A look at EFI

- Interface specification
  - Implementation agnostic
- Abstracts BIOS from OS
  - Decouples development
- Compatible by design
  - Evolution, not revolution
- Modular and extensible
  - OS-Neutral value add
- Provide efficient Option ROM Replacement
  - Common source for multiple CPU architectures

*Other names and brands may be claimed as the property of others*
A couple steps forward

Pre Verifier → Verify

PEI Core
CPU Init
Chipset Init
Board Init

EFI Driver Dispatcher

Device, Bus, or Service Driver

Architectural Protocols

UEFI Interfaces

OS-Absent App
OS Environments
Transient OS Boot Loader

Pre Verifier

OS-Present App
Final OS Boot Loader
Final OS Environment

Security (SEC)
Pre EFI Initialization (PEI)
Driver Execution Environment (DXE)
Boot Dev Select (BDS)
Transient System Load (TSL)
Run Time (RT)

Power on → [ ... Platform initialization ... ] → [ ... OS boot ... ] → Shutdown

*Other names and brands may be claimed as the property of others
Unified EFI Forum, Inc.

A Firmware Standards Organization Gets Created

A Washington non-profit Corporation
- Develops, promotes and manages evolution of Unified EFI Specification
- Continue to drive low barrier for adoption

Promoter members:
- AMD, AMI, Apple, Dell, HP, IBM, Insyde, Intel, Lenovo, Microsoft, Phoenix

More information: [www.uefi.org](http://www.uefi.org)

*Other names and brands may be claimed as the property of others*
How the Forum Works

UEFI Board

USWG

PIWG

UTWG

ICWG

UCST

UNST

USST

Each work group approves/delivers different content to the public.

Each sub-team focuses on specific topics and contributes material to the work group.

Publications/Decisions ratified by the board

*Other names and brands may be claimed as the property of others
UEFI Relationships In Industry

Relationship with DMTF

*Other names and brands may be claimed as the property of others
**Concept Review - Protocols**

### Read Keystroke Example

#### Legacy

<table>
<thead>
<tr>
<th>INT 16h</th>
<th>Input</th>
<th>AH = 10h</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Output</td>
<td>AH = Scan code</td>
</tr>
<tr>
<td></td>
<td>Output</td>
<td>AL = ASCII character</td>
</tr>
</tbody>
</table>

**Caller Sample Code**

```assembly
mov ax, 1000h
int 16h
```

**Handler Sample Code**

```assembly
cmp ah, 10h
jz HandleExtReadKey
cmp ah, 11h
jz CheckForKey
;; Do more checking
HandleExtReadKey:
;; Do real work here
mov ax
ret
```

#### UEFI/Framework

<table>
<thead>
<tr>
<th>Simple Text Input Protocol</th>
<th>Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReadKeyStroke</td>
<td>ReadKeyStroke</td>
</tr>
<tr>
<td>Reset</td>
<td>Reset</td>
</tr>
<tr>
<td>WaitForKey</td>
<td>WaitForKey</td>
</tr>
</tbody>
</table>

**Caller Sample Code**

```c
TextIn->ReadKeystroke (TextIn, &Key);
```

**Handler Sample Code**

```c
ReadKeyStrokeHandler (  
    IN EFI_SIMPLE_TEXT_INPUT_PROTOCOL *This,  
    OUT EFI_INPUT_KEY *Key  
    )  
{  
    // Do real work here  
}  
```

*Other names and brands may be claimed as the property of others*
Moving Data from pre-boot into O/S runtime

EFI System Table

EFI Configuration Table

<table>
<thead>
<tr>
<th>GUID</th>
<th>Pointer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table A GUID</td>
<td>Address A</td>
</tr>
<tr>
<td>Table B GUID</td>
<td>Address B</td>
</tr>
<tr>
<td>Table Y GUID</td>
<td>Address Y</td>
</tr>
<tr>
<td>Table Z GUID</td>
<td>Address Z</td>
</tr>
</tbody>
</table>

Local Configuration/Manageability

*Other names and brands may be claimed as the property of others
Problem Statement

• No standard/interoperable mechanism to address pre-boot based issues like:
  – Localization
    • Standard delivery of string packages
  – Fonts
    • Create standard glyph support along with optional font styles
  – Shared Configuration Infrastructure
    • Alleviate the burden for many configuration engines in a system (e.g. add-in device
      no longer needs to delay boot or poll for hot-keys, etc)

• Should be able to also address:
  – Human -> Machine system configuration
    • Think Setup
  – Machine -> Machine system configuration
    • Think Automation

Platform Configuration/Manageability

*Other names and brands may be claimed as the property of others
UEFI Interactions

• Interaction Component View

IBM Director

In Band

Out of Band

UEFI 2.0

PI Architecture 1.0

Physical SMP Server

BMC

Service Processor

Server

Server

I/O Controller

OS

Local Configuration/Manageability

*Other names and brands may be claimed as the property of others
Interactions with BMC

- Local Interactions / with data being handed to firmware through BMC across reset.
Configure Heterogeneous Targets

Three classes of platforms each with different configuration maps in their FLASH.

Large Corporate Customer
Platform Schema Definition

Administrator sends a configuration directive to the client workstations. The heterogeneous targets must be able to interpret the directives into platform-specific actions.

