Porting OpenVMS Applications to the Itanium™ Processor Family

Gaitan D'Antoni
OpenVMS Engineering
gaitan.dantoni@hp.com

Session 2212

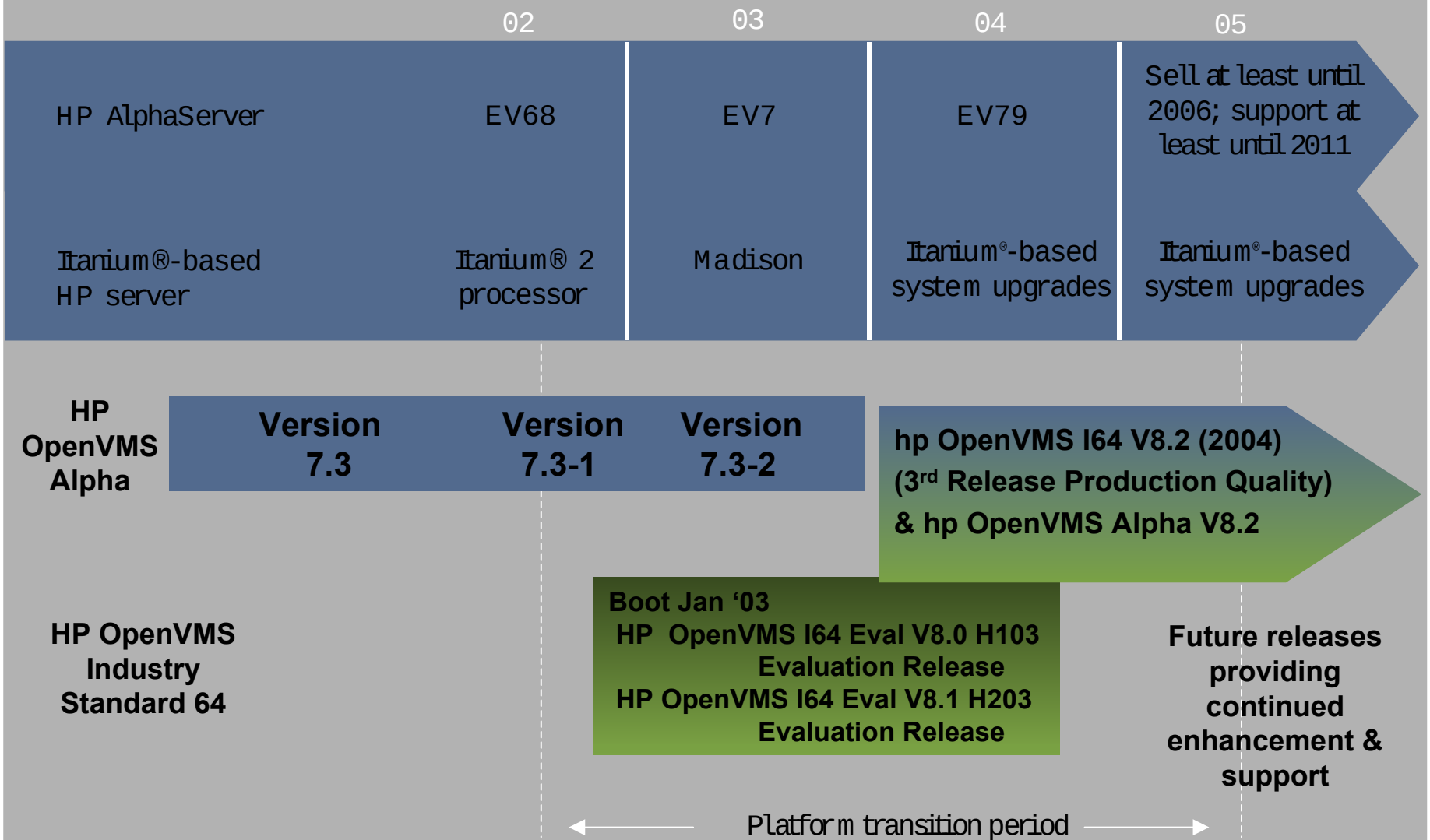
HP WORLD 2003
Solutions and Technology Conference & Expo

