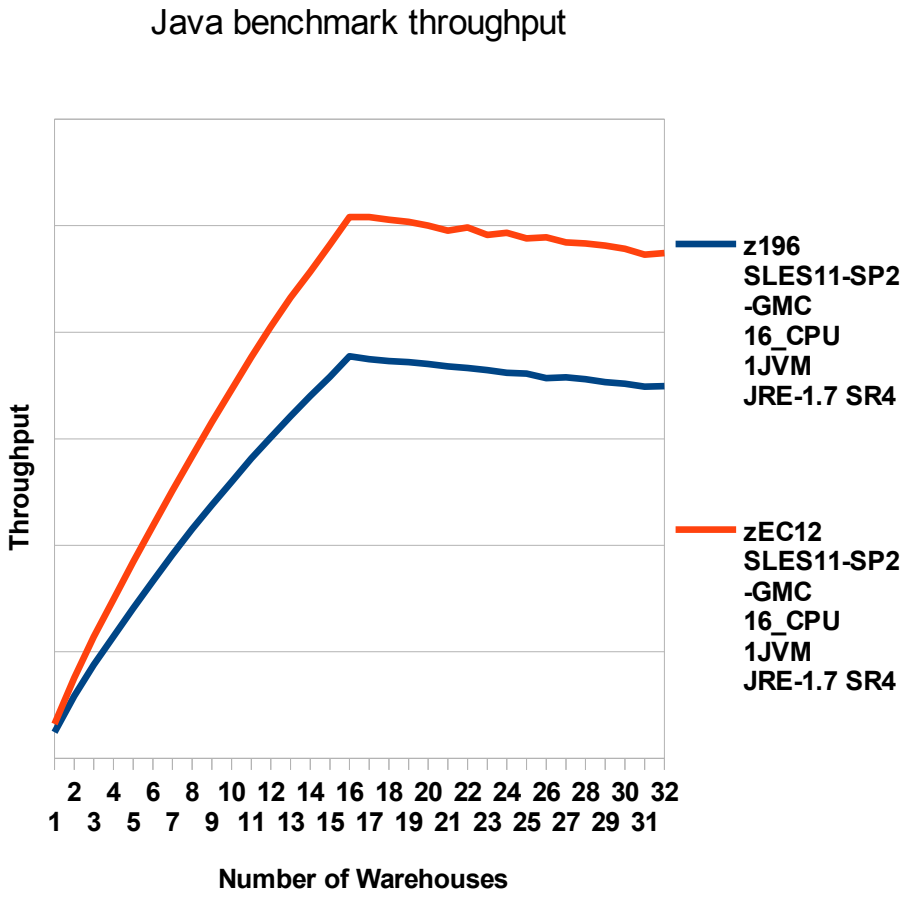
Measured operation	Deviation zEC12 to z196 in %
simple syscall	52
simple read/write	46 /43
select of file descriptors	32
signal handler	55
process fork	25
libc bcopy aligned L1 / L2 / L3 / L4 cache / main memory	0 / 12 / 25 / 10 / n/a
libc bcopy unaligned L1 / L2 / L3 / L4 cache / main memory	0 / 26 / 25 / 35 / n/a
memory bzero L1 / L2 / L3 / L4 cache / main memory	40 / 13 / 20 / 45 / n/a
memory partial read L1 / L2 / L3 / L4 cache / main memory	-10 / 25 / 45 / 105 / n/a
memory partial read/write L1 / L2 / L3 / L4 cache / main memory	75 / 75 / 90 / 180 / n/a
memory partial write L1 / L2 / L3 / L4 cache / main memory	45 / 50 / 62 / 165 / n/a
memory read L1 / L2 / L3 / L4 cache / main memory	5 / 10 / 45 / 120 / n/a
memory write L1 / L2 / L3 / L4 cache / main memory	80 / 92 / 120 / 250 / n/a
Mmap read L1 / L2 / L3 / L4 cache / main memory	0 / 13 / 35 / 110 / n/a
Mmap read open2close L1 / L2 / L3 / L4 cache / main memory	23 / 18 / 19 / 55 / n/a
Read L1 / L2 / L3 / L4 cache / main memory	60 / 30 / 35 / 50 / n/a
Read open2close L1 / L2 / L3 / L4 cache / main memory	27 / 30 / 35 / 60 / n/a
Unrolled bcopy unaligned L1 / L2 / L3 / L4 cache / main memory	35 / 28 / 60 / 35 / n/a
memory	35 / 13 / 45 / 20 / n/a
mappings	34-41

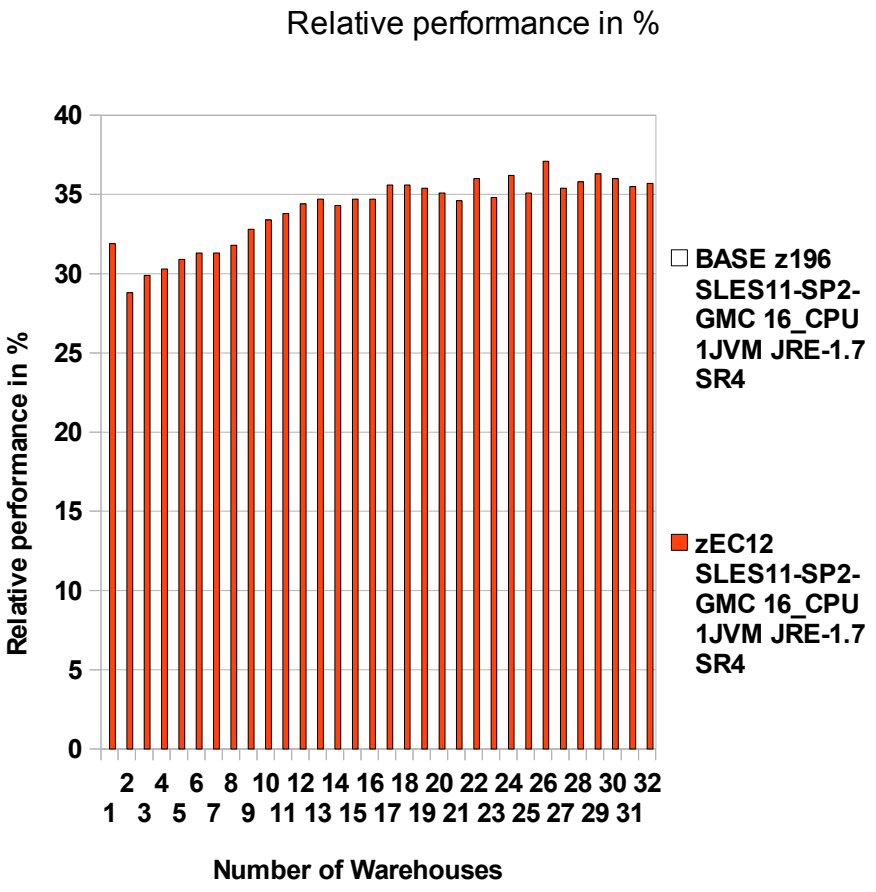
Java Benchmark Description

- Industry standard benchmark
  - Evaluates the performance of server side Java
  - Exercises
    - Java Virtual Machine (JVM)
    - Just-In-Time compiler (JIT)
    - Garbage collection
    - Multiple threads
    - Simulates real-world applications including XML processing or floating point operations
  - Can be used to measure performance of processors, memory hierarchy and scalability
- Configurations
  - 16 processors, 4 GiB memory, 1 JVM, 2GiB max heap size
  - IBM J9 JRE 1.7.0 SR4 64-bit

