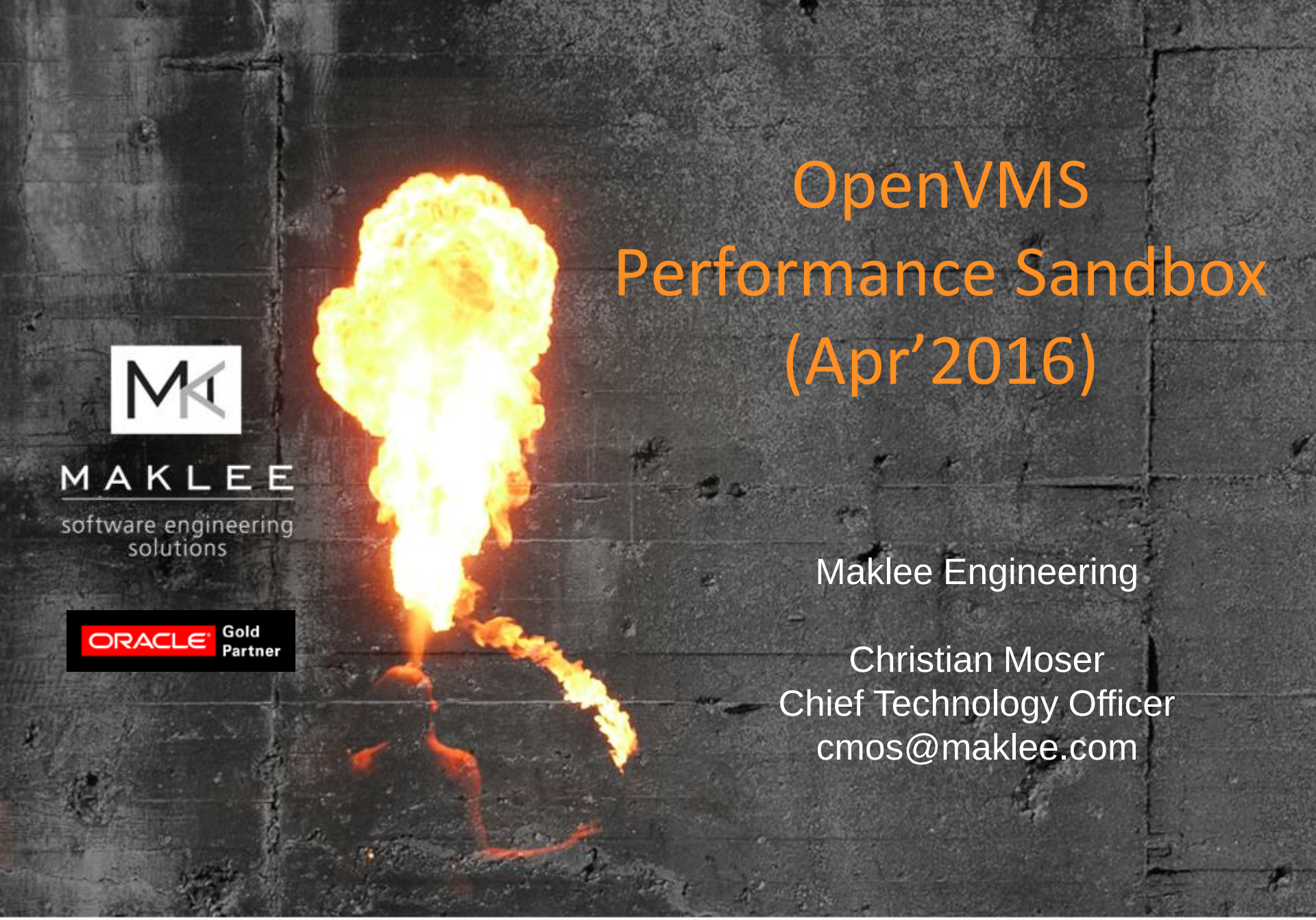


OpenVMS Performance Sandbox (Apr'2016)

MAKLEE
software engineering solutions

ORACLE Gold Partner

Maklee Engineering

Christian Moser
Chief Technology Officer
cmos@maklee.com

Quote

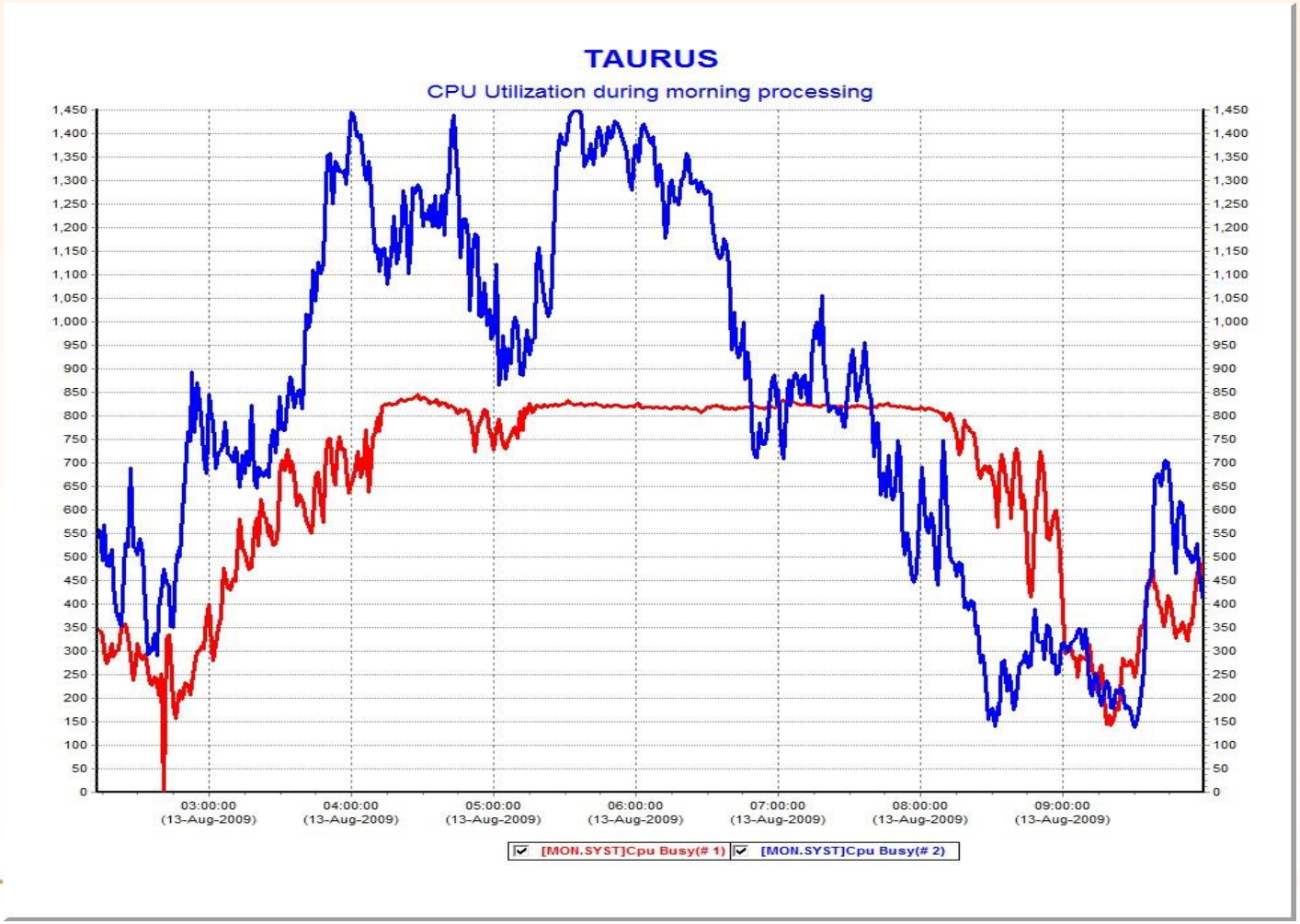
"Keep looking below surface appearances. Don't shrink from doing so just because you might not like what you find."

Colin Powell

MAKLEE

The Easy Way…

- As mentioned MANY times in the past, the easiest way to improve performance is to throw more hardware on the problem

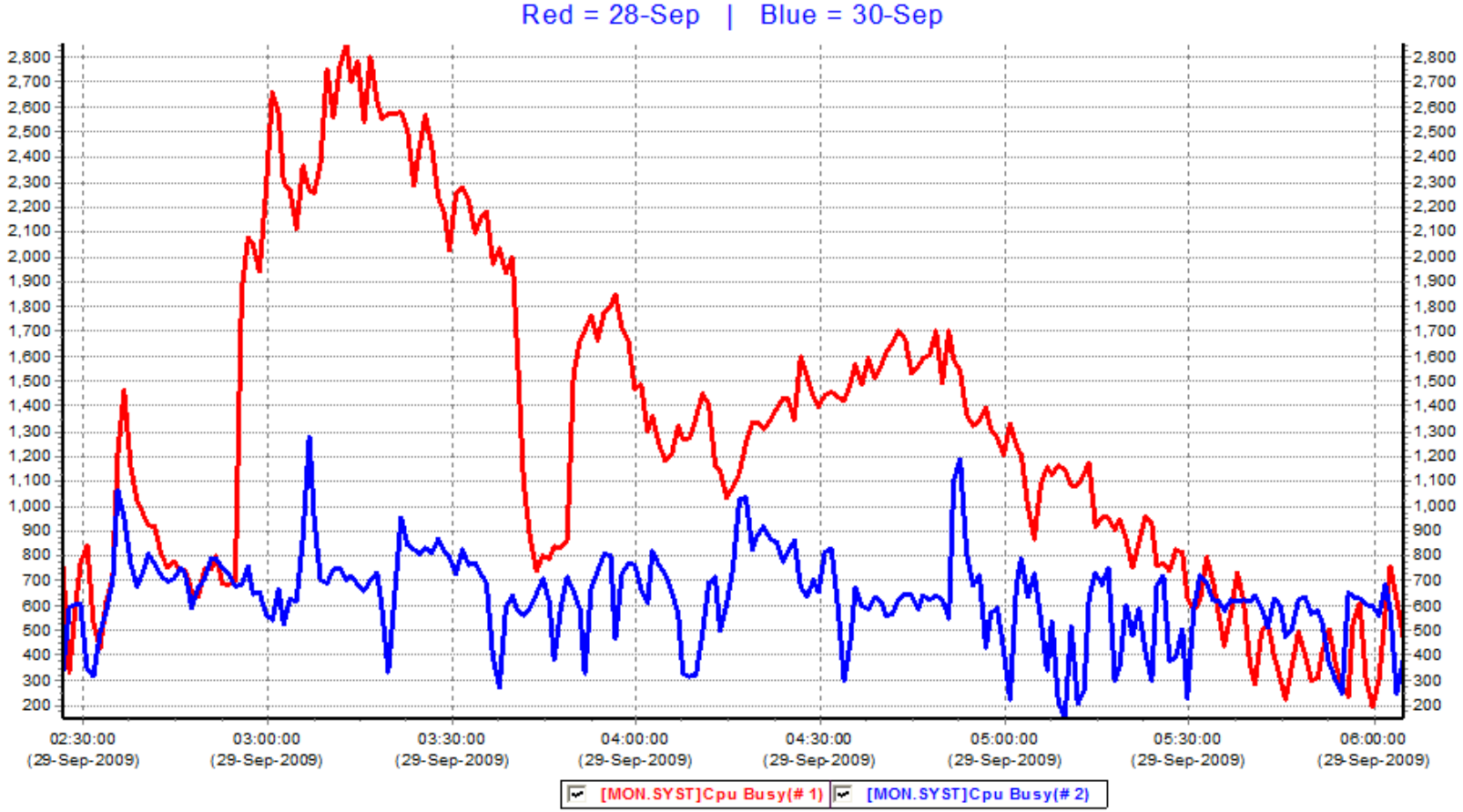


MAKLEE

The Hard Way…

- Problems start when
  - The latest and greatest hardware is already being used
  - Hardware solution is too expensive
  - Extra hardware degrades performance even more
  - Extra hardware improved the situation but is still not enough …

