HP OpenVMS Alpha and Integrity Performance Comparison

Session 1225

Gregory Jordan
OpenVMS Engineering Hewlett-Packard

Overview

- The purpose of this presentation is to provide the appropriate expectations of the performance of OpenVMS on Integrity platforms vs. Alpha platforms

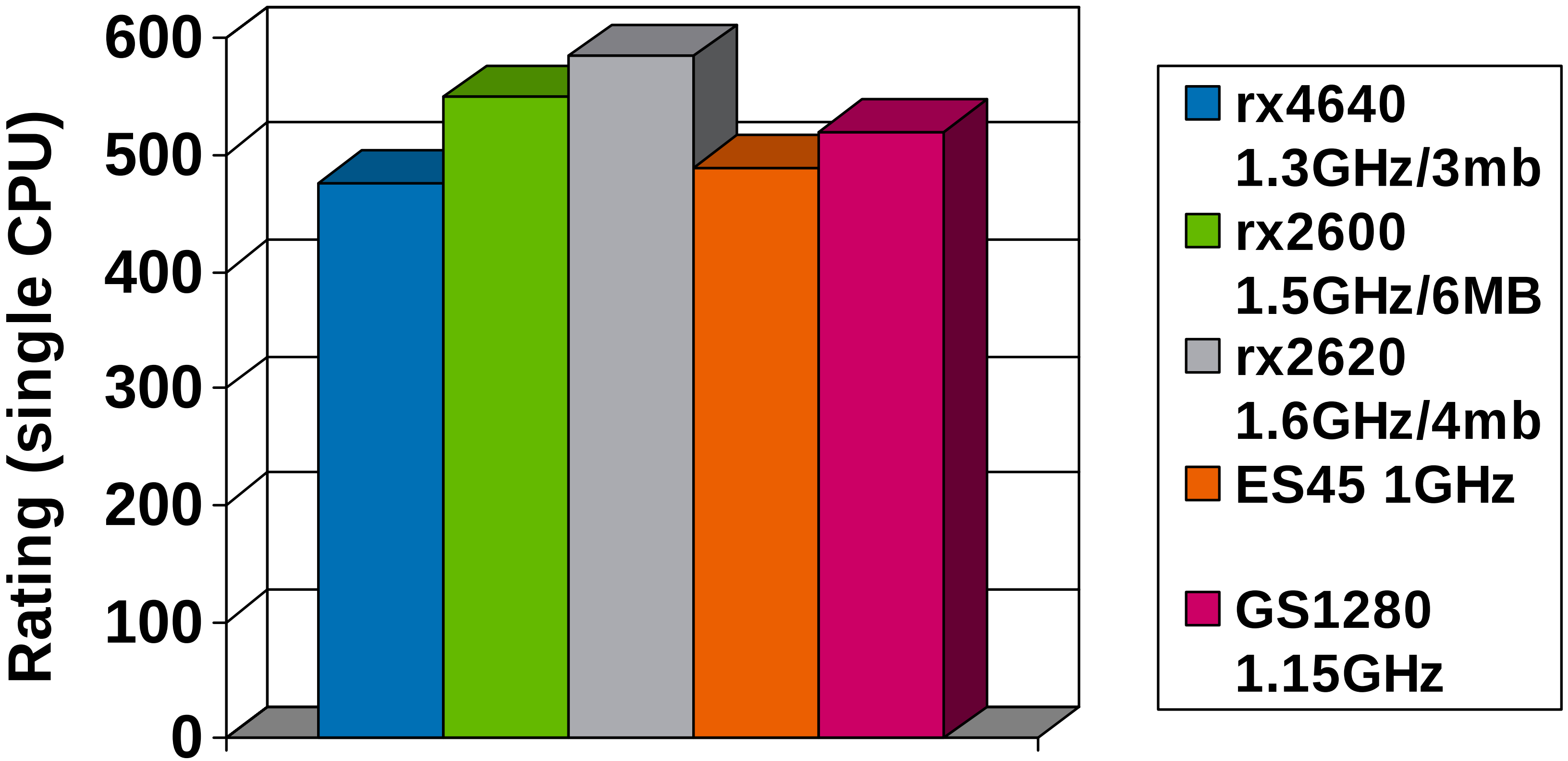
- The performance of various Integrity platforms will be compared with a variety of current Alpha platforms.

Integrity/Alpha Performance Comparison

- The Basics
  - Processors
    - Integer and Floating Point
  - Memory
    - Latency and Bandwidth
  - IO
    - Fibre Channel, Lan
- OpenVMS Performance
  - Various OS Components
  - Improvements in V8.2-1
  - Applications
- Improvement Stories
- Conclusions

