Mellanox DPDK

Quick Start Guide

2.2_2.7

NOTE:
THIS HARDWARE, SOFTWARE OR TEST SUITE PRODUCT (“PRODUCT(S)”) AND ITS RELATED DOCUMENTATION ARE PROVIDED BY MELLANOX TECHNOLOGIES “AS-IS” WITH ALL FAULTS OF ANY KIND AND SOLELY FOR THE PURPOSE OF AIDING THE CUSTOMER IN TESTING APPLICATIONS THAT USE THE PRODUCTS IN DESIGNATED SOLUTIONS. THE CUSTOMER'S MANUFACTURING TEST ENVIRONMENT HAS NOT MET THE STANDARDS SET BY MELLANOX TECHNOLOGIES TO FULLY QUALIFY THE PRODUCT(S) AND/OR THE SYSTEM USING IT. THEREFORE, MELLANOX TECHNOLOGIES CANNOT AND DOES NOT GUARANTEE OR WARRANT THAT THE PRODUCTS WILL OPERATE WITH THE HIGHEST QUALITY. ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT ARE DISCLAIMED. IN NO EVENT SHALL MELLANOX BE LIABLE TO CUSTOMER OR ANY THIRD PARTIES FOR ANY DIRECT, INDIRECT, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES OF ANY KIND (INCLUDING, BUT NOT LIMITED TO, PAYMENT FOR PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY FROM THE USE OF THE PRODUCT(S) AND RELATED DOCUMENTATION EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Mellanox Technologies
350 Oakmead Parkway Suite 100
Sunnyvale, CA 94085
U.S.A.
www.mellanox.com
Tel: (408) 970-3400
Fax: (408) 970-3403

Table of Contents

List of Tables

Document Revision History

Table 1: Document Revision History

Revision	Description

2.2_2.7	• Added the following sections:
	  • ConnectX-4 Command Line Parameters
	  • ConnectX-4/ConnectX-4 Lx ESX: VMWare
	  • ConnectX-4/ConnectX-4 Lx ESX: VM Virtual Machine
	• Updated the following sections:
	  • ConnectX-3/ConnectX-3 Pro Bare Metal
	  • ConnectX-3/ConnectX-3 Pro KVM Hypervisor
	  • ConnectX-3/ConnectX-3 Pro KVM Virtual Machine
	• Removed the following sections:
	  • ConnectX-3/ConnectX-3 Pro ESX: VMWare
	  • ConnectX-3/ConnectX-3 Pro ESX: VM Virtual Machine

2.2_2.3	• Updated the following sections:
	  • ConnectX-4/ConnectX-4 Lx Bare Metal
	  • ConnectX-3/ConnectX-3 Pro KVM Hypervisor
	  • ConnectX-3/ConnectX-3 Pro KVM Virtual Machine
	  • ConnectX-3/ConnectX-3 Pro ESX: VM Virtual Machine
	  • ConnectX-4/ConnectX-4 Lx Bare Metal
	  • General Settings
	  • ConnectX®-4 / ConnectX®-4 Lx
	• Removed the following section:
	  • PowerKVM Settings

2.2_2.1	• Updated the following sections:
	  • ConnectX-4/ConnectX-4 Lx Bare Metal
	• Removed the following sections:
	  • KVM Hypervisor
	  • KVM Virtual Machine

2.2_1.6	• Added the following sections:
	  • TX VLAN Insertion Configuration on ConnectX-4/ConnectX-4 Lx
	  • Enable HW Packet Padding
	  • ConnectX-3/ConnectX-3 Pro ESX: VMWare
	  • ConnectX-3/ConnectX-3 Pro ESX: VM Virtual Machine

2.1_1.1	• Added ConnectX-4 settings in section General Settings. • Updated ConnectX-4 examples in section ConnectX®-4 / ConnectX®-4 Lx
	Major restructure due to ConnectX-4/ConnectX-4 Lx support.
2.0_2.8.4	Initial Release

Related Documents

The following table lists the documents referenced in this User Manual.

