Unified Extensible Firmware Interface (UEFI) Implementation Guidelines

Andrew Ritz
Development Manager
Windows Kernel Team
Microsoft Corporation

Key Takeaways

- Be a leader in advancing 64-bit computing
- Adopt best practices and new tools
- Let's partner on new hardware directions

- Understand importance of Unified Extensible Firmware Interface (UEFI)
- UEFI in industry and Microsoft plans
- Understand how to build UEFI platforms which are Windows compatible

Agenda

- 2 key audiences, 2 focuses
- Interested in UEFI and Microsoft's position?
  - Understand what UEFI is
  - Understand Microsoft firmware goals
  - UEFI timeline
  - Understand Microsoft roadmap
- Implementing a UEFI platform?
  - Explicit guidance on how to construct a system
  - Firmware and Deployment considerations

What Is UEFI?

- UEFI is a firmware interface specification
  - Standardized mechanism to bootstrap Operating System (OS) launch
  - Next-generation replacement for BIOS-based firmware
- UEFI is a platform independent specification
  - Platform specifics defined for Itanium, x64, x86

What Is UEFI?

- UEFI runs in long-mode on x64
  - Great environment for using modern programming techniques and tools
  - By comparison, BIOS is a 16-bit real-mode environment
- UEFI contains formally architected extensibility model
  - Adding in driver support is well-architected
  - Compared to ad-hoc extensibility in BIOS

How UEFI And ACPI Relate

- UEFI complements Advanced Configuration and Power Interface (ACPI) firmware
- ACPI firmware used at runtime, UEFI mainly used during bootstrap
- Limited runtime usage of UEFI firmware

Why Is UEFI Important?
Engineering ease

- Significant benefits to UEFI approach
  - Non-recoverable Engineering cost (NRE) is lessoned with UEFI
  - BIOS has shown its age
  - Innovation still possible with BIOS, but technical limitations make this very difficult
- Changing how ecosystem operates
  - Clean extensibility model changes how systems are integrated
- Significant industry momentum
- A once every 20 years opportunity

Why Is UEFI Important?
User Visibility

- UEFI doesn't greatly effect visible feature set of a platform
  - UEFI dictates internals of how a system is put together
  - Simplifies design of pre-OS components
- Consumers shouldn't need to understand this part of the system
  - Users rarely interact with firmware

Windows Firmware Roadmap
Goals

- Enable mainstream 64-bit computing
- UEFI is a solid technology to bet on
  - Transition away from BIOS firmware to UEFI firmware over time
  - Proactive long-term investment in UEFI
- Achieve firmware independence
  - Consistent with Windows portability goals
  - Requirement for transition period
- Avoid adding complexity to customers during firmware transition

Windows Engineering Methodology
Investing in UEFI

- Microsoft taking advantage of benefits of UEFI internally
- Future releases may include exclusive UEFI scenarios
- Shifting to UEFI first approach
  - Architect, design, build on UEFI
  - Where relevant, port feature to BIOS

Evolution Of UEFI
(1 of 4)

Late 1990's

- Conception of EFI
- Focused alternative to BIOS on Itanium
- Also intended as replacement for BIOS on x86
- Microsoft active in EFI 1.0 specification

2001

- Windows XP 64-bit edition: introduces EFI 1.02 support for Itanium
- Work begins on next-generation Windows Boot Environment
  - Microsoft goal to support EFI on x86

Evolution Of UEFI
(2 of 4)

2003

- Development for Windows XP Professional x64 Edition
- X64 established as Windows strategic direction
  - Need EFI solution to support X64 and gain industry momentum for X64 and EFI

Fall 2004

- Demo of Windows 32-bit EFI boot at Intel Developer Forum
  - Establishes viability of Windows Boot Environment and EFI

Evolution Of UEFI
(3 of 4)

2005

- Windows XP Professional x64 Edition released
- UEFI forum established
  - Goal to define support for x64
  - Goal to drive EFI adoption through ecosystem

Spring 2006

- UEFI 2.0 released
- Demo of Windows x64 UEFI boot at Intel Developer Forum
- Windows plug-fest
  - Ecosystem maturing but not yet ready

Evolution Of UEFI
(4 of 4)