CPU – Integer test program

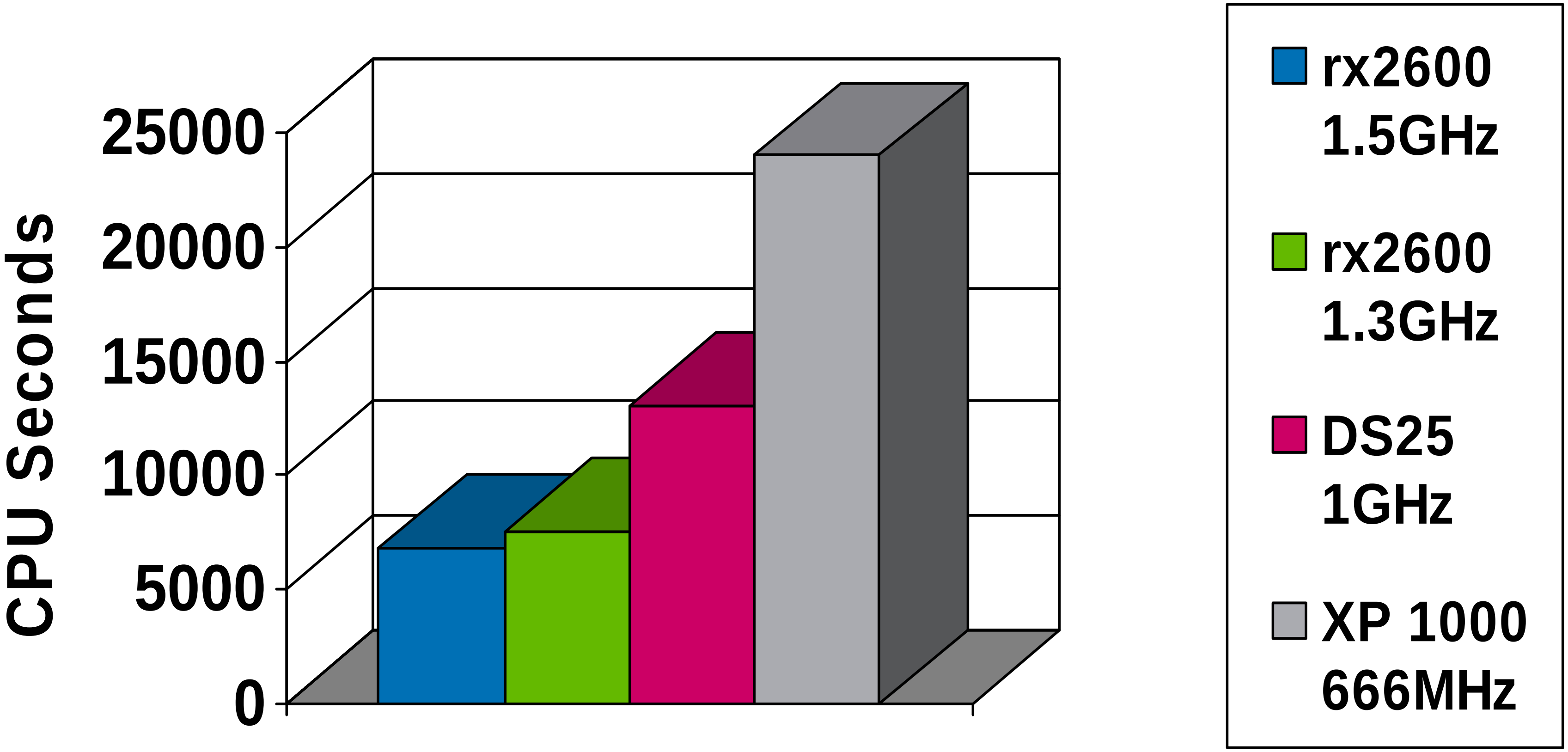
Integer Computations



More is better

SETI

Time to Process a Work Unit



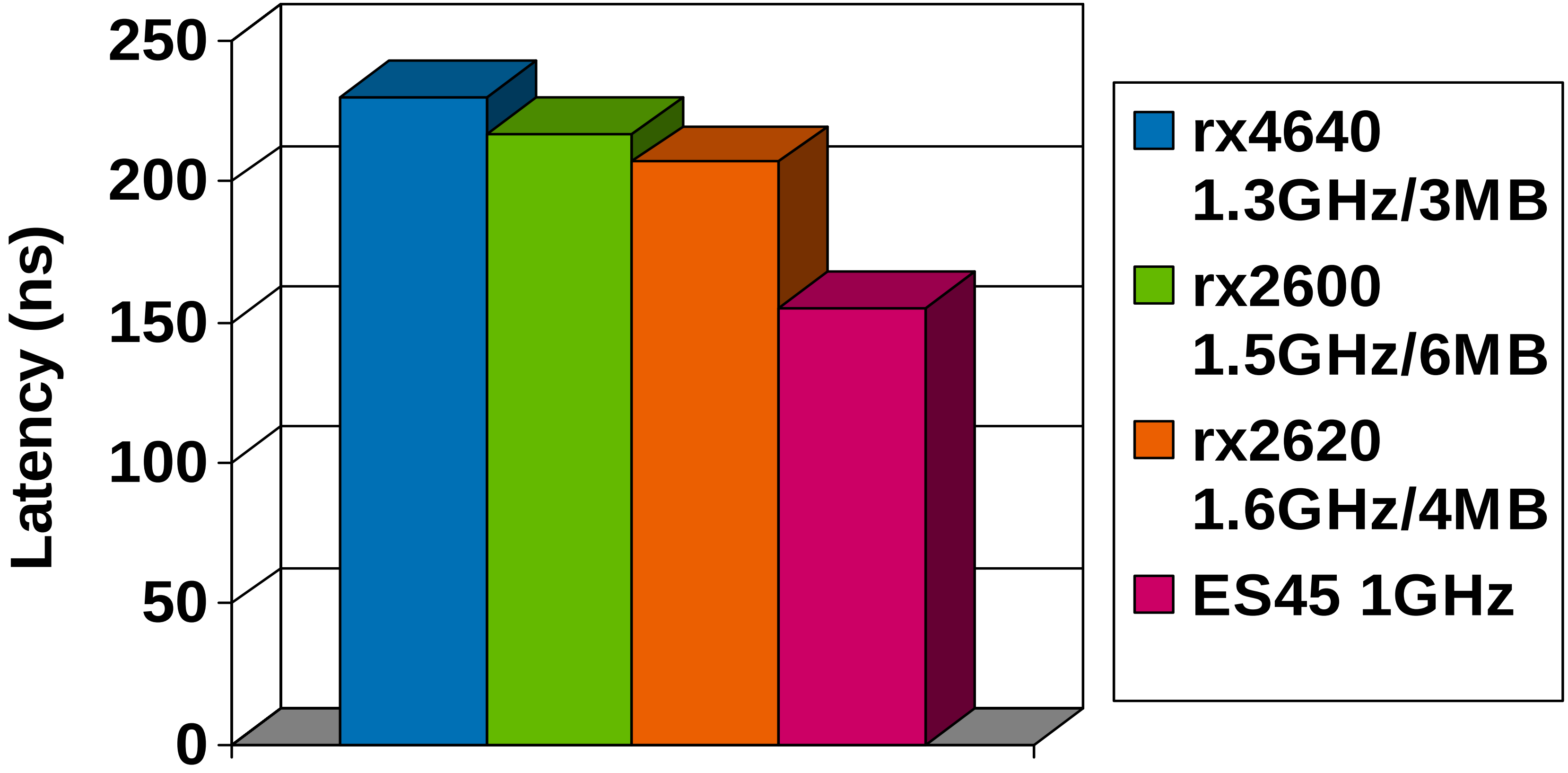
Less is better

Processor Comparison

- The Itanium processors are fast.
  - The current Itanium processors are faster than current Alpha processors
  - Various SPEC benchmarks also show the Itanium processors outperforming Alpha processors

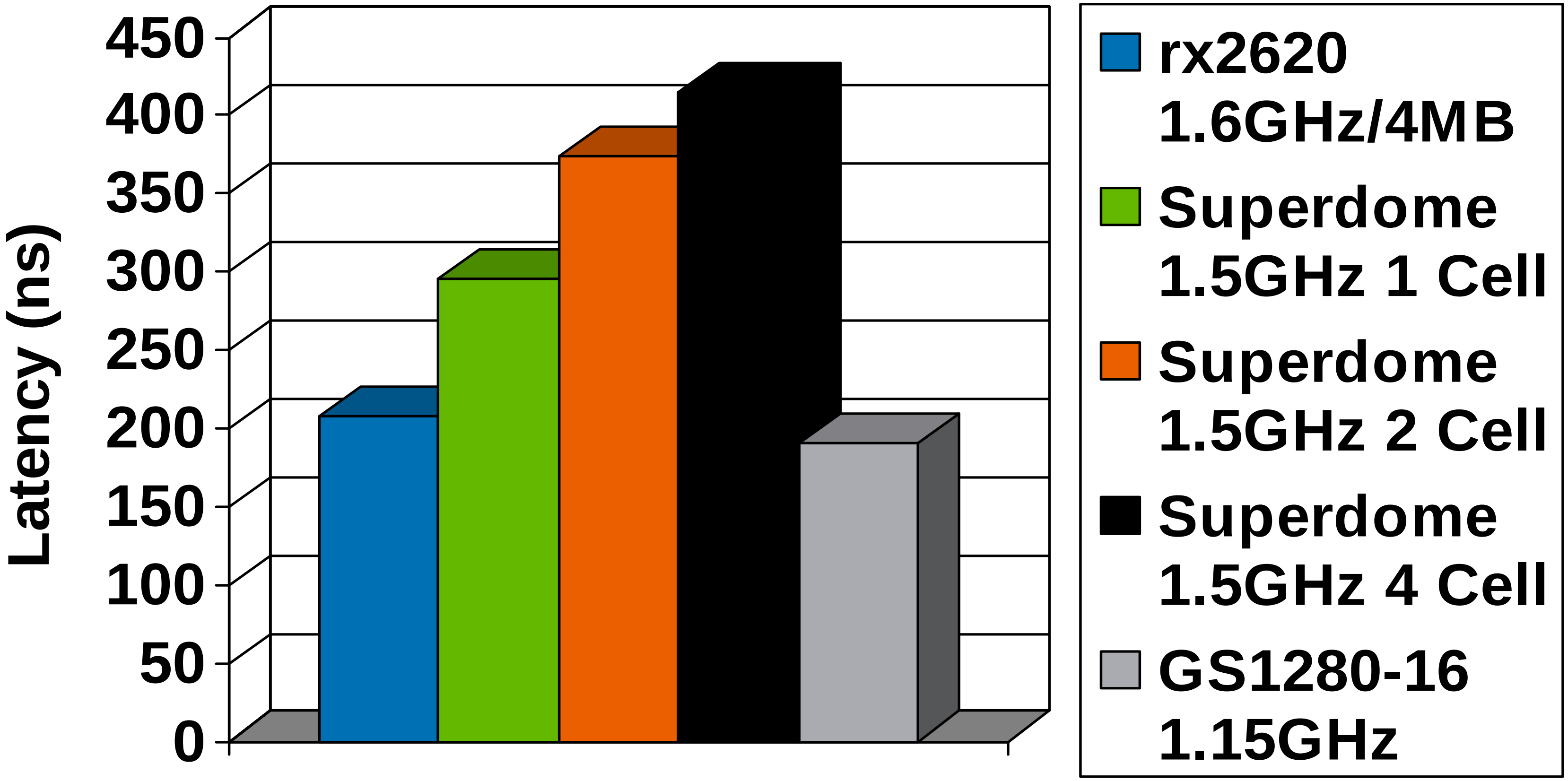
Memory Latency (small servers)

Computed via memory test program



Less is better

Memory Latency (large servers) Computed via memory test program



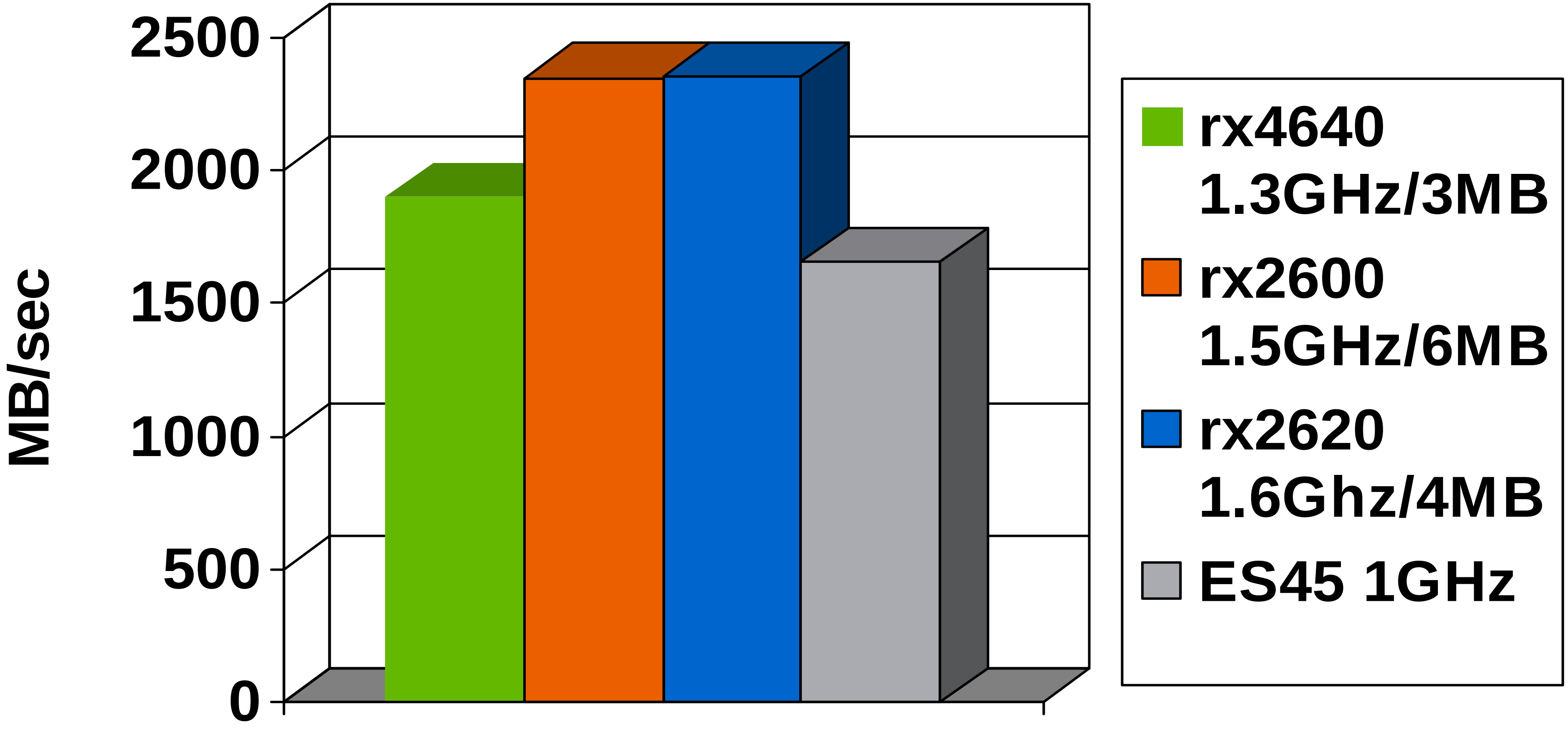
(Superdome memory is interleaved between cells)

Less is better

Memory Bandwidth (small servers)