Java Benchmark

- Business operation throughput improved by approximately 30% to 35%



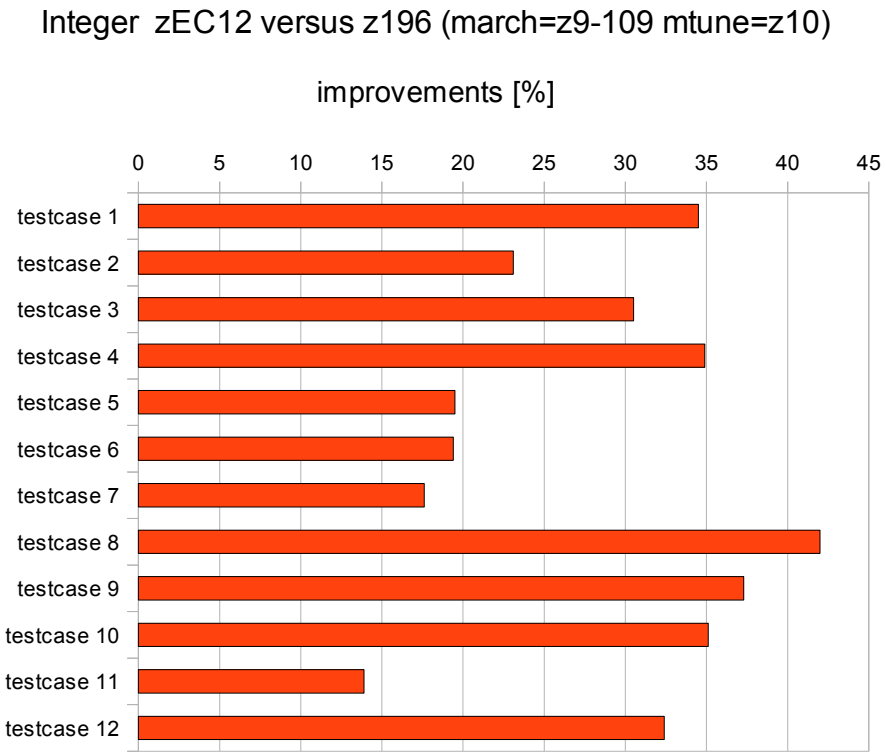


Compute-Intense Benchmark Description

- Industry standard benchmark
  - Stressing a system's processor, memory subsystem and compiler
  - Workloads developed from real user applications
  - Exercising integer and floating point in C, C++, and Fortran programs
  - Can be used to evaluate compile options
  - Can be used to optimize the compiler's code generation for a given target system
- Configuration
  - 1 processor, 2 GiB memory, executing one test case at a time

Single-threaded, compute-intense Workload

- SLES11 SP2 GA, gcc-4.3-62.198, glibc-2.11.3-17.31.1 using default machine optimization options as in gcc-4.3 s390x
- Integer suite 28% more throughput (geometric mean)
- Floating point suite 31% more throughput (geometric mean), not shown in a chart

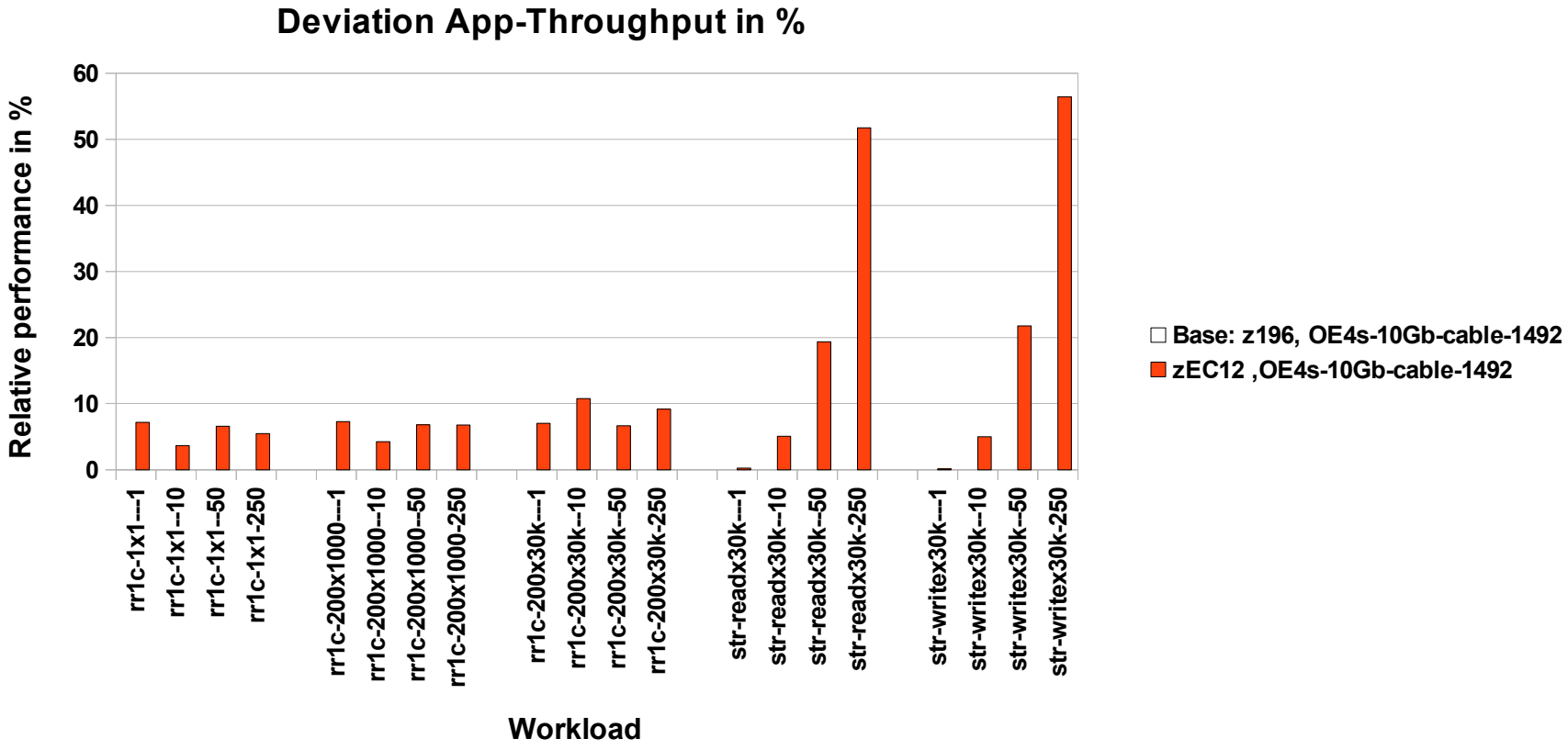


Description Network Benchmark uperf

- Open Source (GPL) network performance tool that supports modeling
- Transactional Workloads
  - rr1c 1 x 1 — Simulating low latency keep-alives
  - rr1c 200 x 1000 — Simulating online transactions
  - rr1c 200 x 30k — Simulating database query
- Streaming workloads
  - str read — Simulating incoming file transfers
  - str write — Simulating outgoing file transfers
- All tests are done with 1, 10, 50, and 250 simultaneous connections
- All that across on multiple connection types
  - LPAR-LPAR / LPAR-z/VM / z/VM-z/VM
  - Physical and virtual connections
  - Different OSA cards and MTU sizes

