Customer Success Stories

Chinmay Ghosh
OpenVMS Engineering

Agenda

- **Existing setups**
- **Issues during migration**
- **Issues post migration**
- **Performance comparison and tuning**
- **Key takeaways**
- **Revisit top 10 porting considerations**
- **Customer Lab**

Customer 1

- **Large telecom customer**
- **OpenVMS is used for billing application**
- **Mission critical**

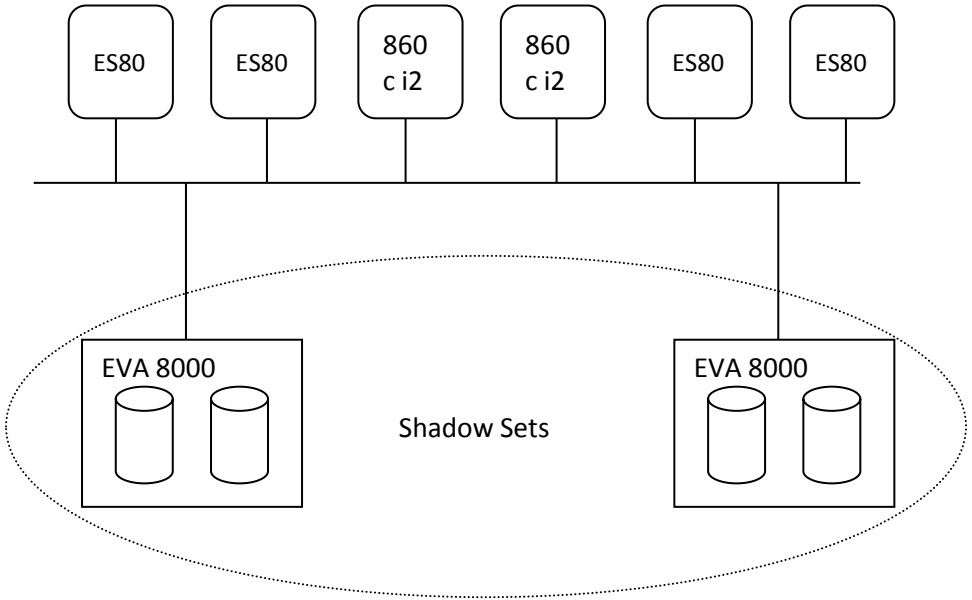
Customer 1- Existing and proposed setups

Existing Environment

- Production : 2 x Alpha ES80
- EVA8000 SAN storage
- DR : same as production

Migrated Environment

- Production : 2 x BL860c i2
- EVA8000 SAN storage
- DR : same as production



Software

COBOL, Fortran, C, ACMS, DECforms, Oracle Rdb, Oracle CDD etc

Issues faced during migration

SLS not available in I64

- Used ABS

COBOL keywords used in source modules(SYMBOL, RETURN-CODE)

- Recompiled with “/NOXOPEN” and “/RESER=NO200X/SWITCH=DC_NO_SYMBOL”.

Floating points

- Recompiled entire code with D_FLOAT

Conditionalized code using __ALPHA

- Modified and added “|| __IA64”