Computed via memory test program

- MEMSpeed – Test Program

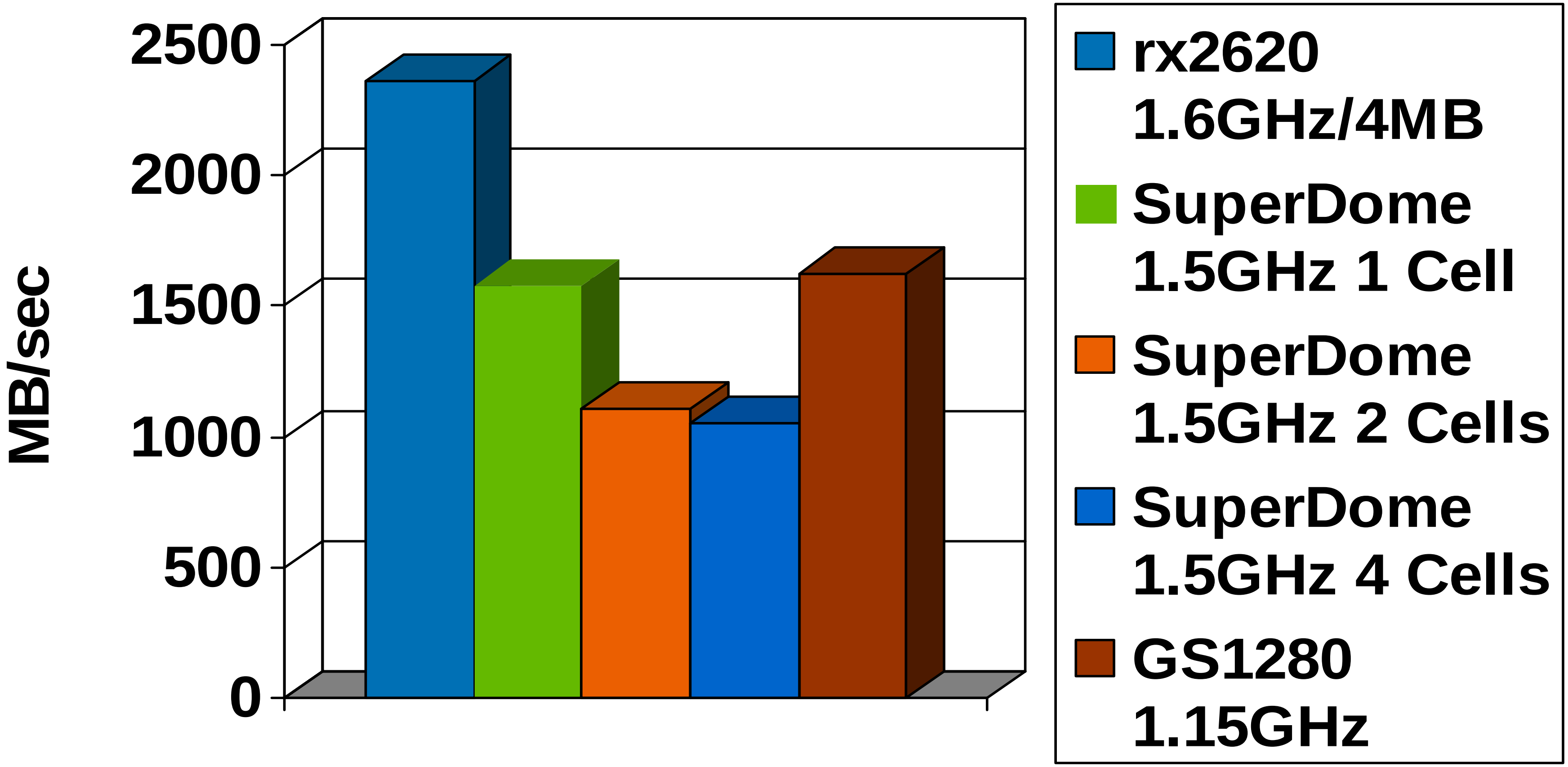


More is better

Memory Bandwidth (large servers)

Computed via memory test program

- MEMSpeed – Test Program



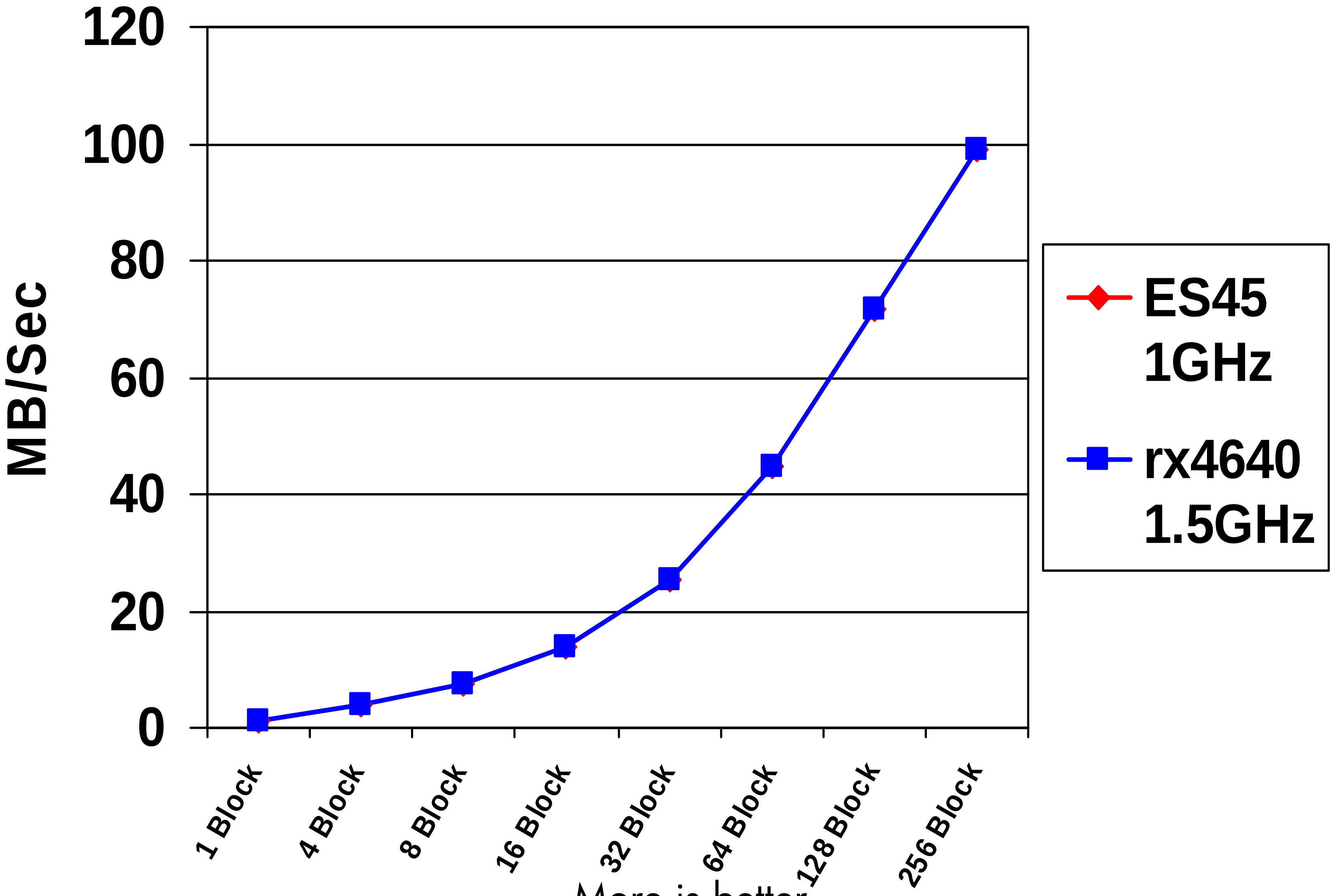
More is better

Memory Comparison

- Alpha Servers have very good memory latency
  - Applications that read small amounts of data from many different memory locations should perform well
- The small Integrity Servers have very good memory bandwidth
  - Applications which move memory around or heavily use caches or RAMdisks should perform well
- The large Alpha Server have very good memory latency and good memory bandwidth

IO MB/Sec – single process

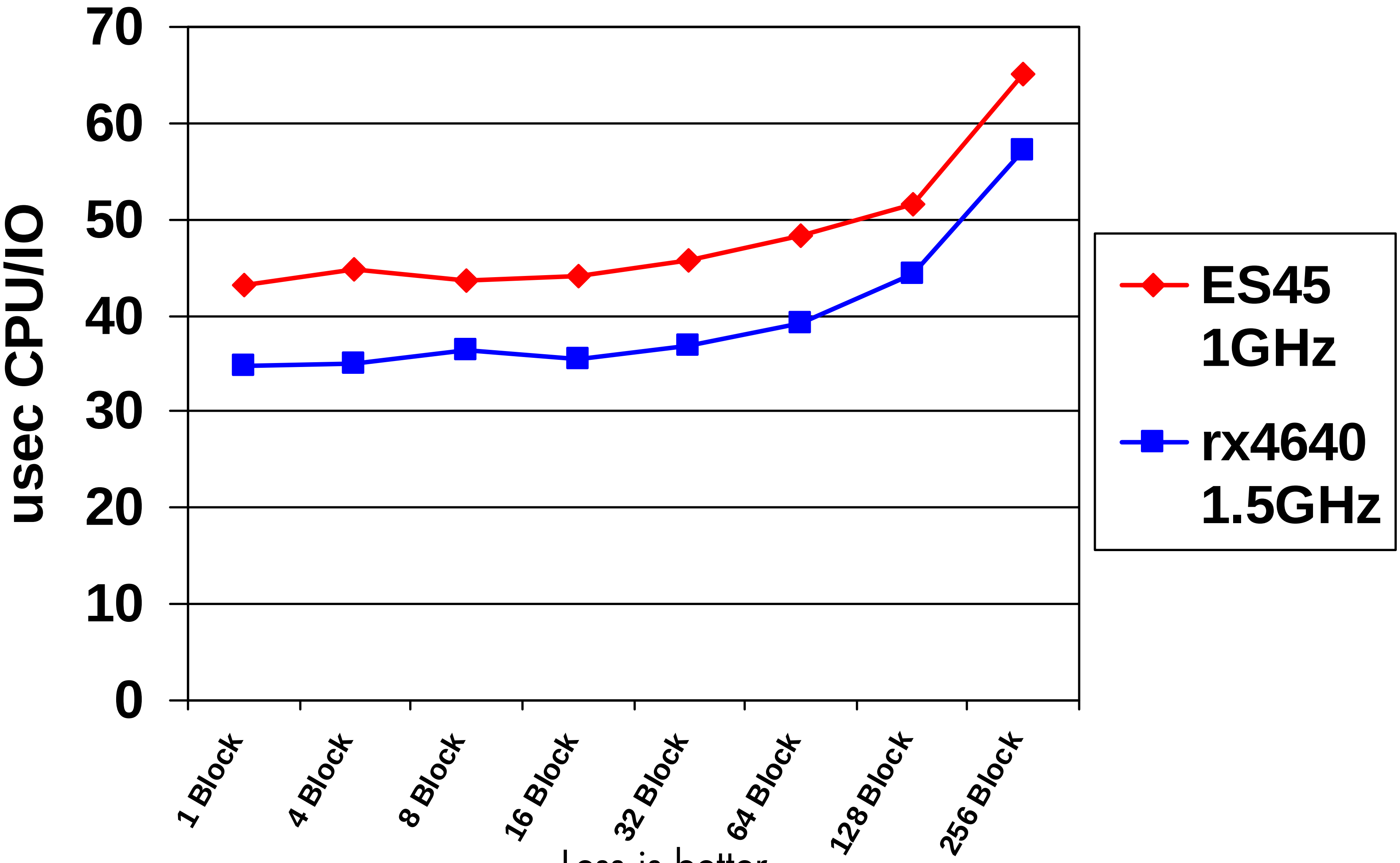
(QLogic ISP2313) 2Gigabit Fiber Channel Card - EVA-GL
Random Read/Write



