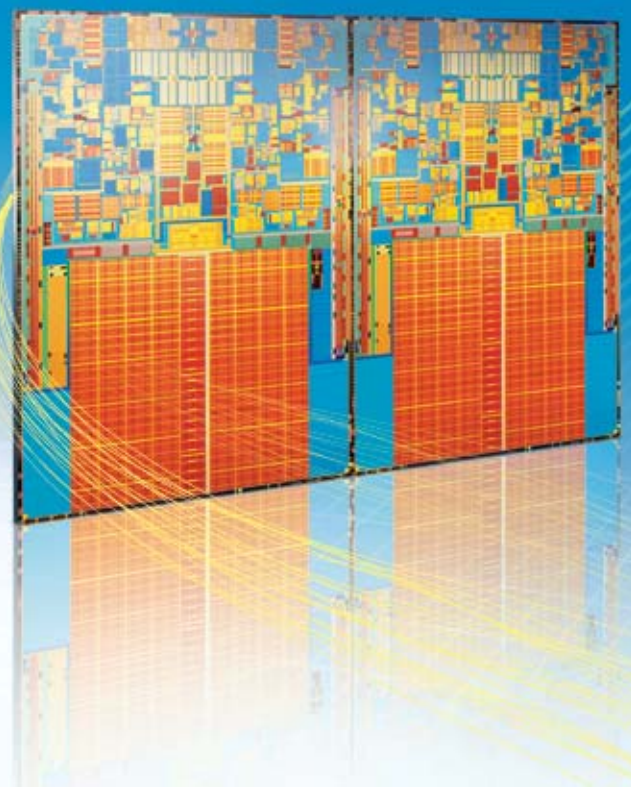
Emerging Unified Extensible Firmware Interface (UEFI) Capabilities



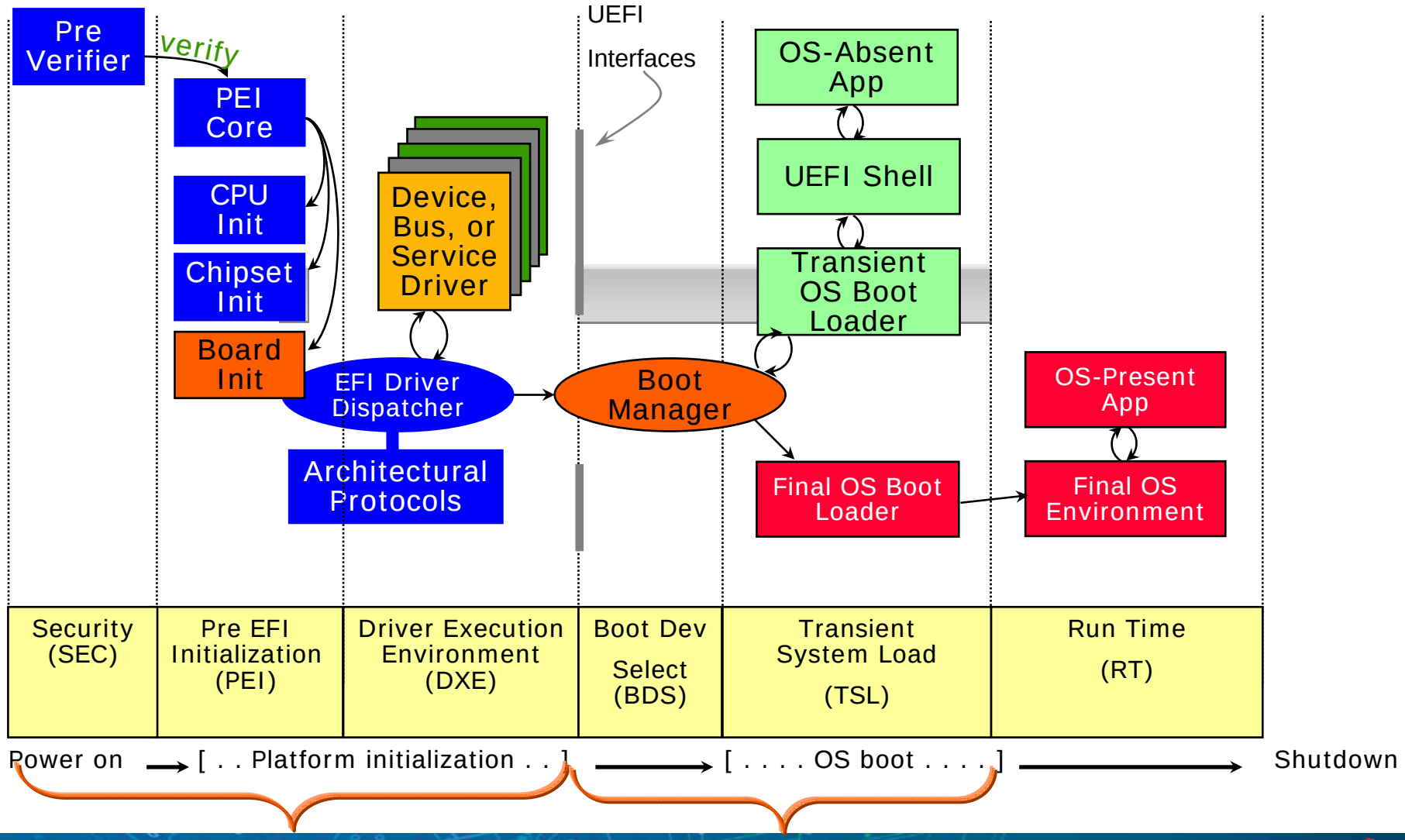
Tim Lewis
Phoenix Technology BIOS Architect
Vincent Zimmer
Principal Engineer at Intel

