Intel® Platform Innovation Framework for EFI and Platform Initialization Overview

Intel Corporation
Software and
Services Group

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)
- Summary

Framework Technical Goals:

Kick the Ivory Tower Design Approach

- Architectural design to last a second 20 years
- Intel® IA-32, Itanium® architecture and Xscale® 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

Framework Design Strategy

- High level design based on Framework plus modular components
- Generalize the Framework
  - 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

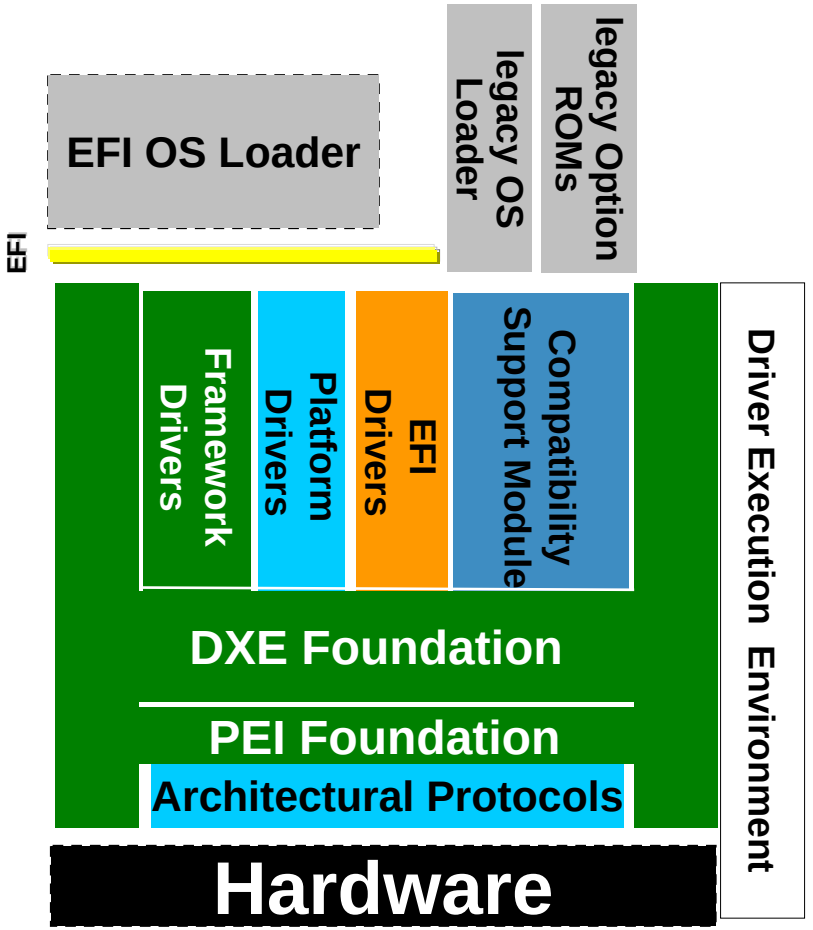
Get to "C" Code Quickly

- Commercial "C" compilers use stack model
  - Requires some memory initialized for a stack
- Split Framework 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 Framework finds memory by using special stack
  - Infrastructure code plus PEI Modules
- Framework 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

What is the Framework?

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



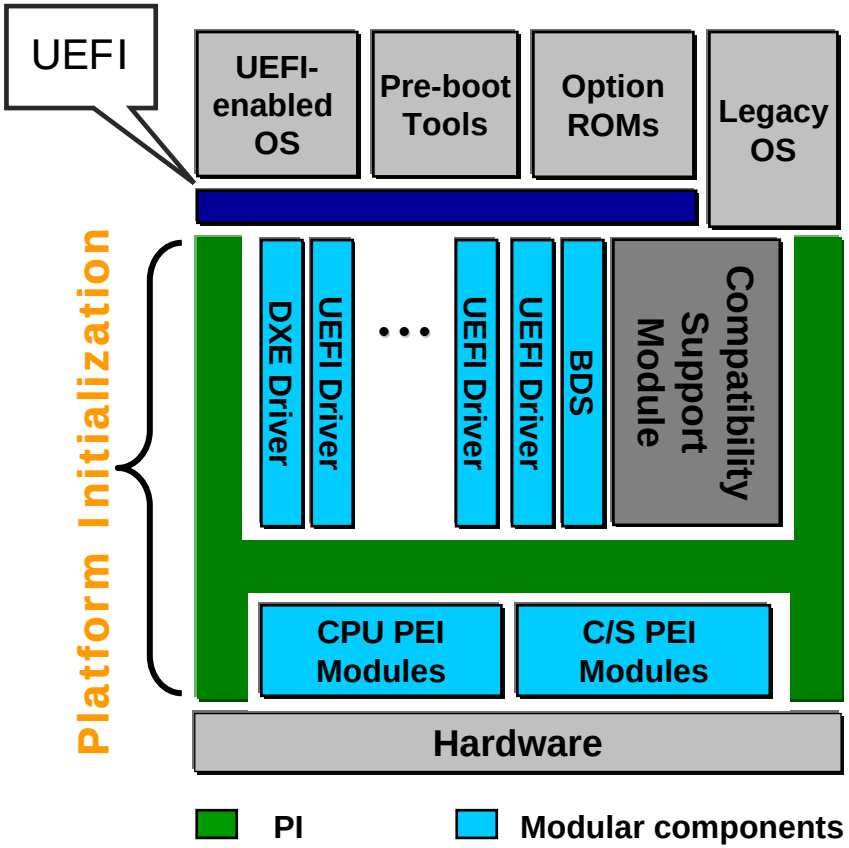
Agenda

- 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)
- Summary

…Enter Unified EFI Forum

- 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

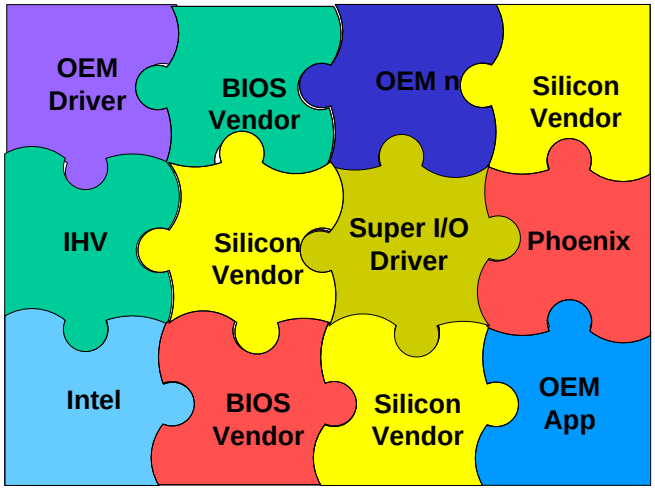


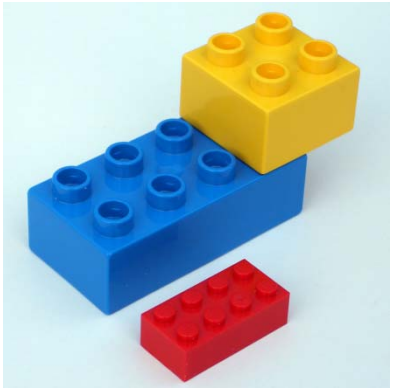
- UEFI Spec is about interfaces between OS, add-in driver and system firmware
  - a new model for the interface between the Operating systems and other high-level software and the platform firmware
- PI Specs relate to making UEFI implementations
  - Promote interoperability between firmware components 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 Intel PEI and DXE Framework specifications
- Board approved formal PI 1.0 Oct, 2006
  - Available for download from http://www.uefi.org
- Starting work from Intel contributed specifications
  - Framework DXE and PEI Core Interface Specs
  - Firmware Storage, HOB, SMBus
- PI Self Certification Test (SCT) available on http://sourceForge.net/projects/pi-sct