More is better

CPU per IO – single process

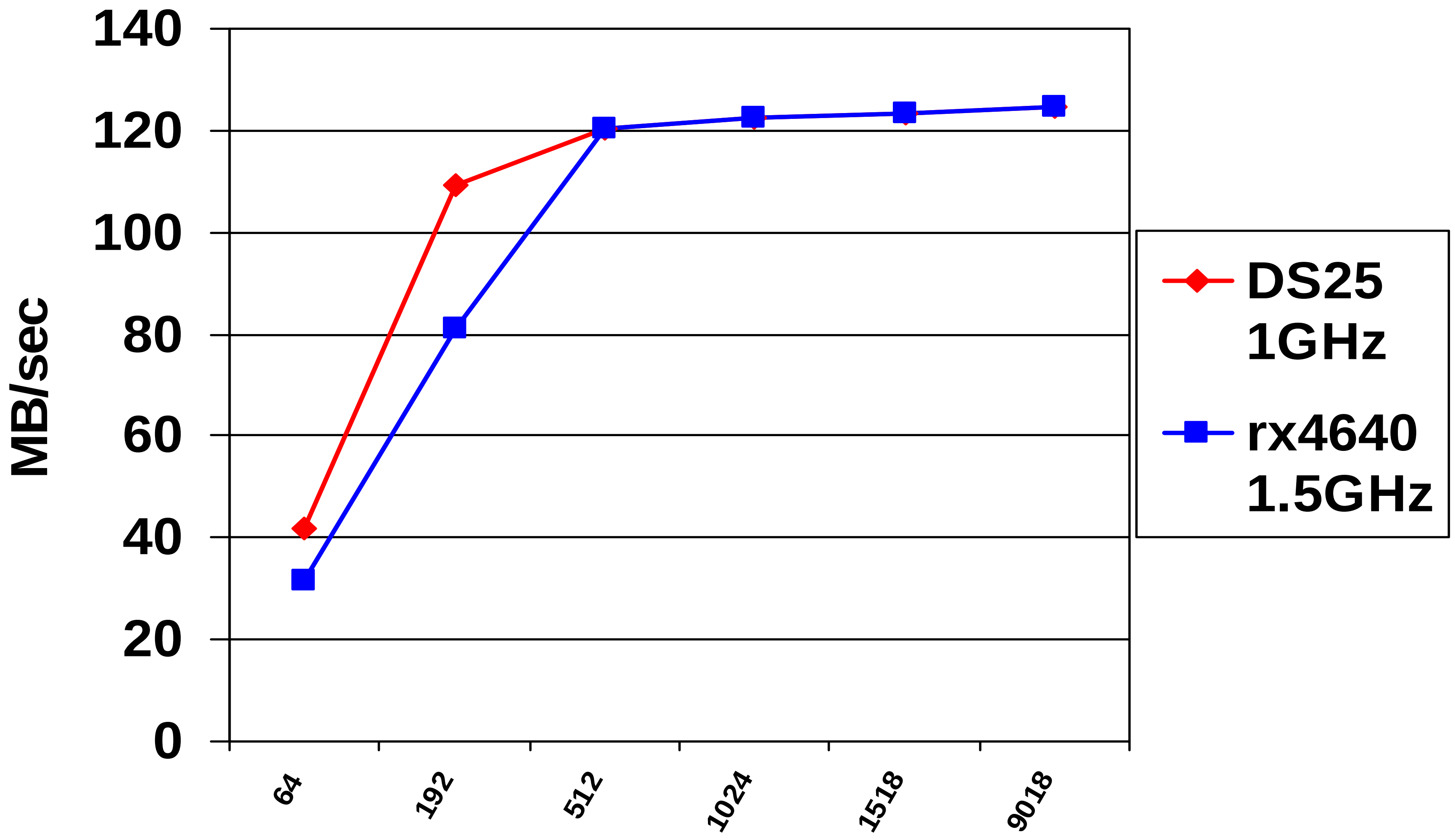
(QLogic ISP2313) 2Gigabit Fiber Channel Card - EVA-GL
Random Read/Write



Less is better

Gigabit Transmit MBytes/Sec

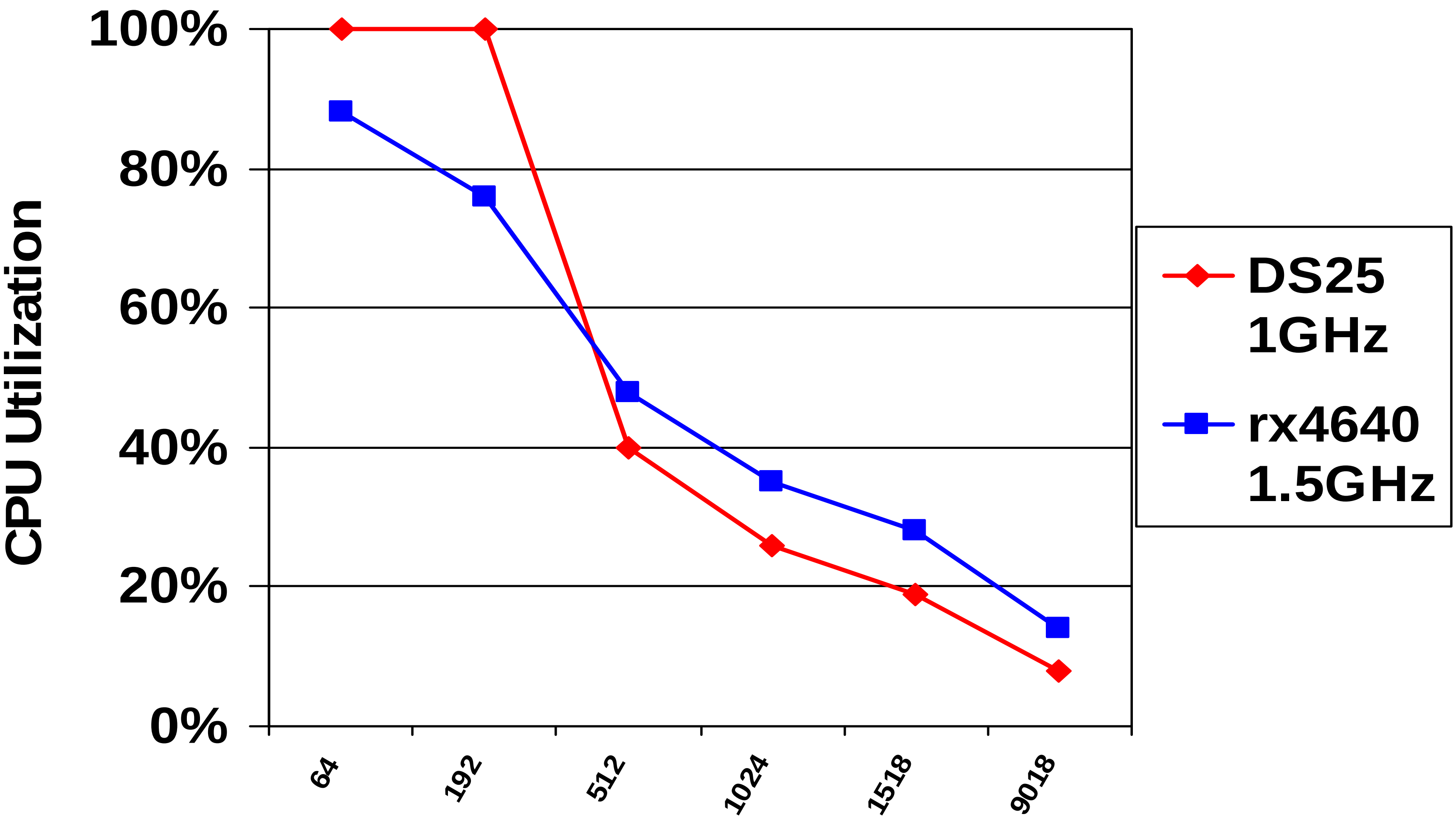
rx4640 1.3GHZ (A6825A - Broadcom 5701) in 64-bit PCI @ 66 mhz
DS25 1GHz (DEGXA - Broadcom 5703) in 64-bit PCI @ 66 mhz



More is better

Gigabit Transmit CPU Utilization

rx4640 1.3GHZ (A6825A - Broadcom 5701) in 64-bit PCI @ 66 mhz
DS25 1GHz (DEGXA - Broadcom 5703) in 64-bit PCI @ 66 mhz



