Intel® Platform Innovation Framework for UEFI and Platform Initialization Overview

Intel Corporation
Software and Services Group

Copyright © 2006-2009 Intel Corporation
Agenda

• Design Approach
• UEFI Spec. and Platform Initialization Spec.
• Boot Execution Flow
• Overview of Each Phase
  – Pre-EFI Initialization (PEI)
  – Driver eXecution Environment (DXE)
  – Boot Device Selection (BDS)
  – Backward Compatibility (CSM)
• Development Environment
• Summary
The framework\textsuperscript{1} Technical Goals:

- Architectural design to last a second 20 years
- Intel\textregistered{} IA-32, Itanium\textregistered{} architecture and Xscale\textregistered{} technology applicability — in one source tree!
- Clean, scalable, architecture
- Modular across companies
- Driver-based design allowing for binary linking
- “C” based, no exotic tools
- Meet size and boot time requirements
- Legacy accommodation

\textsuperscript{1}Intel\textregistered{} Platform Innovation Framework for UEFI
The framework\(^1\) Design Strategy

- High level design based on the framework\(^1\) plus modular components
- Generalize the framework\(^1\) Maximize reuse of infrastructure
  - High degree of independence from platform and market segment specifics
- Specifics encapsulated in the drivers
  - Drivers map to software visible hardware
  - Isolate hardware/platform specifics to support component-based firmware construction

\(^1\)Intel® Platform Innovation Framework for UEFI
Get to “C” Code Quickly

- Commercial “C” compilers use stack model
  - Requires some memory initialized for a stack
- Split the framework\(^1\) infrastructure in two
  - Pre-EFI Initialization (PEI), preamble to get memory
  - Driver Execution Environment (DXE), infrastructure to support “C” coded EFI drivers
- First part of the framework\(^1\) finds memory by using special stack
  - Infrastructure code plus PEI Modules
- The framework\(^1\) uses modules for CPU, chipset and board
  - Minimum initialization to get memory working
- Architecture only requires “enough” memory
  - PEI limited so defer to rich DXE “C” environment

Standard tools Flexible memory initialization

\(^1\)Intel® Platform Innovation Framework for UEFI
What is the framework?

Intel® Platform Innovation Framework for UEFI

- Pre-UEFI Foundation
- DXE Foundation
- The framework¹ Drivers
- Architectural Protocols
- Platform Drivers
- EFI Drivers
- Compatibility Support Module (IA-32 only)

¹Intel® Platform Innovation Framework for UEFI
Agenda

• **Design Approach**
• **UEFI Spec. and Platform Initialization (PI) Spec.**
• **Boot Execution Flow**
• **Overview of Each Phase**
  – Pre-EFI Initialization (PEI)
  – Driver eXecution Environment (DXE)
  – Boot Device Selection (BDS)
  – Backward Compatibility (CSM)
• **Development Environment**
• **Summary**
2005 the Unified EFI (UEFI) Forum was formed.
   - Stakeholders agreed to use EFI 1.10 specification as the starting point

Industry commitment drives need for broader governance on specification
   - Currently 11 Companies on the Board of directors
   - Many more are Contributors and Adopters

Intel and Microsoft contribute - see material on http://uefi.org for updated specification

Intel contributed EFI test suite - UEFI SCT
USWG/PIWG Relationship

- UEFI Spec is about interfaces between OS, add-in driver and system firmware
  - a new model for the interface between the Operating systems and other high-level software and the platform firmware
- PI Specs relate to making UEFI implementations
  - Promote interoperability between firmware components providers
  - Modular components like silicon drivers (e.g. PCI) and value-add drivers (security)

UEFI and PI are Independent Interfaces
Why UEFI and PI

• Modern architecture
  - Easier to support,
  - Easier to differentiate

• Modular architecture
  - Build/buy/share/reuse modules independently
  - Focus internal work where it’s most valuable

