
Software
INTERNATIONAL
Concepts
Worldwide
Managed Services
for OpenVMS and
Rdb
SCI ON OPENVMS X86
OUR STORY
2024
Norman Lastovica
lastovica@sciinc.com
www.sciinc.com

My Background

- Pre-Digital
  - AMD 2901 bit-slice microcode
  - 68000 assembler
- Digital Equipment
  - Systems and applications
  - VAX to Alpha Porting
- Oracle
  - KODA project leader
  - Alpha to I64 Porting
- SCI
  - Performance, Systems, Application
  - VAX, Alpha, I64 to x86 Porting

History

First non-VSI site

Cross Compilation

Native Compilation

Code Changes

Incomplete History of VMS on x86

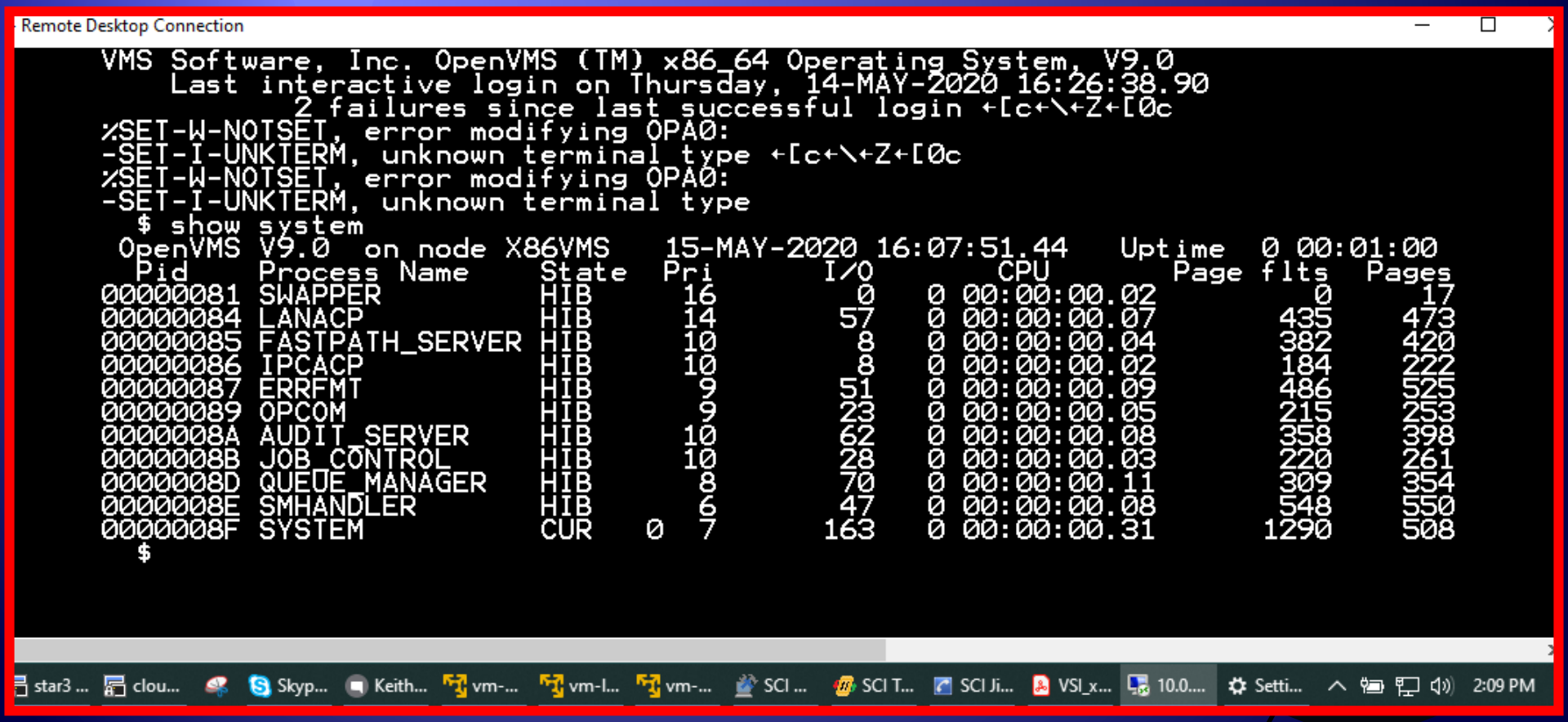
- VAX emulator
- Alpha emulator
- Native rumors
- VSI
- Stronger rumors
- Initial announcements
- Cross build environment
- Invitation
- Install
- More native compilers / building / testing

Are You Ready?