Table 2: Reference Documents

Document Name	Description
Mellanox OFED Linux Release Notes	Describes the new features and changes of the latest MLNX_OFED release.
Mellanox OFED Linux User Manual	Provides general information on the installation, configuration, management and maintenance of the software and hardware of VPI (InfiniBand, Ethernet) adapter cards.
Mellanox DPDK Release Notes	Describes the new features and changes of the latest Mellanox DPDK

1 Overview

This is the Quick Start Guide for mlx4 and mlx5 DPDK Poll-Mode Driver (PMD) for Mellanox ConnectX®-3/ConnectX®-3 Pro and ConnectX®-4/ ConnectX-4 Lx Ethernet adapters.

2 MLNX_DPDK Installation and Configuration

2.1 Installing ConnectX-3/ConnectX-3 Pro

2.1.1 ConnectX-3/ConnectX-3 Pro Bare Metal

1. Install MLNX_OFED v3.3-1.0.0.0.

   MLNX_OFED 3.3-1.0.0.0 can be downloaded from the Mellanox site:

   http://www.mellanox.com/page/products_dyn?product_family=26&mtag=linux

2. Verify that ConnectX-3/ConnectX-3 Pro firmware is 2.36.5000 (Use ibstat command).
3. Set all the ports to Ethernet.

   ```
   connectx_port_config
   ```

   and follow the instructions on the screen.

   For further instructions on how to run the script, please refer to the MLNX_OFED User Manual.

4. Add the following line to `/etc/modprobe.d/mlx4_core.conf`.

   ```
   options mlx4_core log_num_mgm_entry_size=-7
   ```

   NOTE: If VLAN filtering is used, set `log_num_mgm_entry_size=-1`.

   NOTE: Please be aware, perfromance penalty can occur in this case.

5. Restart MLNX_OFED.

   ```
   /etc/init.d/openib restart
   ```

6. Extract the package MLNX_DPDK _2.2_2.7.tar.gz

   The default mlx4 configuration in `config/common_linuxapp` is the following:

   ```
   #
   # Compile burst-oriented Mellanox ConnectX-3 (MLX4) PMD
   #
   CONFIG_RTE_LIBRTE_MLX4_PMD=y
   CONFIG_RTE_LIBRTE_MLX4_DEBUG=n
   CONFIG_RTE_LIBRTE_MLX4_SGE_WR_N=1
   CONFIG_RTE_LIBRTE_MLX4_MAX_INLINE=0
   CONFIG_RTE_LIBRTE_MLX4_TX_MP_CACHE=8
   ```

```
CONFIG_RTE_LIBRTE_MLX4_SOFT_COUNTERS=1
```

7. Compile DPDK.

```
make install T=x86_64-native-linuxapp-gcc
```

For more advanced DPDK compilation options, please refer to dpdk.org documentation:

http://www.dpdk.org/doc/guides/linux_gsg/index.html

8. Configure huge pages according to the NUMA the card is connected to

```
echo $PAGE_NUM >
/sys/devices/system/node/node$NUMA_NODE/hugepages/hugepages-
"$PAGE_SIZE"kB/nr_hugepages

mkdir -p /mnt/huge

HTLB_MOUNTED=$( mount | grep "type hugetlbfs" | wc -l)
if [ $HTLB_MOUNTED -eq 0 ]; then
        mount -t hugetlbfs hugetlb /mnt/huge
fi
```

2.1.2 ConnectX-3/ConnectX-3 Pro KVM

2.1.2.1 ConnectX-3/ConnectX-3 Pro KVM Hypervisor

1. Download MLNX_OFED 3.3-1.0.0.0.

   MLNX_OFED 3.3-1.0.0.0 can be downloaded from the Mellanox site:

   http://www.mellanox.com/page/products_dyn?product_family=26&mtag=linux

2. Install MLNX_OFED v3.3-1.0.0.0 and enable SR-IOV.

```
./mlnxofedinstall --enable-sriov -hypervisor
```

3. Please follow MLNX_OFED User Manual instructions how to enable SR-IOV.
4. Verify that ConnectX-3/ConnectX-3 Pro firmware is **2.36.5000** (Use ibstat command).
5. Add the following line to `/etc/modprobe.d/mlx4_core.conf`.

```
options mlx4_core log_num_mgm_entry_size=-7
```

NOTE: If VLAN filtering is used, set `log_num_mgm_entry_size=-1`.

Please be aware, perfromance penalty can occur in this case.

