PLEASE STAND BY
325
575
50
300
35
30
20
25

OpenVMS on Integrity Servers Part II

Thomas Siebold
Sr. Technology Consultant, HP

Thomas Siebold

Sr. Technology Consultant

Transition Engineering & Consulting

thomas.siebold@hp.com

Agenda

- OpenVMS on Itanium®
  - Status
  - Application/ISV migration

Schedule

Change of Name

OpenVMS on Itanium®

Will be called

„hp OpenVMS Industry Standard 64“

(Official Name)

or

„OpenVMS I64“

(Informal Name)

HP OpenVMS the Road to Itanium®

OpenVMS V8.2 Production Quality Release
(Alpha & Integrity) 2H2004

Mixed Alpha Integrity Superdome Cluster
January, 2004

16Processor System Boot
January, 2004

OpenVMS V8.1 Evaluation Release
December, 2003

Runs in a Superdome Cell
November, 2003

1st ISV Applications ported
August, 2003

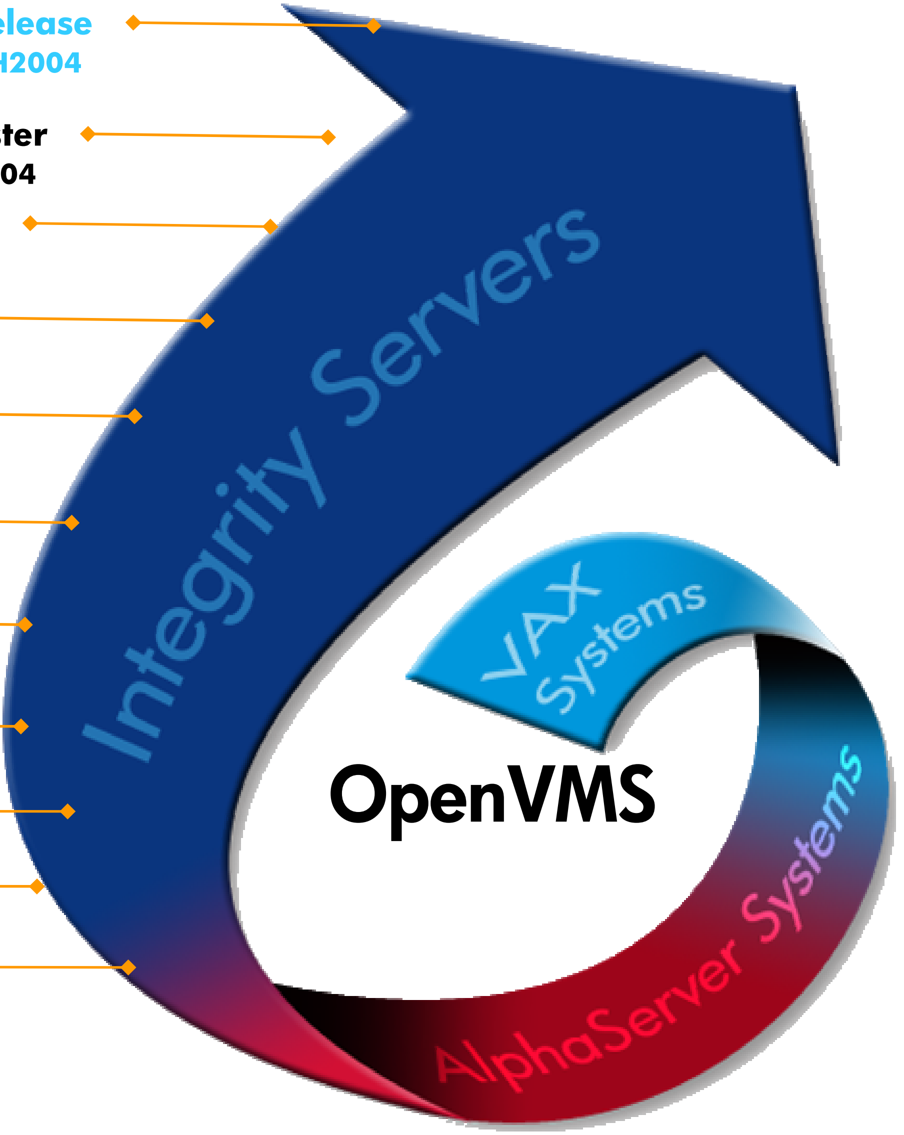
OpenVMS V8.0 in DSPP
August, 2003

OpenVMS V8.0 Evaluation Release
June 30, 2003

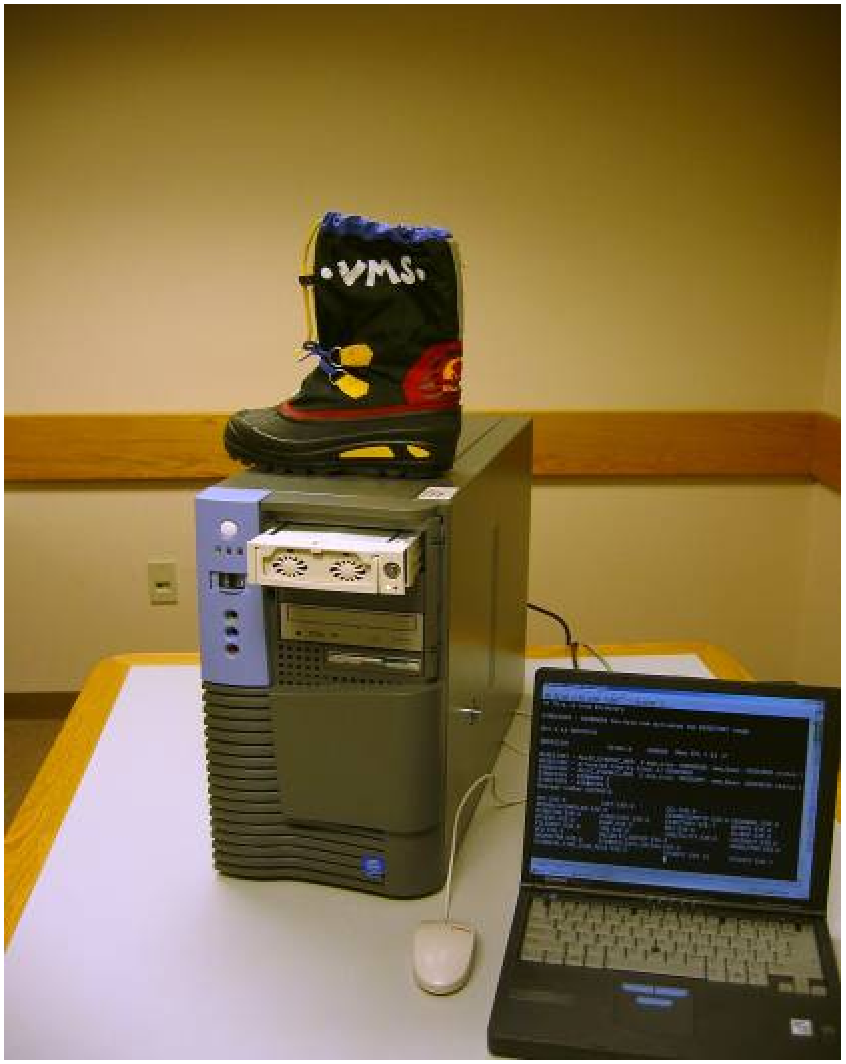
1st Application Port & Mixed Cluster
May 15, 2003

