

## FOREWORD

by **Doug Fisher**

Intel Vice President

General Manager, Systems Software Division

From its roots in 1997 to support Intel® Itanium® based servers and the first published Extensible Firmware Interface (EFI) specification around 2000, Unified Extensible Firmware Interface (UEFI) has now eclipsed legacy BIOS across all computing platforms from high performance servers to mobile devices to deeply embedded devices. The success of UEFI would not have been possible without industry support of the open source implementation and industry-guided platform firmware specification. UEFI as an open standard specification ensured that it received broad support and investment from across the industry to evolve as required to support technology changes in our industry, for example boot time requirements and an ever increasing focus on security. This issue of the Intel Technology Journal is completely focused on UEFI.

The first article is an introduction to UEFI written by Mark Doran, Vincent J. Zimmer, and Michael A. Rothman; it provides both the details on the history of UEFI and how the specifications are managed within the UEFI Forum. The intent here is to not only better understand how UEFI is maintained as an open specification, but to also provide clarity on the usages and capabilities UEFI has made possible within our industry.

Silicon enabling provides vendor-specific value-added features on the platform with key chipset and CPU enabling code for the latest hardware. UEFI Platform Initialization (PI) specifications support this capability. In “Silicon Enabling,” authors Isaac Oram, Tim Lewis, and Vincent J. Zimmer focus on building block elements in PI that are the cornerstone of silicon enabling that provides OEMs with consistent and sufficient interfaces to build real systems without precluding opportunities for differentiation.

Regardless of how the computer industry has evolved over the years, the need for interoperability between the platform elements is unchanged. In “UEFI and the OEM and IHV Community,” authors Nathan Skalsky, Terry Kirch, Al Rickey, and Michael A. Rothman discuss how the UEFI standard introduces the composite pieces for such interoperability, and in so doing, illustrate how the end user benefits by both the OEM and IHV communities’ use of UEFI standards.

Target platforms supported by UEFI span deeply embedded devices to massive server clusters and everything in-between. One common desire

across all of these domains is the desire to reduce the time it takes for the platform to initialize (boot). Authors Michael A. Rothman, and John Mese, in “Booting in an Instant” cover the elements associated with boot performance from various points of view in the technology value chain. They also discuss how the evolution of standards (such as UEFI) have resulted in boot performance enhancements, and give examples of how these technology elements were incorporated into products to provide meaningful results to the end user.

The protection from security threats at all platform states, pre- and post-OS boot are increasingly important for our industry. In “UEFI Networking and Pre-OS security” authors Magnus Nystrom, Martin Nicholes, and Vincent J. Zimmer focus on how UEFI includes security in pre-OS state and networking for the platform boot process from local and remote media, assets to be protected, threats against those assets, and the various technologies that allow for their protection. They also go one step further to discuss forward-looking security capabilities and approaches enabled by UEFI.

For software developers, debugging can be a challenge and robust debugging tools are essential at every phase of development. UEFI is no different, and can be very difficult especially for embedded devices where access is limited. Stefano Righi, Brian Richardson, Jiewen Yao, and Elvin Li in “Debugging Issues with UEFI” provide an overview of common debug solutions including hardware-based debugging, system checkpoints, and source-level debugging. Firmware-specific concepts such as status codes, DEBUG/ASSERT macros and the UEFI debug protocol are introduced. They also discuss source-level debugging support using AMI and Intel solutions, comparing them to hardware-based alternatives.

To further emphasize the broad platform range (printers to high performing servers) associated with UEFI, the final article in this edition of the journal discusses one specific company’s use of UEFI across broad product lines. Dong Wei, Kimon Berlin, and Eugene Cohen all from Hewlett Packard discuss the specific strategy for value add for leveraging UEFI across the HP product portfolio.

This set of articles will show both a) the value of UEFI to our industry and how UEFI is positioned to continue to evolve and extend as required to meet the rapidly evolving requirements of our industry and customers, and b) the

impact of open standards like UEFI and how they are uniquely capable to provide the needed capability without precluding OEM differentiation. UEFI has covered tremendous ground since that first EFI specification around 2000, and I'm looking forward to watching how it evolves for the next 10 years to cover computing capabilities and usages that we've not yet imagined.

Doug Fisher