HP OpenVMS Industry Standard 64 for Integrity Servers

HP OpenVMS I64

HP WORLD 2003
Solutions and Technology Conference & Expo

hp OpenVMS Roadmap



hp OpenVMS Industry Standard 64 (I64) Release Roadmap

H103 **H203** **H104** **H204**

1st Boot on i2000 system, January 31, 2003 3:31 PM EST

Boot on rx2600 system (First Ship platform), March 17, 2003

First Ship

H103: hp OpenVMS I64 Eval V8.0 "Mako" Evaluation Release
Audience: Selected ISVs and Partners
OpenVMS Itanium Operating System, Monitor Utility
Networks: DECnet Phase IV, TCP/IP
Development Tools: Cross Linker, Cross Librarian, Native Debugger
Cross Compilers: C, C++, BLISS, FORTRAN, IMACRO

H203: hp OpenVMS I64 Eval V8.1 "Jaws" Evaluation Release
Audience: Key ISVs, Partners, Early Adopters
Limited cluster functionality (4 nodes)

Native Compilers: C, C++, BLISS, FORTRAN, IMACRO, Pascal, BASIC, COBOL

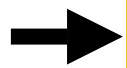
Additional Language Support: JAVA

Additional Layered Products…Networks, Data Serving, Security, eBusiness Integration, Application Development

Internal releases

External releases

Production Quality



HP OpenVMS I64 V8.2

Progress Since First Boot in Jan 2003

- Base O/S (includes RMS, XQP, Image Activator, SMP...)
- Utilities (BACKUP, BAD, COPY, CLI, DCL, DIR, MAIL, UTIL32, EDT, ...)
- SCSI Storage (U160, U320)
- Clusters, IPC
- XFC
- LAN support (Intel 8255x 10/100, Broadcom 5701 10/100/1000)
- LAT
- FDDI
- DECnet Phase IV and TCP/IP
- DFS
- DECwindows client (many apps)
- Cross and Native SDA (System crash, process crash)
- SORT
- RAMdisk
- ODS-5 Support
- INSTAL/PRIV
- RMS Convert and FDLSHR
- MONITOR
- PCSI
- VMSINSTAL
- DECthreads
- DDTM

Progress Since First Boot in Jan 2003

- Cross Linker
- Cross Librarian
- Cross Message Compiler
- Cross and Native Command Definition Utility
- DECset: CMS, MMS, DTM, LSE, TPU
- CHECKSUM
- LIBRTL, CRTL, MATHRTL, CVT
- XDELTA
- DEBUG
- BUGCHECK
- ZIP/UNZIP
- Base security (Except ACME and Security Server)
- Security System Services
- LOGINOUT
- AUTHORIZE
- SET/SHOW Security
- Audit Server and Analyze/Audit
- Encryption
- Accounting

Porting Goals

- Provide an operating system environment, development tools, and documentation to make porting as easy as possible
  - Full port of the Operating System, Runtime Libraries, development tools and most layered products
  - Recompile, relink, requalify
- Use our experiences porting the operating system to make it easier for others to port their applications
  - Internal layered product groups, partners, and customers

Porting Applications

HP WORLD 2003
Solutions and Technology Conference & Expo

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
- Pascal V5.9 – Planned release – H2 2003

Itanium® Compiler Plans (1 of 3)

- C
  - CPQ C
    - Itanium® architecture implementations of OpenVMS CPQ C V6.5 compiler
    - Use for recompile/relink/requalify
    - GEM backend code generator
    - Will be available starting with the initial OpenVMS release
  - C Dialect Support in C++ Compiler
    - Will include some features from CPQ C but may require source code changes
    - Compiler for moving forward
    - Intel® backend code generator
    - We plan to make this available with a future release of OpenVMS

Itanium® Compiler Plans (2 of 3)

