MINIMUM PLATFORM

OPEN SOURCE UEFI FIRMWARE FOR INTEL BASED PLATFORMS

Nate DeSimone

Michael Kubacki

OSFC 2019 September 3rd, 2019

How to Build Intel UEFI FW For a System?

tianocore

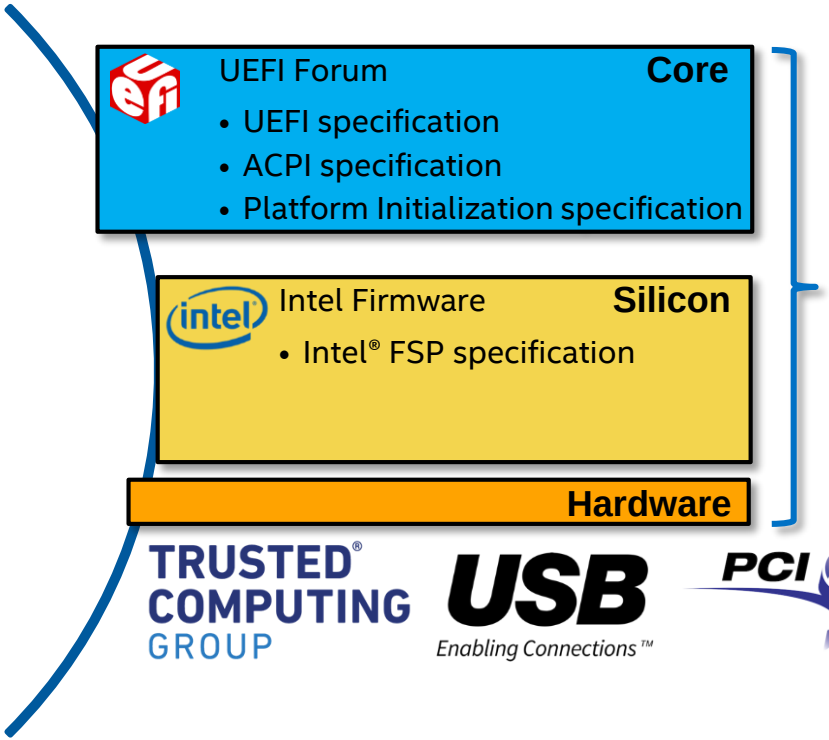
https://github.com/tianocore/edk2

intel

https://github.com/IntelFsp/FSP

Core	Silicon	Platform
• Typically open source. • Industry standard drivers. • Generic firmware infrastructure code.	• Typically closed source. • Has some tie to a specific class of physical hardware. • Sometimes governed by industry standards, sometimes proprietary.	• Typically closed source. • Advanced or platform feature code. • Board specific code for one or more motherboards.

