

Intel® Virtual RAID on CPU (Intel® VROC) UEFI Firmware Management

Specification

Revision 1.0

September 2022

Intel Confidential

Document Number: 743970



You may not use or facilitate the use of this document in connection with any infringement or other legal analysis. 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.

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

All product plans and roadmaps are subject to change without notice.

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 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 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.

© Intel Corporation. Intel, <Intel launch names, if mentioned in the slides below> the Intel logo, and other Intel marks are trademarks of Intel Corporation or its subsidiaries.

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

Copyright© 2022, Intel Corporation. All rights reserved.

Contents

1 Introduction 5
1.1 Intended Audience 5
1.2 Terminology 5
1.3 Reference Documents 6
2 Intel® VROC UEFI Driver Version Reporting 7
2.1 EFI Component Name Protocol 7
2.2 EFI Driver Binding Protocol 7
3 Drive Firmware Management in VROC UEFI Driver 8
3.1 NVMe* Drives Firmware Management Protocol 8
3.1.1 GetImageInfo 9
3.1.2 SetImage 10
3.2 Firmware Management Protocol for SATA Drives 12
3.2.1 GetImageInfo 12
3.2.2 SetImage 13

Figures

Figure 3-1. Firmware Management Class Diagram 8
Figure 3-2. NVMe* Drive SetImage Implementation 11
Figure 3-3. SATA Drive SetImage Implementation 15

Tables

Table 1-1.Terminology 5
Table 1-2. Reference Documents 6
Table 2-1. VROC UEFI Driver Version Encoding 7
Table 3-1. Intel® VROC EFI Firmware Management Protocol Functions 8
Table 3-2. NVMe* Drives EFI_FIRMWARE_IMAGE_DESCRIPTOR Implementation 9
Table 3-3. GetImageInfo Outputs Implementation in Intel® VROC for SATA Drives 12
Table 3-4. ATA DOWNLOAD MICROCODE Subcommand Decision Matrix 16

Revision History

Document Number	Revision Number	Description	Revision Date
743970	1.0	• Initial release	September 2022

§§

1 Introduction

This document outlines the specifications for Intel® Virtual RAID on CPU (Intel® VROC) UEFI driver to support the management protocols used to update the firmware on drives managed by Intel® VROC. It specifies:

- The protocols used to retrieve version of Intel® VROC UEFI driver – Chapter 2
- The protocol used to manage firmware on the drives handled by Intel® VROC – Chapter 3

1.1 Intended Audience

The intended audience for this document is OEMs/ODMs who are preparing drive firmware update tools for platforms that support Intel® VROC.

1.2 Terminology

Table 1-1.Terminology

Term	Description
API	Application Program Interface
FW	Firmware
NVMe*	Non-volatile Memory Express*
OEM	Original Equipment Manufacturer
PCIe	Peripheral Component Interconnect express
SATA	Serial Advanced Technology Attachment
Intel® VROC	Intel® Virtual RAID on CPU
UEFI	Unified Extensible Firmware Interface

1.3 Reference Documents

Table 1-2. Reference Documents

Document	Document No./Location
UEFI Specification 2.9	https://uefi.org/sites/default/files/resources/UEFI_Spec_2_9_2021_03_18.pdf
NVMe Express Base Specification Revision 1.4	https://nvmexpress.org/wp-content/uploads/NVM-Express-1_4-2019.06.10-Ratified.pdf
ATA Command Set – 5 (ACS-5)	https://www.t13.org/system/files/Project%20Drafts/2021/di558r8b-ATA%20Command%20Set%20-5%20%28ACS-5%29.pdf

§§

2 Intel® VROC UEFI Driver Version Reporting

Intel® VROC UEFI driver version can be read by using following protocols:

- EFI Component Name Protocol
- EFI Driver Binding Protocol
- Both ways are described in the subsections.

2.1 EFI Component Name Protocol

The Intel VROC UEFI driver implements EFI_COMPONENT_NAME2_PROTOCOL compliant with UEFI Specification version 2.9. GetDriverName function returns DriverName string which contains the name and version of VROC UEFI driver.

For example, version 7.6.0.1004 is reported as "Intel(R) VROC with VMD Technology 7.6.0.1004" string.

