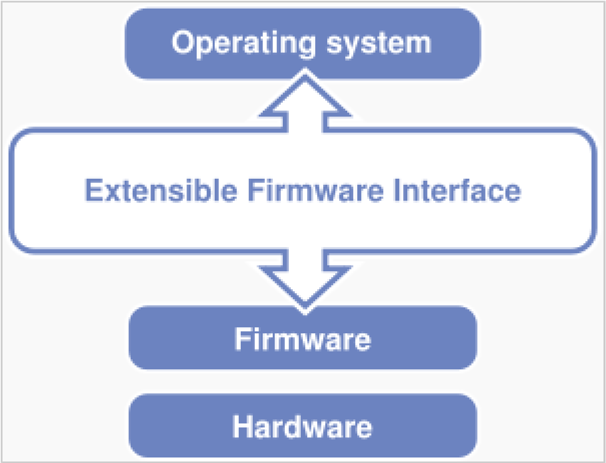
Extensible Firmware Interface

From Wikipedia, the free encyclopedia



Extensible Firmware Interface’s position in the software stack.

The **Extensible Firmware Interface** (**EFI**) is a specification that defines a software interface between an operating system and platform firmware. EFI is a much larger, more complex,[1] OS-like[2]:4 replacement of the older BIOS firmware interface present in all IBM PC-compatible personal computers.[2] The EFI specification was originally developed by Intel, and is now managed by the Unified EFI Forum.

Contents

History

The original motivation for EFI came during early development of the first Intel-HP Itanium systems in the mid-1990s. PC BIOS limitations (16-bit processor mode, 1 MB addressable space, PC AT hardware dependencies, etc.) were seen as clearly unacceptable for the larger server platforms Itanium was targeting. The initial effort to address these concerns was initially called *Intel Boot Initiative* and was later renamed to EFI.[3]

In 2005, the UEFI Forum was created, and is now responsible for EFI development[4] and promotion.

Version 2.1 of the UEFI specification was released on January 7, 2007; as of March 2007. It added cryptography, network authentication, and the User Interface Architecture (Human Interface Infrastructure in UEFI).

Contents

The interface defined by the EFI specification includes data tables that contain platform information, and boot and runtime services that are available to the OS loader and OS.

Some existing enhancements to PC BIOS, such as the Advanced Configuration and Power Interface (ACPI) and System Management BIOS (SMBIOS), are also present in EFI, as they do not rely on a 16-bit runtime interface.

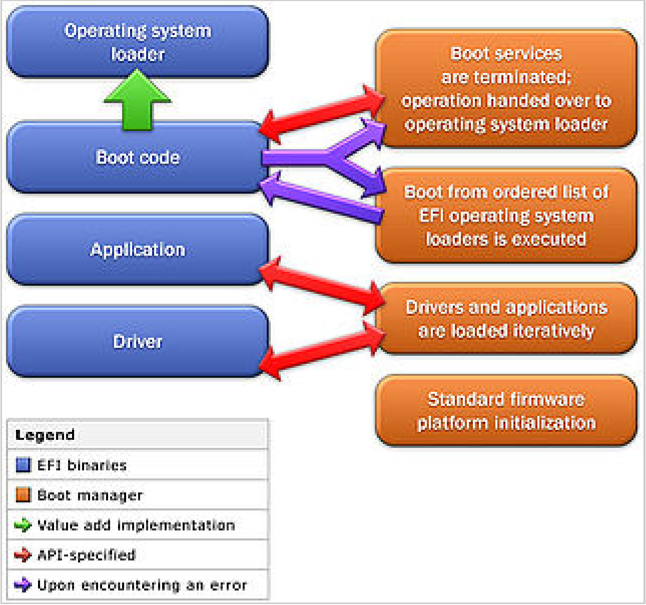
Services

EFI defines *boot services*, which include text and graphical console support on various devices, bus, block and file services, and *runtime services*, such as date, time and NVRAM services.

Protocols

EFI defines *protocol*. A protocol is a set of software interfaces used for communication between two binary modules. All EFI drivers must provide services to others via protocols.