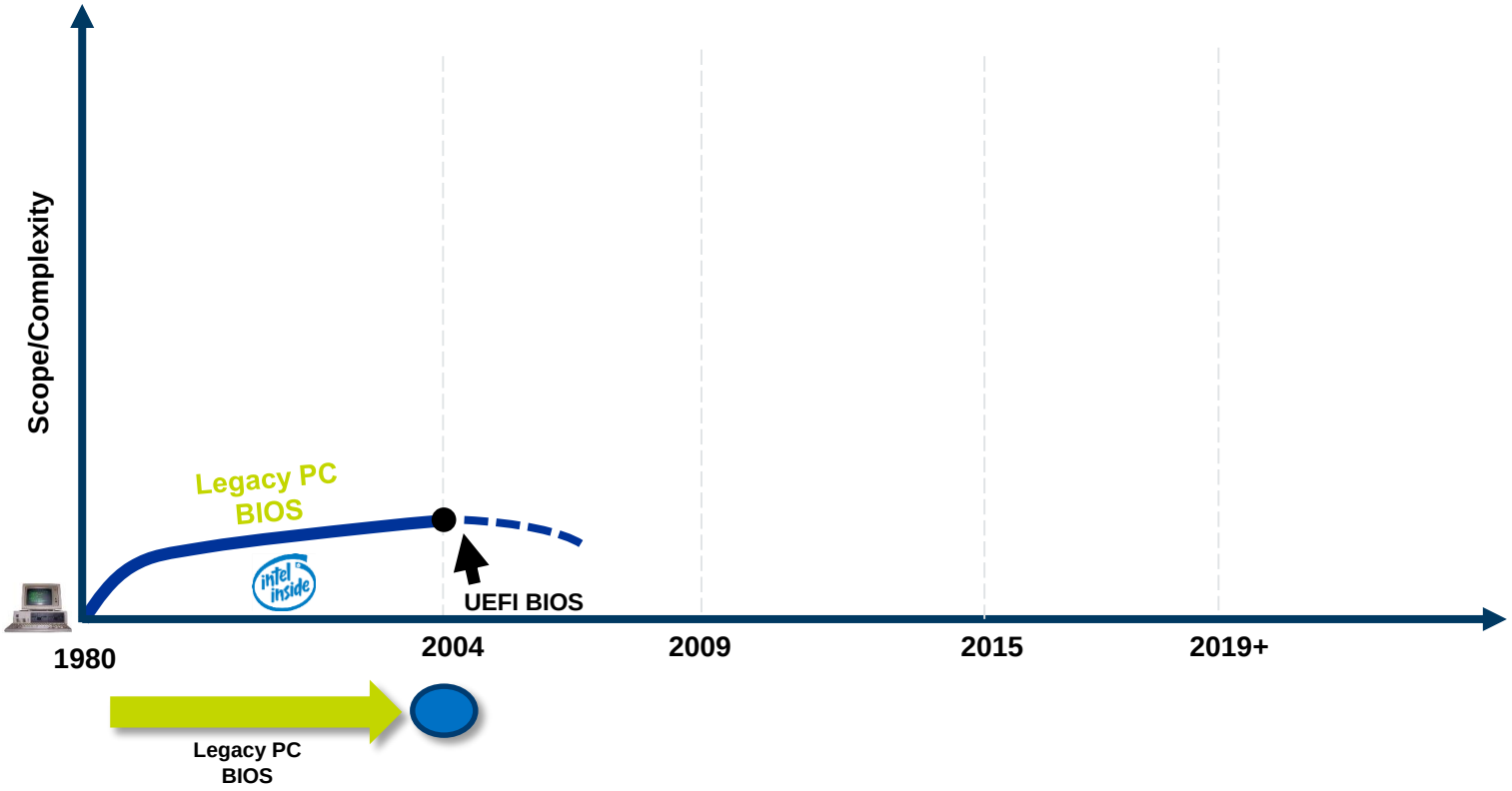
Firmware is Built on Standards



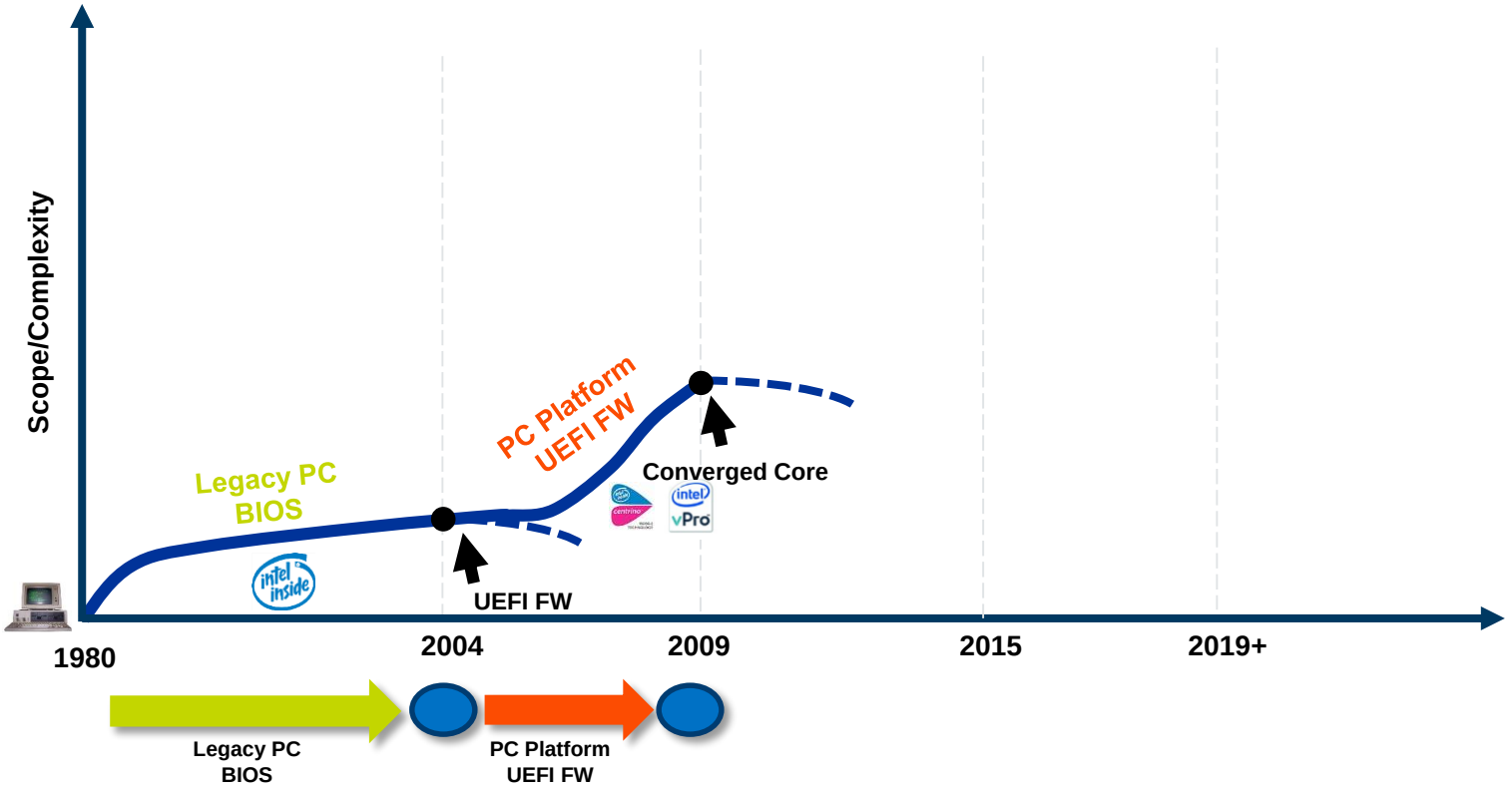
The platform code brings it all together

- Defines the firmware flash map
- Specifies the core and hardware drivers needed
- Calls into the silicon initialization API
- Provides board-specific settings like GPIO values, SPD settings, etc.