Session ID EFIS001

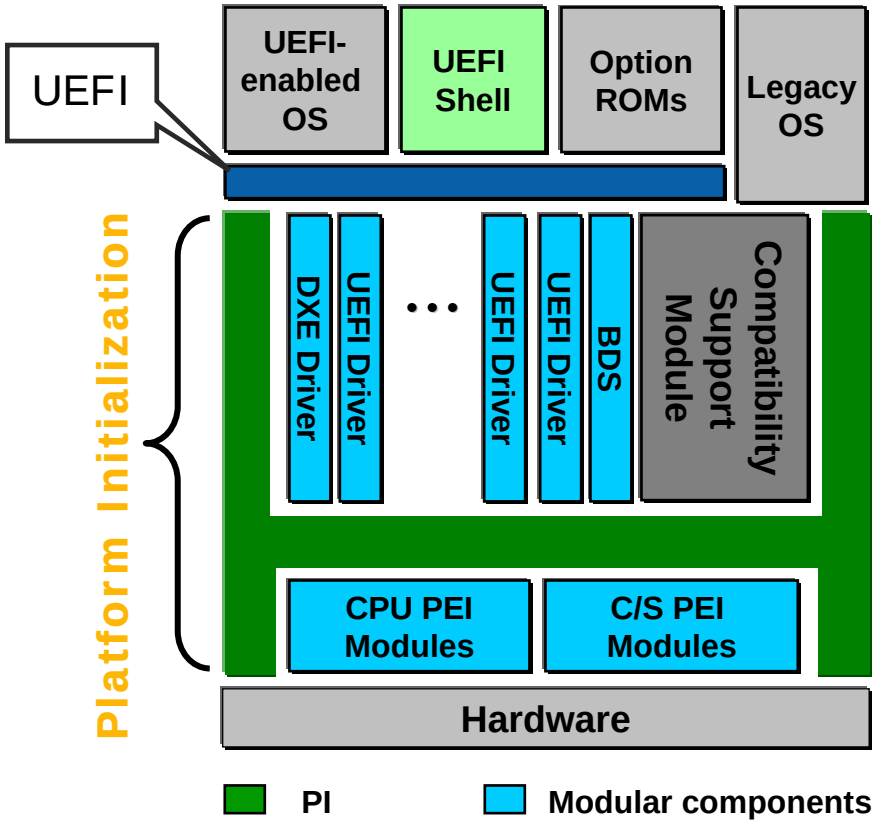
Agenda

- Unified Extensible Firmware Interface (UEFI) & Platform Initialization (PI) Overview
- PI 1.1
  - Building UEFI Platforms
- UEFI 2.2
  - Driver Signing, User Identity, IPv6
- UEFI Shell 1.0
  - Standard command-line app interface

