NVIDIA DOCA DMA Copy

Application Guide

MLNX-15-060570 _v1.5.2 | July 2023

Table of Contents

Chapter 1. Introduction..... 1
Chapter 2. System Design..... 2
Chapter 3. Application Architecture..... 3
Chapter 4. DOCA Libraries..... 5
Chapter 5. Configuration Flow..... 6
Chapter 6. Dependencies..... 8
Chapter 7. Running the Application..... 9
Chapter 8. Arg Parser DOCA Flags..... 11
Chapter 9. References..... 13

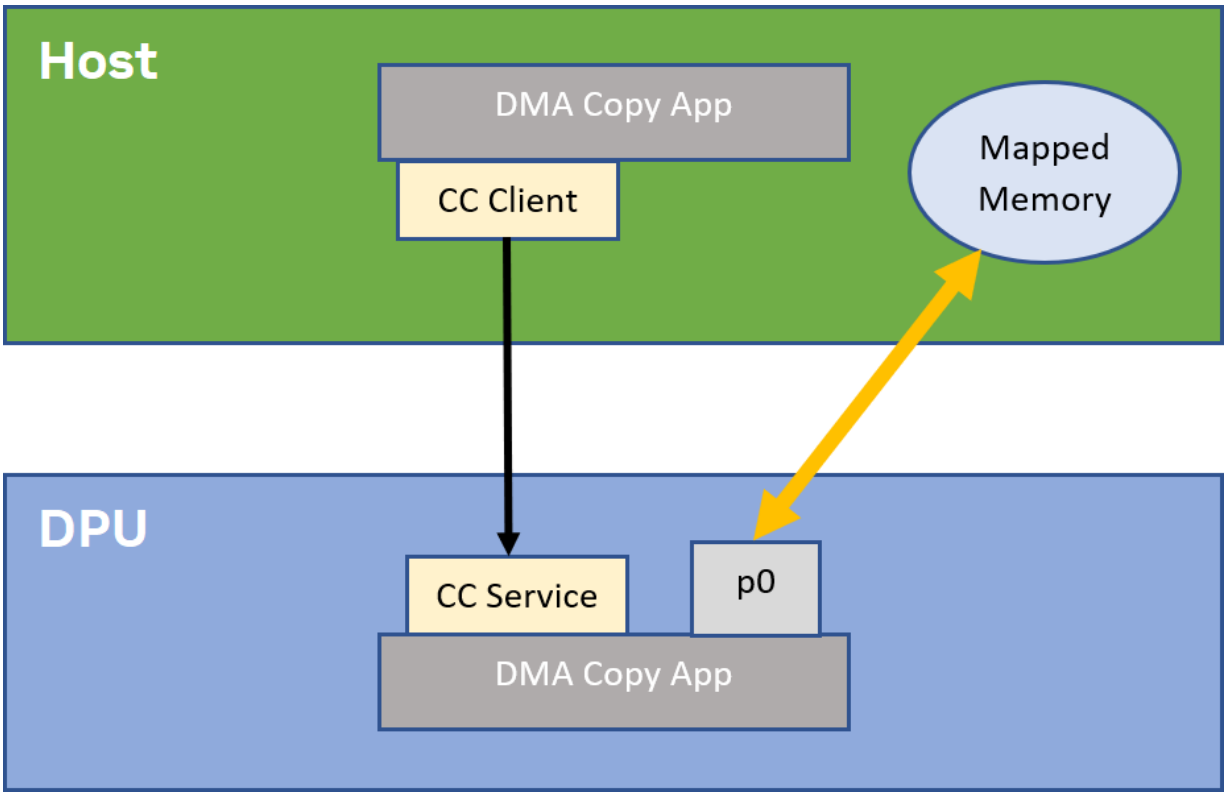
Chapter 1. Introduction

DOCA DMA (direct memory access) Copy application transfers files (data path) up to 1MB in size between the DPU and the x86 host using the DOCA DMA library which provides an API to copy data between DOCA buffers using hardware acceleration, supporting both local and remote memory.

DOCA DMA allows complex memory copy operations to be easily executed in an optimized, hardware-accelerated manner.

