Porting OpenVMS to the Itanium™ Processor Family

Andy Goldstein
OpenVMS Engineering

(acknowledgements to Clair Grant)

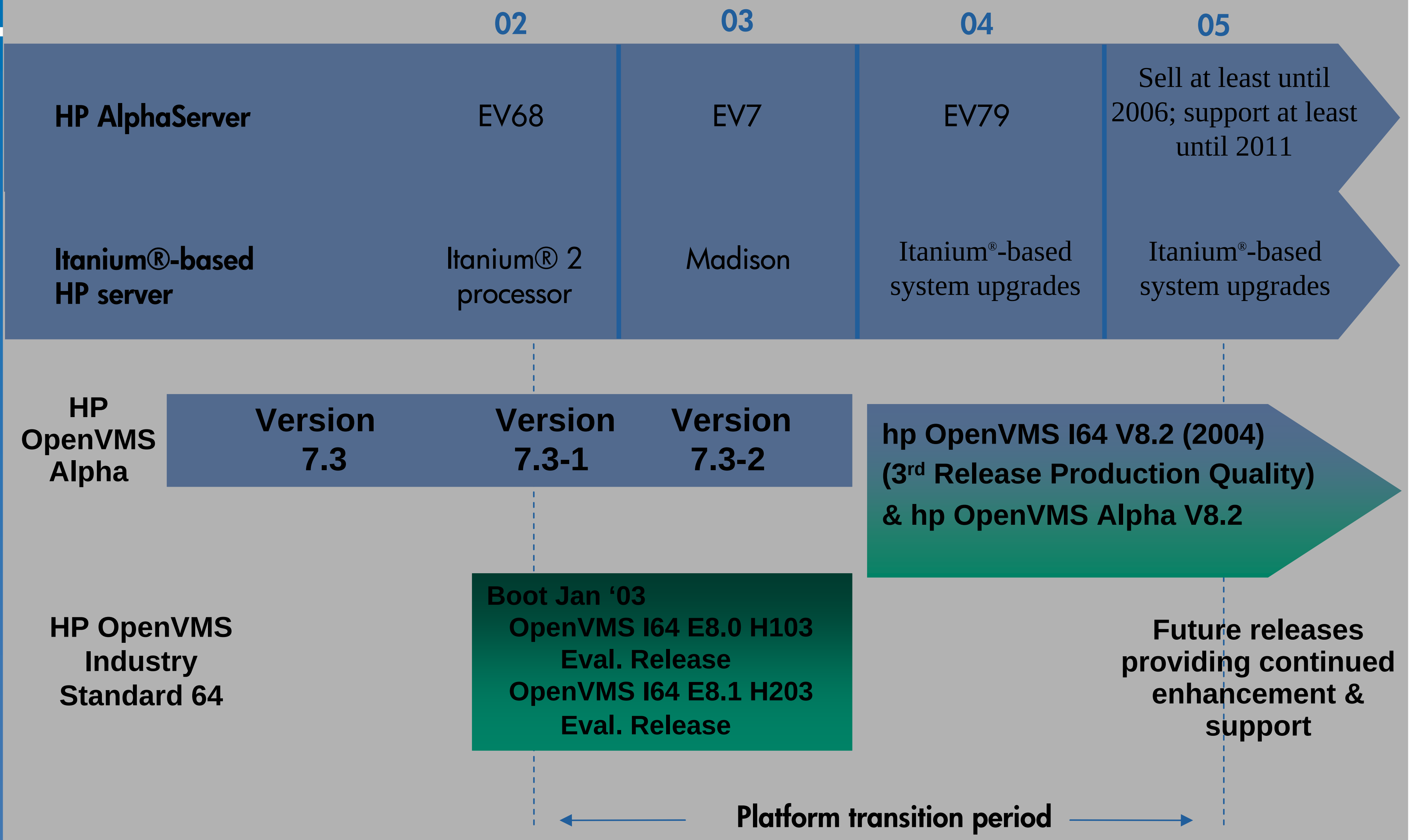
October, 2004

The information contained herein is subject to change without notice

"We will port OpenVMS to the IA64 architecture and ship a production quality release in 2004."

June 25th, 2001
Hewlett-Packard

hp OpenVMS Roadmap



02

03

04

05

HP AlphaServer

EV68

EV7

EV79

Sell at least until 2006; support at least until 2011

Itanium®-based HP server

Itanium® 2 processor

Madison

Itanium®-based system upgrades

Itanium®-based system upgrades

HP OpenVMS Alpha

Version 7.3

Version 7.3-1

Version 7.3-2

hp OpenVMS I64 V8.2 (2004)
(3rd Release Production Quality)
& hp OpenVMS Alpha V8.2

HP OpenVMS Industry Standard 64

Boot Jan '03
OpenVMS I64 E8.0 H103
Eval. Release
OpenVMS I64 E8.1 H203
Eval. Release

Future releases providing continued enhancement & support

Platform transition period

H103 H203 H104 H204

1st Boot on i2000 system, January 31, 2003 3:31 PM EST

Boot on rx2600 system (First Ship platform), March 17, 2003

★

First Ship

H103: hp OpenVMS I64 E8.0 “Mako” Evaluation Release
Audience: Selected ISVs and Partners
OpenVMS Itanium Operating System, Monitor Utility

Networks: DECnet Phase IV, TCP/IP

Development Tools: Cross Linker, Cross Librarian, Native Debugger

Cross Compilers: C, C++, BLISS, FORTRAN, IMACRO

★★

H203: hp OpenVMS I64 E8.1 “Jaws” Evaluation Release
Audience: Key ISVs, Partners, Early Adopters
Limited cluster functionality (4 nodes)

Native Compilers: C, C++, BLISS, FORTRAN, IMACRO, Pascal, BASIC, COBOL

Additional Language Support: JAVA

Additional Layered Products…Networks, Data Serving, Security, eBusiness Integration, Application Development

▨	Internal releases
🟨	External releases

★★★

Production Quality

HP OpenVMS I64 V8.2

Porting Goals

- Provide an operating system environment, development tools, and documentation to make porting as easy as possible
  - Full port of the Operating System, Runtime Libraries, development tools and most layered products
  - Recompile, relink, requalify
- Use our experiences porting the operating system to make it easier for others to port their applications
  - Internal layered product groups, partners, and customers

Porting Philosophy

- This is not a “bug for bug compatible” coding exercise. We are doing much, much more.

- First, we are not just porting to the Itanium® architecture; we are making OpenVMS more portable.

- Second, we are improving code maintainability (and sometimes performance) by replacing VAX assembler code when appropriate.

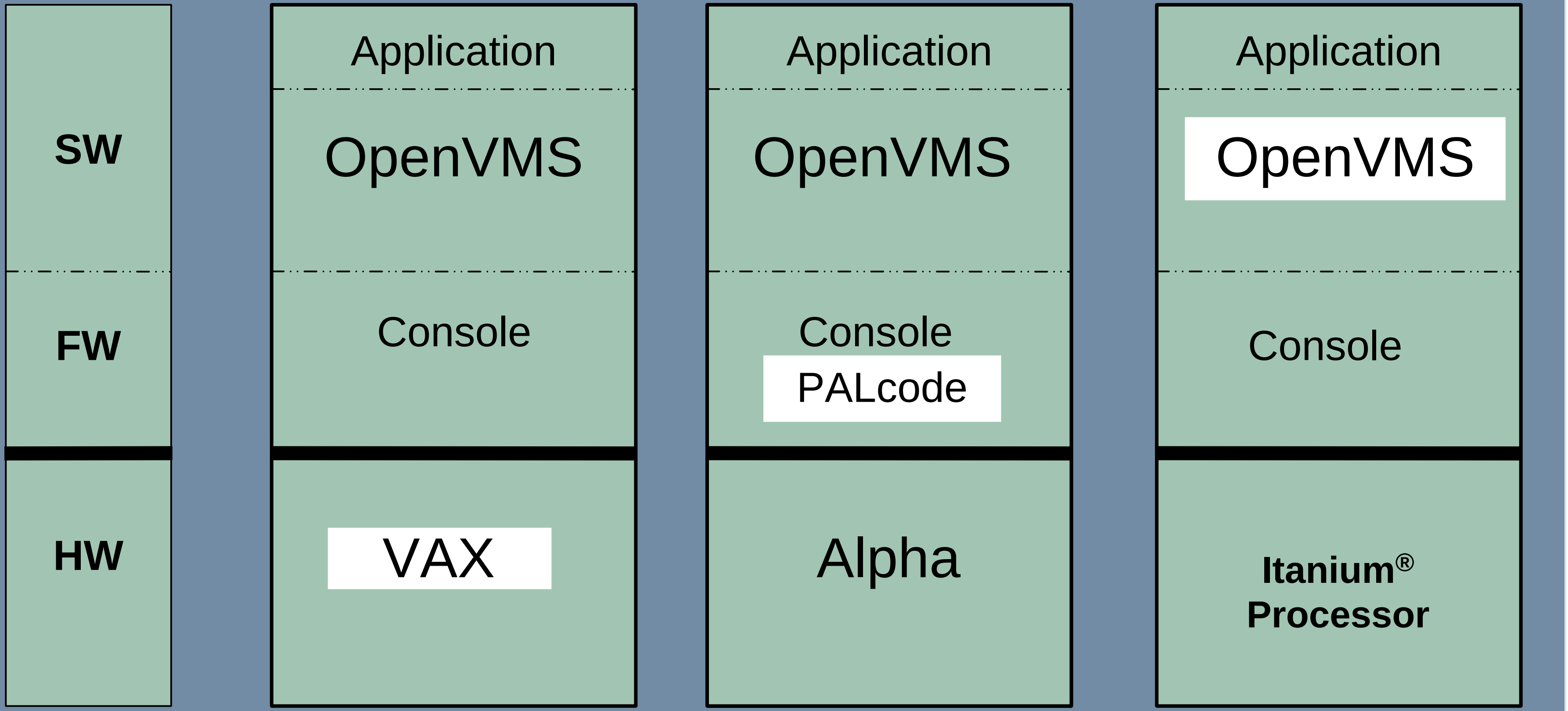
- Third, we are making the system more open to the possibility of exchanging code with other systems, especially analysis tools. (This even helped us with some early debug.)

Big Challenges for the Base OS

- No Alpha Console
  - Booting
  - Device Discovery
  - Interrupts
  - TLB miss handler
- No Alpha PALcode
  - VAX Queue Instructions
  - VAX Registers
  - IPL and mode change

- Different primitives in CPU
  - Register Conventions
  - Exception Handling
  - Atomic Instructions
  - Process Context
- Plus, we decided to change
  - calling standard
  - object language
  - image format

It's All in the Software



SW	Application	Application	Application
	OpenVMS	OpenVMS	OpenVMS
FW	Console	Console PALcode	Console
HW	VAX	Alpha	Itanium$^{®}$ Processor

Console

- Intel-architected boot environment
  - In the FAT partition
    - VMS_LOADER.EFI
    - IPB.EXE
  - Operating system interface to ACPI data
- Boot drivers for SCSI and IDE
- VMS uses standard PAL/SAL console interfaces to get
  - Translation buffer info
  - Clock frequency
  - Machine Check Vectors
  - ….

Alpha PALcode Replacement

- The following are all managed in the PAL on Alpha; VMS does the work on Itanium
  - VAX queue instructions (compilers generate OS calls)
  - VAX internal processor registers
  - AST and software interrupt support
  - IPL
  - Swap context

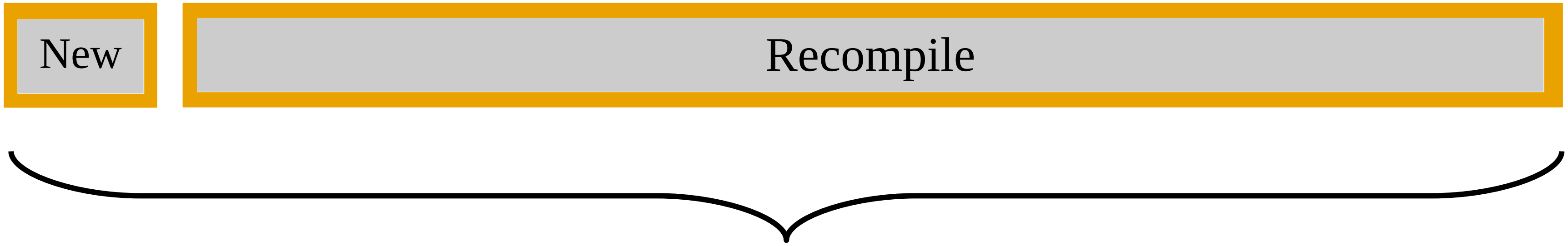
CPU Primitives

- Different register conventions
  - R8/return status; R12/stack pointer
  - Compilers do the translation
- More registers
- IMACRO automatically replaces LDL/STC sequences
- Memory fence replaces memory barrier
- Compilers have builtins for most functions

Calling Standard

- Intel Calling Standard plus VMS extensions
  - Different register conventions
  - Unwind data
- Affected areas
  - Exception handling
  - Signaling
  - Compilers
  - LINKER
  - Debuggers
  - Non-standard routine calls
  - Swap context
  - Kernel Processes enhanced to provide “context-aware” routines for all modes – converted DCL, DECnet, XQP, RMS,……

Lines of Code Perspective



New

Recompile

OpenVMS on Itanium®-based systems

Porting effort is very focused on a few areas of the system.

