UEFI Option ROMS and Plug-Ins

October 12, 2010

Max Lee,
Senior Director, Engineering
Phoenix Technologies
Agenda

• Benefits of native (non-CSM) UEFI Option ROMs

• UEFI Plug-ins – Ways to add value at the BIOS level
This is a summary of an IDF presentation a few weeks ago

Some of the material is borrowed from that joint presentation

Thanks to Intel, Dell and LSI for their contributions
UEFI HII makes for a friendly user experience

- Forms-based model for setup question descriptions
  - Must meet BIOS requirements
    - Scalable UI display support (Server Front Panel to local high resolution monitor).
    - Small encoding size
      - Encoding that is Self Describing
      - Can support scripting
      - Extensible syntax

**UEFI provides a simple yet powerful method to describe configuration data**
Input and Output Localization

- With Forms, localization is straightforward

<table>
<thead>
<tr>
<th>String ID #4</th>
<th>String Representation</th>
<th>H</th>
<th>E</th>
<th>L</th>
<th>L</th>
<th>O</th>
<th>W</th>
<th>O</th>
<th>R</th>
<th>L</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Unicode Encoding</td>
<td>0x0048</td>
<td>0x0045</td>
<td>0x004C</td>
<td>0x004C</td>
<td>0x004F</td>
<td>0x0020</td>
<td>0x0057</td>
<td>0x004F</td>
<td>0x0052</td>
<td>0x004C</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>String ID #4</th>
<th>String Representation</th>
<th>H</th>
<th>O</th>
<th>L</th>
<th>A</th>
<th>M</th>
<th>U</th>
<th>N</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Unicode Encoding</td>
<td>0x0048</td>
<td>0x004F</td>
<td>0x004C</td>
<td>0x0041</td>
<td>0x0020</td>
<td>0x000D</td>
<td>0x0055</td>
<td>0x004E</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>String ID #4</th>
<th>String Representation</th>
<th>你</th>
<th>好</th>
<th>世</th>
<th>界</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Unicode Encoding</td>
<td>0x4F60</td>
<td>0x597D</td>
<td>0x4E16</td>
<td>0x754C</td>
</tr>
</tbody>
</table>

Both input and output localization is supported
• EBC (EFI Byte Code) allows a single image option ROM to operate on multiple CPU environments
• Maximal compatibility while minimizing binary size impact

Legacy Environment

Binary Image for CPU type x
Binary Image for CPU type y
Binary Image for CPU type z

UEFI Environment

EBC Image for all CPU types running on UEFI
Driver Health Protocol for Reporting

- Issues with Current Error reporting model
  - POST flow interrupted every time
  - Errors on multiple devices could result in multiple system reboots

- Driver health protocol
  - Allows consolidation of all the error reporting and user interaction
  - Allows user to address all of the issues at the same time
  - Avoids multiple reboots
UEFI OpROM Benefits Summary

• Faster Time to Market & Cost Savings
  – Modern/well defined and documented architecture
  – Code in C and not assembly
  – Do it once using EBC

• Richer Capabilities
  – Support for > 2.2 TB and latest Hard Disk technologies
  – Unified Interfaces across Option ROM Code
  – Clean and Portable Solutions: UEFI Drivers and Utilities
  – Support for Hybrid Systems - UEFI and Legacy BIOS
  – Ability to quickly implement new features and additional OEM requirements
  – UEFI provides direct access to all of (64-bit) Memory

• Enhanced Usability
  – HII allows IHVs to focus mainly on Functionality and Content rather than carrying own GUI so that:
    • OEMs can maintain their own look-n-feel across their platforms through their HII Browser
  – Easier Localization Support
Agenda

• Benefits of (non-CSM) UEFI Option ROMs

• UEFI Plug-ins – Ways to add value at the BIOS level
Plug-Ins: Added Value for PCs

• Plug-Ins are added value for PCs installed by:
  – The OEM
  – The End User

• What plug-ins do we use today?
  – For MP3 players, it’s earphones, power supplies, etc.
  – For PDAs & Smart Phones, it’s app store software
  – For PCs, plug-ins extend functionality too
Plug-Ins: Added Value for PCs

• OEM Plug-Ins:
  – Likely to exist in source code form
  – Require technical integration into the BIOS in some way (source, adaptation, etc.)
  – Integrated as part of system test

• User Plug-Ins:
  – Need seamless binary installation
  – Lots of issues (security, storage, configuration, compatibility, etc.)
  – Must just work without any “system test” on the user’s part
Plug-Ins: Added Value for PCs

• In the legacy BIOS days, plug-ins made hardware operational– ROM BIOS extensions (OpROMs)

• Today’s add value is less about new hardware options, and more about other things:
  – Virus/Malware Protection
  – Enterprise Management
  – OS Installation
  – Geo-Fencing
  – Instant-On environments
  – Diagnostics
- All new systems shipping with some form of UEFI

- Phoenix creating UEFI solutions for all new silicon solutions

- Green H: Formal packaging of executable entities, run-order, flow control
  - Does away with hooking and patching
There can be several types of UEFI plug-ins that run in different environments

- **Bootware**
  - Applications that run in the pre-OS environment (in what is traditionally the BIOS environment)

- **Autonomous Computing**
  - UEFI applications that run in parallel with the OS

- **Invisible Computing**
  - Applications that run while the system is perceived by the user to be “off” (in S3, S4 or S5)
Summary

• We believe:
  – Plug-Ins are going to take off, as the role of the BIOS/Pre-Boot is standardized and stabilized
  – Importance of Plug-Ins will increase
    • Allows for differentiation and expandability in otherwise closed systems
  – IBVs, ODMs, OEMs, and SVs will pave the way for plug-In manufacturers to add value:
    • First at the source code level as they sell to OEMs
    • Finally at the binary level as end users install their own plug-ins
Questions?

Events@phoenix.com