Device drivers


Operating system loader
Boot code
Application
Driver
Boot services are terminated; operation handed over to operating system loader
Boot from ordered list of EFI operating system loaders is executed
Drivers and applications are loaded iteratively
Standard firmware platform initialization
Legend
EFI binaries
Boot manager
Value add implementation
API-specified
Upon encountering an error

Interaction between the EFI boot manager and EFI drivers

In addition to standard architecture-specific device drivers, the EFI specification provides for a processor-independent device driver environment, called **EFI Byte Code** or **EBC**. System firmware is required by the UEFI specification to carry an interpreter for any EBC images that reside in or are loaded into the environment. In that sense, EBC is similar to Open Firmware, the hardware-independent firmware used in PowerPC-based Apple Macintosh and Sun Microsystems SPARC computers, amongst others.

Some architecture-specific (non-EBC) EFI device driver types can have interfaces for use from the operating system. This allows the OS to rely on EFI for basic graphics and network support until OS specific drivers are loaded.

Boot manager

An **EFI boot manager** is also used to select and load the operating system, removing the need for a dedicated boot loader mechanism (the OS boot loader is an EFI application).

Disk support

In addition to the standard PC disk partition scheme, Master boot record (MBR), EFI adds support for a new partitioning scheme: GUID Partition Table (GPT). GPT does not suffer from many of the limitations of MBR, in particular limits on number and size of disk partitions (up to 4 partitions per disk, up to 2TB per partition). The EFI specification does not include a description for a file system; implementations of EFI typically support FAT32 as their file system.[5]

The EFI shell

EFI supports a shell environment;[6]

The shell can be used to execute other EFI applications, but is rarely used for this, and is not used.

Extensions

Extensions to EFI can be loaded from virtually any non-volatile storage device attached to the computer. For example, an original equipment manufacturer (OEM) can sell systems with an EFI partition on the hard drive, which would add additional functions to the standard EFI firmware stored on the motherboard’s ROM.

Implementation and adoption

Intel EFI

Intel's implementation of EFI (the only working implementation of EFI) is called “Tiano”. It runs on Intel's chips:

XScale, Itanium and IA32. It is proprietary software.

A *portion* of the Intel's EFI implementation has been released under the BSD license or Eclipse Public License as the "TianoCore project". Alone it will not do anything useful.[7]

Platforms that use UEFI or the Framework

Intel’s first Itanium workstations and servers, released in 2000, supported EFI 1.02.

Hewlett-Packard’s first Itanium 2 systems, released in 2002, supported EFI 1.10; they were able to boot Windows, Linux, FreeBSD and HP-UX; OpenVMS added support in June, 2003.

All Itanium and Itanium 2 systems ship with EFI compliant firmware.

In November 2003, Gateway introduced the Gateway 610 Media Center, the first x86 Windows-based computer system to use firmware based on the Framework, Insyde Software's InsydeH2O. It still used BIOS to boot Windows.

In January 2006, Apple Inc. shipped its first Intel-based Macintosh computers. These systems use EFI and the Framework instead of Open Firmware, which had been used on its previous PowerPC-based systems.[8] On April 5, 2006, Apple first released Boot Camp, which produces a Windows drivers disk and a non-destructive partitioning tool to allow the installation of Windows XP or Vista without requiring a reinstallation of Mac OS X. A firmware update was also released that added BIOS support to its EFI implementation. Subsequent Macintosh models shipped with the newer firmware. All current Macintosh systems are also able to boot BIOS Operating Systems such as Windows XP and Vista. (With the exception of the Xserve platform)

The grand majority of Intel motherboards ship with Framework-based firmware. During 2005, more than one million Intel systems shipped with the Framework.[9] New mobile, desktop and server products, using the Framework, started shipping in 2006. For instance, all boards that use the Intel 945 chipset series use the Framework. Production firmware usually does not include EFI support, and instead uses BIOS.[10]

Since 2005, EFI has also been implemented on non-PC architectures, such as embedded systems based on XScale cores.[11]

The EDK includes an NT32 target, which allows EFI firmware and EFI applications to run within a Windows application. But there no hardware direct access allowed by EDK NT32. It means only a subset of EFI application and drivers can be executed at EDK NT32 target.