6. Restart MLNX_OFED.

```
/etc/init.d/openibd restart
```

2.1.2.2 ConnectX-3/ConnectX-3 Pro KVM Virtual Machine

1. Download MLNX_OFED 3.3-1.0.0.0.

   MLNX_OFED 3.3-1.0.0.0 can be downloaded from the Mellanox site:

   http://www.mellanox.com/page/products_dyn?product_family=26&mtag=linux

2. Install MLNX_OFED v3.3-1.0.0.0 and enable the guest.

```
./mlnxofedinstall --guest
```

3. Extract the package MLNX_DPDK_ 2.2_2.7.tar.gz

   The default mlx4 configuration in `config/common_linuxapp` is the following:

```
#
# Compile burst-oriented Mellanox ConnectX-3 (MLX4) PMD
#
CONFIG_RTE_LIBRTE_MLX4_PMD=y
CONFIG_RTE_LIBRTE_MLX4_DEBUG=n
CONFIG_RTE_LIBRTE_MLX4_SGE_WR_N=1
CONFIG_RTE_LIBRTE_MLX4_MAX_INLINE=0
CONFIG_RTE_LIBRTE_MLX4_TX_MP_CACHE=8
CONFIG_RTE_LIBRTE_MLX4_SOFT_COUNTERS=1
```

4. Compile DPDK.

```
make install T=x86_64-native-linuxapp-gcc
```

   For more advanced DPDK compilation options, please refer to dpdk.org documentation:

   http://www.dpdk.org/doc/guides/linux_gsg/index.html

5. Configure huge pages.

```
echo $PAGE_NUM > /sys/devices/system/node/node0/hugepages/hugepages-
"$PAGE_SIZE"kB/nr_hugepages

mkdir -p /mnt/huge

HTLB_MOUNTED=$( mount | grep "type hugetlbfs" | wc -l)
if [ $HTLB_MOUNTED -eq 0 ]; then
        mount -t hugetlbfs hugetlb /mnt/huge
fi
```

2.2 Installing ConnectX-4/ConnectX-4 Lx

2.2.1 ConnectX-4/ConnectX-4 Lx Bare Metal

- ***To upgrade from MLNX_DPDK_BETA 2.2_2.1 compiled with MLNX_OFED 3.2_2.0.0.0:***

1. Uninstall the previous MLNX_OFED version.

```
./ofed_uninstall.sh
```

2. Delete everything that is related to libmlx5 and libibverbs under `/usr/local` and `/usr/`.

```
cd /usr/local/
rm -rf  bin/ib*
rm -rf  lib/libibverbs.so
rm -rf  lib/libmlx5.so
       rm -rf  include/infiniband
       cd /usr
rm -rf  bin/ib*
rm -rf  lib/libibverbs.so
rm -rf  lib/libmlx5.so
       rm -rf  include/infiniband
```

3. Follow the installation steps.

- ***To clean installation or upgrade from any previous MLNX_DPDK GA version:***

1. Download MLNX_OFED 3.3-1.0.0.0.

   MLNX_OFED 3.3-1.0.0.0 can be downloaded from Mellanox site:

   http://www.mellanox.com/page/products_dyn?product_family=26&mtag=linux

2. Install MLNX_OFED 3.3-1.0.0.0.

```
./mlnxofedinstall
```

3. Verify that ConnectX-4 firmware is 12.16.1006 or 12.16.1010 and ConnectX-4 Lx firmware is 14.16.1006 or 14.16.1010.(**Note:** To receive firmware 12/14.16.1010, please contact Mellanox Support: support@mellanox.com)

```
ibv_devinfo
```

4. Set all the ports to Ethernet.

```
mst start

mlxconfig -d <device> set LINK_TYPE_P1/2=1/2/3
* LINK_TYPE_P1=<1|2|3> , 1=Infiniband 2=Ethernet 3=VPI(auto-sense)
```

   For example:

```
mlxconfig -d /dev/mst/mt4115_pciconf0 set LINK_TYPE_P1=2

mlxfwreset -d <device> reset
```

   For further instructions on how to run the script, please refer to the MLNX_OFED User Manual.