How did it go?

- ‘Boot Contest’ on i2000
  - Scheduled: 12 weeks to boot (given “good” linked images)
  - Actual: 10 weeks (Jan. 31, 2003)
- Boot on rx2600
  - Scheduled: 6 weeks
  - Actual: 6 weeks and 1 day (Mar. 17, 2003)
- The effort and difficulty have been about as expected
  - No huge surprises
  - New Calling Standard and Object Language at least 50% of the project

Current Itanium Porting Status

- We're Done! (modulo a couple of bugs)
- Still Using Cross Tools to Build
  - C, C++, Bliss, iMacro, IA64 Assembler cross compilers
  - Cross Linker
  - Cross SDA allows for IA64 dumps analysis on Alpha
- Booted on
  - All supported systems
  - Several that aren't supported yet

Current Itanium Porting status

- What is not yet working (October 6)
  - ~~Edit/Teco~~
  - Delta Debugger — not in V8.2
  - ~~System Code Debugger (SCD)~~
  - ~~INSTALL performance features~~
  - ~~Security Server~~
  - ~~Registry Server~~
  - ~~ACME Server~~
  - ~~Shadowing~~
  - ~~Fibre Channel Boot~~
  - Cluster Satellite Booting — not in V8.2

OpenVMS Alpha and IPF Performance Comparison

Andy Goldstein
OpenVMS Engineering
Hewlett-Packard

(acknowledgements to Greg Jordan)

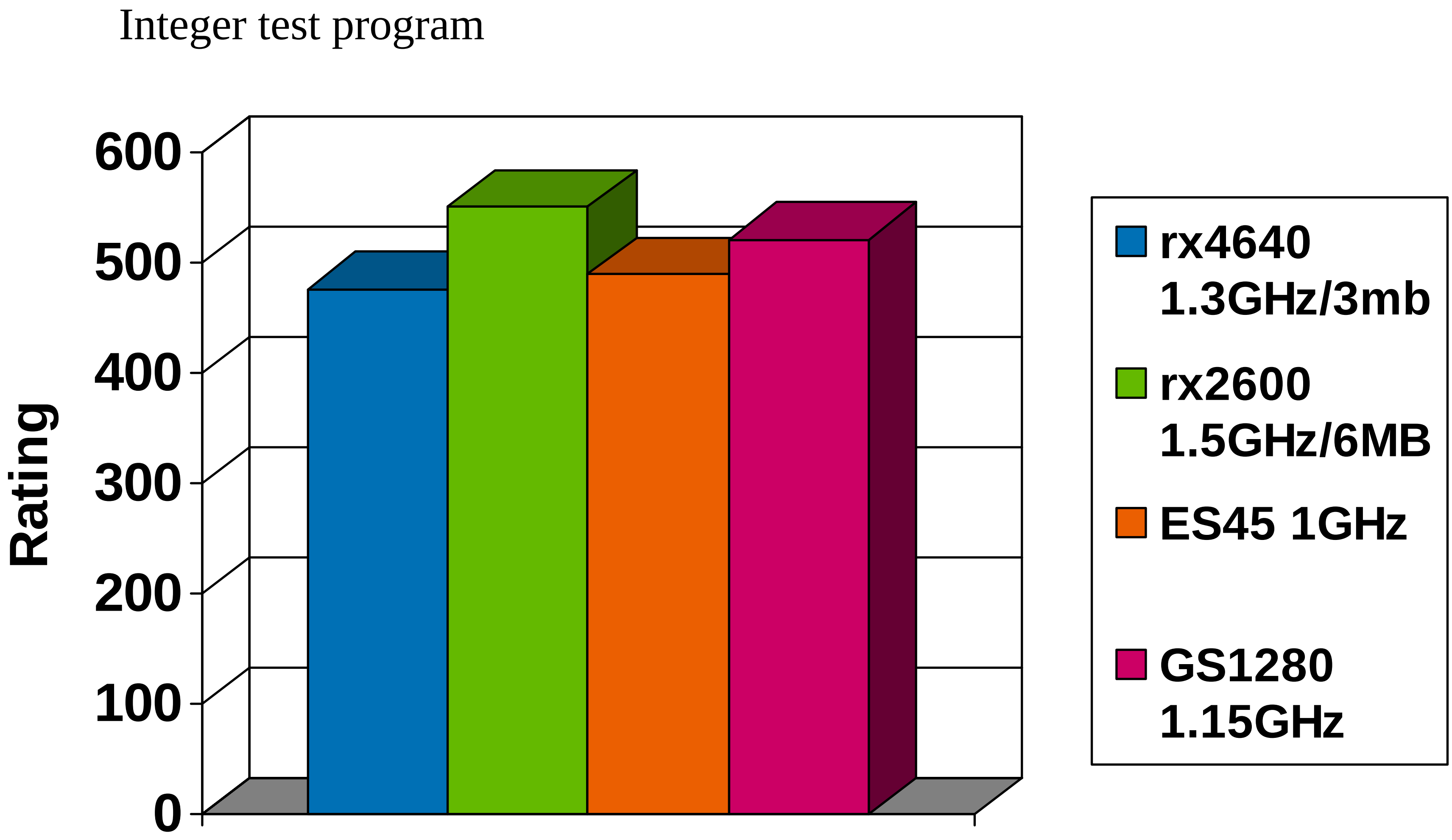
Overview

- The purpose of this presentation is to provide you with appropriate expectations of the performance of OpenVMS on Integrity platforms.
- The performance of various Integrity platforms will be compared with a variety of current Alpha platforms.

IPF/Alpha Performance Comparison

- The Basics
  - CPU
  - Memory
  - IO
- OpenVMS Operating System Performance
- Various Improvement Successes
- Performance Conclusions