Boot to rx 2600 server
March 17, 2003

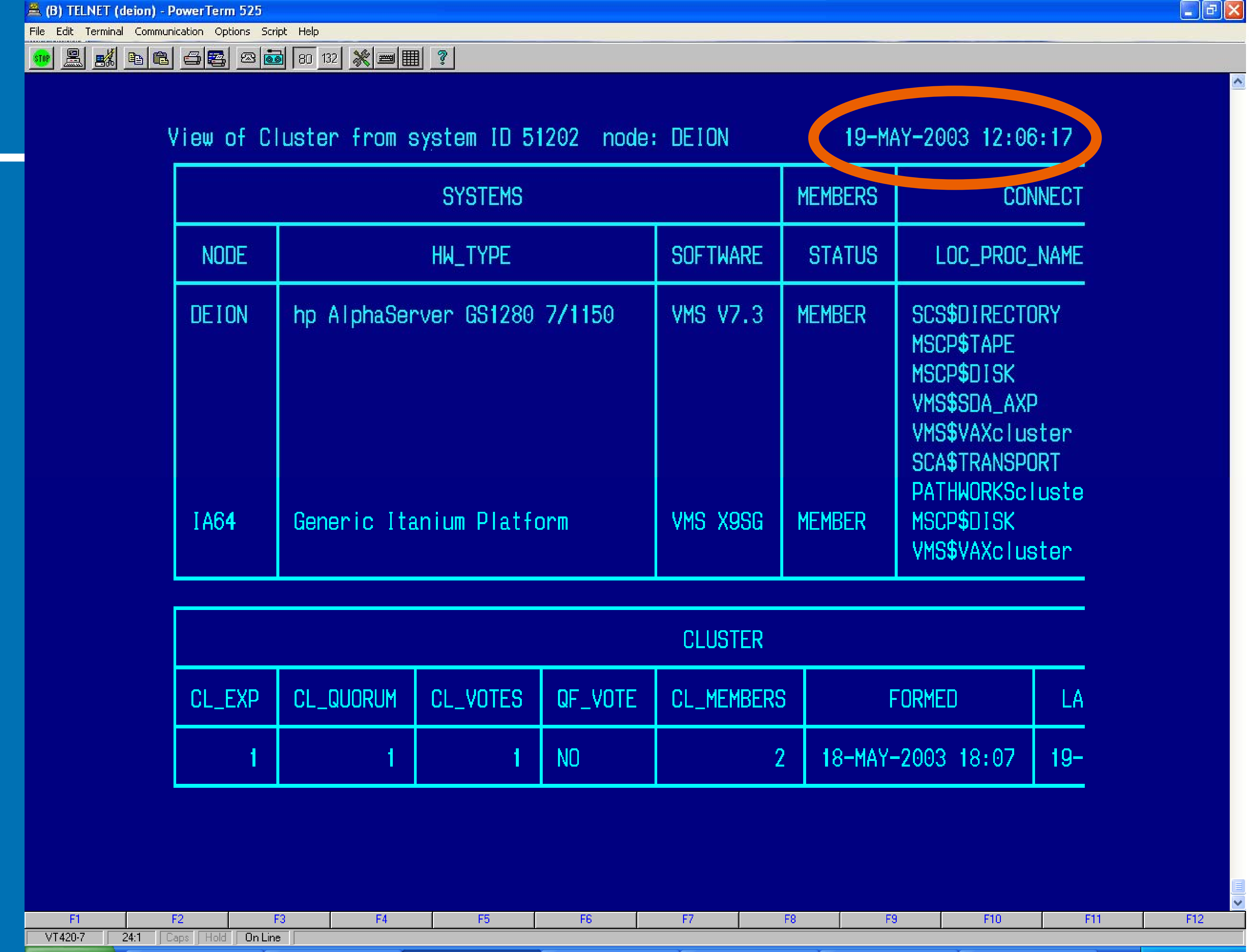
1st Boot to Itanium® system
January 31, 2003