5. Restart MLNX_OFED.

```
/etc/init.d/openib restart
```

6. Extract the package MLNX_DPDK_2.2_2.7.tar.gz

   The default mlx5 configuration in `config/common_linuxapp` is the following:

```
# Compile burst-oriented Mellanox ConnectX-4 (MLX5) PMD
#
CONFIG_RTE_LIBRTE_MLX5_PMD=y
CONFIG_RTE_LIBRTE_MLX5_DEBUG=n
CONFIG_RTE_LIBRTE_MLX5_TX_MP_CACHE=8
```

7. Compile DPDK.

```
make install T=x86_64-native-linuxapp-gcc
```

For more advanced DPDK compilation options, please refer to dpdk.org documentation:

http://www.dpdk.org/doc/guides-2.2/

8. Configure huge pages according to the NUMA the card is connected to.

```
echo $PAGE_NUM >
/sys/devices/system/node/node$NUMA_NODE/hugepages/hugepages-
"$PAGE_SIZE"kB/nr_hugepages

mkdir -p /mnt/huge

HTLB_MOUNTED=$( mount | grep "type hugetlbfs" | wc -l)
if [ $HTLB_MOUNTED -eq 0 ]; then
        mount -t hugetlbfs hugetlb /mnt/huge
fi
```

2.2.2 ConnectX-4/ConnectX-4 Lx KVM

2.2.2.1 ConnectX-4/ConnectX-4 Lx KVM Hypervisor

1. Download MLNX_OFED 3.3-1.0.0.0

   MLNX_ MLNX_OFED 3.3-1.0.0.0 can be downloaded from the Mellanox site:

   http://www.mellanox.com/page/products_dyn?product_family=26&mtag=linux

2. Install MLNX_OFED 3.3-1.0.0.0.
3. Restart MLNX_OFED.

```
/etc/init.d/openib restart
```

4. Verify that ConnectX-4 firmware is 12.16.1006 or 12.16.1010 and ConnectX-4 Lx firmware is 14.16.1006 or 14.16.1010. (**Note:** To receive firmware 12/14.16.1010, please contact Mellanox Support: support@mellanox.com)

```
ibv_devinfo
```

5. Check if SR-IOV is enabled in the firmware.

```
mlxconfig -d /dev/mst/mt4113_pciconf0 q
Device #1: ----------
Device type: Connect4
PCI device: /dev/mst/mt4115_pciconf0
Configurations: Current
SRIOV_EN 1
NUM_OF_VFS 8
```

6. If needed, use mlxconfig to set the relevant fields:

```
mlxconfig -d /dev/mst/mt4113_pciconf0 set SRIOV_EN=1 NUM_OF_VFS=16
mlxfwreset -d <device> reset
```

7. Set all the ports to Ethernet.

```
mst star
mlxconfig -d <device> set LINK_TYPE_P1/2=1/2/3
* LINK_TYPE_P1=<1|2|3> , 1=Infiniband 2=Ethernet 3=VPI(auto-sense)
For example:
mlxconfig -d /dev/mst/mt4115_pciconf0 set LINK_TYPE_P1=2
```

```
mlxfwreset -d <device> reset
```

For further instructions on how to run the script, please refer to the MLNX_OFED User Manual.

8. Write to the sysfs file the number of Virtual Functions you need to create for the PF.

   The following is an example of a standard Linux kernel generated file that is available in the new kernels.

```
echo [num_vfs] > /sys/class/infiniband/mlx5_0/device/sriov_numvfs
```

   For further information, please refer to the MLNX_OFED User Manual section "Configuring SR-IOV for ConnectX-4/Connect-IB".

2.2.2.2 ConnectX-4/ConnectX-4 Lx KVM Virtual Machine

1. Download MLNX_OFED 3.3-1.0.0.0.

   MLNX_OFED 3.3-1.0.0.0 can be downloaded from the Mellanox site:

   http://www.mellanox.com/page/products_dyn?product_family=26&mtag=linux

2. Install MLNX_OFED 3.3-1.0.0.0 and enable the guest.

```
./mlnxofedinstall --guest
```

3. Extract the package MLNX_DPDK-2.2_1.7.tar.gz

   The default mlx5 configuration in `config/common_linuxapp` is the following:

```
#
# Compile burst-oriented Mellanox ConnectX-4 (MLX5) PMD
#
CONFIG_RTE_LIBRTE_MLX5_PMD=y
CONFIG_RTE_LIBRTE_MLX5_DEBUG=n
CONFIG_RTE_LIBRTE_MLX5_TX_MP_CACHE=8
```