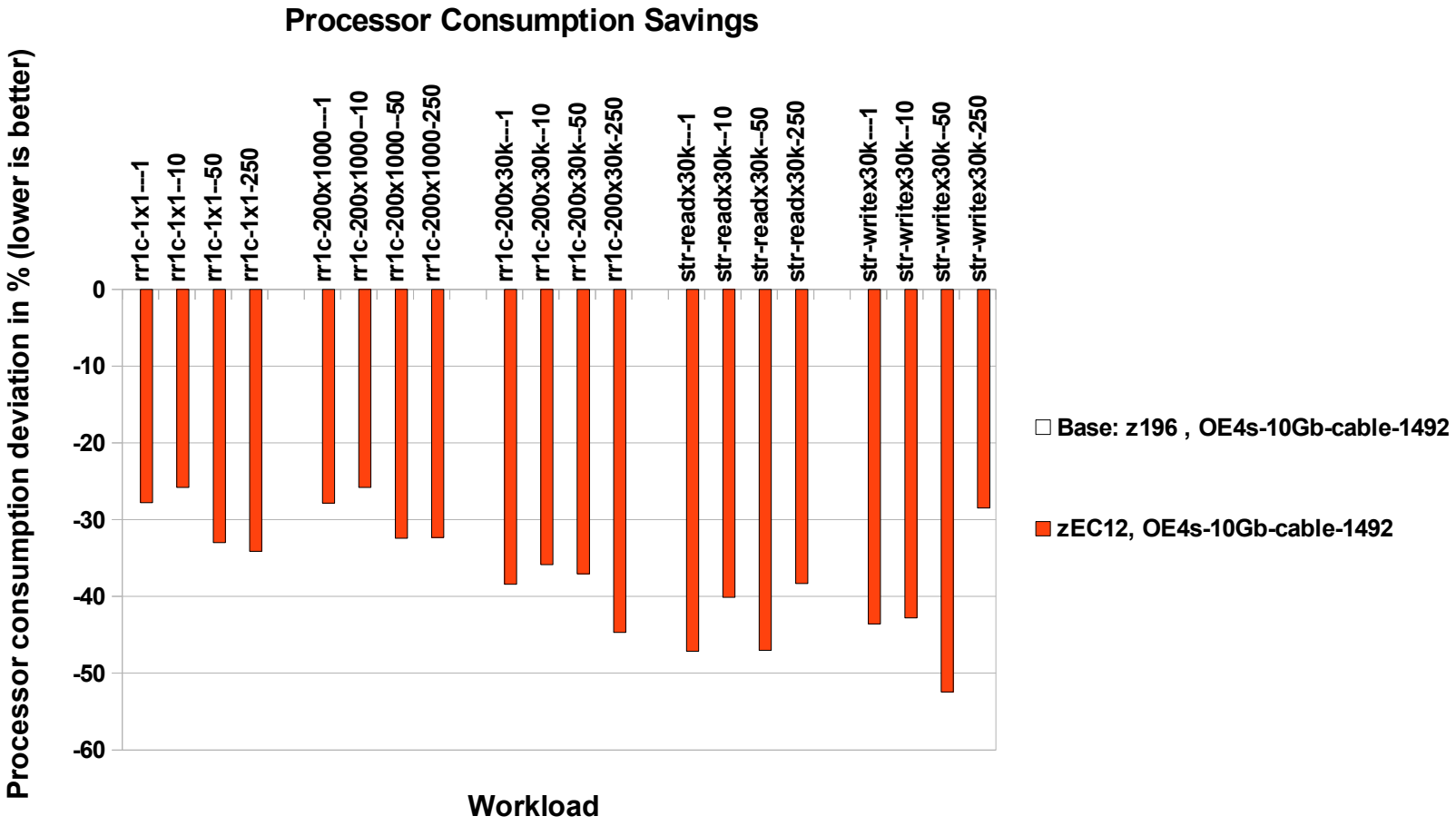
uperf – OSA Express4S 10GiB MTU1492

- Throughput increased
  - Especially with streaming and many parallel connections
- Measured with a SLES-SP2 maintenance kernel (lanidle settings)



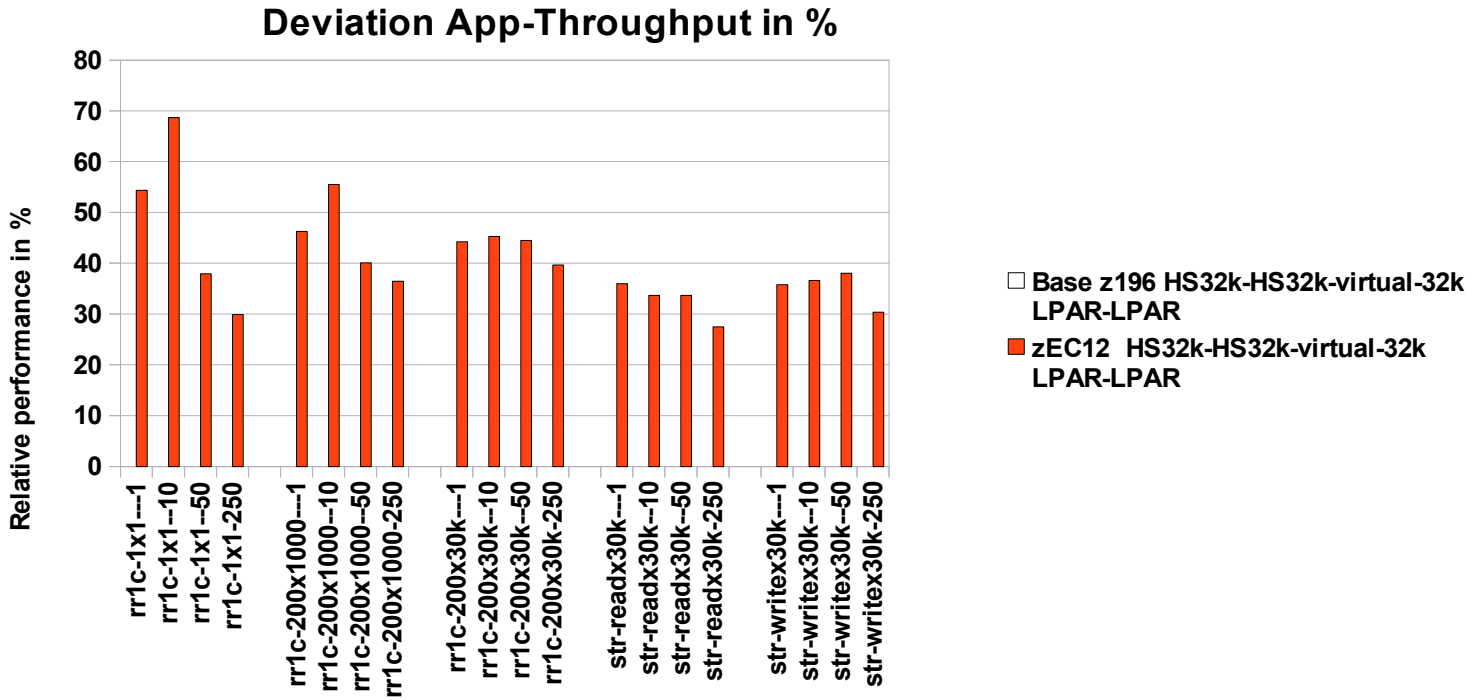
uperf – OSA Express4S 10GiB MTU1492 (cont.)

