

OpenVMS AlphaServer GS Series Systems Performance Session 2A05

John Covert, OpenVMS Engineering
April 2002

DECUS-München 2002 Bonn

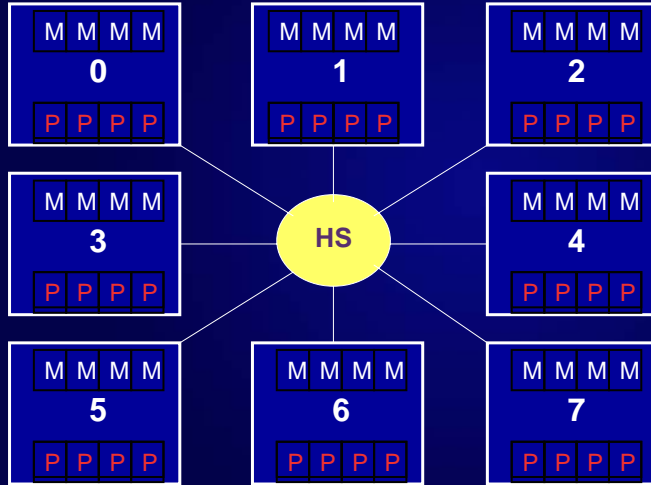
Updated from a presentation by Greg Jordan at CETS September 2001

People, Knowledge, Innovation ... Success!

AlphaServer GS Systems

- System is a set of quad building blocks (QBBs)
- A fully populated QBB contains
 - 4 CPUs
 - 4 memory modules
 - 4 I/O hoses
- GS80 - 2 QBBs
- GS160 - 4 QBBs
- GS320 - 8 QBBs

GS320 = 8 QBBs



1 QBB

4 CPUs

32 GB

4 hoses

System

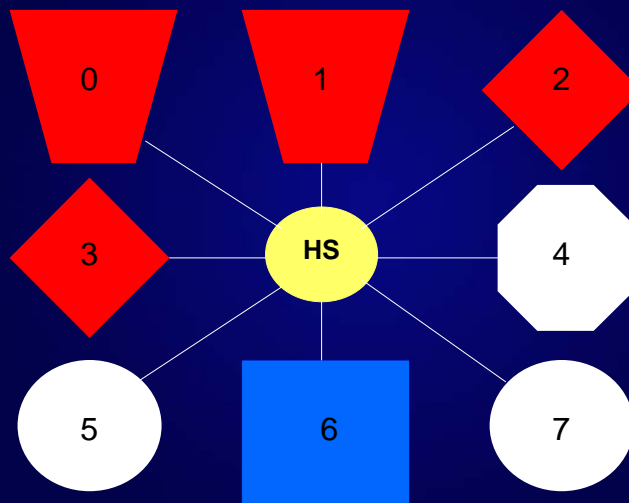
32 CPUs

256GB

32 Hoses

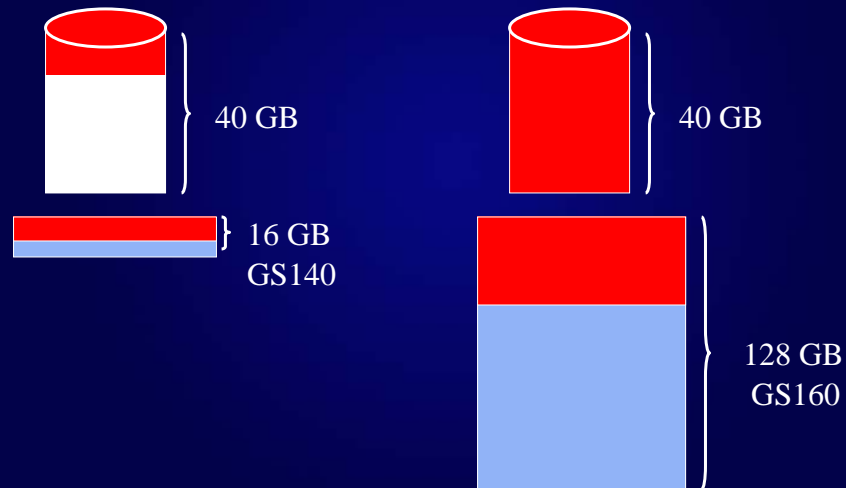
3

Soft Partitions within Hard Partitions



4

Large Databases in Memory



NUMA - Why is it different?

- Non-Uniform Memory Access (NUMA) is an attribute of a system in which access time to any given physical memory location is not the same for all CPUs.
- Given this architecture, you must have consistently good location (but not necessarily 100% of the time) for high performance.
- The performance decision - fairness for all vs. peak for a few.