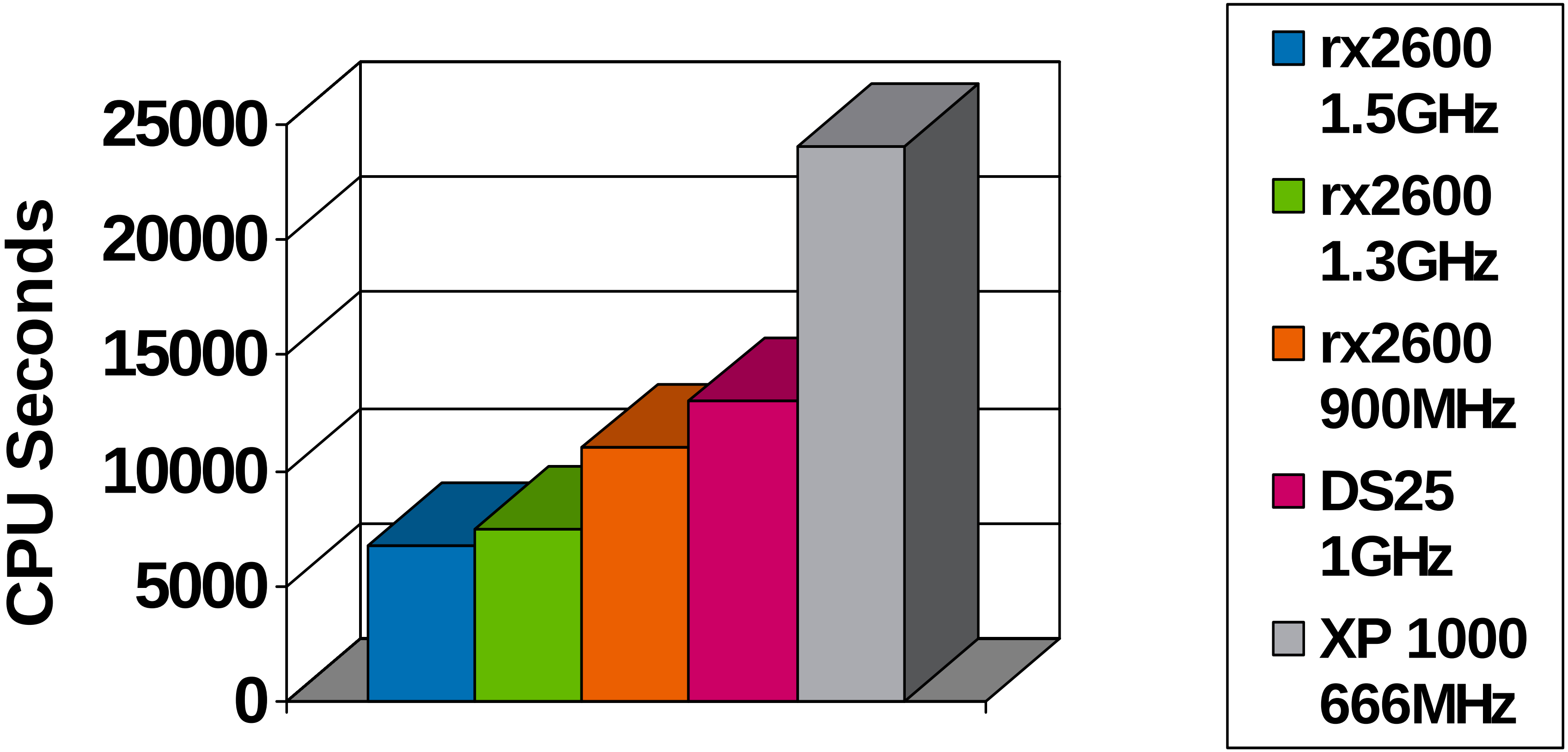
CPU – Integer test program



More is better

SETI

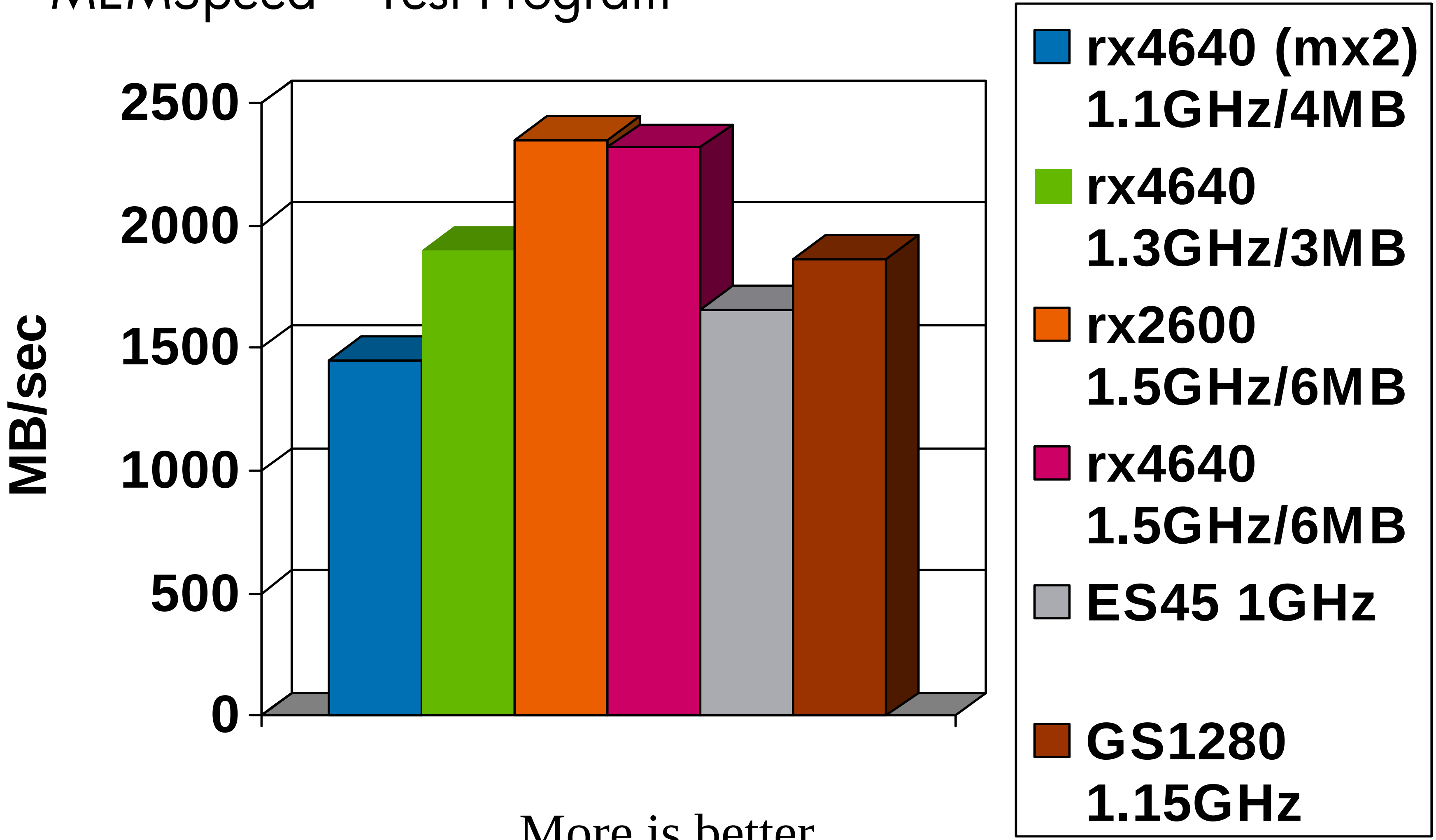
Time to Process a Work Unit



Less is better

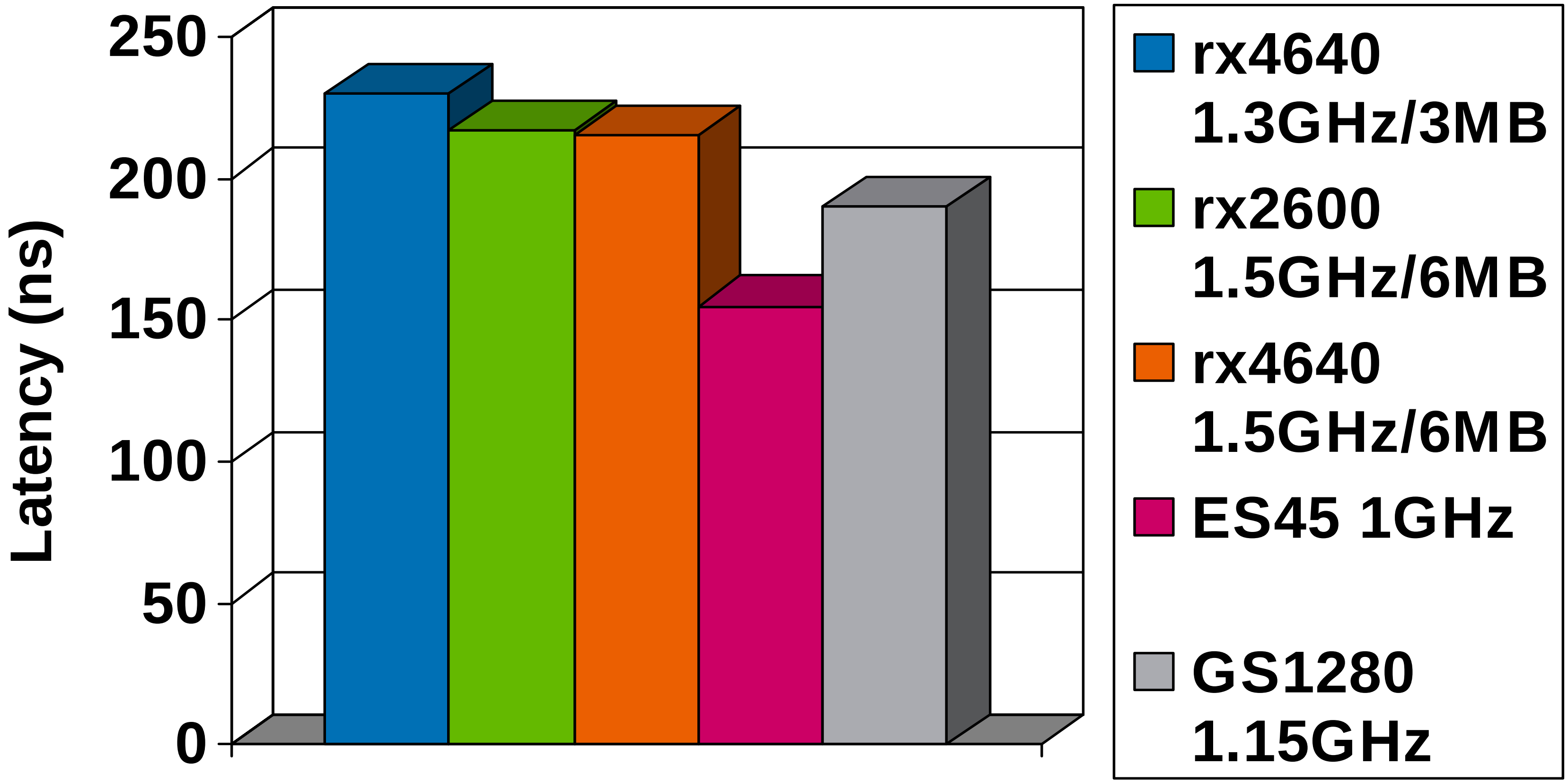
Memory Bandwidth

- MEMSpeed – Test Program



More is better

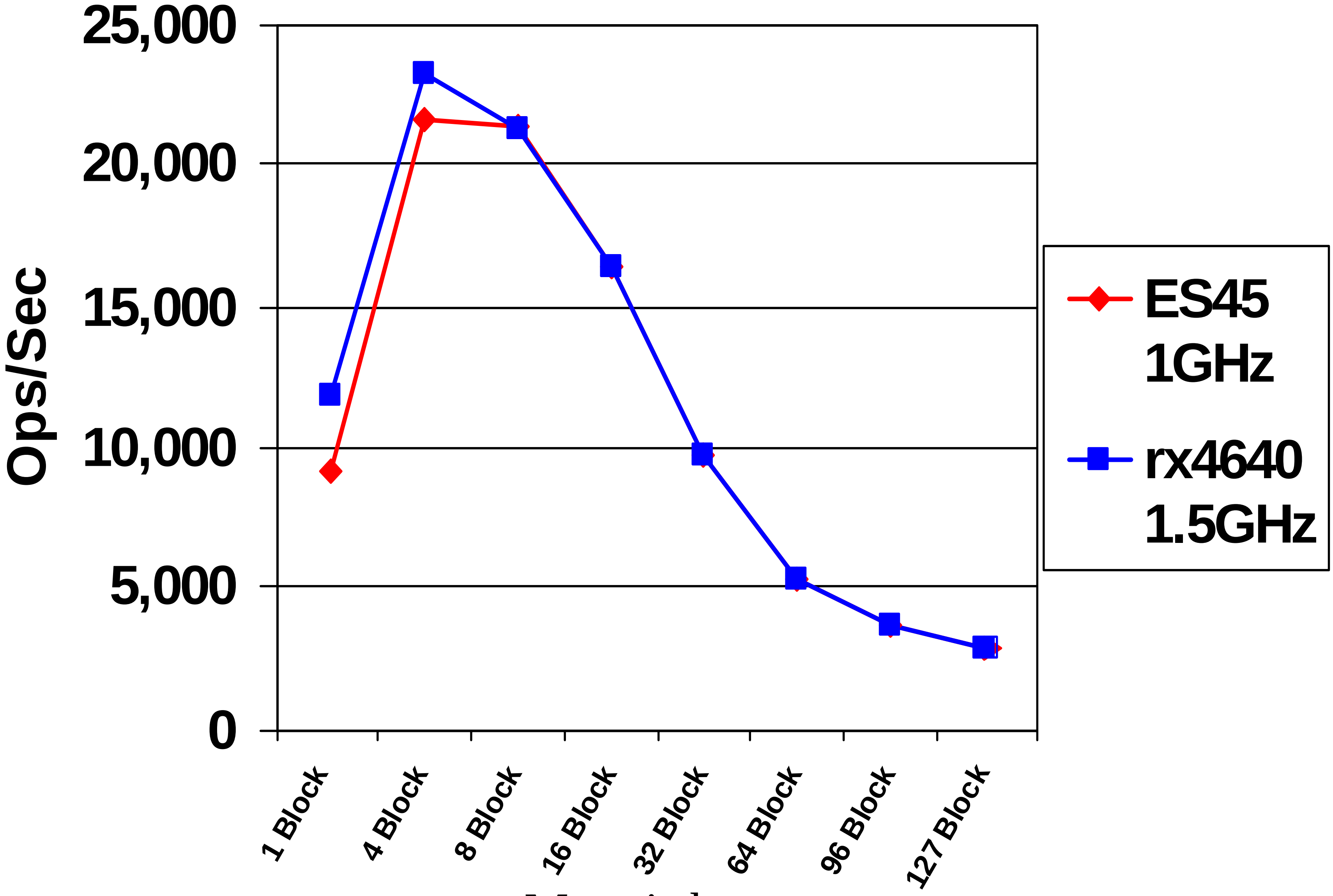
Memory Latency



Less is better

IO Performance – single process

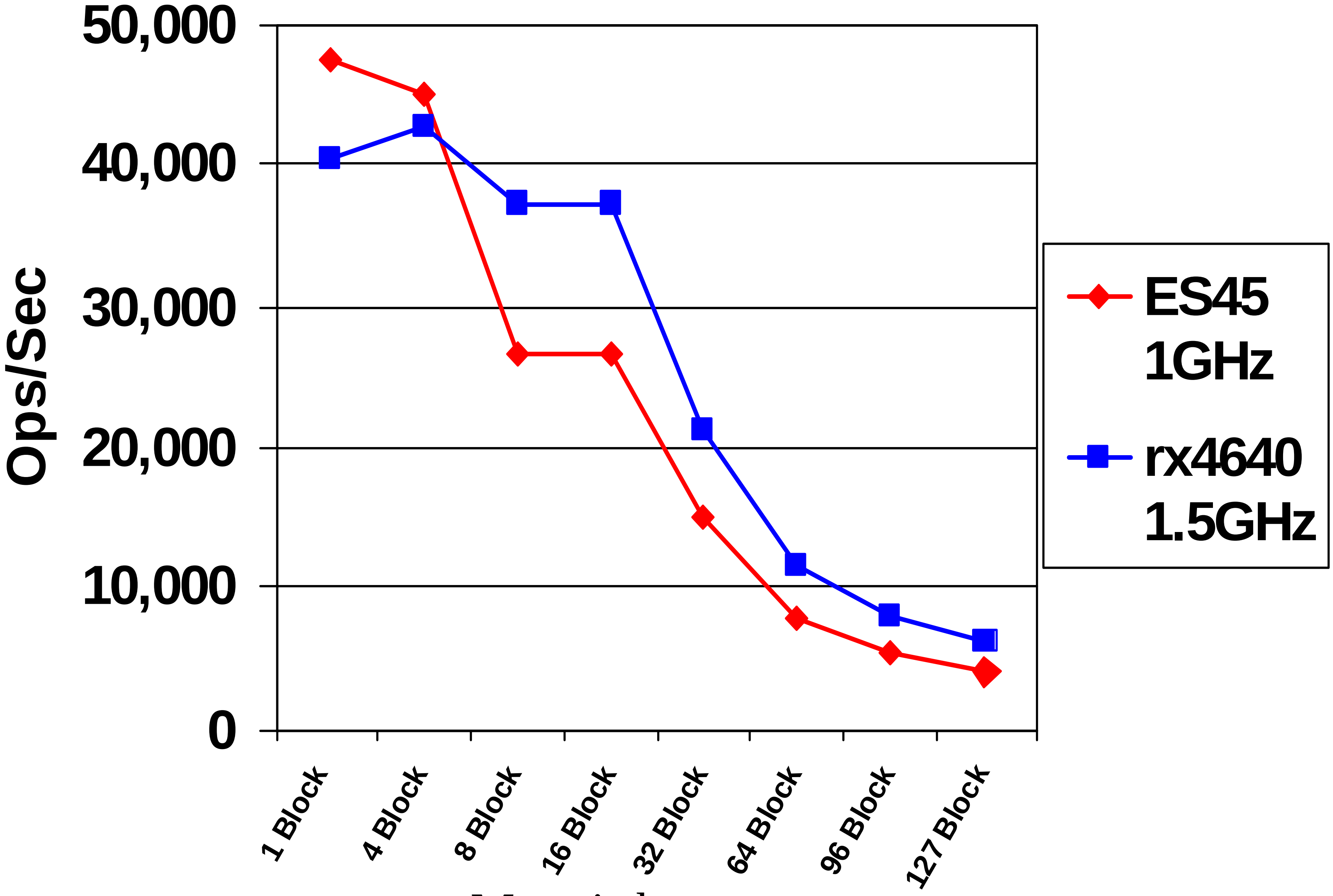
(QLogic ISP23xx) 2Gigabit Fiber Channel Card