The Power of Tuning

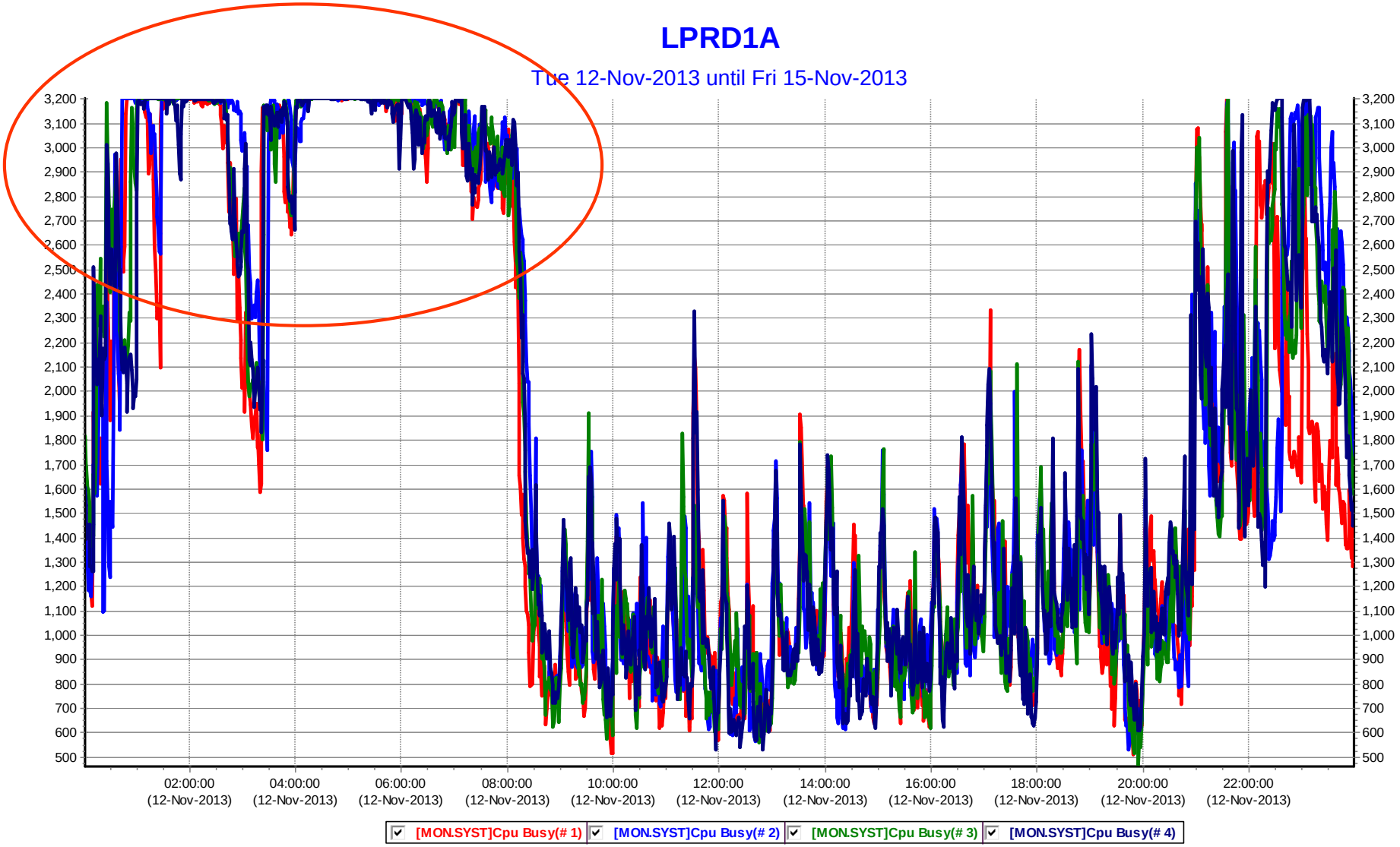


MAKLEE

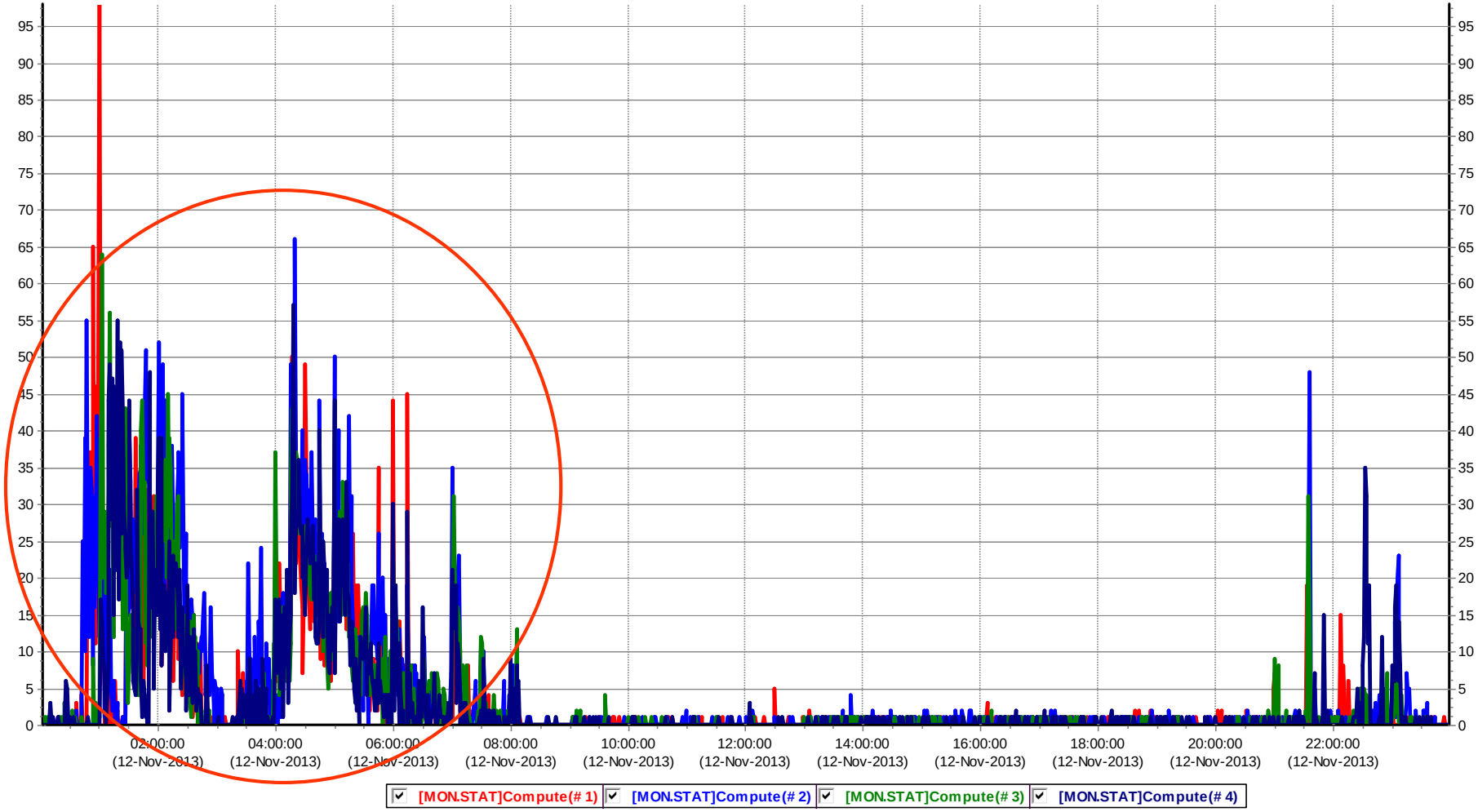
OpenVMS Tuning

- Customer calls and says “Help!”
  - My system is running slow
- How do we determine what are the areas that need improvement?
- Need to independently navigate through the rough waters of corporate politics and different agendas
- T4 is an excellent starting point!
- Builtin and shipping tools
  - many of them not really documented (my bad…)
- The following slides demonstrates some examples
  - What we are looking at
  - Possible tools in the arsenal

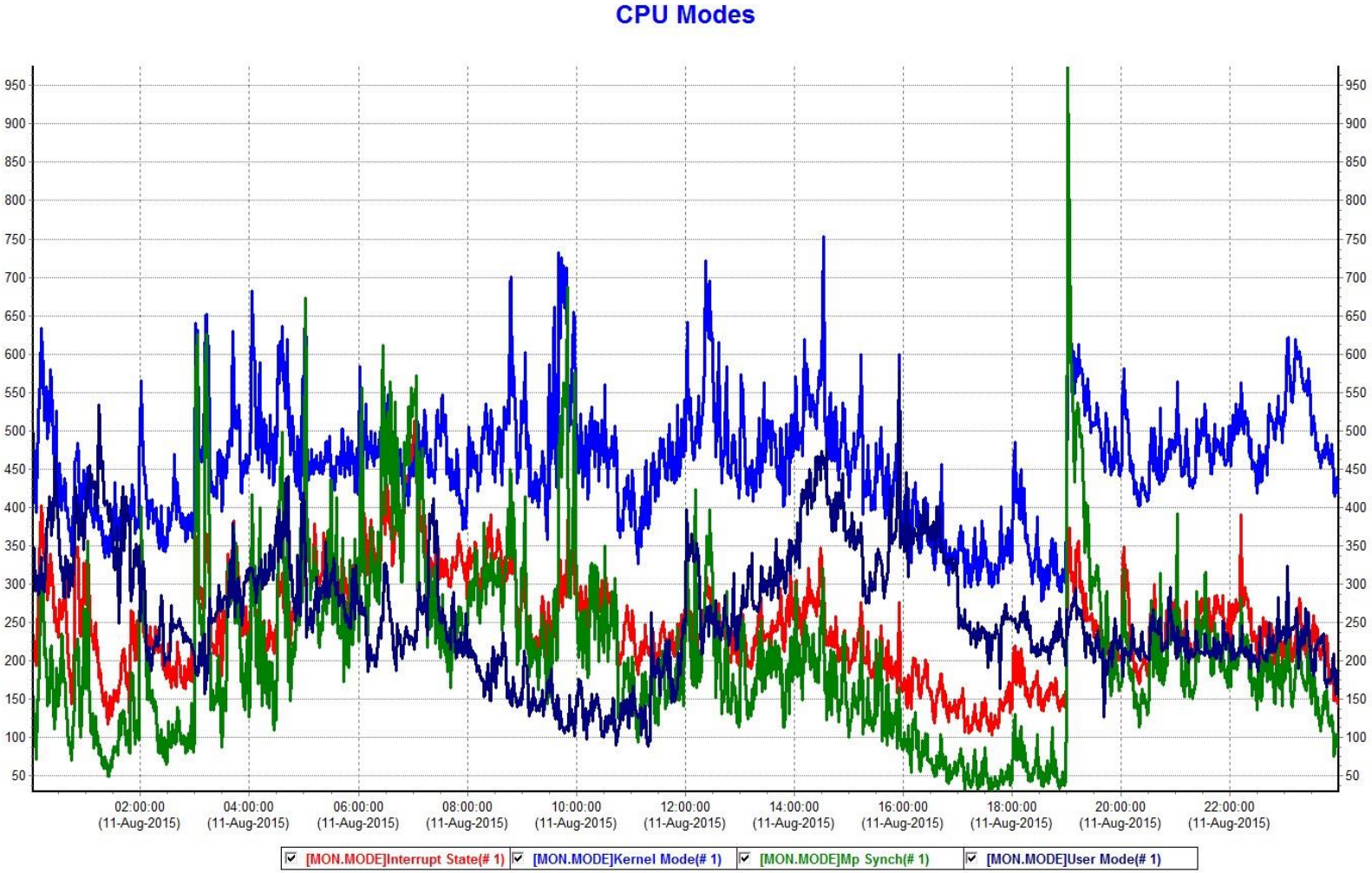
CPU Utilization



COM Queue


LPRD1A
Tue 12-Nov-2013 until Fri 15-Nov-2013
95
90
85
80
75
70
65
60
55
50
45
40
35
30
25
20
15
10
5
0
02:00:00 (12-Nov-2013)
04:00:00 (12-Nov-2013)
06:00:00 (12-Nov-2013)
08:00:00 (12-Nov-2013)
10:00:00 (12-Nov-2013)
12:00:00 (12-Nov-2013)
14:00:00 (12-Nov-2013)
16:00:00 (12-Nov-2013)
18:00:00 (12-Nov-2013)
20:00:00 (12-Nov-2013)
22:00:00 (12-Nov-2013)
[MON.STAT]Compute(# 1)
[MON.STAT]Compute(# 2)
[MON.STAT]Compute(# 3)
[MON.STAT]Compute(# 4)

CPU Modes

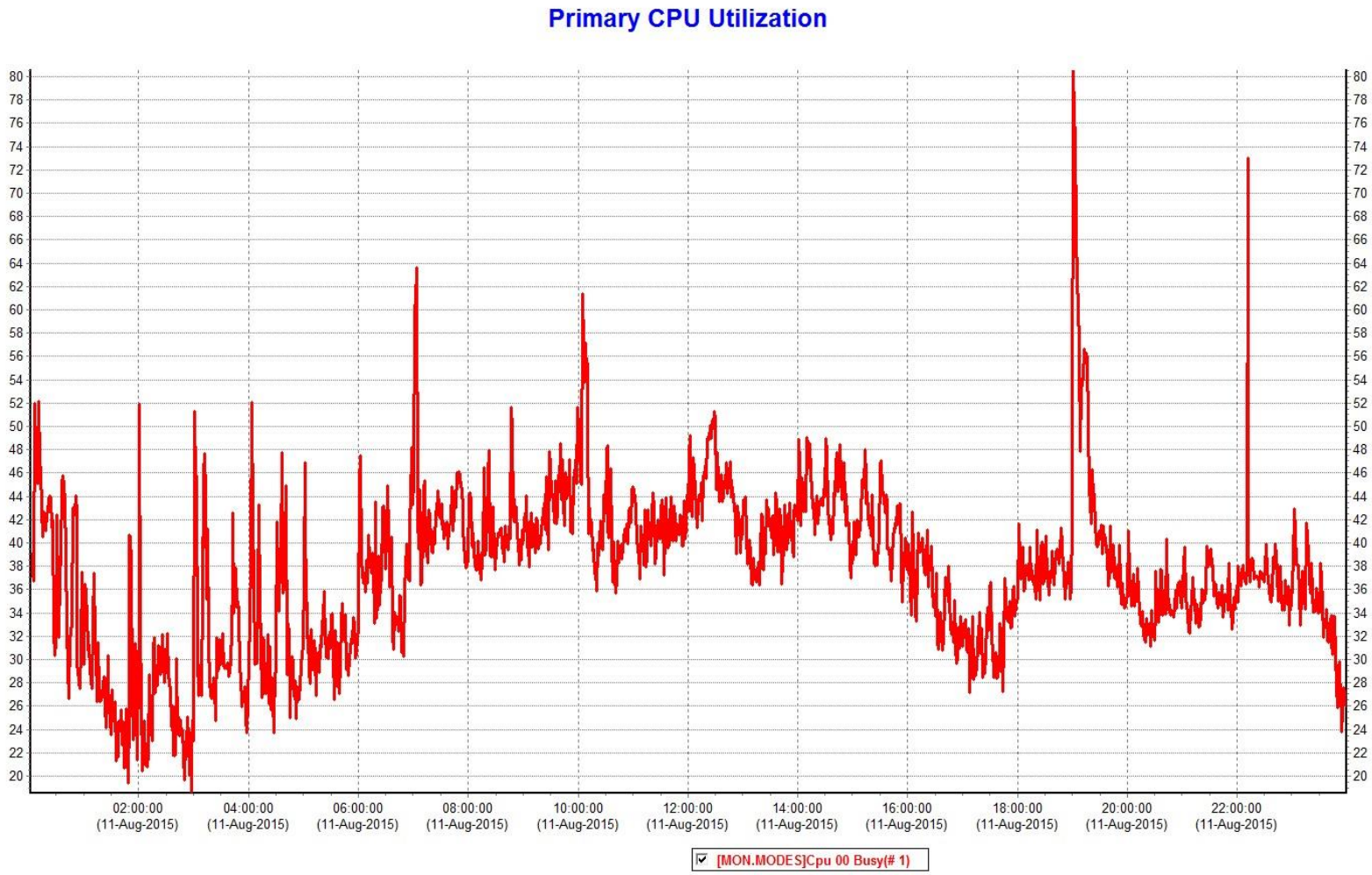


MAKLEE

Primary CPU

- There are certain activities on OpenVMS that can only be performed by the primary CPU
- Avoid running out of ”Primary CPU”