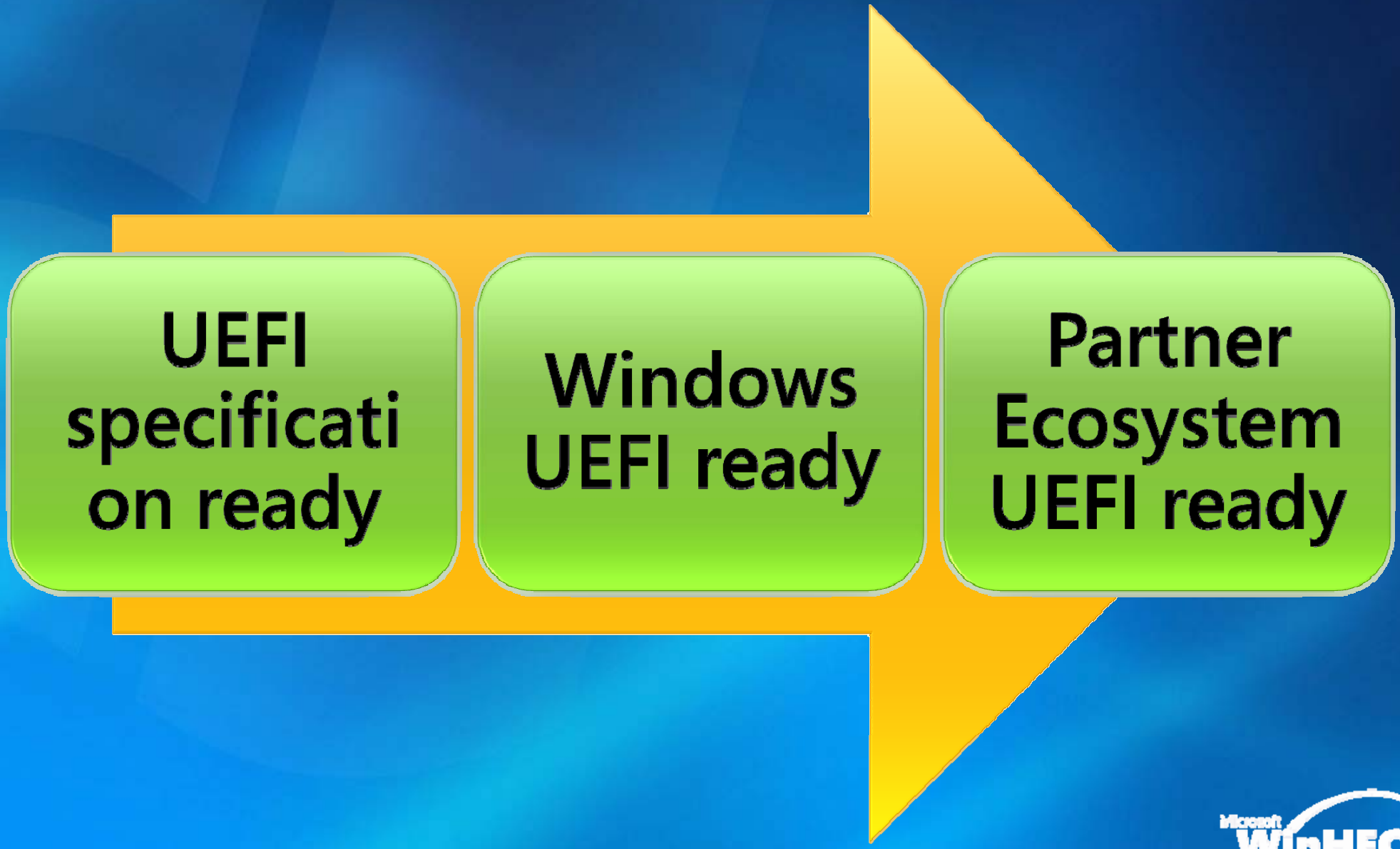
Winter 2006

- Windows Vista released
- UEFI plug-fest
  - More platform support, parity with BIOS
  - Emerging driver support

Spring 2007

- Windows Server codename “Longhorn” UEFI plug-fest
  - Ecosystem maturing
  - Various mature base implementations
  - UEFI 2.1 released

UEFI Solutions Emerging…



Windows Support For UEFI

- Windows Server Longhorn and Windows Vista introduce native UEFI 2.0 support on all 64-bit platforms
  - Emergence of x64 provides an inflection point for transition to UEFI
  - No support for 32-bit platforms planned

Windows Support For Itanium

- Windows supports existing EFI 1.1-based Itanium systems
- Some elements of UEFI 2.1 required depending on platform WHEA integration design choice

What Does UEFI Support Include?