4. Compile DPDK.

```
make install T=x86_64-native-linuxapp-gcc
```

   For more advanced DPDK compilation options, please refer to dpdk.org documentation:

   http://www.dpdk.org/doc/guides-2.2/

5. Configure huge pages.

```
echo $PAGE_NUM > /sys/devices/system/node/node0/hugepages/hugepages-
"$PAGE_SIZE"kB/nr_hugepages

mkdir -p /mnt/huge

HTLB_MOUNTED=$( mount | grep "type hugetlbfs" | wc -l)
if [ $HTLB_MOUNTED -eq 0 ]; then
        mount -t hugetlbfs hugetlb /mnt/huge
fi
```

2.2.3 ConnectX-4/ConnectX-4 Lx ESX6.0 UP1 / ESX 5.5 UP1

2.2.3.1 ConnectX-4/ConnectX-4 Lx ESX: VMWare

1. Download MLNX-NATIVE-ESX-ConnectX-4.

   To receive MLNX-NATIVE-ESX-ConnectX-4 driver, please contact Mellanox Support: support@mellanox.com

2. Install MLNX-NATIVE-ESX-ConnectX-4-4.15.5.10-10 on ESX 6.0 and MLNX-NATIVE-ESX-ConnectX-4-4.5.5.10-10 on ESX 5.5.

```
esxcli software vib install -d /FULL_PATH/<VERSION>.zip
```

3. Update the firmware on ESXi.
4. Download the latest firmware from the Mellanox site (12/14.16.1006).
(**Note:** To receive firmware 12/14.16.1010, please contact Mellanox Support: support@mellanox.com)

   http://www.mellanox.com/page/firmware_download

```
ConnectX-4:
/opt/mellanox/bin/mlxburn -d mt4115_pciconf0 -fw_dir /FULL_PATH/fw-4115-rel-12_16_1006/1010 for ConnectX-4
ConnectX-4 LX:
/opt/mellanox/bin/mlxburn -d mt4115_pciconf0 -fw_dir /FULL_PATH/fw-4117-rel-14_16_1006/1010
```

5. Verify that ConnectX-4 firmware is 12.16.1006 or 12.16.1010 and ConnectX-4 Lx firmware is 14.16.1006 or 14.16.1010

```
/opt/mellanox/bin/flint -d mt4115_pciconf0 q
```

6. Reboot the ESX machine.
7. Enable SR-IOV on the VMware.

```
esxcli system module parameters set -m nmlx5_core -p max_vfs=2
esxcfg-module -g nmlx5_core
```

8. Restart the driver.

```
/opt/mellanox/bin/openibd restart
```

2.2.3.2 ConnectX-4/ConnectX-4 Lx ESX: VM Virtual Machine

1. Install MLNX_OFED v3.3-1.0.0.0.

   MLNX_OFED 3.3-1.0.0.0 can be downloaded from Mellanox site:

   http://www.mellanox.com/page/products_dyn?product_family=26&mtag=linux

```
./mlnxofedinstall --guest
```

2. Extract the package MLNX_DPDK-2.2_2.7.tar.gz

   The default mlx5 configuration in `config/common_linuxapp` is the following:

```
#
# Compile burst-oriented Mellanox ConnectX-4 (MLX5) PMD
#
CONFIG_RTE_LIBRTE_MLX5_PMD=y
CONFIG_RTE_LIBRTE_MLX5_DEBUG=n
CONFIG_RTE_LIBRTE_MLX5_TX_MP_CACHE=8
```

3. Compile DPDK.

```
make install T=x86_64-native-linuxapp-gcc
```

   For more advanced DPDK compilation options, please refer to dpdk.org documentation: http://www.dpdk.org/doc/guides-2.2/

4. Configure huge pages.

```
echo $PAGE_NUM > /sys/devices/system/node/node0/hugepages/hugepages-"$PAGE_SIZE"kB/nr_hugepages

mkdir -p /mnt/huge
```

```
HTLB_MOUNTED=$( mount | grep "type hugetlbfs" | wc -l)
if [ $HTLB_MOUNTED -eq 0 ]; then
        mount -t hugetlbfs hugetlb /mnt/huge
fi
```

2.3 Configuring PMD Debug Mode

➢ ***To enable Debug mode:***

1. Modify the `config/common_linuxapp` file.
   - For ConnectX-3 PMD: `CONFIG_RTE_LIBRTE_MLX4_DEBUG=y`
   - For ConnectX-4 PMD: `CONFIG_RTE_LIBRTE_MLX5_DEBUG=y`
2. Compile DPDK.

```
rm –rf x86_64-native-linuxapp-gcc
```