- Tremendous processor consumption savings
  - Benefit for all kinds of connection characteristics



uperf – Hipersockets LPAR-LPAR MTU32k

- Throughput increased
  - Improvement with all types of connection and sizes measured
- Processor time savings similar as seen with OSA card
- Dedicated processors used in this measurement
- Measured with a SLES11-SP2 maintenance kernel

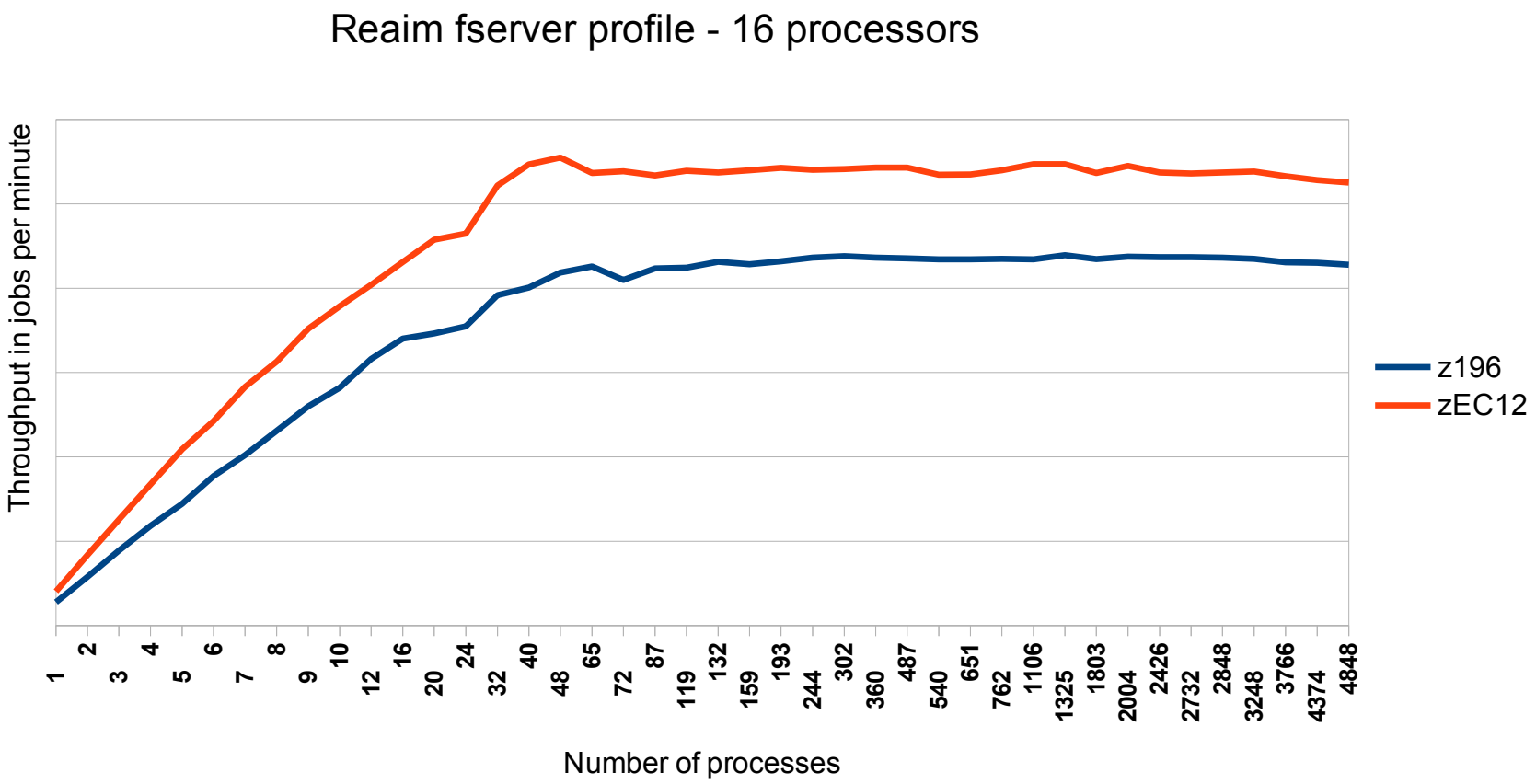


Description – Scalability Benchmark

- Re-Aim 7
  - Open Source equivalent to the AIM Multiuser benchmark
  - Workload patterns describe system call ratios (patterns can be more inter-process-communication, disk or calculation intensive)
  - The benchmark run
    - Starts with one job, continuously increases that number
    - Overall throughput usually increases until #threads ≈ #processors
    - Then threads are further increased until a drop in throughput occurs
    - Scales up to thousands of concurrent threads stressing the same components
  - Often a good check for non-scaling interfaces
    - Some interfaces don't scale at all (1 job throughput ≈ multiple jobs throughput, despite >1 processors)
    - Some interfaces only scale in certain ranges (throughput suddenly drops earlier)
  - Measures the amount of jobs per minute a single thread and all the threads can achieve
  - Configuration
    - 2, 8, 16 processors, 4 GiB memory
    - Using a journaled file system on an xpram device (not be I/O bound)