2.2 EFI Driver Binding Protocol

The Intel VROC UEFI driver implements EFI_DRIVER_BINDING_PROTOCOL described in UEFI Specification version 2.9 with EFI_DRIVER_BINDING_PROTOCOL.Version parameter represents the version of VROC UEFI driver, encoded according to the below table.

Table 2-1. VROC UEFI Driver Version Encoding

Bits	Value (VROC UEFI driver version)
31:24	Major version
23:20	Minor version
19:16	Maintenance version
15:0	Build version

For example, version 7.6.0.1004 is reported as 0x076003EC value.

§§

3 Drive Firmware Management in VROC UEFI Driver

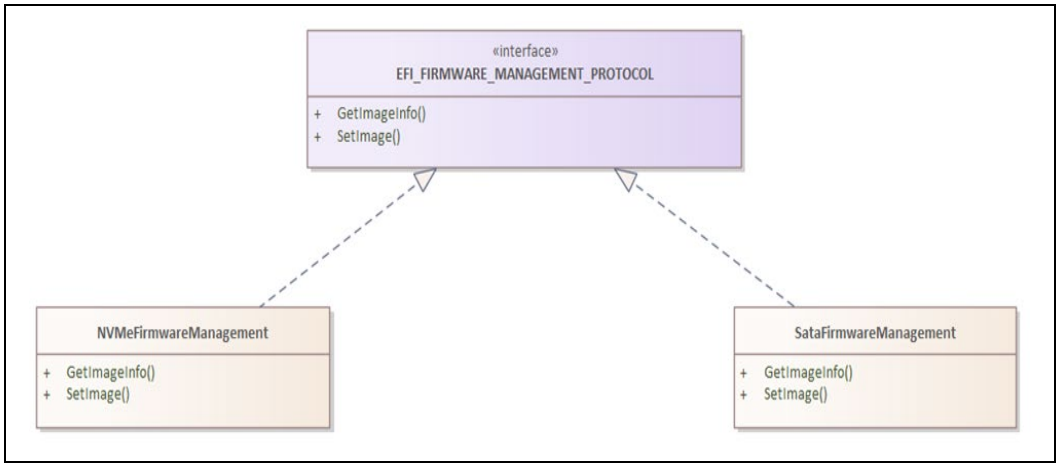
VROC UEFI driver implements EFI_FIRMWARE_MANAGEMENT_PROTOCOL specified in UEFI Specification version 2.9. It is used to manage the firmware of drives that are handled by Intel® VROC. VROC UEFI driver implements a subset of protocol functions defined in UEFI Specification, which is reflected in below table.

Table 3-1. Intel® VROC EFI Firmware Management Protocol Functions

EFI Firmware Management Protocol functions	VROC UEFI Implementation
GetImageInfo	Supported
GetImage	Not supported
SetImage	Supported
CheckImage	Not supported
GetPackageInfo	Not supported
SetPackageInfo	Not supported

Intel® VROC UEFI driver allows use of the EFI_FIRMWARE_MANAGEMENT_PROTOCOL to manage NVMe* drive firmware and SATA drive firmware.

Figure 3-1. Firmware Management Class Diagram



3.1 NVMe* Drives Firmware Management Protocol

This section describes implementation details of EFI_FIRMWARE_MANAGEMENT_PROTOCOL for NVMe* drives.

3.1.1 GetImageInfo

GetImageInfo function returns information about the current firmware image(s) of the NVMe drive. Below table specifies Intel® VROC UEFI driver implementation of values returned by the GetImageInfo function.

GetImageInfo Outputs	VROC UEFI Implementation
ImageInfoSize	As specified in UEFI Specification
ImageInfo	Refer to below table
DescriptorVersion	3
DescriptorCount	Number of NVMe* Firmware Slots with non-zero Firmware Revision
DescriptorSize	As specified in UEFI Specification
PackageVersion	0xFFFFFFFF
PackageVersionName	NULL

Table below specifies mapping firmware image information from NVMe* to EFI_FIRMWARE_IMAGE_DESCRIPTOR.