In 2007 HP released the 8000 series multifunction printers with EFI compliant firmware.[12]

In 2008, more x64 systems have adopted the UEFI technology. While many of these systems still only support booting the BIOS-based OSes via CSM (thus does not appear to the user that the system is UEFI based), other systems started to support booting UEFI-based OSes. For example, IBM x3450 server, MSI motherboards with ClickBIOS, all HP EliteBook Notebook and Tablet PCs, newer HP Compaq Notebook PCs (e.g., 6730b, 6735b, etc.).

In 2009, IBM ships System x3550 M2, x3650 M2, iDataPlex dx360 M2 and BladeCenter HS22 with UEFI support.

Operating systems

An operating system that supports boot from (U)EFI is called (U)EFI-aware OS defined by (U)EFI spec. Here term **boot from (U)EFI** means directly boot system using (U)EFI OS loader stored at any of the storage device. The default location for OS loader is \EFI\BOOT\[architecture name]\boot[architecture name].efi. Here architecture name can be IA32, X64, IA64 and other options. Some OS vendors may have their own OS loader. They may also change the default boot location.

- Linux has been able to use EFI at boot time since early 2000, using the elilo EFI boot loader or, more recently, EFI versions of GRUB.[13]
- HP-UX has used (U)EFI as its boot mechanism on IA-64 systems since 2002.
- HP OpenVMS has used (U)EFI since its initial evaluation release in December 2003, and for production

releases since January 2005[14].
- Apple has adopted EFI for its line of Intel-based Macs. Mac OS X v10.4 Tiger for Intel and Mac OS X v10.5 Leopard support EFI v1.10 in 32-bit mode, even on 64-bit CPUs (newer Macs have 64-bit EFI).[15]
- The Itanium versions of Windows 2000 (Advanced Server Limited Edition and Datacenter Server Limited Edition) supported EFI 1.10 in 2002. Windows Server 2003 for IA-64, Windows XP 64-bit Edition, and Windows 2000 Advanced Server Limited Edition, all of which are for the Intel Itanium family of processors, support EFI, a requirement of the platform through the DIG64 specification.[16]
- Microsoft introduced UEFI support for x64 Windows operating systems with Windows Server 2008 and Windows Vista Service Pack 1. Microsoft has released a video with Andrew Ritz and Jamie Schwarz explaining Pre-OS support involving UEFI on Windows Vista and Windows Server 2008.[17]

Virtualization Support

- HP Integrity Virtual Machine supports UEFI boot on HP Integrity Servers. It also provides virtualized UEFI environment for the guest UEFI-aware OSes.
- Microsoft Hyper-V supports UEFI boot on x64 servers.

Graphic feature

EFI supports graphical menus and features, such as is implemented on Aptio or Great Wall UEFI.[18]

There are two protocols defined by EFI spec can be used for graphics output. The first one is UGA, which stands for Universal Graphic Adapters. The second one is GOP, which stands for Graphic Output Protocol. The two are similar. UGA is only supported at EFI 1.1 or older specs. EFI does not define how a system should look. This depends fully on the implementation by BIOS vendors. Currently most EFI enabled machines only have a legacy BIOS-like UI.

Criticism

EFI has been criticized for bringing more complexity to the system, without bringing significant advantages[1] and disabling fully open-source BIOS replacements like OpenBIOS and Coreboot[19]. It does not solve any of BIOS's long standing problems of requiring two different drivers for most hardware.[20] It is not clear why it is beneficial to have two completely different operating systems running concurrently and doing similar things, or why a new operating system has to be created from scratch.[20]

The Free Software Foundation has criticized Intel for pretending to open up their platform, and for preventing interoperability: "It won't run, and doesn't bring us any closer to a BIOS that does run. It is just a distraction."[7]

See also

- x86-64
- Advanced Configuration and Power Interface (ACPI)
- Basic Input/Output System (BIOS)
- Booting
- Coreboot
- El Torito
- Functional specification
- Live USB
- Open Firmware
- OpenBIOS
- System Management BIOS (SMBIOS)
- System Management Mode
- Unified EFI Forum

References

1. ^ ***a*** ***b*** http://kerneltrap.org/node/6884
2. ^ ***a*** ***b*** Michael Kinney, "Solving BIOS Boot Issues with EFI

(http://www.intel.com/technology/magazine/computing/it09006.pdf)”. *Intel DeveloperUPDATEMagazine*. September 2000, p. 1.