• Fully specified
  - Compliance tests ensure higher quality
  - Reduces integration headaches
Details on Platform Initialization (PI) Specification

- Roughly one year of Specification work
  - Builds on the PEI and DXE framework specifications
- Board approved formal PI 1.0 Oct, 2006
  - Available for download from [http://www.uefi.org](http://www.uefi.org)
- Starting work from Intel Corp. contributed specifications
  - The framework DXE and PEI Core Interface Specs
  - Firmware Storage, HOB, SMBus
- PI Self Certification Test (SCT) available on [http://sourceForge.net/projects/pi-sct](http://sourceForge.net/projects/pi-sct)
- PI 1.1 Approved end of 2007
- PI 1.2 In process of getting approved

¹Intel® Platform Innovation Framework for UEFI
**PI Specification Volumes**

- **VOLUME 1**: Pre-EFI Initialization Core Interface (PEI-CSI)
- **VOLUME 2**: Driver Execution Environment Core Interface (DXE-CSI)
- **VOLUME 3**: Shared Architectural Elements
- **VOLUME 4**: System Management Mode (SMM)
- **VOLUME 5**: Standards

- Over 1200 pages total
Details on what is in PI1.1

Completed End of 2007  PI1.1 includes:

- PCI Specifications – root bridge, host bridge, hot plug, override
- MP protocol for DXE
- SMBIOS table creation API
- S3 boot script infrastructure
- SMM & PMI Component Interface specifications
## Overview of Differences – PI 1.0 Vs. framework

### Components

<table>
<thead>
<tr>
<th>Component</th>
<th>Actions / Exceptions</th>
</tr>
</thead>
</table>
| Compatibility           | Do not access internals of the firmware files  
                          | Do not use ReportStatusCode                                                                                                                        |
| PEI File System         | Minor change to the file header and firmware volume header                                                                                         |
| PPI Updates             | PCI PPI for Extended PCI-express  
                          | New PPI – Terminate End of Temp Memory                                                                                                              |
| DXE Service Table       | Removed Report Status Code service                                                                                                                  |
| New Architectural Protocol | Capsule AP / QueryVariableInfo                                                                   |
| HOB definitions         | More Firmware volume information  
<pre><code>                      | Remove Capsule HOB definition                                                                                                                      |
</code></pre>
<table>
<thead>
<tr>
<th>Component</th>
<th>Actions / Exceptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMM</td>
<td>SMM driver model change. Port code</td>
</tr>
<tr>
<td>S3</td>
<td>New execution requirements for native callbacks. Some interfaces removed</td>
</tr>
<tr>
<td>PCI</td>
<td>New event. Should be able to enhance former implementation</td>
</tr>
<tr>
<td>MP</td>
<td>Clean-up. Port to use new API</td>
</tr>
<tr>
<td>DXE</td>
<td>Cleaned-up DXE to be UEFI 2.0 RT compatible</td>
</tr>
<tr>
<td>SMBIOS table creation</td>
<td>The framework(^1) used data hub. This is a new API</td>
</tr>
</tbody>
</table>

\(^1\)Intel® Platform Innovation Framework for UEFI

**PI 1.1 Expands PI 1.0 infrastructure w/ more building blocks**
UEFI and PI Specifications

**PI 1.2 Approval in Progress**

- Based on the existing framework\(^1\) specifications
  - Highest priority subset chosen
- . . .
- More . . .