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

	Component	Actions / Exceptions
✓	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 Remove Capsule HOB definition

PI 1.0 Introduces Standards To Early Boot

Overview of Differences – PI 1.1 Vs. Framework Components

	Component	Actions / Exceptions
✓	SMM	SMM driver model change. Port code
✓	S3	New execution requirements for native callbacks. Some interfaces removed
✓	PCI	New event. Should be able to enhance former implementation
✓	MP	Clean-up. Port to use new API
✓	DXE	Cleaned-up DXE to be UEFI 2.0 RT compatible
✓	SMBIOS table creation	Framework used data hub. This is a new API

PI 1.1 Expands PI 1.0 infrastructure w/ more building blocks

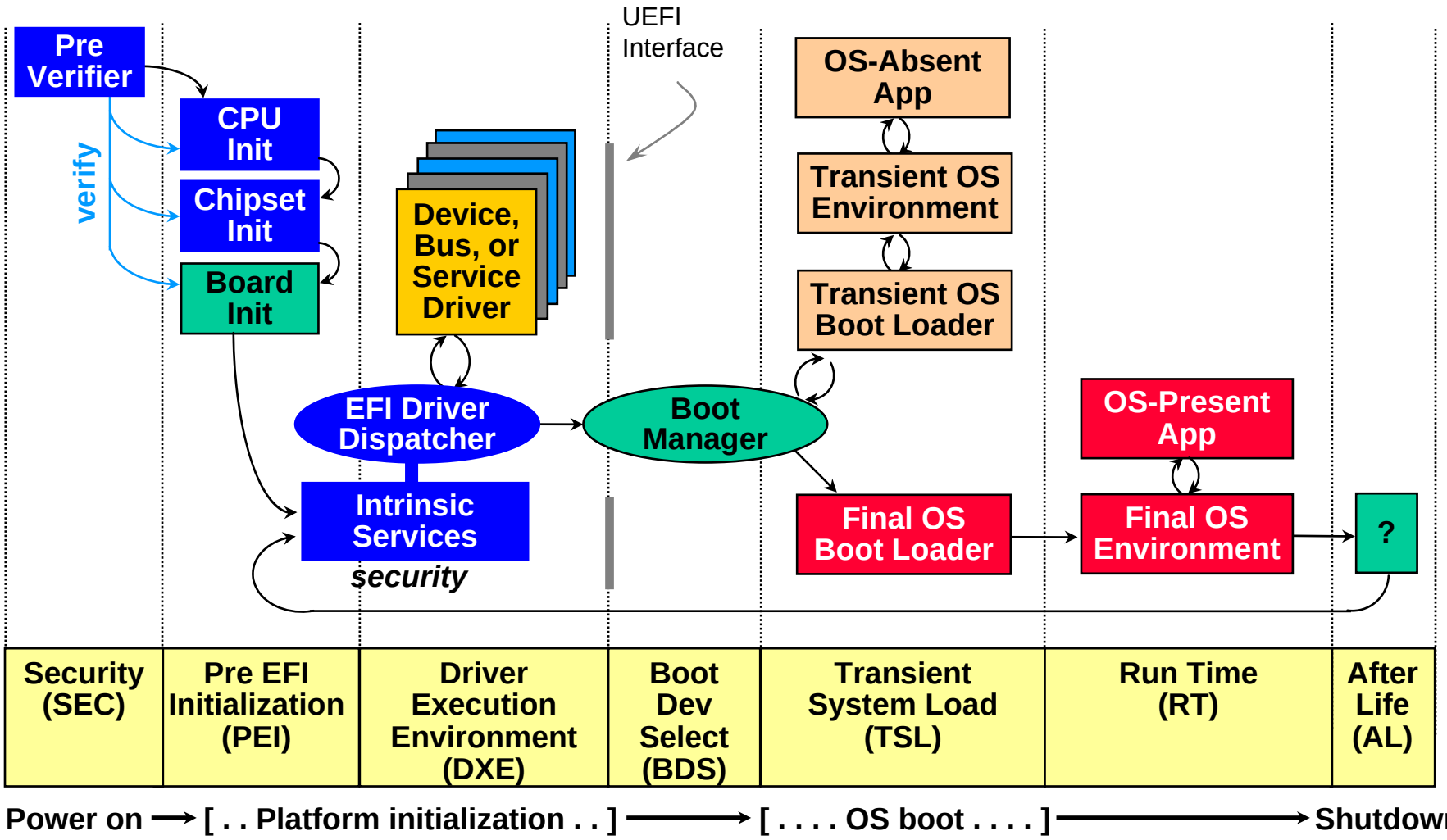
PI 1.2 Work In Progress

- Based on existing Intel Framework Specifications
  - Highest priority subset chosen
- PI Configuration
- Driver distribution
- Error/status reporting
- ….