More is better

IO Performance – two processes

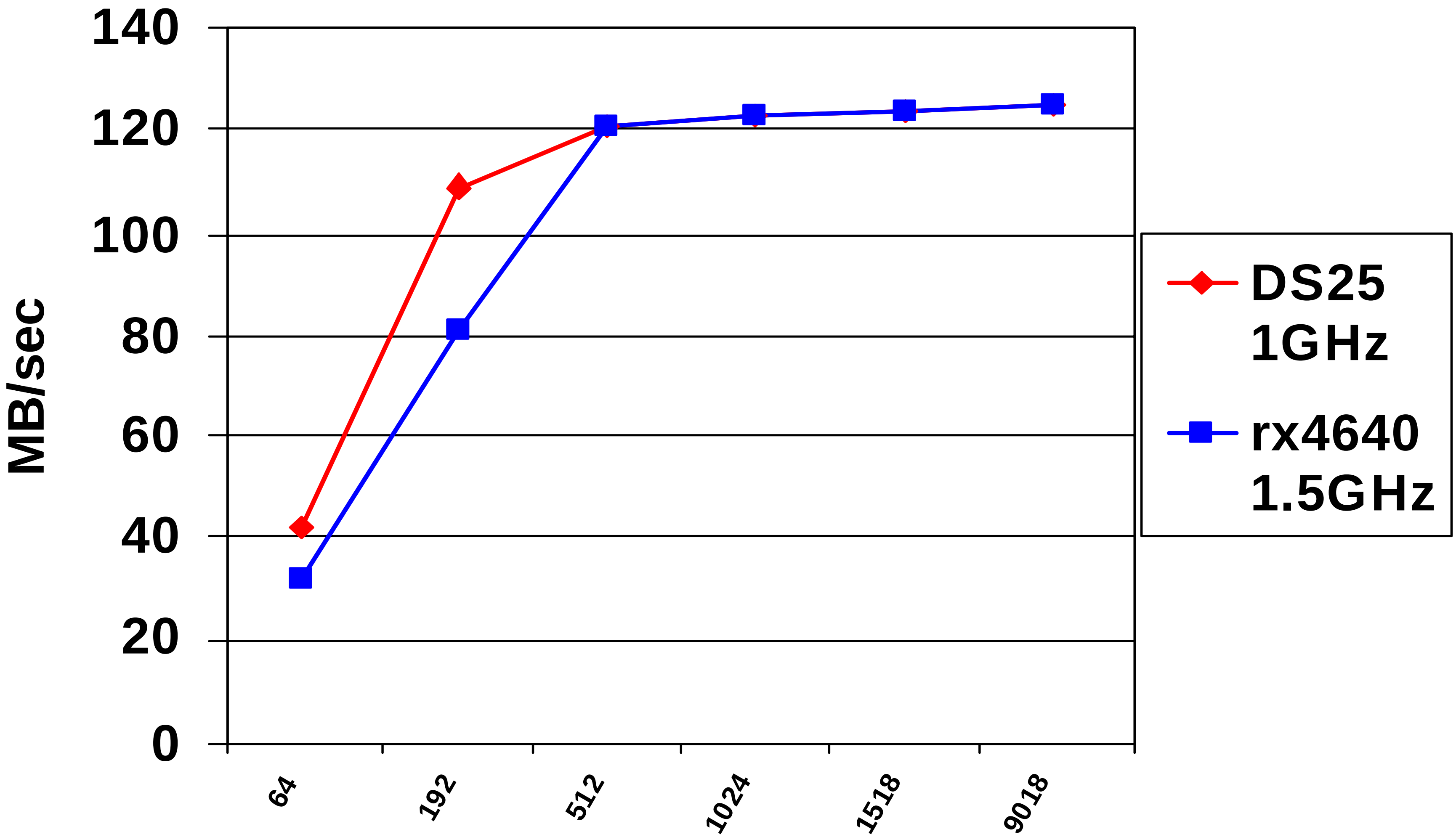
(QLogic ISP23xx) 2Gigabit Fiber Channel Card



More is better

Gigabit Transmit MBytes/Sec

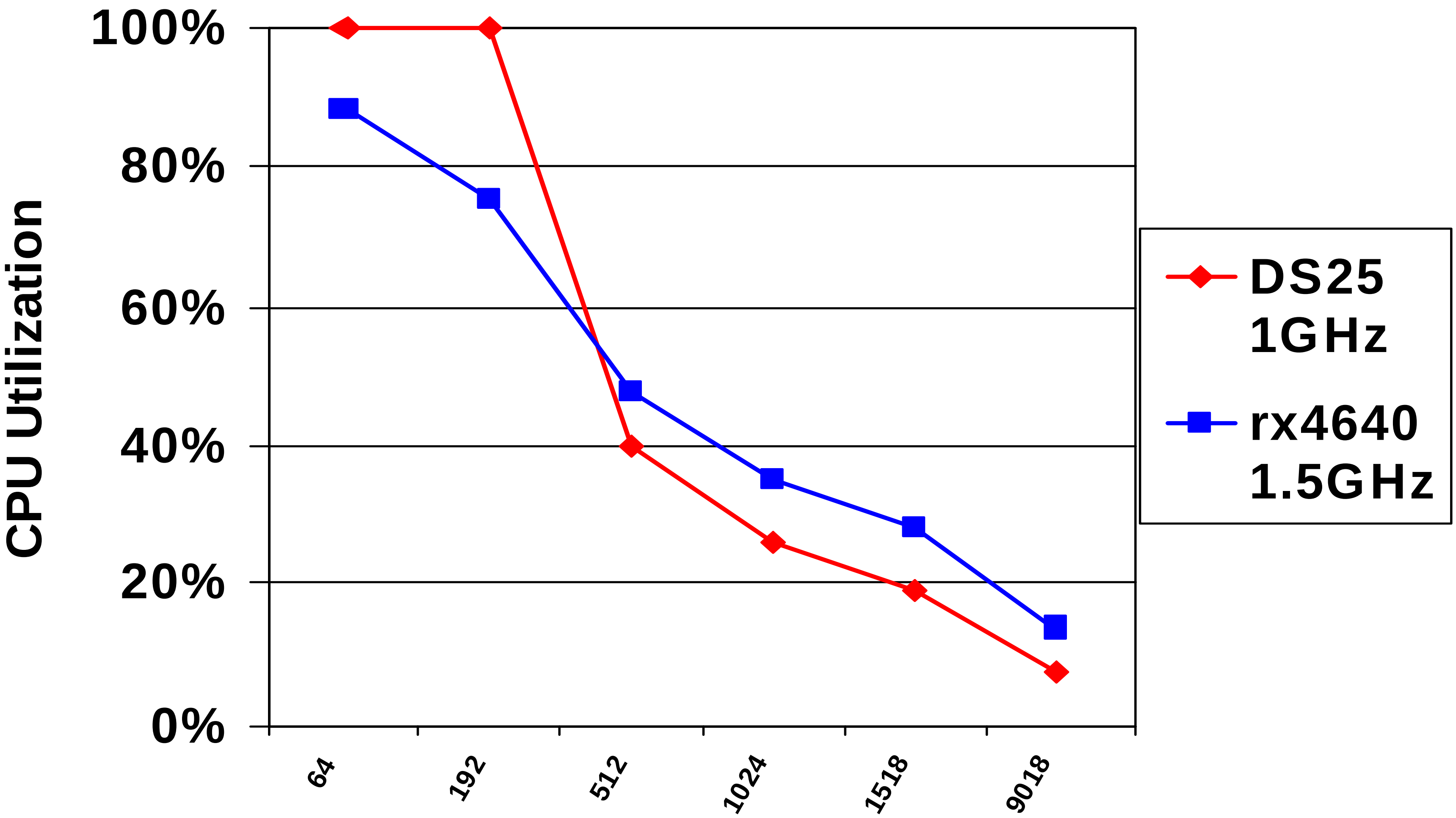
rx4640 1.3GHZ (A6825A - Broadcom 5701) in 64-bit PCI @ 66 mhz
DS25 1GHz (DEGXA - Broadcom 5703) in 64-bit PCI @ 66 mhz