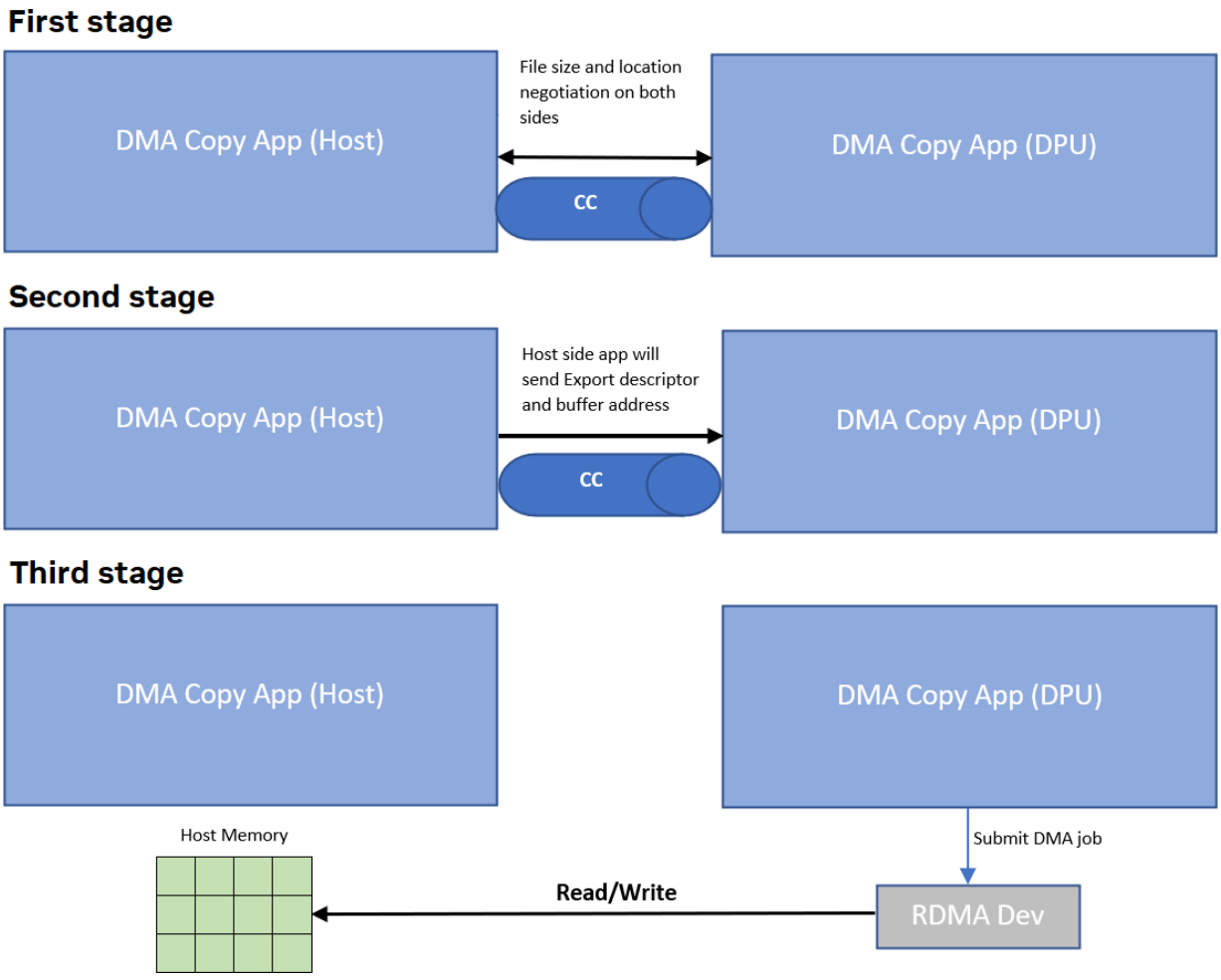
Chapter 2. System Design

DOCA DMA Copy is designed to run on both the instances of the BlueField-2 DPU and x86 host. The DPU application must be the first to spawn as it opens the Comm Channel (CC) service between the two sides on which all the necessary DMA library configuration files (control path) are transferred.



Chapter 3. Application Architecture

DOCA DMA Copy runs on top of DOCA DMA to read/write directly from the host's memory without any user/kernel space context switches, allowing for a fast memory copy.



1. The two sides initiate a short negotiation in which the file size and location are determined.
2. Host side creates the export descriptor with `doca_mmap_export()` and sends it with the local buffer address and length on the Comm Channel to the DPU side application.

3. DPU side application uses the received export descriptor to create a remote memory map locally with `doca_mmap_create_from_export()` and the host buffer information to create a remote DOCA buffer. From this point on, the DPU side application has all the needed memory information and the DMA copy will take place.