- C++
  - Based on the same front end compiler technology as Compaq C++
  - Use for recompile/relink/requalify
  - Intel® backend code generator
- COBOL, BASIC, PASCAL, BLISS
  - Itanium® architecture implementations of the current OpenVMS compilers
  - GEM backend code generator
- Java
  - Itanium® architecture implementation of J2SE V1.4.1

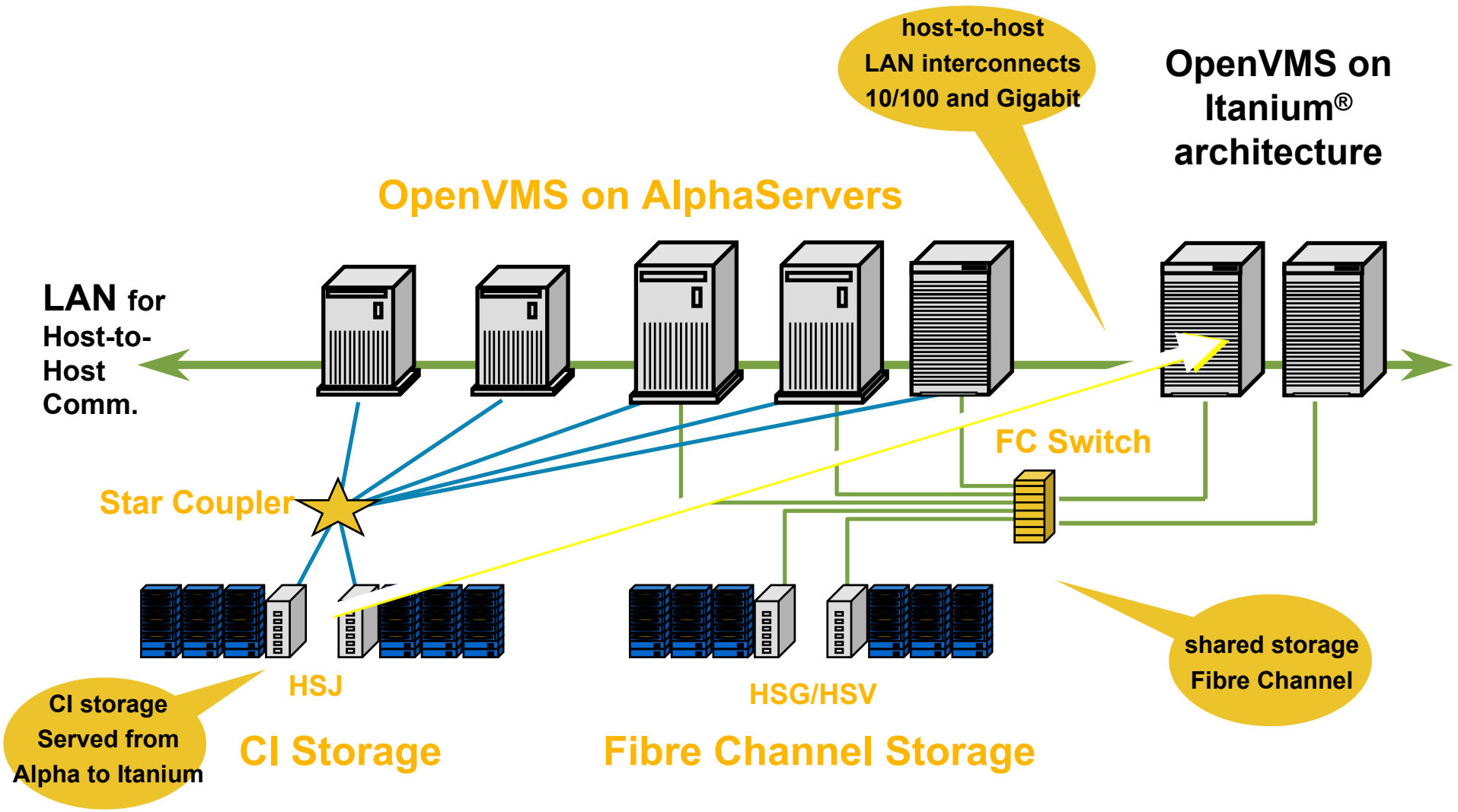
Itanium® Compiler Plans (3 of 3)