- SCI's long close relationship with VSI
  - Former DEC/CPQ/HPE/Oracle engineers at SCI
- 5-MAY-2020
  "*Are you ready for an experiment?*"
- Flattered, honored, and ready

15-MAY-2020

- SCI's First Boot
- "VMS is VMS"



Porting?
SCI has been there & done that

- PDP11 → VAX
- VAX → Alpha
- VAX, Alpha → I64
- VAX, Alpha, I64 → x86
- Freeware
- Customer/commercial Applications

Expectations

One more time…

For most applications…

Recompile & relink will do

Environment, Compilers, Tools

- OpenVMS is OpenVMS & DCL is DCL
- Clustering, MSCP, DECnet, TCPIP, shadowing, queues, PTHREAD, MESSAGE, SET COMMAND, RUNOFF, devices, mail, librarian, HELP, SMP, ACCOUNTING, security, ACLs, FDL, linker, EDT, TPU, SYSGEN, SDA, etc., etc.
- Mostly compiler same front-end Alpha/I64/x86
- Documentation — VMS Software, Inc.

Floating Point : VAX vs IEEE

- Same situation as I64
- Processing VAX vs IEEE floating format
  - Hardware vs software emulation/conversion
  - Performance
- Slight differences in rounding
  - Floating point is an approximation
  - Results will differ in edge cases VAX vs IEEE
- Differences in magnitude and precision

Floating Point

- Best bet, if you've not already, migrate *today* to IEEE where possible on Alpha, I64, x86
  - It isn't 1986 any more
  - Rdb/DBMS?
  - On disk?
  - On network?
- Ensure all modules use same /FLOAT…
- Default floating point format used by LIB$WAIT is F_FLOAT, which does not match the default floating point format used on x86 (S_FLOAT)

IEEE Floating Point Division in Action

```
declare decimal (7,3) pp
declare integer pn

pn = 116
pp = pn/10
print "pp1 ="; pp

pn = 331
pp = pn/10
print "pp2 =";pp
```

```
$ RUN X         ! On VAX
pp1 = 11.6
pp2 = 33.099

$ RUN X         ! On Alpha
pp1 = 11.6
pp2 = 33.099

$ RUN X         ! On Alpha
pp1 = 11.599 ! /REAL=TFLOAT
pp2 = 33.1

$ RUN X         ! On x86 & IPF
pp1 = 11.599 ! /REAL=anything
pp2 = 33.1
```

Data Alignment

- VAX didn't matter much
- Alpha mattered more
- I64 matters much more
- x86 doesn't matter much

- No functionality impact

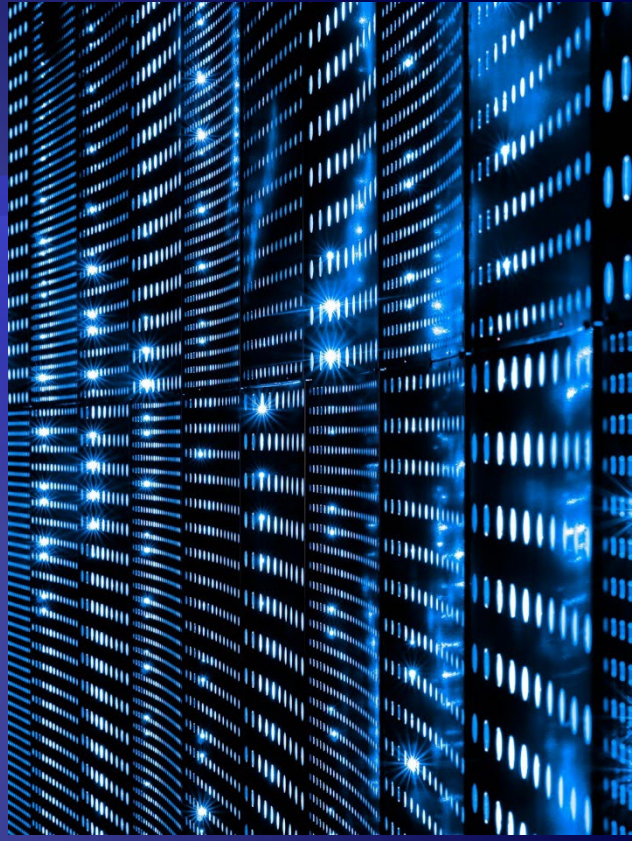
- In all cases, naturally aligned considered 'better'
  - Not worth effort to alter working applications having adequate performance

