Flashless/Streaming Boot using OCP, PCIe, and DMTF Standards

Author: Varun Sampath
Revision: 0.2, February 27, 2024

The goal of Flashless/Streaming Boot is to enable devices to boot in hyperscale environments without requiring the presence of firmware binary in SPI flash. SPI flashes are the bedrock for firmware today due to their low cost, simple interface, and fixed bandwidth and latency. However, the management of flashes cause several hazards that are magnified in hyperscale deployments:

1. Security: the non-volatile nature of SPI flash exposes hazards for
    - boot kits (Secure Boot and Secure Firmware Update bugs)
    - service downtime (Secure Recovery bugs)
    - lack of support of security features within the SPI by many vendors
2. Operations: flash supply chain is complex, and field replaceability is usually not possible
3. Manageability: more firmware with more firmware updates lead to problems.
    - Program/erase cycles take orders of magnitude more time than reads, making background firmware update complex.
    - Flashes wear out, exposing denial of service risks.

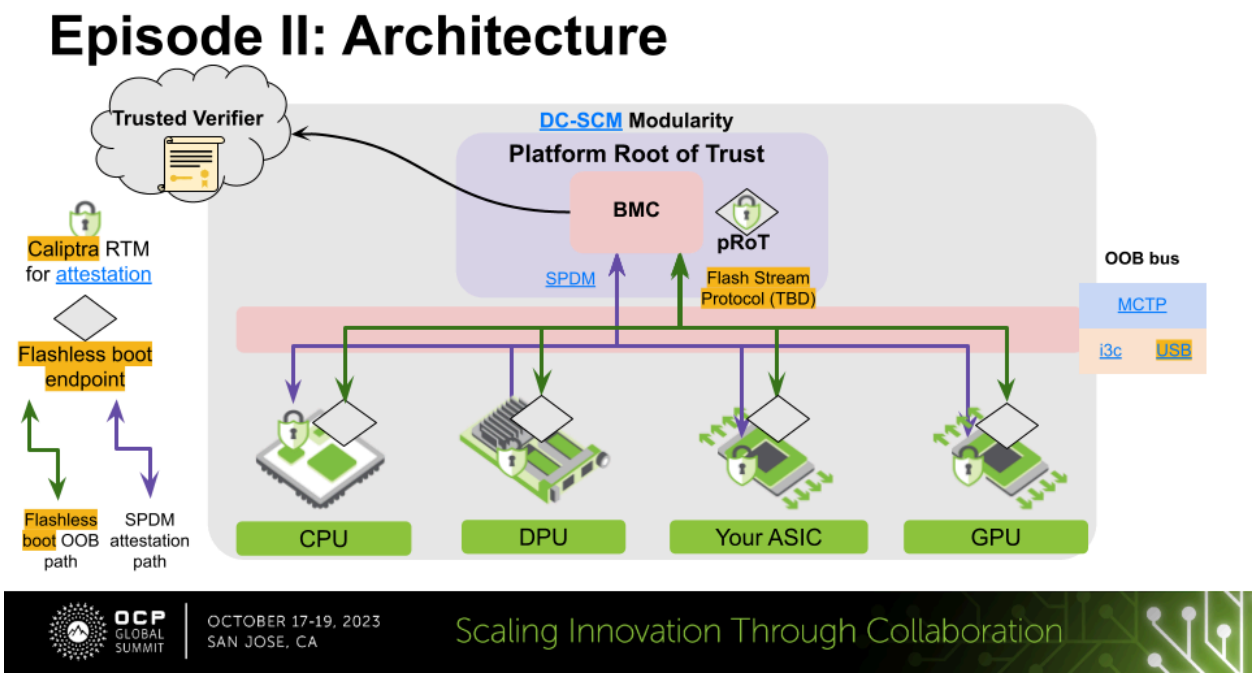
This paper outlines a method for Flashless/Streaming boot of datacenter devices such as CPUs and PCIe devices. This method uses established standards:

1. OCP Recovery
2. PLDM for Firmware Update (DSP0267)
3. MCTP over I3C, USB2.0, and PCIe VDM

The primary advantage of this approach is to avoid a chicken-and-egg problem with pre-existing systems that support flash-based boot and firmware update. Device vendors can leverage their pre-existing implementations to support both flash and Flashless/Streaming systems. A given device can support installation in both systems. This benefits the circular economy of the device.