COBOL MOVE CORRESPONDING issue

- Modified source code to have proper data type

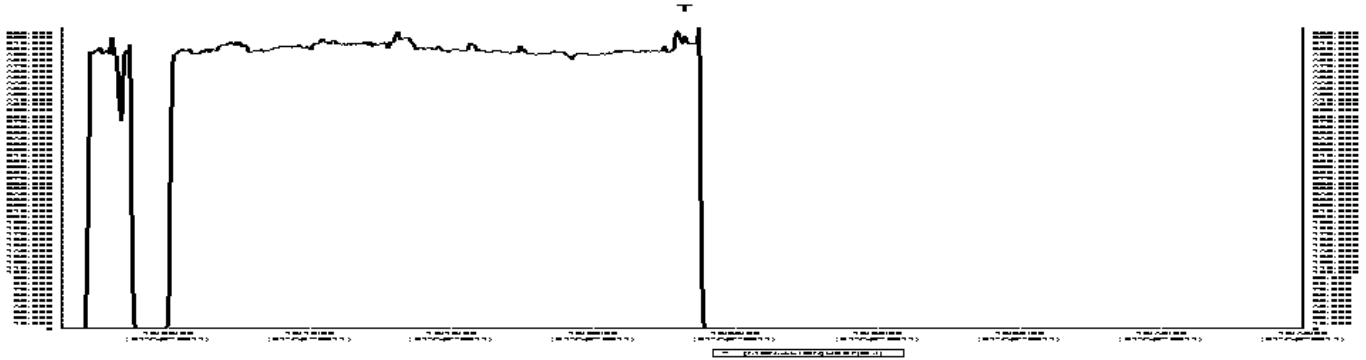
%ILINK-E-INVOVRINI, incompatible multiple initializations for overlaid section

- initialize the database handles in only one of the modules
- Use /NOINITIALIZE_HANDLES with other SQLMOD/precompiler modules

Issues faced post migration

Test	Alpha	Integrity
RMS random write	4:72	2:35
Rdb random read	8:50	18:54

Alignment faults = ~500,000/Second

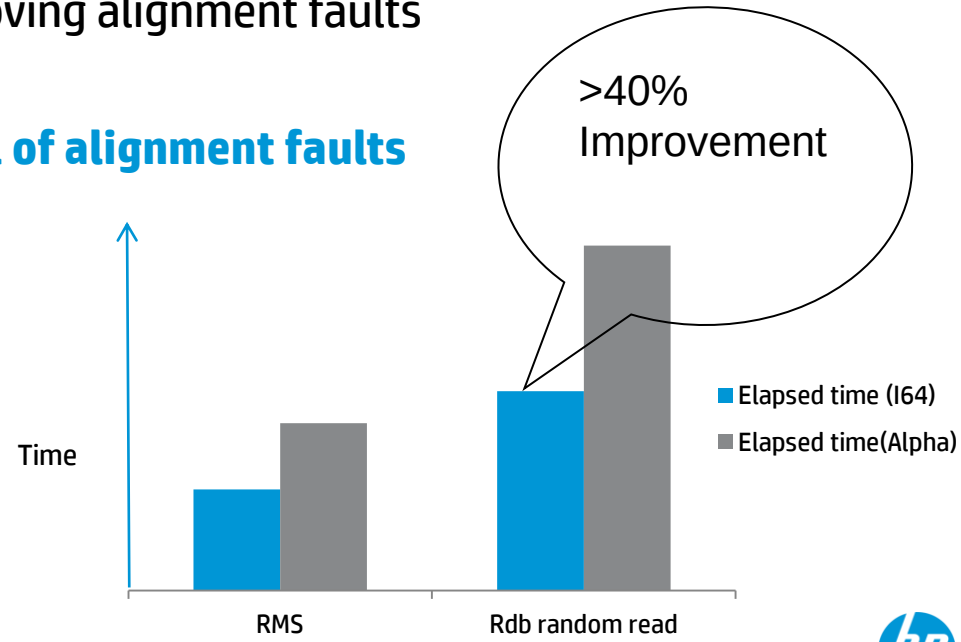


Asked to provide FLT trace

Issues faced post migration – Analysis and Resolution

- **Alignment faults were from COBOL initialize statement and LIBRTL**
  - Resolved in latest releases of COBOL compiler
  - Provided a new LIBRTL imaged by removing alignment faults

- **Performance comparison after removal of alignment faults**



- **Improvement = >40%**

Key takeaways

- **Take a complete inventory of application stack before porting**
- ***Identify hardware dependent codes***
- ***Use of latest COBOL compiler on Alpha would have reduced porting effort***
- **Very little modifications in source code**
- ***Alignments faults are very costly on I64***
- ***Need to have test tool to validate performance***
- ***i2 servers can deliver far better performance***