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

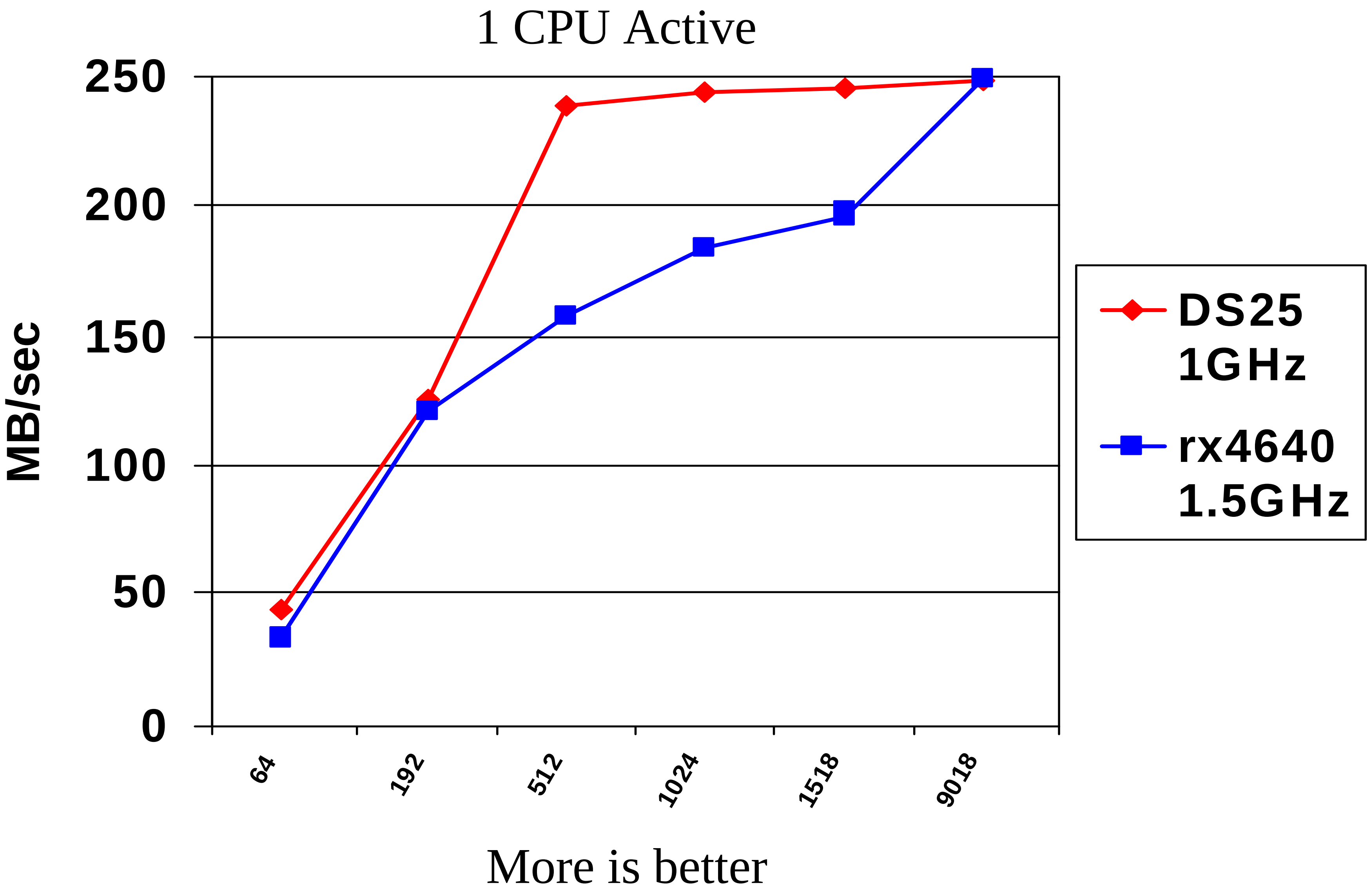


Less is better

Gigabit Transmit/Receive MBytes/Sec

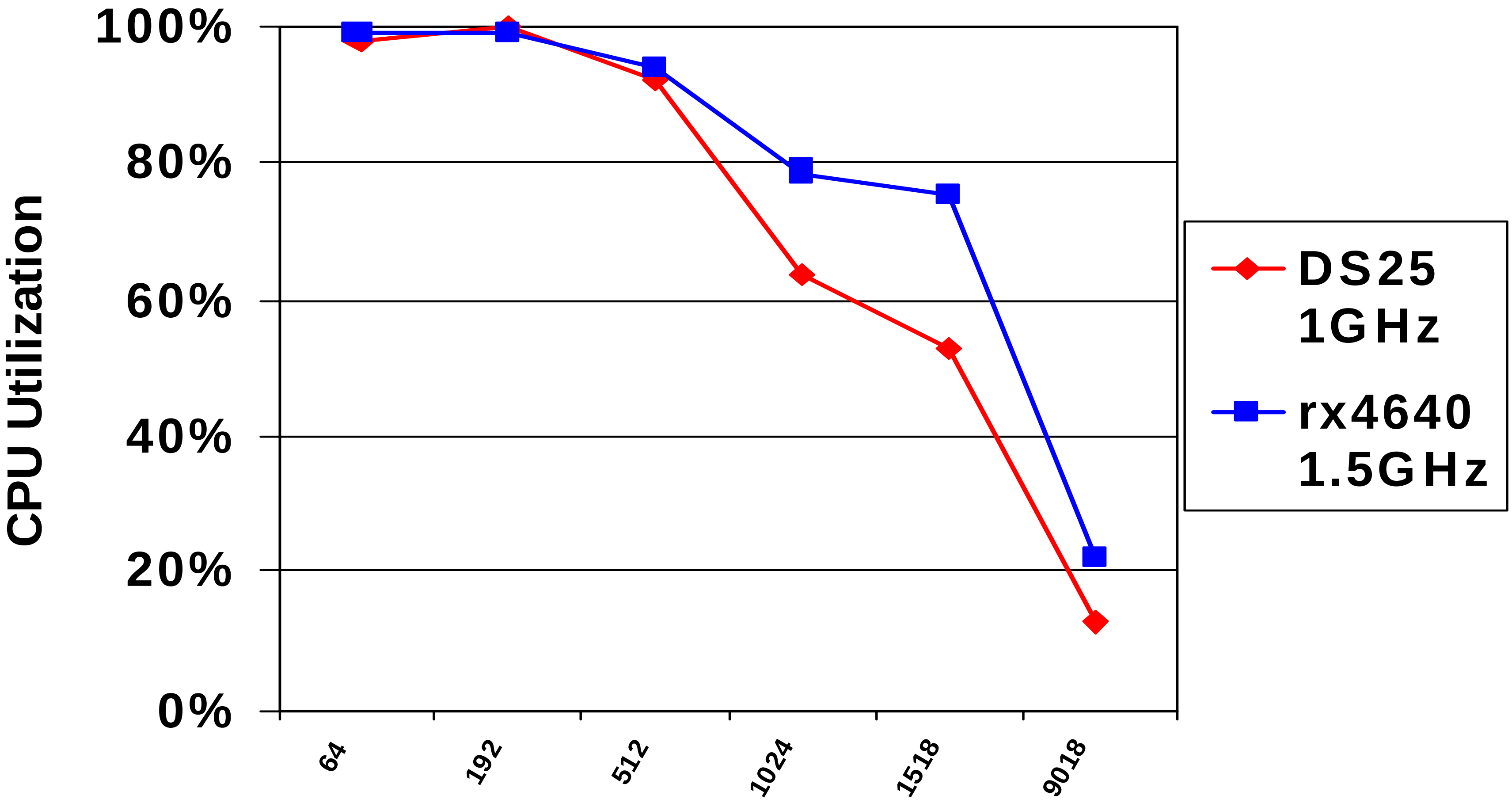
rx4640 1.3GHZ (A6825A - Broadcom 5701) in 64-bit PCI @ 66 mhz

DS25 1GHz (DEGXA - Broadcom 5703) in 64-bit PCI @ 66 mhz



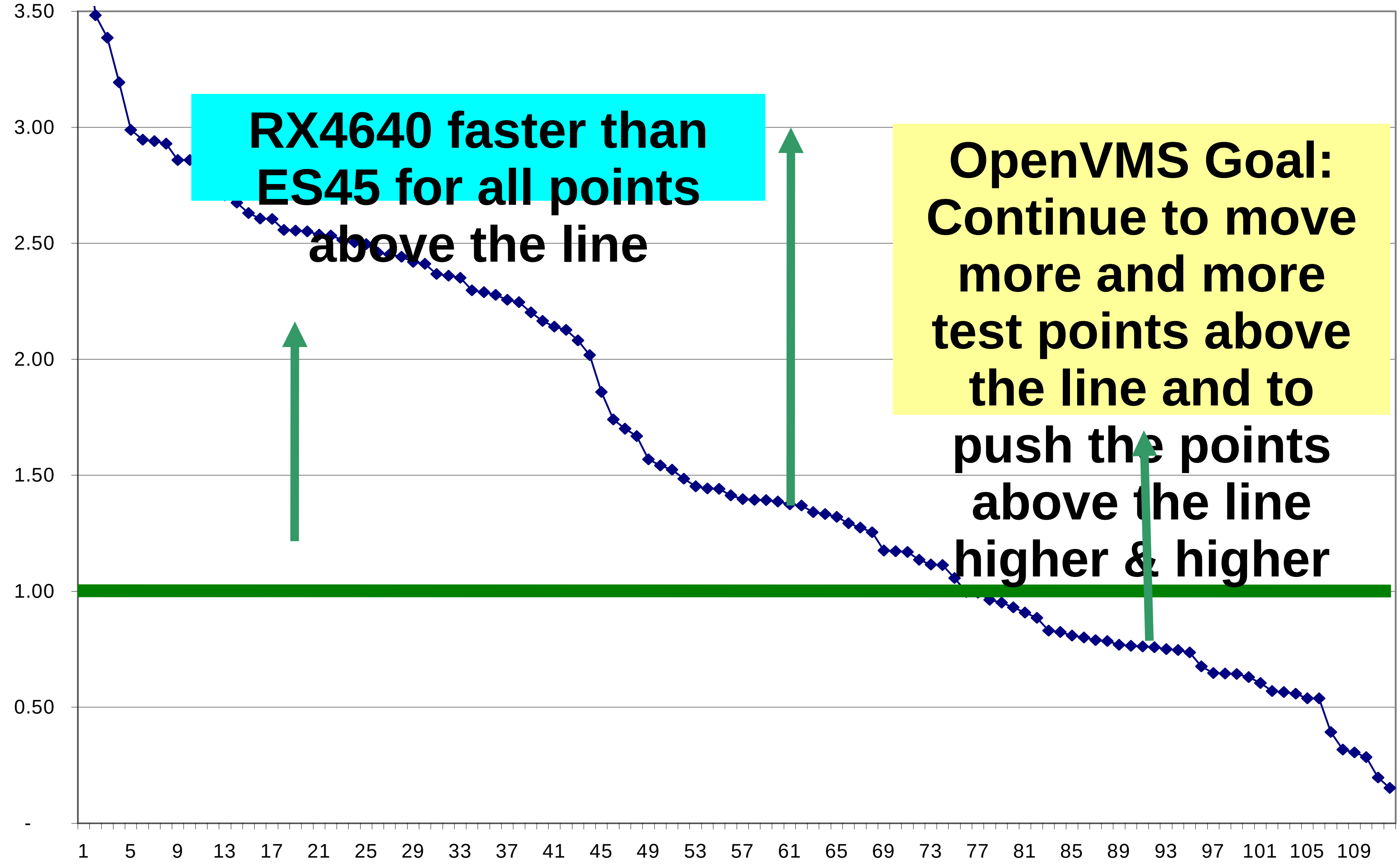
Gigabit Transmit/Receive CPU Utilization

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

System Services – X8.2 vs. 7.3-2

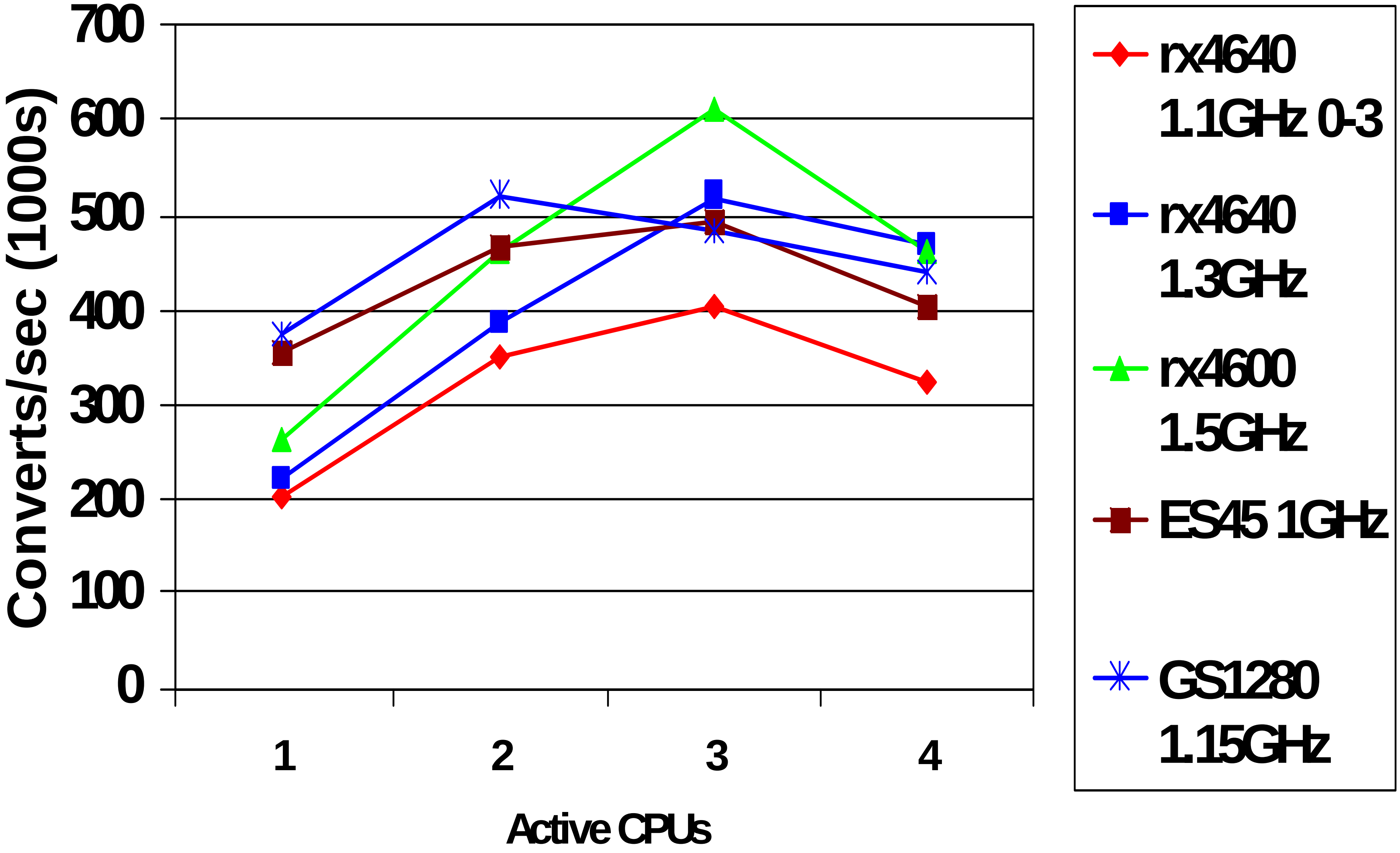


RX4640 faster than ES45 for all points above the line

OpenVMS Goal: Continue to move more and more test points above the line and to push the points above the line higher & higher