System Topology

This paper presumes that a system follows the OCP Composable Security Architecture, where a Platform Root of Trust (PA-RoT) is responsible for provisioning the firmware of CPUs and PCIe devices in the system.



CPUs and PCIe devices possess an Active Component Root of Trust (AC-RoT) which implements the functions of Root of Trust for Recovery (RTRec) and Root of Trust for Update (RTU).

The OCP GPU FW Update Specification proposes a system topology where an Accelerator Management Controller (AMC) manages firmware updates. This paper supports that topology where the AMC has the role of PA-RoT in this context.

The platform connects the PA-RoT and AC-RoTs with out-of-band interfaces such as SMBus, I3C, USB2.0, and PCIe VDM.

Boot Sequence

In this paper, the PA-RoT is used to boot the AC-RoTs. The existing AC-RoT RTRec and RTU functionality are used to load and execute the device firmware. The RTRec is used to load the early firmware (“early-fw”), which consists of the device’s first mutable code and potentially other firmware components. The early-fw implements the RTU. The RTU is used to load the remainder firmware (“remainder-fw”). The advantage of the two-stage approach is to minimize device complexity. The first stage can be implemented with a simpler SMBus interface that can be initialized and managed by hardware, while the second stage can be implemented with a firmware-initialized and managed interface such as I3C, USB2.0, and PCIe VDM.

PCIe Boot Sequencing

PCIe resets are a challenge for Flashless/Streaming boot. One problem is PCIe boot performance. The Flashless/Streaming interface for early-fw (SMBus or I3C) is slower than SPI. Early-fw is likely needed to boot the PCIe controller to train at the highest link speeds and establish device readiness. That firmware needs to be pushed, authenticated, measured, and executed.

To elaborate on the boot timing requirements, the PCIe Base Specification (revision 6.0) states the following:

> *A component that supports Link speeds greater than 5.0 GT/s must enter the LTSSM Detect state within 100 ms of the end of Fundamental Reset (Link Training is described in Section 4.2.5). A component that supports only Link speeds 5.0 GT/s or less must do this within 20 ms. All components are strongly encouraged to minimize this time period.*
>
> *Following exit from a Conventional Reset of a device, within 1.0 s the device must be able to receive a Configuration Request and return a Successful Completion if the Request is valid. This period is independent of how quickly Link training completes. If Readiness Notifications mechanisms are used (see § Section 6.22 .), this period may be shorter, or, with appropriate system support, longer.*

The PCIe Card Electromechanical (CEM) Specification defines the activation of the PERST# signal (rising edge) as the assertion of the Fundamental Reset. CEM (revision 5.0) states the following:

> *The PERST# signal is used to indicate when the power supply is within its specified voltage tolerance and is stable. It also initializes a component's state machines and other logic once power supplies stabilize. On power-up, the de-assertion of PERST# is delayed 100 ms (T_{PVPERL}) from the power rails achieving specified operating limits. Also, within this time, the reference clocks (REFCLK+, REFCLK-) also become stable, at least TPERST-CLK before PERST# is de-asserted. PERST# is asserted in advance of the power being switched off in a power-managed state like S3. PERST# is asserted when the power supply is powered down, but without the advanced warning of the transition.*

To support PCIe in-band recovery, devices may implement a watchdog timer that boots the PCIe interface into a failsafe mode if the firmware dependencies are not available in time. Such an implementation poses problems for flashless boot. PA-RoT streaming time cannot be bounded by device hardware watchdogs.

A second problem is that the platform may be unaware of PCIe resets. PERST# is nominally controlled by the Root Complex, but many different firmware images may execute as part of Host boot and toggle PERST# without PA-RoT coordination. PCIe Conventional Resets and Function Level Resets are issued in-band without platform coordination. In a cloud environment, these in-band resets could be issued by tenants. Tenant operations that lead to increased PA-RoT load poses a denial-of-service threat.