Customer 2

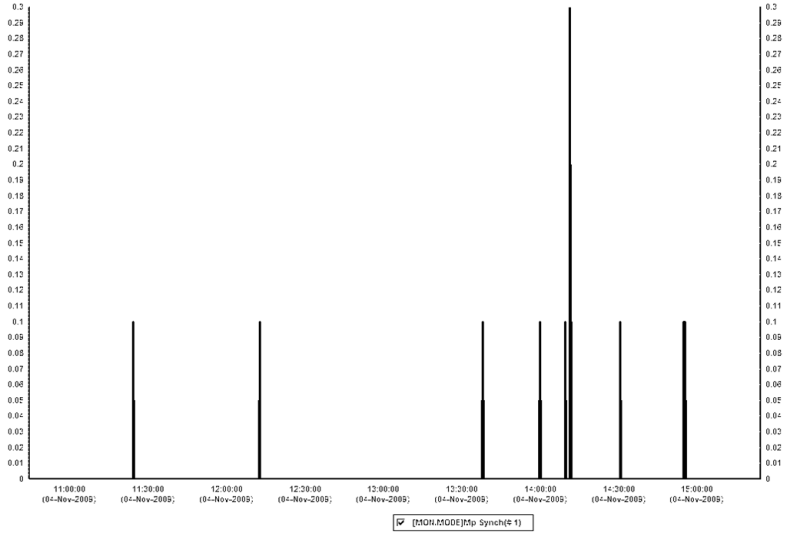
- **Provides travel solution**
- **Performance is very critical**

Customer 2 – Existing and proposed setups

- **Existing Environment**
  - System : Alpha ES40, 2 CPU
- **Migrated Environment**
  - System : HP rx3600 (1.60GHz/6.0MB), 2 CPU
- **Local storage**
- **Application language – BASIC**
- **Issue : Application execution duration on I64 > 4 times compared to Alpha**

Performance analysis

- **T4 data**
  - No apparent issue. MPSYNC very minimal
- **PRF data**
  - 42.62% time in LIBRTL
  - 14.55% time in EXCEPTION
- **EXC trace**
  - Most of the LIBRTL calls were to stack unwinding routines
  - Resulted from RMS-E- RECNOTFOUND



Analysis contd..

- **Encountered exception in $GET when the record is not present**
  - Signaled with exception code RECNOTFOUND from BASRTL while using GET & FIND
    - ➢Resulted in exception handling and stack unwinding
- **Resolution**
  - Modified BASRTL to return error RMS$_EOF instead of signaling
  - BASIC application can be modified with

    nosigconds%(1) = RMS$_RNF ! implicit integer array

    call dbasic$io_no_signal(chan%, nosigconds%())
  - Introduced logical DBASIC$IO_NOSIGNAL (No source code modification)
- **Results**
  - Sample application showed 10 times improvement

Key takeaways

- ***SDA extensions (PRF,EXC, FLT etc..) are very useful***
- **May need to use different monitoring tools**
- ***EXCEPTION handling and stack unwinding are very costly on Integrity***
  - Instead of signaling returning the error helps
- ***Need to have test tool to validate performance***

Customer 3

- **Provides telecom service**
- **Mission critical**
- **Performance is critical**

Customer 3 – Existing and proposed setups

- **Existing Environment :**
  - Production : AlphaServer ES40 (4GB with 2 cpu(s))
- **Migrated Environment :**
  - Production : BL860c i2, OpenVMS 8.4
- **Storage :**
  - EVA8400
- **Software :**

  COBOL, PASCAL, Fortran, C, ACMS, DECforms, Tuxedo, Oracle Rdb, Oracle CDD,XML-C etc.

Issues faced during migration

- **DEC ECP not available**
  - Advised to use TDC
- **Tuxedo startup error %LIB-E-INSVIRMEM**
  - Readjusted OpenVMS SYSGEN, quota and Tuxedo parameters
- **Issues related to floating points**
  - Suggested to use either IEEE or G_FLOAT uniformly
- **Issue with old compiler while using "/pointer=long=argv"**
  - Resolved using latest compilers

Issues faced post migration

- **Tuxedo thread running at real-time priority 63**
  - Known problem with Tuxedo. Oracle has a patch.