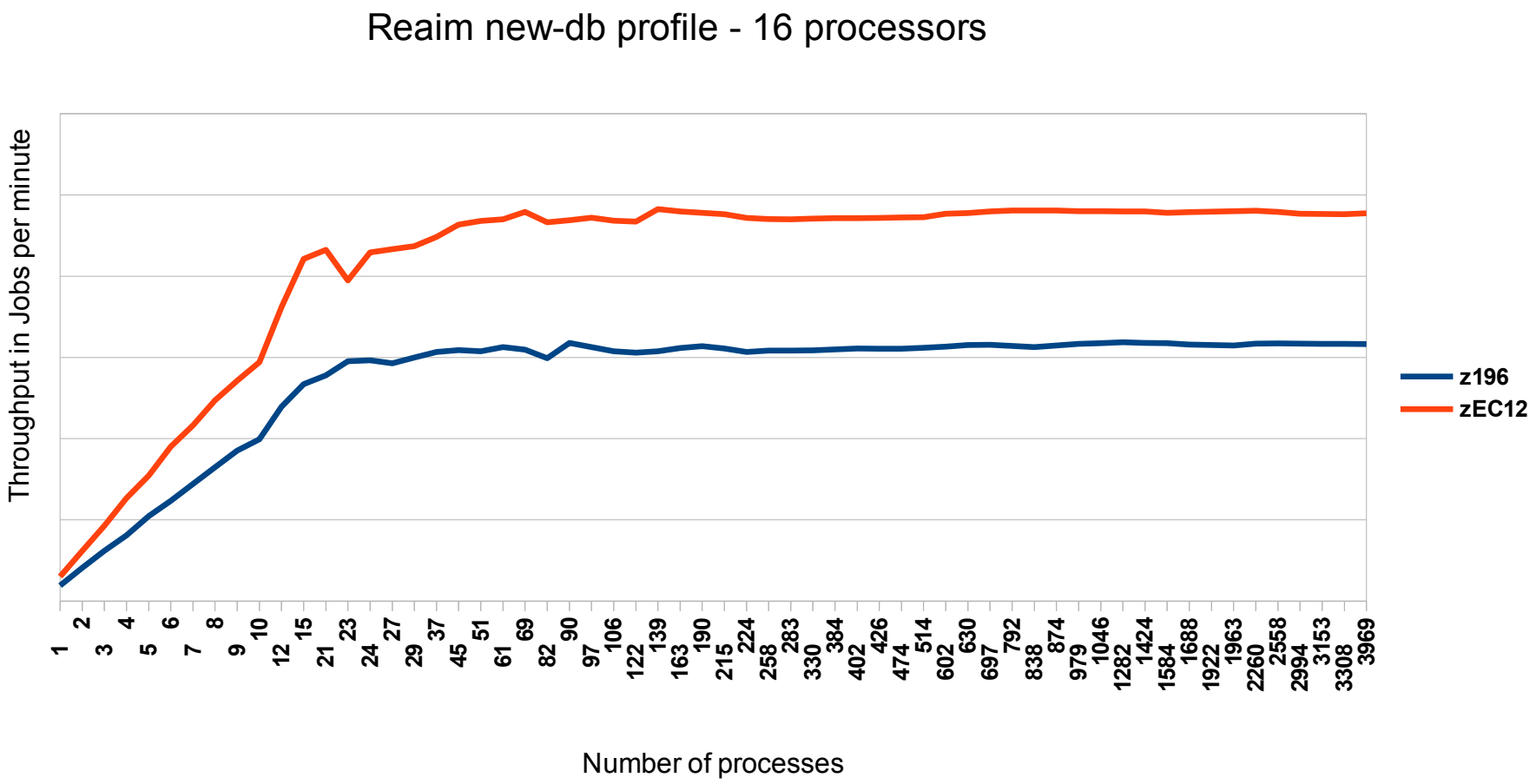
Re-Aim fserver Workload Pattern

- Higher throughput with 4, 8, and 16 processors (25 % to 50 %) at 30 % lower processor consumption



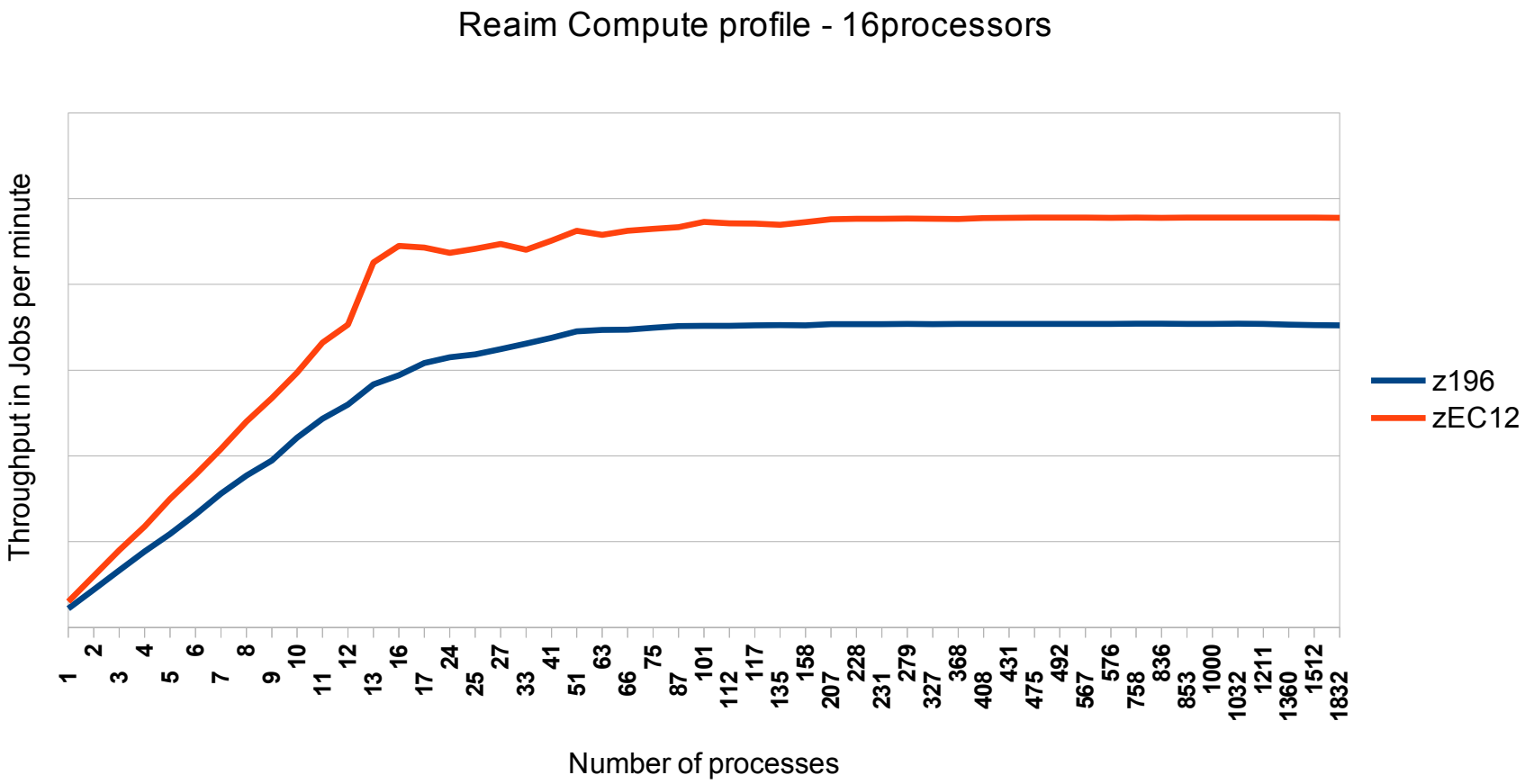
Re-Aim New-DB Workload Pattern

- Higher throughput with 4, 8, and 16 processors (42% to 66 %) at 35 % lower processor consumption



