



Transitioning Custom Applications to HP-UX 11i on Integrity Servers



Peter Swärd, Software Engineer
BCS Transition Engineering and Consulting
(TEC)

© 2004 Hewlett-Packard Development Company, L.P.
The information contained herein is subject to change without notice



Presentation Outline



- HP-UX: Migration to Integrity HP-UX 11i
 - Migration options
 - Doing the transition
 - Compiler optimization

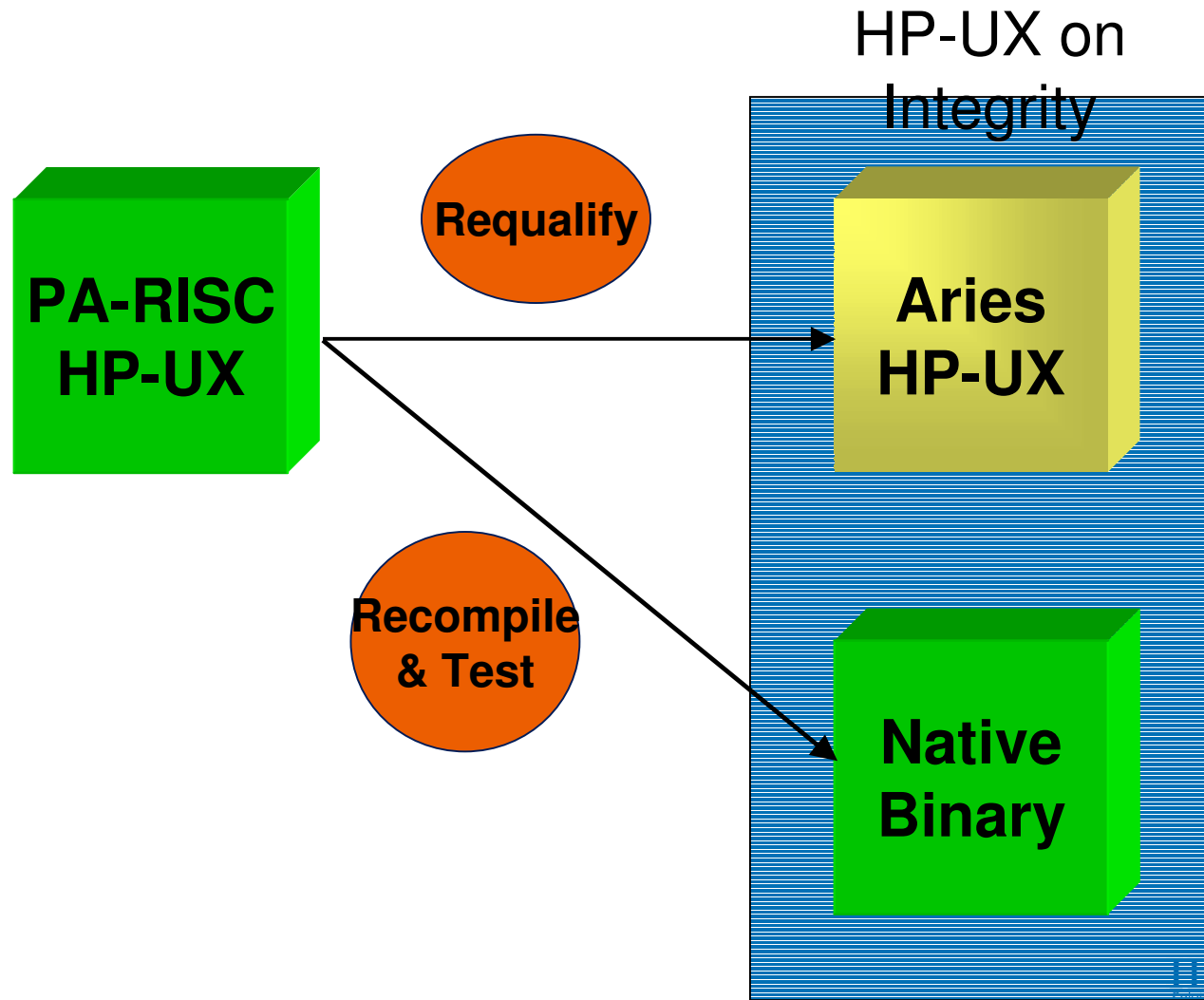
- Tru64 UNIX: Migration to Integrity HP-UX 11i
 - Standards
 - Compilers
 - C code
 - Transition tools



Purpose

- Understand the migration issues and resources available to assist in the transition of application source code to HP-UX for IPF (HP Integrity Servers)
 - Source is HP-UX 11i v1, HP-UX 10.20, HP-UX 11.0 or Tru64 UNIX V4.0D or higher
 - Target is Integrity HP-UX 11i v2

HP-UX: Migration to HP-UX 11i v2 path



Aries - Dynamic Binary Translator



The ARIES translator is available for legacy home grown applications

- ARIES is a binary translator that automatically executes well-behaved PA-RISC applications on Integrity servers running HP-UX 11i
- Designed for instances where the source code is not available, ARIES runs the PA-RISC code in emulation mode
- No user intervention required
- Supports both 32-bit and 64-bit applications
- Highly Reliable
- Proven reliability on many commercial applications
- Portions of HP-UX commands and utilities set are using it

<http://devresource.hp.com/drc/STK/docs/refs/Aries.jsp>

Aries Technology - Some Basics



- When would I use Aries?
 - I have most of my code recompiled for Integrity, but I can't get my partners to recompile an PA-RISC binary that I need to run on Integrity
 - I want to run an PA-RISC binary on Integrity, but I don't have the source
 - I have an internal completer application that is not performance critical

- When would I not use Aries?
 - For a performance-critical application

- Performance varies
 - Expect integer/floating point code to run 3X/5X slower respectively than recompiled
 - For many applications, the code runs faster than on the old PA-RISC system

- Over 100 significant PA-RISC application binaries have run directly on Integrity using Aries technology

Aries Technology - Cautions



– Cautions:

- No privileged PA-RISC hardware instructions
- No device drivers or loadable kernel modules
- Does not support mixed Integrity and PA-RISC shared libraries
- No support for ttrace() ptrace() and profil() system calls
- Does not distinguish fork() and vfork()
- Does not support SigNaN and QNaN identity
- Consumes a small amount of virtual address space
- Processor information is emulated to be PA-RISC

Move to 64-bit?



