



Intel® Virtual RAID on CPU (Intel® VROC) & Intel® Rapid Storage Technology enterprise (Intel® RSTe)

Intel® Virtual RAID on CPU (Intel® VROC) Private UEFI Volume
Info Protocol

Intel Confidential

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

UNLESS OTHERWISE AGREED IN WRITING BY INTEL, THE INTEL PRODUCTS ARE NOT DESIGNED NOR INTENDED FOR ANY APPLICATION IN WHICH THE FAILURE OF THE INTEL PRODUCT COULD CREATE A SITUATION WHERE PERSONAL INJURY OR DEATH MAY OCCUR.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

The products described in this document 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.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Intel and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

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

Copyright © 2017, Intel Corporation. All rights reserved.

Contents

Contents 3

1 Summary 4

1.1 Protocol GUID 4

1.2 Protocol Interface structure 4

1.2.1 Parameters 4

1.2.2 Related Definitions 4

1.3 Description 5

1.4 Example 5

1.4.1 Example sample code output 8

1 Summary

Intel® Virtual Raid on CPU (Intel® VROC) version 5.3 provides a new protocol for retrieving RAID volume information.

Intel VROC volume info protocol can be used to retrieve information for all RAID volumes created thru Intel Rapid Storage Technology enterprise (Intel® RSTe) UEFI mode only, and Intel VROC. Changes can also be seen using the HII and Intel RSTe / Intel VROC UEFI tools.

1.1 Protocol GUID

```
// 6139796A-3DF0-4838-827B-427E7DC8281C}
#define EFI_VROC_VOLUME_INFO_PROTOCOL_GUID \
    { 0x6139796a, 0x3df0, 0x4838, { 0x82, 0x7b, 0x42, 0x7e, 0x7d, 0xc8, 0x28, 0x1c } }
```

1.2 Protocol Interface structure

```
#pragma pack( 1 )
typedef struct _VROC_VOLUME_INFO_PROTOCOL
{
    EFI_STRING  VendorID;
    EFI_STRING  ProductID;
    EFI_STRING  VolumeName;
    UINT64      TotalBlocks;
    UINTN       BlockSize;
    UINT8       RaidLevel;
    UINT8       VolumeType;
} VROC_VOLUME_INFO_PROTOCOL;
#pragma pack()
```

1.2.1 Parameters

<i>VendorID</i>	Volume vendor identification string.
<i>ProductID</i>	Product identification string.
<i>VolumeName</i>	Name of a RAID volume.
<i>TotalBlocks</i>	Total size of volume, in blocks.
<i>BlockSize</i>	Logical block size in bytes.
<i>RaidLevel</i>	RAID Volume level (see related definitions below).
<i>VolumeType</i>	RAID Volume type, determines if volume was created as VROC, SATA or SSATA (see related definitions below).

1.2.2 Related Definitions

Available RAID levels:

```
#define VROC_VOLUME_RAID0 0 // RAID volume level 0 (Stripe)
#define VROC_VOLUME_RAID1 1 // RAID volume level 1 (Mirror)
#define VROC_VOLUME_RAID5 5 // RAID volume level 5 (Parity)
#define VROC_VOLUME_RAID10 10 // RAID volume level 10 (RAID level 0 + 1)
```

Available volume types:

```
#define VROC_VOLUME_SATA      0    // RAID volume created on SATA controller.
#define VROC_VOLUME_sSATA    1    // RAID volume created on sSATA controller.
#define VROC_VOLUME_VROC     2    // RAID volume created on VROC controller.
```

1.3 Description

This protocol is installed on each handle of Intel RSTe/Intel VROC RAID volume.

Caller should not try to deallocate or change values of the memory under returned pointer as this can lead to undefined behavior. This is Intel driver responsibility to allocate and free this memory area whenever it is necessary.

If caller service intends to keep device information i.e. for a life time of a global structure instance, a copy of interface memory should be made and protocol should be closed. This is dictated by fact, that protocol data can change in time if device enumeration is done for i.e. create or delete volume.

To calculate device capacity, TotalBlocks and BlockSize parameters can be used. Following example shows how to get disk size in bytes:

```
UINT64 sizeInBytes = result->TotalBlocks * result->BlockSize;
```