2.4 ConnectX-4 Command Line Parameters

Parameter	Description
`txq_inline`	Amount of data to be inlined during TX operations. This improves latency. It can improve PPS performance of messages < 256B when PCI back pressure is detected and may be useful for scenarios involving heavy traffic on many queues. In this case, it is recommended to set it to 128 It is not enabled by default (set to 0) since the additional software logic necessary to handle this mode can lower performance when back pressure is not expected.
`txq_inline_new`	**[Beta]** An alternative way to inline the data during TX operations. It can improve PPS performance for messages >= 256B when PCI back pressure is detected and may be useful for scenarios involving heavy traffic on many queues. In this case, it is recommended to set it to 64 It is not enabled by default (set to 0) since the additional software logic necessary to handle this mode can lower performance when back pressure is not expected.
`txqs_min_inline`	Enable inline Send only when the number of TX queues is higher or equal to this value. This option should be used in combination with ``txq_inline`` or ``txq_inline_new`` above.
`txq_mpw_en`	A nonzero value that enables multi-packet Send. This feature allows the TX burst function to pack up to five packets in two descriptors in order to save PCI bandwidth. Moreover, it improves performance at the cost of a slightly higher CPU usage. It is enabled by default. To disable please set to 0 It is currently only supported on the ConnectX-4 Lx family of adapters. When multi-packet Send is enabled, there is no support for TX VLAN HW insertion. To use TX VLAN HW insertion please disable it.
`rxq_cqe_comp_en`	A nonzero value that enables optimized received packets' completion algorithm when PCI back pressure is detected. This improves performance at the cost of a slightly higher CPU usage. It is enabled by default. To disable this please set to 0

2.5 Sending and Receiving Jumbo Frames on ConnectX-3

If the `mbuf` size is smaller than the MTU size and you need to use scattered `mbuf`.

1. Modify the `config/common_linuxapp` file.
   - For ConnectX-3 PMD: `CONFIG_RTE_LIBRTE_MLX4_SGE_WR_N=4`
2. Compile DPDK.

```
rm –rf x86_64-native-linuxapp-gcc
make install T=x86_64-native-linuxapp-gcc
```

3. Set the appropriate MTU using the `rte_eth_dev_set_mtu` API.

2.6 Setting RX VLAN Filter on ConnectX-3

Make sure that regular steering mode is configured.

```
cat /sys/module/mlx4_core/parameters/log_num_mgm_entry_size -1
```

3. Modify the /etc/modprob.d/mlnx.conf file if required and restart MLNX_OFED.

   Configure VLAN interface on the port using standard Linux tools.

4. Add or remove VLAN using the `rte_eth_dev_vlan_filter()` DPDK API.

3 System Performance Configuration

3.1 General Settings

- Use the CPU near local NUMA node to which the PCIe adapter is connected, for better performance.

  For Virtual Machines (VM), verify that the right CPU and NUMA node are pinned for the VM according to the above. If possible, connect you NIC near NUMA node 0. Run `mst status -v` to identify the NUMA node to which the PCIe adapter is connected

```
mst status -v
MST modules:
------------
    MST PCI module loaded
    MST PCI configuration module loaded
PCI devices:
------------
DEVICE_TYPE             MST                           PCI       RDMA
NET                     NUMA
ConnectX3Pro(rev:0)     /dev/mst/mt4103_pciconf0
ConnectX3Pro(rev:0)     /dev/mst/mt4103_pci_cr0       04:00.0   mlx4_0
net-eth4,net-eth5
```

- If more than one adapter is used, verify that both adapters are located on the same PCI bus (as each CPU socket on a Crown Pass platform manages its own PCI bus) in order to forward packets from one to the other without NUMA performance penalty.
- **ConnectX-3**:
  - Verify the optimized steering mode is configured

```
cat /sys/module/mlx4_core/parameters/log_num_mgm_entry_size -7
```

  - If not, modify `/etc/modprobe.d/mlx4_core.conf` and restart MLNX_OFED
- **Check the Core Frequency**