Table 3-2. NVMe* Drives EFI_FIRMWARE_IMAGE_DESCRIPTOR Implementation

EFI_FIRMWARE_IMAGE_DESCRIPTOR Member	VROC UEFI Implementation for NVMe* Drive
ImageIndex	Firmware Slot Index (1..N)
ImageTypeId	Preset GUID for NVMe (the same for all NVMe drives) 25D4CA70-05E8-44CB-B13D-ABA064717432
ImageId	Firmware Slot Index (1..N)
ImageIdName	“Drive model: <ModelNumber>, Firmware Slot: <FirmwareSlotIndex>”
Version	0
VersionName	Firmware Slot Information Log Page -> Firmware Revision for Slot [ImageIndex]
Size	1
AttributesSupported	IMAGE_ATTRIBUTE_IN_USE IMAGE_ATTRIBUTE_IMAGE_UPDATEABLE
AttributesSetting	IMAGE_ATTRIBUTE_IN_USE: If this Firmware Slot is active (pointed by AFI 2:0), then attribute set to 1, otherwise attribute cleared to 0 IMAGE_ATTRIBUTE_IMAGE_UPDATEABLE: If this is the first Firmware Slot and FRMW bit 0 is set, then attribute cleared to 0, otherwise attribute set to 1
Compatibilities	0
LowestSupportedImageVersion	3
LastAttemptVersion	0

<table>
<tr><th>EFI_FIRMWARE_IMAGE_DESCRIPTOR Member</th><th>VROC UEFI Implementation for NVMe* Drive</th></tr>
<tr><td>LastAttemptStatus</td><td>Status of last SetImage command if called, otherwise LAST_ATTEMPT_STATUS_SUCCESS</td></tr>
<tr><td>HardwareInstance</td><td>
<table>
<tr><th>Bits</th><th>Value (physical location of NVMe drive)</th></tr>
<tr><td>63:32</td><td>0</td></tr>
<tr><td>31:24</td><td>CPU Socket Number</td></tr>
<tr><td>23:16</td><td>VMD Controller Number</td></tr>
<tr><td>15:8</td><td>Root Port Offset</td></tr>
<tr><td>7:0</td><td>PCIe Slot Number</td></tr>
</table>
</td></tr>
</table>

In below table represents the number of Firmware Slots supported by the NVMe* controller, indicated by FRMW (bits 3:1) in NVMe* specification.

Since the firmware management capabilities of the NVMe* specification are limited:

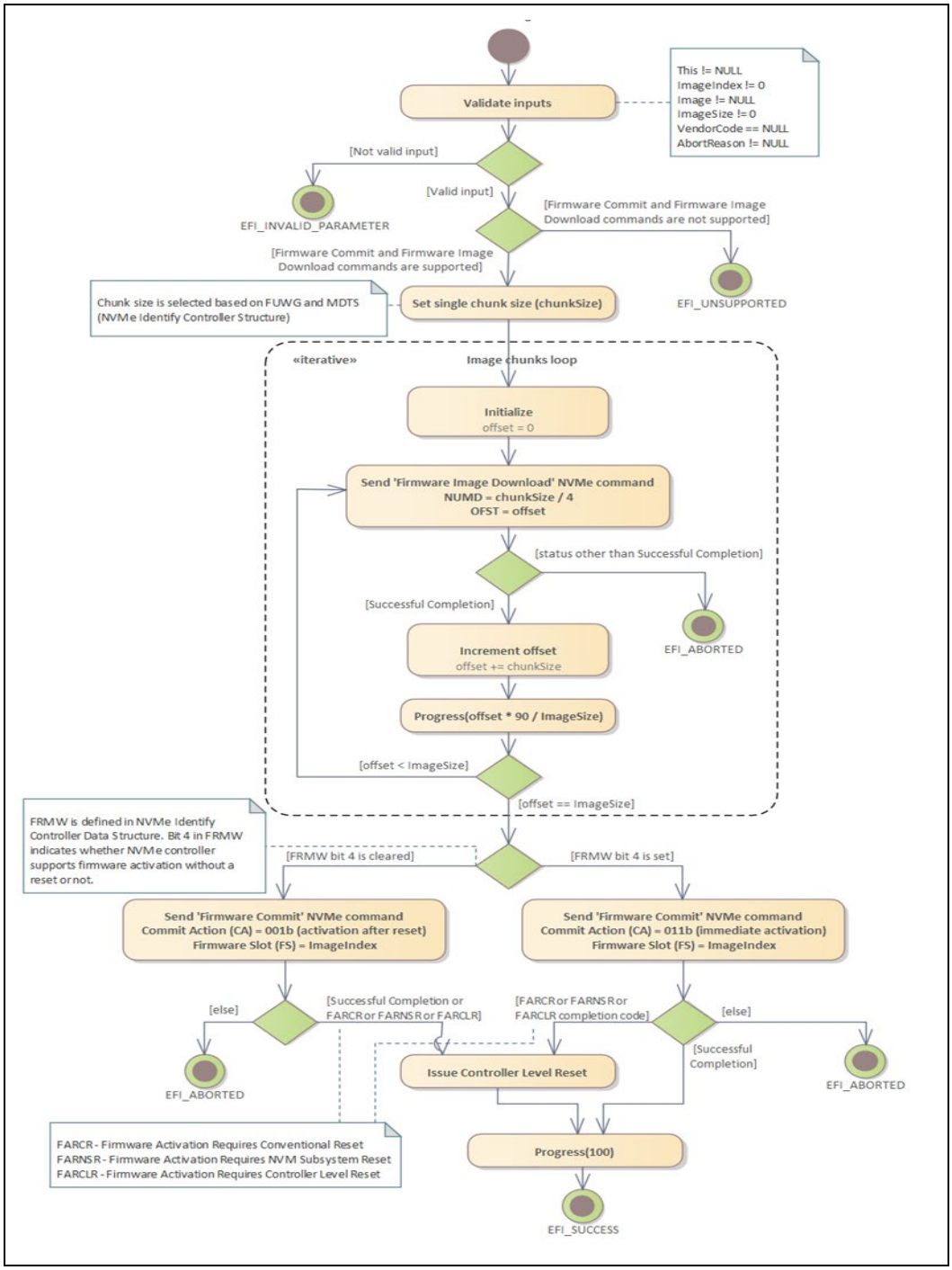
- Version and LastAttemptVersion are always equal to 0
- Size is always equal to 1