- LP64 vs. ILP32 (both on PA-RISC and Integrity)
 - Both are fully supported
 - Move to 64 bits is determined by application requirements

- Reasons to use 64-bit architecture
 - Huge data model (>4Gb datasets)
 - Relieve I/O bound models by enlarging I/O buffering
 - Any memory-constrained applications

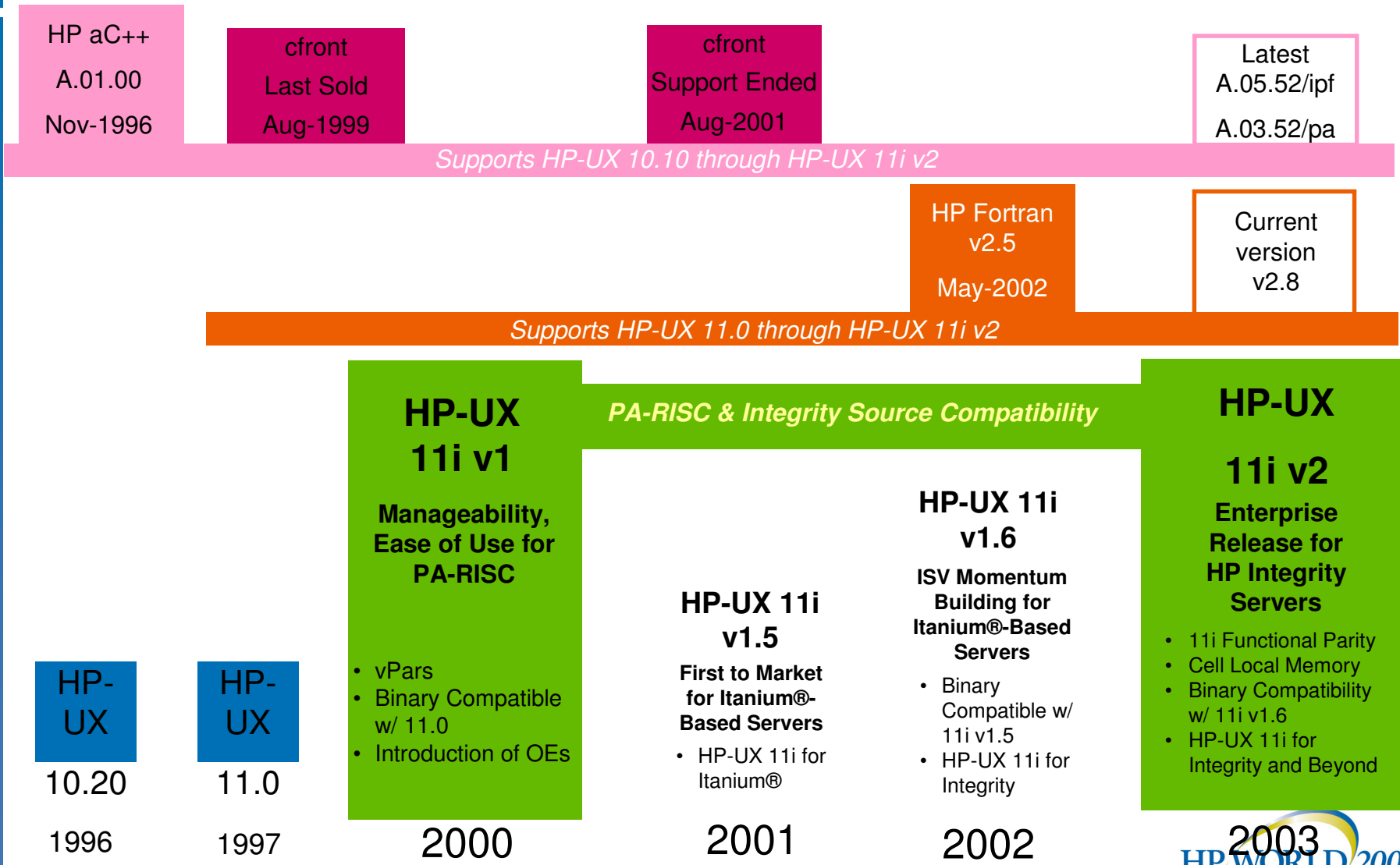
- Reasons to stay on 32-bit architecture
 - Larger program files in 64-bit architecture
 - Data space can double, depending on application
 - Risks in the code migration (alignment, casting, etc.)

Source Code Compatibility



- The C++ compilers for PA-RISC and Integrity share common front-end source code
- HP-UX header files and system APIs are based on shared source code for PA-RISC and Integrity
- HP-UX supports 32-bit applications on Integrity, so applications do not need to port to 64-bit
- All HP-UX compilers and libraries must undergo compatibility testing
- Incompatibilities subject to a thorough review process, allowed only when necessary, and documented as compatibility exceptions
- The Software Transition Kit (STK) can be used to scan an application's source code to look for portability problems
- The HP-UX operating system, its commands and libraries, and dozens of ISV applications (10s of millions of lines of source code), have been compiled with the Integrity compilers
 - Incompatibilities not already identified as exceptions have been treated as defects and fixed

HP-UX Releases and Compatibility: HP-UX & Compilers Release History



HP-UX 11i v1
Manageability, Ease of Use for PA-RISC

- vPars
- Binary Compatible w/ 11.0
- Introduction of OEs

PA-RISC & Integrity Source Compatibility

HP-UX 11i v1.5
First to Market for Itanium®-Based Servers

- HP-UX 11i for Itanium®

HP-UX 11i v1.6
ISV Momentum Building for Itanium®-Based Servers

- Binary Compatible w/ 11i v1.5
- HP-UX 11i for Integrity

HP-UX 11i v2
Enterprise Release for HP Integrity Servers

- 11i Functional Parity
- Cell Local Memory
- Binary Compatibility w/ 11i v1.6
- HP-UX 11i for Integrity and Beyond