- **Performance issue with Oracle Rdb**
  - Installed latest version of Rdb
  - Consulted Rdb support team

Rdb performance Issue (1/4)

Problem Statement

- Customer observed poor performance while executing SQL queries on I64

Alpha, Local DB	I64, Local DB	Alpha, DB on remote I64
10 Sec.	24 Sec.	18 Sec.

Analysis

- Only batch jobs were taking more time. Interactive jobs were executing much faster.

Suggestion

- Customer was asked to provide the location of the batch job log files.
- Asked to relocate the log files if they are in the system disk.

Rdb performance issues (2/4)

Problem Statement

- After relocating the log files from system disk customer observed better performance on I64. Expectation was better performance for I64 where database on remote Alpha.

Alpha, Local DB	I64, Local DB	Alpha, DB on remote I64	Alpha, DB on remote Alpha
10 Sec.	3 Sec.	13 Sec.	13 Sec.

Analysis

- Since I64 local query takes ~3 Sec. customer was expecting ~7 sec for remote queries.
- Customer was executing ROLLBACK after every SELECT query

Suggestion

- Customer was asked to provide T4 data
- Asked to reduce the number of ROLLBACK calls.
- Asked to tune RDB$REMOTE_BUFFER_SIZE or SQL_NETWORK_BUFFER_SIZE
- Asked to keep .RDB and .RDA files on different disk

Rdb performance issues (3/4)

Problem Statement

- After increasing buffer sizes customer saw improvement in remote execution, but still not up to the expectation.

I64, Local DB	Alpha, Local DB	Alpha, DB on remote I64	Alpha, DB on remote Alpha
3 Sec.	10 Sec.	11.5 Sec.	13 Sec.

Analysis

- Customer didn't implement the other suggestion to reduce the number of ROLLBAK calls.
- T4 data didn't show any specific problem with the system

Suggestion

- Customer was asked again to reduce the number of ROLLBACKS

Rdb performance issues (4/4)

Final result

- customer received much improved performance once he reduced the number of calls to ROLLBACK to 1 for every 15 SELECT queries.

I64, Local DB	Alpha, Local DB	Alpha, DB on remote I64	Alpha, DB on remote Alpha
3 Sec.	10Sec.	8.7 Sec.	13 Sec.

Key takeaways

- ***Take a complete inventory of application stack before porting***
- ***Adjusting SYSGEN parameters may be required***
- ***Need to use several performance monitoring tool***
- **Need to have test tool to validate performance**
- **Try to keep the log files away from the system disk**

Customer 4

- **Telecom solution provider**
- **Performance is very critical**

Customer 4 – Existing and proposed setup

- **Existing Environment : HP BL860c (1.59GHz/12MB, 4 CPUs)**
- **Migrated Environment : HP BL860c i2 (1.73GHz/24MB, 8 CPUs) in c7000 enclosure**
- **Storage : EVA4400**
- **ISV delivers the following configurations**
  - Low end: RX2660 with MSA60
  - Medium: Dual node RX2660 with MSA60 or EVA4400, BL860c with EVA4400
  - High end: 3 to 16 nodes BL860c cluster with EVA4400 up to EVA8400

Analysis on BL860c(4 CPUs)

- **MP sync time was considerably higher.**
  - The major contention for SCHED and LCKMGR spinlocks
  - SCHED was taken for AST queuing, hibernate, event flag posting and resched interrupt
    - Deferred SCHED was considered to be a good option for this customer
- **Around 20 processes in the COM queue**
  - Indicates that more processes are waiting for CPU. Suggested increasing the CPUs
- **Heavy local lock operations**
  - Suggested dedicated lock manager if number of CPUs more than 4
- **OpenVMS executive not responsible for excessive CPU usage**
- **Suggested more CPUs to improve performance**
- **Hyper threading result was not encouraging**
- **Pointed out user processes running in the kernel mode and consuming the most CPU time**

Comparison of performance on a clustered BL860c(4 CPUs) & standalone BL860c i2

- Good amount of improvement (almost double) in user mode time in BL860C i2 (from 168/400 to 400/800).