Re-Aim Compute Workload Pattern

- Higher throughput with 4, 8, and 16 processors (25% to 45%) at 20% to 30% percent lower processor consumption

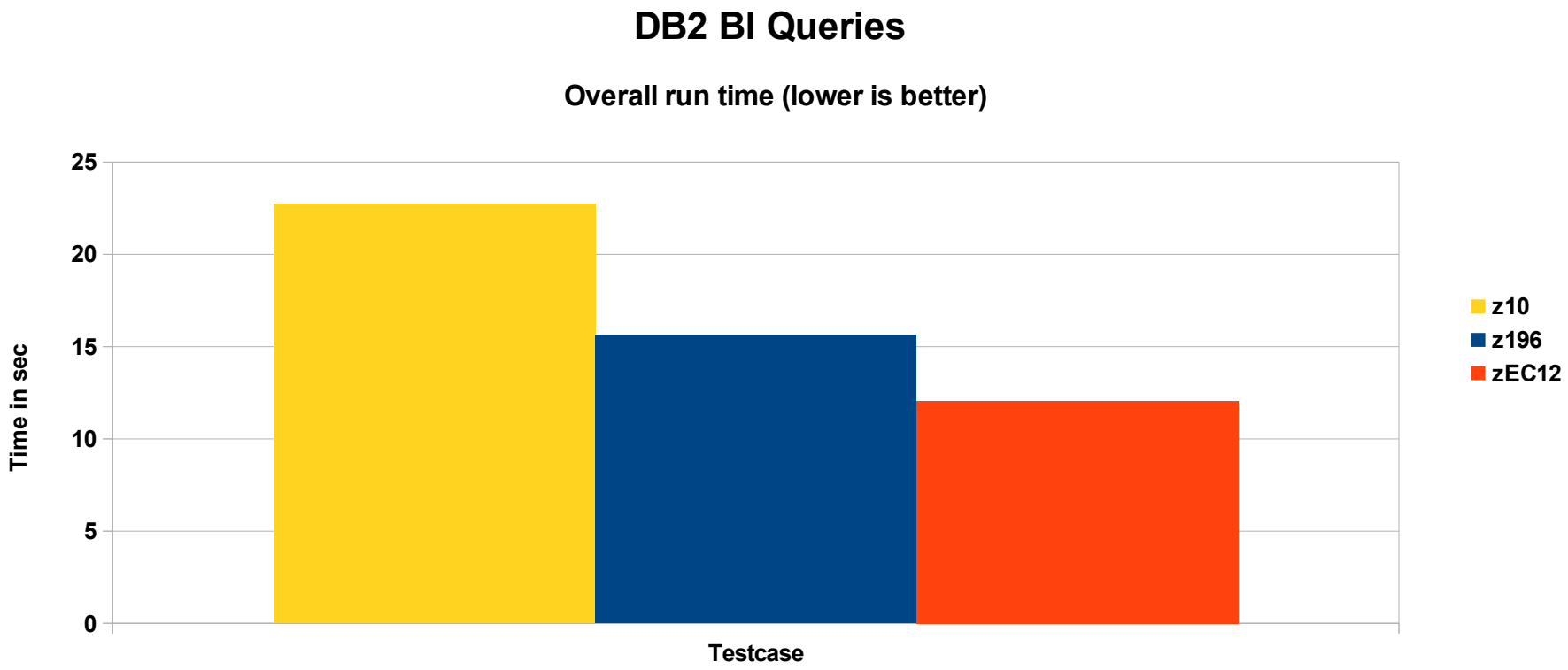


Benchmark Description – DB2 Database BI Workload

- DB2 Query collection (IBM internal)
  - Ten complex database warehouse queries
    - Measured high hit / warm state
  - Provided by the IBM DB2 development (IBM Toronto lab)
- Configuration
  - 16 processors
  - 128 GiB of main memory
  - DB2 10.1 for Linux on System z
  - Not I/O constraint

DB2 Database Complex BI Queries

- 30 percent more throughput comparing zEC12 versus z196
- Close to factor two comparing zEC12 versus z10



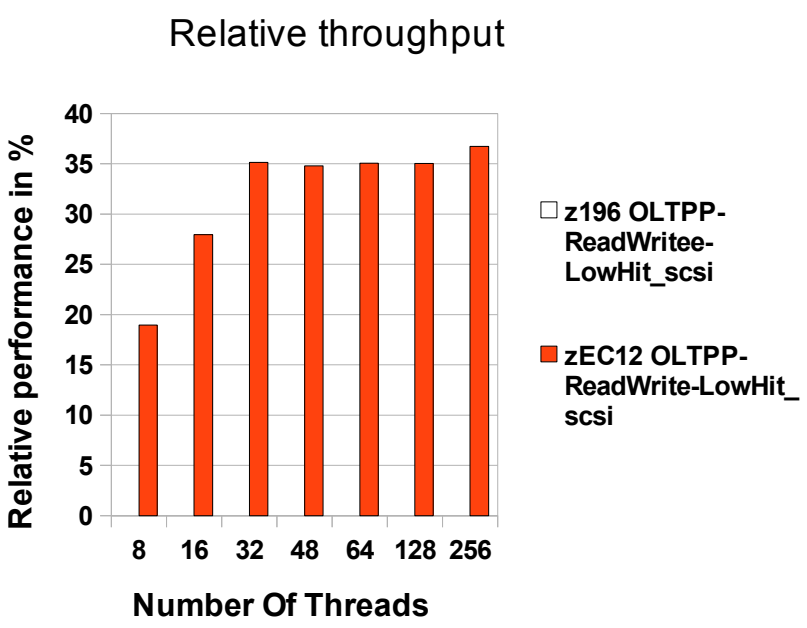
Benchmark Description – SysBench

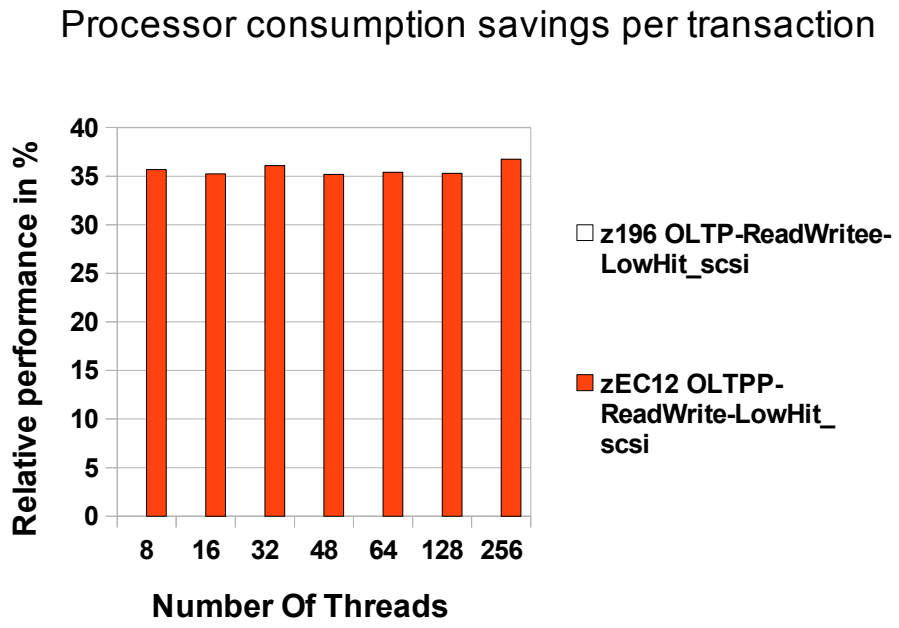
- Scalability benchmark SysBench
  - SysBench is a multi-threaded benchmark tool (among others) for oltp database loads
  - Can be run read-only and read-write
  - Clients can connect locally or via network to the database
  - Database level and tuning is important
    - We use Postgres 9.2.2 with configuration tuned for this workload in our test
  - High/Low Hit cases resemble different real world setup cases with high or low cache hit ratios

Benchmark description – SysBench (cont.)