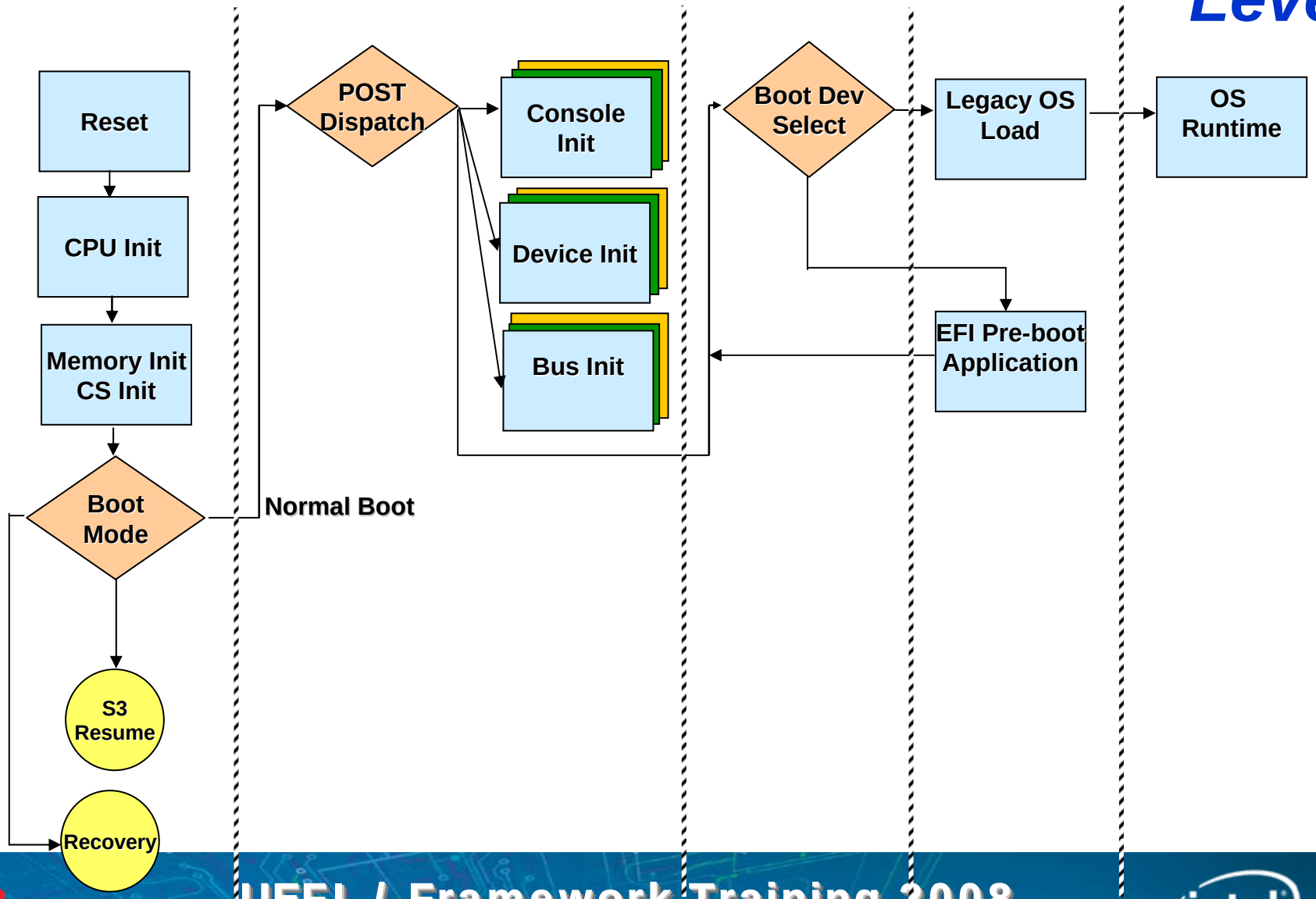
Agenda

- 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)
- Summary

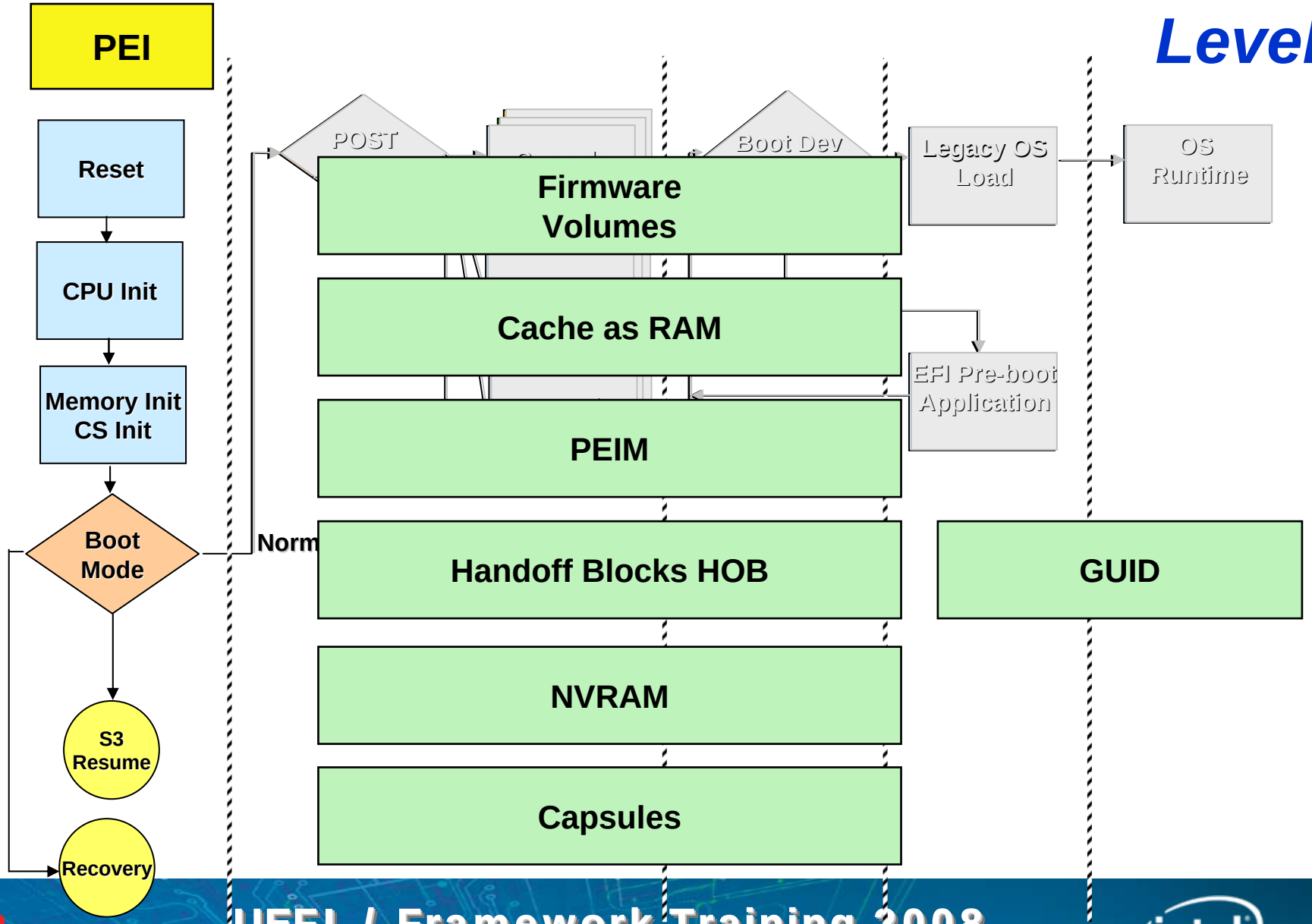
Framework Architecture Execution Flow