1.4 Example

Below code is just a sample code, it shows usage of Intel VROC volume info protocol from a UEFI Shell application.

```
#include <Uefi.h>
#include <Library/UefiApplicationEntryPoint.h>
#include <Library/UefiBootServicesTableLib.h>
#include <Library/UefiLib.h>

/**
  VROC Private Volume Info Protocol GUID.
  {6139796A-3DF0-4838-827B-427E7DC8281C}
**/
#define EFI_VROC_VOLUME_INFO_PROTOCOL_GUID \
  { 0x6139796a, 0x3df0, 0x4838, { 0x82, 0x7b, 0x42, 0x7e, 0x7d, 0xc8, 0x28, 0x1c } }

#define VROC_VOLUME_RAID0      0
#define VROC_VOLUME_RAID1      1
#define VROC_VOLUME_RAID5      5
#define VROC_VOLUME_RAID10     10

#define VROC_VOLUME_SATA      0
#define VROC_VOLUME_sSATA    1
#define VROC_VOLUME_VROC     2

/**
  VROC Private Volume Info Protocol GUID global variable.
**/
EFI_GUID gEfiVROCVolumeInfoProtocolGuid = EFI_VROC_VOLUME_INFO_PROTOCOL_GUID;

/**
  RSTe Volume Info protocol interface.
**/
#pragma pack( 1 )
typedef struct _VROC_VOLUME_INFO_PROTOCOL
```

```

{
    EFI_STRING VendorID;
    EFI_STRING ProductID;
    EFI_STRING VolumeName;
    UINT64      TotalBlocks;
    UINTN       BlockSize;
    UINT8       RaidLevel;
    UINT8       VolumeType;
} VROC_VOLUME_INFO_PROTOCOL;
#pragma pack()

/**
    The user Entry Point for Application.

    @param[in] ImageHandle    The firmware allocated handle for the EFI image.
    @param[in] SystemTable    A pointer to the EFI System Table.

    @retval     EFI_SUCCESS    The entry point is executed successfully.
    @retval     other          Some error occurs when executing this entry point.

**/
EFI_STATUS
EFIAPI
UefiMain(
    IN EFI_HANDLE      ImageHandle,
    IN EFI_SYSTEM_TABLE *SystemTable
)
{
    EFI_HANDLE      * pHandleBuf = NULL;
    UINTN           nHandles = 0;
    UINTN           nIdx = 0;
    EFI_STATUS       nStatus = EFI_SUCCESS;
    VROC_VOLUME_INFO_PROTOCOL * result = NULL;

    Print(L"\nIntel(R) VROC Volume Info Protocol Example.\n\n");

    Print(L"Searching for RSTE_VOLUME_INFO_PROTOCOL instances...\n");

    nStatus = gBS->LocateHandleBuffer(
        ByProtocol,
        &gEfiVROCVolumeInfoProtocolGuid,
        NULL,
        &nHandles,
        &pHandleBuf
    );

    if (nStatus == EFI_OUT_OF_RESOURCES)
    {
        Print(L"ERROR: There is not enough pool memory to store the matching results.\n");
        return nStatus;
    }

    if (nStatus == EFI_INVALID_PARAMETER)
    {
        Print(L"ERROR: One of the parameters has an invalid value.\n");
        return nStatus;
    }

    if (nStatus == EFI_NOT_FOUND)
    {
        Print(L"No instance of VROC volume info protocol was found.\n");
        return nStatus;
    }

    Print(L"Found %d instances.\n\n", nHandles);
    for (nIdx = 0; nIdx < nHandles; nIdx++)
    {
        if (pHandleBuf[nIdx])
        {

```

```

Print(L"%d] Opening RSTE_VOLUME_INFO_PROTOCOL.\n", nIdx);

nStatus = gBS->OpenProtocol(
    pHandleBuf[nIdx],
    &gEfiVROCVolumeInfoProtocolGuid,
    (void*)&result,
    ImageHandle,
    NULL,
    EFI_OPEN_PROTOCOL_BY_HANDLE_PROTOCOL
);

if (nStatus == EFI_INVALID_PARAMETER)
{
    Print(L"%d] ERROR: One of the parameters has an invalid value.\n", nIdx);
    continue;
}

if (nStatus == EFI_ACCESS_DENIED || nStatus == EFI_ALREADY_STARTED)
{
    Print(L"%d] ERROR: Already opened with BY_DRIVER or EXCLUSIVE attribute.\n", nIdx);
    continue;
}

if (result)
{
    UINT64 sizeInBytes = result->TotalBlocks * result->BlockSize;

    Print(L"%d] %s %s (%s)\n",
        nIdx,
        result->VendorID,
        result->ProductID,
        result->VolumeName
    );

    Print(L"        Total blocks: %ld\n", result->TotalBlocks);
    Print(L"        Block size: %d\n", result->BlockSize);
    Print(L"        Size in bytes: %ld\n", sizeInBytes);
    Print(L"        Raid Level: %d\n", result->RaidLevel);

    switch (result->VolumeType)
    {
    case VROC_VOLUME_SATA:
        Print(L"        Volume Type: SATA\n");
        break;
    case VROC_VOLUME_sSATA:
        Print(L"        Volume Type: SSATA\n");
        break;
    default:
        Print(L"        Volume Type: VROC\n");
        break;
    }
}
else
{
    Print(L"%d] ERROR: No interface returned.\n", nIdx);
}

Print(L"%d] Closing VROC_VOLUME_INFO_PROTOCOL.\n\n", nIdx);

gBS->CloseProtocol(
    pHandleBuf[nIdx],
    &gEfiVROCVolumeInfoProtocolGuid,
    ImageHandle,
    NULL
);
}
}

gBS->FreePool(pHandleBuf);

```

```
    return EFI_SUCCESS;  
}
```

1.4.1 Example sample code output

```
FS0:\> rstevRSTeVolumeInfo.efi  
  
Intel(R) VROC Volume Info Potocol Example.  
  
Searching for RSTE_VOLUME_INFO_PROTOCOL instances...  
Found 3 instances.  
  
[0] Opening RSTE_VOLUME_INFO_PROTOCOL.  
[0] Intel VROC RAID 1 Volume (Volume001)  
    Total blocks: 742340608  
    Block size: 512  
    Size in bytes: 380078391296  
    Raid Level: 1  
    Volume Type: VROC  
[0] Closing VROC_VOLUME_INFO_PROTOCOL.  
  
[1] Opening RSTE_VOLUME_INFO_PROTOCOL.  
[1] Intel RAID 1 Volume (Volume002)  
    Total blocks: 927924224  
    Block size: 512  
    Size in bytes: 475097202688  
    Raid Level: 1  
    Volume Type: SATA  
[1] Closing VROC_VOLUME_INFO_PROTOCOL.  
  
[2] Opening RSTE_VOLUME_INFO_PROTOCOL.  
[2] Intel RAID 0 Volume (Volume003)  
    Total blocks: 92776448  
    Block size: 512  
    Size in bytes: 47501541376  
    Raid Level: 0  
    Volume Type: SATA  
[2] Closing VROC_VOLUME_INFO_PROTOCOL.  
  
FS0:\>
```