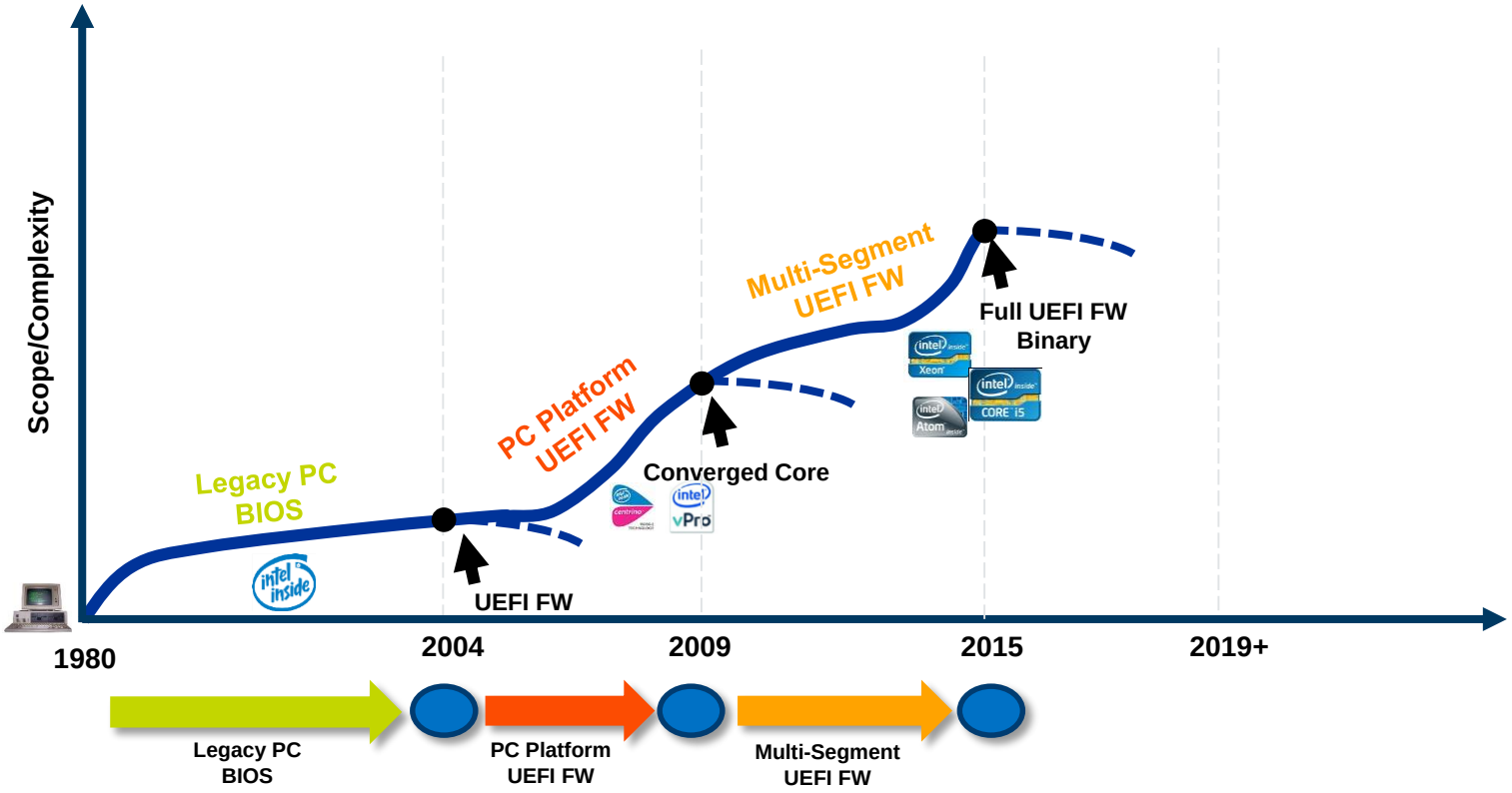
Intel® Legacy BIOS to UEFI



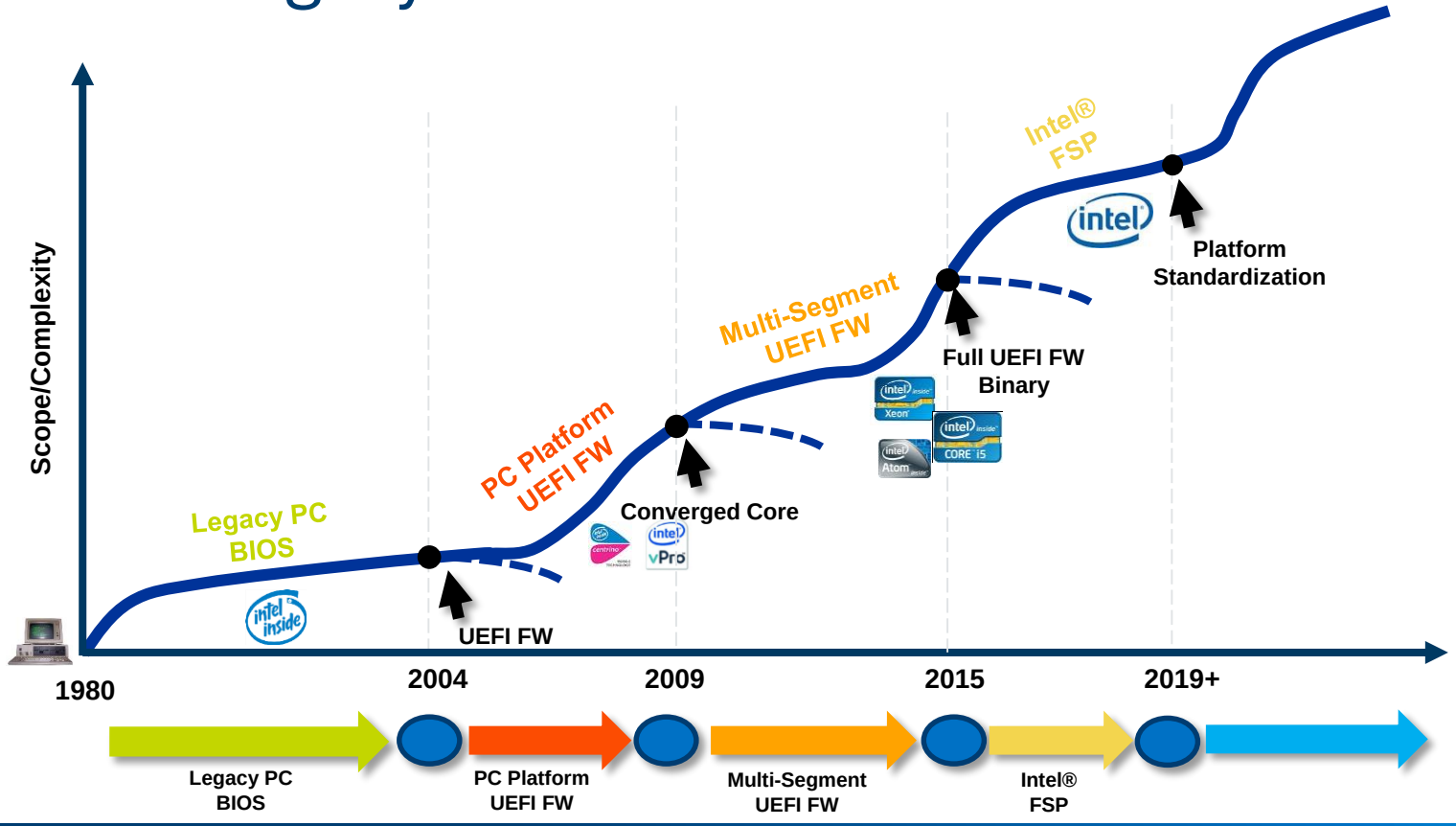
Intel® Legacy BIOS to UEFI



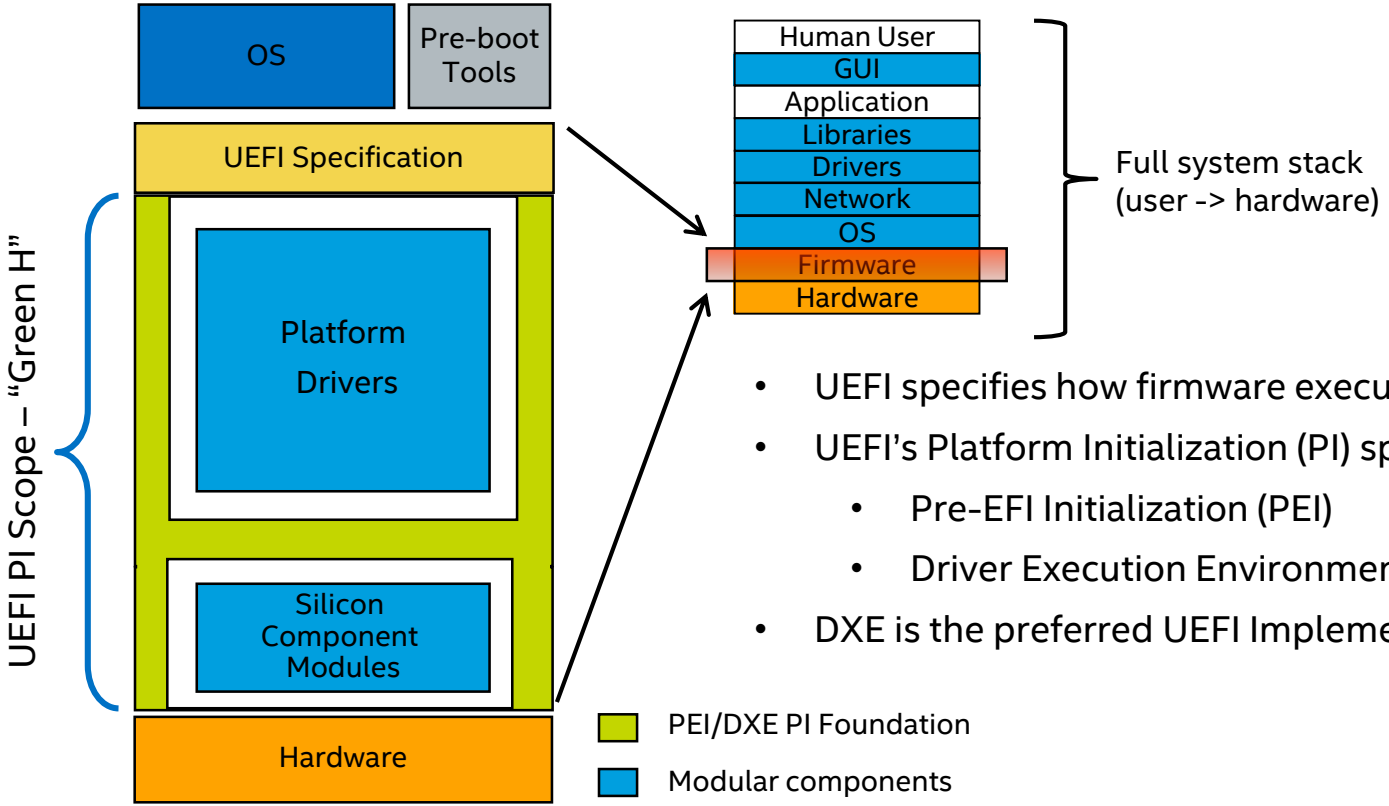
Intel® Legacy BIOS to UEFI



Intel® Legacy BIOS to UEFI



UEFI PI Overview



- UEFI specifies how firmware executes the OS loader.
- UEFI's Platform Initialization (PI) spec. defines:
  - Pre-EFI Initialization (PEI)
  - Driver Execution Environment (DXE)
- DXE is the preferred UEFI Implementation.

Silicon Initialization Overview

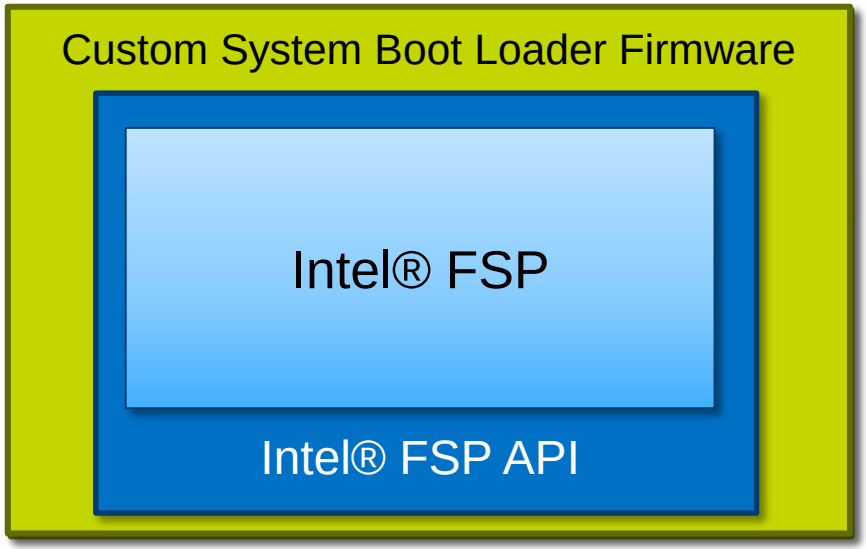
- Intel® FSP is a binary distribution of Intel's silicon initialization code.
  - The resources necessary to implement Intel silicon code are not publicly available.

