**# Title:**

Clarify NVMe device path EUI-64 byte order

**# Status:**

Draft

**# Document:**

UEFI Specification 2.9 (Future Errata)

**# License:**

SPDX-License-Identifier: CC-BY-4.0

**# Submitter:**

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

**# Summary of the change**

Clarify NVMe device path EUI-64 byte order. See details below

**# Benefits of the change**

The UEFI Spec is ambiguous when it comes to the byte-order of EUI-64 fields for NVMe (and Infiniband) device path nodes

See: <https://bugzilla.tianocore.org/show_bug.cgi?id=3291>

and <https://bugzilla.tianocore.org/show_bug.cgi?id=3292>

The UEFI specification for the NVMe() node refers to NVM Express Specification.

**Section 10.3.4.22***“Refer to the latest NVM Express specification for descriptions of Namespace Identifier (NSID) and IEEE Extended Unique Identifier (EUI-64):See “Links to UEFI-Related Documents” (http://uefi.org/uefi under the headings “NVM Express Specification”.”*

**Table 10-61 says:**

*“  
NVMe(NSID,EUI)  
…*

*The EUI is the IEEE Extended Unique Identifier (EUI-64) that is displayed in hexadecimal format represented as a set of octets separated by dashes (hexadecimal notation), e.g., FF-FF-FF-FF-FF-FF-FF-FF.”*

In the "NVM Express Base Specification", NVM Express Revision 1.4b ,September 21, 2020, chapter 7.10.4 IEEE Extended Unique Identifier (EUI64) we find:

"EUI64 is defined in big endian format."

Looking at real examples from EDK2 and Linux dumps of NVMe device paths:

EDK II Device Path

PcieRoot(0x0)/Pci(0x0,0x0)/NVMe(0x1**,E7-66-03-81-B5-38-25-00**)

Variable NV+RT+BS 'EFIGlobalVariable:Boot000B' DataSize = 0x9E

00000000: 01 00 00 00 26 00 55 00-45 00 46 00 49 00 20 00 \*....&.U.E.F.I. .\*

00000010: 53 00 41 00 4D 00 53 00-55 00 4E 00 47 00 20 00 \*S.A.M.S.U.N.G. .\*

00000020: 4D 00 5A 00 56 00 4C 00-57 00 32 00 35 00 36 00 \*M.Z.V.L.W.2.5.6.\*

00000030: 48 00 45 00 48 00 50 00-2D 00 30 00 30 00 30 00 \*H.E.H.P.-.0.0.0.\*

00000040: 4C 00 37 00 20 00 53 00-33 00 35 00 45 00 4E 00 \*L.7..S.3.5.E.N.\*

00000050: 41 00 32 00 4B 00 35 00-33 00 30 00 31 00 38 00 \*A.2.K.5.3.0.1.8.\*

00000060: 35 00 20 00 31 00 00 00-02 01 0C 00 D0 41 08 0A \*5..1........A..\*

00000070: 00 00 00 00 01 01 06 00-00 00 03 17 10 00 01 00 \*................\*

00000080: 00 00 **00 25 38 B5 81 03-66 E7** 7F FF 04 00 4E AC \*...%8...f.....N.\*

00000090: 08 81 11 9F 59 4D 85 0E-E2 1A 52 2C 59 B2 \*....YM....R,Y.\*

Linux:

sudo smartctl -a /dev/nvme0n1

Namespace 1 IEEE EUI-64: **002538 b5810366e7**

sudo nvme id-ns /dev/nvme0n1

eui64 : **002538b5810366e7**

Also (for another disk):

# efibootmgr -C -b 0006 -d /dev/nvme0n1p1 -p 1 -L test -l '/EFI/fedora/shimx64.efi' -e 3 -v | grep 0006

Boot0006\* test PciRoot(0x2)/Pci(0x1,0x4)/Pci(0x0,0x0)/NVMe(0x1,**00-25-38-55-91-50-87-05**)/HD(1,GPT,927f5841-dc54-4ff4-821c-d29b59c48ab1,0x800,0x1ff800)/File(\EFI\fedora\shimx64.efi)

And the relevant binary device path looks like:

00000030 01 00 00 00 **00 25 38 55 91 50 87 05** 04 01 2a 00 |.....%8U.P....\*.|

It seems that Linux (smartctl, nvme, and efibootmgr) is printing the string representation of the NVMe EUI64 identifier in natural order, which is assumed to be big endian per the NVMe spec.

However, EDK2 device path string representation is in reversed order.

**# Impact of the change**

Ideally: Goal is to clarify the UEFI spec without changing implementations (in FW or the OS)

However, this is not possible, since EDK2 and UBoot UEFI implementations are not consistent. Also Linux efibootmgr and EDK2 are not consistent

**Proposal:**

* Clarify the UEFI spec to match the format used by existing TianoCore EDK2 implementation
* Change UBoot to match the new UEFI spec language and EDK2 implementation
* Change efibootmgr to match the new UEFI spec language and EDK2 implementation

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

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

10.3.4.22 NVM Express namespace messaging device path node

…  
Refer to the latest NVM Express specification for descriptions of Namespace Identifier (NSID) and IEEE Extended Unique Identifier (EUI-64):See “Links to UEFI-Related Documents” (http://uefi.org/uefi under the headings “NVM Express Specification”.

*"When an application client displays or otherwise makes the EUI-64 identifiers visible to a user, the values should be displayed in hexadecimal format with byte 0 first (i.e., on the left) and byte 7 last (i.e., on the right) regardless of the internal representation of the EUI-64.”*