Primary CPU Utilization

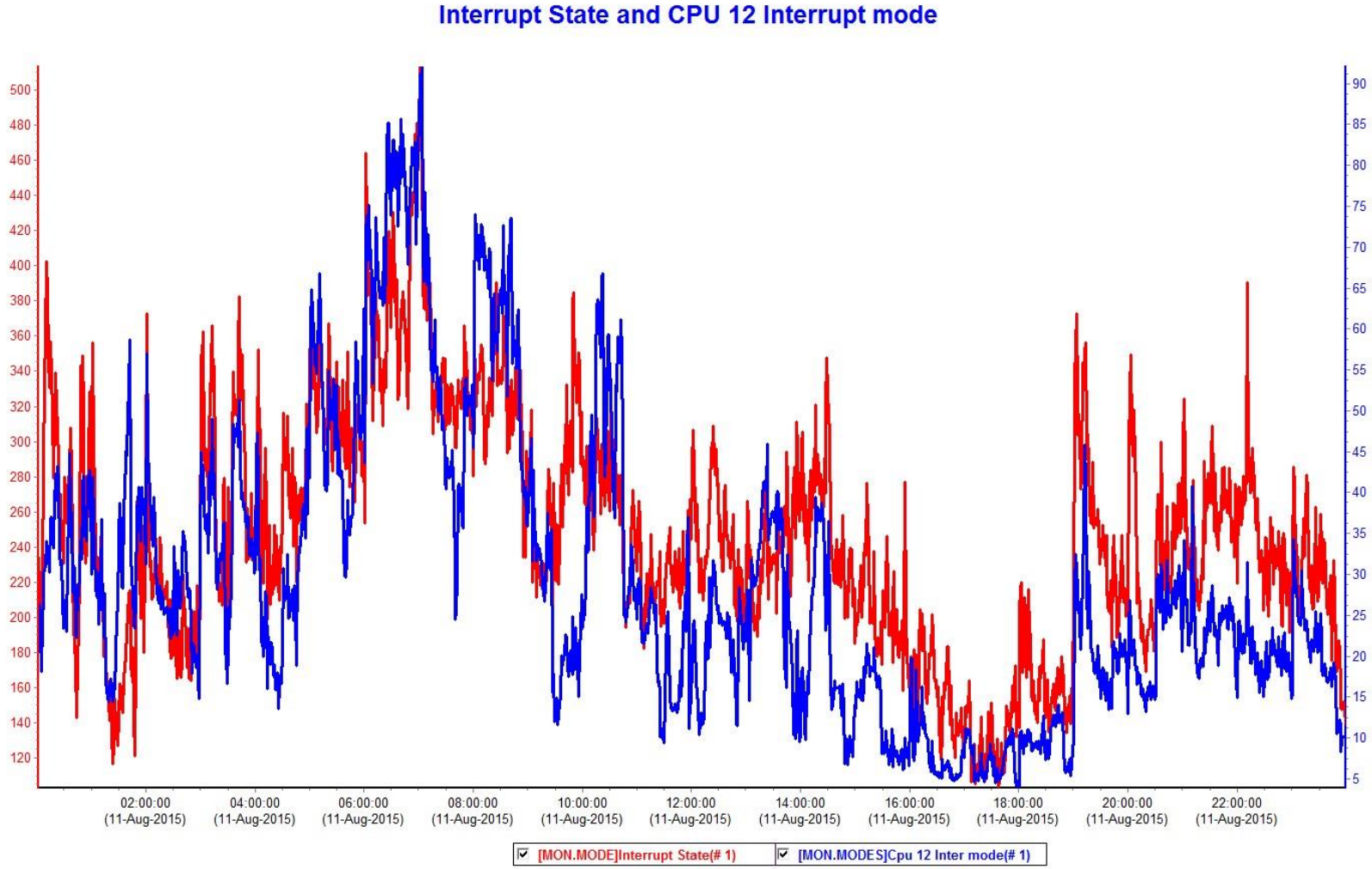


MAKLEE

FastPath Assignment

- Avoid having busy devices served by the same CPU
- Make sure CPU is ”close” to the device

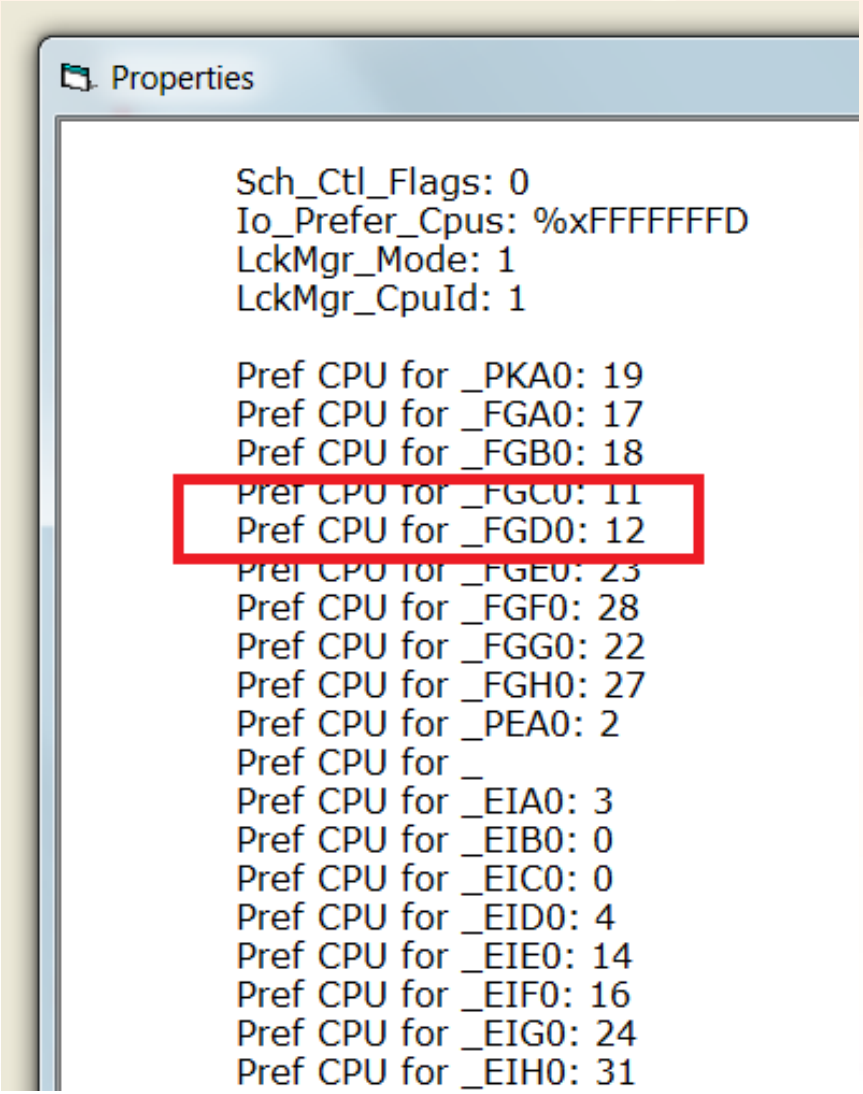
CPU 12 Interrupts



MAKLEE

Device Interrupts

- T4 can be used to determine which device is being served by CPU 12
- In our case it is FGD0

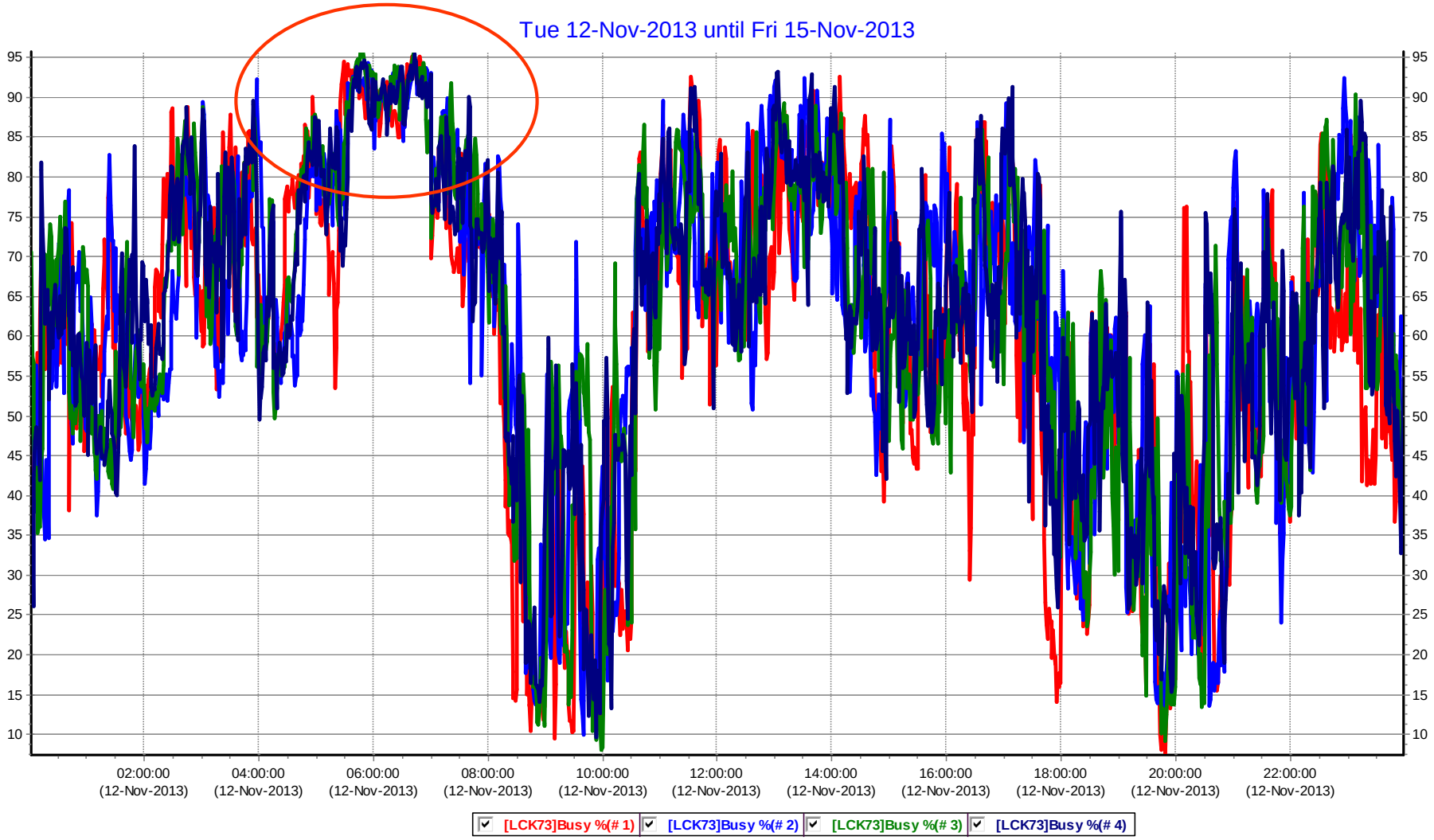


MAKLEE

Dedicated Lock Manager

- The Dedicated Lock Manager always consumes 100% of a single CPU
- T4 allows monitoring the utilization of the lock manager. Once utilization hits 100% the system will not be able to handle additional locking requests
- Determine the location of the memory allocated for the Lock Manager Spinlock and choose a CPU close to this location
  - Especially important for a system with high memory latency like Superdome

Dedicated Lock Manager



TCPIP Packet Processing Engine (PPE)

- The recommendation is to enable PPE, otherwise the penalty for IPINT processing is way too high

- How to enable or disable PPE on the fly

```
$ sysconfig -r inet ppe_enable=1
$ sysconfig -r inet ppe_enable=0
```

- Profiling is needed in order to monitor the performance of PPE since the dedicated CPU is now 100% busy

```
$ sysconfig -r inet profiling=1
```

- Add the following lines to TCPIP$ETC:SYSCONFIGTAB.DAT

```
inet:
    ppe_enable = 1
    profiling = 1
```