3.1.2 SetImage

Intel® VROC UEFI driver does not support specifying a firmware image update policy for the NVMe* drive. Thus, for EFI_FIRMWARE_MANAGEMENT_PROTOCOL instance for NVMe drive, the VendorCode argument of SetImage function shall be NULL. The rest of the SetImage arguments shall be provided according to the UEFI specification. If Progress function is provided by the caller, then the VROC UEFI driver reports the progress of the firmware update process. Arguments This, ImageIndex, Image, ImageSize and AbortReason are mandatory and shall be non-zero values.

In below figure the implementation of SetImage function for NVMe* drive is presented.

Figure 3-2. NVMe* Drive SetImage Implementation



When SetImage function is called on EFI_FIRMWARE_IMAGE_PROTOCOL installed on an NVMe* device handle, the Intel® VROC UEFI driver will first validate input. While input is valid, the driver checks if Firmware Commit and Firmware Image Download commands are supported by the NVMe* drive. If the commands are supported, the driver sends image in chunks to the drive using the Firmware Image Download NVMe* command. If sending all chunks succeeds, the driver follows one of two branches:

- If the drive does not support new firmware activation without reset, the driver sends the Firmware Commit NVMe* command with Commit Action equal to 001b; specifying the new firmware to be activated at the next Controller Level Reset. If the command successfully completes, the driver issues a Controller Level Reset.
- If the drive supports new firmware activation without reset, the driver sends the Firmware Commit NVMe* command with Commit Action equal to 011b; specifying the new firmware to be activated immediately. If the firmware activation is not successful because a reset is required to activate firmware, the VROC UEFI driver will issue a Controller Level Reset.

Note: Controller Level Reset does not guarantee that the new firmware is activated. The drive may require a NVM Subsystem Reset or Conventional Reset. These operations are not supported by Intel® VROC UEFI drivers. It is the caller's responsibility to check whether the new firmware is active or not by calling GetImageFunction.

In the final step, the driver reports 100% progress and returns EFI_SUCCESS.

3.2 Firmware Management Protocol for SATA Drives

This section describes implementation details of EFI_FIRMWARE_MANAGEMENT_PROTOCOL for SATA drives.

3.2.1 GetImageInfo

GetImageInfo function returns information about the current firmware image of the SATA drive. Table below specifies VROC UEFI driver implementation of values returned by the GetImageInfo function.

Table 3-3. GetImageInfo Outputs Implementation in Intel® VROC for SATA Drives

GetImageInfo outputs	VROC UEFI implementation
ImageInfoSize	As specified in UEFI Specification
ImageInfo	Refer to below table
DescriptorVersion	3
DescriptorCount	1
DescriptorSize	As specified in UEFI Specification
PackageVersion	0xFFFFFFFF
PackageVersionName	NULL

Table below specifies mapping firmware image information from ATA to EFI_FIRMWARE_IMAGE_DESCRIPTOR.