Overall View of Boot Time Line



Platform Initialization

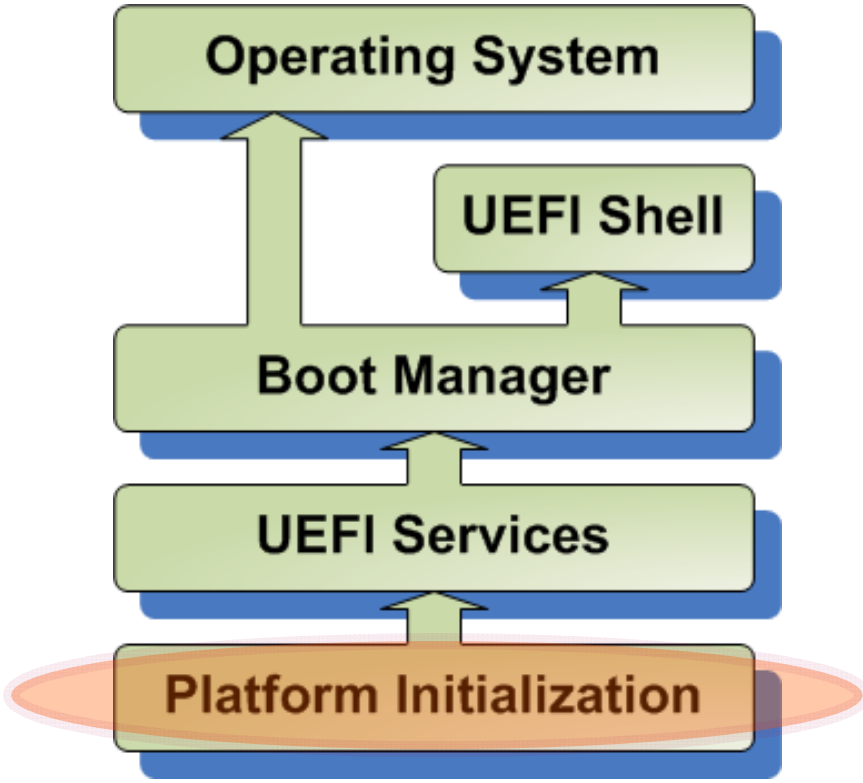


- UEFI: Unified Extensible Firmware Interface
  - a new model for the interface between the OS and platform firmware
- PI: Platform Initialization
  - Standardization: key to interoperability across implementations
  - Modular components like silicon drivers (e.g. PCI) and value-add drivers (security)
  - Preferred way to build UEFI

UEFI is Architected for Dynamic Modularity

Agenda

- UEFI and PI Overview
- **New in PI 1.1**
  - **PCI Host Bridge & Hot Plug**
  - **Multi-Processor Support**
  - **SMBIOS[1]**
  - **S3[2] Resume**
  - **SMM[3] & PMI[4]**
- UEFI 2.2
- UEFI Shell 1.0



[1] System Management BIOS (SMBIOS)
[2] Suspend to the system memory sleep state (S3)
[3] System Management Mode (SMM)
[4] Platform Management Interrupt (PMI)

PCI Details

- Host Bridge
  - Interoperability between chipset host-bridge support and PCI resource allocation
- Platform
  - Override behavior for non-compliant devices
- Hot Plug
  - Resource padding for PCI hot-plug

Standardization of the PCI infrastructure allows for chipset vendors to create a Host-bridge protocol instance that interacts with a generic resource allocator.

The Platform override and hot-plug allows for OEM policy to be created that admits for non-standard devices and resource-padding without interfering with the chipset or generic allocator

Seamless SI, OEM PCI resource management

Multiprocessor (MP) Details

The PI 1.1 Multiprocessor protocol provides a means by which to manage and run code on alternate processors in a system.

Useful for configuration (e.g., setting MTRR's on all CPU's, running diagnostics, etc) in the pre-OS

Leadership opportunities, such as parallelize pre-OS operations (e.g., multi-processor, concurrent memory testing)

```
#define EFI_MP_SERVICES_PROTOCOL_GUID \
 {0x3fdda605,0xa76e,0x4f46,{0xad,0x29,0x12,0xf4,0x53,0x1b,0x3d,0x08}}

typedef struct _EFI_MP_SERVICES_PROTOCOL {
 EFI_MP_SERVICES_GET_NUMBER_OF_PROCESSORS        GetNumberOfProcessors;
 EFI_MP_SERVICES_GET_PROCESSOR_INFO              GetProcessorInfo;
 EFI_MP_SERVICES_STARTUP_ALL_APS                 StartupAllAPs;
 EFI_MP_SERVICES_STARTUP_THIS_AP                 StartupThisAP;
 EFI_MP_SERVICES_SWITCH_BSP                      SwitchBSP;
 EFI_MP_SERVICES_ENABLEDISABLEAP                 EnableDisableAP;
 EFI_MP_SERVICES_WHOAMI                          WhoAmI;
} EFI_MP_SERVICES_PROTOCOL;
```

Consistent interoperability for MP

SMBIOS Tables

The PI 1.1 System Management BIOS (SMBIOS) protocol allows for creation of SMBIOS tables per the Desktop Management Task Force (DMTF) latest SMBIOS specification

Well-known API so different information producers/drivers can contribute information

```
#define EFI_SMBIOS_PROTOCOL_GUID \
{ 0x3583ff6, 0xcb36, 0x4940, 0x94, 0x7e, 0xb9, 0xb3, 0x9f, 0x4a, 0xfa, 0xf7 };

typedef struct _EFI_SMBIOS_PROTOCOL {
    EFI_SMBIOS_ADD          Add;
    EFI_SMBIOS_REMOVE       Remove;
    EFI_SMBIOS_ENUM         Enum;
    UINT8                   MajorVersion;
    UINT8                   MinorVersion;
} EFI_SMBIOS_PROTOCOL;
```

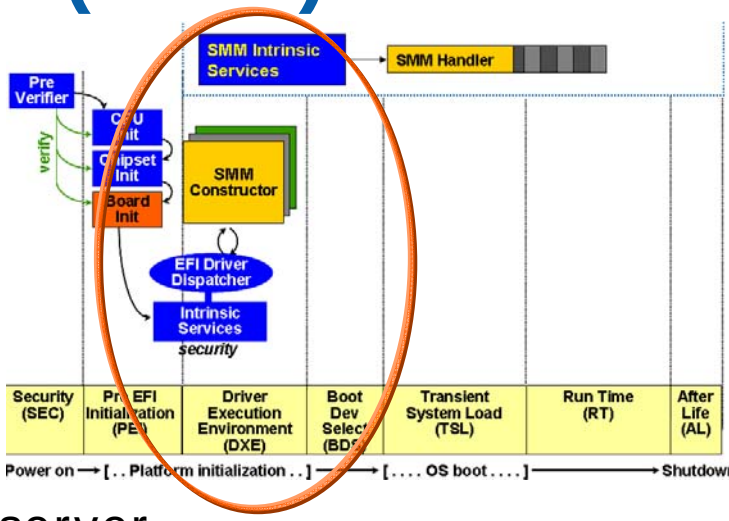
Open up SMBIOS creation to different producers