On-Disk

- Same-same
- RMS files unchanged
- Disk structure unchanged

TCPIP V6

- Bulk of stack same
  - Some pieces refreshed
- New SSH/SCP/SFTP
  - Separate product
  - OpenSSH key format
  - 'Largely' compatible



Processor

- No impact / visibility to 99% of developers / porting
- VAX = CISC architecture
- Alpha = RISC architecture
- I64 = VLIW/EPIC architecture
- x86 = CISC architecture

Calling Standard

- Data types, registers, argument passing, flow of control, stack unwinding, exception handling
- Family familiar
  - Differences VAX, Alpha, I64, x86
    - Register(s) save/restore
    - Exception and mechanism array and access
    - Parameter passing
- Anyone who writes code on OpenVMS
  - VSI OpenVMS Calling Standard March 2024
  - VSI OpenVMS Calling Standard — VMS Software, Inc.

SCI

SCI Internal x86 OpenVMS Systems

- Hardware / environment constraints
  - OpenVMS running on
  - Oracle Virtual Box running on
  - VMware running on
  - E5-2667 v3 @ 3.20GHz
- Functional; not performant solution
- Humorous when VSI suggests OpenVMS bug due to our configuration

Multi-Architecture Cluster

- Unsupported Alpha/VAX V6.2 in cluster with V9.2
  - V7.3 and higher required

View of Cluster from system ID 1031 node: STAR4 28-MAR-2024 09:39:09

SYSTEMS			MEMBERS		
NODE	HW_TYPE	SOFTWARE	VOTES	STATUS	TRANSITION_TIME
STAR4	HP rx2620 (1.60GHz/3.0MB)	VMS V8.4-2L1	1	MEMBER	8-MAR-2024 07:46
YDOTY	innotek GmbH VirtualBox	VMS V9.2-2	0	MEMBER	27-MAR-2024 21:27
STAR3	HP rx2620 (1.60GHz/3.0MB)	VMS V8.4-2L3	1	MEMBER	8-MAR-2024 08:32
OBIWAN	AlphaServer ES40 6/667	VMS V7.3-2	1	MEMBER	15-MAR-2024 16:19

CLUSTER						
CL_EXP	CL_QUORUM	CL_VOTES	QF_VOTE	CL_MEMBERS	FORMED	LAST_TRANSITION
3	2	3	NO	4	17-NOV-2022 16:03	27-MAR-2024 21:27

SCI Build Environment

- Components
  - Multiple languages
  - ~2000 modules
- Platforms & versions
  - VAX, Alpha, I64, x86
  - V6.2, V7.3, V7.3-2, V8.2, V8.4, V9.2
- Batch queue per platform
  - VAX, Alpha, I64, x86-cross, x86-native

x86-cross & x86-native

- Initially x86 code built on i64 with cross tools
- Over time migrate components one-by-one to native building
- A hand full of stragglers due to
  - Missing tools (e.g. SDL)
  - Overly complicated open source build procedures
  - Minor issues

