UEFI and Windows

September 14, 2009

Abstract

This paper provides an overview of Unified Extensible Firmware Interface (UEFI) technology and describes support for UEFI in the Windows® family of operating systems. It highlights the capabilities that UEFI offers and points out differences between UEFI and basic input/output system (BIOS) firmware.

This information applies for the following operating systems unless explicitly noted otherwise:

 Windows 7
 Windows Vista® SP1 and later

 Windows Server® 2008 R2
 Windows Server 2008
 Windows Server 2003

References and resources discussed here are listed at the end of this paper.

The current version of this paper is maintained on the Web at:

<http://www.microsoft.com/whdc/system/platform/firmware/UEFI_Windows.mspx>

Disclaimer: This is a preliminary document and may be changed substantially prior to final commercial release of the software described herein.

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This White Paper is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

Unless otherwise noted, the example companies, organizations, products, domain names, e-mail addresses, logos, people, places and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, email address, logo, person, place or event is intended or should be inferred.

© 2006–2009 Microsoft Corporation. All rights reserved.

Microsoft, Windows, Windows Server, and Windows Vista are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

Document History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date | Change |  |  |  |
| April 20, 2006 | First publication with title of “EFI and Windows Vista” |
| July 25, 2008 | Updated for Windows Server 2008 and Windows Vista SP1 |
| September14, 2009 | Added more detail to UEFI features and Windows 7 support |

Contents

[Introduction 4](#_Toc240688408)

[Types of PC Software 4](#_Toc240688409)

[PC Firmware Types 5](#_Toc240688410)

[UEFI Overview 6](#_Toc240688411)

[Compatibility with Earlier BIOS 6](#_Toc240688412)

[Support for Large Disks 6](#_Toc240688413)

[CPU-Independent Architecture 6](#_Toc240688414)

[CPU-Independent Drivers 7](#_Toc240688415)

[Flexible Pre-OS Environment 7](#_Toc240688416)

[Modular Design 7](#_Toc240688417)

[Windows Support for UEFI 7](#_Toc240688418)

[Windows Features on UEFI 8](#_Toc240688419)

[Current Windows-Specific UEFI Highlights 8](#_Toc240688420)

[Multicast Deployment 8](#_Toc240688421)

[Fast Boot and Resume from Hibernate 8](#_Toc240688422)

[Future UEFI Capabilities Under Investigation 9](#_Toc240688423)

[Rootkit Prevention 9](#_Toc240688424)

[Network Authentication 9](#_Toc240688425)

[UEFI Summary 9](#_Toc240688426)

[Resources 10](#_Toc240688427)

# Introduction

The Unified Extensible Firmware Interface (UEFI) specification is the product of an industry-wide effort to improve software interoperability and address the limitations imposed by earlier firmware designs.

This paper introduces firmware concepts in general and UEFI in particular for business planners, system builders, and anyone curious about UEFI technology. For technical details about UEFI implementation, see “UEFI Support and Requirements for Windows Operating Systems,” which is listed in the “Resources” section at the end of this document.

# Types of PC Software

The software that runs on a PC can be classified into vertically integrated components, as shown in Figure 1:



Figure 1. Basic system components

* **Firmware**

Firmware is the hardware-specific code that directs the hardware’s response to commands from higher level software. Firmware is typically embedded in non-volatile storage that is directly attached to a hardware device such as a motherboard, but firmware also resides on optional device hardware such as a graphics adapters and storage controllers. Basic input/output system (BIOS) and UEFI are both examples of firmware.

Firmware provides the first set of instructions that run during the boot process. After the firmware finishes detecting hardware and initializing the system, it passes control to a boot application such as an operating system (OS) or a utility that runs before the OS is loaded (sometimes called a pre-OS utility).

* **Operating System Software**

An operating system acts as an interface between hardware and higher-level software. The Windows® operating system coordinates background activity and manages shared hardware and software resources among multiple applications.

The primary Windows boot application is the boot manager (bootmgr). The boot manager uses services that the PC firmware provides to access key hardware resources such as storage devices, graphics devices, and system memory, so that it can start to load the rest of the operating system.

During the first stages of the boot process, all operating systems use services that the firmware makes available, to access hardware and load other operating system components. Initially, Windows uses firmware services to load early system components, but after device drivers are loaded, Windows no longer interacts directly with platform firmware services for hardware access. From that point forward the system relies primarily on high-performance device drivers rather than firmware services.

Windows limits the use of firmware services as much as possible to help ensure system reliability. Although most interaction with the firmware occurs during the boot process, Windows can also interact with firmware at run time.

* **Application Software**

After the operating system has prepared a suitable environment, application software uses a standard set of system-supplied interfaces to perform more specific tasks independent of the specific details of system management. Users can install and remove applications without modifying the core operating system or system firmware.

