Platform Architecture

Base System Architecture (BSA)
  - Defines hardware requirements

Base Boot Requirements (BBR)
  - Defines firmware requirements

These specifications require a minimum set of hardware and firmware implementations that will ensure OS and firmware will interoperate

EBBR is the BBR for the embedded systems
To define the firmware and bootloader requirements for Arm embedded devices which require a standardized interface between the platform (firmware) and the OS (system software).

Differs in scope from Server Base Boot Requirements (SBBR)

- SBBR is targeted at general purpose compute
  - strictly requires UEFI and ACPI
  - Driven by requirements from enterprise OS vendors

- EBBR targets embedded when firmware/OS separation is required
  - For example, class-A embedded devices like networking platforms
  - Friendly to U-Boot and Devicetree implementations
  - Intended to support embedded OS vendors and community distributions
Rationale and Scope

This specification is intended to be both practical and aspirational at the same time

• Practical in that it seeks to solve the problems OS vendors face today with regard to controlling the boot path of the platform.
• Practical in that it codifies what is feasible now or in the near future (ex. UEFI interfaces in U-Boot)
• Aspirational in that it specifies important features which may still require development to achieve. (ex. U-Boot support for UEFI Runtime services)
• Aspirational in that it seeks to move the bar on bare minimum expectations on how Arm platforms should behave.
Overview

This specification defines a set of base firmware requirements for embedded devices
- Compliant platforms must confirm to all the mandatory interfaces specified in EBBR
- Firmware must implement a subset of the UEFI specification
  - Must implement boot services
  - Must implement runtime services for changing boot variables
- May supply either Devicetree or ACPI system description data, but must conform to the specifications for the selected mechanism.

This specification is intended to be OS-neutral. It leverages the prevalent industry standard firmware specifications of UEFI and can be implemented using U-Boot and Devicetree
Details - UEFI

- Compliant with v2.7 of UEFI spec or later
- UEFI running at EL2 if hypervisors are supported, EL1 otherwise
- If system boots from local block storage, storage must be GPT formatted and use a FAT formatted system partition
- Support running UEFI 64-bit PE/COFF loaded images containing only A64 code
- Optionally may implement Secure Boot
  - but if implemented it must confirm with UEFI specification for Secure Boot
Details – UEFI Runtime Services

- UEFI Runtime services must be implemented
  - must support running at either EL1 or EL2 (if implemented)
- UEFI Memory Map minimum page size is 4K
- To support 64K pages in the OS, all 4k pages within the same 64k block must use identical page attributes.
- SetTime() / GetTime() must be implemented for interacting with real time clock
Details – UEFI Runtime Services – System Reset

- EfiResetCold()/EfiResetShutdown() must be implemented, and shutdown must not reset the system.
- EfiWarmReset() must be implemented if used to support capsule updates.
- System software shall use EFI runtime services for all system resets.
  - Must not call PSCI directly (UEFI may have work to do).
Details – UEFI Runtime Services – Variables

- Non-Volatile variables must be implemented and may not be emulated with RAM.
- Non-Volatile storage by runtime services must not require OS intervention (i.e., to store to eMMC).
Questions

• Do we have the correct scope?
• Are the requirements too tight, or not tight enough?
• Are runtime services something useful for this market segment?
  ○ Runtime services are used to manipulate variables after booting an OS
  ○ The alternative is to do all variable manipulation from a UEFI application before ExitBootServices() is called.
EBBR

Specification Development:
Specification Location:
Mailing list: arm.ebbr-discuss@arm.com.
Future venues: ELCE, UEFI Plugfest
Thank You

#SFO17
SFO17 keynotes and videos on: connect.linaro.org
For further information: www.linaro.org