S3 Resume

- S3 is an ACPI sleep state where most hardware powered down but memory contents preserved.
- PI 1.1 S3 infrastructure allows DXE drivers to to create a script of operations that is replayed during the S3 boot path in order to re-initialize hardware prior to resuming control to the OS
- Boot Script Op Codes include the following

```
#define EFI_BOOT_SCRIPT_IO_WRITE_OPCODE                0x00;
#define EFI_BOOT_SCRIPT_IO_READ_WRITE_OPCODE           0x01;
#define EFI_BOOT_SCRIPT_MEM_WRITE_OPCODE               0x02;
#define EFI_BOOT_SCRIPT_MEM_READ_WRITE_OPCODE          0x03;
#define EFI_BOOT_SCRIPT_PCI_CONFIG_WRITE_OPCODE        0x04;
#define EFI_BOOT_SCRIPT_PCI_CONFIG_READ_WRITE_OPCODE 0x05;
#define EFI_BOOT_SCRIPT_SMBUS_WRITE_OPCODE             0x06;
#define EFI_BOOT_SCRIPT_STALL_OPCODE                   0x07;
#define EFI_BOOT_SCRIPT_DISPATCH                       0x08;
#define EFI_BOOT_SCRIPT_TERMINATE_OPCODE               0xFF;
```

System Management Mode (SMM) Uses



- Infrastructure to
  - Install SMM DXE drivers
  - Provide resource management of SMRAM
  - Allow for registration of different System Management Interrupt (SMI) sources
- Consumers of SMM
  - Provide RAS / Error Logging handlers for server
  - Work-around for chipset/CPU issues
  - OS-independent power-management
  - OEM/IBV value-add
- Platform Management Interrupt (PMI) is the SMM for Itanium® Processor
  - Infrastructure allows for registration of Machine Check Abort (MCA) and INIT

SMM Driver Model for features/workarounds

Overview of Differences – PI 1.0 Vs. Framework Components

	Component	Actions / Exceptions
✓	Compatibility	Do not access internals of the firmware files Do not use ReportStatusCode
✓	PEI File System	Minor change to the file header and firmware volume header
✓	PPI Updates	PCI PPI for Extended PCI-express New PPI – Terminate End of Temp Memory
✓	DXE Service Table	Removed Report Status Code service
✓	New Architectural Protocol	Capsule AP / QueryVariableInfo
✓	HOB definitions	More Firmware volume information Remove Capsule HOB definition

PI 1.0 Introduces Standards To Early Boot

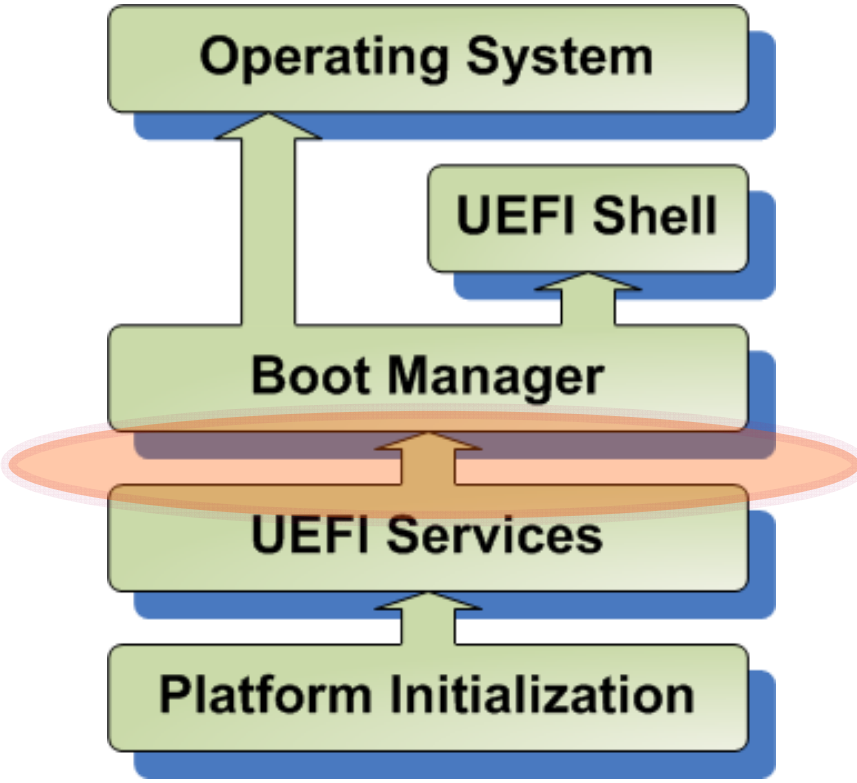
Overview of Differences – PI 1.1 Vs. Framework Components