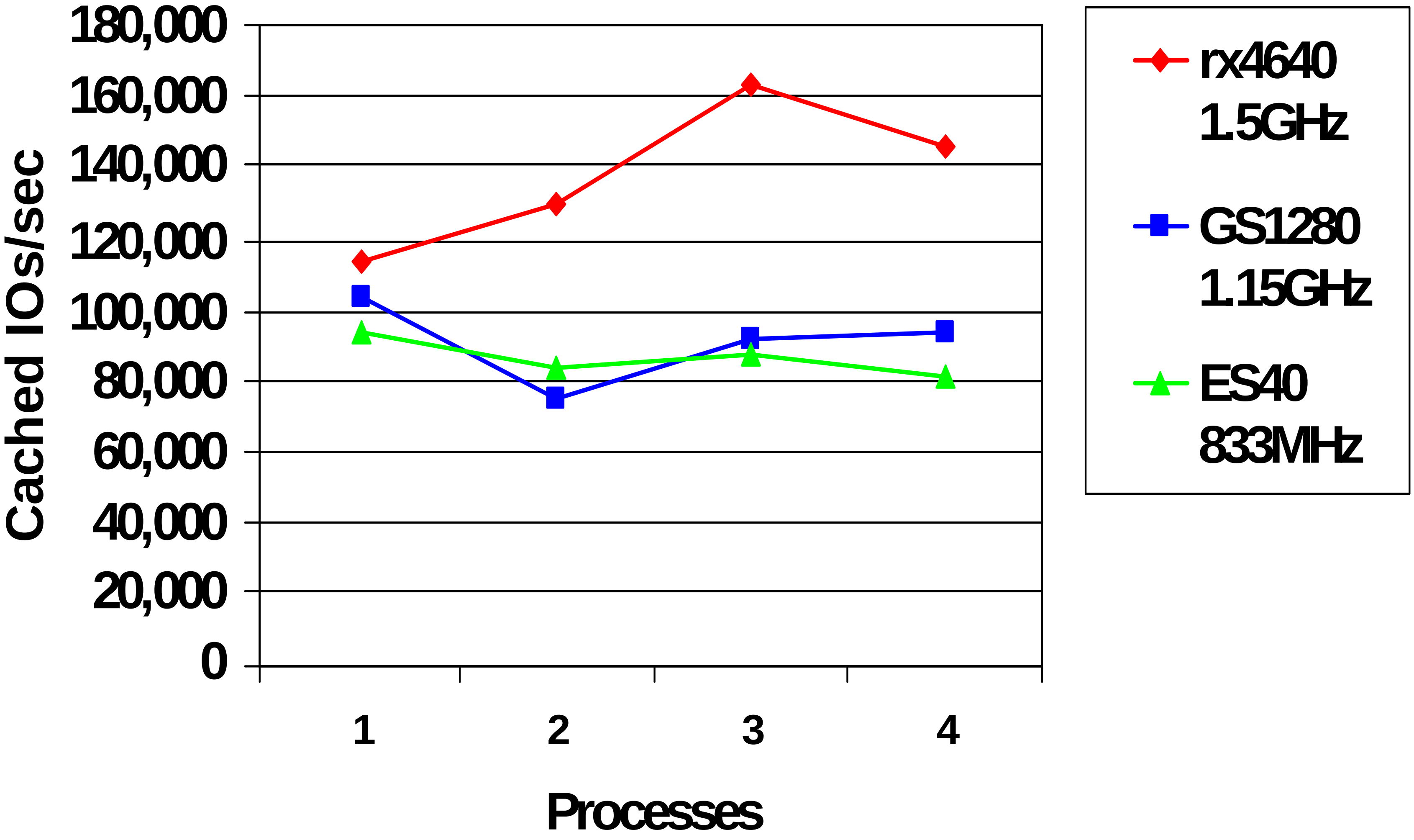
Lock Manager Stress Test

4 Processes



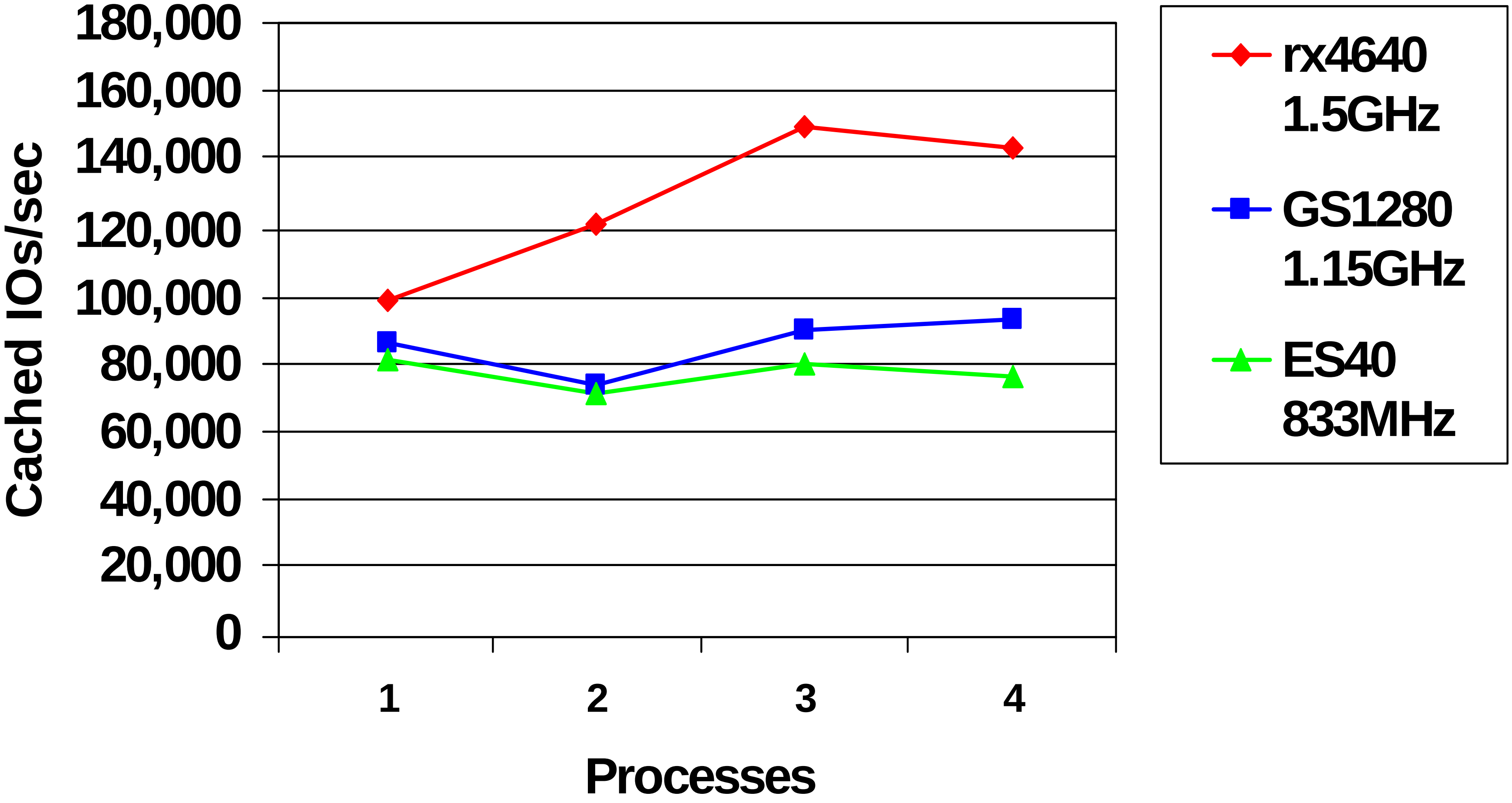
More is better

XFC Cached 1 Block IOs



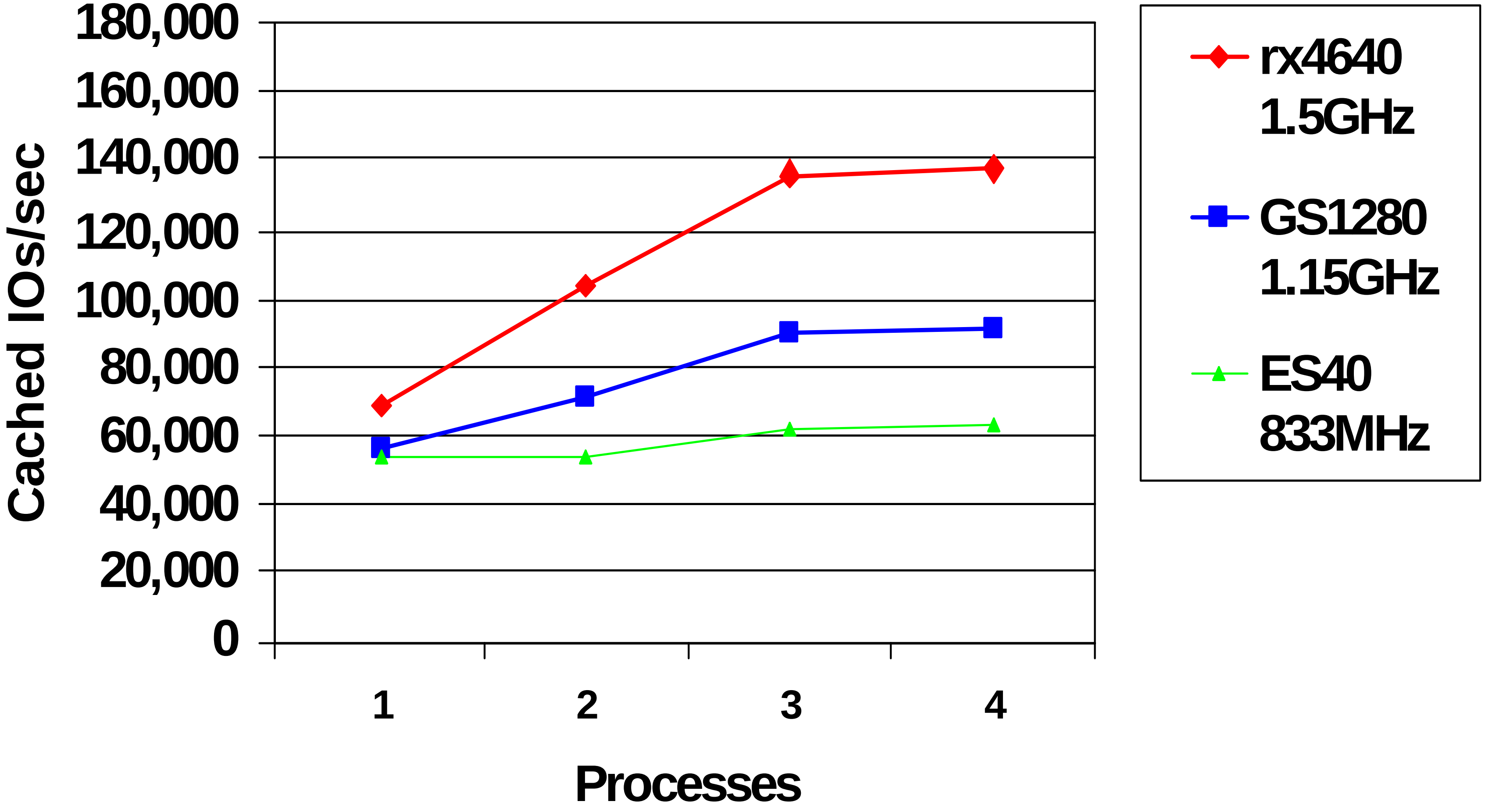
More is better

XFC Cached 4 Block IOs



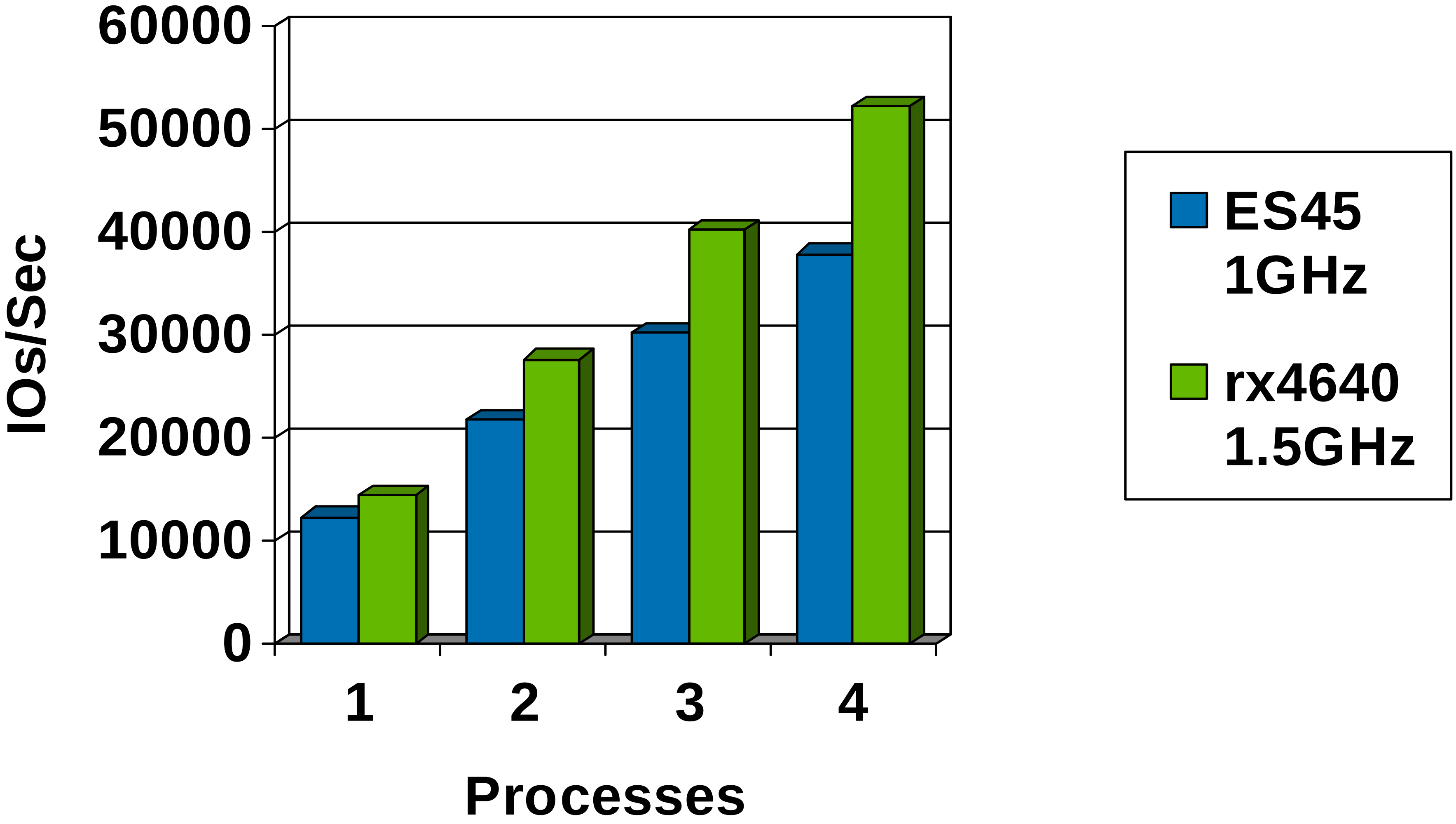