- FORTRAN
  - Itanium® architecture implementation of the current OpenVMS Fortran 90 compiler
  - GEM backend code generator
  - Our plan is to replace GEM with the Intel® backend code generator in a future release in order to take advantage of enhancements in processor chip technology
- IMACRO
  - Compiles ported VAX Macro-32 code for Itanium® architecture
  - Itanium® architecture equivalent of AMACRO
  - GEM backend code generator
- 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

Itanium® architecture and OpenVMS Clusters



So, what's different? (1 of 4)

- Calling Standard
  - publicly available today at http://www.hp.com/products1/evolution/alpha_retaintrust/openvms/resources.html
  - Intel® calling standard with OpenVMS modifications
    - No frame pointer (FP)
    - Multiple stacks
    - only 4 preserved registers across calls
    - Register numbers you're familiar with will change
  - All OpenVMS provided tools will "know" about these changes
  - Your code that "knows" about the Alpha standard will almost certainly need to change

So, what's different? (2 of 4)

- Object file format
  - ELF/DWARF industry standards plus our extensions
    - ELF - Executable and Linkable Format, Itanium® Architecture object code, images, etc.
    - DWARF - Debugging and traceback information (embedded in ELF).
  - All OpenVMS provided tools will "know" about these changes
  - User written code that "knows" the object file format may have to change
  - We will be publishing these specifications in the near future

So, what's different? (3 of 4)

- Floating point data types
  - Itanium® architecture supports IEEE float only
  - All compilers that currently support F, D, G, S, T, and X (S and T are native IEEE formats) will continue to do so on Itanium® architecture
  - IEEE will be the default
  - HP will update the appropriate Runtime Libraries to add IEEE interfaces where needed
  - White Paper with technical details about the differences between VAX Float and IEEE Float is available at http://www.hp.com/products1/evolution/alpha_retaintrust/openvms/resources.html

So, what's different? (4 of 4)

- Source Code that May Need to Change
- Architecture Specific code
  - All Alpha assembler code must be rewritten
- Conditionalized code
  - Build command files
    - $ if .not. Alpha ! Assumes VAX
  - Application source code
    - #ifndef (alpha) // Assumes VAX
    - C asm code
- We will be providing a new Porting Guide with details

The HP OpenVMS Itanium® Calling Standard

HP WORLD 2003
Solutions and Technology Conference & Expo

What's the Problem?

- On VAX and Alpha
  R0 = function return value
- On Itanium® architecture
  R0 = zero

What's a Calling Standard?