- Slight reduction in kernel mode time and interrupt time, thereby providing more CPU time for application.

- During peak load 8 processes in “CUR” state - indicates that the increase in the number of CPUs is helping the application.

- Increase in the logical name translations which is almost doubled.

Resource Utilization	Clustered BL860c (4 CPUs)	Standalone BL860c i2 (8 CPUs)
Time spent in each mode	40% kernel 42% user 11% interrupt 2% MP synch	36% kernel 50% user 2% interrupt 9% MP synch
Cur/COM state	4/20	8/20
Lock Conversions	70,000	70,000
Locks in total	46,000	86,000
TQE	385	485
Logical Name Translations	2,200	4,200

Comparison of performance on a clustered BL860c(4 CPUs) & clustered BL860c i2

- Noticeable amount of change when the node is in cluster environment.
- Major increase in MP synch time from 9% to 14.5%.
- Decline in user mode time.
- Increase in Buffered I/O rate.
- Suggested to enable dedicated lock manager in order to control the MP sync.

Resource Utilization	Clustered BL860c (4 CPUs)	Standalone BL860c i2 (8 CPUs)	Clustered BL860c i2 (8 CPUs)
Time spent in each mode	40% kernel 42% user 11% interrupt 2% MP synch	36% kernel 50% user 2% interrupt 9% MP synch	37% kernel 34%user 6.5% interrupt 14.5% MP sync
Buffered I/O rate	17,500	5,000	24,000
Lock Conversions	70,000	70,000	56,000
Locks in total	46,000	86,000	86,000
TQE	385	485	520
Logical Name Translations	2,200	4,200	2,700

Comparison of performance on clustered BL860c i2 with Dedicated Lock Manager

- Reduction(around 40%) in the time spent in MP sync.

- Reduction in alignment faults.

- Improvement seen in Buffered I/O, logical name translations and timer load which indicates more work was done.

- Kernel mode is higher because one CPU is dedicated for lock manager activities.

- Dedicated lock manager enabled on one node. Suggested enabling in both the cluster nodes.

- Pointed out processes consuming more CPU in each of the test cases.

Resource Utilization	Clustered BL860c (4 CPUs)	Standalone BL860c i2 (8 CPUs)	Clustered BL860c i2 (8 CPUs)	Clustered i2 with DLM
Time spent in each mode	40% kernel 42% user 11% interrupt 2% MP sync	36% kernel 50% user 2% interrupt 9% MP sync	37% kernel 34%user 6.5% interrupt 14.5% MP sync	50% kernel 35% user 5.7% interrupt 8.7% MP sync
Buffered I/O	17,500	5,000	24,000	28,000
Lock Conversions	70,000	70,000	56,000	64,000
Total Locks	46,000	86,000	86,000	85,155
TQE	385	485	520	700
Logical Name Translations	2,200	4,200	2,700	3600

Conclusions

- **BL860c 4 CPUs setup - Increase in number of CPUs should improve performance.**
- **Standalone BL860c i2 - Good improvement in performance.**
- **clustered BL860c i2 - Slight decrease in performance compared to standalone but still good compared to BL860c setup.**
- **Dedicated lock manager enabled on one node – Good improvement in performance**
- **Suggested to enable dedicated lock manager on both the nodes.**
- **Hyper threading – No encouraging result both in BL860c and BL860c i2 setup.**
- **Suggested to optimize the application to reduce kernel mode and CPU time.**

Few more interesting issues..

C++ compiler – Problem with a class derived from an empty base class

- On alpha
  - creates an extraneous eight byte offset before the defined member variable
  - Increases size of the derived class
  - Because of old ARM object model
  - Can impose interoperability problem with other languages

- On Integrity
  - Model is ABI
  - Different from /MODEL=ARM and /MODEL=ANSI
  - Doesn't add extraneous eight byte

Few more interesting issues contd..