Source Compatibility Exceptions



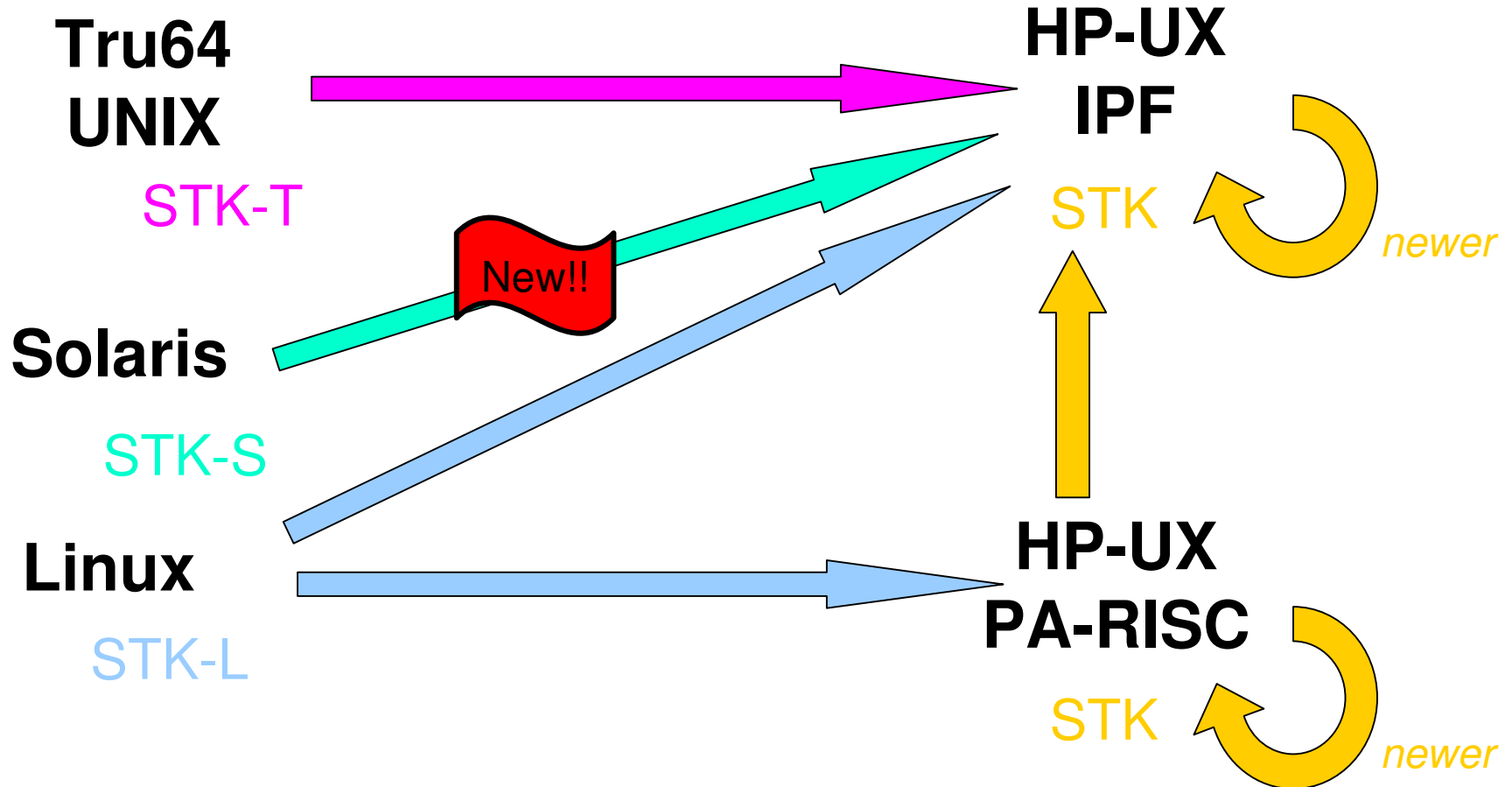
- K&R C is no longer supported
- Convex parallelization pragmas and library functions are no longer supported
 - OpenMP is supported on both platforms as a replacement
- Architecture-specific code, options, and pragmas must be modified for Integrity
 - These include PA-RISC assembly code, inline assembly operations, options and pragmas used for tuning code for the PA-RISC architecture and runtime, and calls to any system APIs that are supported only on PA-RISC
- The `#pragma HP_ALIGN` is no longer supported
 - `#pragma pack`, which is common to Gnu and Sun compilers, should be used instead
- Floating-point operations may result in slightly different results
 - Integrity will usually give greater accuracy
 - Applications may observe differences in the treatment of NaNs, denorms, infinities, signed zeroes, exceptions, and flush-to-zero
- The use of any third-party library is subject to the availability and support of that library on Integrity and HP-UX 11.23
 - Integrity code and PA-RISC code cannot be mixed within a single program

What is the STK?



- A collection of **documentation** and **tools** to help developers get their software ported/transitioned to the latest version of HP-UX
- **It helps developers with questions such as:**
 - How can I transition my software to Integrity, the new HP architecture?
 - Do I want a 32-bit or 64-bit version of my software?
 - Can I qualify my existing software or do I need to rebuild (create a new HP-UX executable on my destination platform)?
 - What new features does the latest HP-UX release offer, and how can I take advantage of them?

HP-UX Software Transition Kits



HP-UX Software Transition Kits



- File Scanners

- Assist developers with the identification and resolution of compatibility issues between source and target platforms
- Filescanner modes
 - *scansummary*
 - *scandetail*
- *scanwizard*, a wizard for filescanner options

<http://devresource.hp.com/STK/>

- Developer's Documentation

- Transition Documents
 - Transitioning source code
 - Understanding 64-bit
 - Porting Guides
- Technical Reference Material
 - 32/64 bit
 - Compiler related
 - Portability
 - Run-time architecture (PA-RISC & Integrity)
 - Threads and MP
 - HP-UX man pages

scanwizard



```
sward's X desktop (rackem.zk3.dec.com:1)
atterm
Window Edit Options Help
The configuration file (/usr/users/sward/.scanwizardrc) contains the following options:

SOURCE OS:                DESTINATION OS:
  o Tru64 UNIX              o HP-UX
VERBOSE MODE:             REPORT TYPE:
  o OFF                     o Summary

OUTPUT FORMAT:           TITLE OF THE OUTPUT REPORT:
  o html                   o Summary Report

OUTPUT FILE NAME:
  o "stk_summary.html"

SORTING ORDER:
  o Sort by number of instances

PROBLEM SEVERITY:
  o Critical

PROBLEM TYPE(S):
  o Changed
  o Enhancement
  o Obsolete

PROBLEM CLASS(ES):
  o Capacity, scaling, and limits
  o Date
  o Header file
  o I18N and L10N
  o I/O
  o Kernel
  o Migration Environment
  o Networking
  o Security
  o Signals
  o Threads
  o Unclassified

PROBLEM IDENTIFIER(S):
  o C/C++ function or macro
```



STK File Scanner

- Scan C, C++, Fortran, scripts and Makefiles

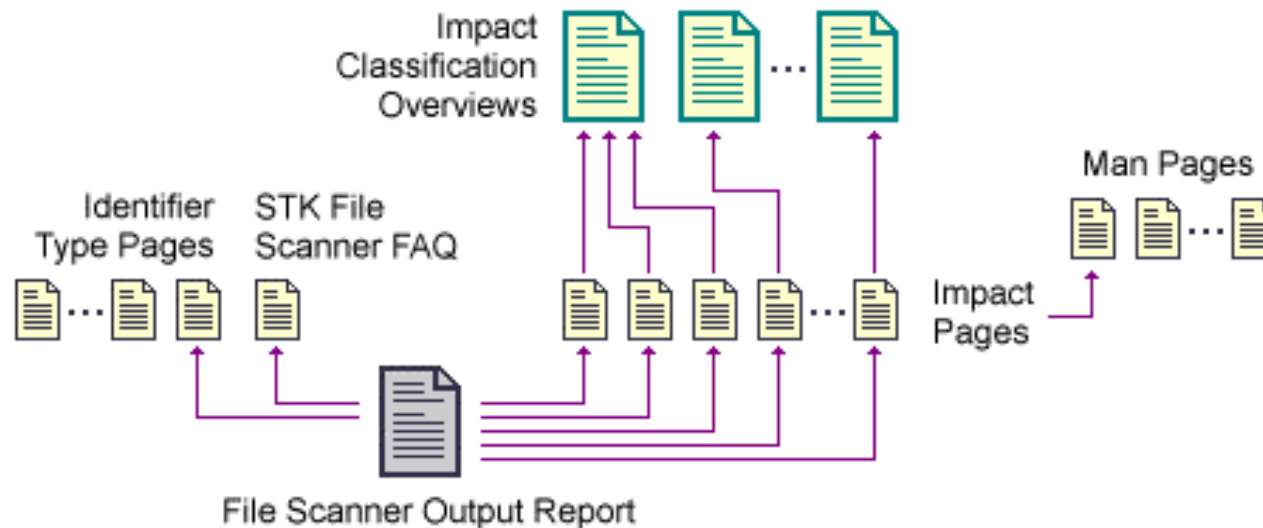
- Scan for incompatibilities in:
 - functions
 - commands
 - macros
 - structures and structure members
 - header files
 - language keywords
 - libraries
 - variables