- Intel's FSP Strategy:
  1. Distribute binaries of our proprietary silicon code to the public
  2. Enable this binary to plug into arbitrary firmware designs (coreboot, TianoCore, etc.)

  - Secondary goal is to abstract the complexity of silicon initialization.
  - Expose a limited number of well-defined interfaces.

Intel® FSP 2.0

- Intel® FSP is treated as a binary blob.
- Intel® FSP header provides 32-bit entry points to APIs defined in the FSP specification.
- Intel® FSP specification defines data structures used for input and output from the API functions.



Intel® FSP 2.1

- **Primary objective:** Seamlessly integrate with UEFI PI firmware

Now we need an open source UEFI PI platform wrapper for Intel® FSP…

- **Non-UEFI PI Firmware: API mode**
  - Same “mode” provided with 2.0 using the same binary API
  - Uses UPDs for configuration
- **UEFI PI Firmware: Dispatch mode (* New)**
  - The FSP wrapper uses Intel® FSP the same as any other firmware file system partition
  - Directly uses UEFI PI architecture executables

Takeaway: Dispatch mode can improve efficiency if a UEFI PI wrapper is used.

Lack of Platform Code Consistency

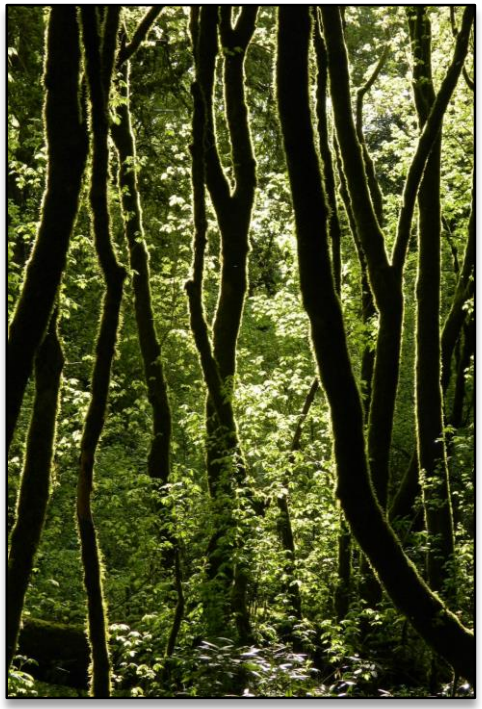
Platform code today needs work to encourage collaboration.

- It is designed with a specific device and segment in mind.
  - Lacks feature modularization
- It is difficult to understand and debug.
  - Boot flows vary arbitrarily between systems
- It is difficult to secure.
  - Same thing done different ways

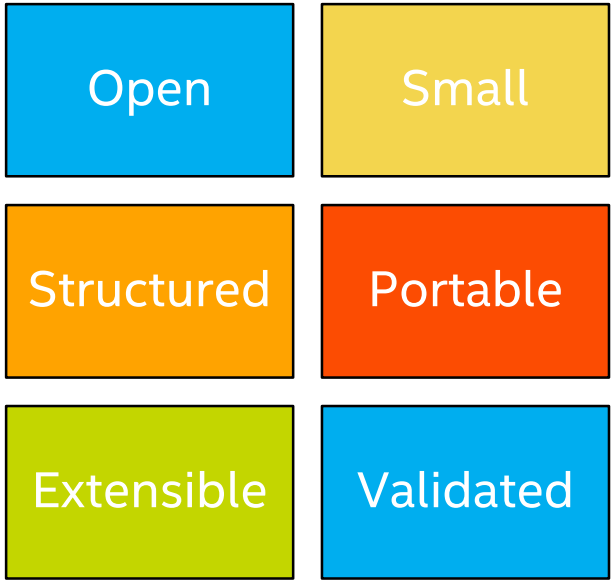
Server

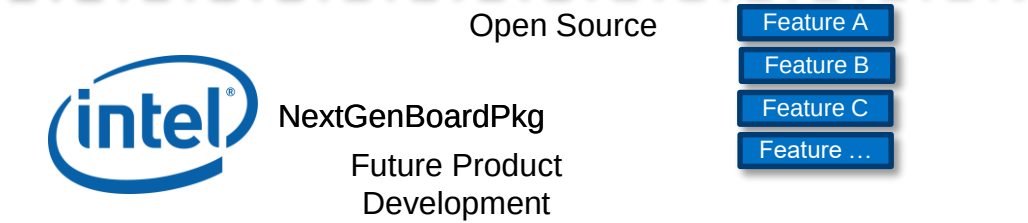
Client

Ultra Mobile

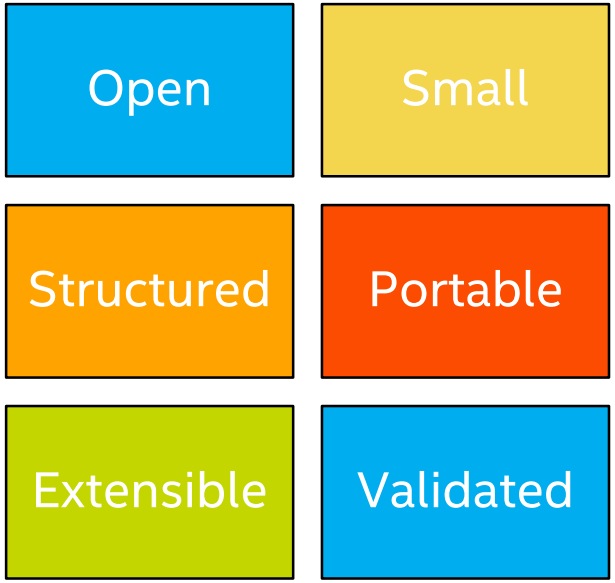


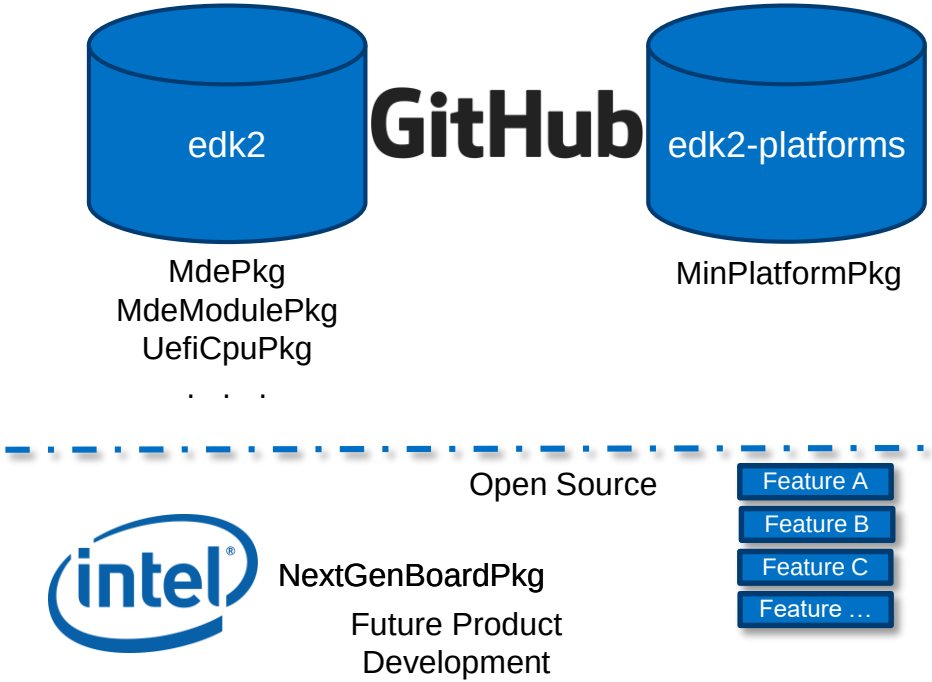
Evolving Intel Open Source Platform Firmware