  Check that the output CPU frequency for each core is equal to the maximum supported and that all core frequencies are consistent.
  - Check the maximum supported CPU frequency:

```
#cat /sys/devices/system/cpu/cpu*/cpufreq/cpuinfo_max_freq
```

  - Check that the core frequencies are consistent

```
#cat /proc/cpuinfo | grep "cpu MHz"
```

  - Check that the output frequencies are the same as the maximum supported
  - Check the current CPU frequency to check whether it is configured to max available frequency:

```
#cat /sys/devices/system/cpu/cpu*/cpufreq/cpuinfo_cur_freq
```

When the following CPU frequency modules are loaded, CPU scaling is enabled, and you can improve performance by setting the scaling mode to performance:

- **Set the scaling mode to performance for every CPU**

```
# echo performance >
/sys/devices/system/cpu/cpu<cpunumber>/cpufreq/scaling_governor
```

For further information, please refer to Performance Tuning Guide for Mellanox Adapters (http://www.mellanox.com/related-docs/prod_software/Performance_Tuning_Guide_for_Mellanox_Network_Adapters.pdf)

- **Verify that Max_Read_Req BIOS parameter is set to 4K**

To obtain the current setting for the Max_Read_Req BIOS parameter:

```
setpci -s <NIC BIOS address> 68.w

example:
setpci -s 21:00.0 68.w
5026
```

If the output is different than 5XXX, set it by:

```
setpci -s <NIC BIOS address> 68.w=5XXX
```

NOTE: The XXX can be different on different systems. Make sure to configure according to the setpci output..

For example:

```
setpci -s 84:00.0 68.w
2026
Run: setpci -s 84:00.0 68.w=5026
```

- **Disable pause frames on all network interfaces managed by mlx4_en/mlx5_en**

```
lldpad stop
```

```
ethtool -A eth16 rx off tx off
```

NOTE: In certain systems, pause frames are used to increase performance.

- **Use 1Gb huge pages**
- **Hyper threading**

In certain DPDK application, enabling hyper threading results in better performance. For benchmarking purposes, it is recommended to disable hyper threading

- **Make sure that unnecessary System Management Interrupts (SMIs) are disabled**

SMI that are used for Power Monitoring and for Memory PreFailure Notification are recommended to be disabled. Please refer to your server provider guides for recommended platform tuning.

- **Isolate used cores**

  Use `isolcpus` command for boot configuration.

  For example, add the following to kernel boot parameters:

```
isolcpus=2,3
```

- **Disable kernel memory compaction**

```
echo never > /sys/kernel/mm/transparent_hugepage/defrag
echo never > /sys/kernel/mm/transparent_hugepage/enabled
echo 0 > /sys/kernel/mm/transparent_hugepage/khugepaged/defrag
sysctl -w vm.swappiness=0
sysctl -w vm.zone_reclaim_mode=0
```

- **Interrupts configuration**

NOTE: Interrupts configuration should be performed only if the needed performance was not achieved**.**

  - Stop irqbalancer

```
service irqbalance stop
```

  - Set all possible interrupts to different NUMA:

```
Example: echo '6-9' | sudo tee /proc/irq/*/smp_affinity_list
```

  - Set NIC interrupts to same NUMA:

    Example:

```
set_irq_affinity_cpulist.sh 0-1 ethX
```

  - Set other NIC interrupts to different NUMA:

    Example:

```
set_irq_affinity_cpulist.sh 6-9 ethY
```

3.2 KVM Settings

Make sure that Hypervisor kernel is 3.16 or newer (For example Ubuntu 14.10 or Fedora 20/21 can be used).

5. Configure boot with "`iommu=pt`".
6. Use 1G huge pages.
7. Make sure to allocate a VM on huge pages

   Example:

   qemu is started with the following commands:

```
umount /mnt/huge 2> /dev/null
mount -t hugetlbfs none /mnt/huge &&
echo 8192 > /sys/kernel/mm/hugepages/hugepages-2048kB/nr_hugepages &&

numactl --cpunodebind 1 --membind 1 -- \
qemu-system-x86_64 \
-smp 24 \
-m 4G \
-mem-path /mnt/huge \
-mem-prealloc \
```

```
-enable-kvm \
-cpu host \
-serial tcp::999,server,nowait \
-nographic \
-vga none \
- -device pci-assign,host=83:00.1 \
-device pci-assign,host=84:00.1 \
-drive snapshot=on,file=/opt/vm/ubuntu-14.04-template.qcow2 \
-drive file=/data/data.img
```

Since both adapters are installed on NUMA node 1 PCI slots (CPU socket 1), `*numactl` is used to bind qemu to CPU threads and memory from that node only, which makes a Virtual Machine without NUMA internally.