NUMA and Performance

- It is impossible to make generalized comparisons (e.g. GS140 vs. GS160), except

“If you do not scale well on the GS140 you will not scale well on the GS160.”

- Each application environment is different
- Application’s structure dictates performance options

7

Location, Location, Location

- Where is the code?
- Where is the data?
- Where is the I/O device?

8

Partitioning the Workload

Dedicating a subset of the system's resources to do a specific task

- Hard partitions
- Soft Partitions (Galaxy instances)
- CPU affinity or CPU capabilities
- Dedicated CPU lock manager
- Specifying process and/or data RAD location
- Fastpath device drivers

9

Variables to Consider

- number of processes
- predictability of processes' data access
- memory requirements
- amount of "sharing" between processes
- use of certain OpenVMS features
- application "locks" and their location
- I/O requirements

10

Resource Affinity Domain (RAD)

- A RAD is a software grouping of physical resources that have similar access characteristics.
- GS80/160/320 - A quad-building block (QBB) is seen as a RAD by OpenVMS when a single instance of OpenVMS runs on multiple QBBs.

11

OpenVMS and NUMA

- OpenVMS V7.2-1H1 contains NUMA support
 - Support for RADs
 - Process is scheduled in a RAD
 - Process Private memory allocated from the RAD
 - Code Replication
 - NUMA Aware APIs

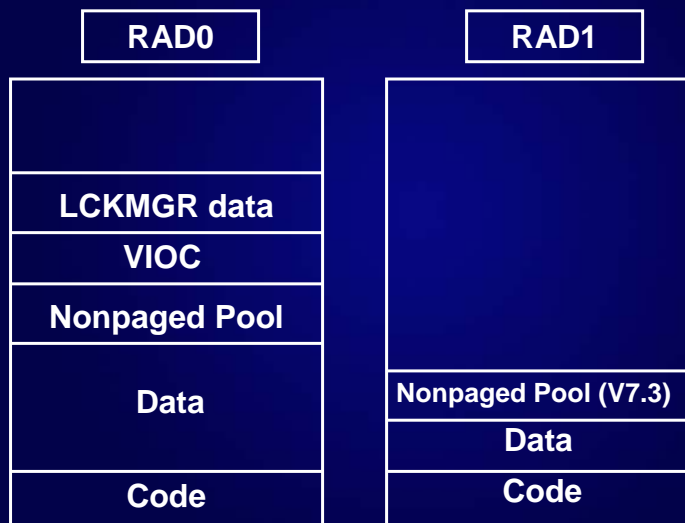
12

NUMA-aware APIs

- Reserved memory for a global section can be specified in per-RAD chunks
- Find, set initially, and change the home RAD of a process
- Placement of global section pages
- SYS\$GETSYI - RAD configuration info
- System parameter RAD_SUPPORT - many control options
- Programming examples on web