PCIe Device Behavior

This paper outlines a Power-On Reset architecture for PCIe devices to meet PCIe boot timing requirements while supporting Flashless/Streaming boot. This approach also requires platform boot sequencing modifications. The device approach is to:

1. Support OCP Recovery-based loading of early-fw.
2. Support entering Flashless Boot mode via OCP Recovery command. This device mode will suppress any device watchdogs for PCIe link up..
3. Support firmware loading independent of PERST# state.
4. Ensure firmware does not require reload on any platform or PCIe resets except Power-On Reset.
5. Support Device Readiness Status (DRS).
6. Include as part of early-fw:
    - All firmware dependencies for training at the maximum capable Link Speed and DRS.
    - RTU firmware for remainder-fw load support.

Power-On Reset may be generated on power-good of main rails; it is not required to derive from auxiliary power (either 3v3 Vaux defined in PCIe CEM Specification or AUX_PWR_EN defined in the OCP NIC Specification).

Platform Behavior for Flashless/Streaming Boot of PCIe Devices

The platform needs to suppress PCIe link training until devices are initialized. PCI-SIG System Firmware Intermediary (SFI) and PCIe Management Interface (PCIe-MI) can achieve this.

Boot Sequence Summary

1. Platform enables power to all devices.
2. PA-RoT enumerates devices via SMBus or I3C.
3. PA-RoT reboots devices into Flashless/Streaming Boot Mode if they don't default to it.
4. PA-RoT boots early-fw of Flashless/Streaming devices via OCP Recovery.
5. PA-RoT programs Root Ports and Switches via PCIe-MI and SFI to enable DFP links for training.
6. Platform trains PCIe Links and enumerates PCIe Functions.
7. Platform enumerates MCTP endpoints via PCIe VDM, I3C, or USB2.0. MCTP-over-I3C or MCTP-over-USB2.0 may be enumerated after step 4.
8. Platform uses PLDM for Firmware Update to enable boot of the remainder-fw.
9. PA-RoT requests and verifies attestation reports from the devices.

Runtime Behavior

The Flashless/Streaming boot is complete once the remainder-fw is booted. The process does not restart until the platform reboots (PERST# activation).

It is possible to use the Flashless/Streaming boot interface of PLDM for Firmware Update to initiate impactless updates while the device is operational.

A device can attest to its Flashless/Streaming boot state via SPDM.

OCP Recovery Protocol

The OCP Recovery Specification describes the SMBus interface for loading the early-fw. This paper's approach assumes devices implement the following OCP Recovery capabilities:

1. Identification: this enables the PA-RoT to find the correct early-fw.
2. Flashless Boot: this enables the PA-RoT to put the device into a Flashless/Streaming boot environment without needing to erase the flash.
3. Management Reset or Device Reset: this enables the PA-RoT to restart the OCP Recovery process if there are errors.
4. Device Status: this enables the PA-RoT to monitor the device.
5. Recovery memory access and Push C-image support: this enables the PA-RoT to write the early-fw into device memory.

This paper suggests that the following amendments to the OCP Recovery Specification be amended to implement:

1. Multi-stage activation.
2. Flashless/Streaming Boot enable.
3. I3C support.

Multi-Stage Activation

Multi-stage activation enables the early-fw to consist of multiple components. Multi-stage activation can be supported by adding a recovery image index to the recovery status as shown in the highlights below.

RECOVERY_STATUS cmd=39	ro	2	Recovery status: Recovery Debug status of device Byte 0: **Device recovery status** Bit [3:0]: 0x0: Not in recovery mode 0x1: Awaiting recovery image 0x2: Booting recovery image 0x3: Recovery successful 0xc: Recovery failed 0xd: Recovery image authentication error 0xe: Error entering Recovery mode (might be administratively disabled) 0xf: Invalid component address space Bit [7:4]: Recovery Image Index Byte 1: **Vendor specific status** Bit [7:0]: Vendor Defined

Flashless/Streaming Boot Enable

This paper suggests amending the DEVICE_RESET command, byte 1 to add an enum 0xE “Enter flashless mode on next platform reset”. Flashless Mode differs from Recovery Mode by disabling PCIe watchdogs that would cause the device to autonomously boot its upstream facing port without firmware.