	Component	Actions / Exceptions
✓	SMM	SMM driver model change. Port code
✓	S3	New execution requirements for native callbacks. Some interfaces removed
✓	PCI	New event. Should be able to enhance former implementation
✓	MP	Clean-up. Port to use new API
✓	DXE	Cleaned-up DXE to be UEFI 2.0 RT compatible
✓	SMBIOS table creation	Framework used data hub. This is a new API

PI 1.1 Expands PI 1.0 infrastructure w/ more building blocks

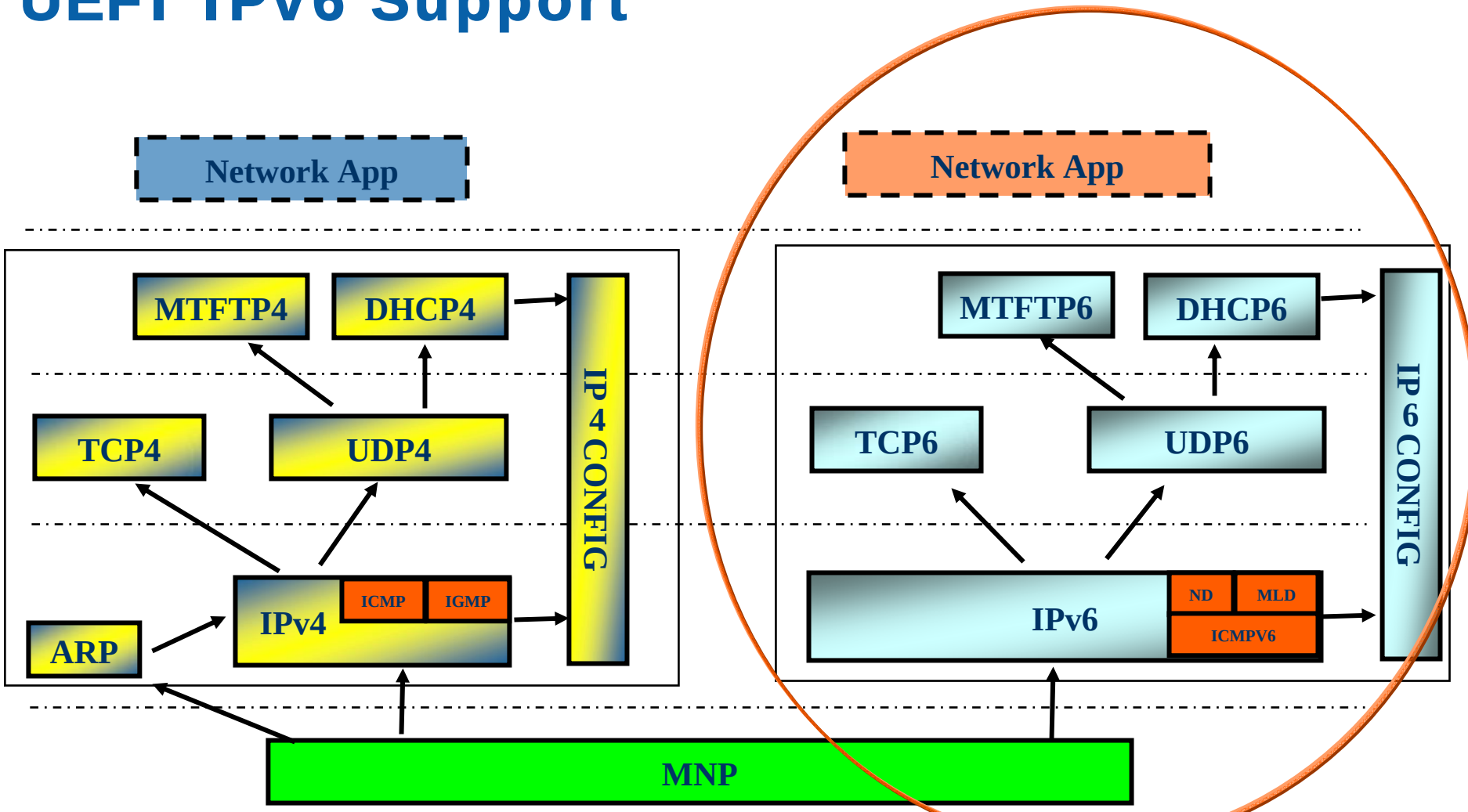
Agenda

- UEFI and PI Overview
- PI 1.1
- **UEFI 2.2**
  - **IPv6 Networking**
  - **Driver Signing**
  - **User Identification**
  - **User Interface Updates**
  - **Others…**
- UEFI Shell