More is better

XFC Cached 16 Block IOs



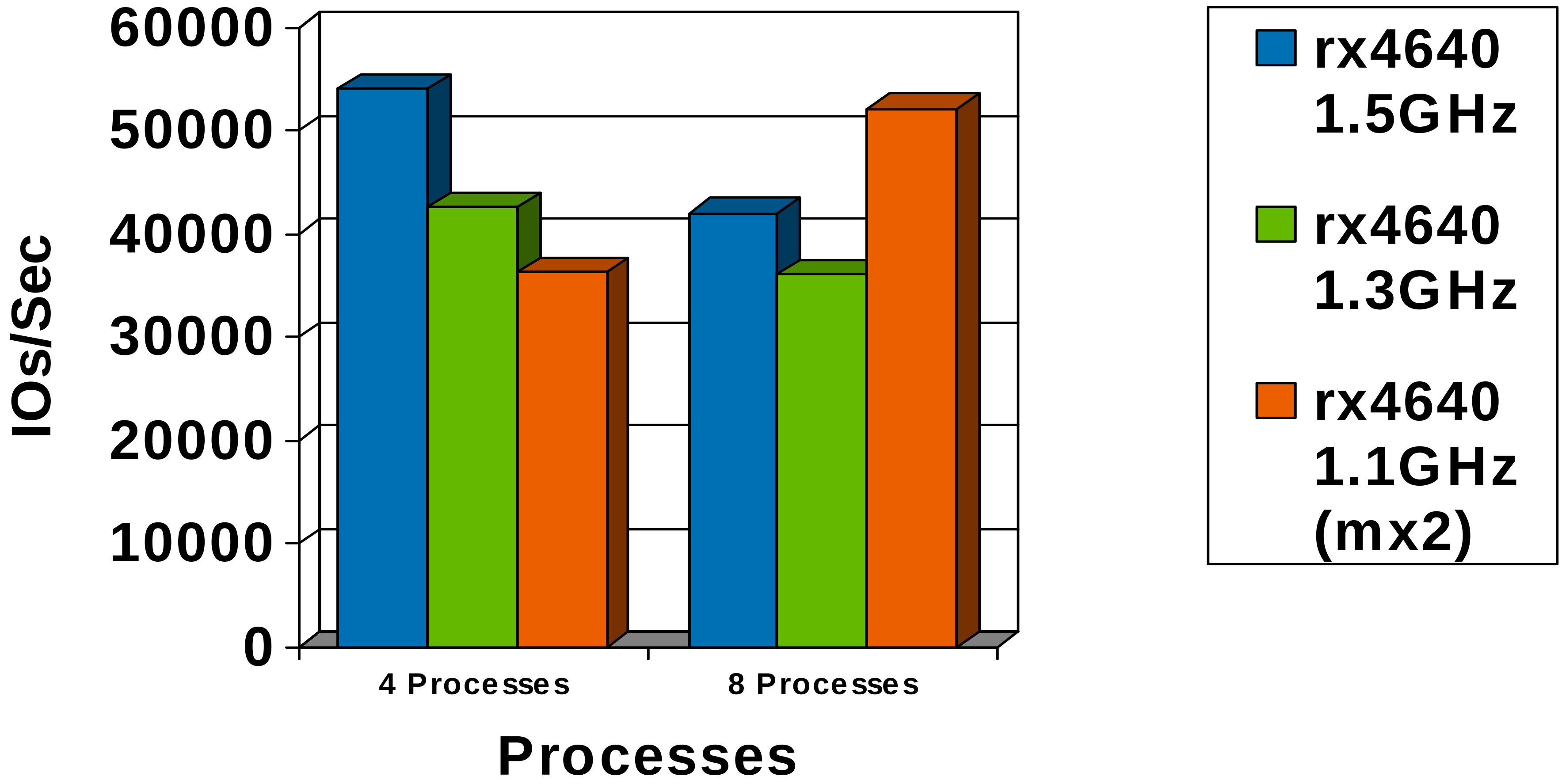
More is better

RMS1 (RAMdisk)



More is better

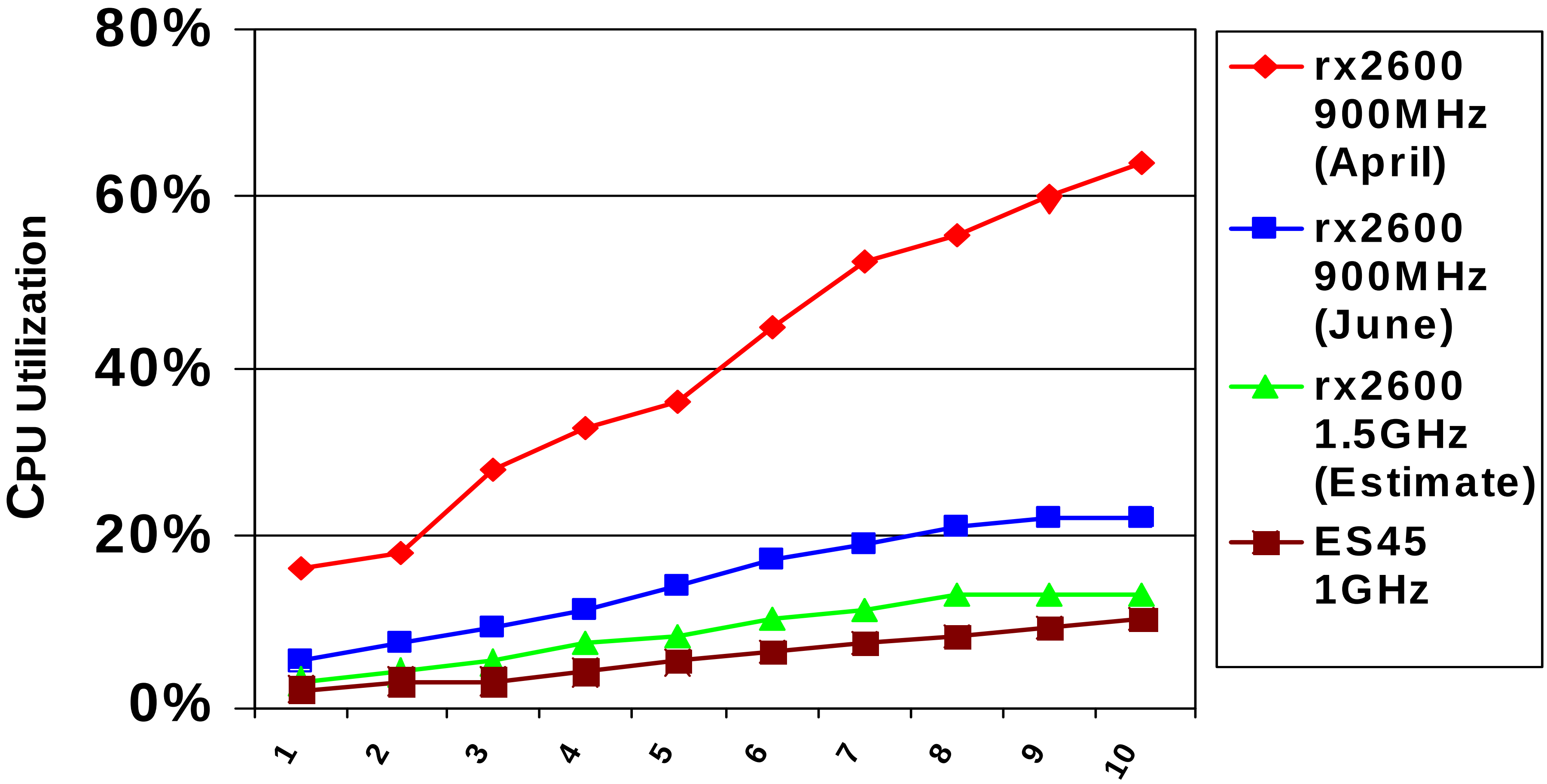
rx4640 vs. rx4640-8 (mx2 module)



More is better

OpenVMS InfoServer CPU Usage

(Advanced Development Project)