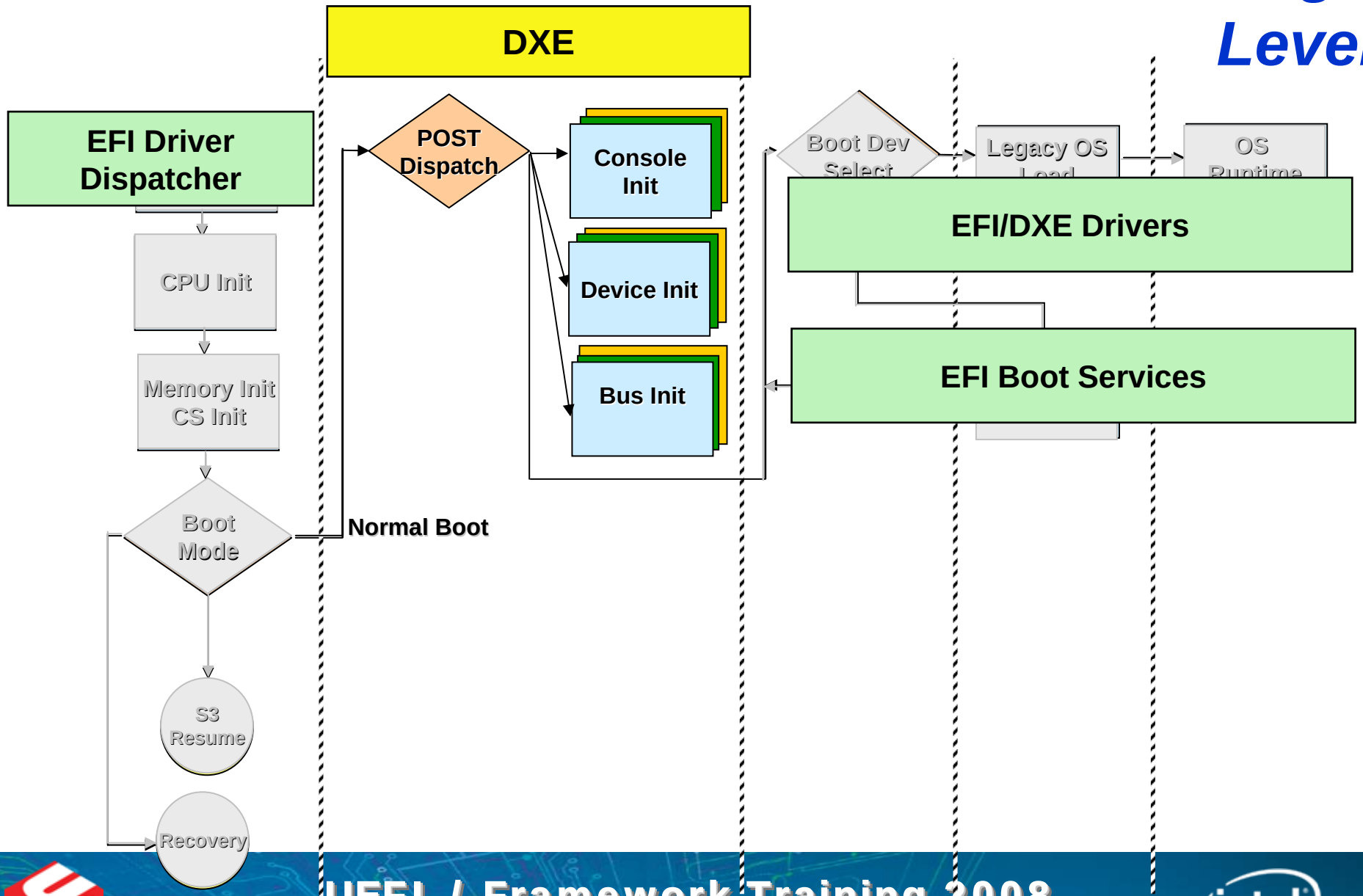
POST Execution Flow – High Level



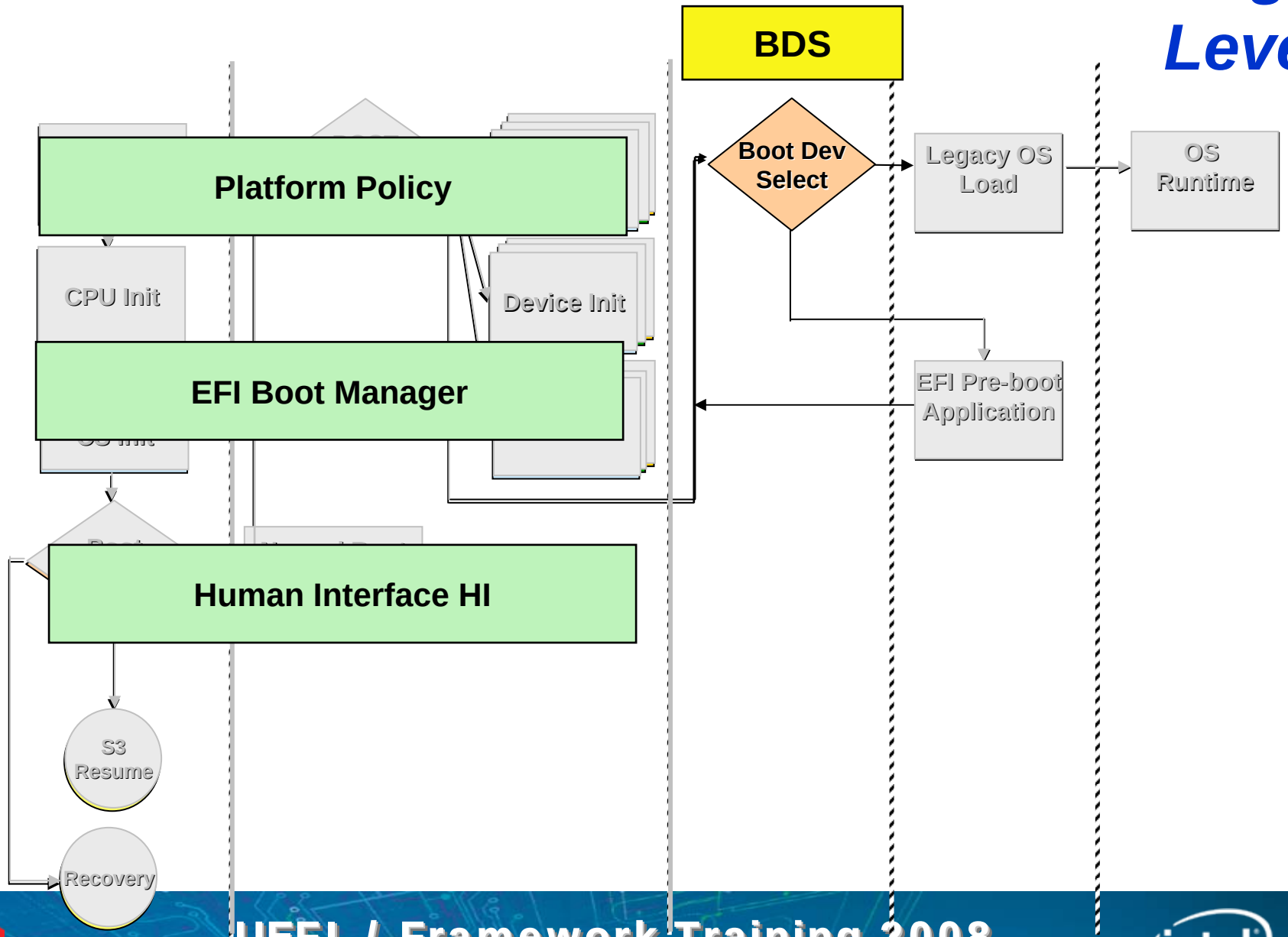
POST Execution Flow – High Level



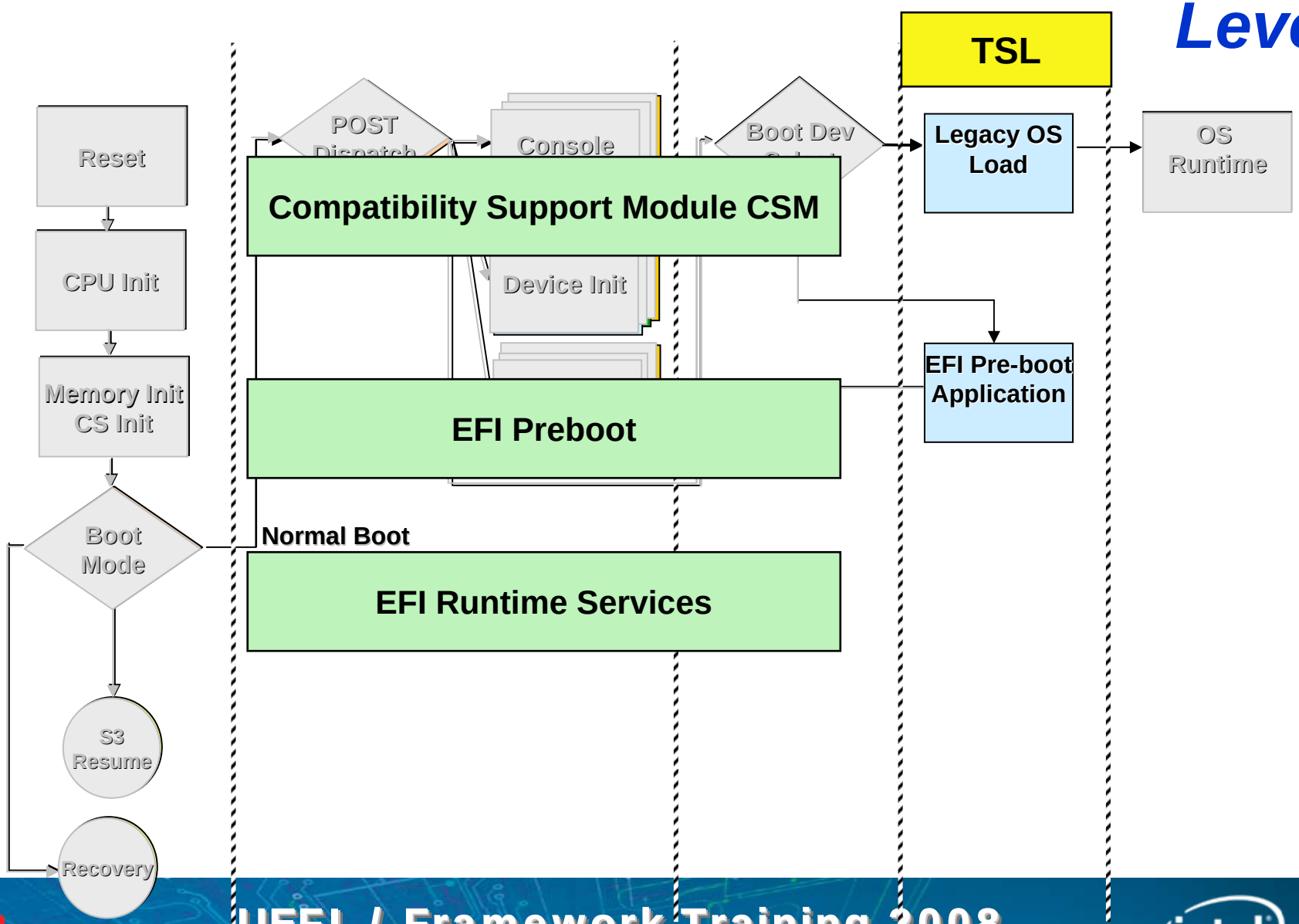
POST Execution Flow – High Level