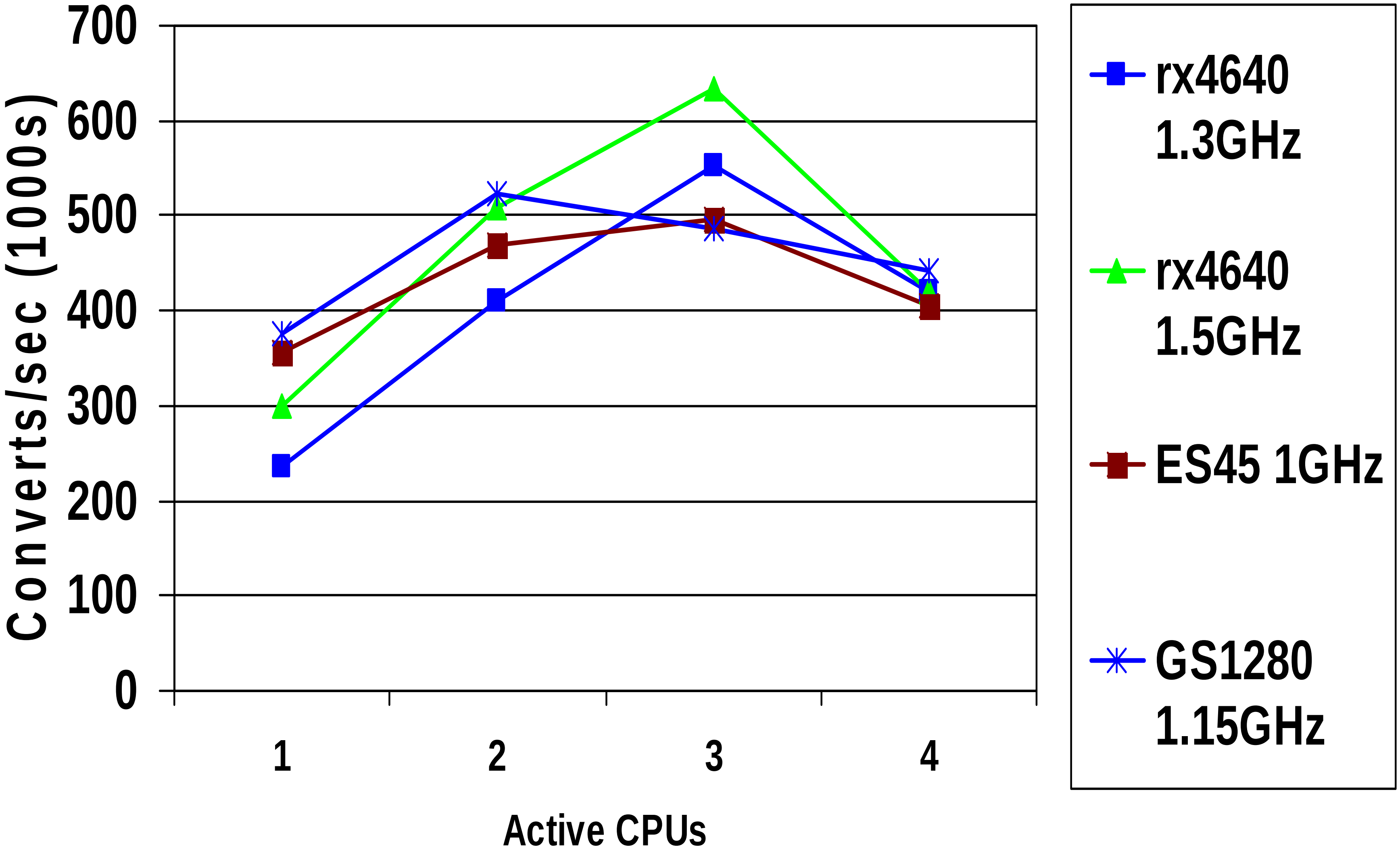
Less is better

IO Conclusions

- IO appears comparable between Alpha and Integrity Servers
  - Both Integrity and Alpha can driver IO adapters at comparable levels
  - CPU cost per IO appears better on Integrity servers
- We don’t have IO Stress tests from Superdome class systems at this point