OpenVMS on Itanium® -- 31. Januar 2003 15:31



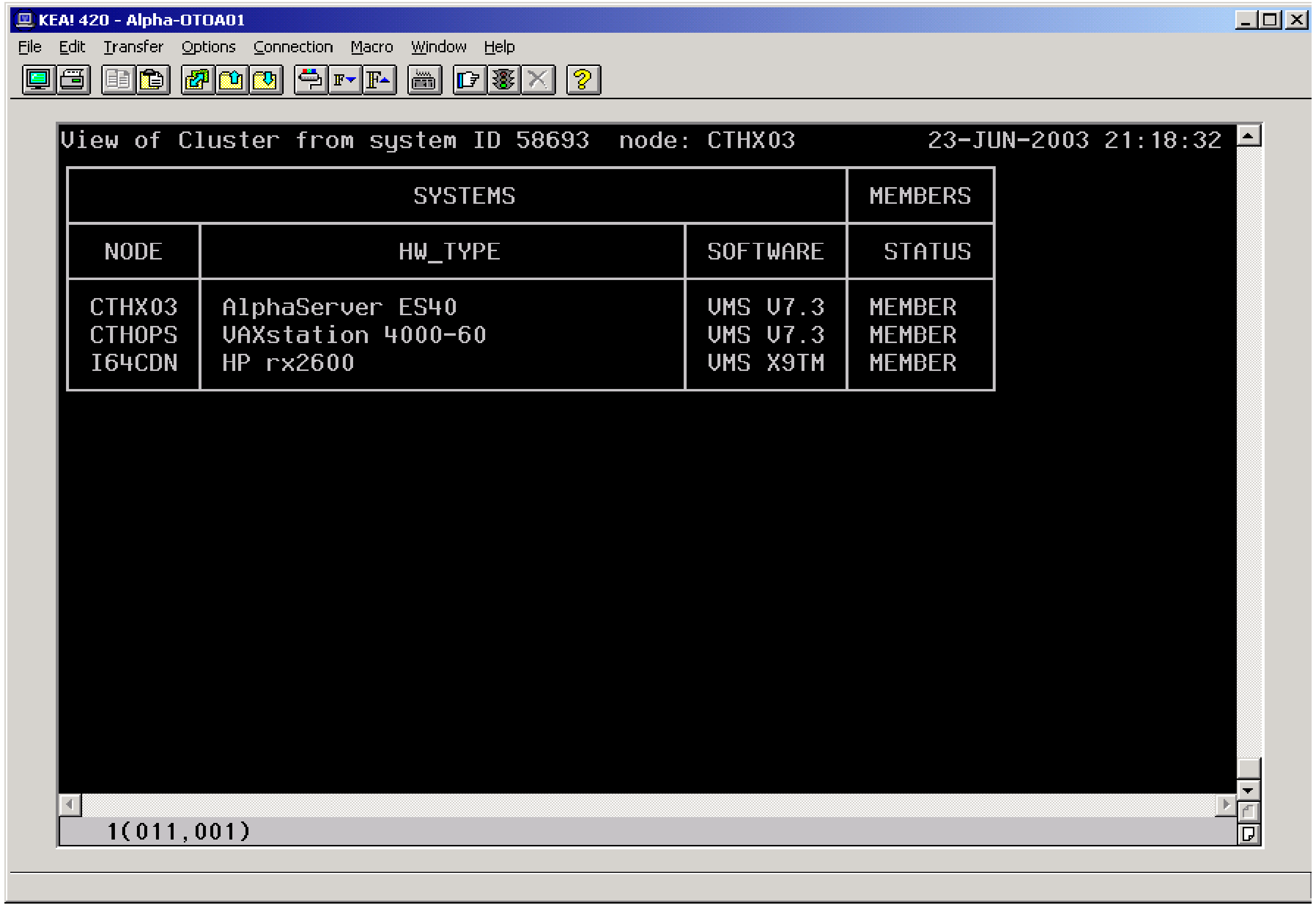
June 25, 2003


(B) TELNET (deion) - PowerTerm 525
File Edit Terminal Communication Options Script Help
F1 F2 F3 F4 F5 F6 F7 F8 F9 F10 F11 F12
VT420-7 24:1 Caps Hold On Line

```
View of Cluster from system ID 51202  node: DEION          19-MAY-2003 12:06:17
+-----------------------------------------------------+---------+----------------
|                       SYSTEMS                       | MEMBERS |        CONNECT
+--------+--------------------------------+-----------+---------+----------------
|  NODE  |            HW_TYPE             | SOFTWARE  | STATUS  |   LOC_PROC_NAME
+--------+--------------------------------+-----------+---------+----------------
| DEION  | hp AlphaServer GS1280 7/1150   | VMS V7.3  | MEMBER  | SCS$DIRECTORY
|        |                                |           |         | MSCP$TAPE
|        |                                |           |         | MSCP$DISK
|        |                                |           |         | VMS$SDA_AXP
|        |                                |           |         | VMS$VAXcluster
|        |                                |           |         | SCA$TRANSPORT
|        |                                |           |         | PATHWORKScluste
| IA64   | Generic Itanium Platform       | VMS X9SG  | MEMBER  | MSCP$DISK
|        |                                |           |         | VMS$VAXcluster
+--------+--------------------------------+-----------+---------+----------------

+--------------------------------------------------------------------------------
|                                  CLUSTER
+--------+-----------+----------+---------+------------+-------------------+-----
| CL_EXP | CL_QUORUM | CL_VOTES | QF_VOTE | CL_MEMBERS |      FORMED       |  LA
+--------+-----------+----------+---------+------------+-------------------+-----
|      1 |         1 |        1 | NO      |          2 | 18-MAY-2003 18:07 | 19-
+--------+-----------+----------+---------+------------+-------------------+-----
```

OpenVMS VAX-Alpha-IA64 Cluster Demo



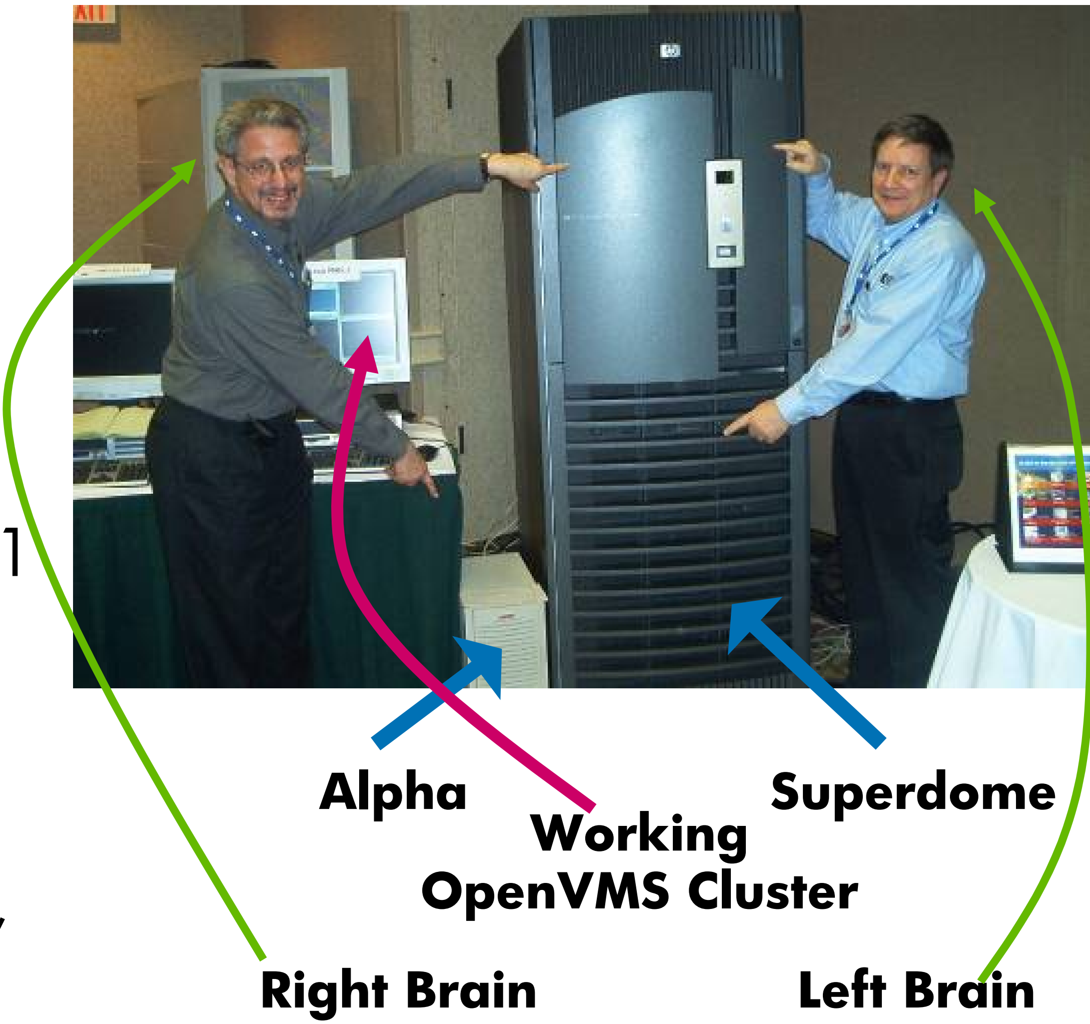
```
View of Cluster from system ID 58693  node: CTHX03          23-JUN-2003 21:18:32
+-----------------------------------------------------+---------+
|                      SYSTEMS                        | MEMBERS |
+--------+---------------------------------+----------+---------+
|  NODE  |             HW_TYPE             | SOFTWARE |  STATUS |
+--------+---------------------------------+----------+---------+
| CTHX03 | AlphaServer ES40                | VMS V7.3 | MEMBER  |
| CTHOPS | VAXstation 4000-60              | VMS V7.3 | MEMBER  |
| I64CDN | HP rx2600                       | VMS X9TM | MEMBER  |
+--------+---------------------------------+----------+---------+

   1(011,001)
```