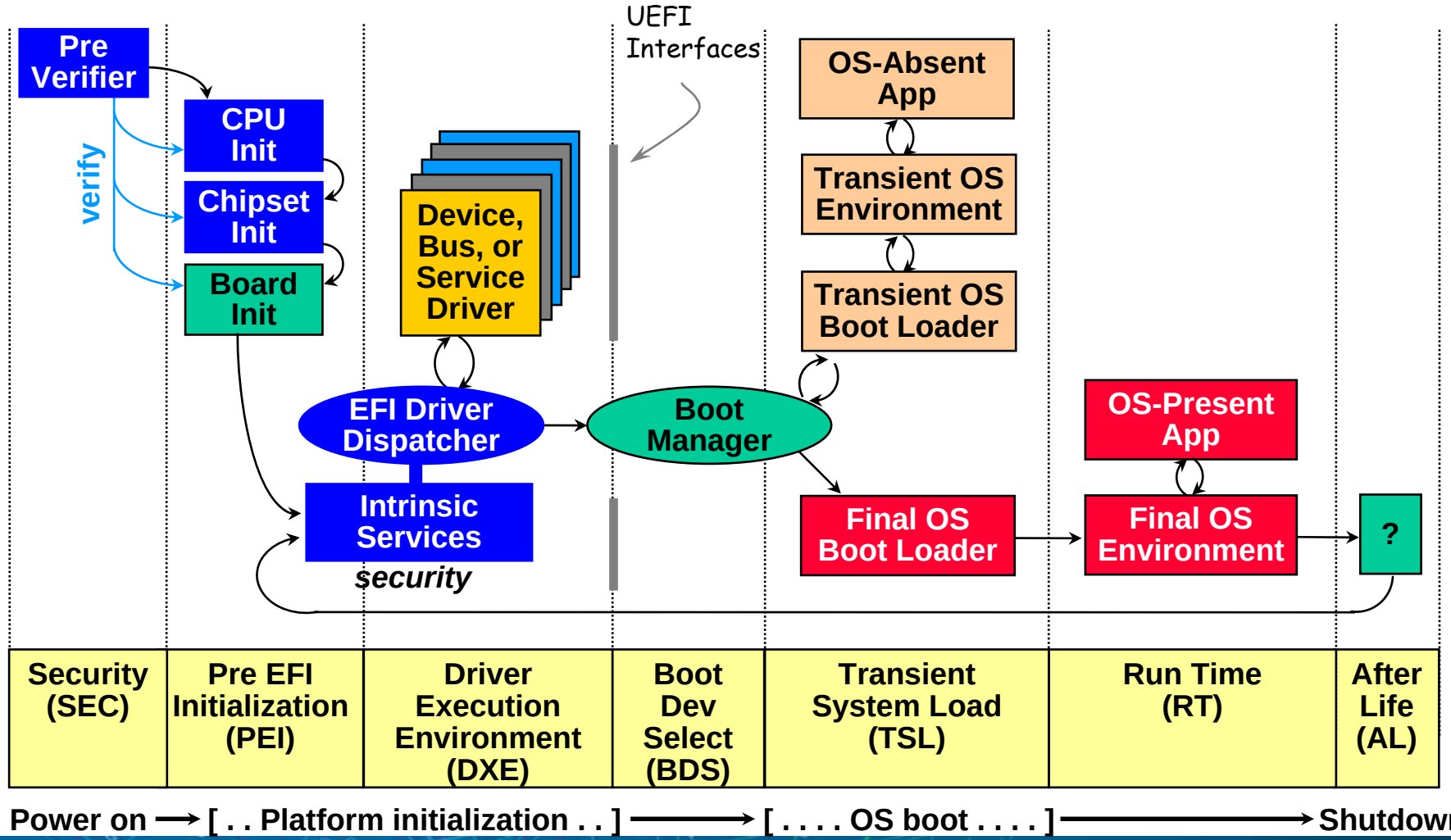
POST Execution Flow – High Level



POST Execution Flow – High Level



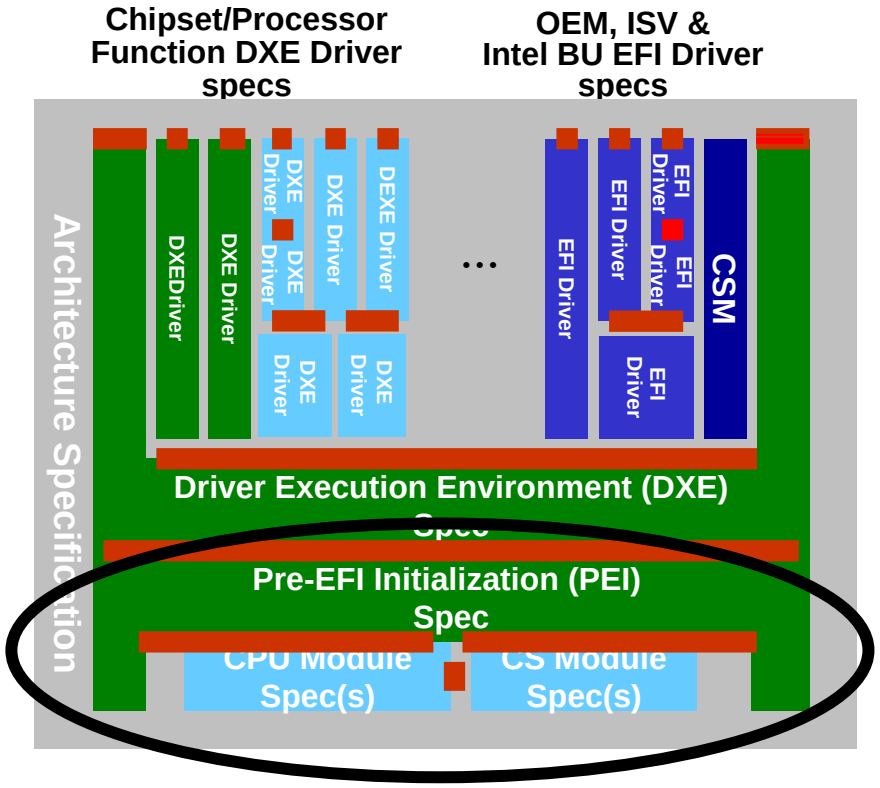
Framework Architecture POST Execution Flow – Summary