MAKLEE

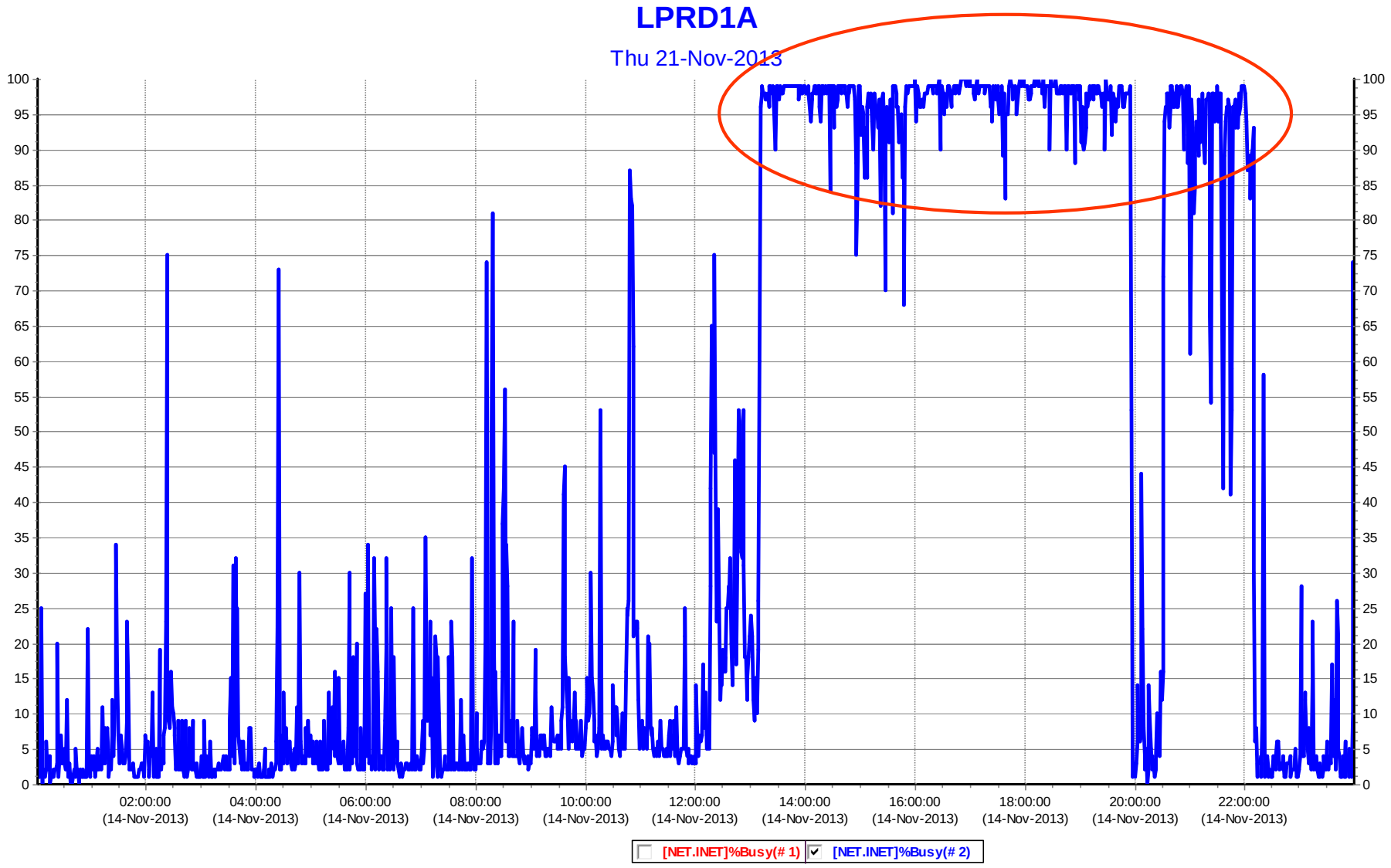
PPE Performance Monitoring

- The following commands can be used to monitor the performance of PPE

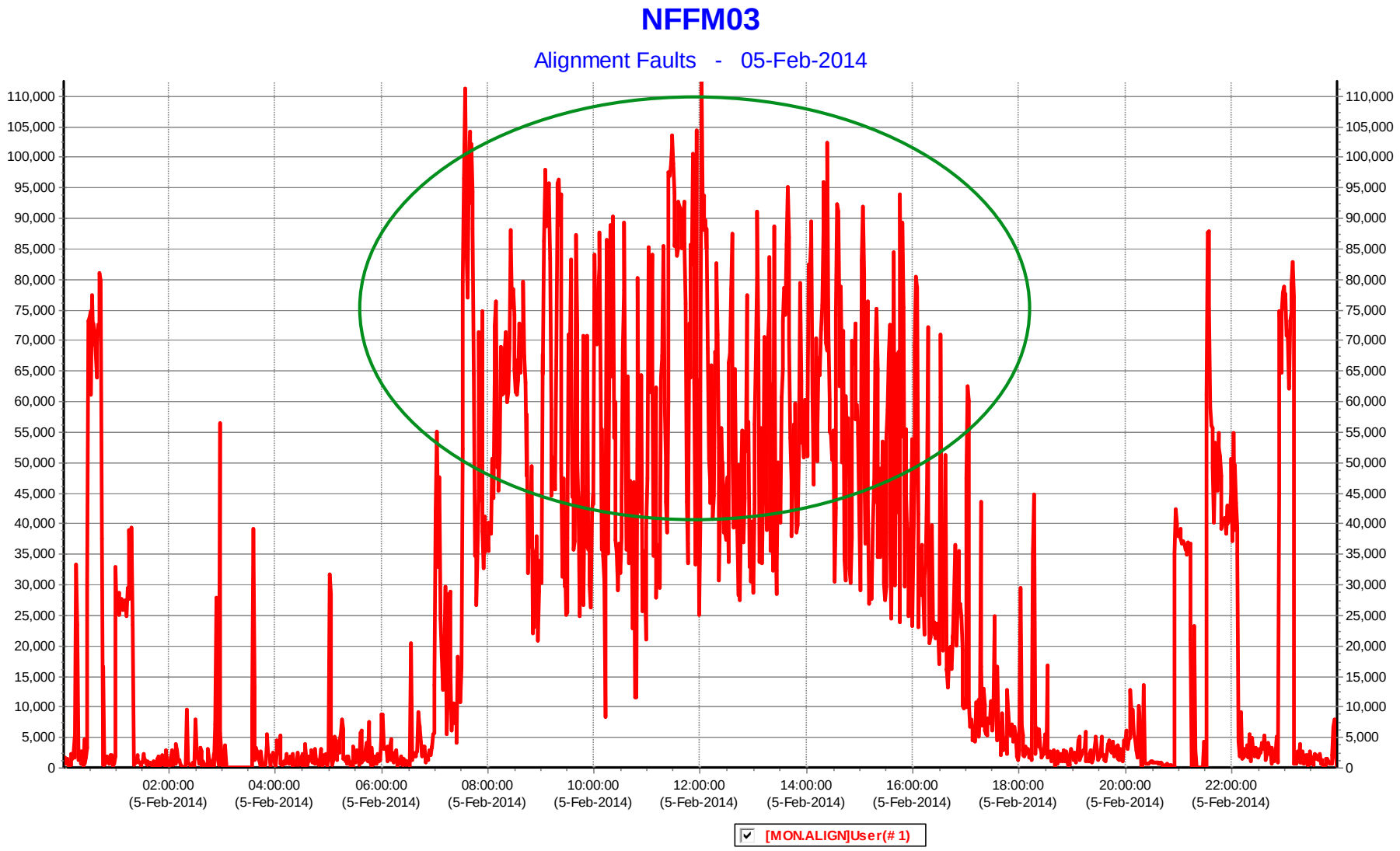
```
$ set command tcpip$examples:tcpip$tcp_mon
$ tcpmon /show=inet /sample=1
$ tcpmon /show=all /sample=5 /csv=ppe.csv
```

- Some key statistics
  - %busy — indicates how busy PPE is (100% means saturation)
  - #Krp — shows how many kernel request packets (KRP) have been processed per second
  - KrpQMax — shows maximum queue depth of the TCPIP kernel work queue
- TlViz can analyze those csv files

TCPIP PPE


LPRD1A
Thu 21-Nov-2013
100
95
90
85
80
75
70
65
60
55
50
45
40
35
30
25
20
15
10
5
0
02:00:00 (14-Nov-2013)
04:00:00 (14-Nov-2013)
06:00:00 (14-Nov-2013)
08:00:00 (14-Nov-2013)
10:00:00 (14-Nov-2013)
12:00:00 (14-Nov-2013)
14:00:00 (14-Nov-2013)
16:00:00 (14-Nov-2013)
18:00:00 (14-Nov-2013)
20:00:00 (14-Nov-2013)
22:00:00 (14-Nov-2013)
[NET.INET]%Busy(# 1)
[NET.INET]%Busy(# 2)

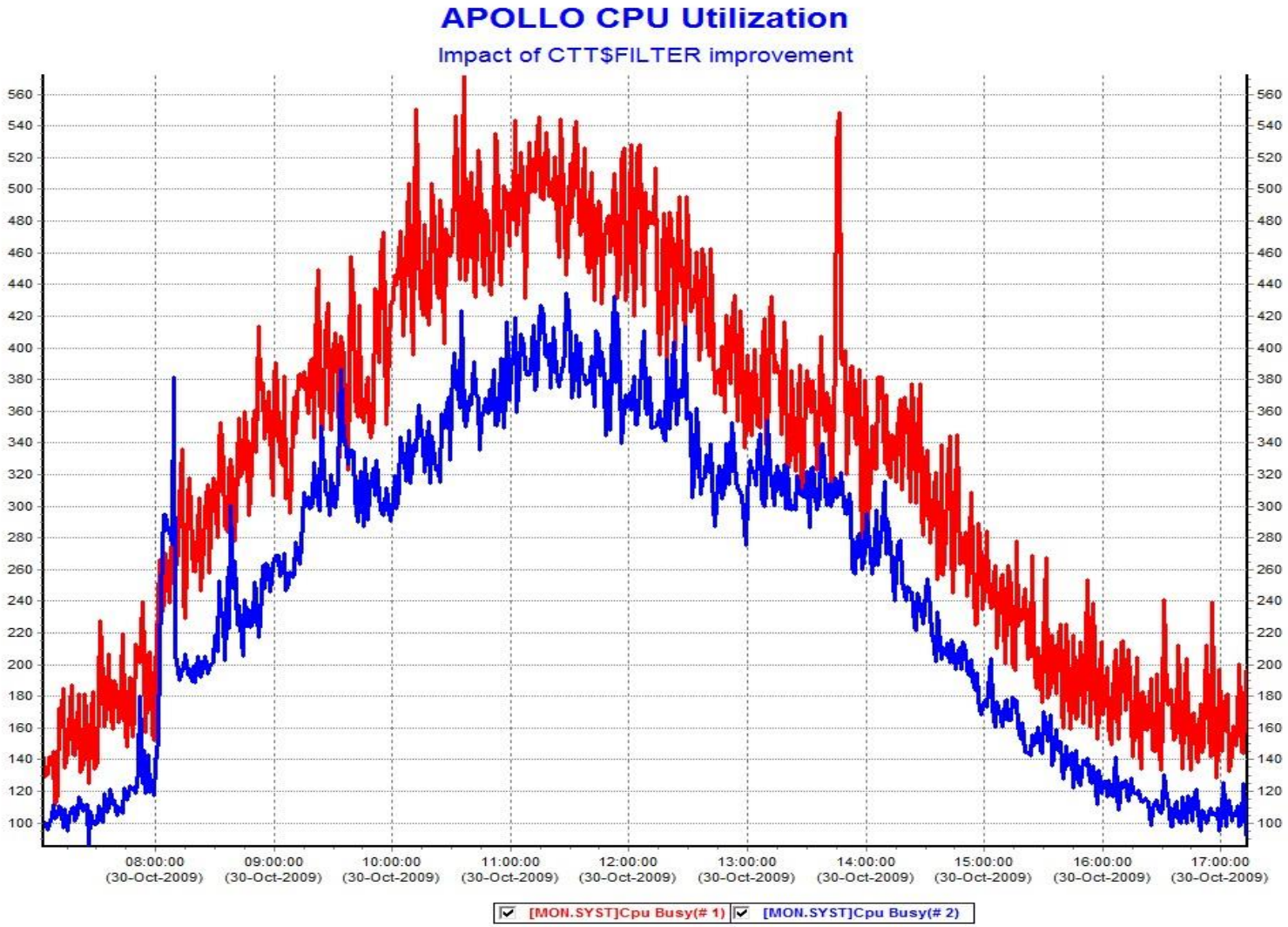
Alignment Faults



Alignment Faults (cont'd)

- Alignment faults are VERY expensive on Itanium
- However… Alignment faults are not unique to Itanium and can be experienced on Alpha as well
- On Alpha alignment faults impact the process generating the faults vs the entire system
- MONITOR ALIGN does not work on Alpha, but SDA PRF does
- The following slide demonstrates the impact of eliminating alignment faults on Alpha
  - 16 cores GS1280 (1.3Ghz)
  - OLTP application, for this particular workload every millisecond counts
  - One line code change to align one data structure
  - 25% performance improvement

Alignment Faults (cont'd)