Richer pre-OS capabilities for new scenarios/interoperability

UEFI IPv6 Support



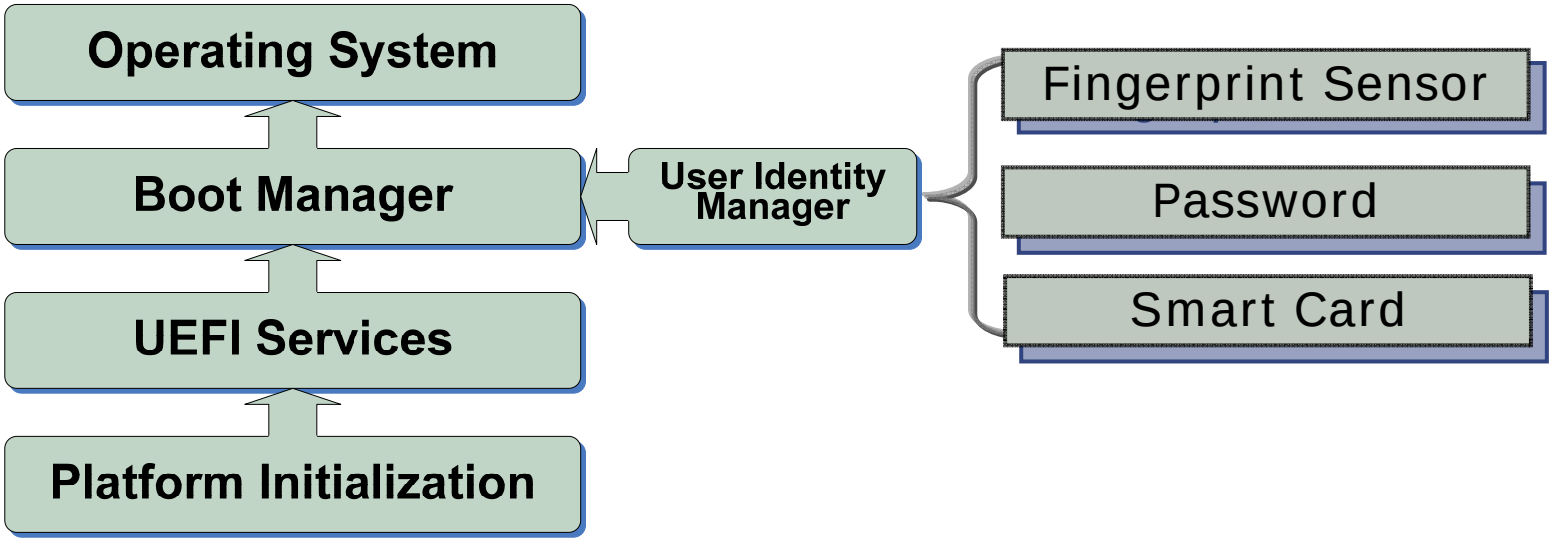
Building-blocks for IPV6 network bootstrap and applications

Driver Signing

- Expands the types of signatures recognized by UEFI
  - SHA-1, SHA-256, RSA2048/SHA-1, RSA2048/SHA-256 & Authenticode
- Standard method for configuring the “known-good” and “known-bad” signature databases.
- Provides standard behavior when execution is denied to provide policy-based updates to the lists.

Allow for more platform owner control of what occurs in UEFI pre-OS

UEFI User Identification



- Standard framework for user-authentication devices such as smart cards, smart tokens & fingerprint sensors.
- Uses UEFI HII to display information to the user.
- Introduces optional policy controls for connecting to devices, loading images and accessing setup pages.

Pre-boot authentication (PBA) framework and ability to assign rights to different users

UEFI Interface Updates

- New form type for support of non-UEFI configuration standards (e.g. DMTF)
- Translate question values between standards using new IFR operators:
  - EFI_IFR_GET, EFI_IFR_SET, EFI_IFR_READ, EFI_IFR_WRITE, EFI_IFR_MAP
- Setup Page Security
  - New IFR operator suppresses or disables forms and questions based on security permissions
- HII Animation
  - Data format for standard GIF-style animations.

Supports Industry Configuration, Security & Splash Screen User-Interface Requirements

