**# Title:**

Clarify EFI\_LOAD\_OPTION.FilePathList[] device path definition

**# Status:**

Draft

**# Document:**

UEFI Specification 2.9 (Future Errata)

**# License:**

SPDX-License-Identifier: CC-BY-4.0

**# Submitter:**

* Samer El-Haj-Mahmoud, ARM
* TianoCore Community (<https://www.tianocore.org>)

**# Summary of the change**

Clarify EFI\_LOAD\_OPTION.FilePathList[] device path definition

**# Benefits of the change**

The UEFI spec knows two types of separators for device paths. Both have type 0x7F (End of Hardware Device Path) but differ by the sub-type:

Sub-Type 0xff – End Entire Device Path

Sub-Type 0x01 – End Instance of a Device Path

Field EFI\_LOAD\_OPTION.FilePathList[] is described in the UEFI spec as follows:

"A packed array of UEFI device paths. The first element of the array is a device path that describes the device and location of the Image for this load option."

It is not immediately clear if the separators between the array elements are of sub-type 0xff or 0x01. The description in the UEFI spec should be reworked for more clarity.

**# Impact of the change**

Change is compatible with existing TianoCore EDK2 and UBoot implementations. No FW or OS code changes are expected.

**# Detailed description of the change [normative updates]**

* Insertions in **green**
* Removals in ~~red~~

**3.1.3 Load Options**

…

*FilePathList* A packed array of UEFI device paths. The first element of the array is a device path that describes the device and location of the Image for this load option. The *FilePathList[0]* is specific to the device type. Other device paths may optionally exist in the *FilePathList*, but their usage is OSV specific. Each element in the array is variable length, and ends at the device path end structure. Because the size of *Description* is arbitrary, this data structure is not guaranteed to be aligned on a natural boundary. This data structure may have to be copied to an aligned natural boundary before it is used.

*OptionalData* The remaining bytes in the load option descriptor are a binary data buffer that is passed to the loaded image. If the field is zero bytes long, a **NULL** pointer is passed to the loaded image. The number of bytes in *OptionalData* can be computed by subtracting the starting offset of *OptionalData* from total size in bytes of the **EFI\_LOAD\_OPTION**.

***Note***: Each device path in the FilePathList can be a single instance or a multi-instance device path.

**Related Definitions**

**…**