- Configuration
  - Scaling – read-only load with 2, 8, 16 processors, 8 GiB memory, 4GiB DB (High-Hit)
  - Scaling Net – read-only load with 2, 8, 16 processors, 8 GiB memory, 4GiB DB (High-Hit)
  - Scaling FICON / FCP High Hit ratio – read-write load with 8 processors, 8 GiB memory, 4GiB DB
    - RW loads still need to maintain the transaction log, so I/O is still important despite DB<MEM
  - Scaling FICON / FCP Low Hit ratio – read-write load with 8 processors, 4 GiB memory, 64GiB DB
    - This is also I/O bound to get the data into cache
  - All setups use
    - HyperPAV (FICON) / Mulitpathing (FCP)
    - Disk-spread over the Storage Server as recommended + Storage Pool Striping
    - Extra Set of disks for the WAL (Transaction Protocol)

SysBench – OLTP Read-Write Low-Hit SCSI Test



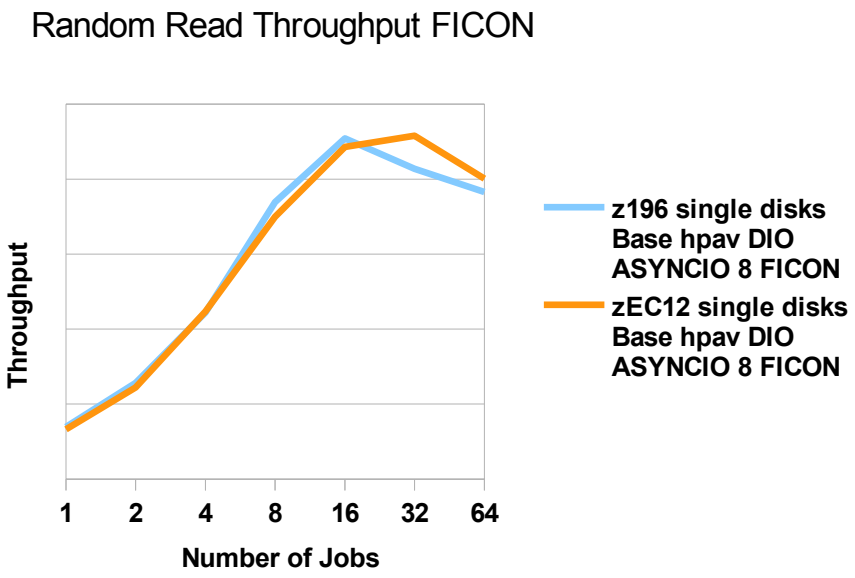


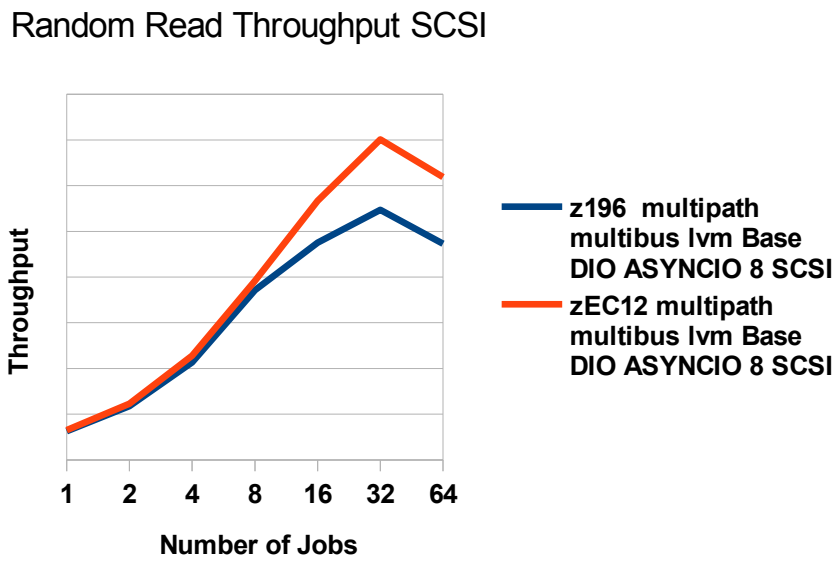
- Overall throughput increased by 35% after ramp-up
- About 35% less processor consumption
- Much lower latencies

Benchmark Description – Disk I/O

- Flexible I/O Testing Tool (FIO)
  - Benchmarking and hardware verification / stress I/O devices
  - Open Source (GPLv2)
  - Easy to customize to individual needs
    - Provides information regarding throughput, latencies, system utilization and much more
- Configuration
  - 8 processors
  - 512 MB main memory
  - z196 connected to DS8800 / zEC12 connected to a DS8870
  - FICON Express 8s
  - 64 single disks, each in FICON and SCSI

FIO – Throughput Random Read DIO ASYNC



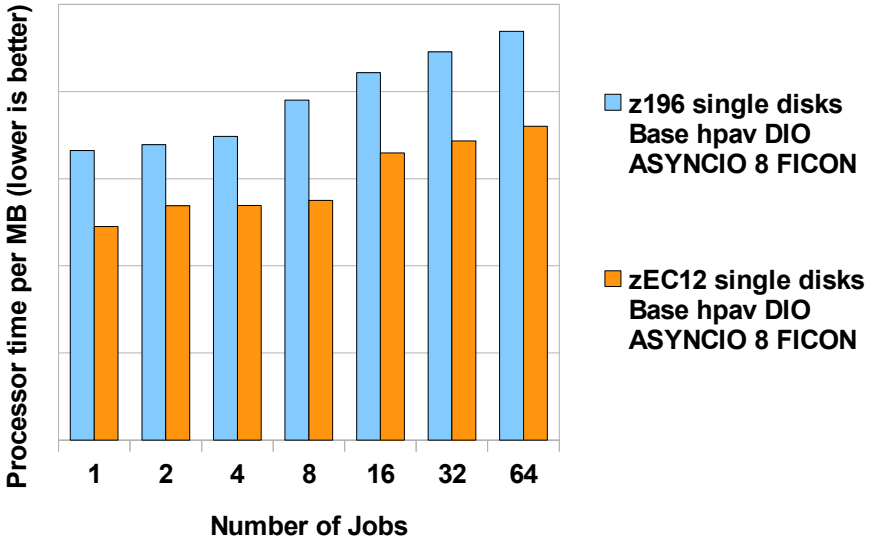


- z196 connected to DS8800, zEC12 connected to DS8870, so we expect increased throughput from newer hardware
  - Bigger caches
  - Faster disks
- Throughput charts intended to show that the same amount of data or even more is transferred during a period of time