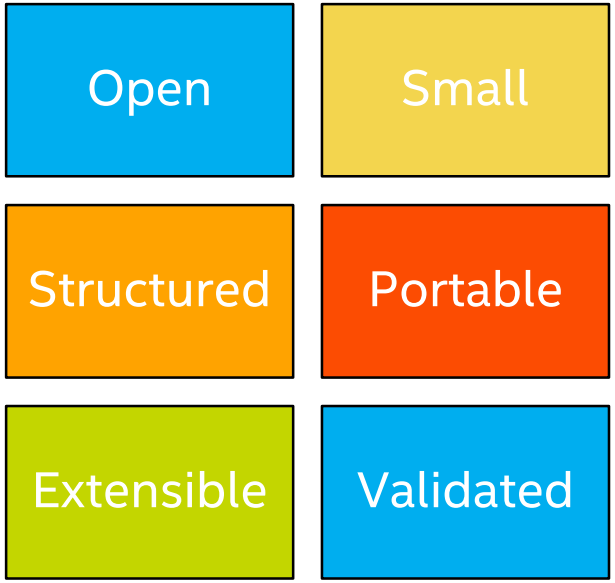


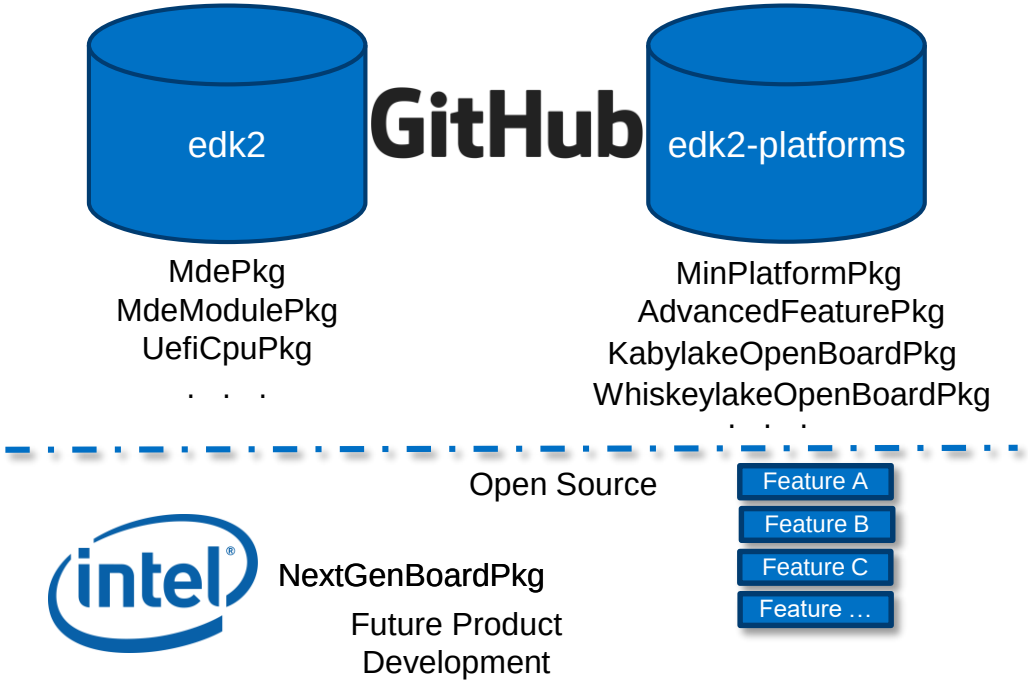
Evolving Intel Open Source Platform Firmware



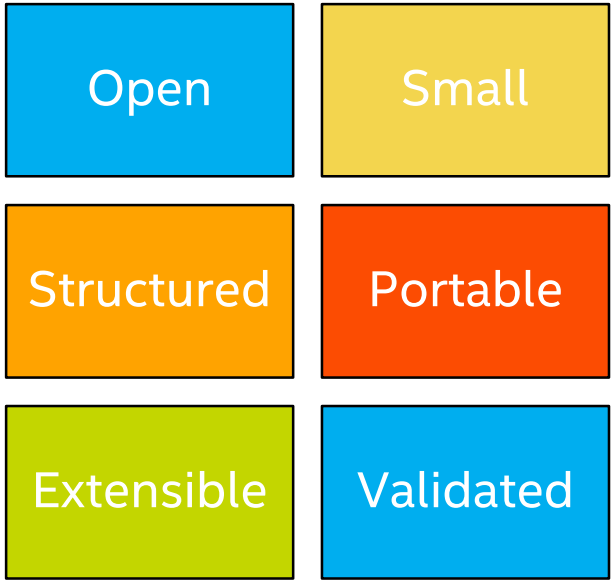


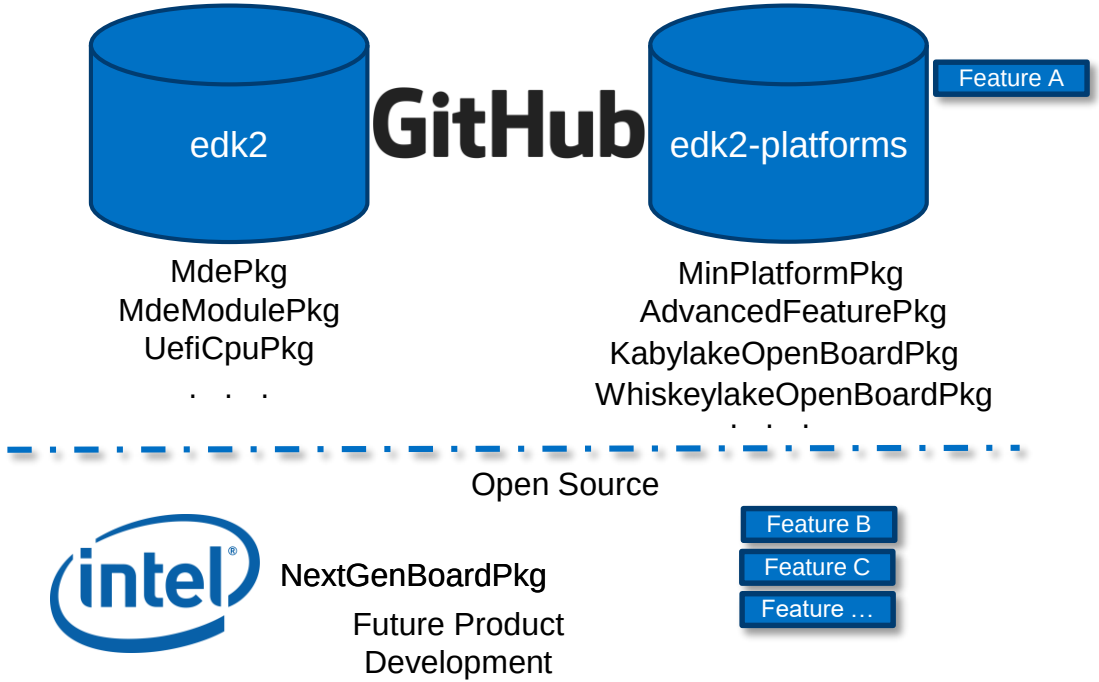
Evolving Intel Open Source Platform Firmware