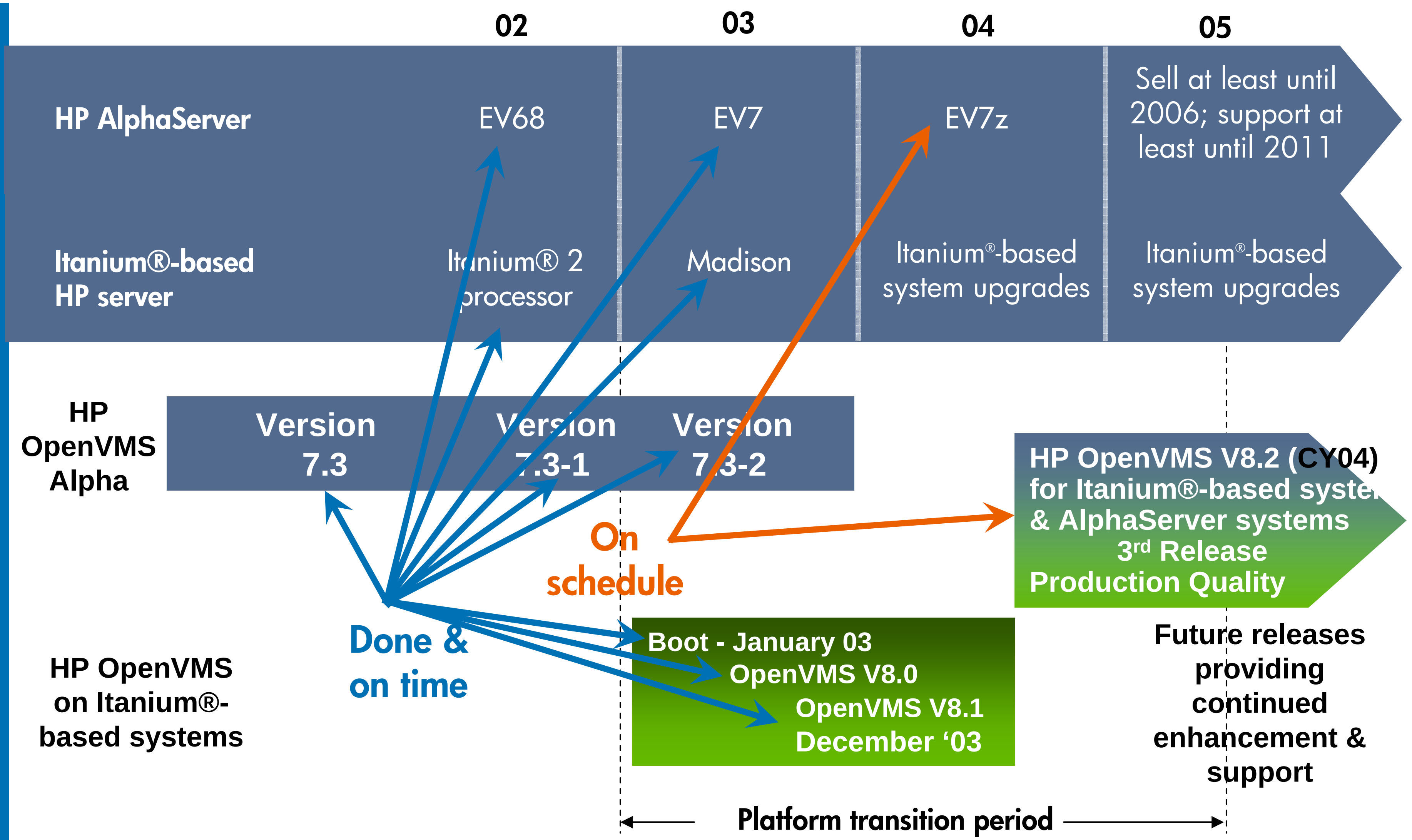
Even though clustering VAX with Alpha-IA64 will not be supported, Engineering is not doing anything to prevent it from working. The above proves it now works. Btw - Clustering three totally different HW architectures with a fully shared read-write active-active-active cluster file system with one OS (OpenVMS) is way cool. ☺

OpenVMS @ Analyst Summit 1/13/04

- OpenVMS on all multi-OS slides
- Rich Marcello discussed OpenVMS Integrity progress and partner support
- Demo with OpenVMS V8.1 and Superdome/Alpha cluster
- Some reactions:
  - "Cool"
  - "Two different OS versions?"
  - "Impressive"
  - "It's good to see OpenVMS back on the front burner."



OpenVMS Roadmap


02
03
04
05
HP AlphaServer
EV68
EV7
EV7z
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
On schedule
Done & on time
HP OpenVMS V8.2 (CY04) for Itanium®-based syste
& AlphaServer systems
3rd Release
Production Quality
HP OpenVMS on Itanium®-based systems
Boot - January 03
OpenVMS V8.0
OpenVMS V8.1 December '03
Future releases providing continued enhancement & support
Platform transition period

OpenVMS on Integrity Servers 2005 Coming Attractions

- V8.2 Code Freeze: April (planned)
- V8.2 External FT: June (planned)
- V8.2 FRS: Q4'CY04 (planned)

OpenVMS for Integrity Servers Rollout Plan

- V8.2:
  - rx1600, rx2600, rx4640 (4-8 CPUs!)
  - No cell based systems
  - Madison 6M CPU & dual CPU module (Hondo)
  - No more McKinley support!
  - Q4 CY04
- V8.x:
  - Add more systems
  - Cell based systems: rx7620, rx8620, Superdome
  - 50bit physical addressing needed for cell based systems
  - Madison 9M and Montecito CPUs, Arches chip set
  - „Performance“

What is being ported ??
And how ??