# PC Firmware Types

Every PC is preloaded with firmware, which is typically stored in programmable read-only memory (PROM), electrically erasable programmable read-only memory (EEPROM), or flash memory. PCs typically use one of the following types of firmware:

* Basic input/output system (BIOS). BIOS firmware descends directly from the earliest personal computers in the 1970s. Although BIOS is still the most prevalent firmware type, BIOS is increasingly limited because it supports only 16-bit processor mode and 1 megabyte (MB) of addressable memory space. Adding support for new hardware on BIOS systems is also relatively complex because no universal BIOS standard exists and BIOS implementations can vary from one vendor to the next.
* Unified Extensible Firmware Interface (UEFI). As the limitations of BIOS firmware became more apparent, the PC industry recognized the need for a more flexible standard. The first Extensible Firmware Interface (EFI) specification was completed in the late 1990s, and in 2005 the Unified EFI Forum was formed to standardize and promote UEFI implementations. Over 140 leading technology companies currently participate in the UEFI forum, led by AMD, AMI, Apple, Dell, HP, IBM, Insyde, Intel, Lenovo, Microsoft®, and Phoenix Technologies.

In contrast to BIOS, UEFI defines a set of boot and runtime services that have standard syntax and semantics for interfaces and data structures. This means that all UEFI implementations behave essentially the same way, making it possible to test and develop standard drivers and applications. This greatly improves interoperability, reduces the complexity of supporting new hardware, and helps computer manufacturers to update and maintain firmware more rapidly.

# UEFI Overview

In addition to better interoperability, UEFI firmware provides several technical advantages:

* Compatibility with operating systems that support only BIOS
* Ability to boot from large disks
* CPU-independent architecture
* CPU-independent drivers
* Flexible pre-OS environment
* Modular design

## Compatibility with Earlier BIOS

Almost all current UEFI implementations include a Compatibility Support Module (CSM) that emulates earlier BIOS. Therefore, systems with UEFI firmware can boot operating systems that are UEFI-aware and older operating systems that support only BIOS. This feature provides flexibility and compatibility for end users.

## Support for Large Disks

Starting with the original IBM PCs in the early 1980s, PCs have used the master boot record (MBR) partitioning scheme to describe hard disk partitions. BIOS systems with MBR disks use 32-bit values to describe the starting offset and length of a partition. This method allows a maximum disk size of roughly 2.2 terabytes (TB) (232 sectors \* 512 bytes per sector), and a maximum of four primary partitions.

UEFI supports a more flexible partitioning scheme called GUID Partition Table (GPT). GPT disks use 64-bit values to describe partitions, which allows a maximum disk size of roughly 16.8 million TB and over 100 primary hard disk partitions.

## CPU-Independent Architecture

BIOS firmware is specific to the original Intel x86 processor architecture, and it still relies on the former 16-bit “real mode” interface. All pre-OS device drivers (such as RAID controllers) on BIOS systems must also be 16-bit. This requirement limits the addressable memory to 64 KB in the early stages of boot and consequently constrains performance.

UEFI is not specific to any processor architecture; it can support modern 64-bit versions of any chipset. The 64-bit capability enables the system to address more than 17.2 billion gigabytes (GB) of memory from the earliest stages of boot.

## CPU-Independent Drivers

On BIOS systems, PCI add-on cards must include a large ROM that contains a separate driver for all supported CPU architectures, or the card vendor must provide a unique stock-keeping unit (SKU) for each processor architecture.

All UEFI implementations that conform to the UEFI specification include an EFI Byte Code (EBC) interpreter. EBC images are drivers that are compatible across all processor architectures. This allows device driver and application developers to create a single EBC image that can run on any system. Additionally, because EBC images are highly compact and universally applicable, option ROMs (that is, drivers) in a PCI card can be much smaller than on BIOS systems and can serve multiple markets. This helps reduce cost and confusion and makes it much easier for system vendors to update or replace drivers as necessary.

## Flexible Pre-OS Environment

UEFI drivers and applications are designed to execute in the boot environment with very few constraints. Therefore, UEFI can provide a full network stack with high-resolution graphics and access to all devices, even if no functional operating system is available. Because UEFI supports a flexible pre-OS programming environment, UEFI applications can perform a wide variety of tasks with any type of PC hardware. For example, UEFI applications can perform diagnostics and firmware upgrades, repair the operating system and notify technicians, or authenticate to remote servers.

## Modular Design

BIOS implementations must be carefully customized for a specific set of hardware, and the tightly interwoven components often make even small changes difficult to accomplish without wide-ranging effects. When new hardware and protocols are introduced, significant portions of BIOS firmware typically must be rewritten, which is expensive and time-consuming.

UEFI defines modular components and generic interfaces that intentionally abstract the details of the hardware/software interface. This approach enables firmware vendors to introduce new hardware and protocols, fix bugs, or alter the behavior of specific components with minimal effects on the rest of the system.

# Windows Support for UEFI

Windows support for UEFI began in 2002. Since then, support for UEFI has become standard in all 64-bit editions of Windows, as follows:

* Windows client operating systems beginning with Windows Vista® SP1:

Windows Vista SP1 and Windows 7 support native UEFI 2.0 or later on 64-bit systems.

* Windows Server® operating systems beginning with Windows Server 2003:

Windows Server 2003 supports EFI 1.10 on Intel Itanium platforms.

Windows Server 2008 and Windows Server 2008 R2 support EFI 1.10 on Intel Itanium systems, and native UEFI 2.0 or later on x64 systems.