\(^1\)Intel® Platform Innovation Framework for UEFI
Agenda

• Design Approach
• UEFI Spec. and Platform Initialization Spec.
• Boot Execution Flow
• Overview of Each Phase
  – Pre-EFI Initialization (PEI)
  – Driver eXecution Environment (DXE)
  – Boot Device Selection (BDS)
  – Backward Compatibility (CSM)
• Development Environment
• Summary
Boot Execution Flow

Architecture Execution Flow

<table>
<thead>
<tr>
<th>Security (SEC)</th>
<th>Pre EFI Initialization (PEI)</th>
<th>Driver Execution Environment (DXE)</th>
<th>Boot Dev Select (BDS)</th>
<th>Transient System Load (TSL)</th>
<th>Run Time (RT)</th>
<th>After Life (AL)</th>
</tr>
</thead>
</table>

Power on → [ . . Platform initialization . . ] → [ . . . . OS boot . . . . ] → Shutdown
POST Execution Flow – High Level

- **Reset**
- **CPU Init**
- **Memory Init**
- **CS Init**
- **Boot Mode**
- **POST Dispatch**
- **Console Init**
- **Device Init**
- **Bus Init**
- **Boot Dev Select**
- **Legacy OS Load**
- ** EFI Pre-boot Application**
- **Normal Boot**
- **S3 Resume**
- **Recovery**
- **OS Runtime**

Boot Execution Flow
POST Execution Flow – High Level

- **PEI**
  - Reset
  - CPU Init
  - Memory Init
  - CS Init
  - Boot Mode
    - S3 Resume
    - Recovery

- **POST**
  - Firmware Volumes
  - Cache as RAM
  - PEIM
  - Handoff Blocks HOB
  - NVRAM
  - Capsules

- **Boot Dev**
  - Legacy OS Load
  - EFI Pre-boot Application

- **OS Runtime**
  - GUID

**Boot Execution Flow**
POST Execution Flow – High Level

**POST Execution Flow**

- EFI Driver Dispatcher
  - CPU Init
  - Memory Init
  - CS Init
  - Boot Mode
    - S3 Resume
    - Recovery
  - Normal Boot

- DXE
  - POST Dispatch
  - Console Init
  - Device Init
  - Bus Init

- EFI/DXE Drivers
  - Boot Dev Select
  - Legacy OS Load
  - OS Runtime

- EFI Boot Services
POST Execution Flow – High Level

Boot Execution Flow

- Platform Policy
- EFI Boot Manager
- Human Interface HII
- Boot Dev Select
- Legacy OS Load
- BDS
- EFI Pre-boot Application
- OS Runtime

Flow Diagram:
- POST Dispatch
- EFI Pre-boot Application
- Legacy OS Load
- CPU Init
- Memory Init
- CS Init
- Boot
- Normal Boot
- Recovery
- S3 Resume
- S3 Recovery
- Device Init
- Bus Init
- Console Init
- Device Init
- Boot Dev Select
- BDS
- Legacy OS Load
- OS Runtime

UEFI Training 2009

Copyright © 2006-2009 Intel Corporation
*Other trademarks and brands are the property of their respective owners

Slide 22
POST Execution Flow – High Level

Boot Execution Flow

Compatibility Support Module CSM

EFI Preboot

EFI Runtime Services

 Legacy OS Load

 EFI Pre-boot Application

 OS Runtime

Reset

CPU Init

Memory Init

CS Init

Boot Mode

S3 Resume

Recovery

POST Dispatch

Console

Boot Dev

Normal Boot
Boot Execution Flow

Architecture POST Execution Flow – Summary

Pre EFI

Initialization

(PREI)

Driver Execution Environment

(DXE)

Boot
Dev
Select

(BDS)

Transient
System Load

(TSL)

After
Life
(Al)

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

UEFI Training 2009

Copyright © 2006-2009 Intel Corporation

*Other trademarks and brands are the property of their respective owners
Agenda

• Design Approach
• UEFI Spec. and Platform Initialization Spec.
• Boot Execution Flow
• Overview of Each Phase
  – Pre-EFI Initialization (PEI)
  – Driver eXecution Environment (DXE)
  – Boot Device Selection (BDS)
  – Backward Compatibility (CSM)
• Development Environment
• Summary
Pre-EFI Initialization Foundation
Driver Execution Environment