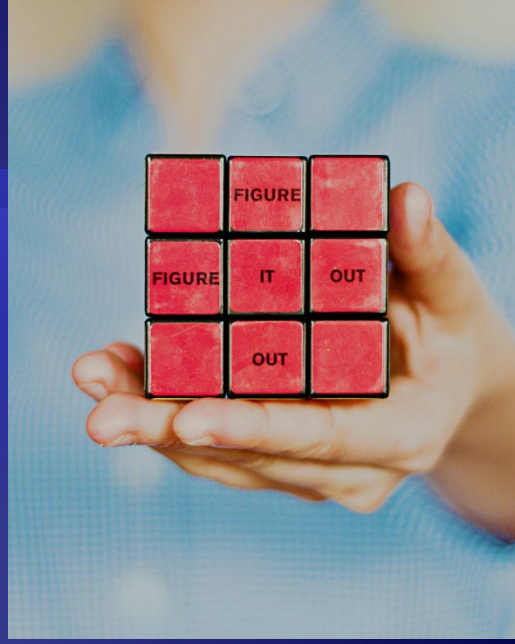
Code Changes

- Code relying on hardware model must change
- Most time spent within
  - DCL command procedures and HLL learn about x86
    - `F$GETSYI("ARCH_NAME")`
    - `F$GETSYI("ARCH_TYPE")`
    - `F$GETSYI("HW_MODEL")`
    - `#if/#ifdef/#ifndef`
    - `%BLISS(BLISS…)`
- Small impact maybe
  - Builtins
  - Errant data structure accesses

Generally Same Behavior

- Mostly Alpha and I64 behavior on x86
- If only we could have seen the future
  - `#ifdef alpha vs #ifndef vax`
- DCL e.g.
  - `ARCH := 'F$GETSYI ("ARCH_NAME")`
  - `PLAT := 'F$EXTRACT (0, 1, F$GETSYI ("ARCH_NAME"))`
  - `B$VAX = ARCH .EQS. "VAX"`
  - `B$ALPHA = ARCH .EQS. "ALPHA"`
  - `B$IA64 = ARCH .EQS. "IA64"`
  - `B$X86_64 = ARCH .EQS. "X86_64"`

Areas Problems Found



- Not bug-for-bug compatible
  - Long latent flaws
  - E.g. uninitialized variables/heap
  - Stack access out of scope
- Field testing compiler & OpenVMS flaws
  - Close cooperation with VSI engineering
- Increase quotas liberally (e.g. working set, PGFLQUO, BYTLM)

Less Usual Cases

- Signaled SS$_HPARITH, SS$_FLTINV, SS$_FLTDIV
- AMACRO, IMACRO, MACRO32
- Code with knowledge about calling standards, image format, and debug format must change
- If you are exposed in these areas, talk with me after the session

RUNOFF Flaw

- Migrated GREP to native build
- RUNOFF failed ~50% of the time
  - Long-standing bug in XPORT; stack-local misuse
  - Corrected VMS922X_UPDATE V1.0

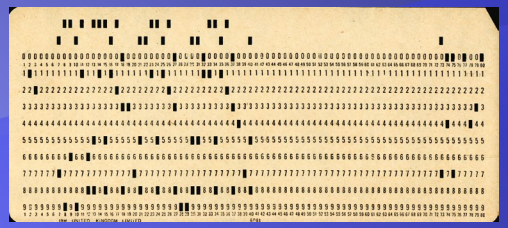
```
Ydoty VTA5:> set def DISK$BUILD:[SCI_BUILD.ENV.GREP]
Ydoty VTA5:> runoff grep.rnh
%XPO-E-TEXT, error parsing '<ƒÿÿ€:[SCI_BUILD.ENV.GREP]GREP.RNH;1'
-XPO-E-NOMSG, Message number 0020A0B2
-XPO-E-NOMSG, Message number 0020A2E2
%XPO-E-NOMSG, Message number 0020A1F2
Ydoty VTA5:>
```

Editor Written in 1962

- "*probably the least portable code*" – VSI engineer
- Downloaded freeware TECOC; made minor adjustments to get it to build; not done much testing beyond this and don't intend to