EFI_FIRMWARE_IMAGE_ DESCRIPTOR Member	VROC UEFI implementation for NVMe Drive
ImageIndex	1
ImageTypeId	Preset GUID for SATA 9C9A55B0-A5CD-456F-9AC3-76765D68733D
ImageId	1
ImageIdName	"Drive model: <ModelNumber>"
Version	0
VersionName	IDENTIFY DEVICE DATA Firmware Revision
Size	1
AttributesSupported	IMAGE_ATTRIBUTE_IN_USE IMAGE_ATTRIBUTE_IMAGE_UPDATEABLE
AttributesSetting	IMAGE_ATTRIBUTE_IN_USE: Always set to 1 IMAGE_ATTRIBUTE_IMAGE_UPDATEABLE: If DOWNLOAD MICROCODE or DOWNLOAD MICROCODE DMA is supported, then attribute set to 1, otherwise attribute cleared to 0
Compatibilities	0
LowestSupportedImageVersion	3
LastAttemptVersion	0
LastAttemptStatus	Status of last SetImage command if called, otherwise LAST_ATTEMPT_STATUS_SUCCESS
HardwareInstance	AHCI Port Number

Sine firmware management capabilities in ATA specification are limited:

Version and LastAttemptVersion are always equal to 0

Size is always equal to 1

3.2.2 SetImage

Intel® VROC UEFI driver allows to optionally specify firmware image update policy for SATA drive. Thus, for EFI_FIRMWARE_MANAGEMENT_PROTOCOL instance for a SATA drive, the VendorCode argument of SetImage function shall be NULL or shall be a pointer to the structure, defined as the following:

```
typedef struct _VROC_FW_UPDATE_POLICY
{
    UINT8 VrocFwUpdatePolicyVersion;                    //Version of this
structure
    UINT8 VrocFwUpdateAtaSubcommand; //ATA subcommand used to update SATA
drive's FW
} VROC_FW_UPDATE_POLICY;
```

VrocFwUpdatePolicyVersion shall be set to 1, because it is the first version of the structure.