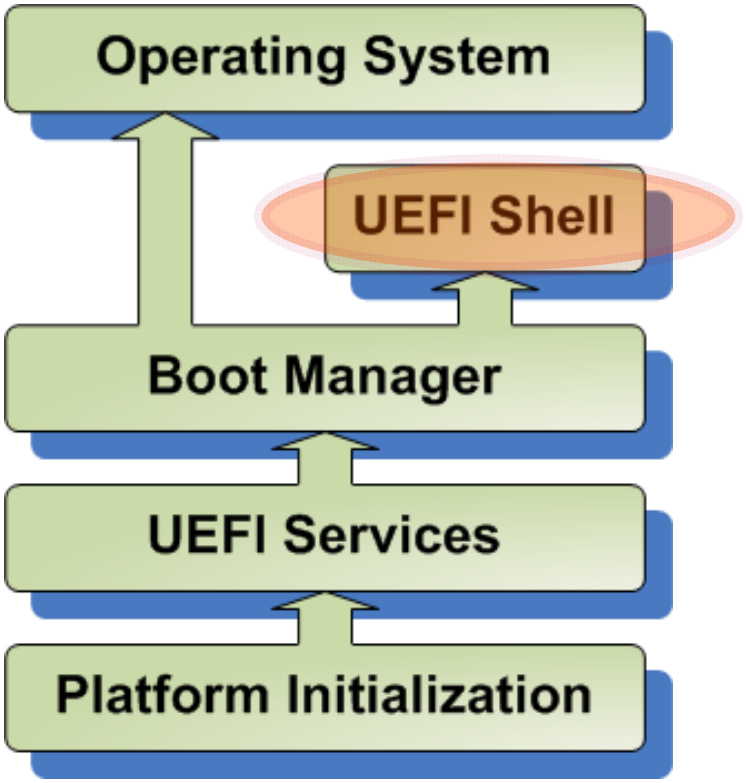
Miscellaneous

- EFI_ATA_PASS_THRU Protocol
  - Gives direct access to ATA devices
- UEFI Driver Health
  - Allow for a driver to fix/re-configure (e.g. rebuild RAID set)
- ABI Updates/Clarifications
  - Floating Point/MMX/XMM
  - 16-Byte stack alignment
- EFI_LOAD_FILE2 Protocol
  - Loads non-boot-option EXEs (PCI option ROMs & apps)
  - Modifies LoadImage() behavior
- EFI_LOADED_IMAGE Protocol
  - Associates entire device path with EXE image

Independent Hardware Vendor (IHV) Driven Additions

Agenda

- UEFI and PI Overview
- PI 1.1
- UEFI 2.2
- **UEFI Shell**
  - **Standardized CLI**
  - **Sits directly on UEFI firmware**
  - **Optimized for small footprint**
  - **Supports rich applications**
  - **Standard scripting & API**



A consistent way to move from DOS or proprietary CLI

UEFI Shell vs. EFI Shell

Small Size Profiles	0: Shell API only 1: Basic scripting support 2: File support cmds (cd,cp,mv) 3: Adds interactive CLI + profiles
New Shell API	Smaller executable size Expose previously hidden shell capabilities Execution break support
Enhanced Scripting	Compatible with existing scripts Added input redirection & piping Enhanced if command
Shell Commands	Standardized existing usage Updated for UEFI 2.1+ Standardized argument usage and output

UEFI Shell Standard Commands

Level/Profile	Commands
Level 0	None
Level 1	else, endfor, endif, exit, for, goto, if, shift
Level 2	attrib, cd, cp, date, del, load, map, mkdir, mv, reset, rm, set, time, timezone, touch
Level 3	alias, cls, dir, help, ls, mode, pause, type, ver
UEFI Debug Profile	bcfg, comp, dblk, dmem, dmpstore, echo, edit, eficompress, efidecompress, hexedit, loadpcirom, mem, memmap, mm, pci, sermode, setsize, smbiosview
UEFI Network Profile	ipconfig, ping
UEFI Driver Profile	connect, devices, devtree, dh, disconnect, drivers, drvcfg, drvdiag, openinfo, reconnect, unload

Choose the amount of shell capability for market segment/platform type

UEFI Shell API Overview

Group	Functions
File Manipulation	CreateFile, DeleteFile, ReadFile, WriteFile, DeleteFileByName, CloseFile, FindFiles, FindFilesInDir, GetFilePosition, SetFilePosition, GetFileInfo, SetFileInfo, FreeFileList, OpenFileByName, OpenFileList, OpenRoot, OpenRootByHandle, GetFileSize, RemoveDupInFileList
Mapping, Alias & Environment Variables	GetMapFromDevicePath, GetFilePathFromDevicePath, GetDevicePathFromFilePath, GetDevicePathFromMap, SetMap, SetAlias, GetEnv, SetEnv, GetCurDir, SetCurDir
Launch Application Or Script	Execute, BatchIsActive, IsRootShell
Miscellaneous	GetPageBreak, EnablePageBreak, DisablePageBreak, GetHelpText, GetDeviceName

EFI_SHELL_PROTOCOL is installed on application image handle

Summary / Call to Action

- PI 1.1 for silicon and platform enabling interoperability
  - OEM's, silicon vendors and IBV's
- UEFI 2.2 for pre-boot security and IPv6 networking enhancements
  - OEM's, IBV's, OSV's, ISV's – build in support for and take advantage of these new capabilities
- UEFI Shell 1.0 for a DOS replacement, scripting and configuration
  - OEM's, ODM's, ISV's, IHV's, IBV's…. – start leveraging advantages of a standardized pre-OS shell

All – get more information at UEFI.org

Additional UEFI Sessions Room No: 2007:

Session	EFI #	Company	Time
Emerging Unified Extensible Firmware Interface (UEFI) Capabilities	S001	Intel / Phoenix	Done
Intel Framework Customization for Optimized Platform Boot Performance	S002	Intel	Wed 13:40
Microsoft Windows* on Unified Extensible Firmware Interface (UEFI)	S004	Microsoft	Wed 14:40
Debugging Under (UEFI): Addressing DXE Driver Challenges	S003	Insyde SW	Thur 10:10
Towards a Common Firmware Update Mechanism	S005	Intel/ IBM	Thur 11:10
Q&A open forum - Chalk Talk	C001	Intel / UEFI VP	Thur 14:40