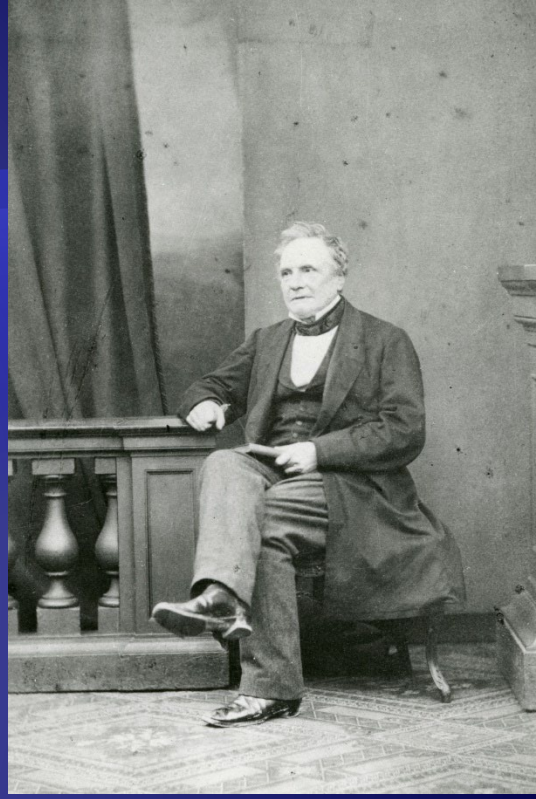
```
X86vms TTA0:> sho sys/nopr
OpenVMS V9.0-B  on node X86VMS    9-JUL-2020 18:17:27.35   Uptime  0 00:10:27
X86vms TTA0:> teco
*erx.x$y$$
*ht$$
$ version = f$getsyi("version") - "." - "-"
$ arch = f$getsyi("arch_name")
$ if  version .ges. "V84"
$ then  set rms/index/sequ/rel/blo=255/buf=10/net=127/ext=512
$ else  set rms/index/sequ/rel/blo=127/buf=10/net=127/ext=512
$ endif
```

Legacy Compatibility?



```
$ HELP JOB
…
      Identifies the beginning of a batch job submitted through a card
      reader. Each batch job submitted through the system card reader
      must be preceded by a JOB card.

      JOB cannot be abbreviated.

      Format

        JOB  user-name

$ HELP SET CARD_READER
…
        Defines the default translation mode for cards read from a card
        reader. All subsequent input read from the specified card reader
        is converted using the specified mode.

        Format

          SET CARD_READER  device-name[:]
```

Aside



- Charles Babbage proposed use of "Number Cards", "pierced with certain holes and standing opposite levers connected with a set of figure wheels ... advanced they push in those levers opposite to which there are no holes on the cards and thus transfer that number together with its sign" in description of Calculating Engine's Store

- No evidence that he built a practical example

Customer Application 1

- Trading system front end / message passing system
- "Future proof" port
  - No immediate requirement to deploy on x86
  - Be able to demonstrate
- 99% PASCAL
  - ~100 modules / ~100,000 lines of code
  - Modified several build procedures / header files (alpha vs i64 vs x86)
  - Compile – link – run
- Interface with CA (aka VSI aka Polycenter) Watchdog
  - Created stub module to resolve missing symbols at link