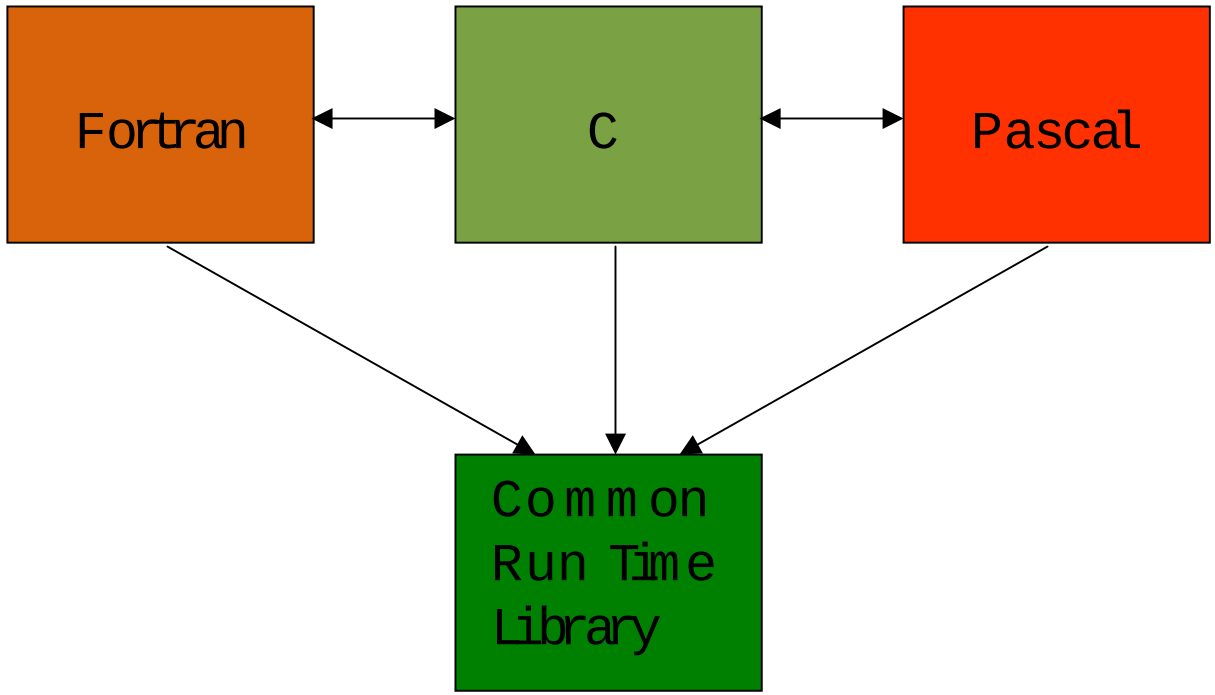
- In C

  Z = func (X, Y);

- Binary representation of subroutine linkage:
  - Where are my arguments?
  - Where's my execution environment?
  - How do I get back to my caller?
  - How do exceptions get handled?

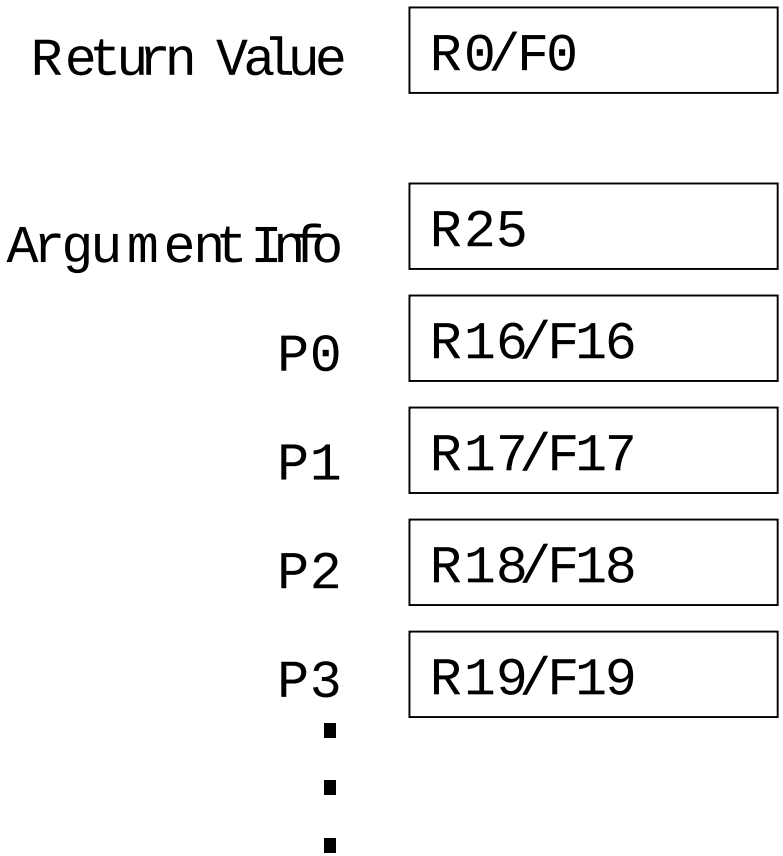
Why Have a Calling Standard?