What are porting and How?

- Single source code base to produce the Alpha and Intel® Itanium® architecture variants
  - About 95% of the code is common
  - Support for Itanium® architecture added to OpenVMS AlphaServer code base
  - Releases created from the same sources for both architectures
  - All non-hardware dependent and performance improvements to be incorporated into both versions without multiple changes to the source code and to minimize the time required to perform qualification testing.
- The first Itanium® architecture release will reflect on-going OpenVMS development work
- Allows ISVs and end-user developers to continue using their current and future Alpha systems while migrating to the future Itanium® platforms. Integrating Integrity Servers will be cost effective
- OpenVMS is made more portable and maintainable by replacing VAX assembler
- OpenVMS is made more open to exchanging code with other systems by using new standards

Current Itanium Porting Status

- Native Tools
  - C, Bliss, Cobol, Fortran, DECset, SWS (Apache)
  - Linker, SDA, ….
  - GNV, Kerberos, Freeware Tools, ….
- Console
  - Serial line
  - Management Port (no graphics, keyboard, or mouse yet)
- Booted on
  - I2000, rx2600 (McKinley & Madison), zx2000 (McKinley), rx4640(Madison), rx1600(Deerfield)

Current Itanium Porting status

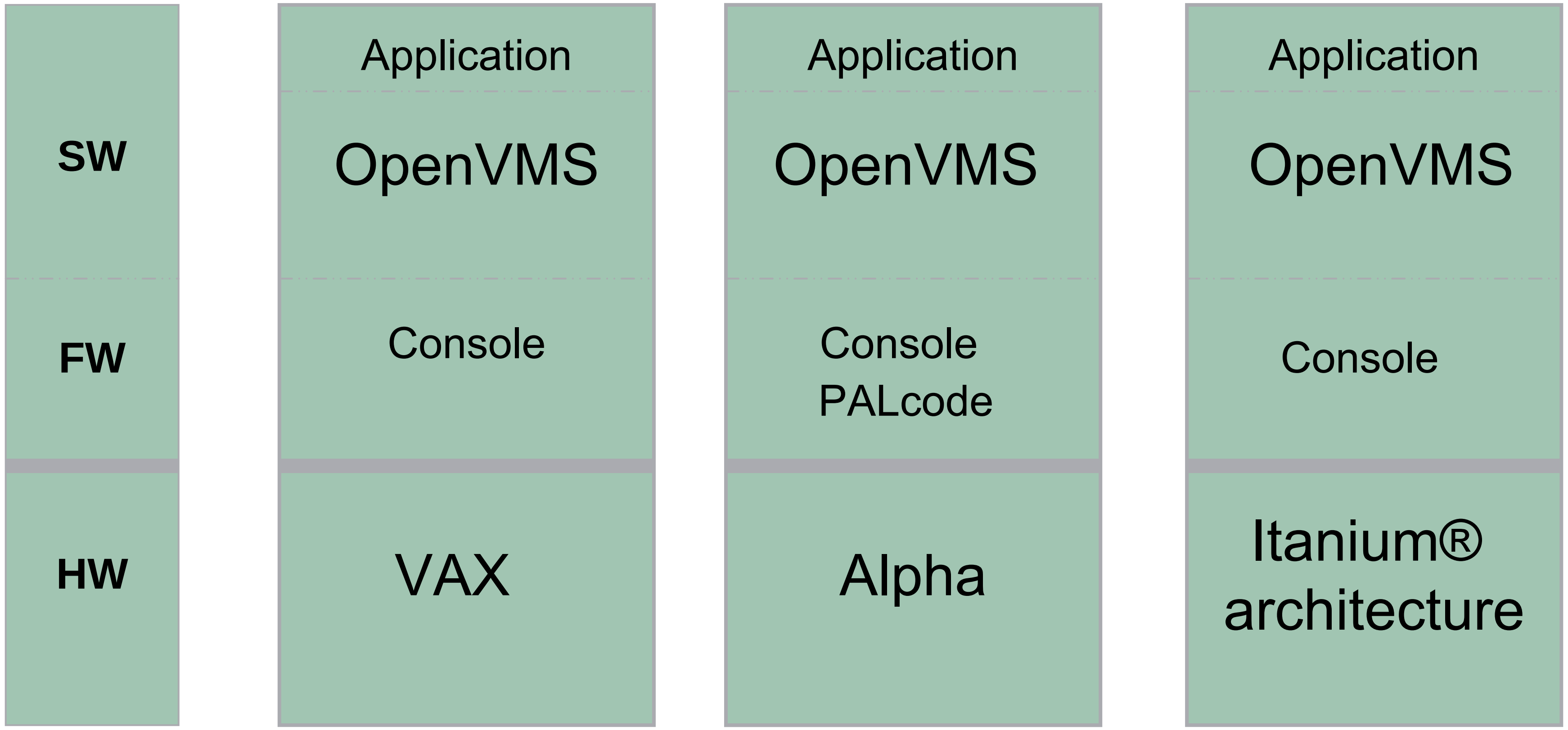
- What is not yet working
  - Edit/Teco
  - Delta Debugger
  - System Code Debugger (SCD)
  - Security Server
  - Registry Server
  - ACME Server
  - Shadowing
  - Cluster Satellite Booting
  - Java

Challenges

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



PALcall Builtins -- Replacement

- Most, but not all, PALcall builtins result in system service calls on IPF
  - [C] __PAL_BPT(); => [asm] break
  - [C] rd_ps = __PAL_RD_PS(); => [asm] br.call br0 = SYS$PAL_RD_PS
- Some service calls are generated directly by compilers
- Otherwise, there are definition files
    - C - builtins.h ➔ pal_builtins.h ➔ pal_services.h
    - BLISS - builtins.b32
    - MACRO - ia64_macros.mar
- We can determine whatever is best in each case
- Changes can be made anytime

Remove from head of queue, interlocked

- **VAX**: microcoded instruction REMQHI
- **Alpha**: CALL_PAL REMQHIL
- **Itanium® Architecture**: *OpenVMS* system service SYS$PAL_REMQHIL

WHAT did you say ? Or The nomenclature game

<u>Bytes</u>	<u>Intel®</u>	<u>Alpha</u>
1	~~***byte***~~	***byte***
2	~~halfword~~	***word***
4	~~***word***~~	longword
8	~~doubleword~~	***quadword***
16	~~***quadword***~~	octaword