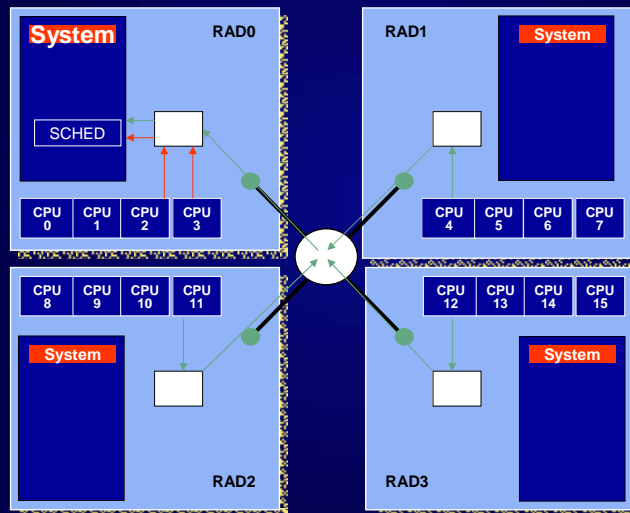
13

Exec Layout on NUMA Systems



14

Spinlocks



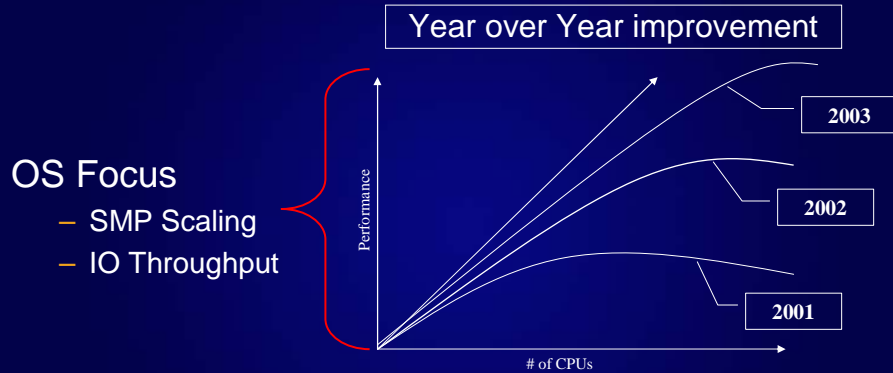
15

Spinlocks

- Need to compensate for hardware unfairness in order to make spinlock acquisition reasonably fair
- Traditional acquisition algorithm behavior until high lock contention is detected
- If contention is low then RAD0 has some advantage
- Testing has shown it to be extremely fair during high contention
- Not just for NUMA; it is for all platforms
 - single bus systems sometimes exhibit unfairness
 - mixed CPU speeds

16

VMS Performance Futures



Projects to improve I/O Subsystem:

- Finer Grain Locking
- Distributed Interrupts
- Improve locking for TCP/IP and PEdriver

Performance Improvements in V7.3 and V7.2-2

- General
 - Dedicated-CPU lock manager
 - Process scheduling, idle loop
 - MUTEX without SCHED spinlock
 - SYS\$RESCHED (used by DECthreads and Oracle)
 - SYS\$GETJPI
 - MailBox driver
- I/O
 - Fibre fastpath (V7.3)
 - SCSI fastpath (V7.3)

Dedicated CPU Lock Manager

- Greatly reduced lock manager spinlock contention
- \$ENQ/ \$DEQ
 - creates request packet in a queue
 - process spins at IPL2 for completion status
- LCKMGR_SERVER process
 - hard CPU affinity to a single CPU
 - excellent use of CPU cache
 - services requests and returns status
- LCKMGR_MODE dynamic system parameter

19

Example #1 - Unpredictable Users

- Overall I/O
 - High disk traffic (fibre)
 - High cluster traffic (CI)
- High lock manager use
- Unpredictable individual user needs
 - CPU time
 - memory access
 - I/O

20

I/O

- Put the I/O in RAD0
 - 2 fibre channel adapters for storage
 - CI adapter for cluster traffic

- \$ SET DEVICE /PREFERRED_CPU=1 FGA0:
- \$ SET DEVICE /PREFERRED_CPU=2 FGB0:

- \$ SET DEVICE /PREFERRED_CPU=3 PNA0:

21

Dedicated CPU Lock Manager

- Make it a close partner of the CI driver to take advantage of CPU's cache

```
SYSGEN> set LCKMGR_MODE n ; dedicated mode
```

```
SYSGEN> set LCKMGR_CPUID 3 ; CPU 3
```

n = dedicated mode if there are n active CPUs

22

Client Processes in RADs 1, 2, and 3

- Process given a Home RAD of zero need to be moved
- In SYLOGIN.COM

```
$ if f$getjpi("",HOME_RAD) .eq. 0
$ then
$   new_rad = random number 1 to 3
$   set proc/rad=home='new_rad
$ endif
```