- Output formats
 - html (default)
 - text

STK File Scanner Report

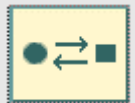



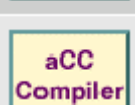
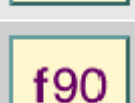
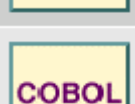
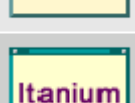


The file scanners generate reports that provide links to detailed impact pages describing each impact and its solution. The impact pages in turn link to impact classification overviews and man reference pages.



STK Impact Classifications

Describe the API impacts that can occur in transitioning source code

-  **32 - 64 bit interoperability impacts summary**
-  **64-bit API impacts summary**
-  Binary compatibility impacts summary
-  Date impacts summary
-  HP aC++ and C compiler impacts summary
-  HP Fortran compiler impacts summary
-  HP Micro Focus COBOL 4.x compiler impacts summary
-  **Itanium architecture impacts summary**

STK File Scanner



– scansummary

- Helps investigate or plan a transition
- Reports **number and types** of API transition impacts in source files

– scandetail

- Helps perform a transition
- Identifies **each instance** of an API transition impact in source files

STK scansummary example



sward's X desktop (rackem.zk3.dec.com:1)

Netscape: HP-UX STK 2.3 scansummary report

File Edit View Go Communicator Help

HP-UX STK 2.3
invent

Fri Apr 16 12:59:58 2004

[identifier type legend](#)
[options used](#)
[interpreting the output report](#)
[faq for scandetail and scansummary](#)

output format:

number of instances: (Identifier type) problem synopsis (synopsis ID)

23: **F** [printf - formatted I/O now converts IEEE infinity and NaN values \(NcWn222\)](#)
23: **F** [printf - 64-bit changes in formatted I/O \(CrCh447\)](#)
23: **F** [printf - floating hex support \(CrCh815\)](#)
15: **F** [syslog - prototype has changed \(CrCh195\)](#)
9: **F** [strlen - new performance archive library \(NcEn669\)](#)
8: **M** [sin_port - range of automatically assigned socket port numbers has changed](#)
8: **S** [sockaddr_in - range of automatically assigned socket port numbers has changed](#)
7: **F** [strcmp - new performance archive library \(NcEn669\)](#)
6: ***** [file - new option prevents following symbolic links \(NcEn469\)](#)
6: **M** [pw_passwd - information may be incompatible with previous versions \(CrWn833\)](#)
5: **F** [fprintf - formatted I/O now converts IEEE infinity and NaN values \(NcWn222\)](#)
5: **F** [fprintf - 64-bit changes in formatted I/O \(CrCh447\)](#)
5: **F** [setsockopt - X/Open Sockets parameters change type from size_t to socklen_t](#)
5: **F** [setsockopt - function values bound by kernel values \(NcEn707\)](#)
5: **F** [sprintf - formatted I/O now converts IEEE infinity and NaN values \(NcWn222\)](#)
5: **F** [sprintf - new version eliminates risk of buffer overflow \(NcEn302\)](#)

Slide 37 of 104

STK scandetail example



```
sward's X desktop (rackem.zk3.dec.com:1)
Netscape: HP-UX STK 2.3 scandetail report
File Edit View Go Communicator Help
ftp.c:987: printf - 64-bit changes in formatted I/O (CrCh447)
ftp.c:987: printf - formatted I/O now converts IEEE infinity and NaN values (CrCh447)
ftp.c:1004: * file - new option prevents following symbolic links (NcEn469)
ftp.c:1015: * ls - new command lc provided (NcEn472)
ftp.c:1041: sockaddr_in - range of automatically assigned socket port numbers
ftp.c:1045: printf - floating hex support (CrCh815)
ftp.c:1045: printf - 64-bit changes in formatted I/O (CrCh447)
ftp.c:1045: printf - formatted I/O now converts IEEE infinity and NaN values
ftp.c:1046: printf - floating hex support (CrCh815)
ftp.c:1046: printf - 64-bit changes in formatted I/O (CrCh447)
ftp.c:1046: printf - formatted I/O now converts IEEE infinity and NaN values
ftp.c:1048: printf - floating hex support (CrCh815)
ftp.c:1048: printf - 64-bit changes in formatted I/O (CrCh447)
ftp.c:1048: printf - formatted I/O now converts IEEE infinity and NaN values
ftp.c:1049: printf - floating hex support (CrCh815)
ftp.c:1049: printf - 64-bit changes in formatted I/O (CrCh447)
ftp.c:1049: printf - formatted I/O now converts IEEE infinity and NaN values
ftp.c:1052: printf - floating hex support (CrCh815)
ftp.c:1052: printf - 64-bit changes in formatted I/O (CrCh447)
ftp.c:1052: printf - formatted I/O now converts IEEE infinity and NaN values
ftp.c:1054: printf - floating hex support (CrCh815)
ftp.c:1054: printf - 64-bit changes in formatted I/O (CrCh447)
ftp.c:1054: printf - formatted I/O now converts IEEE infinity and NaN values
ftp.c:1056: printf - floating hex support (CrCh815)
ftp.c:1056: printf - 64-bit changes in formatted I/O (CrCh447)
ftp.c:1056: printf - formatted I/O now converts IEEE infinity and NaN values
ftp.c:1058: printf - floating hex support (CrCh815)
ftp.c:1058: printf - 64-bit changes in formatted I/O (CrCh447)
ftp.c:1058: printf - formatted I/O now converts IEEE infinity and NaN values
ftp.c:1059: printf - floating hex support (CrCh815)
ftp.c:1059: printf - 64-bit changes in formatted I/O (CrCh447)
ftp.c:1059: printf - formatted I/O now converts IEEE infinity and NaN values
ftp.c:1061: printf - floating hex support (CrCh815)
ftp.c:1061: printf - 64-bit changes in formatted I/O (CrCh447)
ftp.c:1061: printf - formatted I/O now converts IEEE infinity and NaN values
ftp.c:1064: printf - floating hex support (CrCh815)
ftp.c:1064: printf - 64-bit changes in formatted I/O (CrCh447)
ftp.c:1064: printf - formatted I/O now converts IEEE infinity and NaN values
ftp.c:1066: printf - floating hex support (CrCh815)
ftp.c:1066: printf - 64-bit changes in formatted I/O (CrCh447)
100% | http://devresource.hp.com/STK/impacts/iR15.html
Side 30 of 104
```