Chapter 4. DOCA Libraries

This application leverages following DOCA libraries:

- DOCA Comm Channel
- DOCA DMA Programming Guide

Chapter 5. Configuration Flow

1. Parse application argument.
   a). Initialize arg parser resources and register DOCA general parameters.
   ```
   doca_argp_init();
   ```
   b). Register application parameters.
   ```
   register_dma_copy_params();
   ```
   c). Parse app flags.
   ```
   doca_argp_start();
   ```
2. Initialize DOCA App Shield lib context.
   ```
   init_cc();
   ```
   a). Create Comm Channel endpoint.
   b). Parse user PCIe address for Comm Channel device.
   c). Open Comm Channel DOCA device.
   d). Parse user PCIe address for Comm Channel device representor (on DPU side).
   e). Open Comm Channel DOCA device representor (on DPU side).
   f). Set Comm Channel endpoint properties.
3. Open the DOCA hardware device from which the copy would be made.
   ```
   open_dma_device();
   ```
   a). Parse the PCIe address provided by the user.
   b). Create a list of all available DOCA devices.
   c). Find the appropriate DOCA device according to specific properties.
   d). Open the device.
4. Create all required DOCA core objects.
   ```
   create_core_objects();
   ```
5. Initiate DOCA core objects.
   ```
   init_core_objects();
   ```
6. Start host/DPU DMA copy.
   a). Host-side application:
   ```
   host_start_dma_copy();
   ```
   i. Start negotiation with the DPU side application for the location and size of the file.
   ii. Allocate memory for the DMA buffer.
   iii. Export the memory map and send the output (export descriptor) to the DPU side application.

iv. Send the host local buffer memory address and length on the Comm Channel to the DPU side application.
v. Wait for the DPU to notify that DMA Copy has ended.
vi. Close all memory objects.
vii. Clean resources.

b). DPU-side application:

```
dpu_start_dma_copy();
```

i. Start negotiation with the host side application for file location and size.
ii. Allocate memory for the DMA buffer.
iii. Receive the export descriptor on the Comm Channel.
iv. Create the DOCA memory map for the remote buffer on the host.
v. Receive the host buffer information on the Comm Channel.
vi. Create two DOCA buffers, one for the remote (host) buffer and one for the local buffer.
vii. Submit the DMA job into the work queue.
viii. Send a host message to notify that DMA copy ended.
ix. Clean resources.

7. Destroy Comm Channel.

```
destroy_cc();
```

8. Destroy DOCA core objects.

```
destroy_core_objects();
```

9. Arg parser destroy.

```
doca_argp_destroy();
```

Chapter 6. Dependencies

The minimum required firmware version is 24.32.1010.

Chapter 7. Running the Application

1. Refer to the following documents:
   - NVIDIA DOCA Installation Guide for Linux for details on how to install BlueField-related software.
   - NVIDIA DOCA Troubleshooting Guide for any issue you may encounter with the installation, compilation, or execution of DOCA applications.
   - NVIDIA DOCA Applications Overview for additional compilation instructions and development tips of DOCA applications.
2. The DMA Copy example binary is located under `/opt/mellanox/doca/applications/dma_copy/bin/doca_dma_copy`. To build all the applications together, run:

```
cd /opt/mellanox/doca/applications/
meson build
ninja -C build
```

3. To build only the DMA Copy application:

   a). Edit the following flags in `/opt/mellanox/doca/applications/meson_options.txt`:

   - Set `enable_all_applications` to `false`
   - Set `enable_dma_copy` to `true`

   b). Run the commands in step 2.

> **Note:** `doca_dma_copy` will be created under `./build/dma_copy/src/`.

Application usage:

```
Usage: doca_dma_copy [DPDK Flags] -- [DOCA Flags] [Program Flags]

DOCA Flags:
  -h, --help                        Print a help synopsis
  -v, --version                     Print program version information
  -l, --log-level                   Set the log level for the program
 <CRITICAL=20, ERROR=30, WARNING=40, INFO=50, DEBUG=60>

Program Flags:
  -f, --file                        Full path to file to be copied/created after
 a successful DMA copy
  -d, --dev-pci                     Comm Channel DOCA device PCI address
  -r, --rep-pci                     Comm Channel DOCA device representor PCI
 address
```

4. Running the application on BlueField, CLI example:
```
/opt/mellanox/doca/applications/dma_copy/bin/doca_dma_copy -d 03:00.1 -r b1:00.1
 -f /tmp/file_to_read.txt
```
5. Running the application on the host, CLI example:
```
/opt/mellanox/doca/applications/dma_copy/bin/doca_dma_copy -d b1:00.1 -f /tmp/
file_to_create.txt
```

Note: Refer to section "Running DOCA Application on Host" in NVIDIA DOCA Virtual Functions User Guide.

6. To run `doca_dma_copy` using a JSON file:
```
doca_dma_copy --json [json_file]
```
For example:
```
cd /opt/mellanox/doca/applications/dma_copy/bin
./doca_dma_copy --json /root/dma_copy_params.json
```

Chapter 8. Arg Parser DOCA Flags

Refer to NVIDIA DOCA Arg Parser User Guide for more information.