**DXE Foundation Overview**

Most features during DXE phase implemented as DXE drivers
Boot Device Selection

Boot Device Select Overview

- Pre EFI Initialization (PEI)
- Device, Bus, or Service Driver
- Boot Device Selection (BDS)
- Transient System Load (TSL)
- Final OS Boot Loader
- Security (SEC)
- Pre EFI Initialization (PEI)
- Driver Execution Environment (DXE)
- Boot Device Select (BDS)
- Transient System Load (TSL)
- Run Time (RT)
- After Life (AL)

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

UEFI Training 2009

Copyright © 2006-2009 Intel Corporation
*Other trademarks and brands are the property of their respective owners
Transient System Load

Power on → [Platform initialization] → [OS boot] → Shutdown
Backward Compatibility

Compatibility Support Module

Chipset/Processor Function DXE Driver specs

OEM, ISV & Intel BU EFI Driver specs

Driver Execution Environment (DXE) Spec

Pre-EFI Initialization (PEI) Spec

CPU Module Spec(s)  CS Module Spec(s)

Hardware/Driver

Architecture Specification
Run Time

Power on ➔ [ . . Platform initialization . . ] ➔ [ . . . . OS boot . . . . ] ➔ Shutdown
Agenda

- Design Approach
- UEFI Spec. and Platform Initialization Spec.
- Boot Execution Flow
- Overview of Each Phase
  - Pre-EFI Initialization (PEI)
  - Driver eXecution Environment (DXE)
  - Boot Device Selection (BDS)
  - Backward Compatibility (CSM)
- Development Environment
- Summary
Development Environment

- Open Source - EDI I and EDK II
- Practical approach: good tools make developers’ lives easier
- Commercial C compiler
- Self-hosted development of drivers
- Libraries created for common tasks
- Source language debugging
- Connection to HW debuggers possible back to source code
- Even extends to SMI environment
“Burden the Tools”

- Automated “make” generation
  - Build configuration file selects components for image
  - Config file drives construction of make hierarchy
- Where possible, use tools in the build environment to catch problems
  - Avoid carrying error handling code in ROM
  - E.g. static check to catch driver dependency loops
- Test coverage: build in flow checking
  - Dump of paths visited during test run
  - Graphical tool to show code paths without test coverage
- Performance instrumentation
  - Time spent in major phases and individual modules
  - Helps focus boot speed optimization on real problems
Debug Support

- Source level, “C”, debugging of components and applications with hardware emulators
- Debug Text Output
  - Allows formatted text printing to console and/or serial stream
  - Early in PEI, all of DXE
- Assertions
- Status Codes
  - Allows logging (to DataHub) of progress, error, and debug codes and optional data.
  - Output to Port80 driver
  - Early in PEI, all of DXE
Agenda

- Design Approach
- UEFI Spec. and Platform Initialization Spec.
- Boot Execution Flow
- Overview of Each Phase
  - Pre-EFI Initialization (PEI)
  - Driver eXecution Environment (DXE)
  - Boot Device Selection (BDS)
  - Backward Compatibility (CSM)
- Development Environment
- Summary
Architecture POST Execution Flow – Summary

- Pre EFI Initialization (PEI)
- Chipset Init
- Board Init
- Device, Bus, or Service Driver
- EFI Driver Dispatcher
- Intrinsic Services
- Boot Manager
- OS-Absent App
- Transient OS Environment
- Transient OS Boot Loader
- OS-Present App
- Final OS Environment
- Final OS Boot Loader

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

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

Copyright © 2006-2009 Intel Corporation
*Other trademarks and brands are the property of their respective owners*
Summary

• Not just a new “core” code base…a new purpose-built architecture for firmware
• Backward compatibility allows for a non-disruptive industry transition
• Modularity, C and architectural stability provides the environment for industry innovation
• Allows ODM to customize product for customer needs
• The shape of platform “BIOS” enabling to come
• Available through participating BIOS vendors for the entire industry