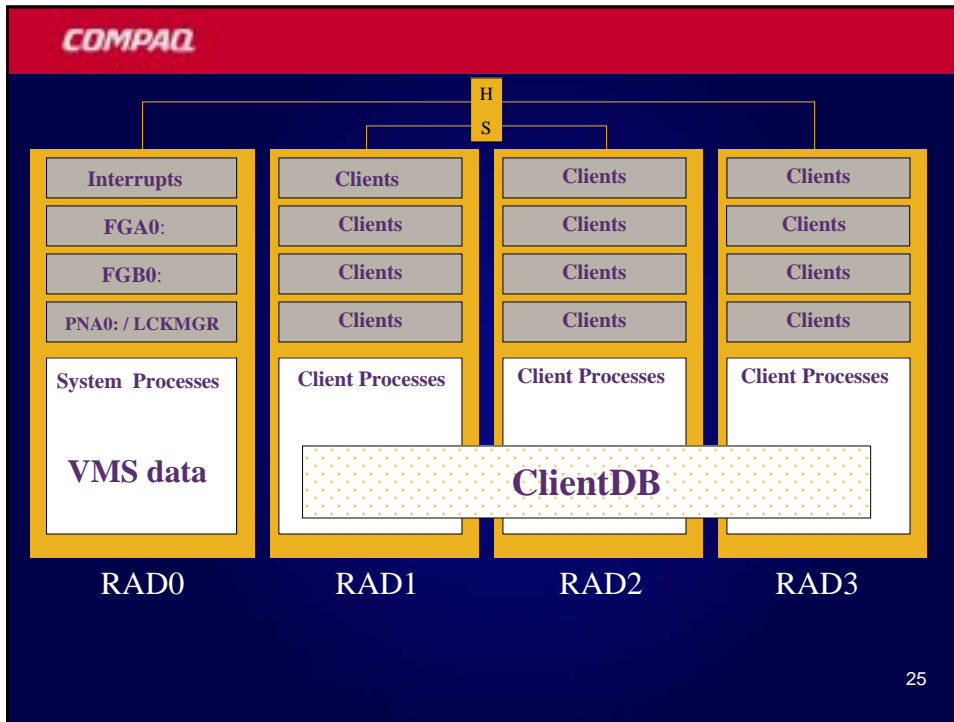
23

Database in RADs 1, 2, and 3

- SYSMAN> reserved_memory add "ClientDB" -
/page_tables /zero /size=1000 /RAD=1
- SYSMAN> reserved_memory extend "ClientDB" -
/size=1000 /RAD=2
- SYSMAN> reserved_memory extend "ClientDB" -
/size=1000 /RAD=3

ClientDB will be *huge page striped* over the RADs.

24



- COMPAQ**
- ## Unused Cycles?
- Should CPUs 0-3 run client processes?
 - MONITOR MODES / CPU=(0,1,2,3)
 - Interrupt Stack + MP_synch time on CPUs less than 50%?
 - Overall use of CPUs 1-3 less than 75% busy?
- 26

Example #2 - Known User / Data Access

- Web server
- Heavy Network I/O
- Multiple databases
- 4 server processes
- Each server
 - accesses a particular database
 - has 4 threads

27

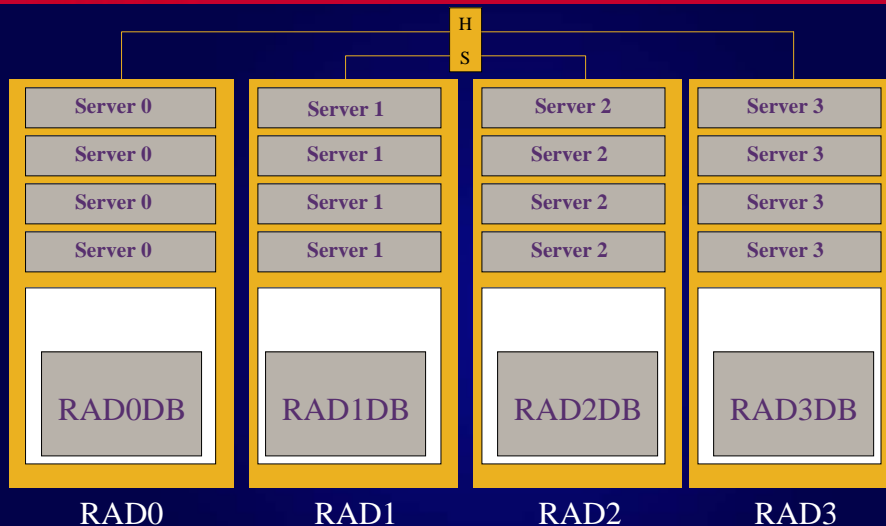
RAD-Specific Process Placement

- SYS\$CREPRC (stsflg=prc\$m_home_rad, rad)
or
- SET PROCESS /RAD=HOME=*rad_number*
- SYS\$CRMPSC_GDZRO_64 (gs_name_64="RADxDB",
flags=sec\$m_rad_hint, rad)
- SYSGEN> SET MULTITHREAD 4

28

Home RAD vs. CPU Affinity

- **CPU Affinity:** better use of CPU cache but only that one CPU will ever select the process to run
- **Home RAD:** less effective use of CPU cache but 4 CPUs will schedule the process to run



Example #3 - Time Constrained

- Financial customer has a nightly 1-2 hour window to produce analysis reports
- Run currently takes 50 minutes
- The run is made up of a variable number of analysis processes and a reporter process
- Data is numerous RMS indexed files, some R/W, some Read only, some created R/W

31

Current Environment

- GS140 12 CPUs, 8gig memory
- All data is on a RAMdisk (about 3gig)
- Current run takes about 50 minutes
- Data set grows over time and the length of run grows
- Typical tuning method: Buy faster hardware