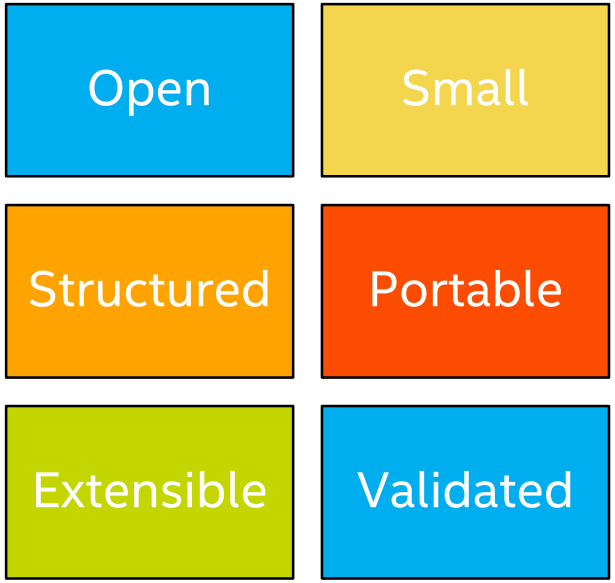


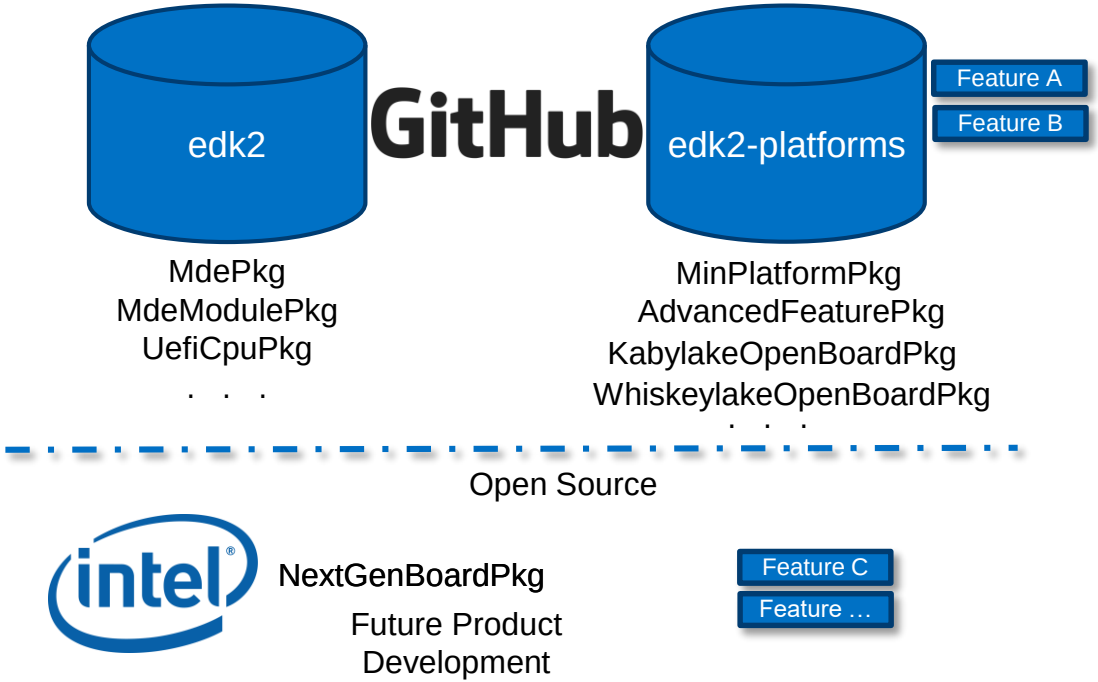
Evolving Intel Open Source Platform Firmware



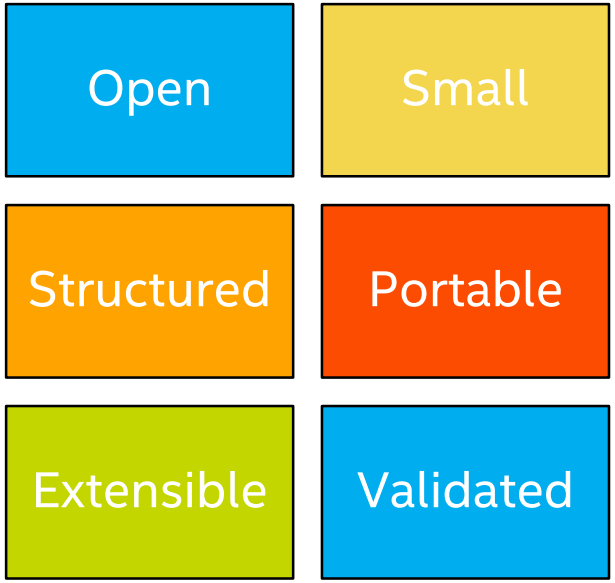


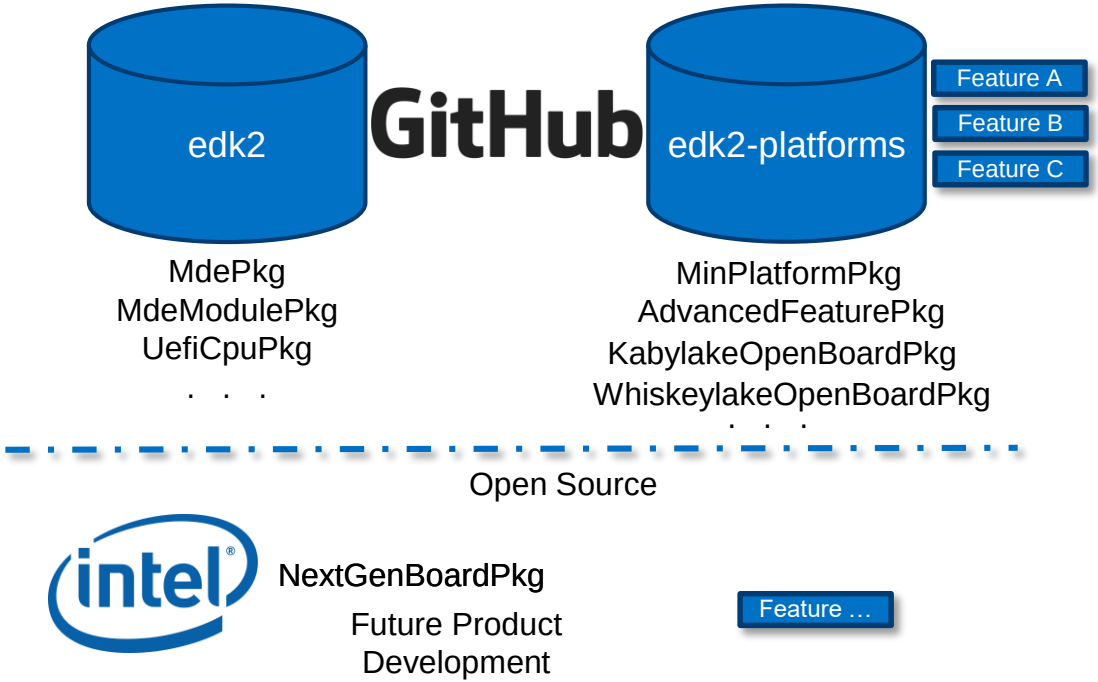
Evolving Intel Open Source Platform Firmware