Migrating Applications

Alpha Compilers

- Latest/Next Releases on Alpha Platform
  - C V6.5, C++ V6.5
  - Fortran V7.5 (F90)
  - Basic V1.5
  - COBOL V2.8
  - Java 1.4.1
    - (1.4.2 is in beta)
  - Pascal V5.8
    - V5.9 Planned for mid-2004
- HP recommends that you build your applications on OpenVMS Alpha using these versions of the compilers prior to starting your port to OpenVMS I64

OpenVMS on Integrity Servers Compiler Plans

- C
  - Itanium® architecture implementation of OpenVMS HP C V6.5 compiler
- C++
  - Based on the same front end compiler technology as HP C++
  - This is not a port of HP C++ V6.5 but it will be able to compile the same source code as HP C++ V6.5
- COBOL, BASIC, PASCAL, BLISS
  - Itanium® architecture implementations of the current OpenVMS compilers

OpenVMS on Integrity Servers Compiler Plans

- FORTRAN
  - Itanium® architecture implementation of the current OpenVMS Fortran 90 compiler
- Java
  - Itanium® architecture implementation of J2SE V1.4.2
- IMACRO
  - Compiles ported VAX Macro-32 code for Itanium® architecture
  - Itanium® architecture equivalent of AMACRO
- ADA
  - We will provide an Ada-95 compiler
  - We will not port the existing Ada-83 compiler

Binary Translator

- Will translate Alpha OpenVMS binary images and libraries linked under all OpenVMS versions from 6.2 to current version
- Will translate a VESTed image that was translated by DECmigrate from a VAX binary image
- Will translate images written in C, C++, FORTRAN, or COBOL
  - Will not translate applications written BASIC, Pascal, PL/1, or Ada
- Restrictions:
  - Alpha binary code
  - Only user-mode apps
  - No privileged instruction
  - No self-modifying code
  - No sys. Memory space reference
  - No user-written system services

Development Tools

- All development tools and utilities that ship with OpenVMS are being ported
- Developers can use existing procedures for developing, debugging, testing, and deploying their applications
- DECset tools shipped with V8.1
  - Language-Sensitive Editor/Source Code Analyzer (LSE/SCA)
  - Code Management System (CMS)
  - Module Management System (MMS)
  - Digital Test Manager
  - Performance and Coverage Analyzer (PCA) (ships H2/04)

OpenVMS Integrity Operating Environment Phase Rollout Plan

	Q4 2004	Q1 2005	Q2 2005
Foundation Operating Environment (FOE)	•OpenVMS Operating System w/ unlimited users •TCP/IP Services •DECnet-Plus End System •Decnet Phase IV •DECwindows Motif •Secure Web Server (SWS) •Java SDK (Classic VM) •XML Technology •SOAP Toolkit •Enterprise Directory •Kerberos •CDSA •SSL (Secure Socket Layer)	• SWS Tomcat • SWS PHP • Secure Web Browser • Java SDK (Hotspot) • Netbeans • TDC2 Data Collector	• Bridgeworks • COM
Enterprise Operating Environment (EOE)	•RMS Journaling •Volume Shadowing •DECram •Management Tools: Web Agents, Management Station Availability Mgr.	• Management Tools: •WEBM/CM •Enterprise Capacity Planner	
	• OpenVMS Clusters (available separately)	**Mission Critical Operating Environment (MCOE)**	• Reliable Transaction Router – Backend • OpenVMS Clusters

OpenVMS Integrity Layered Product Phase Rollout Plan

Q4/2004	Q1/2005	Q2/2005	Q3/2005
• Compilers: BASIC, Fortran, C, C++, COBOL, Pascal • DECset: CMS, MMS, LSE, DTM, PCA & SCA • Distributed File System • DECprint Supervisor • DQS • WEBES • DCE • Archive Backup System • Data Cartridge Server • Disk File Optimizer (DFO) • Hierarchical Storage Mgmt. • Media Robot Utility • RAID Software • Save Set Manager (SSM) • GKS • Phigs • FMS • BASEstar Family • Datatrieve • Device Access Software • OMNI API/MMS		•Reliable Transaction Router (RTR) •X.25	•ACMS (including TP Web & TP Desktop Connectors) •Advanced Server •DECforms
			Q4/2005
			•Soft Partitioning (ie. Galaxy/vPars) •Storage Library System (SLS)

Compiler Version Mapping Alpha vs. Itanium(r)

Compiler	***Alpha***	***Itanium***
Basic	V1.5	tbs
Bliss	V1.10-030	T1.1-049
Cobol	V2.8-1286	T2.8-1340
Fortran 77	--	na (Alpha only)
Fortran 90	V7.5	T8.0
C	V6.5	T7.0
C++	V6.5	tbs
Java	1.4.2-beta	1.4.2-beta1
Macro-32	V4.1-18	T1.0-77
Macro-64	V1.2	na (Alpha only)
Pascal	V5.8A	tbs

Example 1: Database vendor

- Application 1: written in C; no problems at all
- Application 2a: written in VAX assembler
  - Using HW knowledge in code
  - Hand coded kernel threads
  - Use calling standard knowledge
  - Hand coded save/restore of stack
    - VAX: ok
    - Alpha: using AMACRO, luckily it worked
    - Itanium(r): using IMACRO, very large effort
- Application 2b: written in C
  - Issue: uses functionality not yet implemented under UNIX Portability Initiative (fork, semaphore handling,...)

Example 2: Cadture

- 801 Fortran modules, about 2500 routines, 6 needed /nowarning
- Successful run after first link
- Found one programming error (status code)
- Compile time 10min total
- Dynamics:

– Alpha	Fortran noopt/opt	1:3
– Itanium	Fortran noopt/opt	1:5

Example 2: Cadture

- „VMS-bound“, virtual Fortran arrays, system services, IMG-services, X11 und Motif
- Conflicts: „Classical Fortran (Dispatch)“: Computed/Assigned Goto results in too many warnings: „Possible illegal jump into code block“.
- Program uses Floating, Integer, Character and Byte.
- To start only a text file is necessary, no floating conversion of old data

Code Changes necessary

Code that will require changes

- Alpha Macro 64 Assembler code.
  - This code must be rewritten in another language.
- Conditionalized code for Alpha or VAX systems.
  - This code must be revised to express an I64 condition.
- Code that uses OpenVMS system services that have dependencies on the Alpha architecture.
- Code with other dependencies on the Alpha architecture.
- Code that uses floating point data types.
- Code that uses threads, in particular, custom-written tasking or stack switching.
- Privileged code.