STK Impact Statement



sward's X desktop (rackem.zk3.dec.com:3)

Netscape: HP's software transition kit

File Edit View Go Communicator Help

transition impacts

hp software transition kit

[printable version](#)

hp STK

- »home
- »announcements
- »overview
- »tools
- »documentation
- »transition impacts
- »identifier types
- »software transition
- »system updates
- »other tools
- »what's new in HP-UX
- »FAQ
- »glossary
- »HP-UX home
- »DSPP Developer Edge

hp-ux 11.00 critical impact:
**printf(), *scanf() - 64-bit changes in formatted I/O (CrCh447)*

problem description

Using 32-bit integer format specifiers with 64-bit integers and pointers may cause your applications to fail. This is a problem only if you recompile your application for 64-bit mode.

identifiers

- `fprintf` `scanf` `vfprintf` `vscanf`
- `fscanf` `sprintf` `vfscanf` `vsprintf`
- `printf` `sscanf` `vprintf` `vsscanf`

old behavior

```
long    i;  
...  
printf("i=%d &i=%d\n",i,&i);
```

STK Impact Statement (continued)



Edge
»Tru64 UNIX migration home
»Linux home
»C/C++ compiler
»send us your feedback
»site map

see also

- Background information on [64 bit API impacts](#)

solution description

Prefix the format specifiers for long or pointer variables with a lowercase l, indicating "long", (that is, %d becomes %ld). Pointers should use the p format specifier when possible.

new behavior

```
long    i;  
printf("i=%ld &i=%p\n", i, &i);
```

see also

- [printf\(3S\)](#), HP-UX 11.00 Version
- [scanf\(3S\)](#), HP-UX 11.00 Version
- [vprintf\(3S\)](#), HP-UX 11.00 Version
- [vscanf\(3S\)](#), HP-UX 11.00 Version

problem summary

classifications	source types	OS release	severity	type
64	C, C++	HP-UX	critical	changed

Binary Scanner



- The abiscanner is a binary compatibility identification tool
 - It scans dynamically linked executables, libraries or object files
 - reports the use of “private” Application Binary Interfaces (ABIs), for each target
- Private system interfaces are **not** supported by HP with regard to binary compatibility
- This does not mean the application will fail, only that the ABI is not supported by HP

http://devresource.hp.com/drc/STK/bin_scanner.jsp

abiscanner is under ‘Other HP-UX Transition Aids’





The Debugger - WDB

- Based on the GNU debugger (gdb) core technology
 - Common technology:
 - Works with other interfaces (e.g. emacs, ddd)
 - Same commands as on many other platforms
 - HP Enhancements
 - Faster startup and reduced memory use
 - Dynamic memory usage debugging
- Supported by HP
 - Support via HP compiler support contract
 - HP added value in areas unique to our platform, e.g. HP compilers, 64-bit, shared libraries, thread support, ...

<http://www.hp.com/go/wdb>



The Debugger - WDB Features

- Supports HP-UX 10.20 --> HP-UX 11i v2
 - PA-RISC and Integrity
- HP C, aC++ and Fortran support
- 32 bit & 64-bit debugging
- Threaded and non-threaded apps
- User space and kernel thread (HP-UX 11.x) support
- Provides command line as well as GUI interface
- GUI interface
 - Designed to be fast
 - Designed to be straightforward to use

Preparing PA-RISC source code for Integrity



- Use Software Transition Kit (STK) to identify Integrity transition issues
 - Ensure libraries and APIs are available on Integrity HP-UX 11i
 - See “Preparing Source Code for HP-UX on Integrity” document at

<http://devresource.hp.com/drc/STK/docs/refs/prepsource.jsp>

- Use shared libraries
 - Remove archive libraries from the build
- Use kernel threads (not CMA “DCE” threads)
 - Do not use libcma

Threads Impact



sward's X desktop (rackem.zk3.dec.com:1)

Netscape HP-UX Software Transition Kit

File Edit View Go Communicator Help

» **Dev Resource Central**

»HP STK home

HP-UX STK

- »Home
- »Overview
- »Tools
- »Documentation
- »**Transition impacts**
- »Identifier types
- »Impact list
- »Porting to HP-UX
- »FAQ
- »Glossary
- »Help

»Send us feedback

Site maps

- »HP-UX STK
- »Dev Resource Central

HP-UX 11.00 critical impact:

threads – APIs changed for kernel-based threads (CrCh637)

Problem description

The threads implementation of 11.x brings HP-UX in line with the POSIX 1003.1c standard, which includes the standard POSIX interface (1003.1), the POSIX real-time scheduling standard (1003.1b), and the implementation of kernel threads (1003.1c).

Identifiers

<code>pthread_cleanup_push</code>	<code>pthread_exit</code>
<code>pthread_cond_init</code>	<code>pthread_getspecific</code>
<code>pthread_cond_timedwait</code>	<code>pthread_join</code>
<code>pthread_create</code>	<code>pthread_mutex_init</code>
<code>pthread_detach</code>	<code>pthread_once</code>
<code>pthread_equal</code>	<code>pthread_setspecific</code>

See also

- Background information on [standards compliance impacts](#)
- Background information on [threads impacts](#)
- `pthread_cleanup_push(3T)`, HP-UX 10.20
- `pthread_cond_init(3T)`, HP-UX 10.20

Shells



- HP-UX 11i v2 includes the following shells:
 - POSIX shell : sh, rsh
 - Korn shell: ksh, rksh
 - Key Shell: keysh (ksh with soft-key menus and context sensitive help)
 - C shell: csh

As of HP-UX 11i v1.5, the traditional bourne shell has been replaced by the POSIX shell. The POSIX shell provides a high degree of compatibility with bourne shell scripts.

- See the sh-posix man page:

<http://www.docs.hp.com/hpux/onlinedocs/B2355-60103/00/03/390-con.html>

Java



- Java Versions (Java 2 Standard Edition)
 - Simultaneous releases on PA-RISC and Integrity
 - SDK and RTE 1.3 & 1.4 - released March 2004
 - Built from common source
- Java Tools
 - Same tool
 - Same tuning process across PA-RISC and Integrity
- Java Runtime Parameters
 - In general, the same
 - A few exceptions (<http://www.hp.com/go/java>)

Linking



- Default linking is to shared libraries before linking archive libraries
- EPIC architecture benefits from “inline” functions, linker supports this (PA-RISC 64 also has inline, but more limited)
- Default search path for libraries:

	PA-RISC	Integrity
32 Bit	/usr/lib /usr/ccs/lib	/usr/lib/hpux32 /usr/ccs/lib/hpux32
64 Bit	/usr/lib/pa20_64 /usr/ccs/lib/pa20_64	/usr/lib/hpux64 /usr/ccs/lib/hpux64

Use Dynamic/Shared Libraries



- Avoids binding in architecture dependencies
- Pick up patches as libraries are fixed, improved
- Performance advantage of static libraries has eroded
- As of HP-UX 11i v2, system libraries are only provided in shared format (i.e. libc.so, not libc.a)

PA-RISC Libraries on Integrity



- A few libraries are currently not available in Integrity native format
 - CDE
 - Audio (/opt/audio/lib/*)
 - Image (/opt/image/lib/*)
 - DCE
 - Graphics (Starbase, PEX, and PHIGS)

- *An application using these libraries must be built as a PA-RISC binary and execute on Integrity through Aries*

If your port to Integrity requires any of these, we want to hear about it!