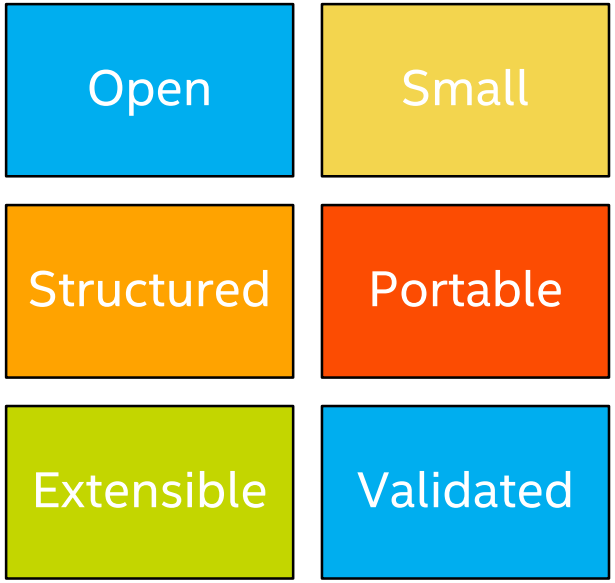


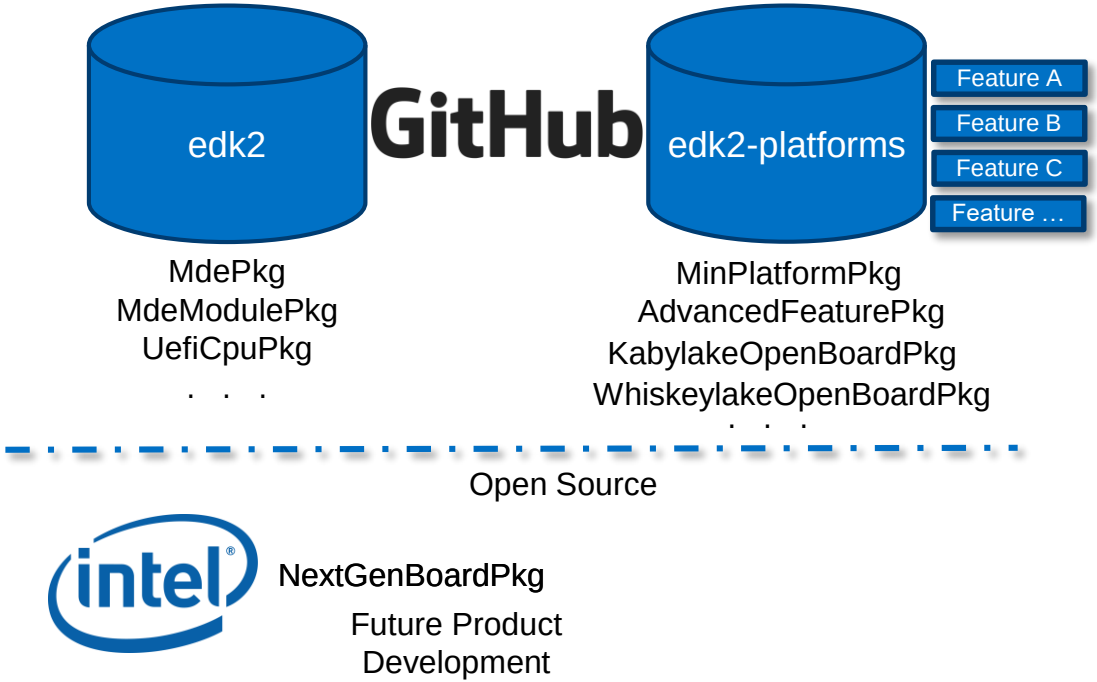
Evolving Intel Open Source Platform Firmware



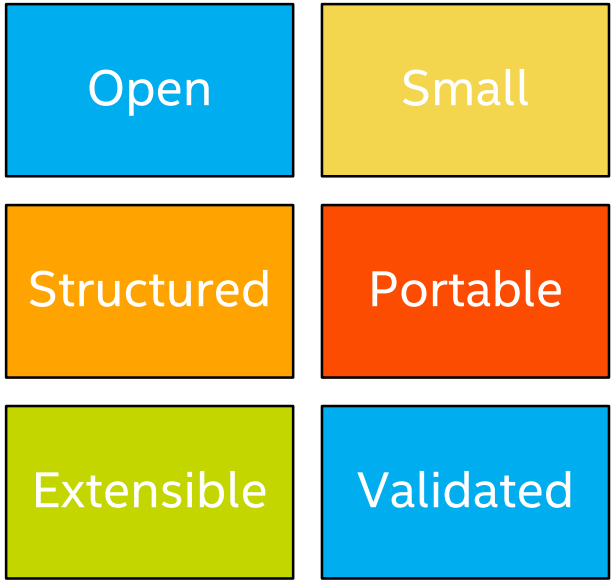


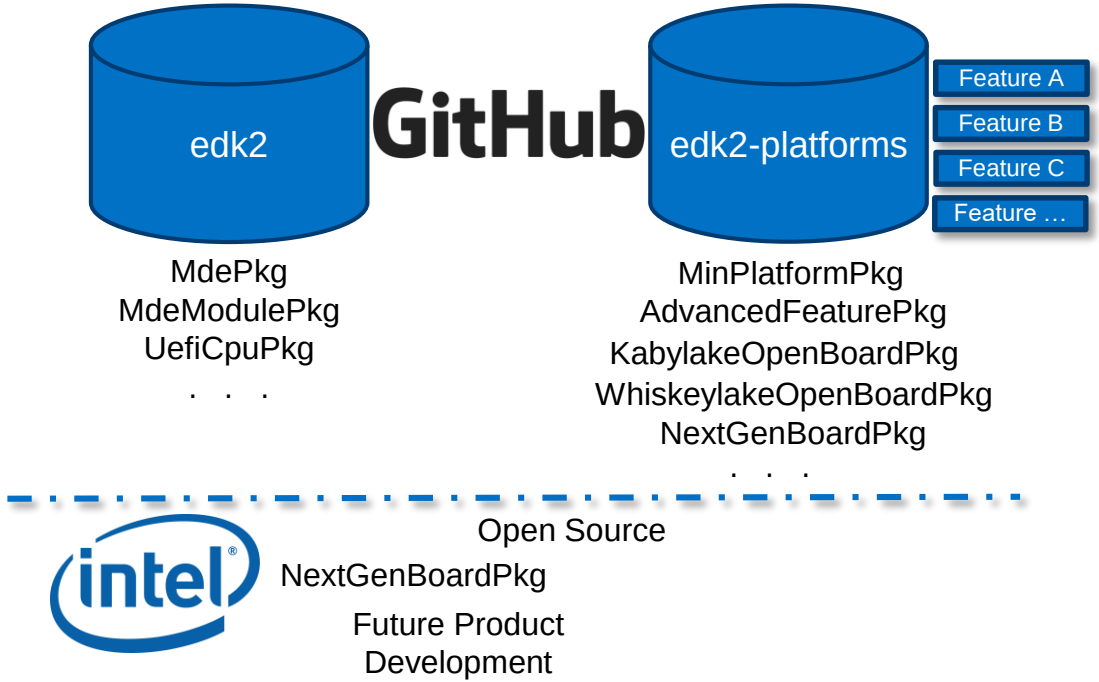
Evolving Intel Open Source Platform Firmware