MAKLEE

Convert

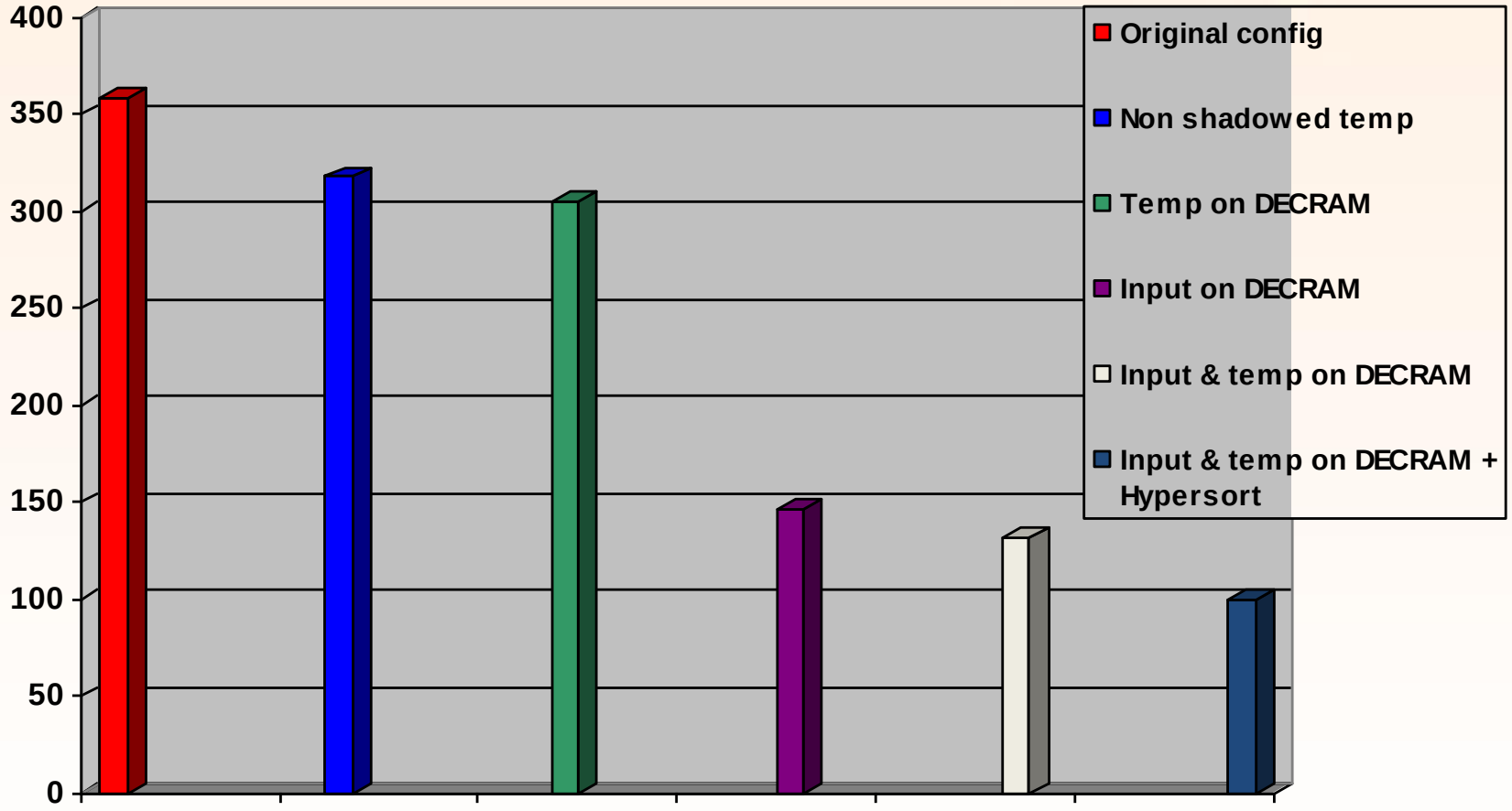
- Improving performance of CONVERT operations is always challenging
- Convert is using two types of temporary work files
  - Sort work files (used when /sort is specified)
  - Convert work files
- To speed up convert operations, we evaluated the impact of moving temporary work files to other disks
  - VMS allows controlling the location of temporary work files using logical names
- On Itanium, the MCOE & EOE packages, include the DECRAM license

Convert (cont'd)

- CONVERT
  - 32 cores RX8640
  - 96GB RAM
  - OpenVMS V8.3-1H1
  - Input file - 1.3GB RMS indexed file
  - Temporary work file peaked at 7GB
  - Convert performed using the /NOSORT qualifier

- The convert operation required 6 minutes to complete. The customer needed to complete the operation as quickly as possible

Convert (cont'd)



Elapsed time (seconds) to sort 1.3GB
Less is better

MAKLEE

Sort Generations on OpenVMS

- Sort of the 80's
  - SORTSHR image
  - SOR$INIT_SORT callable interface routine
- Sort of the 90's
  - SORTSHR image
  - SOR$BEGIN_SORT callable interface routine
  - Provides routine to convert older interface routine parameters and automatically pass them into newer interface routine
- High-performance sort
  - HYPERSORT image
  - Interchangeable with 2nd generation sort
  - Does not provide archaic old callable interface routine

Convert / Sort

- It has been observed that convert on Itanium takes much longer on Itanium than it took on Alpha
- Sometimes it fails on Itanium with ”virtual address space full” error
- Fix to avoid VASFULL and INSVIRMEM errors is to use Hypersort instead of the older traditional sort
  - First of all it works
  - Second it is much faster
- Additional RMS tuning will further speed up the convert operation
- If you insist in using SORTSHR then trim down the process working set extent limit before the sort or convert operation
  - $ set working /extent=n
  - Leave PQL_MWSEXTENT high, otherwise every process gets punished on the system

MAKLEE

CONVSHR vs HYPERSORT

```
$ sort/key=(pos:1,siz:10)/stat $1$dga1250:[dbload]p7911109.cnv junk.dat

                  OpenVMS Sort/Merge Statistics

Records read:    37573770          Input record length:      235
Records sorted:  37573770          Internal length:          237
Records output:  37573770          Output record length:     235
Working set:      2000000          Sort tree size:        930948
Virtual memory:    452528          Number of initial runs:     1
Direct I/O:        375525          Maximum merge order:        1
Buffered I/O:         121          Number of merge passes:     1
Page faults:        29225          Work file alloc:     17586817
Elapsed time: 00:06:50.99          Elapsed CPU:      00:06:46.07

$ def sortshr sys$share:hypersort
$ set rms/buf=8/block=127/ext=65000
$ sort/key=(pos:1,siz:10)/stat $1$dga1250:[dbload]p7911109.cnv junk.dat

                  OpenVMS Sort/Merge Statistics

Records read:    37573770          Input record length:      235
Records sorted:  37573770          Internal length:            0
Records output:  37573770          Output record length:       0
Working set:      2000000          Sort tree size:             0
Virtual memory:    230672          Number of initial runs:     0
Direct I/O:        300247          Maximum merge order:        0
Buffered I/O:         190          Number of merge passes:     0
Page faults:        14324          Work file alloc:            0
Elapsed time: 00:02:36.52          Elapsed CPU:      00:02:58.34
```

more than 2x faster

MAKLEE

Multiple Kernel Threads

- Evaluated the impact of disabling multiple kernel threads on a Java based benchmark
- Single threaded Java program performing CPU intensive operation (encryption)
- SD32B, 32 CPUs, OpenVMS V8.3-1H1, Java 5
- Used SET IMAGE to disable multiple kernel threads

Encryption Test

- Multiple kernel threads (MKT) enabled

```
Accounting information:
Buffered I/O count:              103709      Peak working set size:     891216
Direct I/O count:                  7279      Peak virtual size:        2652928
Page faults:                      55739      Mounted volumes:                0
Charged CPU time:        0 00:02:36.81      Elapsed time:       0 00:15:54.98
```

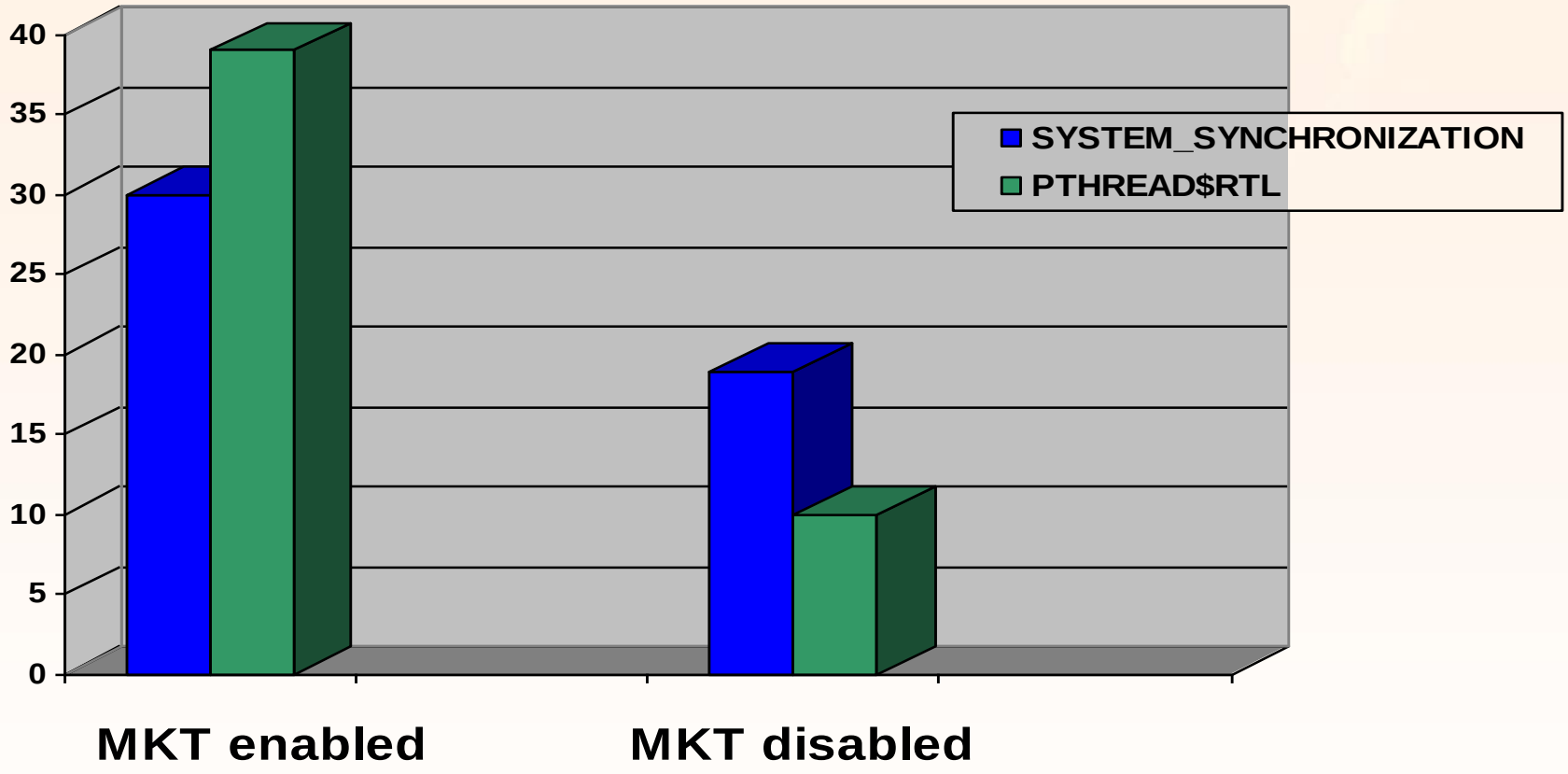
- Multiple kernel threads (MKT) disabled

```
Accounting information:
Buffered I/O count:              102399      Peak working set size:     841424
Direct I/O count:                  7145      Peak virtual size:        2584064
Page faults:                      52623      Mounted volumes:                0
Charged CPU time:        0 00:01:35.80      Elapsed time:       0 00:15:18.83
```