- 64-bit Client and Server support in the Windows Server Longhorn timeframe
  - Parity support for all BIOS-based platform scenarios on UEFI platforms
  - Native deployment and boot on UEFI platforms

Guidelines For Building A UEFI Platform

Principles For Firmware Requirements

- Parity support for all scenarios on BIOS and UEFI systems
- Support UEFI on mainstream x64 systems
- Allow boot of older operating systems (e.g., Windows XP) on UEFI platforms during transition
  - UEFI does support a firmware compatibility layer to support boot of prior BIOS-based operating systems
- Windows Server Longhorn and Windows Vista start the clock ticking for dropping BIOS backwards compatibility

Required Firmware Elements

- Must support UEFI 2.0 specification
  - Elements of UEFI 2.1 required
- Future-proofed: Windows does not explicitly check for newer revisions
- Support Windows boot, Windows installation and Windows compatibility requirements
- ACPI 2.0+ runtime firmware support

UEFI Firmware Installation Requirements

- UEFI-based installation requires boot via UEFI
- Must support appropriate boot service protocols for installation mechanism
- Must support runtime variable services
- Must support GPT partitioning scheme

DVD Media Requirements

- Windows uses same media for UEFI and BIOS installation
  - Windows uses UDFS bridge format for DVD media
  - Windows uses El Torito multiple boot catalog support
  - Windows OEM Preinstall Kit (OPK) and Windows Admin Installation Kit (AIK) include updated version of cdimage.exe that supports creation of multiple boot catalog image
  - Details provided in deployment guide

DVD Firmware Requirements

- UEFI Firmware must support El Torito multiple boot catalog support for DVD boot
  - UEFI firmware must detect catalog entry with 0xEF platform tag
  - UEFI firmware boot manager must execute \EFI\BOOT\BOOTX64.EFI from catalog
  - Must ignore catalog entry with 0x0 platform tag
- Platforms with BIOS support must support same media