- Resolution
  - Integrity behavior is perfect
  - ANSI model on Alpha is modified to provide I64 like feature
  - On Alpha recompile and relink all sources and libraries with “/model=ansi” qualifier
  - Development is in progress to make it the default behavior on Alpha

Top 10 Porting Considerations…

1. Do a complete inventory of all 3rd party software products and HP OpenVMS layered products before you start your port. These may be required for development, QA, or production deployment. Ensure you know the status of each of these on OpenVMS I64 before you go too far in your port.
2. Make sure your application builds cleanly and runs on OpenVMS Alpha V7.3-1 (or greater) using the latest released compilers and development tools
3. Check for hardware architecture consistently in all source code and DCL command procedures
4. Have automated regression tests as much as possible and clearly documented manual regression tests where necessary
5. Document your build procedure / process
6. Read the Porting Guide and various Release Notes (Really do it!)
7. Update any Fortran 77 code to Fortran 90
8. Reduce / Recode / eliminate any Alpha Macro (Macro64 code) and PL/I
9. Where possible, use IEEE floating point
10. Have a working development / QA environment on OpenVMS Alpha near by so you can compare results easily between Alpha and Integrity systems.

11. **Sit back and just… Re-compile, Re-Link, and run :-)**

Customer Lab

To help customers in

- Setting up similar environment
- Porting and migration
- POC preparation
- Fixing issues related to migration
- Benchmarking application performance
- Tuning

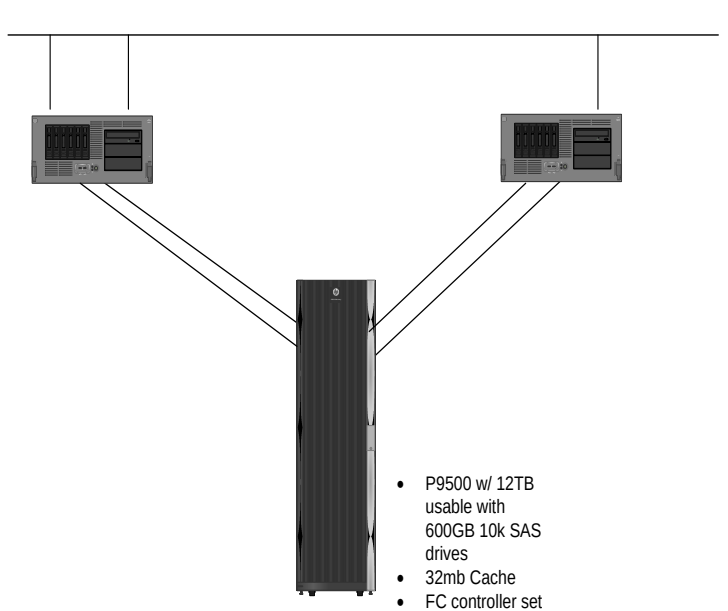
Achievements

- Several customers used this lab successfully to port or benchmark their applications

Domain – Financial Services

- **Existing Setup**
  - Alpha GS1280 and EVA
- **Targeted environment**
  - BL890C i2/BL870C i2
- **Storage**
  - 600 SAS drives, RAID 1
- **Expectation**
  - Achieve better performance than GS1280
- **Achievement**
  - BL890C i2
    - 45% improvement in 4Gb FC over GS1280
    - 55% improvement in 8Gb FC over GS1280
  - BL870C i2
    - 25% improvement in 4Gb FC over GS1280
    - 35% improvement in 8Gb FC over GS1280

- (1) 870 i2 w/ 64GB and 2 processors with 4 cores each
- (1) RX2800I2 with 64gb memory and dual 9340 processors
- C-Class Chassis w/ 2 Flex 10 Ethernet
- 2 VC SAN Modules
- QLOGIC HBA
- OPENVMS 8.4
- VMS Cluster
- SAN multiipath



- P9500 w/ 12TB usable with 600GB 10k SAS drives
- 32mb Cache
- FC controller set for 4GB
- Dual path cabling

Further Information and contact

Office of OpenVMS Programs : openvms.programs@hp.com

Q&A

Thank you