Customer Application 2

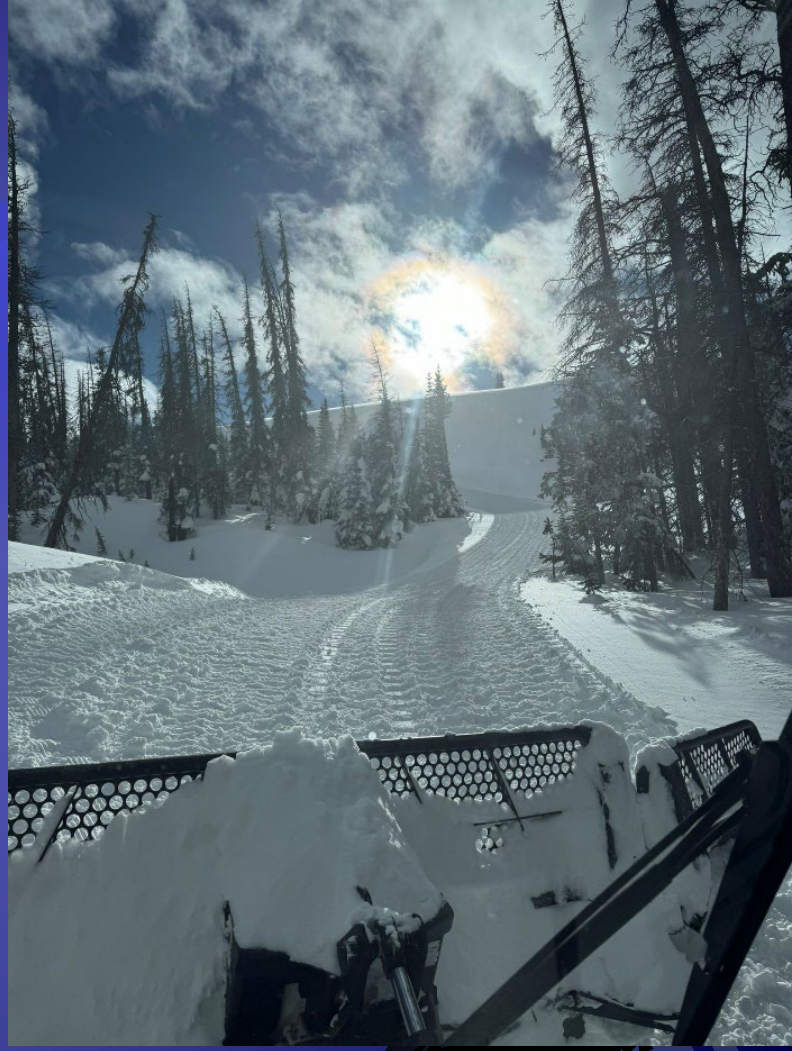
- International monetary exchange system
- File change detection and reporting facility
  - Assist identifying malicious actors modified, added, or deleted files
- BASIC & C
  - ~Dozen modules
  - OpenSSL
  - RMS file access
- Minor build command procedure and code change due migrating to SSL3

Customer Application 2 Continued

- Built with first field test BASIC compiler
- Application compiles and links
  - Fails to run correctly
- Flaw in field test BASIC
  - Frontend set external routine references to WEAK in error
  - Potentially BLISS compiler bug
- Worked around by explicitly linking with SYS$LIBRARY:SECURESHR/SHARE

Today at SCI

- OpenVMS V9.2-2
- A few missing pieces limit SCI native build edge-to-edge
- DBMS & Rdb?
  - LibMariaDB / MariaDB?
- Other third-party products?



Performance

- Bit early to tell
- Require robust host
  - Our VM-within-VM not ideal
- More optimizations from compilers?
  - /ARCHITECTURE currently ignored for x86-64 systems … expect to provide options to pass along to LLVM …
- Measure and model to gain experience
  - CPU, network, disk for various applications

Where You Likely Have To Do Some Work

- Build procedures
- Inner (ie, non-user) modes
- Linking /SYSEXE
- Knowledge of call stack formats, exception frames, PTE, PFN, PC, FP, AP
- Strict floating point behavior requirements

Where You Likely Have To Do Some Work (cont'd)