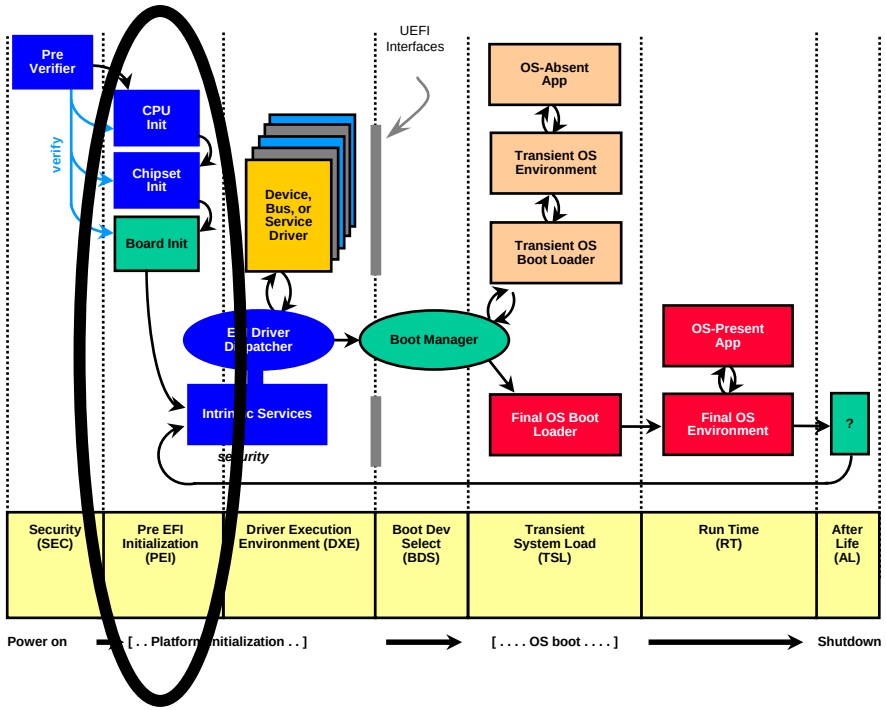


Agenda

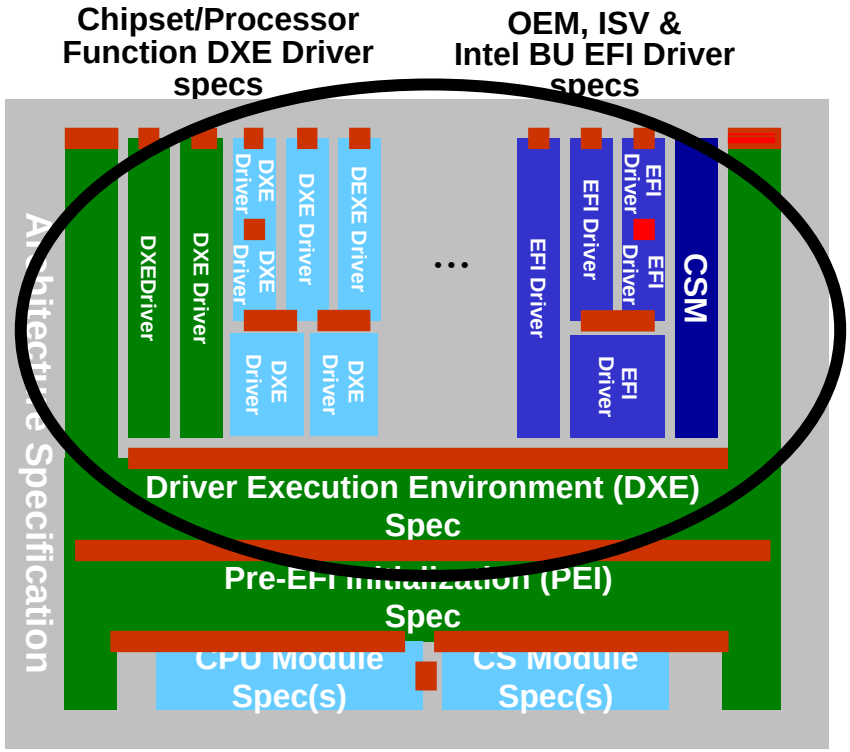
- Design Approach
- UEFI Spec. and Platform Initialization Spec.
- Boot Execution Flow
- **Overview of Each Phase**
  - Pre-EFI Initialization (PEI)
  - Driver eXecution Environment (DXE)
  - Boot Device Selection (BDS)
  - Backward Compatibility (CSM)
- Summary