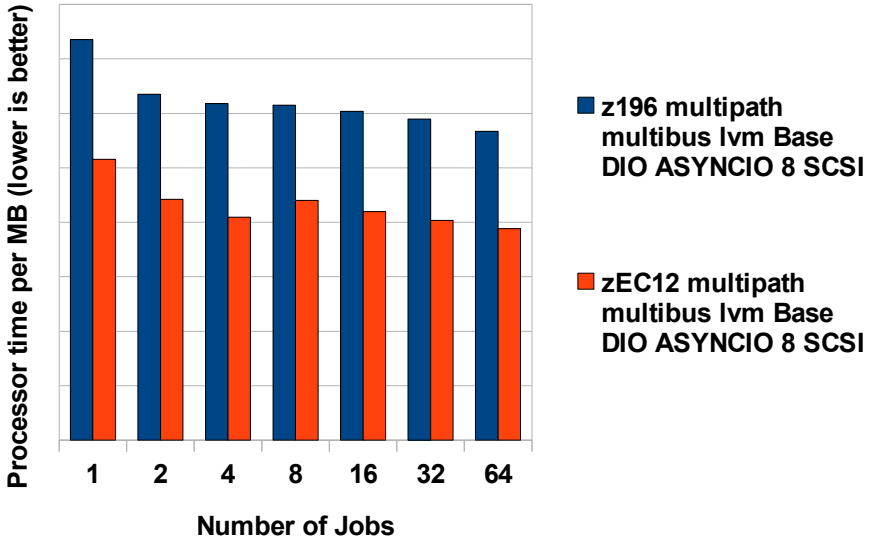
FIO – Processor Consumption Random Read DIO - ASYNC

- zEC12 needs much less processor time to read the same amount of data using random access pattern
  - Savings in the FICON measurement 24% to 28%
  - Savings in the SCSI measurement 29% to 34%

Total processor consumption per MB FICON



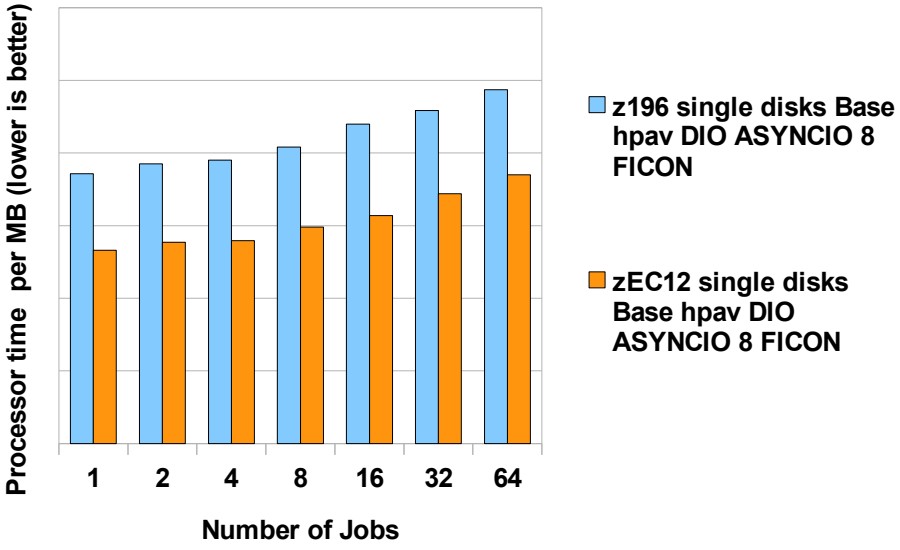
Total processor consumption per MB SCSI



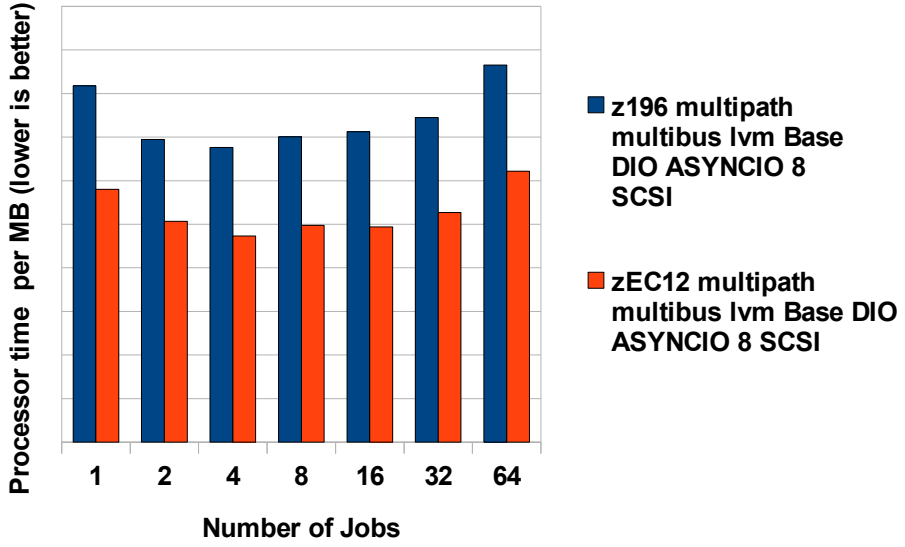
FIO – Processor Consumption Random Write DIO - ASYNC

- zEC12 needs much less processor time to write the same amount of data using random access pattern
  - Savings in the FICON measurement 24% to 29%
  - Savings in the SCSI measurement 28% to 31%

Total processor consumption per MB FICON



Total processor consumption per MB SCSI



Summary

- Tremendous performance gains
  - Performance improvement seen in almost to all areas measured
  - Often combined with processor consumption reduction
  - More improvement than was to be expected just from higher rate
    - Rate is up from 5.2 GHz to 5.5 GHz which means close to 6 percent higher
  - New cache setup with much bigger caches
  - Out-of-order execution of the second generation
  - Better branch prediction
- Some exemplary performance gains with Linux workloads
  - About 30 and up to 35 percent for Java
  - Up to 30 percent for complex database
  - Up to 31 percent for single threaded CPU intense
  - About 38 to 68 percent more throughput when scaling processors and / or processes
  - Processor consumption much lower for disk I/O per transferred unit
  - Increased throughput at lower processor consumption with network

Questions ?

- Further information is located at
  - Linux on System z – Tuning hints and tips
    http://www.ibm.com/developerworks/linux/linux390/perf/index.html
  - Live Virtual Classes for z/VM and Linux
    http://www.vm.ibm.com/education/lvc/

Mario Held

Linux on System z
System Software
Performance Engineer

IBM Deutschland Research & Development
Schoenaicher Strasse 220
71032 Boeblingen, Germany

Phone +49 (0)7031–16–4257
Email mario.held@de.ibm.com