Compiler Optimization Levels (PA-RISC and Integrity)



- +O1 performs local optimizations (default)
- -g enables debugging and +O1 optimizations
- +O2 (same as -O)
 - Performs intraprocedural optimization, plus user-directed inlining
- +O3 performs interprocedural optimization within a source file and high level loop optimizations
- +O4 performs cross-module optimization within a load module
- +Ofast provides a combination of options valid for most applications:
 - O +Onolimit +Ofitacc=relaxed +FPD +DSnative +Oshortdata

Compiler Optimization +O4 Optimization



- Combination of PBO and +O4 is the most powerful of all build processes
- Above -O (a.k.a. +O2), the High-Level Optimizer (HLO) optimizes whole files or programs across procedure boundaries
 - +O4 code is optimized at link time
- +O4, the HLO's scope is all the .o files compiled with +O4
 - For large products, this can imply use of very large amounts of virtual memory
 - Compiler switch available to control scope of +O4 optimization and limit memory use

Optimization Techniques



Libm Inlining Performance Results

4X-8X increase in throughput for inlined sequences vs libm calls, in tight software-pipelined loops

	Single	Double	Extended
sqrt (default)	5.62	5.99	5.75
sqrt (relaxed)	6.61	6.38	6.32
exp (default)	6.99	7.20	8.76
exp (relaxed)	7.02	7.65	7.20
log (default)	5.29	4.38	3.40
log (relaxed)	4.81	3.72	4.14
COS (default)	6.47	7.19	5.03
COS (relaxed)	6.47	7.19	6.93

–without loss of accuracy or special-case behavior

–without requiring recoding for vector interface

–many (~175) sequences

C Compiler portability Diagnostic and lint



- +w1 (one) - enables all warnings
- -Aa - enables strict ANSI mode
- Lint - dropping support, obsolete in UNIX2003
- Flexelint

<http://www.gimpel.com/html/products.htm>

- Documentation

- HP aC++ Transition Guide

http://devresource.hp.com/STK/partner/acc_transguide.pdf

- Using Templates in HP aC++

<http://devresource.hp.com/STK/partner/templates.pdf>

Integrity Optimization - PBO



- Profile-Based Optimization (PBO)
 - Optimizations based on observed run-time characteristics of the program
 - The most effective build process optimization
 - Can be a difficult build process change to implement for products
 - Requires three steps
 - Instrument, collect, optimize
 - Greatest impact on +O2 or higher optimizations
 - C, C++ and Fortran 90 supported

Integrity Optimization - PBO Usage



– What products use HP-UX PBO?

- The HP-UX kernel
- All the commercial databases
- All the HP-UX compilers and linker
- Synopsys Design Compiler:
http://www10.edatoolsafe.com/nbc/articles/view_article.php?articleid=28928
- SAP supply chain management :
http://www.hp.com/products1/servers/rackoptimized/rp7410/infolibrary/5980-7535EN_SAP.pdf

Optimization Techniques



Contributors to SPECint2000 Performance

	Performance Delta
gcc -O	
aCC -O	27%
aCC -O +PBO	38%
aCC +O3 +PBO	45%
aCC +O4 +PBO	64%
aCC <i>all base options</i>	73%
aCC <i>all peak options</i>	76%

All measurements on 1GHz Integrity System running HP-WORLD 2004
Solutions and Technology Conference & Expo

Integrity Performance Tools - Caliper



- HP Caliper includes support for:
 - Programs compiled for Integrity based systems
 - Code generated by HP aC++, C++ and Fortran compilers
 - Including inlined functions and C++ exceptions
 - Programs compiled with optimization or debug information, or both
 - Including support for both the +objdebug and +noobjdebug options
 - Both ILP32 (+DD32) and LP64 (+DD64) programs
 - Both 32-bit and 64-bit ELF formats
 - Archive-bound or shared-bound executables
 - Both single- and multi-threaded applications, including MxN threads
 - Applications that fork() or vfork() or exec() themselves or other executables
 - Shell scripts and the programs they spawn

Migrating
from Tru64 UNIX
to HP-UX 11i v2

Tru64 UNIX V5.1B and Its Updates



V5.1B, with its enhancements, will be supported for at least 5 years beyond last sale date of AlphaServer systems

V5.1B-1 "Vail" 2003

- Scaling to 64 CPUs
- Continued leadership storage SAN support
- Resiliency enhancements
- Enhanced availability and disaster tolerance
- Migration tools availability

V5.1B-2 "Utah" 2004

- Support EV7z AlphaServers
- Improved storage management and new storage options
- Resiliency enhancements
- HP-UX 11i compatibility tools

Maintain binary compatibility,
Continue focus on quality and stability.

Current as of March 18, 2004 – Tru64 UNIX future plans subject to change without notice – Customer Viewable



Tru64 UNIX and HP-UX Comparison



- HP-UX is based on System V with features from 4.x BSD
- Tru64 UNIX environment is based on 4.x BSD with features from System V
- HP-UX and Tru64 UNIX conform to multiple common standards
- See appendix for more details about standards

Development Environment Comparison



- Compilers
- Linkers
- Debuggers
- Make
- Threads
- Libraries and APIs
- Floating Point Exception Handling
- Data Alignment
- Endianism
- Security

Compilers



- C, C++, Fortran, Java and assembler are covered in porting guide
- Includes tables mapping Tru64 UNIX compiler options to HP-UX compiler options
- Tru64 UNIX Migration Environment for HP-UX includes a cc, c++ and a linker driver to map Compaq C compiler options to the equivalent options for HP C
 - The drivers generate 64bit code by default
- NOTE:
 - HP-UX compilers generate 32-bit objects by default
 - Use +DD64 option to generate 64-bit image

C Compilers



- Both Compaq and HP C compilers support ANSI C
- Strictly conformant code will compile and run without change
- Turn on strict ANSI checking (-std1 option) to find non-compliant code
- HP ANSI C compiler for Integrity platforms does not support K&R mode
- Use Prototypes

C++ Compilers



- Both Compaq and HP C++ compilers support ANSI C++
- Default for Compaq C++ is `-std ansi` , which supports commonly used extensions.
- Use `-std strict_ansi` option for Compaq C++ to flag any nonstandard code
- Use `-Aa` option with HP aC++ to enable ANSI C++ standard features like standard scoping rules for variables declared in conditional statements like for-loops