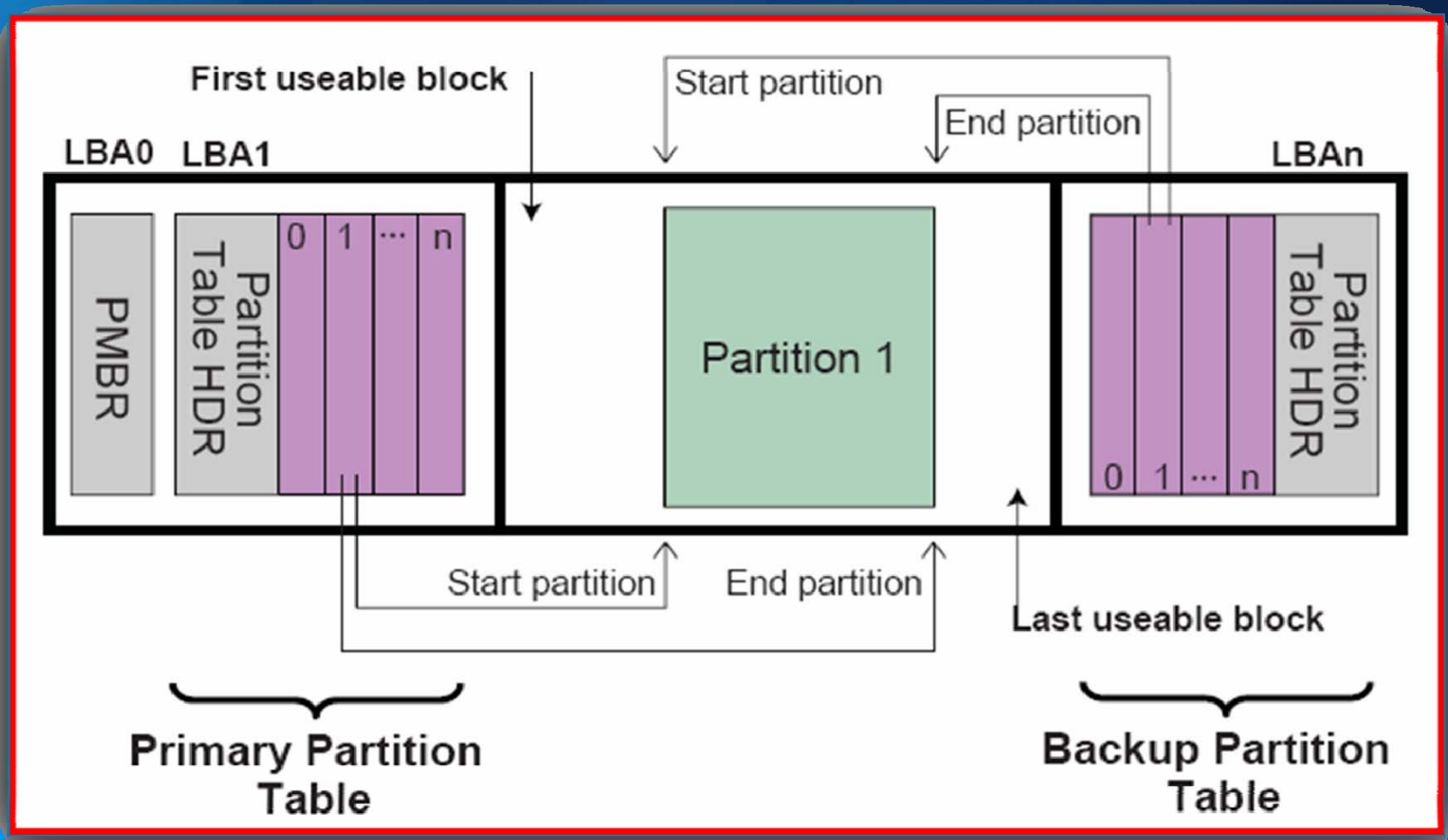
Network Boot Requirements

- Must send out PXE broadcast with correct client system architecture tag (0x00 0x07)
- Must support TFTP download
  - Initial download is Windows Boot Manager

Understanding GPT

- GUID Partition Table (GPT) partitioning scheme required for UEFI boot
  - GPT proven scheme from Itanium deployments
  - GPT already supported for data disks on Windows Server 2003 SP1
- Key features of GPT
  - No boot code on disk
  - Robustness
  - Flexibility: extensible partition types and counts
  - Modern: Extended disk sizes (>2TB)

GPT Disk Partitioning



Windows GPT Requirements
Required Partitions

- EFI system partition (ESP)
  - Minimum size 200MB
  - Formatted with FAT32 file system
- Microsoft reserved partition (MSR) – 128 MB
- Windows Operating System (OS) partition
  - Formatted with NTFS file system

Firmware Boot Requirements
Required Protocols

- Firmware must implement the following protocols for disk boot
  - Block I/O protocol and Device Path protocol
- Firmware must implement the following protocols for input/output
  - Simple Input protocol
  - Graphics output protocol
  - Simple Text output protocol
  - Firmware must implement the following protocol for BitLocker™ support
    - EFI TCG protocol

Understanding Window Graphics Usage

- Windows OS Loader places system into graphics mode
  - Required for localized text to be rendered
  - Windows prefers graphics output protocol (GOP)
  - Windows supports EFI 1.1 UGA protocol but UGA is deprecated in UEFI 2.0 specification
  - Better long-term choice is GOP support
  - Firmware may not manipulate frame buffer after mode is set by OS Loader
- Windows requires 1024x768 or 800x600 resolution with
  32-bit color
  - If neither supported, Windows reverts to simple text mode and English

Understanding Graphics Output
Runtime compatibility

- GOP does not support runtime calls
  - Windows sets the mode in OS Loader
  - Preserve mode and frame buffer after ExitBootServices() and until Windows performance driver takes over
- For Windows Server Longhorn VGA support still requires int10h support
  - Required for many video cards
  - Wish to loosen this restriction in future releases
  - Specify the VGA not present ACPI flag on server systems without video card

Runtime Firmware Requirements

- Firmware must implement EFI variable services
- Storage Sizing recommendations
  - Windows limits use of variable services for boot settings
  - Most settings stored on ESP in Windows BCD store

Variable Services And WHEA

- WHEA Error record persistence required for Windows Server Longhorn
- Multiple design alternatives exist
  - Using EFI variable services requires UEFI 2.1 variable services compliance
- Storage Sizing
  - 1KB minimum size required for x64 WHEA Error records
  - 100KB minimum size required for Itanium WHEA Error records
- More details in WHEA presentation