Flag Type	Short Flag	Long Flag/JSON Key	Description	JSON Content
General flags	`l`	`log-level`	Set the log level for the application: ▶ CRITICAL=20 ▶ ERROR=30 ▶ WARNING=40 ▶ INFO=50 ▶ DEBUG=60	`"log-level": 60`
	`v`	`version`	Print program version information	N/A
	`h`	`help`	Print a help synopsis	N/A
Program flags	`f`	`file`	Full path to file to be copied/created after a successful copy **Note:** This is a mandatory flag.	`"file": "/tmp/sample.txt"`
	`p`	`dev-pci`	Comm Channel DOCA device PCIe address **Note:** This is a	`"dev-pci": "b1:00.0"`

Flag Type	Short Flag	Long Flag/JSON Key	Description	JSON Content
			mandatory flag.	
	r	rep-pci	Comm Channel DOCA device representor PCIe address **Note:** This is a mandatory flag only on the DPU.	`"rep-pci": "b1:02.0"`

Chapter 9. References

- /opt/mellanox/doca/applications/dma_copy/src/dma_copy.c
- /opt/mellanox/doca/applications/dma_copy/src/dma_copy_core.c
- /opt/mellanox/doca/applications/dma_copy/src/dma_copy_core.h

Notice

This document is provided for information purposes only and shall not be regarded as a warranty of a certain functionality, condition, or quality of a product. NVIDIA Corporation nor any of its direct or indirect subsidiaries and affiliates (collectively: "NVIDIA") make no representations or warranties, expressed or implied, as to the accuracy or completeness of the information contained in this document and assume no responsibility for any errors contained herein. NVIDIA shall have no liability for the consequences or use of such information or for any infringement of patents or other rights of third parties that may result from its use. This document is not a commitment to develop, release, or deliver any Material (defined below), code, or functionality.

NVIDIA reserves the right to make corrections, modifications, enhancements, improvements, and any other changes to this document, at any time without notice.

Customer should obtain the latest relevant information before placing orders and should verify that such information is current and complete.

NVIDIA products are sold subject to the NVIDIA standard terms and conditions of sale supplied at the time of order acknowledgement, unless otherwise agreed in an individual sales agreement signed by authorized representatives of NVIDIA and customer ("Terms of Sale"). NVIDIA hereby expressly objects to applying any customer general terms and conditions with regards to the purchase of the NVIDIA product referenced in this document. No contractual obligations are formed either directly or indirectly by this document.

NVIDIA products are not designed, authorized, or warranted to be suitable for use in medical, military, aircraft, space, or life support equipment, nor in applications where failure or malfunction of the NVIDIA product can reasonably be expected to result in personal injury, death, or property or environmental damage. NVIDIA accepts no liability for inclusion and/or use of NVIDIA products in such equipment or applications and therefore such inclusion and/or use is at customer's own risk.

NVIDIA makes no representation or warranty that products based on this document will be suitable for any specified use. Testing of all parameters of each product is not necessarily performed by NVIDIA. It is customer's sole responsibility to evaluate and determine the applicability of any information contained in this document, ensure the product is suitable and fit for the application planned by customer, and perform the necessary testing for the application in order to avoid a default of the application or the product. Weaknesses in customer's product designs may affect the quality and reliability of the NVIDIA product and may result in additional or different conditions and/or requirements beyond those contained in this document. NVIDIA accepts no liability related to any default, damage, costs, or problem which may be based on or attributable to: (i) the use of the NVIDIA product in any manner that is contrary to this document or (ii) customer product designs.

No license, either expressed or implied, is granted under any NVIDIA patent right, copyright, or other NVIDIA intellectual property right under this document. Information published by NVIDIA regarding third-party products or services does not constitute a license from NVIDIA to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property rights of the third party, or a license from NVIDIA under the patents or other intellectual property rights of NVIDIA.

Reproduction of information in this document is permissible only if approved in advance by NVIDIA in writing, reproduced without alteration and in full compliance with all applicable export laws and regulations, and accompanied by all associated conditions, limitations, and notices.

THIS DOCUMENT AND ALL NVIDIA DESIGN SPECIFICATIONS, REFERENCE BOARDS, FILES, DRAWINGS, DIAGNOSTICS, LISTS, AND OTHER DOCUMENTS (TOGETHER AND SEPARATELY, "MATERIALS") ARE BEING PROVIDED "AS IS." NVIDIA MAKES NO WARRANTIES, EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE MATERIALS, AND EXPRESSLY DISCLAIMS ALL IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE. TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL NVIDIA BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF NVIDIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Notwithstanding any damages that customer might incur for any reason whatsoever, NVIDIA's aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms of Sale for the product.

Trademarks

NVIDIA, the NVIDIA logo, and Mellanox are trademarks and/or registered trademarks of Mellanox Technologies Ltd. and/or NVIDIA Corporation in the U.S. and in other countries. The registered trademark Linux® is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a world¬wide basis. Other company and product names may be trademarks of the respective companies with which they are associated.

Copyright

NVIDIA Corporation | 2788 San Tomas Expressway, Santa Clara, CA 95051
http://www.nvidia.com