C++ Compilers

- Functionality in ANSI/ISO C++ standard not yet supported by HP aC++:
 - Support for universal character sequences (\uxxxx)
 - Template features:
 - Separation model for template compilation (export keyword)
 - Template template parameters
 - Omission of template parameter names
 - Compaq C++ supports all but **export** keyword

Fortran



- Both Compaq and HP Fortran compilers conform to Fortran 90 and Fortran 95 standards
- Both support a full OpenMP implementation
- Compaq Fortran highly compatible w/ FORTRAN 77. HP Fortran does not support FORTRAN 77 nonstandard extensions
- See HP Fortran Programmer's Guide for info on migrating to Fortran 90:

<http://docs.hp.com/hpux/dev/index.html#Fortran>



Library Search Order

- Library search order differs between Tru64 UNIX and HP-UX linkers:
 - Tru64 UNIX searches all directories in the search path for shared libraries and then makes a second pass to search for archives
 - HP-UX linker makes one pass of all directories, looking for shared libraries then archives in each one
 - like `-oldstyle_liblookup` on Tru64 UNIX
- May need to explicitly specify library names or change order of directory specifications on link line

Uninitialized Global Data



- Resolution of uninitialized global data differs between HP-UX and Tru64 UNIX:
 - On Tru64 UNIX, external definition is used if available, otherwise, uninitialized element used for symbol resolution
 - On HP-UX, uninitialized global data references are implicitly initialized to zero, and data references are resolved by first definition encountered rather than continuing to search for an initialized data object



Debuggers

- See Guide to Using HP WDB for Ladebug Users
 - <http://h30097.www3.hp.com/transition/apps/usage.html>
- Use +noobjdebug linker option on Integrity to put debug info in executable (to match Tru64 UNIX behavior)
- Debugging shared libraries requires use of the chatr(1) command on Integrity, or pxdB on PA-RISC, if attaching to a running process:
 - On Integrity:
% chatr +debug enable a.out
 - On PA-RISC:
% /usr/ccs/bin/pxdb -s on a.out

Make



- Tru64 UNIX provides three versions of make
- `/usr/bin/posix/make` is most like the HP-UX System V make
- The default make on Tru64 UNIX (`/usr/bin/make`) has been ported to HP-UX (available in the Tru64 UNIX Migration Environment for HP-UX)
- `cc`, `c++` and `ld` drivers in Tru64 UNIX Migration Environment for HP-UX may be used to help port make files

Floating-Point Exception Handling



- Default handling mode for floating-point exceptions is different between Tru64 UNIX and HP-UX:
 - HP-UX suppresses all exceptions by default (IEEE behavior); Tru64 UNIX does not
 - Tru64 UNIX ignores underflow and inexact exceptions, but overflow, divide-by-zero, and invalid operations will generate a trap
 - Tru64 UNIX denormalized values are flushed to zero (fast underflow) by default
- Tru64 UNIX default behavior can be obtained on HP-UX by using the **+FPOVZD** compiler option:

```
% cc +FPOVZD foo.c
```

Data Alignment



- Alpha, PA-RISC and Itanium processors require data to be aligned on natural boundaries
- Tru64 UNIX provides support for handling unaligned data accesses, either by silently correcting or issuing message
- HP-UX does not perform unaligned access fixups
 - SIGBUS signal is raised, allowing application to handle or fatally exit
- Unaligned data access fixup on Alpha causes performance hit (on HP-UX, a fatal error occurs)

Compiler Support for Accessing Unaligned Data



- Tru64 UNIX compilers provide `__unaligned` keyword to generate code for unaligned loads and stores
 - More expensive than aligned data access, but faster than generating alignment faults
- HP C and C++ compilers on Integrity, and C++ compiler on PA-RISC provide `UNALIGN` pragma which affects the next typedef declaration:

```
#pragma UNALIGN 1
typedef int ua_int; // ua_int has 1 byte alignment
typedef int a_int: // a_int has natural alignment
```

Handling Data Storage in Endian Neutral Formats



- Store data in defined endian format
 - e.g. network data and Java traditionally use big endian

- Add additional data to indicate format
 - Fortran allows endianness control based on OPEN statement

- Store data in endian neutral format such as ASCII strings
 - e.g. XML data

Transition Tools



- Binary scanner (appscan)
 - Utility to list all dependencies (shared libraries and symbols) of an executable and the disposition on HP-UX of each API
- Tru64 UNIX to HP-UX Software Transition Kit
 - Identifies compatibility issues between Tru64 UNIX and HP-UX
- Tru64 UNIX Migration Environment for HP-UX
 - Runtime environment on HP-UX for select Tru64 UNIX APIs, tools and commands

<http://www.hp.com/go/STK>

Tru64 UNIX Migration Environment



Runtime Environment for Tru64 UNIX APIs, development tools and commands/utilities on HP-UX

- Assists you in migrating Tru64 UNIX applications rapidly, in their move towards becoming native HP-UX applications
- Expected to be integrated in future versions of HP-UX for better compatibility

<http://devresource.hp.com/STKT/>

Tru64 UNIX Migration Environment is under “Other Tru64 UNIX Transition Aids”



Migration Environment Utilities and Commands

- Tru64 UNIX make
- Tru64 UNIX cc, cxx, ld wrappers
- Tru64 UNIX mkcatdefs
- Tru64 UNIX dspcat
- Tru64 UNIX dspmsg
- Tru64 UNIX mcs
- Tru64 UNIX tar
- Tru64 UNIX cpio
- Tru64 UNIX pax
- HP-UX stty (Modified to add -dec)

Migration Environment Libraries



– libtru64.so

- Contains APIs intended to become native on HP-UX
 - safe_open, mkdtemp, mkstemp
 - flock, memcntl

– libtru64_ext.a

- Static library that contains APIs that will NOT move forward to HP-UX
 - TIS APIs
 - Use the threads libraries on HP-UX
 - sigvec, sigsetmask, sigblock
 - Use POSIX routines on HP-UX

Migration Environment



– cc and cxx jacket

- The cc jacket translates Tru64 UNIX compiler command line to HP-UX compiler command line and executes it
- It can also be used to port makefiles and scripts to HP-UX by reporting translations for Tru64 UNIX cc commands (-port_info option)

– ld jacket

- The ld jacket translates Tru64 UNIX link command line to HP-UX command line and executes it
- It can also be used to port makefiles and scripts to HP-UX by reporting translations for Tru64 UNIX ld commands (-port_info option)

Software Links



- ARIES
<http://devresource.hp.com/drc/STK/docs/refs/Aries.jsp>
- STK Homepage <http://devresource.hp.com/STK/>
- abiscanner
http://devresource.hp.com/drc/STK/bin_scanner.jsp
- WDB <http://www.hp.com/go/wdb>
- Java <http://www.hp.com/go/java>
- Flexelint <http://www.gimpel.com/html/products.htm>
- Fortran <http://docs.hp.com/hpux/dev/index.html#Fortran>
- Migration Environment
<http://h30097.www3.hp.com/transition/apps/me.html>