39% less CPU time

MAKLEE

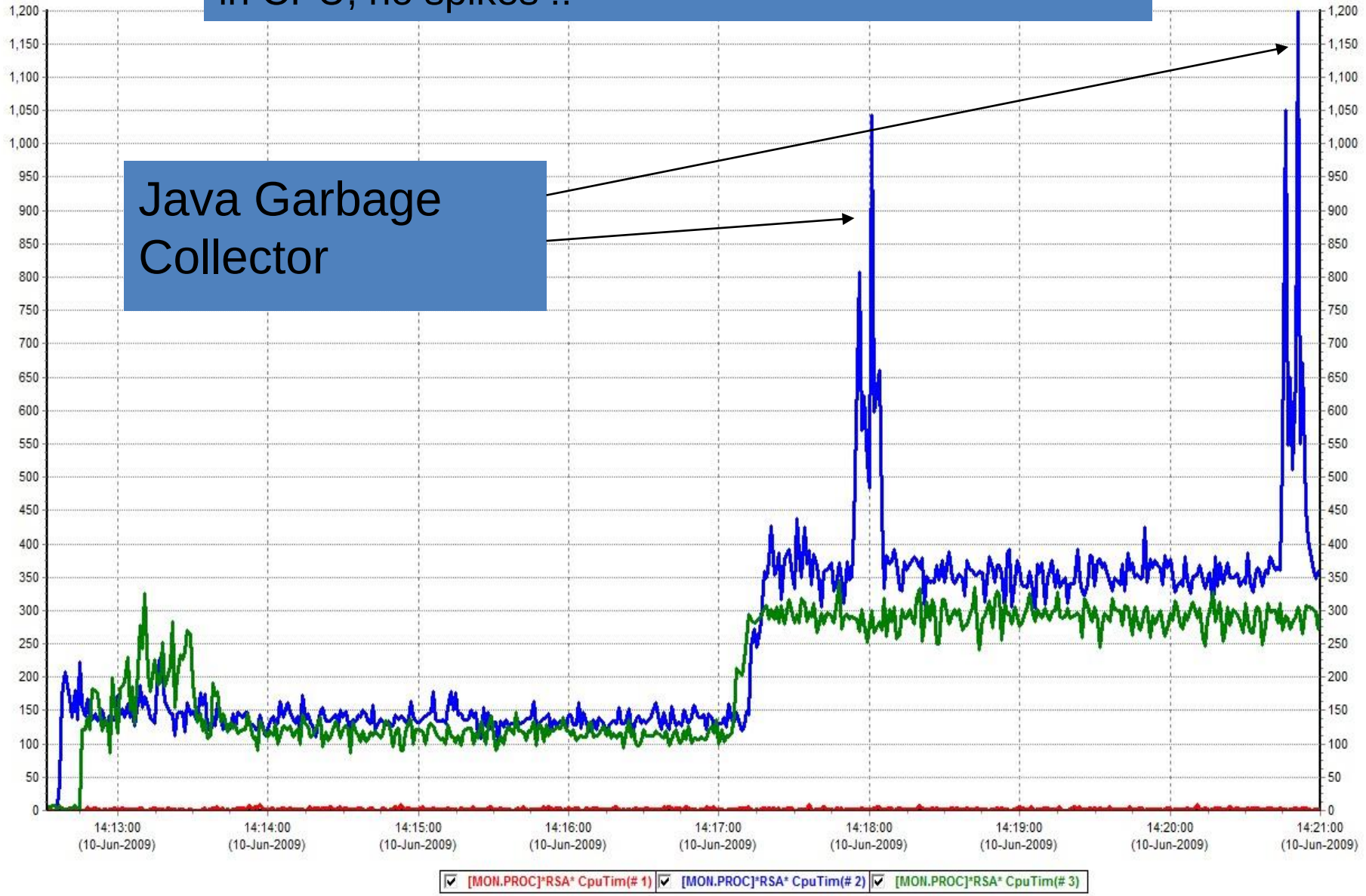
PC Sampling



Number of CPU cycles (in percent of total)
Less is better

MAKLEE

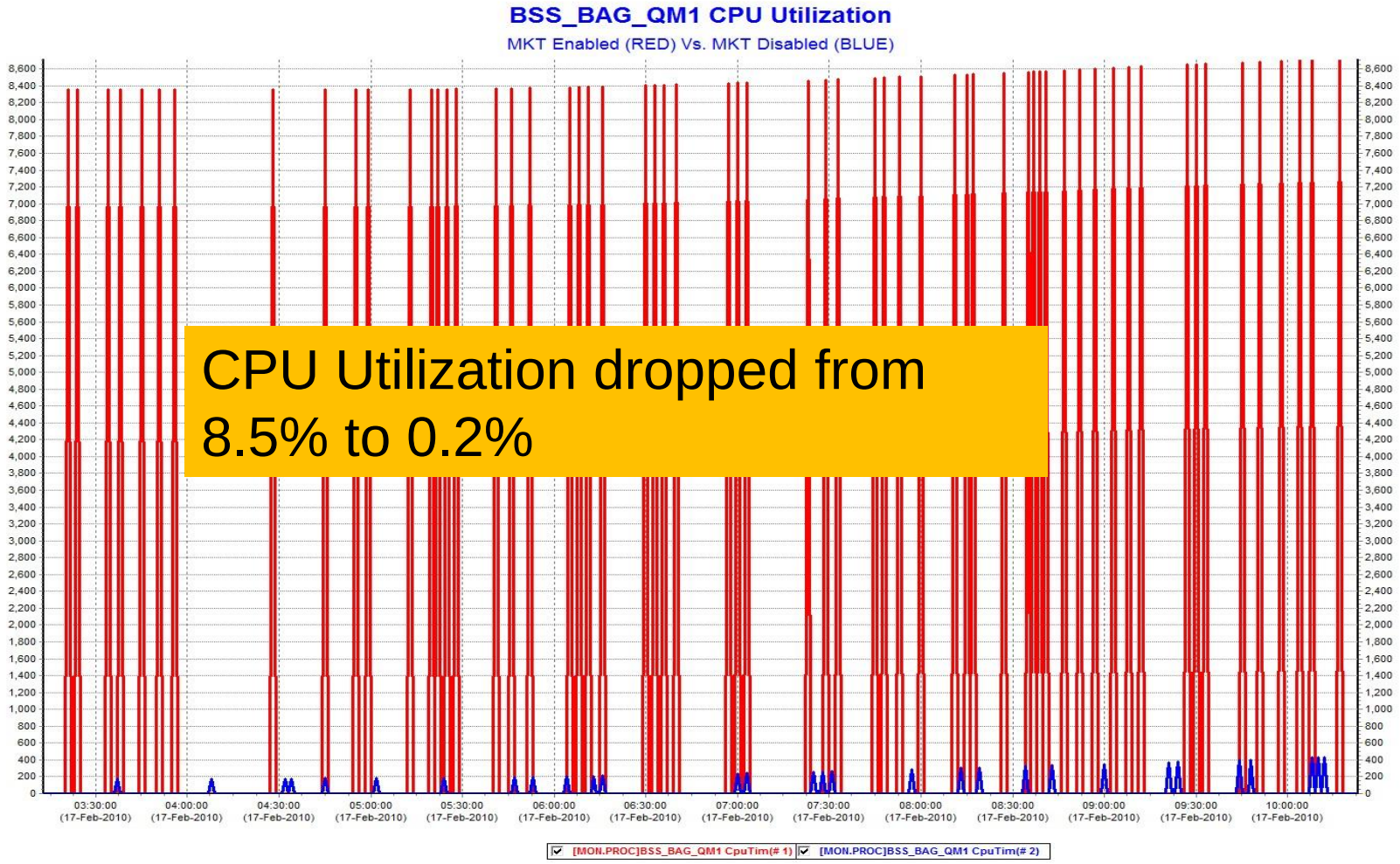
CPU utilization during benchmarks.15% reduction in CPU, no spikes !!



PTHREAD API

- A program was using the PTHREAD run-time library (pthread_X API) but was not a multi-threaded program
- Even though the program was NOT multi-threaded, OpenVMS created 16 kernel threads when activating the image (rx6600, HT enabled)
- The OpenVMS threads manager has to keep all the kernel threads synchronized
- Profiling of the program uncovered that significant CPU time is spent synchronizing the kernel threads
- As the application was not multi-threaded, multiple kernel threads has been disabled for the image, resulting in SIGNIFICANT reduction of CPU time consumed by the process

PTHREAD API (cont'd)



MAKLEE

Reduce Locking

- Reducing locking footprint of single process operating on a file
- Methods to use No Query Locking (NQL)
  - Open files with NQL bit set or perform $GET operations with NQL bit set
  - Open files with NLK+RRL bits set or perform $GET operations with NLK+RRL bits set, plus RMS switch to turn NLK+RRL into NQL
- RMS still locks the record eventhough NLK+RRL bits are set
  - $ SET RMS /QUERY=DISABLE /SYSTEM
  - will not lock data record for any $GET operation with NLK+RRL bits set

Reduce Locking (cont'd)

- Risk
  - Very low risk to disable query locking systemwide via
  - $ SET RMS /QUERY=DISABLE /SYSTEM
  - Only affects $GET operations with NLK+RRL bits set

- Performance benefit
  - Anything between 0% and a positive percent number
  - Depends on how many $GETs have NLK+RRL bits set
  - Depends on how busy the dedicated lock manager is

No Query Locking

- Before

```
Accounting information:
Buffered I/O count:                169      Peak working set size:      19552
Direct I/O count:                241835     Peak virtual size:         186144
Page faults:                       1320     Mounted volumes:                0
Charged CPU time:        0 00:03:06.73      Elapsed time:       0 00:21:55.09
```

- After

```
Accounting information:
Buffered I/O count:                173      Peak working set size:      19536
Direct I/O count:                241358     Peak virtual size:         186144
Page faults:                       1367     Mounted volumes:                0
Charged CPU time:        0 00:01:49.36      Elapsed time:       0 00:14:44.91
```

  - More than 30% reduction in elapsed time
  - More than 40% reduction in CPU time

MAKLEE

RMS Global Buffers

- RMS global buffers on hot files will reduce locking activity on buckets
  - Maklee recommends setting a fixed value for the global buffer count
  - We do not trust in dynamic global buffers
- What is the magic number?
  - Anything smaller than 100 is most likely not helpful
  - Anything larger than like 5,000 is most likely overkill
  - Monitor the global cache hit rate and adjust up or down
- Global buffers are very useful if many processes have the same RMS indexed file open at the same time
  - Sharing the same index buckets, less disk I/Os, less locking
  - The first one to open a file which has a high GBC count will get punished, because all the global buffers and lock resources need to be created

RMS Global Buffers (cont'd)

- Example
  - Opening a file with 50,000 global buffers
  - Need to allocate all those 50,000 buffers at once
  - Allocate 50K lock blocks and resource blocks from the lock manager to synchronize access
  - Any new process opening the same file will simply map those global buffers, but do not need to create all the structures
  - The last one to close the file will tear down the 50K global buffers and lock manager resources and locks
- If you need a high number of global buffers on a file which is not open all the time, then have an idle process which does nothing, except opening the file and then hibernating forever
  - No need to do random I/Os as they are overhead and will trash the cache

Monitoring RMS File Statistics

```
$  mc rms_stats -o=a EXTEL$EXSHARE_DATABASE_DAT

 0.36 IO/Ops.  Ops: 2827615  Cache:    0% Global: 76%  EXSHARE_DATABASE.DAT
Local  Cache trials  4273169  hit     663  read      43   write   298
Global Cache trials  4272626  hit 3264265  read 1008450   write     0

Locks:              Enqueue     Dequeue     Convert         Wait   Block-ast
-----------------+----------+-----------+-----------+-----------+----------
 Shared file:          1293        1294        1294            0           0
 Local buffer:           43        1634        1589            0           0
 Global buffer:     2017529     1008912         608            0           0
 Append locks:            0           0           0            0           0
 Global section:       1293        1294        3880            0          57
 Data record:       2497111     2497126           0            0

Open:         1293  Close:      1294  Con:        1293  Discon:       1294
Rewind:          0  Flush:         0  Extend         0  blocks:          0
Outbufquo:       0  Xqpqio:     1293  Truncate       0  blocks:          0
Reads            0  Bytes:         0  Writes         0  bytes:           0

Find seq:        0  key:           0  rfa:           0
Get  seq: 2643917  key:      183391  rfa:           0  bytes:   143047680
Put  seq:        0  key:           0                    bytes:           0
Deletes:         0                    Updates      307  bytes:     1414656
Splits:          0  multy:         0  OutbufQuo      0
```

- Only updates and deletes should lock records
- Total of 5 Mio unnecessary lock/unlock operations during reads

MAKLEE

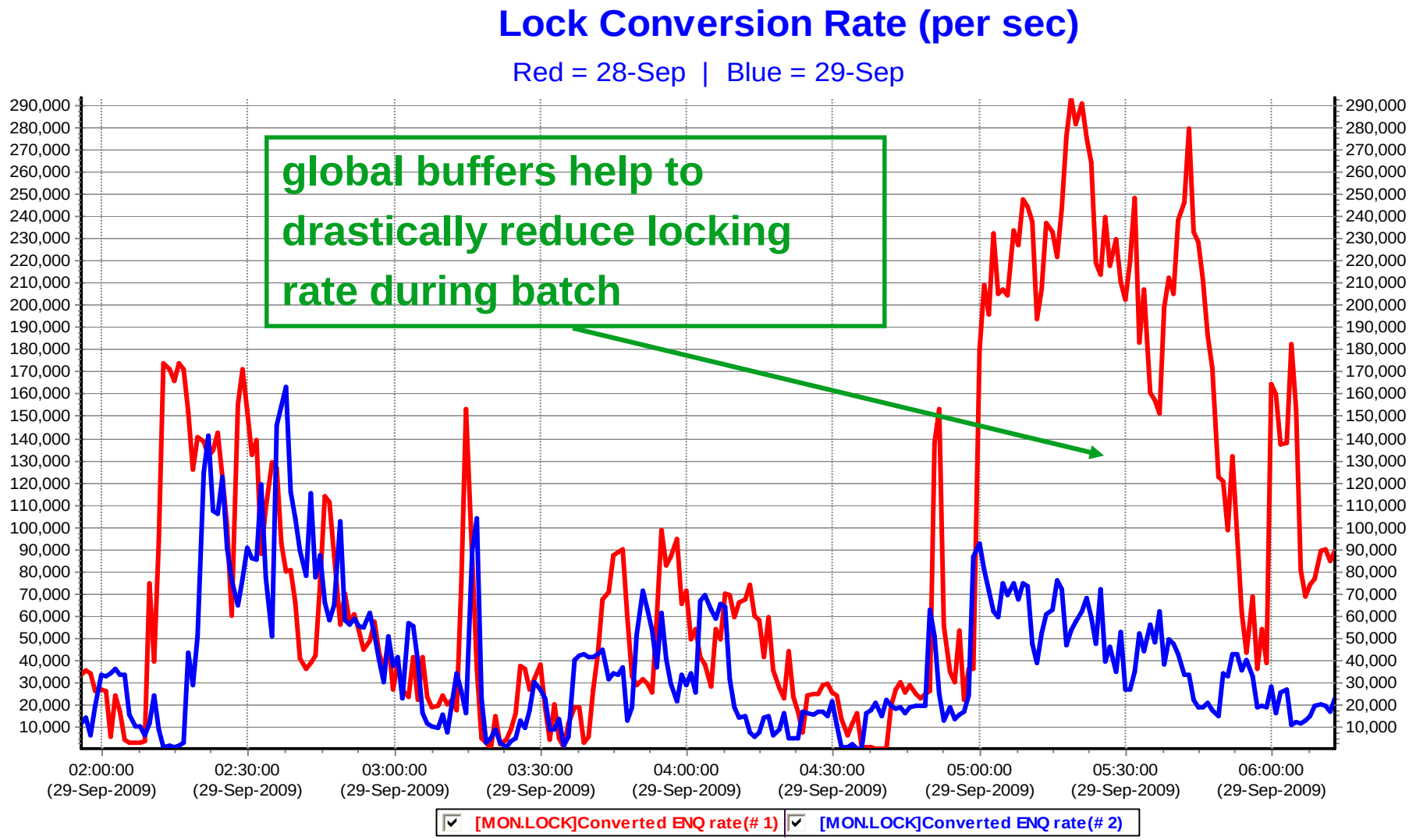
RMS Bucket Size

- Increase bucket size to reduce bucket locking activity
  - Higher lock activity if bucket size is too small
    - Lock bucket, read data, unlock bucket
    - Lock next bucket, read data, unlock next bucket etc
    - For mostly read-only, use the largest bucket size to fit as many records into a single bucket (i.e. bucket_size=63)