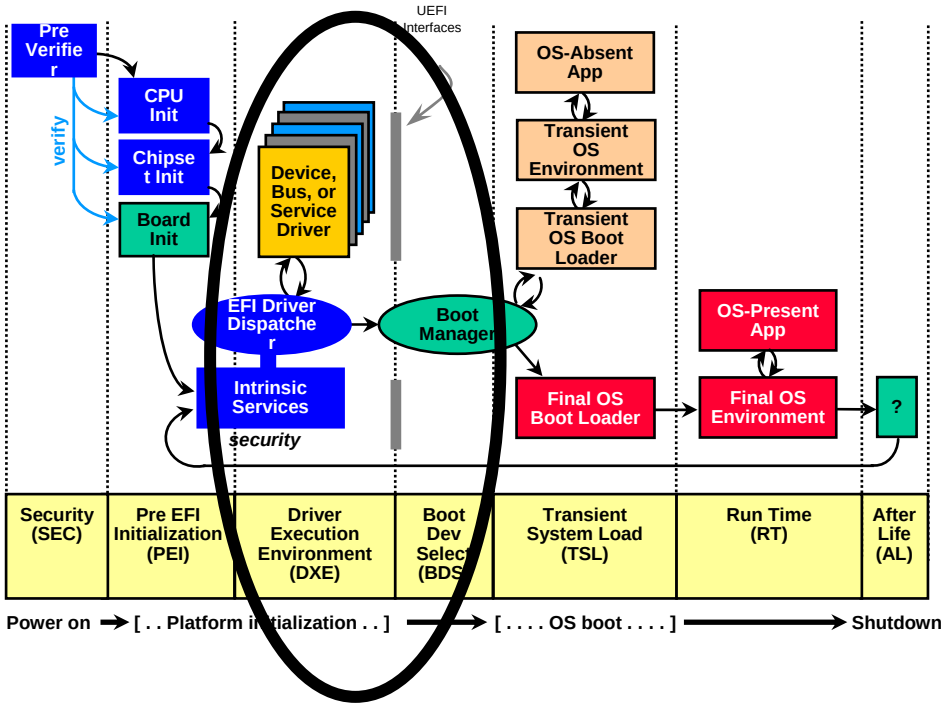
PEI Foundation





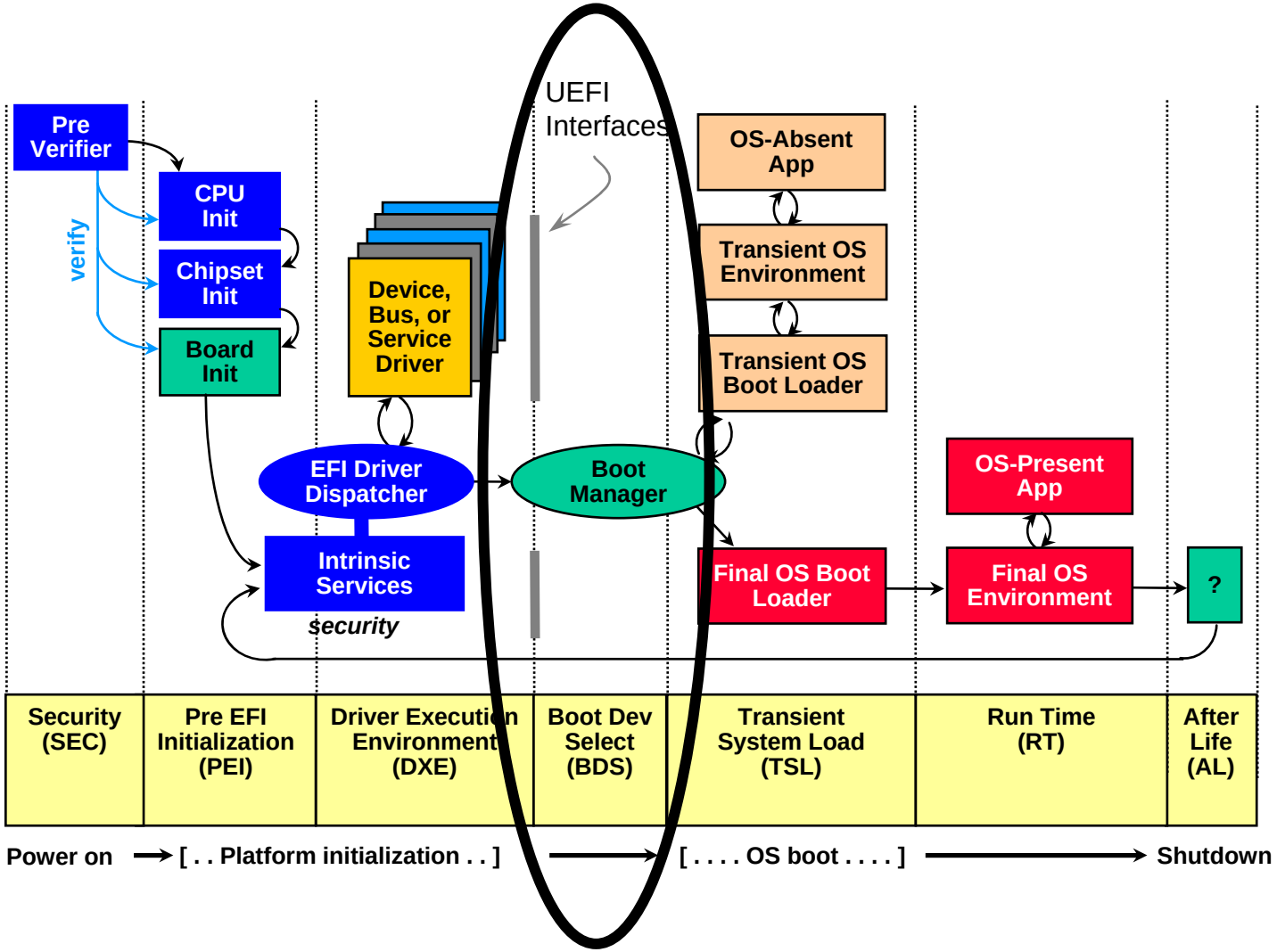
DXE Foundation Overview



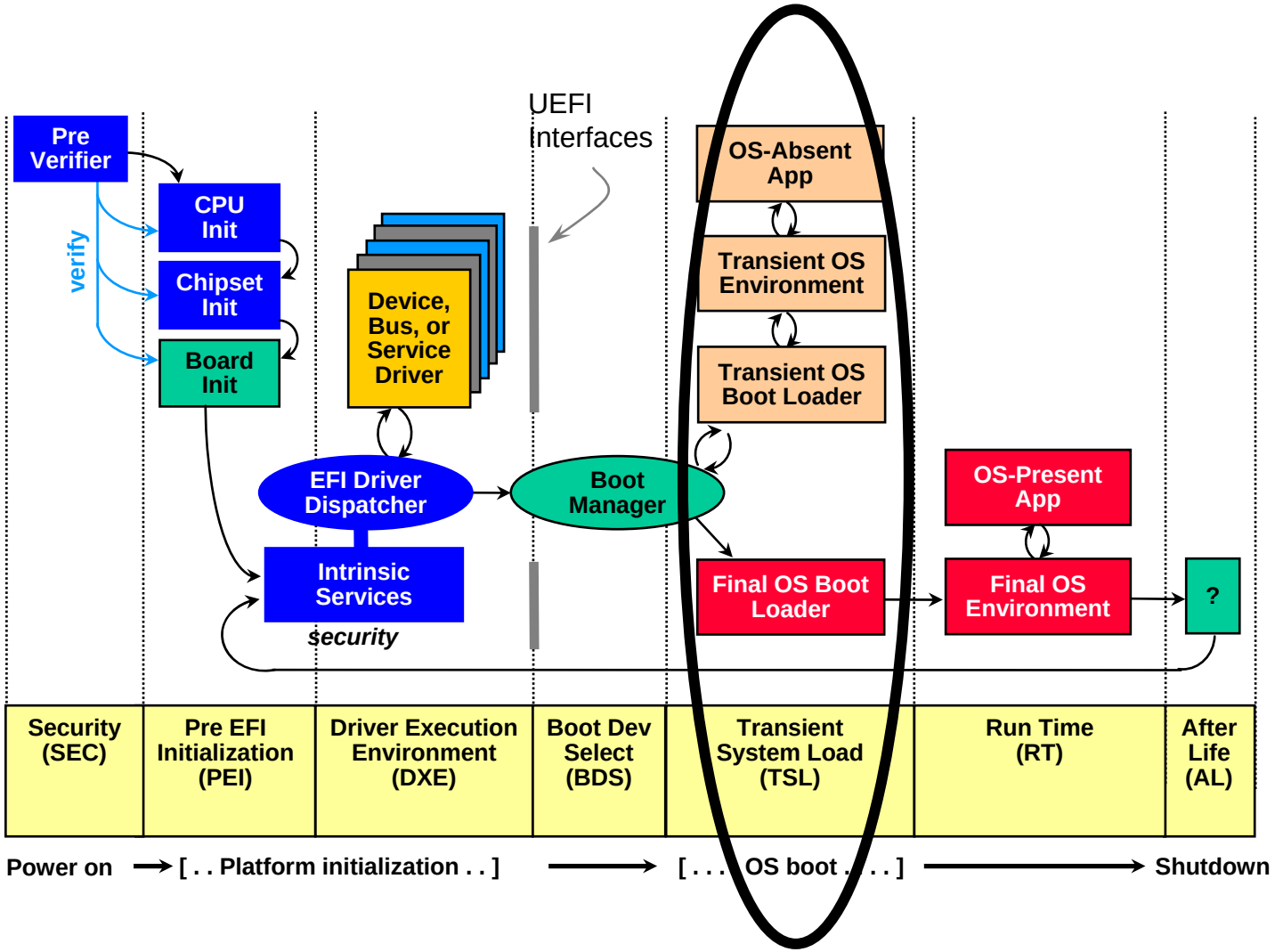


Most features during DXE phase implemented as DXE drivers

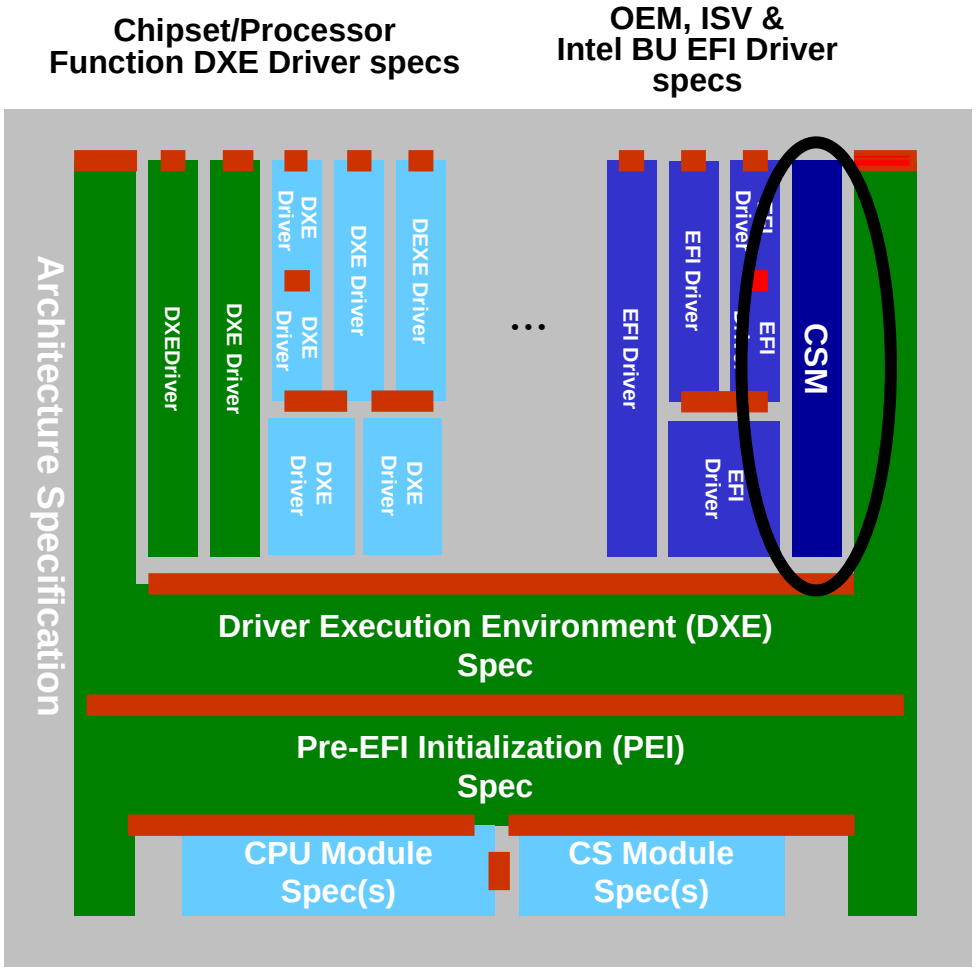
Boot Device Select Overview



Transient System Load