Documentation Links

- Documentation
 - HP aC++ Transition Guide
http://devresource.hp.com/STK/partner/acc_transguide.pdf
 - Using Templates in HP aC++
<http://devresource.hp.com/STK/partner/templates.pdf>
 - “Preparing Source Code for HP-UX on Integrity” document
<http://devresource.hp.com/drc/STK/docs/refs/prepsource.jsp>





HP WORLD 2004

Solutions and Technology Conference & Expo

Co-produced by:



RECOMMENDED TRAINING VENUE FOR THE
HP Certified Professional





i n v e n t

Supplemental Material



Porting Code to 64-bit

- Why?
 - Large files & address space
 - Performance advantage of memory over disk I/O
- Beware!
 - Data size differences (pointers, longs and predefined types)
 - Data truncation (casts, return values...)
 - Data type promotion pitfalls
 - Data alignment and padding differences
 - Hard coded constants
- 64-bit Safe vs. 64-bit Aware
 - Compile +DD64, fix compile/link errors and possibly +M2 warnings
 - Convert program to really use 64-bit addressing
- Impacts
 - Performance costs are real but tuning can help



Preparing for Port to 64-bit

- Get the whitepaper
 - <http://devresource.hp.com/drc/STK/docs/refs/64concepts.jsp>
- Find all your source code - check under the couch
- Acquire 64-bit versions of all 3rd party libraries
- Port all of your own libraries to 64-bit
- Rewrite assembly code to be 64-bit
- Add +DD64 +M2 to makefiles
- Run lint -Q to catch int/pointer issues or use flexelint
- Get the STK and use it to catch porting issues

Standards Conformance



Standard	Tru64 UNIX	HP-UX
IEEE POSIX 1001.3c Kernel threads	✓	✓
IEEE POSIX 1003.1-1996 System calls	✓	✓
IEEE POSIX 1003.1b Real-time APIs	✓	✓
IEEE POSIX 1003.2 Commands and Utilities	✓	✓
X/Open Portability Guide (XPG3, XPG4)	✓	✓
Single UNIX Specification V1 (UNIX 95)	✓	✓
Single UNIX Specification V2 (UNIX 98)	✓	Almost!

Standards Conformance (continued)



Standard	Tru64 UNIX	HP-UX
System V Interface Definition (SVID3)	✓	✓
X11 Window System, Font Server and Clients	R6.5	R6.2
OSF/Motif 2.1	✓	✓
FIPS 151-2	✓	✓
FIPS 189	✓	✓
LP64	✓	✓

Namespaces



Tru64 UNIX	HP-UX	Standard
-D_OSF_SOURCE (Default)	-D_HPUX_SOURCE (Default for -Ae)	Proprietary interfaces
-D_XOPEN_SOURCE=500	-D_XOPEN_SOURCE=500 or -DUNIX_STD=98	UNIX 98
-D_XOPEN_SOURCE_EXTENDED	-D_XOPEN_SOURCE_EXTENDED	UNIX 95
-D_XOPEN_SOURCE (Default)	-D_XOPEN_SOURCE	XPG4
-D_POSIX_SOURCE	-D_POSIX_SOURCE	POSIX
-D_ANSI_C_SOURCE	Default for -Aa and c89	ANSI C



Example: Define Namespace when using `-Aa` on HP-UX



- `/* open.c */`
- `#include <fcntl.h>`
- `#include <stdio.h>`
- `main(int argc, char *argv[]) {`
- `int fd1;`
- `fd1 = open("testfile", O_CREAT|O_RDWR, 0666);`
- `if(fd1 < 0) { perror("open testfile"); exit(1);}`
- `printf("file descriptor %d opened\n", fd1);`
- `}`

- `% cc -Aa open.c`
- Error 172: "open.c", line 9 # Undeclared variable 'O_CREAT'
- `% cc -Aa -D_POSIX_SOURCE open.c`

Example: Calling Functions w/o Prototypes on HP-UX



```
/* noproto.c */
#include <stdio.h>
int main() {
    long a;
    a = foo();
    printf ("The returned value is:
    0x%lx\n", a);
    return 0; }
```

```
% cc +DD64 noproto.c foo.c
```

```
% ./a.out
```

```
The returned value is: 0x55667788
```

```
/* foo.c */
long foo()
{ return 0x1122334455667788L; }
```

```
% cc +tru64 +DD64 noproto.c foo.c
```

```
% ./a.out
```

```
The returned value is:
0x1122334455667788
```



Scoping Rules for Variables in a for loop



```
• /* forloop.C */
• int main() {
• int sum = 0;
• for (int k = 0; k!=100; ++k) /* line 3 */
•     sum += k;
• for (int k = 100; k!=0; k--) /* line 5 */
•     sum += k;
• }

• % aCC forloop.C
• Error 173:"forloop.C", line 5 # Redefined symbol 'k';
  previously defined at ["forloop.C" line 3].
• % aCC -Wc,-ansi_for_scope,on forloop.C
```



Example: Library Search Order

**Libraries L1/libA.a
L2/libA.so
L2/libB.so**

Tru64 UNIX

```
% cc -LL1 -LL2 main.c -lA -lB -  
  rpath L1:L2  
% ldd a.out  
/usr/bin/ldd a.out  
Main => a.out  
libA.so => L2/libA.so  
libB.so => L2/libB.so  
libc.so => /usr/shlib/libc.so
```

HP-UX

```
% cc -LL1 -LL2 main.c -lA -lB  
  -Wl,+vtype,files  
Loading main.o:  
Loading L1/libA.a[a.o]:  
Loading L2/libB.so:  
Loading /usr/lib/hpux32/libc.so:  
Loading /usr/lib/hpux32/libdl.so.1:
```



Example – Uninitialized Global Data

```
• % cat a.c
• int object; /* Uninitialized global symbol */
• void a()
• { printf("\tA object is %d\n", object); }
• % cat b.c
• int object=1; /* Initialized global symbol */
• void b()
• { }
• % cat main.c
• int main(int argc, char* argv[])
• { a(); }
```

Example – Uninitialized Global Data (continued)



Tru64 UNIX	HP-UX
<pre>% cc -c main.c % cc -shared a.c -o libA.so % cc -shared b.c -o libB.so % cc main.o -L. -rpath . -lA -lB - o test % ./test object is 1</pre>	<pre>% cc +DD64 -c main.c % cc +DD64 -b a.c -o libA.so % cc +DD64 -b b.c -o libB.so % cc +DD64 main.o -L. -Wl,+b,. -lA -lB -o test2 % ./test2 object is 0 NOTE: result is link order dependent!</pre>

Resolution: Declare object as extern in a.c:
`extern int object;`



i n v e n t