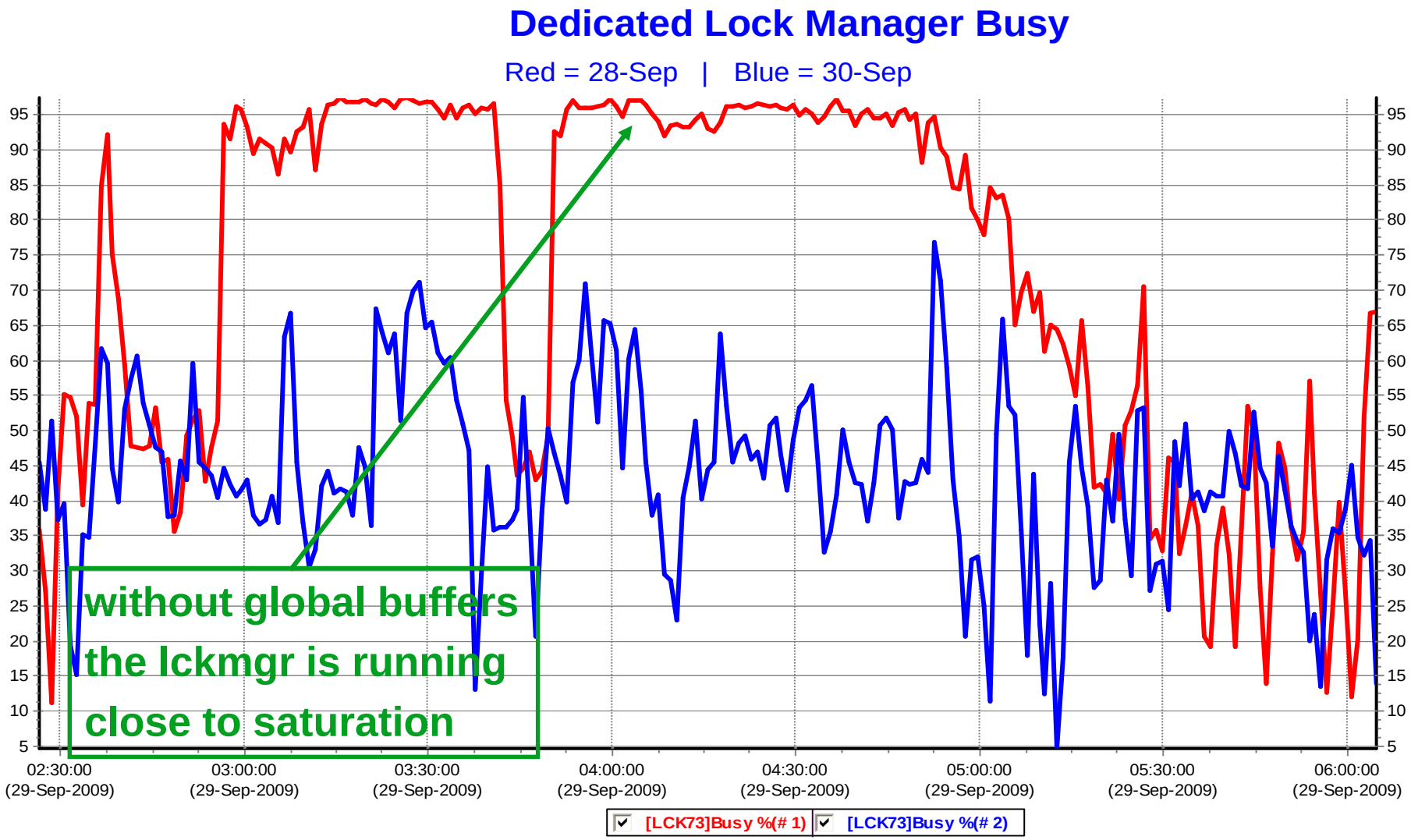
```
$ mc rms_tune_check_v2 -v extel$exshare_database_dat
* 20-NOV-2013 18:46:00.18 ALQ=119203072 BKS=16 GBC=30000 extel$exshare_database_dat
- KEYS:  Are those 13 keys really all needed? ( 0, 123, 132, 148, 305, 4113, 2693, 194, 206,
    175, 2229, 3952, 2508 )
- RECS: Only 2969 records in 1001 bucket sampled. 3.0 rec/bkt. Inefficient?

$ mc rms_tune_check_v2 -v extel$exshare_supp_dat
* 20-NOV-2013 18:49:05.11 ALQ=38139648 BKS=16 GBC=30000 extel$exshare_supp_dat

$ mc rms_tune_check_v2 -v extel$exbond_database_dat
* 20-NOV-2013 18:49:39.00 ALQ=1690624 BKS=12 GBC=10000 extel$exbond_database_dat
- ROOT: Primary  key 0 (IBS=12,DBS=12) index root level is high: 3 (goal=2).
```

MAKLEE

RMS Global Buffers (cont'd)



RMS Global Buffers (cont'd)



Special Sauce RMS Tracing

- I modified and enhanced RMS$SDA to display the image and datafile name for reads which do incur record locking
  - The following procedure will trace for 30 seconds, then grab the results and sort the output file ignoring duplicates

```
$ set noon
$ ana/sys
rms start trace/buffer=1000/special
wait 00:00:30
rms stop trace
set out/nohead/noindex rms.log
rms show trace/special
set out tt
exit
$ def/nolog sortshr sys$share:hypersort
$ sort/nodup rms.log rms.tmp
$ dele/nolog rms.log.*
$ type rms.tmp
$ exit
```

Special Sauce RMS Tracing (cont'd)

```
$ @rms

OpenVMS system analyzer

RMS$DEBUG already loaded...
Tracing started...
Tracing stopped...
------------------------------  ------------------------------------------------------------
AVOX_GENERATE_DAILY_REQUESTS    DISK$HOT_FILES:[VMS$COMMON.SYSEXE]rightslist.dat;2
AVOX_GENERATE_DAILY_REQUESTS    DISK$HOT_FILES:[VMS$COMMON.SYSEXE]sysuaf.dat;2
AVOX_GENERATE_DAILY_REQUESTS    DISK$IA64_VMS:[VMS$COMMON.SYSEXE]TCPIP$HOST.DAT;1
AVOX_GENERATE_DAILY_REQUESTS    DSKDB07:[EXSHARE_UNIVERSES]FULLDB.DAT;68
BONDEYE_CREATE_XREF             DSKDB03:[FTS_EVALS]EVALS.DAT;14
CAPOLY$TRAP                     DISK$HOT_FILES:[VMS$COMMON.SYSEXE]rightslist.dat;2
DB_CREATE_WORD_UNIVERSES        DSKDB07:[EXSHARE_UNIVERSES]BEYESF.DAT;1
FTI_FULL_AUTOGEN_BARCHS         DISK$HOT_FILES:[VMS$COMMON.SYSEXE]rightslist.dat;2
FTI_FULL_AUTOGEN_BARCHS         DISK$HOT_FILES:[VMS$COMMON.SYSEXE]sysuaf.dat;2
FTI_FULL_AUTOGEN_BARCHS         DISK$IA64_VMS:[VMS$COMMON.SYSEXE]TCPIP$HOST.DAT;1
FTI_FULL_AUTOGEN_BARCHS         DSKDB01:[EXSHARE_DATABASE]EXSHARE_DATABASE.DAT;1
FTI_FULL_AUTOGEN_BARCHS         DSKDB02:[EXSHARE_SUPP]EXSHARE_SUPP.DAT;1
FTI_FULL_AUTOGEN_BARCHS         DSKDB02:[EXSHARE_SUPP]MIFID_REF_DATABASE.DAT;1
MIS_UPDATE_UNIVERSE_TOTALS      DISK$HOT_FILES:[VMS$COMMON.SYSEXE]rightslist.dat;2
MIS_UPDATE_UNIVERSE_TOTALS      DISK$HOT_FILES:[VMS$COMMON.SYSEXE]sysuaf.dat;2
MIS_UPDATE_UNIVERSE_TOTALS      DISK$IA64_VMS:[VMS$COMMON.SYSEXE]TCPIP$HOST.DAT;1
MIS_UPDATE_UNIVERSE_TOTALS      DSKDB07:[EXSHARE_UNIVERSES]EQTIES.DAT;68
QUEUEINQUIRY                    DISK$HOT_FILES:[VMS$COMMON.SYSEXE]sysuaf.dat;2
SSG_PROCESSOR                   DISK$HOT_FILES:[VMS$COMMON.SYSEXE]rightslist.dat;2
SSG_PROCESSOR                   DISK$IA64_VMS:[VMS$COMMON.SYSEXE]TCPIP$HOST.DAT;1
SSG_PROCESSOR                   DSKDB04:[EXSHARE_EXSHARE]EXCHG_LC.DAT;2
SSG_PROCESSOR                   DSKDB04:[EXSHARE_EXSHARE]EXTCUR.DAT;16
SSG_PROCESSOR                   DSKDB04_S03:[EXSHARE_EXSHARE]C1.DAT;1
SSG_PROCESSOR                   DSKDB04_S03:[EXSHARE_EXSHARE]EXTCTY.DAT;14
SSG_PROCESSOR                   DSKDB04_S03:[EXSHARE_EXSHARE]FTICE2.DAT;2
SSG_PROCESSOR                   DSKDB04_S03:[EXSHARE_EXSHARE]ISOCTY.DAT;6
SSG_PROCESSOR                   DSKDB10:[COMSTOCK]FTS_CMSTK_EXSHARE.DAT;2
SSG_PROCESSOR                   DSKDB10:[COMSTOCK]FTS_CMSTK_MASTER_XREF.DAT;845
SSG_PROCESSOR                   DSKDB10:[RTS_SNAPS]RTS_1630.DAT;402
SUBMIT                          DISK$HOT_FILES:[VMS$COMMON.SYSEXE]sysuaf.dat;2
UM_V4                           DSKDB04:[EXBOND_DATABASE]EXBOND_DATABASE.DAT;1
UM_V4                           DSKDB09:[BONDEYE_DATABASE]BONDEYE.DAT;180
```