- Multi-language interoperable programming environment



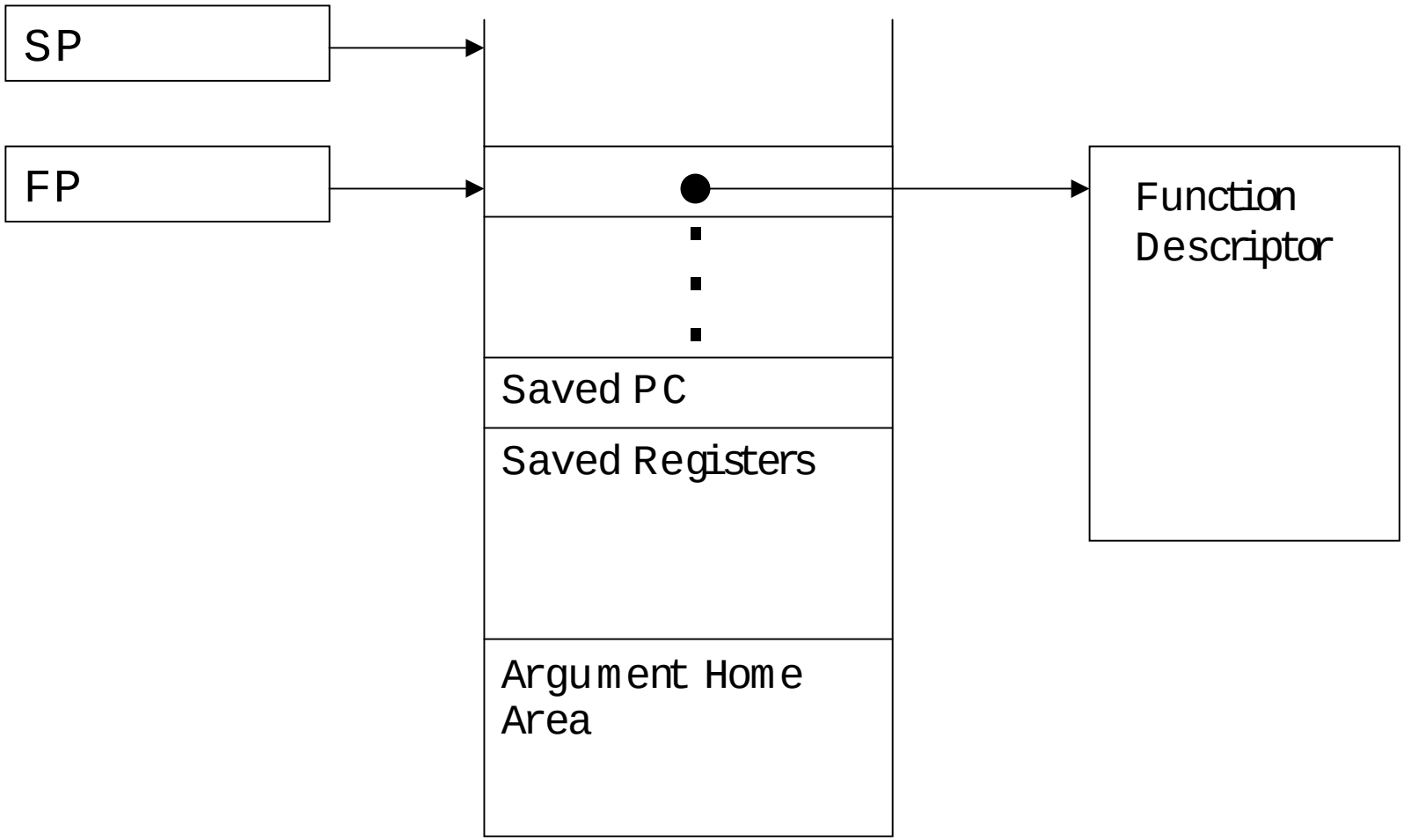
The Past

- Alpha – Argument List & Return Value



The Past

- Alpha – Call Frame and Function Descriptor



The Present

- Intel® defined Itanium® Calling Conventions
- Comparable to Alpha calling standard, but lacking
  - Argument count / information
  - VAX/Alpha floating point datatypes
  - Support for translated images
  - Definition of invocation context handle