**Note: 32-bit versions of Windows do not support UEFI features.** Only 64-bit versions of Windows can take advantage of the features enabled by 64-bit UEFI firmware. Fortunately, current UEFI implementations include a Compatibility Support Module (CSM) that emulates BIOS support. The CSM enables 32-bit operating systems and other operating systems that do not support UEFI to boot on hardware that has UEFI firmware. However, operating systems that require a CSM to boot cannot use UEFI-specific features because the CSM emulates earlier BIOS.

## Windows Features on UEFI

Because of the widespread availability of 64-bit hardware, the capabilities that UEFI offers, and the rapid transition to UEFI firmware, Microsoft has chosen to implement all new firmware-related features on UEFI systems first. Microsoft will evaluate the possibility of additional architectural work to support new features on older BIOS systems on a case-by-case basis.

## Current Windows-Specific UEFI Highlights

Two of the most notable Windows features for UEFI systems are the following:

* Multicast deployment, which enables large scale network-based image deployment in manufacturing and enterprise settings.
* Fast boot and resume from hibernation, which improves user experience.

Brief descriptions of both features follow.

### Multicast Deployment

Most large organizations and system builders use image-based deployment to install an operating system that is preconfigured to meet their specific requirements. The original equipment manufacturer (OEM) or other large organization first creates a customized system image that includes the appropriate applications and settings. When new machines are added or older ones require operating system reinstallation, the image is sent over the network to the target machine.

Traditional unicast image deployment methods require each system to set up an individual connection with a central server and then download the full image over the network before installation can proceed. Unicast deployments often consume considerable network bandwidth and frequently overwhelm central image servers when too many connections are initiated at the same time.

Windows systems that support UEFI can perform multicast image deployment. During a multicast deployment, a central image server can send an image to multiple “listeners” at the same time. Any client that joins while the multicast is underway can receive the latter portion of the image, and then wait for the server to start another broadcast to fill in the first portion. This approach is especially useful in a manufacturing environment, because many clients can simultaneously receive images without overwhelming the network or the image server.

### Fast Boot and Resume from Hibernate

Disk I/O speed significantly affects the time required to boot a computer or load the contents of a hibernation file into memory. The ability to read more data at faster speeds allows the CPU to operate more efficiently and makes both boot and resume from hibernate faster. Earlier BIOS systems use a firmware interface called Interrupt 13h (Int 13) to access block storage devices such as a hard disk drive. By using the Int 13 BIOS interface, software can read data only 64 KB at a time, but the EFI block I/O protocols enable data to be read 1 MB at a time. Windows systems with UEFI can therefore read data more efficiently, which improves boot and resume times.

## Future UEFI Capabilities Under Investigation

The rich UEFI interface provides ample room for innovation in the development of operating system features. Along with the other members of the Unified EFI Forum, Microsoft is investigating the following:

* Rootkit prevention
* Network authentication

### Rootkit Prevention

A common axiom in computer security is, “Whoever touches the hardware first wins.” By running early in the boot process, rootkits can perform malicious actions and then hide their presence from operating system and security software that runs later. This is why rootkits are so dangerous.

UEFI firmware today supports Authenticode digital signatures in the pre-OS environment. By using this capability the firmware can verify each module before it executes and ensure that no untrusted code runs before the operating system loads. This enables the operating system to establish a secure foundation for all the other software on the computer. Microsoft supports this capability and encourages hardware partners to take advantage of it.

### Network Authentication

As part of its pre-OS driver and application support, UEFI can provide network connectivity and authentication in the earliest stages of the boot process. These features could potentially assist in performing secure network-based image deployment. In effect, clients would be able to trust the server from which they download an image, and the server would know that the client was a legitimate corporate asset that should receive a particular image. This capability could potentially be used in a future version of Windows or with other Microsoft deployment technologies.

# UEFI Summary

Although many of the current advantages of UEFI—flexibility, reduced update and development cost, and so on—seem to apply most directly to OEMs and software developers, the end-to-end benefits apply to PC consumers as well. Cheaper, more reliable UEFI systems with a wider variety of capabilities enable new, innovative solutions to difficult problems.

Along with more than 140 other companies in the Unified EFI Forum, Microsoft supports the standardization and interoperability that UEFI provides and welcomes the industry-wide transition to UEFI firmware. For more information about UEFI, see the UEFI Forum Web site, which is listed in “Resources.”

# Resources

#### Windows Hardware Developer Central (WHDC) Web site

UEFI Support and Requirements for Windows Operating Systems

<http://www.microsoft.com/whdc/system/platform/firmware/uefireg.mspx>

#### Firmware Specifications

Advanced Configuration & Power Interface (ACPI) Specification

<http://www.acpi.info/spec.htm>

Unified Extensible Firmware Interface Specifications

<http://www.uefi.org/specs/>

EFI Specification Version 1.10 (included here for historical purposes)

<http://www.intel.com/technology/efi/main_specification.htm>

#### Standards Organizations

Unified Extensible Firmware Interface Forum (UEFI Forum)

<http://www.uefi.org/home/>