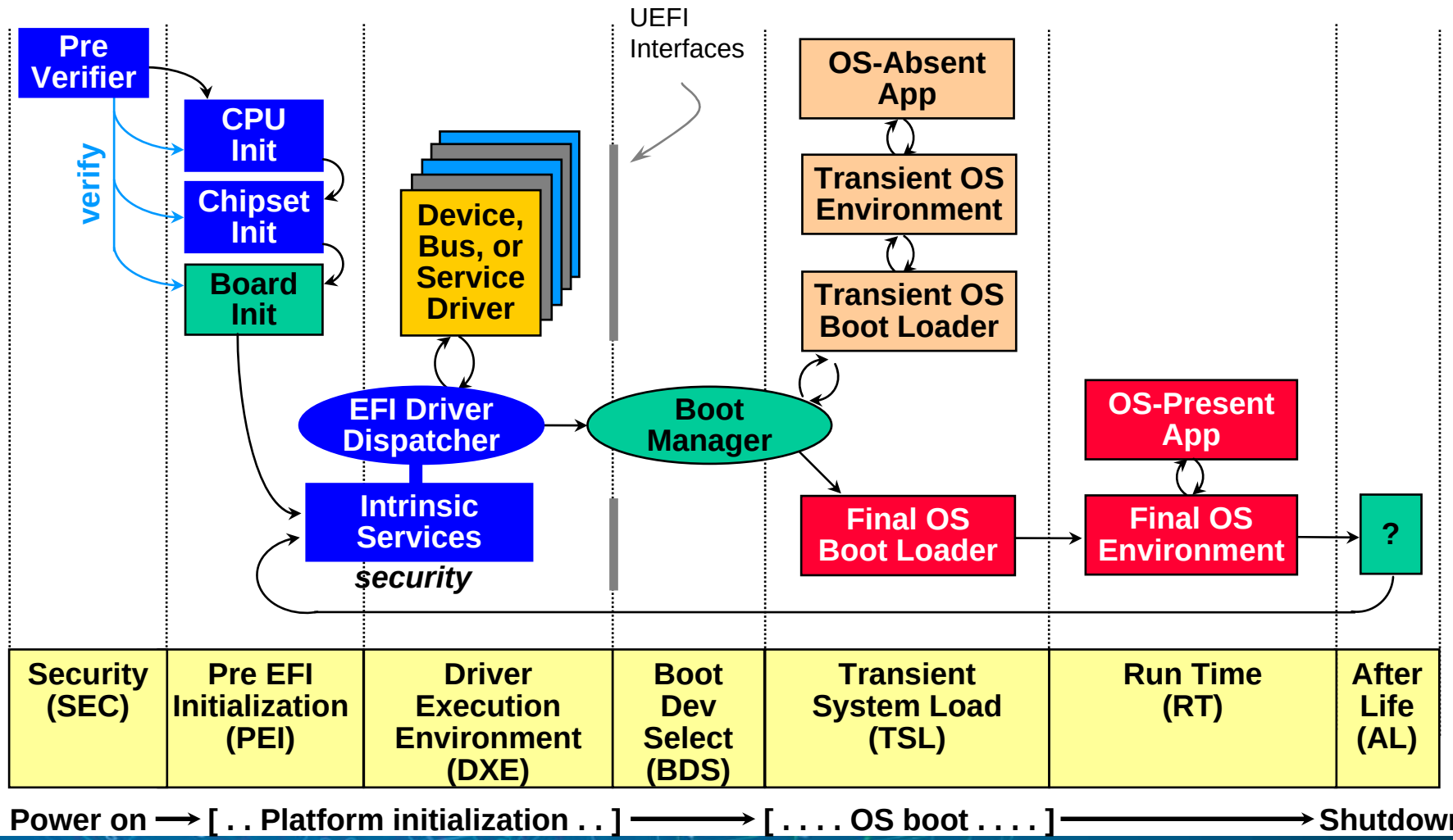
Compatibility Support Module



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)
- **Summary**

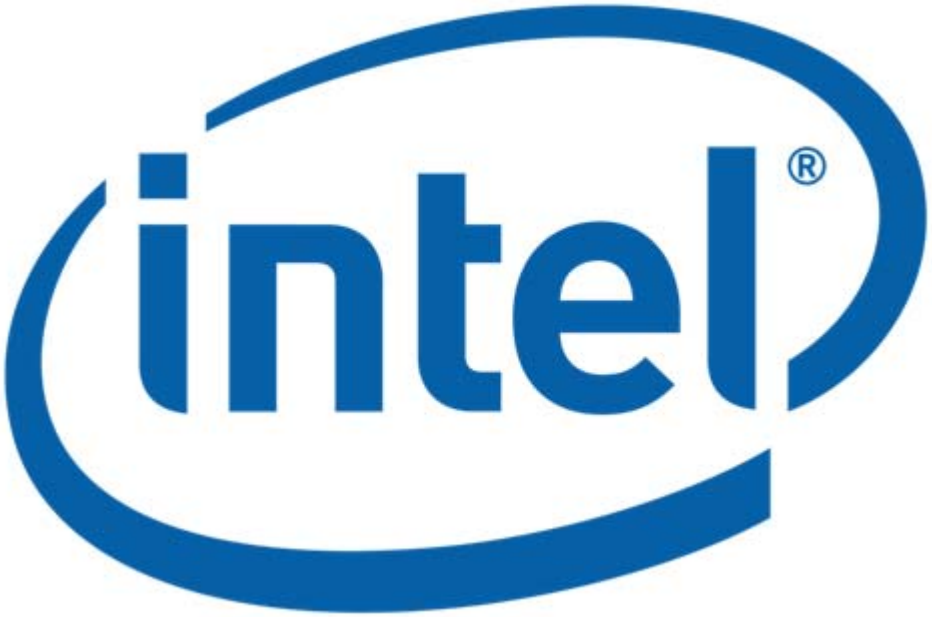
Framework Architecture POST Execution Flow – Summary



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

Q & A



Back up