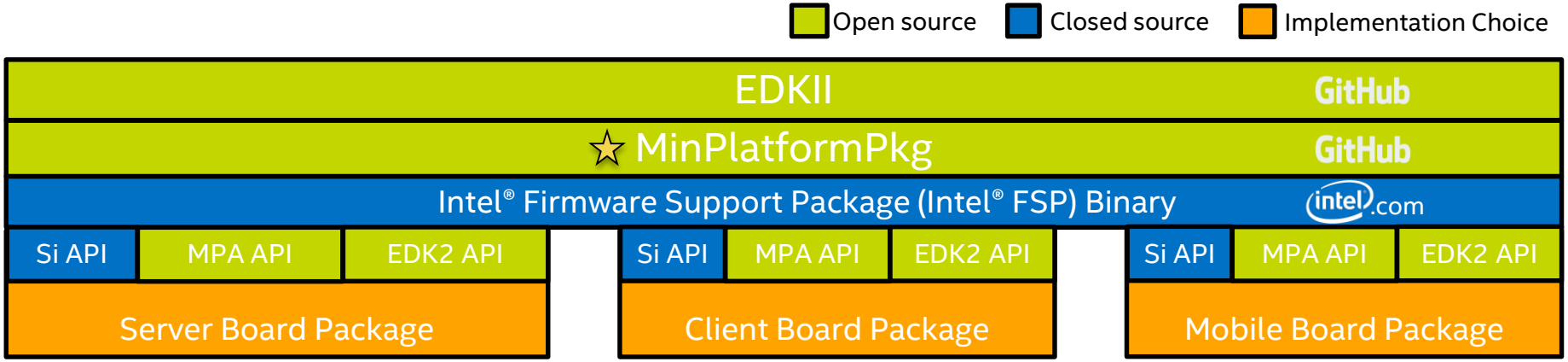


Evolving Intel Open Source Platform Firmware



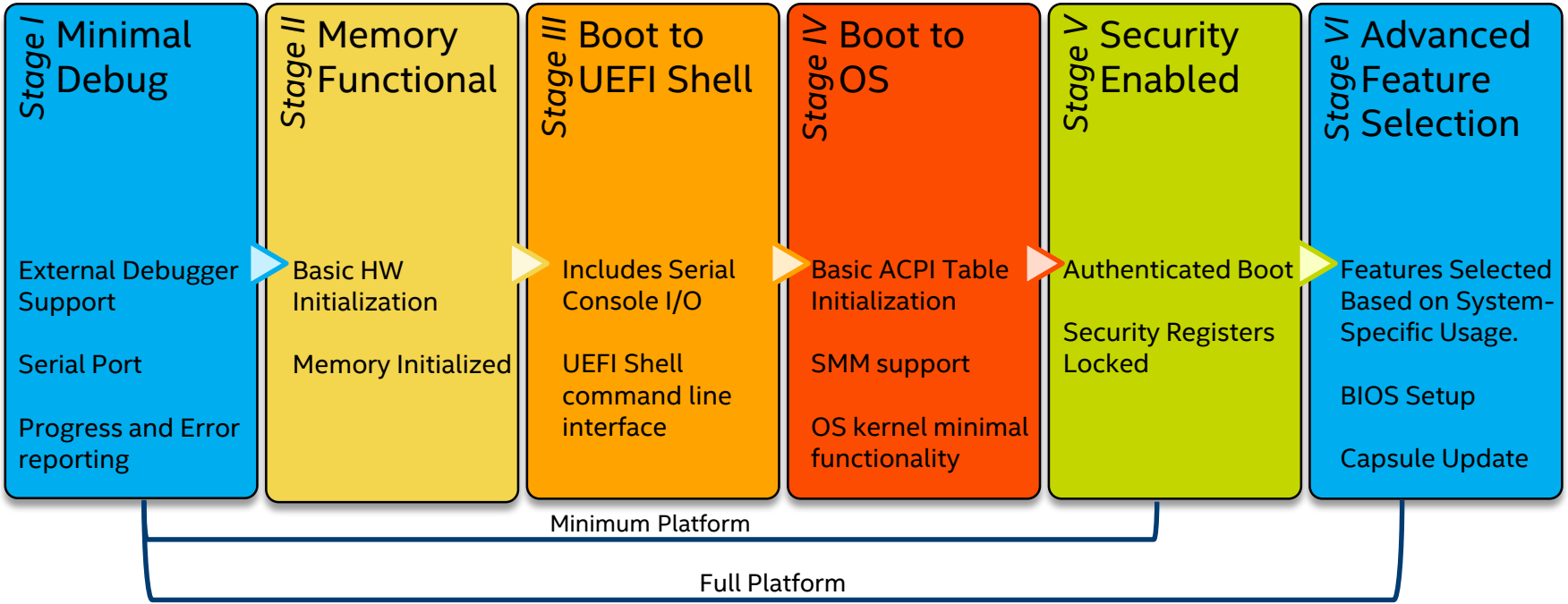


Intel Open Platform Firmware Stack

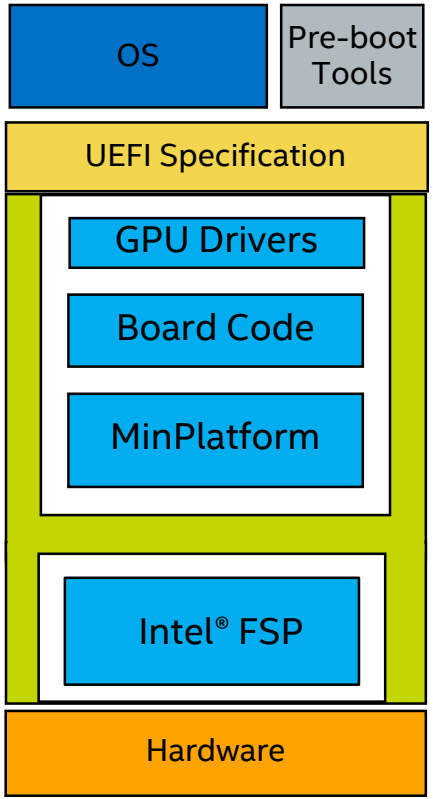


Consistent boot flows and interfaces
Approachable across the ecosystem
Scalable from pre-silicon to derivatives

The Staged Platform Approach



Intel Open Platform – Minimum Platform + Intel® FSP

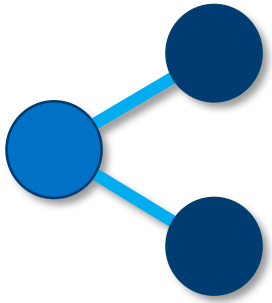


- UEFI is built with the PC supply chain in mind.
  - Open and closed modules can co-exist in a system.
  - Minimum Platform seeks to increase the overall share of open source UEFI firmware code available.
- UEFI's component based design gives OEM's choices:
  - Wide array of choice in ICs:
    - CPU
    - GPU
    - I/O Controllers (USB, Disk, etc.)
- Silicon vendors can provide pluggable UEFI components that adhere to specifications.

Call to Action

Contributions welcome!

Create and modify Intel system firmware



Share platform features

- Embedded system development
- Simple sample code for porting to other firmware

Create new board packages:

1. Start with a sample OpenBoard package.
2. Update the board-specific data such as GPIOs.
3. Get a simple stage 4 boot to OS boot functional and add advanced board features.
4. Customize the Intel FSP configuration settings for your needs.

Current Status & Upcoming Plans

Platforms Currently Supported:

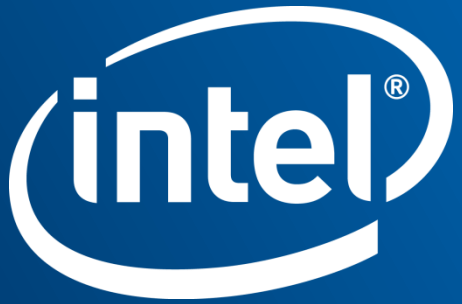
7th Generation Intel® Core™ i7 Processors (products formerly Kaby Lake)	Intel® Xeon® Scalable Processors with Intel® C620 Series Chipsets (products formerly Purley)	8th Generation Intel® Core™ i7 Processors (products formerly Whiskey Lake)
Kaby Lake U DDR3 RVP	Mt. Olympus	Whiskey Lake U DDR4 RVP
System76* Galago Pro 3		

Note: Sky Lake U DDR3 RVP works with KabylakeOpenBoardPkg.

EDK II Minimum Platform draft specification published.

Upcoming Plans:

1. Continue to roll out more Intel open source platform code.
2. Expand advance feature code and quality.
3. Support open source community continuous integration for minimum platforms.


intel®

Legal Disclaimer

You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Intel products described herein. You agree to grant Intel a non-exclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Performance varies depending on system configuration. No computer system can be absolutely secure. Check with your system manufacturer or retailer or learn more at intel.com.

Intel technologies may require enabled hardware, specific software, or services activation. Check with your system manufacturer or retailer.

The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest Intel product specifications and roadmaps

Copies of documents which have an order number and are referenced in this document may be obtained by calling 1-800-548-4725 or visit www.intel.com/design/literature.htm.

Intel, the Intel logo, and Core are trademarks of Intel Corporation in the U.S. and/or other countries.

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

© 2019 Intel Corporation. All rights reserved.

Glossary

- **edk2**: The tianocore.org git source code repository.
- **edk2-platforms**: The tianocore.org platforms git source code repository.
- **EDK II**: Cross-platform firmware development environment for the UEFI and PI specifications.
- **Intel® FSP**: Intel® Firmware Support Package. A binary distribution model for Intel silicon component code.
- **Green H**: Allowed components to build portable UEFI & PI arch components. APIs, standards, and libraries.
- **GPIO:** General Purpose Input/Output
- **Minimum Platform**: EDK II compatible minimum and cross-platform code with a well-defined control flow.
- **MPA:** Minimum Platform Architecture.
- **SPD:** Serial Presence Detect
- **Tiano**: A UEFI implementation combining TianoCore with Intel's closed source platform and silicon modules.
- **TianoCore:** An open source project that creates the most widely used UEFI implementation: **EDK II**.
- **UEFI:** Unified Extensible Firmware Interface.
- **UEFI.org / UEFI Forum**: Industry standards body.
- **UEFI PI Arch specs**: Platform Initialization. Firmware construction specifications. Defines: SEC, PEI, DXE, BDS.
- **UEFI Specification**: A firmware to OS interface specification.