Differences between Calling Standards

- Registers 2 to 11 preserved on OpenVMS VAX
- Registers 2 to 15 preserved on OpenVMS Alpha
- Registers 4 through 7 preserved on OpenVMS I64
- OpenVMS I64 has more ‘volatile’ registers
- OpenVMS I64 returns values in R8/R9 instead of R0/R1
- R0 is readonly (RAZ) in the Itanium™ architecture
- Arguments in stacked registers in the Itanium architecture. R32-R39 for OpenVMS I64.

Register Map

	Alpha : I64	
return info	R0 = R8	return info
return info	R1 = R9	return info
preserved	R2 = R28	
preserved	R3 = R3	
preserved	R4 = R4	preserved
preserved	R5 = R5	preserved
preserved	R6 = R6	preserved
preserved	R7 = R7	
preserved	R8 = R26	
preserved	R9 = R27	
preserved	R10 = R10	
preserved	R11 = R11	
preserved	R12 = R30	
preserved	R13 = R31	
	R14 = R20	
	R15 = R21	

	Alpha : I64	
args	R16 = R14	volatile
args	R17 = R15	volatile
args	R18 = R16	volatile
args	R19 = R17	volatile
args	R20 = R18	volatile
args	R21 = R19	
	R22 = R22	
	R23 = R23	
	R24 = R24	
	(AI) R25 = R25 (AI)	
	(RA) R26 = tmp register	
	(PV) R27 = tmp register	
volatile	R28 = tmp register	
	(FP) R29 = R29	
	(SP) R30 = R12 (SP)	
	(RZ/sink) R31 = R0 (RZ)	

Alpha Macro 64 Code

- Rewrite in another language!

Conditionalized Code

- Old:
- #ifdef __vax
- ...
- #endif
- #ifdef __alpha
- ...
- #endif

- New:
- #ifdef __vax
- ...vax
- #endif
- #ifdef __alpha
- ...alpha
- #endif
- #ifdef __ia64
- ...ia64
- #endif

Conditionalized Code, cont'd.

- Better:
- #ifdef __vax
- ... 32bit path
- #else
- ... 64bit
- ... Path (alpha & I64)
- #endif

System Services & Alpha dependencies

- SYS$GOTO_UNWIND
- uses 32bit invocation context handle
- Change to:
  - SYS$GOTO_UNWIND_64
  - uses 64bit invocation context handle
  - Different set of library routines to return a 64bit invocation context handle
  - See *HP OpenVMS Calling Standard*

System Services & Alpha dependencies, cont'd…

- SYS$LKWSET & SYS$LKWSET_64
- SYS$ULWSET & SYS$ULWSET_64
- Replace with LIB$LOCK_IMAGE, LIB$UNLOCK_IMAGE
  - Only on Alpha and IA64!
  - No need for code that finds code, data and linkage sections and locks them
  - Addresses for these difficult to find on IA64

Alpha Architecture Dependancy

- Condition handling using SS$_HPARITH
  - Alpha: signaled for several arithmetic error conditions
  - I64: never signaled for arithmetic error conditions
  - I64: use SS$_FLTINV or SS$_FLTDIV instead
- Mechanism Array Data structure
  - Content is different
- Alpha Object/Image File Format
  - I64 uses a different formats
    - Object: Executable and Linkable Format (64bit version)
      - http://www.caldera.com/developers/gabi
    - Image & DST: DWARF V3
      - http://www.egercon.com/dwarf/dwarf3std.htm

Floating Point Data Type Usage

  - Float wait_time = 2.0;
  - Lib$wait (&wait_time);
- IA64: sends S_FLOATING to routine
- LIB$WAIT expects F_FLOATING -> FLTINV condition
- Better:
  - #ifdef __ia64
  - Int float_type = LIB$K_IEEE_S;
  - #else
  - Int float_type = LIB$K_VAX_F;
  - #endif
  - Float wait_time = 2.0;
  - Lib$wait (&wait_time,0,&float_type);

Code using threading

- All thread interfaces are supported on OpenVMS I64
- I64 code use much more stack space than Alpha code
  - may receive stack overflow as ACCVIO (V8.1) STKOVF (V8.2)
- I64: default stack size larger
- I64: may need to increase size if application requests specific stack size

Unaligned Data

- Unaligned data seriously degrades performance
- No difference for OpenVMS Alpha and I64

Reliance on Alpha Calling Standard

- OpenVMS I64 calling standard based on Intel calling standard with modification
- Different from Alpha
- Differences include:
  - Register numbers are different
  - No frame pointer (FP)
  - Multiple stacks
  - Only 4 registers preserved across calls

Privileged Code

- See SYS$LKWSET example
- Terminal drivers
  - Interface changed from JSB to call based interface
  - (JSB uses registers to pass arguments)

OpenVMS Infrastructure Changes

- IPF and Alpha only
- Privileged Images only (link against system [/SYSEXE])
- Dependancy on following subsystems
  - SYS$K_VERSION_IO
  - SYS$K_VERSION_MEMORY_MANAGEMENT
  - SYS$K_VERSION_CLUSTERS_LOCKMGR
  - SYS$K_VERSION_FILES_VOLUMES
  - SYS$K_VERSION_CPU
  - SYS$K_VERSION_MULTI_PROCESSING
- Increase of version number
- How to find out dependancy:
  - $ ANAL/IMAGE your_image.exe/OUT=image.txt
  - $ SEARCH image.txt "SYS$K"

Kernel Process Extensions

- Usage of Kernel Processes now allowed in outer modes and all IPLs
- **Alpha and IPF only change**
- Code with private threading packages can now make use of Kernel Processes
- Some changes to the KPB$ data structure were necessary
- No source changes necessary for existing Alpha code
- Recompile and relink required (image has "SYS$K" matches)

CPU Name Space

- OpenVMS current architectural limit of maximum CPU Id of 31
- Increase this limit to
  - maximum of 64 for Alpha
  - Maximum of 1024 for IPF
- V8.2 release will not support any systems (IPF or Alpha) with CPU Ids larger than 31
- Some kernel data structures maintain 32-bit CPU Id masks
- Increase the space allocated for these CPU Id masks
- Existing longword symbols for CPU masks will continue to be maintained
- With the exception of rebuilding, there should be no impact to privileged images and drivers.
- Recompile and relink required (image has "SYS$K" matches)

64Bit Logical Block Number (LBN)

- OpenVMS today supports LBNs of only 31 bits
- This limits a disk volume to 1 terabyte
- Various LBN fields in data structures are promoted from longwords to quadwords
- Longword symbols will continue to be maintained
- This will allow for future operating system support of volumes larger than 1 terabyte
- No plans to support volumes larger than 1 terabyte for V8.2
- Recompile and relink required (image has "SYS$K" matches)