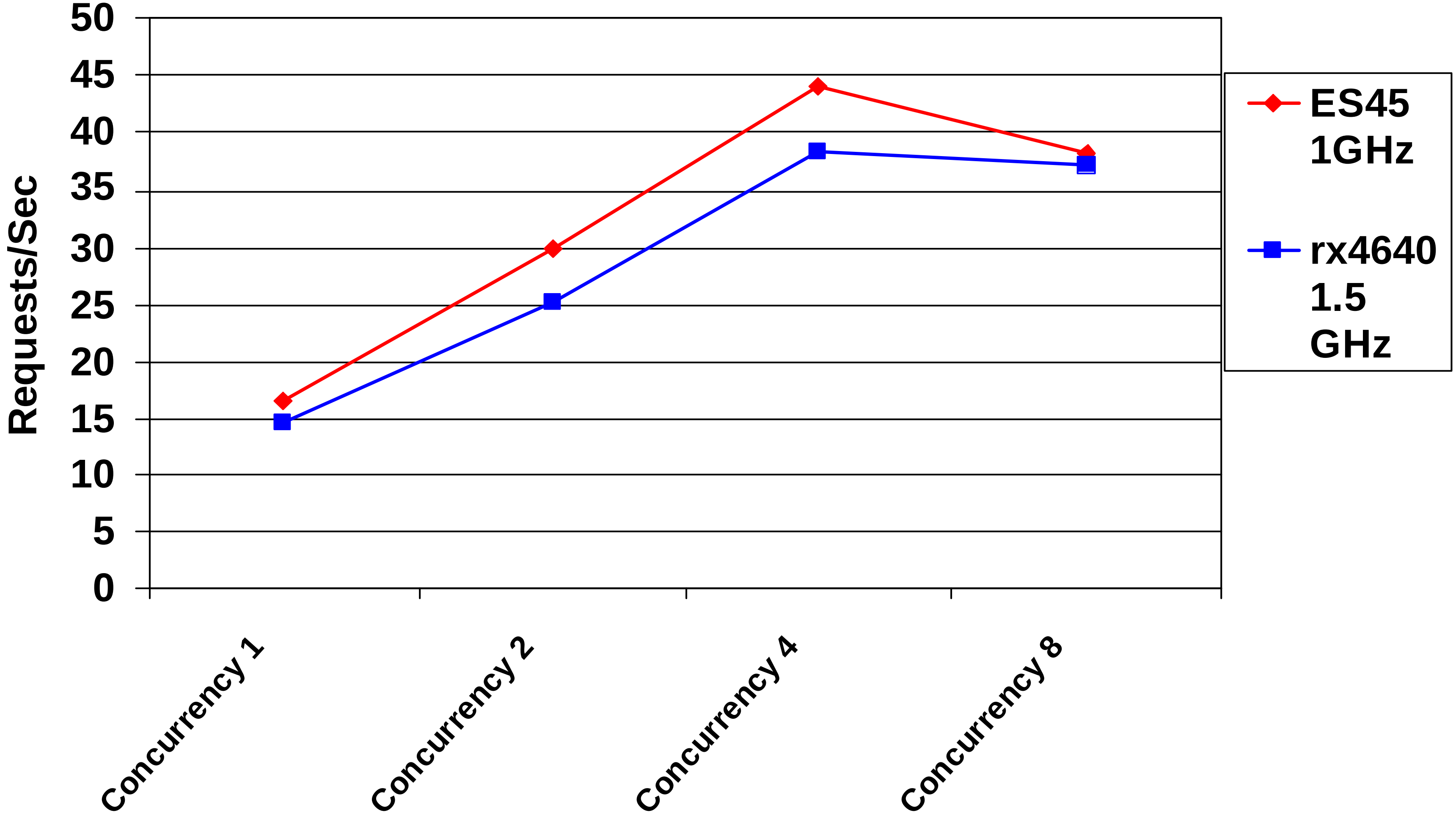
Active Client Loads

Backup/Image from InfoServer to local device

Less is better

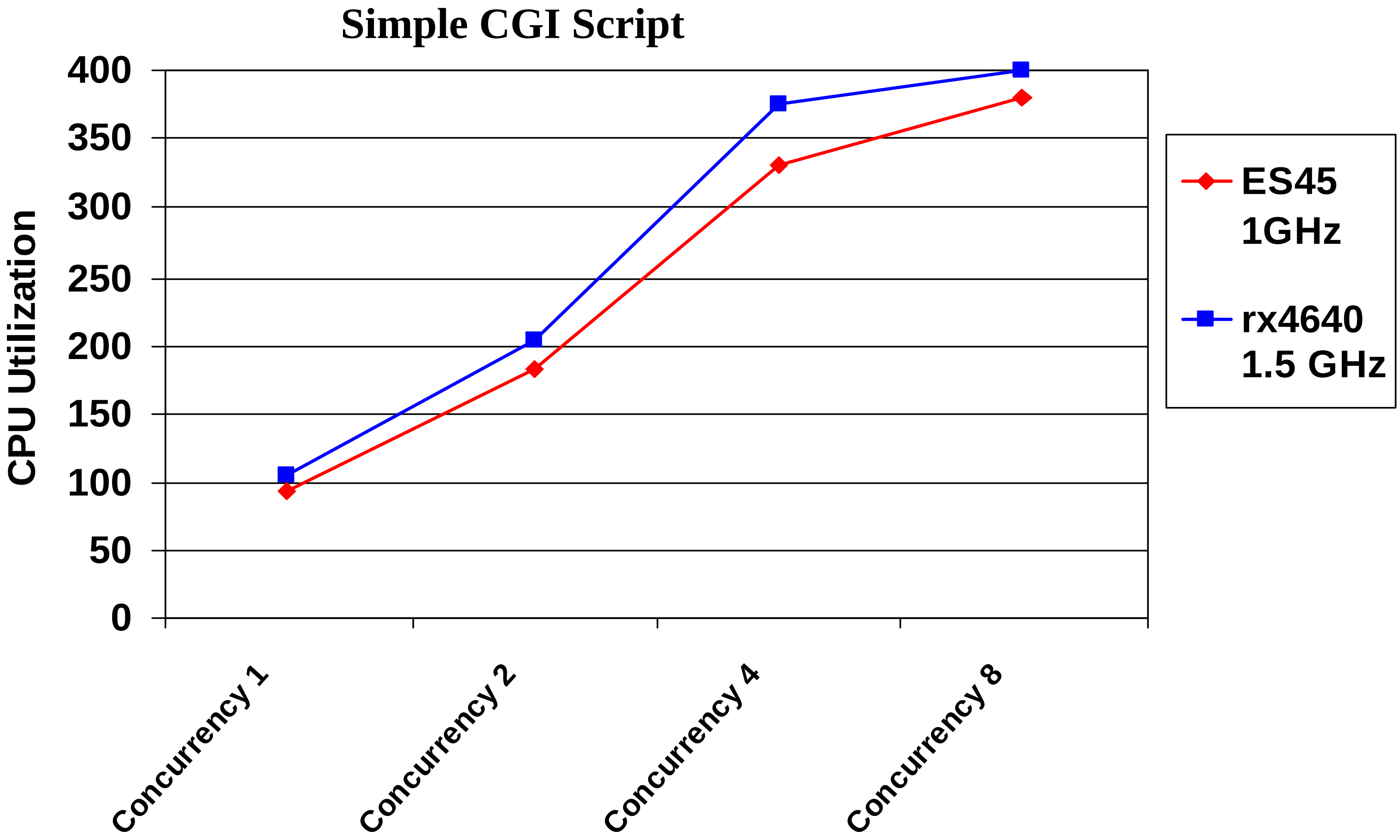
Apache Requests Per Second

Simple CGI Script



More is better

Apache – CPU Utilization



Less is better

Various Improvement Successes

- Heavy IO loads to disks performed very poorly
  - rx2600 2,084 IOs/sec
- Spinlock Analysis showed VERY heavy usage of MMG for the IPF which didn’t appear for the Alpha
- Further Analysis showed IPF was never caching any KPB structures.
- One line fix ->
  - rx2600 10,999 IOs/sec

OTS$MOVE and OTS$MOVEM

- OTS$MOVE and OTS$MOVEM are low level routines called by compilers to move data.
  - Macro calls this for MOVC3 and MOVC5 instructions
  - C calls this routine for memcpy
  - BLISS call this for ch$move
- Highly optimized versions of these routines have recently been integrated into OpenVMS
- This resulted in significant performance improvements for tests that did heavy memory copies
  - The RMS1 test improved by about 15% for single stream and by about 38% for 4 streams!

Queue Instructions

- The various VAX architecture queue instructions were initially implemented as system services
  - These needed to be done in Kernel mode to insure the operation was atomic
  - We knew they would be slow and they were as shown by a small test program doing insque/remque in loop.

ES45: 0:05.21 rx2600: 2:58.21 (34 times slower)

- The implementation of the queue instructions has been changed to no longer use the system service dispatcher
- They now use the EPC (Enter Privileged Code)

rx2600: 0:14.59 (< 3 times slower)

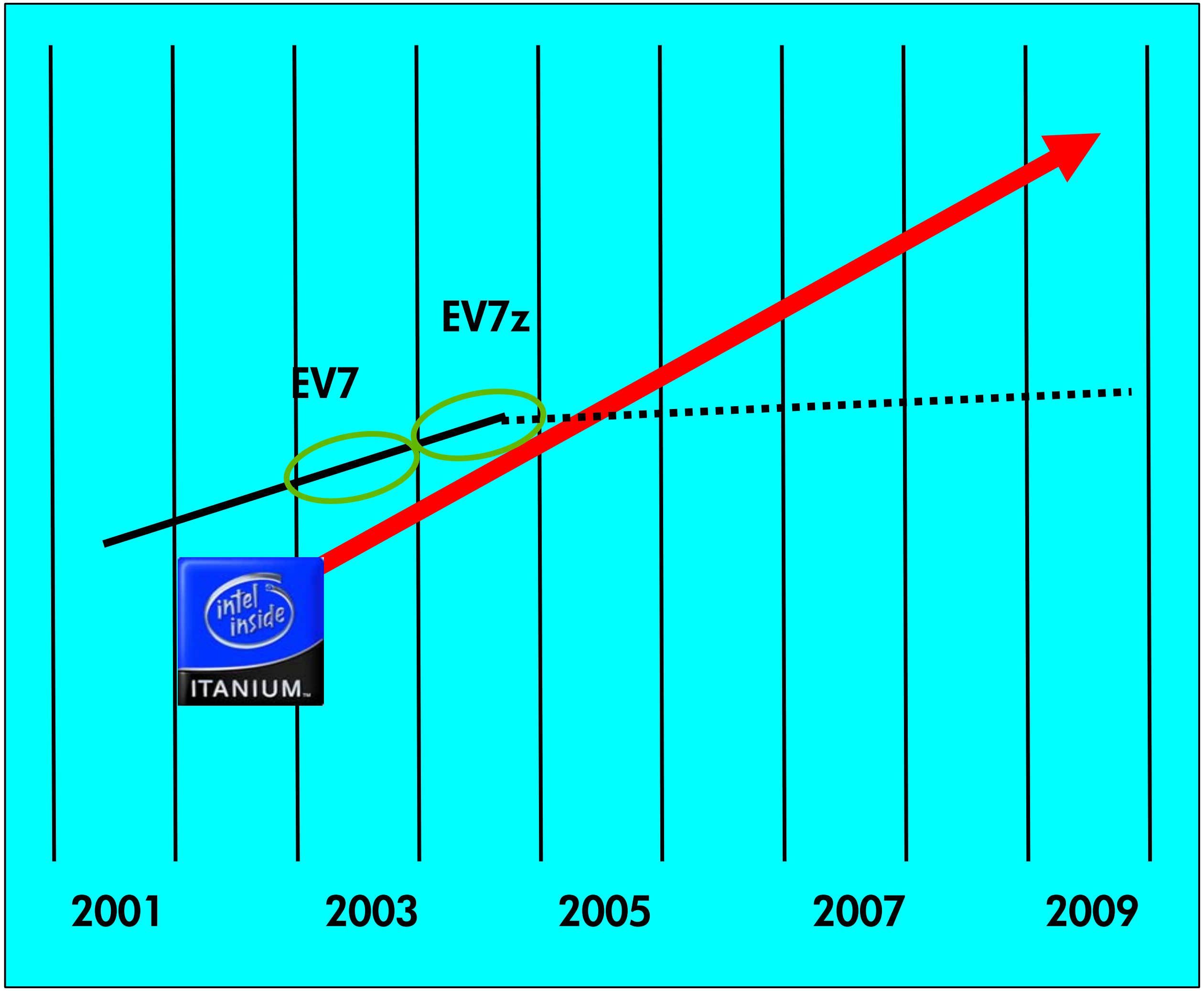
Areas that are Slower on IPF

- There are several areas where the equivalent operations on IPF systems are slower
  - Queue Instructions
  - Various PAL calls which now go through the system service dispatcher
  - Exception Handling
    - Exceptions Frames much larger
    - Finding Exception Handlers takes longer
- Images also are typically 3 times as large
  - This can impact image activation time
  - Requires More IO
  - Increase page faults

Projected Performance Crossover Point predicted two years ago

OpenVMS on

ITANIUM



Conclusions

- 1.5GHz rx4640 Integrity systems perform similarly to 1GHz ES45 Alpha systems
- There will continue to be improvements in both the OS and Compilers prior to the release of OpenVMS V8.2
- OS improvements coupled with future hardware speed ups will allow OpenVMS on IPF to out perform OpenVMS on Alpha

hp
invent