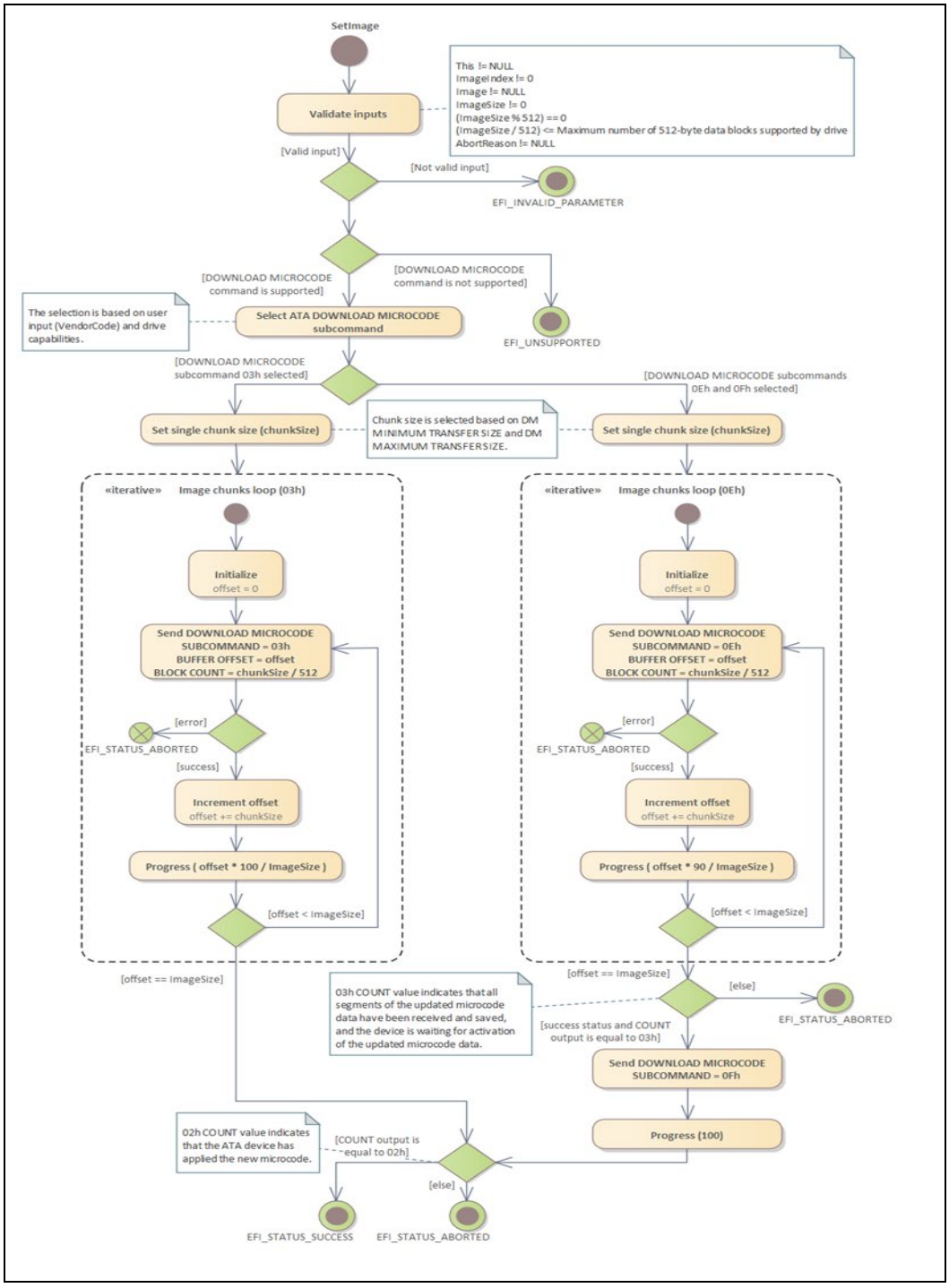
VrocFwUpdateAtaSubcommand shall be set to one of the following values:

```
#define VROC_FW_UPDATE_ATA_SUBCOMMAND_03                0x03
#define VROC_FW_UPDATE_ATA_SUBCOMMAND_0E_0F             0x0E
```

The VendorCode argument is optional, when SetImage function is called with NULL VendorCode, the VROC UEFI driver applies the default policy. The rest of the SetImage arguments must be provided according to the UEFI specification. If Progress function is provided by the caller, then VROC UEFI driver reports progress of firmware update process. Arguments This, ImageIndex, Image, ImageSize and AbortReason are mandatory and shall be non-zero values.

In figure below, the implementation of SetImage function for SATA drive is presented.

Figure 3-3. SATA Drive SetImage Implementation



When the SetImage function is called on EFI_FIRMWARE_IMAGE_PROTOCOL installed on a SATA device handle, the VROC UEFI driver firstly validates input. If the input is valid, the VROC UEFI driver selects the ATA subcommand based on user input (VendorCode) and drive capabilities. Then, the driver follows one of two branches:

1. If subcommand 03h is selected, the driver sends firmware in chunks to the drive and reports progress after sending every chunk.
2. If subcommands 0Eh and 0Fh are selected, the driver sends firmware in chunks to the drive using subcommand 0Eh and report progress after sending every chunk (up to 90%), and then activates new firmware using subcommand 0Fh and reports 100% progress.

If all commands are completed with success, the driver returns EFI_SUCCESS.

The selection policy for ATA DOWNLOAD MICROCODE subcommand is presented in table below as a decision matrix, where rows represent SATA drive capabilities, columns represent preferences provided by user (by VendorCode argument) and elements show one of the VROC UEFI driver flows:

- UNS – returns EFI_UNSUPPORTED status
- 03h – updates firmware using ATA DOWNLOAD MICROCODE subcommand 03h
- 0Eh, 0Fh – updates firmware using ATA DOWNLOAD MICROCODE subcommands 0Eh and 0Fh

Table 3-4. ATA DOWNLOAD MICROCODE Subcommand Decision Matrix

Supported Subcommands / User Preference	None	Prefer 03h	Prefer 0Eh, 0Fh
None	UNS	UNS	UNS
03h	03h	03h	UNS
0Eh, 0Fh	0Eh, 0Fh	UNS	0Eh, 0Fh
03h, 0Eh, 0Fh	0Eh, 0Fh	03h	0Eh, 0Fh

§§