Lock Manager Stress Test

4 Processes

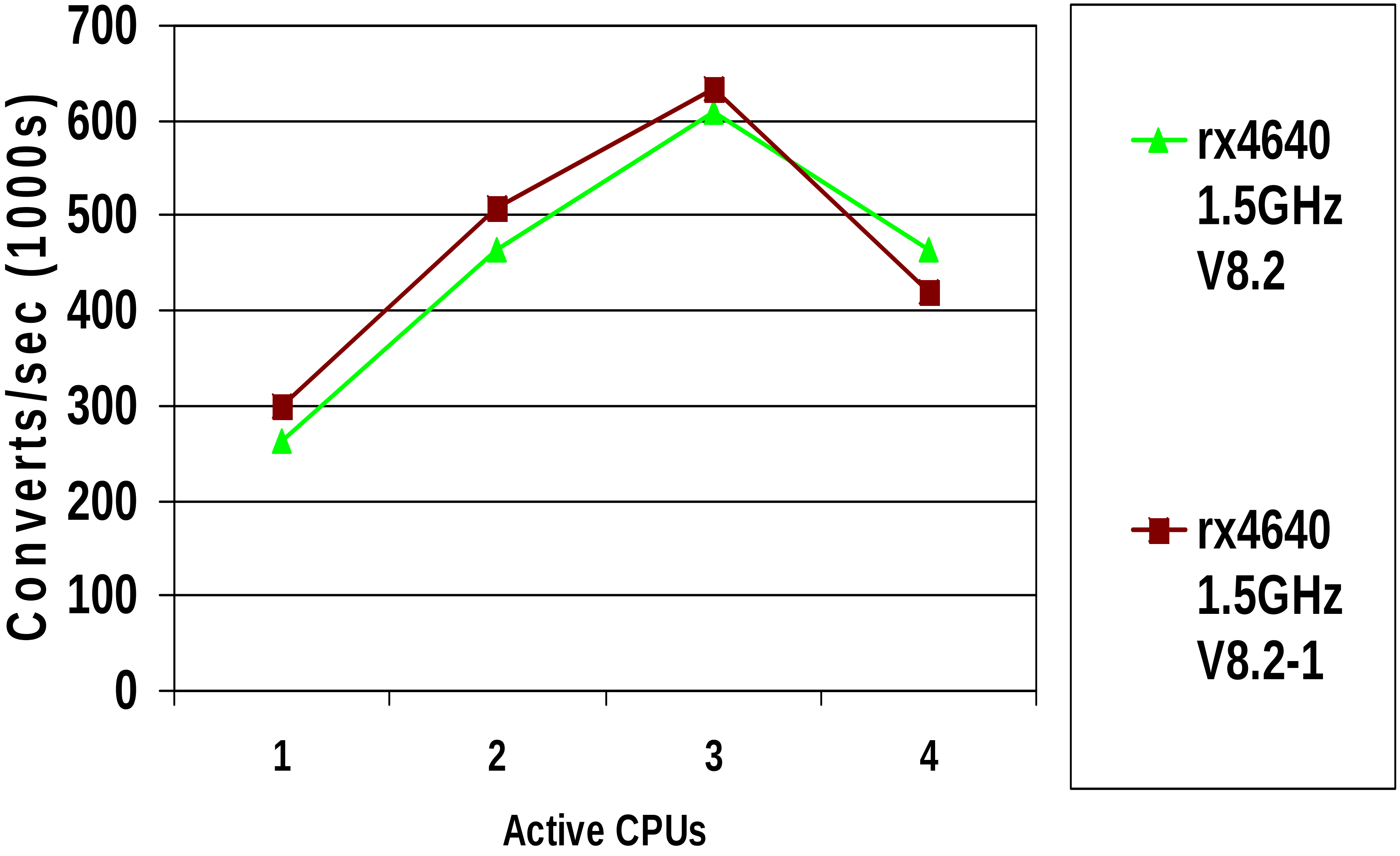


More is better

Lock Manager Stress Test

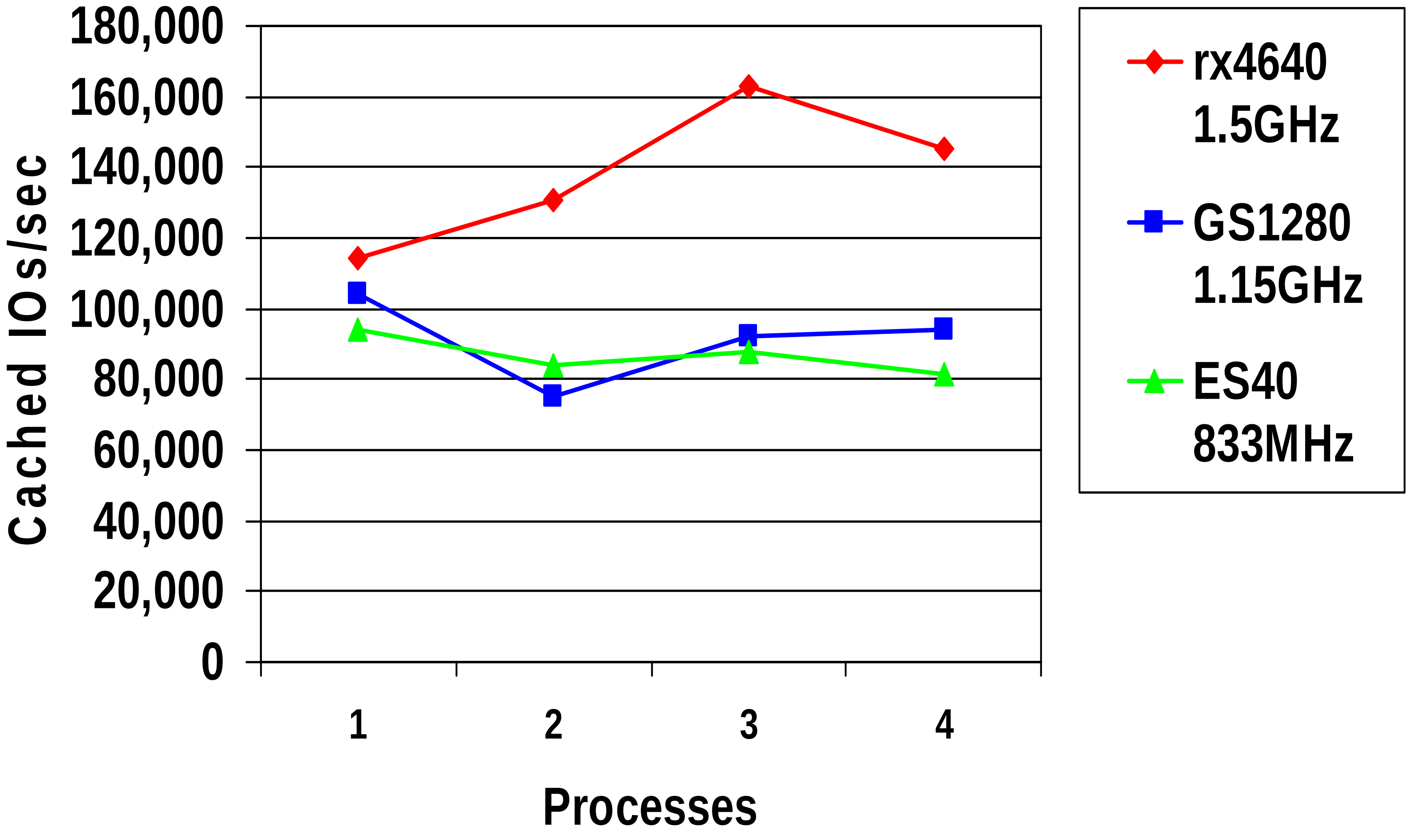
V8.2 compared to V8.2-1

4 Processes



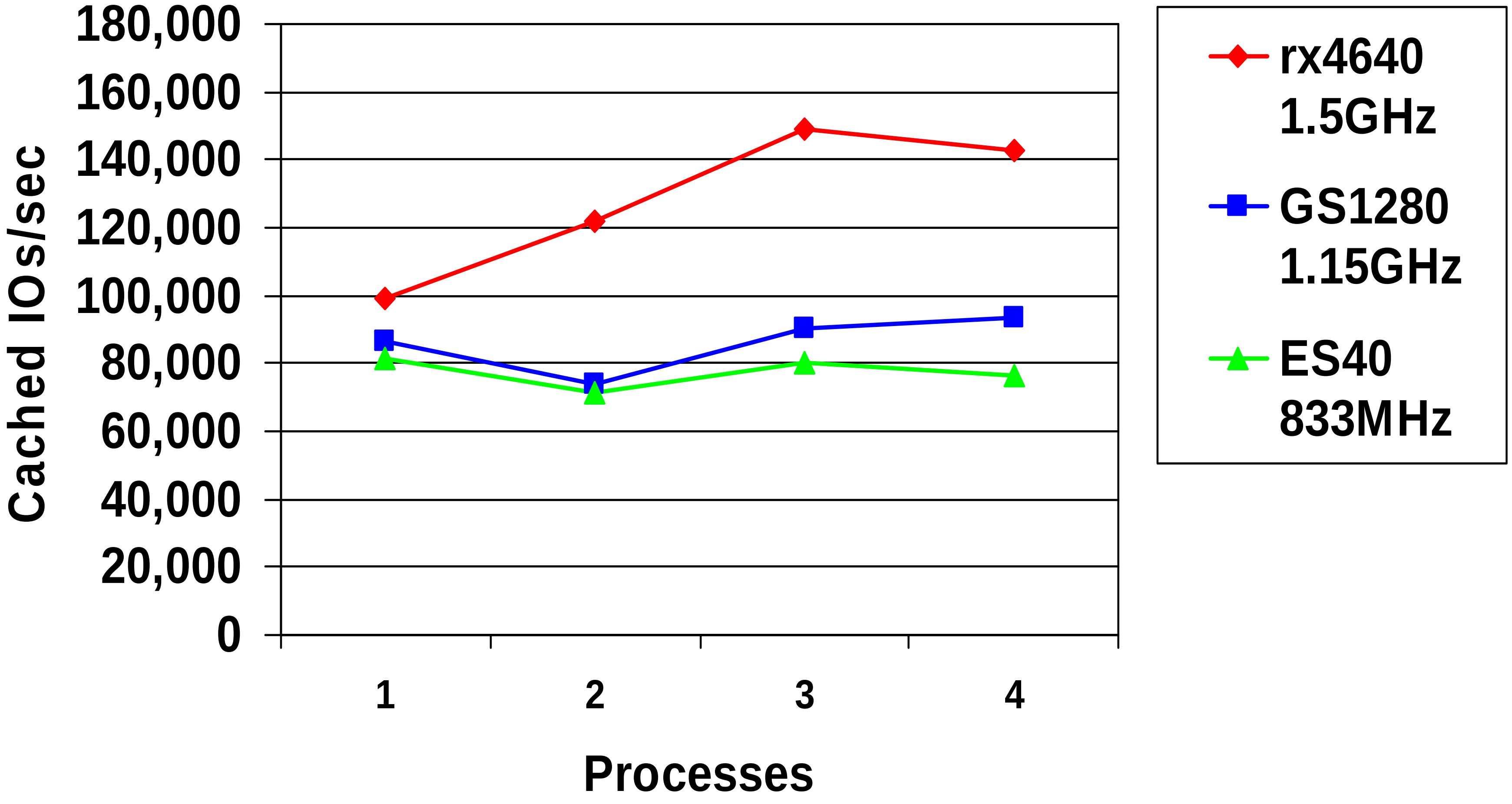
More is better

XFC Cached 1 Block IOs



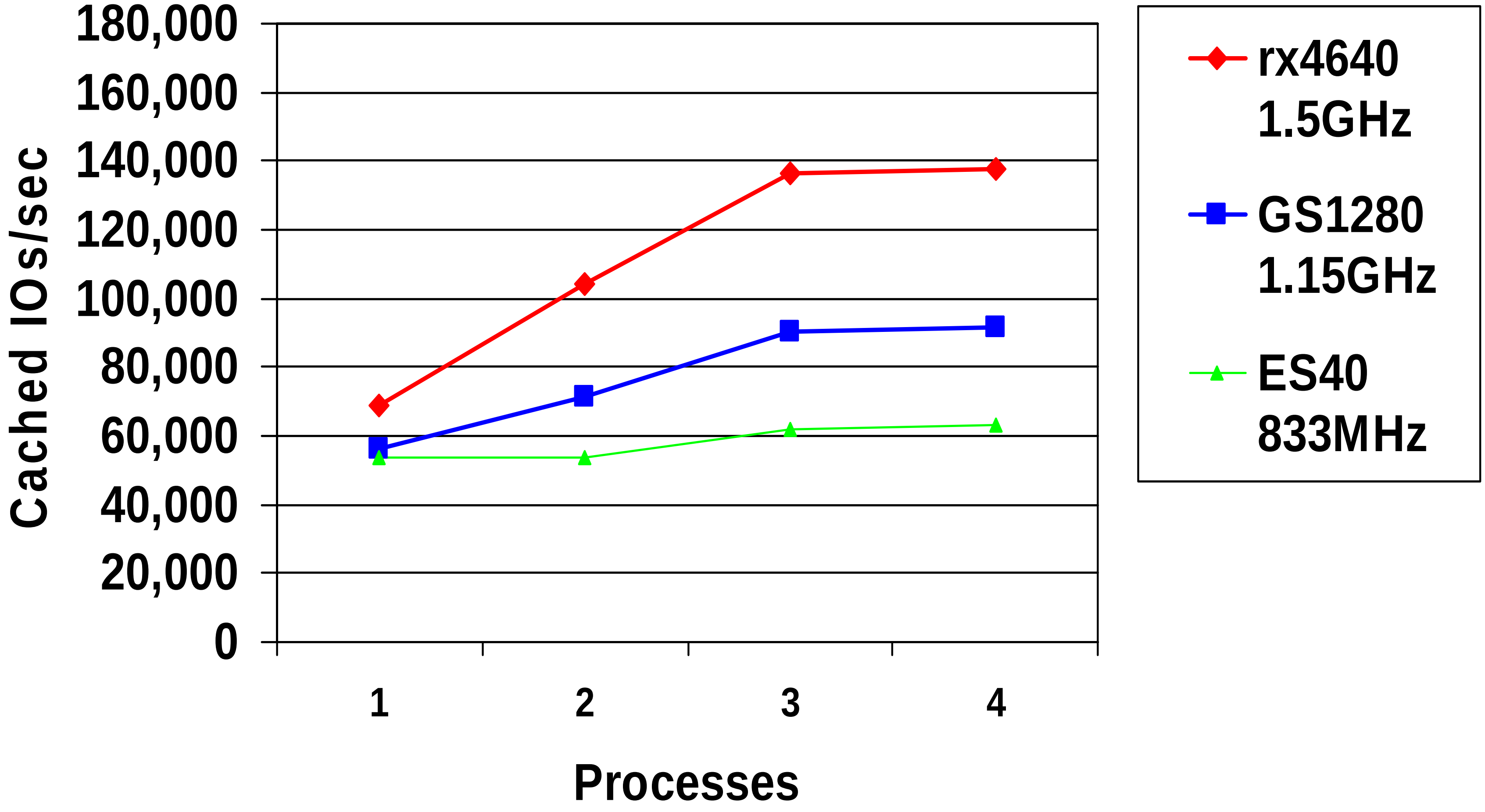
More is better

XFC Cached 4 Block IOs



More is better

XFC Cached 16 Block IOs