- More web based info: www.tianocore.org www.uefi.org
- www.intel.com/technology/framework
- Technical book from Intel Press: "Beyond BIOS: Implementing the Unified Extensible Firmware Interface with Intel's Framework" www.intel.com/intelpress
- Show case Booth #221 UEFI and Booth #531 Phoenix
- Win a iPod touch at the EFIS005 session Thursday 11:10 Room 2007

Session Presentations - PDFs

The PDF for this Session presentation is available from our IDF Content Catalog at the end of the day at:

www.intel.com/idf

Or

https://intel.wingateweb.com/US08/scheduler/public.jsp

Please Fill out the Session Evaluation Form

Place form in evaluation box at the back of session room

Thank You for your input, we use it to improve future Intel Developer Forum events

Q&A

Legal Disclaimer

- INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL® PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. INTEL PRODUCTS ARE NOT INTENDED FOR USE IN MEDICAL, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS.
- Intel may make changes to specifications and product descriptions at any time, without notice.
- All products, dates, and figures specified are preliminary based on current expectations, and are subject to change without notice.
- Intel, processors, chipsets, and desktop boards may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request.
- Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance.
- Intel, Intel Inside, and the Intel logo are trademarks of Intel Corporation in the United States and other countries.
- *Other names and brands may be claimed as the property of others.
- Copyright © 2008 Intel Corporation.

Risk Factors

This presentation contains forward-looking statements that involve a number of risks and uncertainties. These statements do not reflect the potential impact of any mergers, acquisitions, divestitures, investments or other similar transactions that may be completed in the future. The information presented is accurate only as of today's date and will not be updated. In addition to any factors discussed in the presentation, the important factors that could cause actual results to differ materially include the following: Demand could be different from Intel's expectations due to factors including changes in business and economic conditions, including conditions in the credit market that could affect consumer confidence; customer acceptance of Intel's and competitors' products; changes in customer order patterns, including order cancellations; and changes in the level of inventory at customers. Intel's results could be affected by the timing of closing of acquisitions and divestitures. Intel operates in intensely competitive industries that are characterized by a high percentage of costs that are fixed or difficult to reduce in the short term and product demand that is highly variable and difficult to forecast. Revenue and the gross margin percentage are affected by the timing of new Intel product introductions and the demand for and market acceptance of Intel's products; actions taken by Intel's competitors, including product offerings and introductions, marketing programs and pricing pressures and Intel's response to such actions; Intel's ability to respond quickly to technological developments and to incorporate new features into its products; and the availability of sufficient supply of components from suppliers to meet demand. The gross margin percentage could vary significantly from expectations based on changes in revenue levels; product mix and pricing; capacity utilization; variations in inventory valuation, including variations related to the timing of qualifying products for sale; excess or obsolete inventory; manufacturing yields; changes in unit costs; impairments of long-lived assets, including manufacturing, assembly/test and intangible assets; and the timing and execution of the manufacturing ramp and associated costs, including start-up costs. Expenses, particularly certain marketing and compensation expenses, vary depending on the level of demand for Intel's products, the level of revenue and profits, and impairments of long-lived assets. Intel is in the midst of a structure and efficiency program that is resulting in several actions that could have an impact on expected expense levels and gross margin. Intel's results could be impacted by adverse economic, social, political and physical/infrastructure conditions in the countries in which Intel, its customers or its suppliers operate, including military conflict and other security risks, natural disasters, infrastructure disruptions, health concerns and fluctuations in currency exchange rates. Intel's results could be affected by adverse effects associated with product defects and errata (deviations from published specifications), and by litigation or regulatory matters involving intellectual property, stockholder, consumer, antitrust and other issues, such as the litigation and regulatory matters described in Intel's SEC reports. A detailed discussion of these and other factors that could affect Intel's results is included in Intel's SEC filings, including the report on Form 10-Q for the quarter ended June 28, 2008.

Backup Slides

Different Levels Of Shell Support

- Different levels of support for different usage scenarios and space constraints:
  - Level 0: No CLI. No shell commands. Only shell API.
  - Level 1: Adds basic scripting support
  - Level 2: Adds basic commands (cd, cp, mv)
  - Level 3: Adds interactive CLI
- Shell support level can be detected using environment variable.
- Beyond level 3, additional command “profiles” are defined for debug, networking and driver support.

Support levels save space in low-resource environments

UEFI Shell Scripts

- Text files with **`.nsh`** extension are shell scripts
- Supports command-line arguments using positional parameters **`%0`** - **`%9`** and **`shift`**.
- Supports standard script commands
  - **`if/else/endif`**
  - **`goto`**
  - **`for/endfor`**
  - **`echo`**
  - **`exit`**
- Supports input & output redirection & pipes.
  - Can also redirect to/from environment variables!