RESET cmd=37	rw	3	Reset control - For devices which support reset, this register will reset the device or management entity. Byte 0: **Device Reset Control (Write 1 Clear e.g., after action device starts 0x0)** 0x0: No reset 0x1: Reset Device (PCIe PRESET or equivalent. This is likely bus disruptive) 0x2: Reset Management. This reset will reset the management subsystem. If supported, this reset MUST not be bus disruption (cause re-enumeration) 0x3-FF: Reserved Byte 1: **Forced Recovery** 0x0 - No forced recovery 01-D - Reserved 0xE - Enter flashless mode on next platform reset 0xF - Enter recovery mode on next platform reset 0x10-FF: Reserved Byte 2: **Interface Control** 0x0: Disable Interface mastering 0x1: Enable Interface mastering

I3C Support

An OCP Recovery over I3C specification would provide the following advantages:

1. Improve early-fw load performance.
2. Enable OCP Recovery commands while the bus is in I3C mode. PCIe 6.2 Sideband Specification disallows I3C Mixed Fast Bus mode. This implies that the OCP Recovery SMBus interface must be disabled when the bus is enumerated in I3C mode.
3. Simplify devices that don’t need SMBus compatibility.

Here is a strawman for OCP Recovery over I3C updates:

- I3C Basic 1.1.1
- SDR operation only for first generation of spec
- Target operation only
- Dynamic addressing only, ENTDAA
- Must have a unique address compared to MCTP-I3C endpoint.
    - Can be implemented with two I3C Devices on the same bus.
    - Can be implemented as a Virtual Target of the same I3C Device.
- Use I3C Read Transfer and I3C Write Transfer.
- Max write and read data length of 256 bytes including command code and PEC, advertised via GETMRL and GETMWL.

    - Max Recovery data bytes is 252:
        - 1 byte command code
        - 2 bytes length
        - 1 byte PEC
- Need a DCR value assigned from MIPI
- Recommend same CCC set as recommended by MCTP-I3C Table 4
- Same command set and codes as OCP Recovery SMBus.
- Write Data 1 is command code.
- Write Data 2-3 is the byte length of the transfer.
- Read Data 1-2 is the byte length of the transfer following a repeated start.
- Polling CMS acknowledgement can be optionally implemented as IBI

PLDM for Firmware Update Protocol

The PLDM for Firmware Update Specification (DSP0267) defines messages and data structures for obtaining firmware code and data. The approach of this paper is to leverage this protocol for Flashless/Streaming boot of the remainder-fw. The PA-RoT implements the Update Agent (UA) and the AC-RoT early-fw implements the Firmware Device (FD).

There are multiple benefits to this approach:

1. Leverage: DSP0267 over MCTP is an established protocol for firmware update. This avoids a chicken-and-egg problem for PA-RoTs and AC-RoTs to implement a new protocol. It also leverages the existing out-of-band MCTP fabrics that will have low utilization during the device boot process.
2. Modularity: DSP0267 supports multiple firmware components within a firmware package. An update may only be to a subset of components that a FD supports. This property is valuable for implementing impactless updates. The packaging model is also already mapped to Redfish for inventory purposes per the OCP GPU FW Update Specification.
3. “Pull” model: “RequestFirmwareData” command is initiated by the FD instead of the UA. This lets the FD perform flow control which is beneficial for staging firmware bytes. It also enables the FD to request the same bytes if it detects errors.

Flashless/Streaming boot in turn can simplify the FD behavior because writing firmware bytes to local RAM provides greater performance determinism than performing program/erase cycles of an SPI flash.

The following sections detail changes in DSP0267 flow to facilitate Flashless/Streaming boot.

PLDM Flashless/Streaming Boot Enablement

Flashless/Streaming boot enablement is a function of the package contents, the UA state, and the FD state.

Ideally, all packages are capable of being used for Flashless/Streaming or flash updates. To advertise this capability, this paper suggests amending DSP0267 Firmware Device ID Record “DeviceUpdateOptionFlags” bit 1 from “Reserved” to “Support Flashless/Streaming boot”. If set, the UA may attempt to Flashless/Streaming boot the package.

The UA needs to tell the FD to Flashless/Streaming boot a component instead of writing it to SPI flash. This paper suggests amending DSP0267 “UpdateComponent” command “UpdateOptionFlags” bit 3 from “Reserved” to “Flashless/Streaming Boot”. When set, the UA requests that the FD load the component to local memory instead of to firmware storage. Additionally, this paper suggests amending DSP0267 “UpdateOptionFlagsEnabled” bit 3 from “Reserved” to “Flashless/Streaming Boot Enabled”. When set, the FD will load the component to local memory.