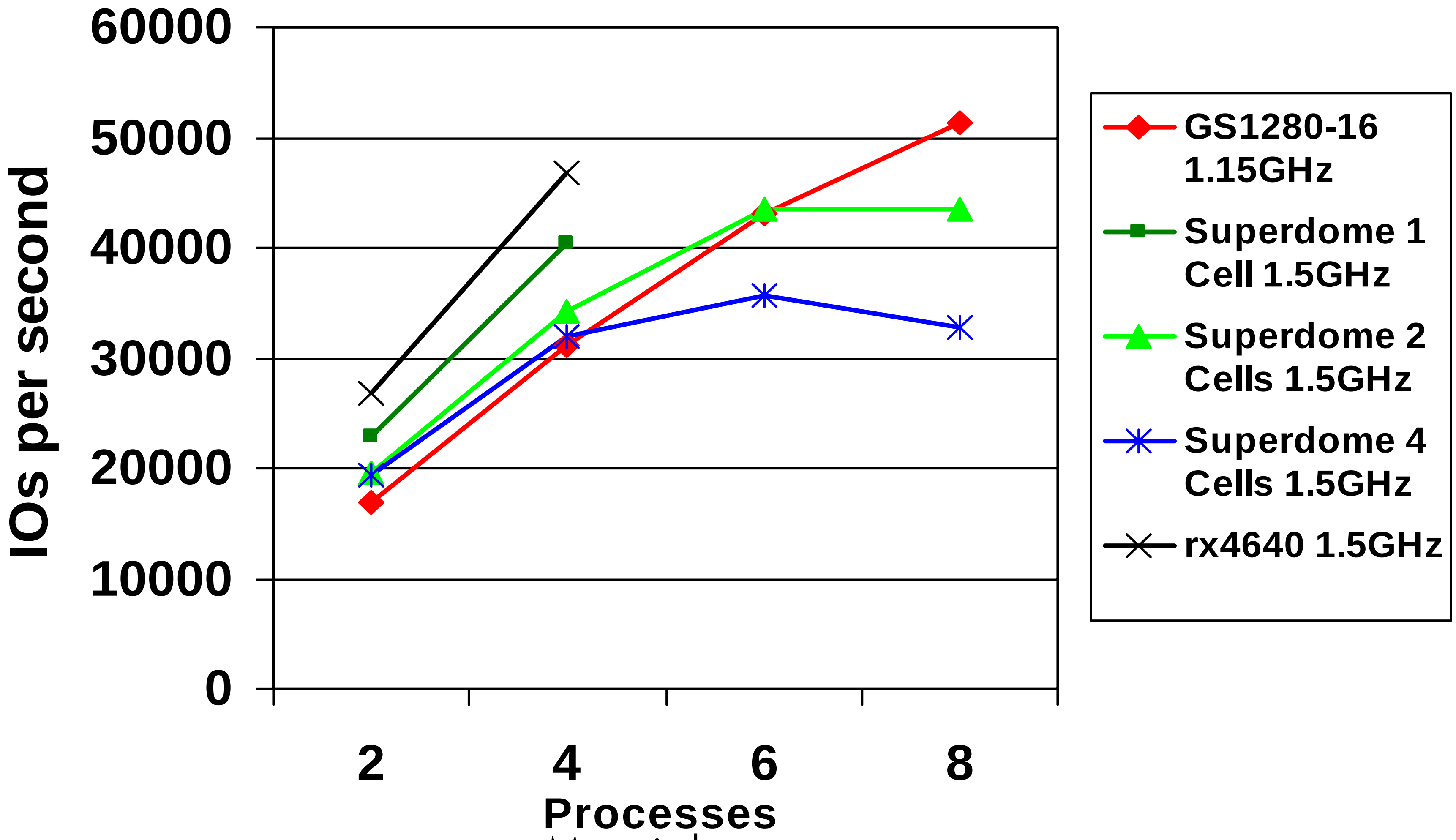
More is better

Performance Improvements in V8.2-1

- SWIS Improvements
  - Reduced overhead
- System Service Dispatching
- $SETSTK_64 implemented as an EPC service
- Alignment Fault fixes
  - Macro32, LBRSHR, LIB$MATCH_COND, LCKMGR
- DEC$BASRTL (in V8.2 Tima)

RMS1 (Ramdisk)

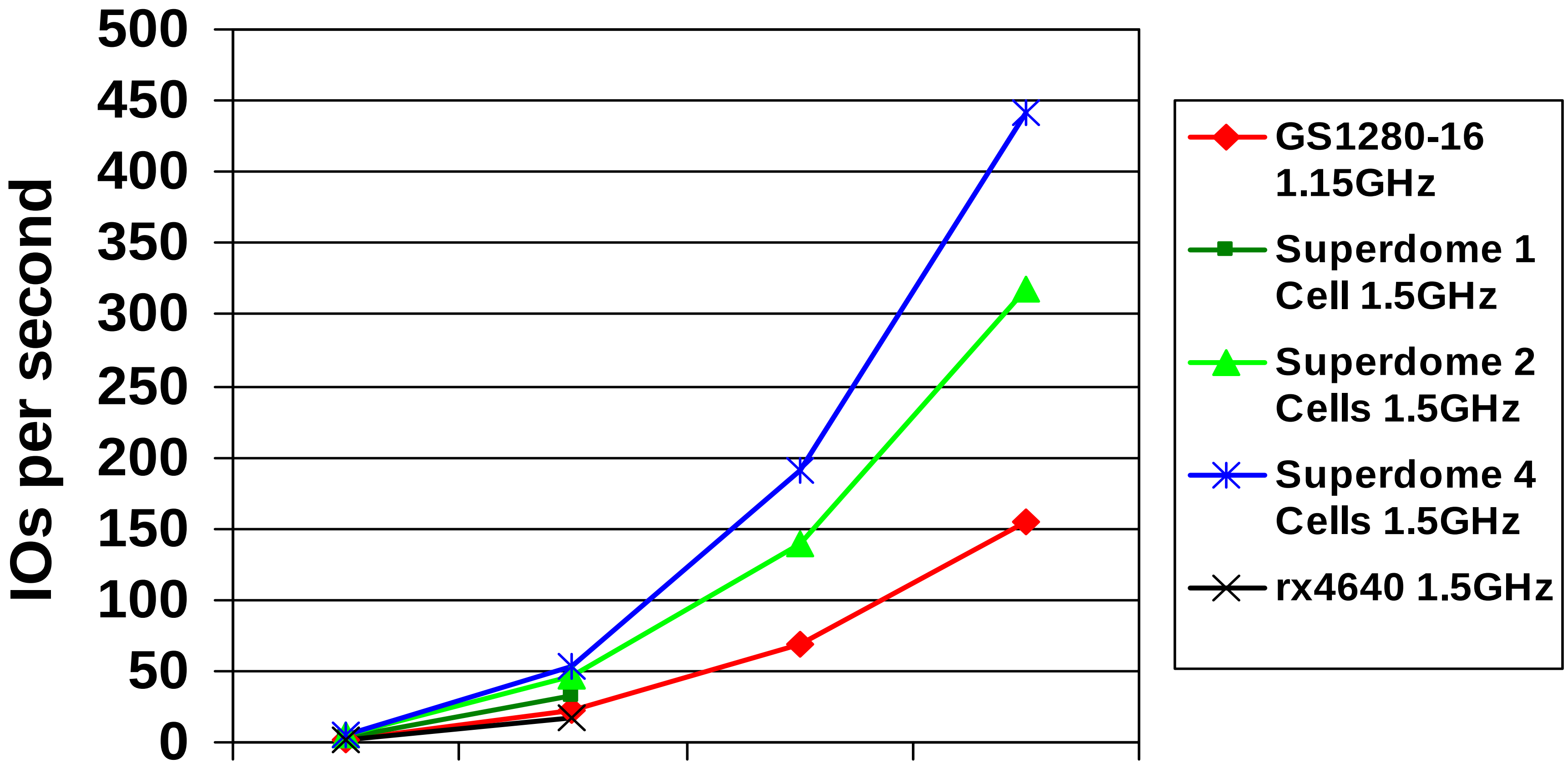
Direct IOs



More is better

RMS1 (Ramdisk)