3. ^ Dong Wei, *Beyond BIOS* (foreword), Intel Press, 2006, ISBN 9-780974-364902
4. ^ Unified EFI Forum. “UEFI - About UEFI (http://www.uefi.org/about/)"
5. ^ fat-driver.Tianocore.org (http://fat-driver.tianocore.org/) for info on EFI FAT driver. (requires registration)
6. ^ efi-shell.tianocore.org (https://efi-shell.tianocore.org/) for EFI shell information
7. ^ ***a b*** http://www.fsf.org/campaigns/free-bios.html
8. ^ Apple Computer. “Universal Binary Programming Guidelines, Second Edition: Extensible Firmware Interface (EFI) (http://developer.apple.com/documentation/MacOSX/Conceptual/universal_binary/universal_binary_diffs/chapter_3_section_
9. ^ Intel, *Framework Overview (http://www.intel.com/technology/framework/overview2.htm)*
10. ^ [1] (http://www.intel.com/cd/channel/reseller/asmo-na/eng/336115.htm)
11. ^ Overview of the Intel Platform Innovation Framework (http://www.intel.com/technology/framework/overview1.htm)
12. ^ HP Printer’s site [2] (https://h30046.www3.hp.com/campaigns/2007/promo/edgeline-mfp/index.php?regioncode=NA&langcode=USENG)
13. ^ EFI version of Grub (Debian Linux) (http://packages.debian.org/sid/grub-efi) - Retrieved on 1 May 2008.
14. ^ HP OpenVMS Release History (http://h71000.www7.hp.com/openvms/os/openvms-release-history.html) - Retrieved on 16 September 2008
15. ^ rEFIt - Windows Vista and EFI (http://refit.sourceforge.net/info/vista.html)
16. ^ Microsoft Windows Server TechCenter. “Extensible Firmware Interface (http://technet2.microsoft.com/WindowsServer/en/Library/6b03fad7-665e-49f3-8e7d-e1a6a5be9cf01033.mspx)"
17. ^ Microsoft Pre-OS Video (http://channel9.msdn.com/showpost.aspx?postid=292774)
18. ^ Intel shows PC booting Windows with UEFI firmware (http://apcmag.com/5862/intel_shows_pc_booting_windows_with_uefi_firmware)
19. ^ Interview: Ronald G Minnich (http://archive.fosdem.org/2007/interview/ronald+g+minnich)
20. ^ ***a b*** http://www.youtube.com/watch?v=X72LgcMpM9k

External links

- UEFI Homepage (http://www.uefi.org/)
- EFI Architecture (http://www.ddj.com/dept/embedded/199500688?cid=RSSfeed_DDJ_All) Dr. Dobbs Portal Article
- EFI Linux gnu toolchain (gnu-efi) (http://gnu-efi.sourceforge.net/)
- EFI Linux Loader project (ELILO) (http://elilo.sourceforge.net/)
- rEFIt, a boot menu and maintenance toolkit for EFI-based machines (http://refit.sourceforge.net/).
- Intel’s EFI page (http://www.intel.com/technology/efi/)
- Intel’s Framework page (http://www.intel.com/technology/Framework/)
- Intel List of commands (http://softwarecommunity.intel.com/articles/eng/2553.htm)
- Intel-sponsored open-source EFI Framework initiative (http://www.tianocore.org/)

Retrieved from "http://en.wikipedia.org/wiki/Extensible_Firmware_Interface"
Categories: BIOS | Macintosh firmware
Hidden categories: All articles with dead external links | Articles with dead external links since April 2009