PLDM Flashless/Streaming Boot Activation

A component transferred with Flashless/Streaming boot enabled must be available for self-contained or automatic activation. This avoids additional device resets that would complicate the boot flow compared to a flash-based boot. If a flash-based firmware update requires a reset or power cycle, this paper suggests that the FD use the “ApplyComplete” command field “ComponentActivationMethodsModification” to indicate that a self-contained or automatic activation is now possible.

While DSP0267 allows multiple components to be updated, fewer components will reduce the load on the UA for facilitating the boot of the device. The UA may set a limit on how many components can be updated during the boot process.

The OCP GPU FW Update Specification specifies pre-checks and post-checks for firmware update activation. These can collapse to SPDM-based attestation of the device since it is executing the firmware that was transferred. There’s no need to handle pending states.

Memory Requirements

The device cannot depend on PA-RoT availability for firmware fetch during runtime operation of the platform. The reason is that unlike a SPI flash, a PA-RoT cannot provide deterministic latency or availability for firmware fetch requests. In multi-tenant environments, it is critical that tenant actions (such as PCIe in-band resets) do not cause increased load on the PA-RoT that can lead to denial of service of the platform.

Performance

The performance of Flashless/Streaming boot compared to flash-based boot depends on the following factors:

1. Size of early-fw.
2. SMBus or I3C clock speed.
3. Size of remainder-fw.
4. Bandwidth of MCTP fabric used for transferring remainder-fw.
5. DSP0267 command processing latency for PA-RoT and AC-RoT.

It is important to minimize the size of early-fw. SMBus is 400kbps or slower in datacenter platforms with long trace lengths. This is orders of magnitude slower than other interfaces. A 200KB early-fw will take approximately 4 seconds to transfer.

Depending on the MCTP fabric, the Flashless/Streaming boot interface may be faster than SPI. PCIe VDM (multi-gigabit) and USB2 (480Mbps) are much faster than most SPI implementations (20-100Mbps). I3C (2-25Mbps) is likely slower. But note that SPI protocol has overhead of 4-byte response per read while I3C protocol allows for 64B payload per request.

The overhead of PLDM for firmware update and MCTP, assuming a RequestFirmwareData size of 4KB and a packet payload size of 64 bytes, is approximately 1.15x.

This paper's approach ensures that Flashless/Streaming boot operations only occur at platform boot. This reduces the load on PA-RoT and AC-RoT. However, given that two firmware entities are needed to communicate to do transfer, it is unlikely that MCTP fabric line rates will be achieved and Flashless/Streaming boot will likely be slower than flash-based boot. This is compounded by reduction in boot concurrency compared to local flashes; there is only one PA-RoT to aid booting multiple AC-RoTs. However, if device boot can be performed in parallel to platform boot (e.g., DRAM training) it is possible that the Flashless/Streaming boot overhead is amortized.

Summary

This paper describes an approach to Flashless/Streaming boot that leverages existing OCP, PCIe, and DMTF standards with some modifications. This approach enables a phased migration for hyperscalers and device vendors. Flashless/Streaming boot aims to provide security benefits and TCO reductions to hyperscalers by avoiding boot and update dependencies on local non-volatile SPI flashes attached to every device in the datacenter.

Future areas to consider are PCIe state changes like hot-plug or exit from device $D3_{cold}$. Both of these events require the Host to coordinate with the PA-RoT to enable reboot of the device. PA-RoTs may need to scale in complexity or delegate their logic with helper controllers to meet runtime needs.

Prior Presentations

- OCP Security Call, January 10 2023
- OCP Flashless Boot Specification April 2023
- Composable Security Architectures: Episode II October 2023
- OCP Flashless Boot Update April 2024
- OCP Streaming Boot Implementation Update October 2024

Revision History

- February 7, 2024: 0.1 first draft
- February 27, 2024: 0.2
    - OCP Recovery changes for Flashless Boot mode and I3C
    - PCIe-MI and SFI for Platform boot sequence.