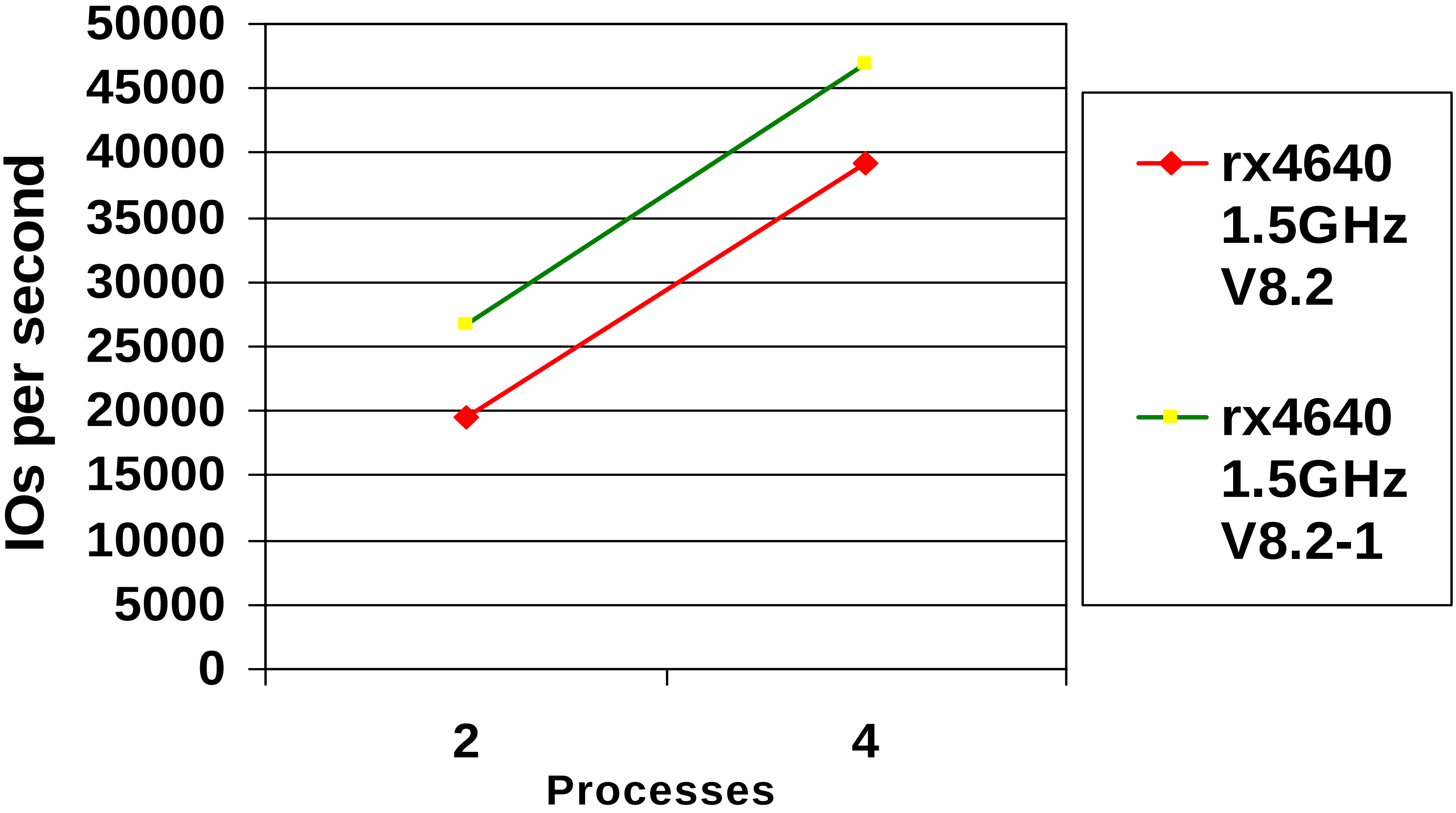
MP Synch



Processes

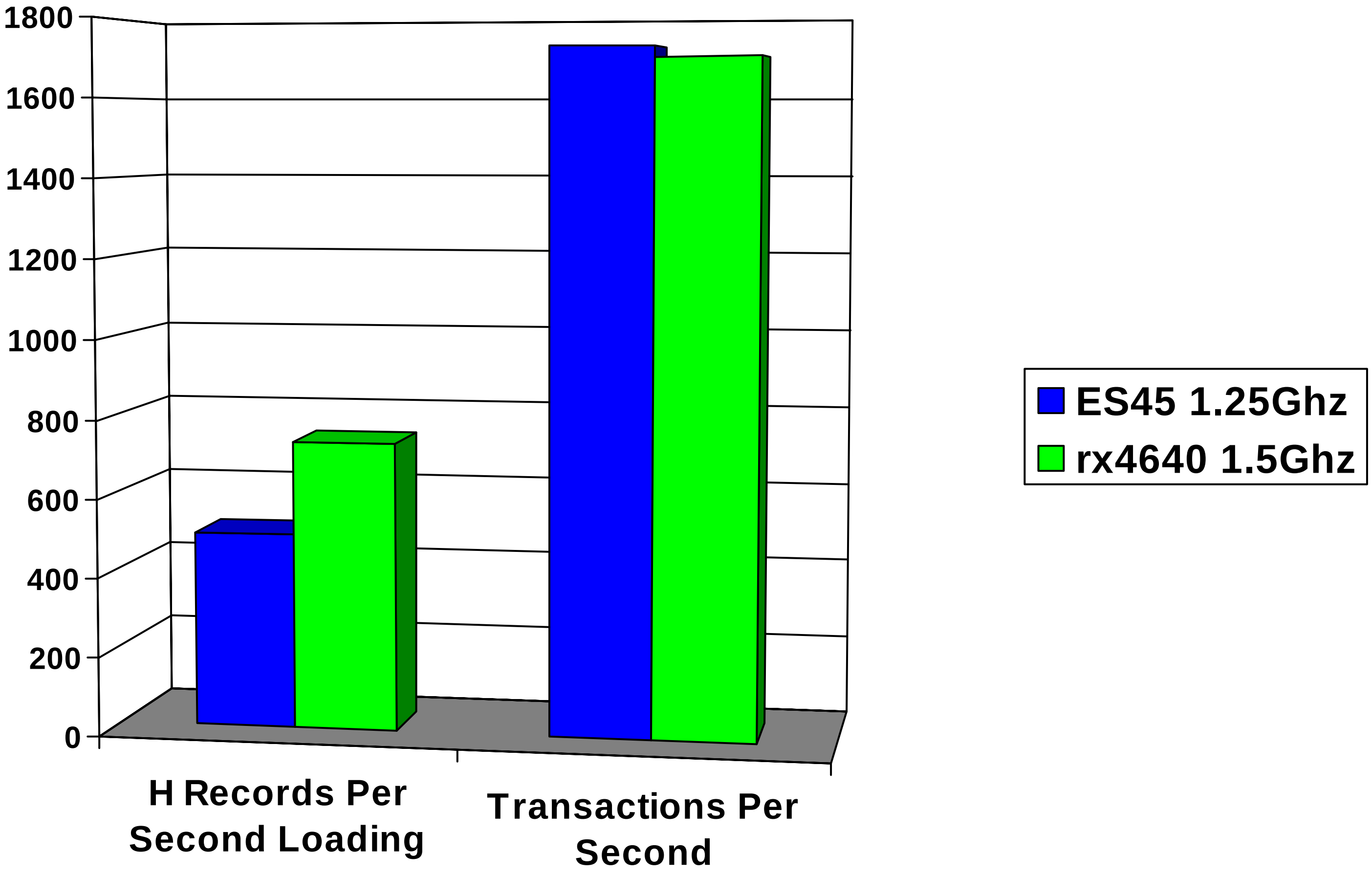
Less is better

RMS1 (Ramdisk)
V8.2 compared to V8.2-1



More is better

Oracle RDB Performance Comparison



30 process OLTP workload

Areas that are Slower on Integrity

- There are several areas where the equivalent operations on Integrity systems are slower
  - **Alignment Faults!**
  - **Exception Handling!**
    - Establishing/Finding Exception Handlers takes longer
    - Exceptions Frames much larger
  - VAX Floating Point Data Types (due to conversions)
  - Did you compile /NOOPTIMIZE?
- Integrity Images are typically 3 times as large
  - This can impact image activation time
    - Requires more IO
    - Increased page faults
  - Require larger GH regions if images are installed resident
  - Requires more disk space for listings and object files

Macro Compiler Story

- Macro Compiler
  - Customer had a regression test suite of 7 batch jobs
    - Spent most of the time compiling Macro32 code
    - Alpha 1000 (244MHz CPUs)
      - Total CPU Time: 5:29:23 Total Elapsed Time: 22:17:07
    - IPF system: rx2600 dual 1.4GHz CPUs
      - Total CPU Time 10:26:17 Total Elapsed Time: 21:54:41
  - PC Sampling revealed very heavy alignment faults
  - About 10 fixes were made to avoid the alignment faults
    - IPF system: rx2600 dual 1.4GHz CPUs
      - Total CPU Time 1:46:21 Total Elapsed Time: 7:07:49

Application Signaling Story

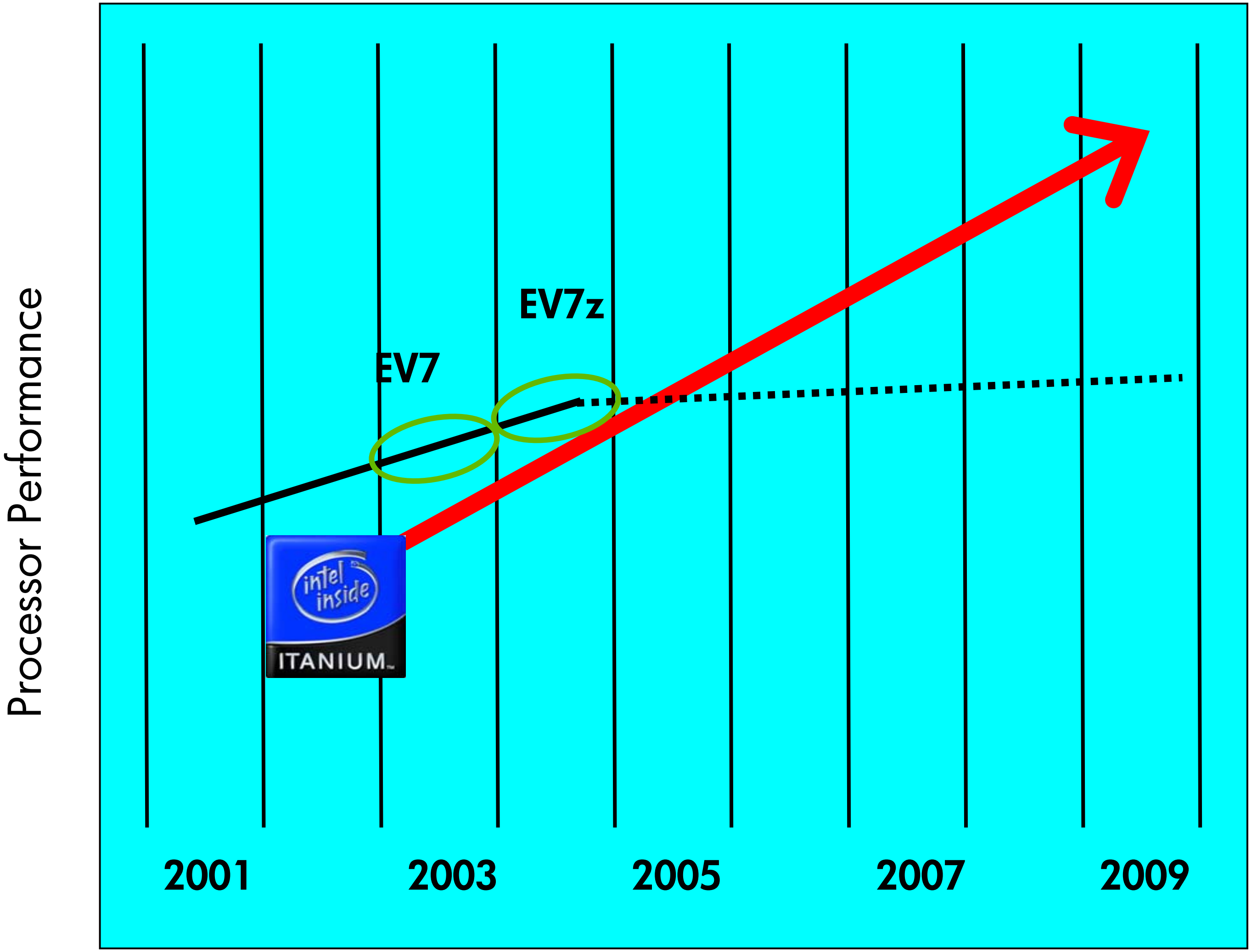
- Recent testing of an application on Integrity showed very poor performance
  - There was very heavy CPU usage compared to Alpha
- PC sampling showed the problem to be with establishing condition handlers
  - These condition handlers were very short lived
  - The code contained calls to the old CMA library routines to perform mutex and condition variable operations
    - The TRY/CATCH_ALL/ENDTRY macros were necessary to catch error statuses – the CMA routines did not directly return status codes
- Solution:
  - call pthreads directly and the condition handling setup is no longer required

OS Update Story

- An HP field person was testing a small program on Integrity for a large financial institution
  - In order for the customer to move to Integrity in the next 6-9 months, this test had to perform well
- The test took twice as long and used twice the CPU on integrity when compared to Alpha
- Analysis showed very heavy memory allocation and deallocation on Integrity that did not appear on Alpha
  - The OS code in question used KP services to create contexts to execute each transaction
    - This was a porting change since usage of stack context is much more involved on Integrity than on Alpha
  - An update was made to cache the data structures and provided back to the HP field person for testing
- The test program now run faster in both elapsed time and CPU usage on Integrity than on Alpha

Projected Performance Crossover Point predicted three years ago

OpenVMS on



Conclusions

Conclusions – hardware

- The Itanium Processors are fast
  - Future processors will continue to increase the performance over current alpha processors
- IO performance is also comparable between Alpha and Integrity
  - Integrity has an edge in CPU cost per IO and in cached and DECram IO
- The rx4640 and rx26xx systems have great memory bandwidth and good memory latency
  - In most cases, these systems should perform similar or better to comparable Alpha systems
- The larger Integrity servers have slower memory latency
  - If your application taxes a large GS1280, we recommend testing on larger Integrity servers before moving performance critical applications

Conclusions - software

- The OpenVMS operating system performs well on Integrity servers with just a few caveats
  - Alignment faults and exception handling
- OpenVMS V8.2-1 shows performance improvements over V8.2
  - Additional Integrity Server improvements can continue to be expected in the next release of OpenVMS
- We expect most applications to perform well on Integrity servers
  - If you port an application and are disappointed in performance we want to know
    - Please contact me with performance issues at:

Gregory.Jordan@hp.com