Forking to Dynamic Spinlock

- To scale OpenVMS on large SMP systems, some areas in the OS use dynamic spinlocks (=/ static spinlocks, limited)
- The fork dispatcher will now use dynamic spinlocks (V8.2)
- Need to extend the size of the FKB$ data structure and adding a FKB$L_SPINLOCK field. This spinlock field will only be referenced if FKB$B_FLCK contains the value SPL$C_DYNAMIC.
- Recompile and relink is required (image has "SYS$K" matches)
- A very small subset of applications may need to make code changes if they allocate FKB structures using a hard coded value of the old structure size of 32 bytes (use FKB$C_LENGTH as size of a FKB structure)
- Also, if copying FKB structures need to take the new field into account

Fast Device Create/Delete

- Device list (UCBs) associated with a controller (DDB) is a zero terminated singularly linked list
- When creating and deleting a UCB, these lists must be walked until the appropriate location is found in order to add or remove a UCB from the list
- Will now be a doubly linked list (still zero terminated) to avoid the sequential search when creating and deleting a UCB
- This requires the addition of some new cells in the UCB and DDB.
- Recompile and relink (image has "SYS$K" matches)
- Code which modifies the list of UCBs associated with a DDB should be updated to utilize VMS provided routines
  - IOC_STD$CLONE_UCB, IOC_STD$COPY_UCB, IOC_STD$LINK_UCB, IOC_STD$DELETE_UCB
- Code walking the list of UCBs still works correctly without any changes

UCB Field Promotions

- The UCB$W_UNIT field promoted to a longword
- Support more than 64k unit numbers for a device
- The UCB$W_UNIT field will still be maintained
- Recompile and relink (image "SYS$K" matches)

Terminal Driver Updates

- Fields in the terminal driver's UCB extension will be promoted from bytes and words to longwords
- Existing field names will continue to overlay the promoted fields
- Recompile and relink (image has "SYS$K" matches)

Relationship of VMS Facts of Life To Hardware

- VAX (first platform for VMS)
  - Modes, AST triggering, IPLs, Software Interrupts Implemented in "Hardware"
- Alpha
  - Modes largely hardware
  - IPL 16-31 largely hardware
  - IPL 0-15, AST trigger, Software Interrupts, glue between hardware and VAXiness "PALcode" firmware
- Itanium
  - Modes largely hardware
  - IPL 16-31 largely hardware
  - IPL 0-15, AST trigger, Software Interrupts, glue between hardware and VAXiness in "SWIS" OS software

SWIS – SoftWare Interrupt Services

- Implements IPLs (including 16-31)
  - Manages Itanium CPU's interrupt resources
  - Implements Software Interrupts
- Implements AST triggering
  - Manages Itanium CPU's mode changing mechanisms
- Implements other mechanisms that were supplied by PAL on Alpha
  - Interrupt/Exception Dispatching
  - Swap Process Context
  - "Internal Processor Registers"
    - SP for non-current modes
    - Request/Enable ASTs, Request Software Interrupts, etc

Porting OpenVMS applications VAX to Alpha to Itanium

Application Migration | QA / Certification / Field Test / Release

VAX to Alpha

- 32 Bit to 64 Bit
- two different OS code bases
- not all layered products ported
- Majority of time spent in porting the application and getting it working.

Application Migration | QA / Certification / Field Test / Release

Alpha to Itanium

- 64bit to 64bit
- one common OS code base
- all layered products ported
- QA time is not architecture specific and remains the same

What can I do today?

- Ensure you are building and running OpenVMS V7.2 or higher (V7.3 or V7.3-1 is ideal).
- Make sure your build processes and regression tests are clean and complete
- Examine your code for known differences and architecture dependencies
- Create a detailed inventory of all layered products (including compilers, OpenVMS layered products, 3rd party products, etc.) for your development, regression test, and production systems. Include version numbers.
- Move to latest version of compilers and use latest standards (i.e. Fortran 95 vs. Fortran 77, Ada 95 vs. Ada 83)
- Recode Alpha Macro or PL/1 to another language, such as C

Cross-section of leading OpenVMS ISVs committed to HP Integrity servers

hp invent

TD

bea

Deutsche Börse Systems

ORACLE

Ada Core TECHNOLOGIES, INC

InterSystems CACHÉ Post-Relational Database

bmcsoftware

PROCESS SOFTWARE

JAMS

NETRICS

IDX

Legacy

Martin Group The One Source

CERNER

SIEMENS

BCD modelling

logicaCMG

bChat

spiritsoft go beyond jms

TIBCO The Power of Now

ca Computer Associates

CINCOM

ISE INC

Schlumberger WORLDWIDE IT PARTNER

Wall Street Systems

BROOKS AUTOMATION

NORTHROP GRUMMAN

COMPUWARE

sct

ERICOM SOFTWARE

SYNERGEX™ Extending Enterprise Applications

ATTUNITY

itheon

ROLFE & NOLAN

SANCHEZ

ROSS SYSTEMS

HEROIX Find it. Fix it. Forget it.

POINT SECURE

WRQ

ACCELATION SOFTWARE "Accelerating Innovation"

Vista Control Systems

iMSURE NETWORK, INC.

LEGATO

OpenVMS I64 V8.1 rx2600 config

hp rx2600 Recommended system configuration detail

Part #	Qty	Description
A6870B	1	HP rx2600 1.3GHz CPU server Solution
A9872A	0-1	hp rx2600 1.3-GHz CPU w/3-MB cache
A9910A	1	4GB DDR memory quad
A6829A	1	Dual-channel Ultra160 SCSI adapter card
A9919A	1	Read-only Optical Drive (DVD+R/CD+R)
A6825A	1	Single Port GigE-TX (gigabit copper) ***(OR)***
A6847A	1	Single Port GigE-SX (gigabit fiber)
AB232A	1	PCI-X 1-port FCA2404 2GB FC ***(OR)***
A6826A	*1*	*dualport 2GB/1GB FC Universal PCI-X*
A9896A	1-4	36GB 15k rpm ultra3 scsi disk drive

Note: See further configuration support detail in notes section

OpenVMS I64 V8.1 rx4640 config

hp rx4640 Recommended system configuration detail

Part #	Qty	Description
A6961A	1	HP rx4640 1.3GHz CPU server Solution
A7159A	0-3	rx4640 1.3GHz Itanium 2 CPU w/ 3MB cache
A6967A	1-4	1GB memory quad
A6829A	1	Dual-channel Ultra160 SCSI adapter card
A7163A	1	Read-only Optical Drive (DVD+R/CD+R)
A6825A	1	Single Port GigE-TX adapter card
AB232A	1	PCI-X 1-port FCA2404 2GB FC ***(OR)***
A6826A	*1*	*dualport 2GB/1GB FC Universal PCI-X*
A9896A	1-4	36GB 15k Hot Plug Ultra320 SCSI

Note: See further configuration support detail in notes section

oJOGIN

i n v e n t