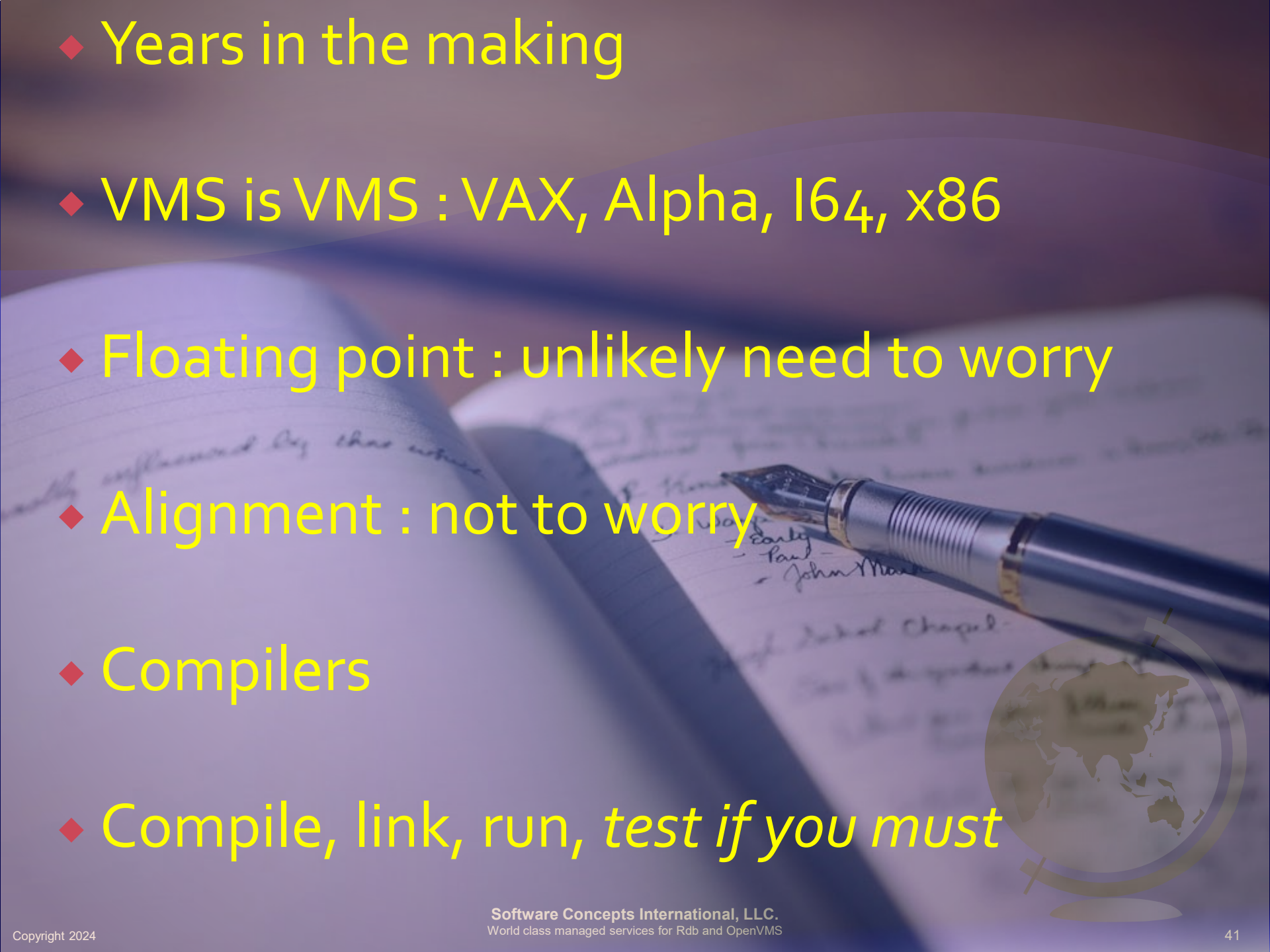
- MACRO
  - Consider moving to HLL
  - If porting directly from VAX
    - Transfer vectors?
- ASM within C
  - Avoid IPF & x86 assembler – write native C

See Also

- Porting "real" applications to OpenVMS I64, Guy Peleg, OpenVMS Systems Division Hewlett-Packard Company

- Rdb On IA64: IEEE Floating Point, John Howard & Norm Lastovica, Oracle Rdb Engineering

- Years in the making
- VMS is VMS : VAX, Alpha, I64, x86
- Floating point : unlikely need to worry
- Alignment : not to worry
- Compilers
- Compile, link, run, *test if you must*



- Camiel Vanderhoeven
- Christian Moser
- Clair Grant
- Doug Gordon
- Greg Jordan
- Hartmut Becker
- Homi Faris
- Ian Smith
- John Gemignani
- John Reagan
- Marcin Zablocki
- Mike Zaharee
- Richard Bishop
- Rob Brooks
- Taylor Newill


Questions?
Thoughts?
Software
INTERNATIONAL
Concepts
Worldwide
Managed Services for
OpenVMS and Rdb

About Software Concepts International

Managing OpenVMS systems and databases requiring the highest levels of performance and availability – worldwide

Software Concepts International

- Located in Nashua, NH (USA)
  - 30 years in business supporting OpenVMS!
- International reputation as leading provider of
  - Managed services for OpenVMS & databases
  - OpenVMS performance and consulting services
- Proven global track record
  - Actively managing 100s of systems and databases
  - At many sites 24x365 since 1995

Software Concepts International

- System and database performance consulting
- VAX/Alpha emulation – CHARON reseller
- Migration consulting
  - Specializing in minimal downtime migrations
- Oracle's worldwide provider of CODASYL DBMS training