ACPI Sleep State Requirements

- Windows Vista requires S3 and S4 support
- Windows Server Longhorn requires S4 support
- Firmware must ensure that physical memory is consistent across S4 sleep transitions
  - Size and location must both be maintained
  - Required to restore physical memory across transition
  - Windows will fail to resume from S4 if these conditions not satisfied
  - Physical memory map retrieved via GetMemoryMap() interface

BCD Deployment

- Windows stores boot settings in Boot Configuration Data (BCD) store
- Common abstraction for UEFI and BIOS platforms
  - Windows Server 2003 had different mechanisms and no abstraction
- Also allows access to UEFI NVRAM variables
- Access to BCD store available via
  - BCD WMI provider
  - BCDEdit
- Excellent web resources available

BCD Deployment

- BCD store is created on ESP
  - Consider backup plan for BCD store
- Boot settings static during installation
  - Modify boot settings post setup via scripting

UEFI And Image Deployment

- Image Deployment for UEFI is similar to image deployment on BIOS
- Capture the Windows partition image
- Run setup.exe to deploy image on target
  - All details taken care of for you
- If you deploy offline, you must deploy Windows partition and ESP
- Extra steps detailed in whitepaper

A Note On Bootstrap Extensibility

- Ad-hoc mechanisms for extensibility not supported with UEFI
- Windows supports custom bootstrap actions
- UEFI boot manager supports other extensibility
- Please talk with Microsoft about how to best integrate your value-add software
  - Opportunities to enhance UEFI specification

Call To Action

- Understand industry momentum around UEFI
  - Consider impact of UEFI for your company
  - Understand why Microsoft loves UEFI
- Start building Windows UEFI platforms
  - Understand requirements for Windows UEFI platforms
- Get engaged with Microsoft and UEFI
  - Familiarize with Windows UEFI support
  - Run the UEFI SCT
  - Get involved in Windows and UEFI plug-fests

Additional Resources

- Web Resources
  - Portal: http://www.microsoft.com/whdc/system/platform/firmware/default.mspx
  - UEFI Requirements: http://www.microsoft.com/whdc/system/platform/firmware/uefireg.mspx
  - Custom Boot Actions
    - http://www.microsoft.com/whdc/system/platform/firmware/OEMBoot_Vista.mspx
- Related Sessions
  - SVR-T326 WHEA Systems: Design and Implementation
- Questions and Comments:

Winboot @ microsoft.com

FACT:
36% of tested consumers think Logo'd devices provide a premium experience

Microsoft®

Your potential. Our passion.™

Ł 533:#P lfurvriw#Frusrudwrq1#Do#ulj kw#lnvhuyhg1#P lfurvriw#Z lggrzv/#Z lggrzv#Xlwd#dqg#rwkhu#surgxfw#qdp hv#duh#ru#p d|#eh#hnj lwhuhg#wudghp dunv#dqg2ru#wudghp dunv#q#wkh#KVI#dqg2ru#rwkhu#frxqwulhv1 Wkh#qirup dwlrq#khuhlq#v#ru#qirup dwlrqdd#xusrvhv#rqo|#dqg#hsuhvhqw#wkh#fxuhqw#yhz#ri#P lfurvriw#Frusrudwrq#dv#ri#wkh#gdwh#ri#wkh1v#suhvhqwdwlrq1##Ehfdxvh#P lfurvriw#p xv#uhvsrqg#wr#Ekdqj lqj#p dunhw#frqglwrqv/#w# vkrxog#grw#eh#qwhusuhwhg#wr#eh#3#frp p lw#hqw#rq#wkh#sduw#ri#P lfurvriw#dqg#P lfurvriw#fdqqrw#jxdudqwhh#wkh#liffxuf|#ri#dq|#qirup dwlrq#surylghg#diwu#wkh#gdwh#ri#wkv#suhvhqwdwlrq1# P IFUR VR IW#P DNHV#Q R #Z DUUDQ WLHV/#H[SUHVV/#P SOHG #R U#WDWXWR U\/#D V WR #WKH#Q IR UP DWIR Q #Q #WKIV#SUHVHQ WDWIR Q 1


Microsoft
WinHEC
2007