Our Approach

- Adopt the mainstream Itanium® standards
  - Intel® calling standard
  - ELF object / image file format
  - DWARF debug information format
- Extend / specialize as needed to support existing VMS features

Benefits

- Intel® has accepted our extensions
- Intel compilers (C / C++, Fortran)
- Industry standard tools
  - Object file post-processors / analyzers
  - Linux linker

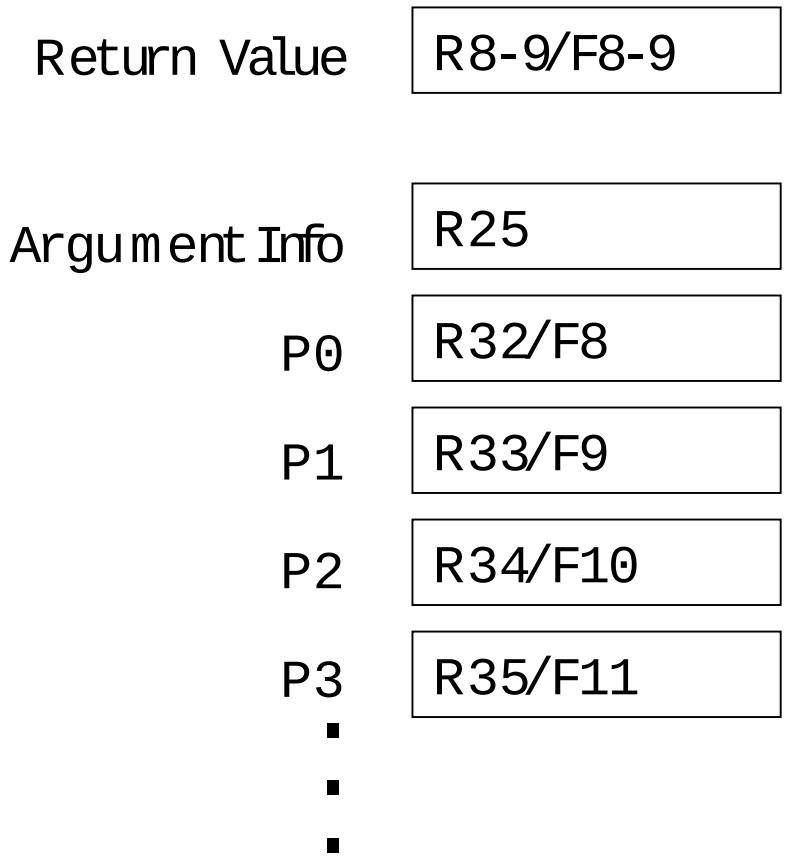
The Future

- OpenVMS Itanium® Calling Standard
- Based on industry standard Itanium Calling Conventions
- Extended for OpenVMS
  - Argument count / information register
  - VAX/Alpha floating point formats
  - Translated image support
  - Additional definitions

Differences Between Alpha and Itanium® Architectures

- Different registers for arguments and return value
- Rotating registers and separate register stack
- GP (global data pointer) register
- Different sets of scratch and saved registers
- PC range based condition handling

Argument List & Return Value



Compiler Support for Migration

- Register Mapping
  - VAX Macro compiler maps Alpha register numbers to their Itanium® architecture equivalents
  - Bliss compiler supports user switchable mapping

Impact on Applications

- Mostly none
- Except for
  - Explicit register use
  - Knowledge of stack and exception frame format

For further Information about OpenVMS on Itanium

- OpenVMS on the Itanium® Architecture Web Sites
  - General OpenVMS on Itanium information
    http://www.hp.com/products1/evolution/alpha_retaintrust/openvms/resources.html
  - Layered products schedules
    http://www.hp.com/products1/evolution/alpha_retaintrust/openvms/openvms_move.html
  - Layered products plans (products that either will not be ported or are under review)
    http://www.hp.com/products1/evolution/alpha_retaintrust/openvms/openvms_plans.html
  - OpenVMS Partner plans
    http://www.hp.com/products1/evolution/alpha_retaintrust/openvms/partners.html

Interex, Encompass and HP bring you a powerful new HP World.