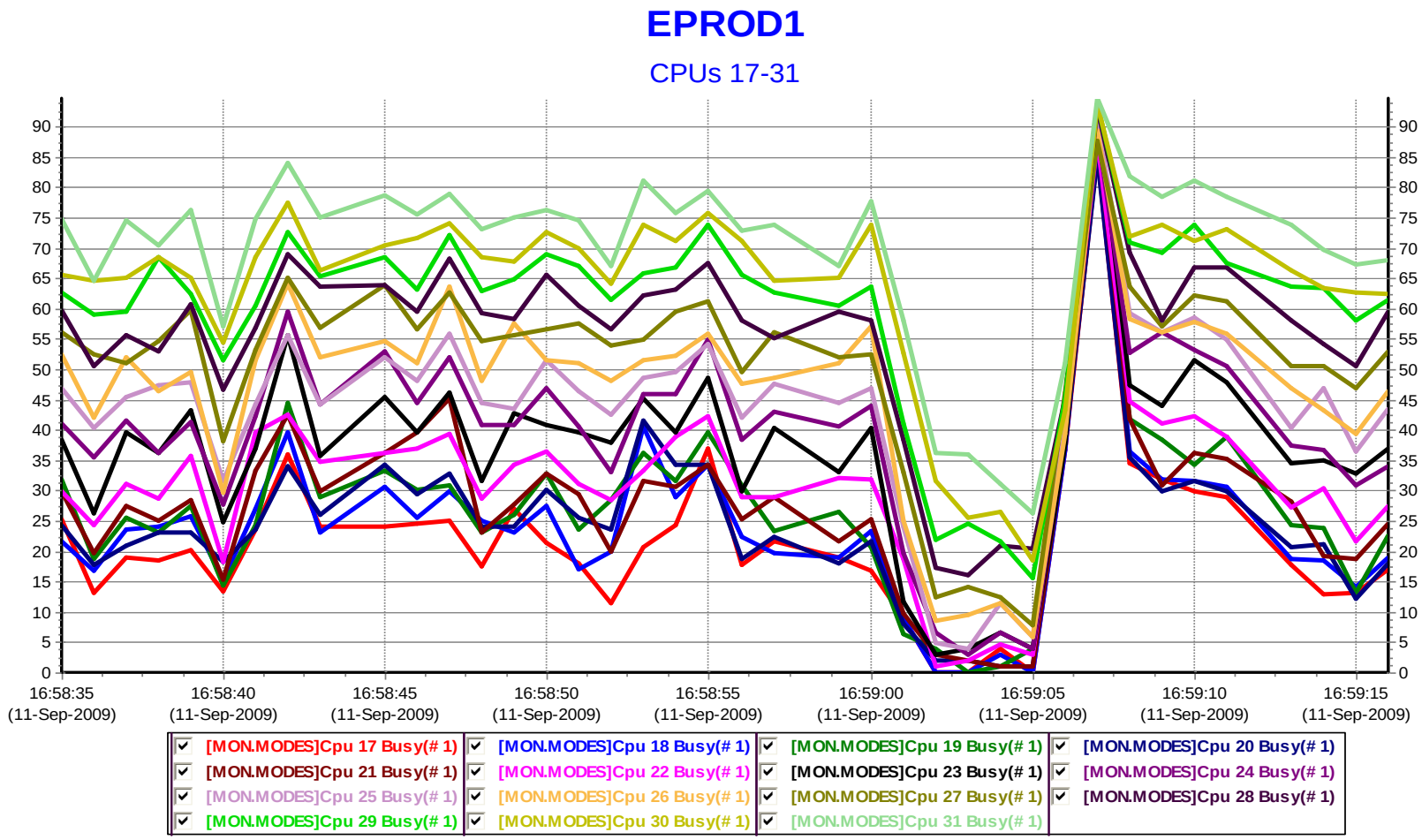
MAKLEE

RMS Default Values

```
$ show rms
          MULTI-   |                 MULTIBUFFER COUNTS                 | NETWORK
          BLOCK    | Indexed  Relative            Sequential            |  BLOCK
          COUNT    |                        Disk    Magtape  Unit Record |  COUNT
Process     0      |    0         0          0        0          0       |    0
System     32      |    0         0          0        0          0       |  127

          Prolog    Extend Quantity     PATH_TIMEOUT
Process     0              0                  0
System      0              0                 10

          QUERY_LOCK      CONTENTION_POLICY
Process    System             System
System     Enabled            Never

$ set rms /query=disable /buffer=8 /block=127 /extent=65000 /system

$ show rms
          MULTI-   |                 MULTIBUFFER COUNTS                 | NETWORK
          BLOCK    | Indexed  Relative            Sequential            |  BLOCK
          COUNT    |                        Disk    Magtape  Unit Record |  COUNT
Process     0      |    0         0          0        0          0       |    0
System    127      |    0         0          8        8          8       |  127

          Prolog    Extend Quantity     PATH_TIMEOUT
Process     0              0                  0
System      0          65000                 10

          QUERY_LOCK      CONTENTION_POLICY
Process    System             System
System     Disabled           Never
```

MAKLEE

”Black Hole”



MAKLEE

TCPIP Hostname Translation

- When TCPIP needs to translate a hostname to an IP address the order is
  - SYS$SYSTEM:TCPIP$HOST.DAT
  - TCPIP$ETC:IPNODES.DAT
  - Bind Resolver

- For real time processes, verify all addresses are defined in the local host database

Host-Based Shadowing

- Host-based volume shadowing between local and remote data centers

- Ensure that system reads data from local shadowset member
  - Set local systems SHADOW_SITE_ID to 1
  - Set local disks site id to 1
    - $ set device /site=1 1dga30xxx
  - Set remote disks site id to 2
    - $ set device /site=2 1dga28xxx

JAVA$60_SETUP Procedure

- JAVA$60_SETUP procedure will define incorrect logicals and not use installed images, unless the semi-colon for the file version is removed

- If a logical pointing to a shareable image contains a semi-colon at the end, then the image activator will find the file and use it, but we want to use the memory resident version instead

- Modify JAVA$60_SETUP procedure to remove trailing semi-colons
  - DISK$JAVA111:[VMS$COMMON.JAVA$60.COM]JAVA$60_SETUP.COM
  - Example:
    - […]JAVA$JAVA_SHR.EXE; -> […]JAVA$JAVA_SHR.EXE

MAKLEE

Installing Java Images Resident

- Following procedure will install Java6 images memory resident and with shared address space
  - Order of install commands is important due to shared address space
  - Logical names are required to resolve dependancies

```
$ define java$java_vms_shr    disk$java111:[vms$common.java$60.jre.lib.ia64]java$java_vms_shr
$ define java$jvm_shr         disk$java111:[vms$common.java$60.jre.lib.ia64.hotspot]java$hotspot_shr
$ define java$hpi_shr         disk$java111:[vms$common.java$60.jre.lib.ia64.native_threads]java$hpi_shr
$ define java$verify_shr      disk$java111:[vms$common.java$60.jre.lib.ia64]java$verify_shr
$ define java$java_shr        disk$java111:[vms$common.java$60.jre.lib.ia64]java$java_shr
$ define java$jli_shr         disk$java111:[vms$common.java$60.jre.lib.ia64]java$jli_shr
$ define java$zip_shr         disk$java111:[vms$common.java$60.jre.lib.ia64]java$zip_shr
$ define java$net_shr         disk$java111:[vms$common.java$60.jre.lib.ia64]java$net_shr
$ define java$nio_shr         disk$java111:[vms$common.java$60.jre.lib.ia64]java$nio_shr
$ define java$java            disk$java111:[vms$common.java$60.bin]java$java
$ define java$mlib_image_shr  disk$java111:[vms$common.java$60.jre.lib.ia64]java$mlib_image_shr
$ define java$awt_shr         disk$java111:[vms$common.java$60.jre.lib.ia64]java$awt_shr
$ define java$mawt_shr        disk$java111:[vms$common.java$60.jre.lib.ia64.xawt]java$mawt_shr
$ define java$fontmanager_shr disk$java111:[vms$common.java$60.jre.lib.ia64]java$fontmanager_shr
```

MAKLEE

Installing Java Images Resident (cont'd)

```
$ install repl/open/head/shar=addr/resi java$java_vms_shr
$ install repl/open/head/shar=addr/resi java$jvm_shr
$ install repl/open/head/shar=addr/resi java$hpi_shr
$ install repl/open/head/shar=addr/resi java$verify_shr
$ install repl/open/head/shar=addr/resi java$java_shr
$ install repl/open/head/shar=addr/resi java$jli_shr
$ install repl/open/head/shar=addr/resi java$zip_shr
$ install repl/open/head/shar=addr/resi java$net_shr
$ install repl/open/head/shar=addr/resi java$nio_shr
$ install repl/open/head/shar=addr/resi java$java
$ install repl/open/head/shar=addr/resi java$mlib_image_shr
$ install repl/open/head/shar=addr/resi java$awt_shr
$ install repl/open/head/shar=addr/resi java$mawt_shr
$ install repl/open/head/shar=addr/resi java$fontmanager_shr
```

Queue Full

- Use SANCP utility to check for ”queue full” and reduced ”credits available”
  - Dev I/Os - number concurrent I/Os in flight
  - Cred I/Os - number of pending I/Os because no credit available
  - QF seen - number of queue full event seen
  - Cred Avl - number of credits available (resources available)
- After a queue full event, OpenVMS will throttle, but should reset the credits once things are back to normal
  - This is a VMS bug which re-appeared, need to log a call with HP
- Use SANCP to reset credits available if too low
  - $ mc sancp set wwid /pro=hsv360 /port=5001.4380.1137.461A /default

SAN Queue Full

```
$ mc sancp show wwid
                                                    | Conn  Port  Dev   Cred  Appr | QF    Cred  User  Current
 Product ID Node WWID             Port WWID         | Cnt   I/Os  I/Os  I/Os  I/Os | Seen  Avl   Cap   QF Alg
 ---------- -------------------- -------------------- + ----  ----  ----  ----  ---- + ----  ----  ----  -------
 HSV360     5001.4380.1137.4610  5001.4380.1137.4618 | 0002  0000  0000  0000  0000 | 000E  002C  0000  T/5000
 HSV360     5001.4380.1137.4610  5001.4380.1137.4619 | 0002  0000  0001  0000  0000 | 0018  0047  0000  T/5000
 HSV360     5001.4380.1137.4610  5001.4380.1137.461A | 0002  0000  0000  0000  0000 | 0050  0018  0000  T/5000
 HSV360     5001.4380.1137.4610  5001.4380.1137.461B | 0002  0000  0000  0000  0000 | 0000  8000  0000  T/5000
 HSV360     5001.4380.1137.4610  5001.4380.1137.461C | 0002  0000  0000  0000  0000 | 0000  8000  0000  T/5000
 HSV360     5001.4380.1137.4610  5001.4380.1137.461D | 0002  0000  0001  0000  0000 | 0000  7FFF  0000  T/5000
 HSV360     5001.4380.1137.4610  5001.4380.1137.461E | 0002  0000  0000  0000  0000 | 0000  8000  0000  T/5000
 HSV360     5001.4380.1137.4610  5001.4380.1137.461F | 0002  0000  0000  0000  0000 | 0000  8000  0000  T/5000

$ mc sancp set wwid/prod=HSV360/default
                                                    | Conn  Port  Dev   Cred  Appr | QF    Cred  User  Current
 Product ID Node WWID             Port WWID         | Cnt   I/Os  I/Os  I/Os  I/Os | Seen  Avl   Cap   QF Alg
 ---------- -------------------- -------------------- + ----  ----  ----  ----  ---- + ----  ----  ----  -------
 HSV360     5001.4380.1137.4610  5001.4380.1137.4618 | 0002  0000  0001  0000  0000 | 000F  7FFF  0000  T/5000
 HSV360     5001.4380.1137.4610  5001.4380.1137.4619 | 0002  0000  0000  0000  0000 | 001A  8000  0000  T/5000
 HSV360     5001.4380.1137.4610  5001.4380.1137.461A | 0002  0000  0000  0000  0000 | 0055  17E8  0000  T/5000
 HSV360     5001.4380.1137.4610  5001.4380.1137.461B | 0002  0000  0000  0000  0000 | 0000  8000  0000  T/5000
 HSV360     5001.4380.1137.4610  5001.4380.1137.461C | 0002  0000  0002  0000  0000 | 0000  7FFE  0000  T/5000
 HSV360     5001.4380.1137.4610  5001.4380.1137.461D | 0002  0000  0000  0000  0000 | 0000  8000  0000  T/5000
 HSV360     5001.4380.1137.4610  5001.4380.1137.461E | 0002  0000  0000  0000  0000 | 0000  8000  0000  T/5000
 HSV360     5001.4380.1137.4610  5001.4380.1137.461F | 0002  0000  0000  0000  0000 | 0000  8000  0000  T/5000
```

MAKLEE

SDA Extensions

- Many SDA extensions are shipping with OpenVMS
  - Rarely documented
- Same concept for all tracing tools
  - Loadable execlet xxx$DEBUG.EXE in SYS$LOADABLE_IMAGES
  - User-mode SDA extension xxx$SDA.EXE in SYS$SHARE
- Usage
  - SDA> xxx help
  - SDA> xxx LOAD
  - SDA> xxx START TRACE /BUFFER=2000
  - SDA> xxx SHOW TRACE
  - SDA> xxx STOP TRACE

SDA Extensions (cont'd)

- Incomplete list
  - CNX Connection Manager tracing
  - FC Fibrechannel tracing
  - FLT Alignment Fault tracing
  - IO Buffered and Direct IO tracing
  - LCK Lock Manager tracing
  - LNM Logical Name tracing
  - RMS RMS tracing
  - SPL Spinlock tracing
  - RDB Rdb Lock tracing
  - TR Debug and trace print

Buffered and Direct I/O Tracing

- Drill down on MONITOR IO if high I/O rate
- Who is doing all those buffered I/Os?
- Which device is doing most of the direct I/Os?
- Collect I/O per process
  - SDA> IO START TRACE
  - SDA> IO START COLLECT /PROCESS
  - SDA> IO SHOW COLLECT
- Collect I/O per device
  - SDA> IO START TRACE
  - SDA> IO START COLLECT /DEVICE
  - SDA> IO SHOW COLLECT

Buffered and Direct I/O Tracing (cont'd)

- Collect file system activity (XQP)
  - SDA> TR LOAD
  - SDA> TR START TRACE
  - SDA> IO START TRACE /XQP
  - SDA> TR SHOW TRACE

Lock Manager Tracing

- Drills down MONITOR LOCK and MONITOR DLOCK if high locking activity
- Which resource tree is having the highest activity?
- Where are resources mastered?
- How busy is the dedicated lock manager?
- Which process does heavy locking?
- Lock manager statistic example
  - SDA> LCK STATISTIC
- Active resource tree example
  - SDA> LCK SHOW ACTIVE

Lock Manager Tracing (cont'd)

- Dedicated lock manager usage
  - SDA> LCK SHOW LCKMGR /INTERVAL=10 /REP=10
- Per-Process lock activity example
  - SDA> LCK LOAD
  - SDA> LCK START TRACE
  - SDA> LCK START COLLECT /PROCESS
  - SDA> LCK SHOW COLLECT

Rdb Lock Decoding and Tracing

- Only RDB$SDA, no loadable execlet required
- Ships with sources and build procedure in SYS$EXAMPLES (V7.3-2) and also a ready image (V8.1 or better)
- Understands Rdb resources and decodes them into verbose readable format
- Can quickly identify what Rdb database, page and/or record lock is blocking a process
  - SDA> RDB SHOW PROCESS /ID=xxx /WAITING /CONVERT
- Can also monitor Rdb resource contentions
  - SDA> RDB SHOW CONTENTION /INTERVAL=0.5
- Can identify active and heavily Rdb databases
  - SDA> RDB SHOW ACTIVE