32

Initial GS160 Tests

- Application tested on a 3 QBB, 12 CPU GS160 - 48gb memory
- Application runs slower than GS140
 - Why? Faster CPUs, newer hardware...
 - Performance Data shows:
 - heavy RamDisk IO - 15,000+ per second
 - heavy locking 150,000 operations per second - high mp_synch from the locking contention

33

Analysis

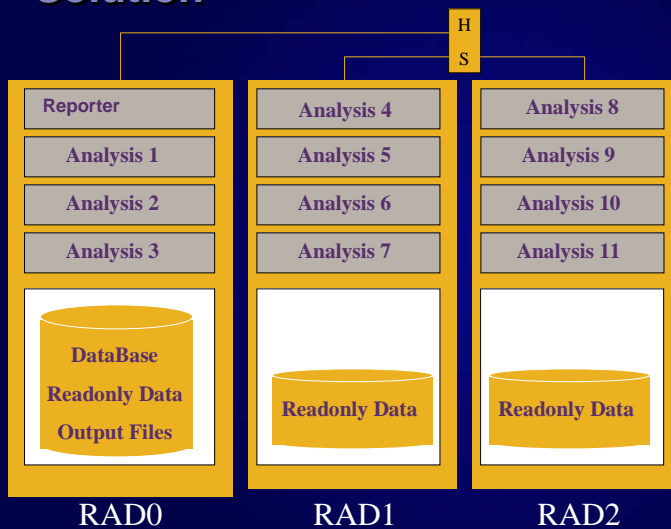
- Heavy Ramdisk IO means that 66% of the IO is cross QBB.
- Heavy locking results in OS interaction with most associated memory in RAD 0

34

Recommendations

- Ramdisk per QBB for Read Only data
 - Reduces cross QBB memory references
- Ramdisk on QBB 0 hold all R/W data
 - Allows final processing by a the single thread reporter job to have local access to all data
- Use Hard Affinity, Reporter in QBB 0
- Reduce Locking and thus OS interaction with RMS Global Buffer
 - Adding RMS global buffers to files reduces locking down to about 45,000 operations per second
 - This also reduces cross QBB memory references to OS data
- Got to 28 minutes.

Solution



Where are the problem areas?

- I/O interrupts?
 - Don't schedule any processes on CPU0?
 - SYSGEN> SET SCH_CTLFLAGS 3
- Mostly in User Mode?
 - application measurement tool
 - application parameters to adjust

37

High MPsynch?

- A new Spinlock trace tool can provide very detailed data about MP Synch time
- In V7.3 and V7.2-2, a command procedure is provided to collect data and generate a report
 - \$ @SYS\$EXAMPLES:SPL.COM ; during high MP_Synch activity
- Send output file to CSC, Account Rep, or Ambassador
- The command procedure was not distributed with V7.2-1H1, but will work on V7.2-1H1

38

MONITOR MODE

```

+-----+          TIME IN PROCESSOR MODES
| CUR |          on node DECRDB
+-----+          12-JAN-2001 09:00:06.64

Combined for 16 CPUs  0          400          800          1200          1600
+ - - - - + - - - - + - - - - + - - - - +
Interrupt State      47 |a
MP Synchronization  749 |aaaaaaaaaaaaaaaaaaaa
Kernel Mode          475 |aaaaaaaaaaa
Executive Mode       116 |aa
Supervisor Mode      21 |
User Mode            16 |
Idle Time            180 |aaaa
+ - - - - + - - - - + - - - - + - - - - +

```

39

Performance Improvements Beyond V7.3

- Scaling and NUMA Enhancements
- OpenVMS SMP & System Performance

Scaling and NUMA Enhancements

- Scaling Enhancements
 - Support for very large mailbox buffer quotas
 - Larger WSMAX or BALSETCNT due to reduced usage of SOS1 by the operating system
 - Removed pre-allocated kernel process data structures from the balance set slots (16 pages per)
- NUMA Enhancements
 - Some OS Read-only data is replicate on each RAD
 - Per RAD KPB Caches
 - Per RAD PCB Caches

41

OpenVMS SMP & System Performance

Faster system & application performance!

- Performance Improvements in the areas of:
 - AST Delivery
 - Mailbox IO
 - Timer Queue Processing
 - RMS Global Buffers Locking
 - Faster SYS\$GETJPI service used by CTRL
 - Improved Kernel Threads performance for multi-thread applications
 - Improved IO Completion:
 - RAMdisk, Mailbox I/O, Shadowing IO
 - Streamlined cluster communications paths

42

Summary

- NUMA Systems present new tuning and performance challenges
- What worked well on traditional bus based systems is not always optimal for a NUMA platform
- Good performance requires taking advantage of the NUMA architecture

43