Platform Configuration/Manageability

*Other names and brands may be claimed as the property of others
Network Infrastructure

Remote Configuration/Manageability

*Other names and brands may be claimed as the property of others
**UEFI Configuration/Manageability Infrastructure**

- **Keyboard Localization**

- **String/Text Localization**

<table>
<thead>
<tr>
<th>String ID #4</th>
<th>String Representation</th>
<th>HELLO</th>
<th>WORLD</th>
<th>HOLALA</th>
<th>MUNDO</th>
<th>你 好 世 界</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Unicode Encoding</td>
<td>0x0048 0x0045 0x004C 0x004F</td>
<td>0x0020 0x0057 0x004F 0x0052</td>
<td>0x004C 0x0044 0x0000</td>
<td>0x004D 0x0055 0x004E 0x0044 0x004F 0x0000</td>
<td></td>
</tr>
</tbody>
</table>

*Other names and brands may be claimed as the property of others*
Glyph Support

• Dependent on Int10h character support? No....

*Other names and brands may be claimed as the property of others.
Using Forms

- 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
  - Position Independent
  - Can support scripting
  - Extensible syntax

- Exact look and feel defined by the browser and not defined in UEFI 2.1.
  - Developer/OEM/IHV defines questions to ask and what strings to display
  - Browser determines “how” to display the questions
Defining a User Interface – leave to OEM/IBV

*Other names and brands may be claimed as the property of others
UEFI Interactions

• Device Access APIs (usable today – talk about again in futures)...

typedef struct {
    EFI_HII_EXTRACT_CONFIG ExtractConfig;
    EFI_HII_ROUTE_CONFIG RouteConfig;
    EFI_HII_FORM_CALLBACK Callback;
} EFI_HII_CONFIG_ACCESS_PROTOCOL;

Local Configuration/Manageability

*Other names and brands may be claimed as the property of others
Backup
**Local Configuration/Manageability**

*Other names and brands may be claimed as the property of others*
UEFI Interactions

• Local Interactions – Exporting up to the O/S

Local Configuration/Manageability

*Other names and brands may be claimed as the property of others
Moving Data through Capsules

Introduce new APIs for complex data communication.

Local Configuration/Manageability

*Other names and brands may be claimed as the property of others
Configuration Portability

Platform Configuration/Manageability

*Other names and brands may be claimed as the property of others
Network-based Configuration Interactions

• Construction of configuration packets
UEFI Interactions

Remote Configuration/Manageability

*Other names and brands may be claimed as the property of others
Use in Manufacturing

0. Vendor Build generates blob. Put on vendor’s server.

Step 1: Integrator downloads data blobs describing cards and mobo to his notebook.

Step 2: Integrator configures cards for his application.

Step 3: When done, data is downloaded to manufacturing server.

Step 4: Later, systems start manufacture and are configured during integration.

Remote Configuration/Manageability

*Other names and brands may be claimed as the property of others.
UEFI Overview

In – UEFI 2.1 target

• Produce configuration infrastructure specification (What everyone is reviewing now)

• Purpose:
  – Enable IHVs to have a standard to write against for purposes of HW config.
  – Enable Industry (OEM/IBV/etc) for proprietary platform configuration and display mechanisms within the pre-boot and extend this into O/S runtime enablement.
  – Other functional extensions are possible....
UEFI Configuration Sub-team (UCST)

Stage 2 – to be worked on (UEFI 2.2 or whitepaper or both)

• Using the efforts in Stage 1, further describe how to move from a proprietary platform config mechanism to an environment which mixes multiple proprietary and standard namespace-based ones.
  – Issues to work on:
    • Refining our description of how exactly to determine the config keyword from a pre-existing namespace (e.g. CLP) and apply it to the platform.
    • Challenge will be if existing namespaces are insufficient in describing a “keyword”, get it so that they do, or establish some whitepaper material for guiding people on how to do it.
    • There is an open challenge to avoid creating our own namespace for many reasons – but it is always an option, just not one I relish the thought of doing.

Futures

*Other names and brands may be claimed as the property of others
UEFI & DMTF Work Register
-Now Approved-

Current Directions / UEFI & DMTF

*Other names and brands may be claimed as the property of others
DMTF Register Actions

• Drive UEFI representation through schema, profile(s) and mapping specification(s) to ensure that the proper support (such as configuration capabilities and namespace requirements) exists for both traditional BIOS as well as UEFI standards.
  • CIM Schema 2.16 (3Q2007)
  • BIOS Profile 1.0 (3Q2007)
  • BIOS SM CLP Mapping Specification 1.0 (3Q2007)
  • Investigate and Contribute to BIOS Profile(s).

• Promote relevant DMTF material back into the UEFI 2.2 specification. (1Q2008)

• Inclusion of updates as appropriate to the upcoming DASH Management Initiative updates.
  • Inclusion of BIOS Profile in DASH 1.1 (4Q2007)

Futures

*Other names and brands may be claimed as the property of others
DMTF Register Actions - II

• Collaboration on appropriate UEFI requirements on specifications developed within DMTF Working Groups. This includes:
  • System and Option ROM Identifiers
  • Command/Response strings

• This would result in the following specifications that would need to be shared with UEFI:
  • SM CLP Specification Updates & Work in Progress Drafts
  • SM CLP Mapping Specification Updates & Work in Progress Drafts
  • CIM Schema Work in Progress Drafts
  • DMTF Profile Work in Progress Drafts

• This would result in the following specifications that would need to be shared with DMTF:
  • UEFI Specification Updates & Work in Progress Drafts

Futures

*Other names and brands may be claimed as the property of others
**Protocol Diagram**

- **Red (solid)** lines are new (for UEFI 2.0) network protocol APIs that can be accessed by multiple applications and drivers at the same time.

- **Green (broken)** lines are existing network protocol APIs (from EFI 1.1 or earlier) that can only be accessed by one application or driver at a time.

---

*Other names and brands may be claimed as the property of others*
Driver Model Relationship

Device Driver

Configuration Access Protocol

Configuration Access Protocol

BUS Driver

*Other names and brands may be claimed as the property of others