All its memory is allocated from huge pages in /mnt/huge.

- After loading VM, verify huge pages on your Hypervisor is used by VM:

```
cat /sys/devices/system/node/node<NUM>/hugepages/hugepages-<PAGE-
SIZE>/free_hugepages
```

- Make sure to set CPU pining

  For example if you run qemu:

```
(qemu) info cpus
CPU #0: pc=0xffffffff81056306 (halted) thread_id=2719
CPU #1: pc=0xffffffff81056306 (halted) thread_id=2720
CPU #2: pc=0xffffffff81056306 (halted) thread_id=2723
CPU #3: pc=0xffffffff81056306 (halted) thread_id=2724
taskset -p 0x1 2719
taskset -p 0x2 2720
taskset -p 0x4 2723
taskset -p 0x8 2724
```

4 Running DPDK Application with Mellanox Poll-Mode Driver

4.1 ConnectX®-3 / ConnectX®-3 Pro

Since mlx4 PMD is compiled statically with MLNX_DPDK 2.1, no special requirements are needed to run the application with mlx4 PMD.

For example:

```
./testpmd -c 0xe000 -n 4  --socket-mem=0,2048 -- --port-numa-config=0,1,1,1
--socket-num=1  --burst=64 --txd=256 --rxd=256 --mbcache=512  --rxq=1 --
txq=1 --nb-cores=2 --i
```

Specific PCI address can be used to specify the NIC:

```
./testpmd -c 0xe000 -n 4 -w 0000:08:00.0 --socket-mem=0,2048 -- --port-numa-
config=0,1,1,1 --socket-num=1  --burst=64 --txd=256 --rxd=256 --mbcache=512
--rxq=1 --txq=1 --nb-cores=2 --i
```

NOTE: In ConnectX-3 NICs, a single PCI address represents 2 ports.

When running bi-directional traffic, for better performance, use the receive-inline feature that can be enabled by the env variable MLX4_INLINE_RECV_SIZE.

Example: for 64B messages

```
MLX4_INLINE_RECV_SIZE=64 ./testpmd -c 0xe000 -n 4  --socket-mem=0,2048 -- --
port-numa-config=0,1,1,1 --socket-num=1  --burst=64 --txd=256 --rxd=256 --
mbcache=512  --rxq=1 --txq=1 --nb-cores=2 --i
```

4.2 ConnectX®-4 / ConnectX®-4 Lx

Since mlx5 PMD is compiled statically with MLNX_DPDK 2.2, no special requirements are needed to run the application with mlx5 PMD.

For example:

Specific PCI address can be used to specify NIC's ports:

```
./testpmd -c 0x1ff -n 4 -w 0000:08:00.0,txq_inline=128
-w 0000:08:00.1,txq_inline=128  --socket-mem=2048,0 -- --port-numa-
config=0,0,1,0 --socket-num=0 --burst=64 --txd=1024 --rxd=256 --mbcache=512
--rxq=4 --txq=4 --nb-cores=8  --rss-udp -i
```

NOTE: In ConnectX-4 NICs, a PCI address represents each port.

NOTE: It is recommended to use **txq_inline=128** parameter for best performance for 64B and 128B messages with more multiple cores to achive max perfromance.

NOTE: It is recommended to use **txq_inline_new=64** parameter for best performance for messages >= 256B with multiple cores to achive max perfromance.

NOTE: `--rss-udp testpmd` option should be used to achieve best spreading of UDP flows.

Some ConnectX-4 Lx cards are single port cards. To run testpmd fwd test on one port:

```
./testpmd -c 0xe000 -n 4 -w 0000:08:00.0,txq_inline=200    --socket-
mem=2048,0 -- --port-numa-config=0,0,1,0 --socket-num=0 --burst=64 --
txd=4096 --rxd=1024 --mbcache=512 --rxq=4 --txq=4 --nb-cores=4 --rss-udp --i
```

NOTE: It is recommended to use **`txq_inline=200`** parameter for best performance for 64B messages with multiple cores to achive maximum perfromance

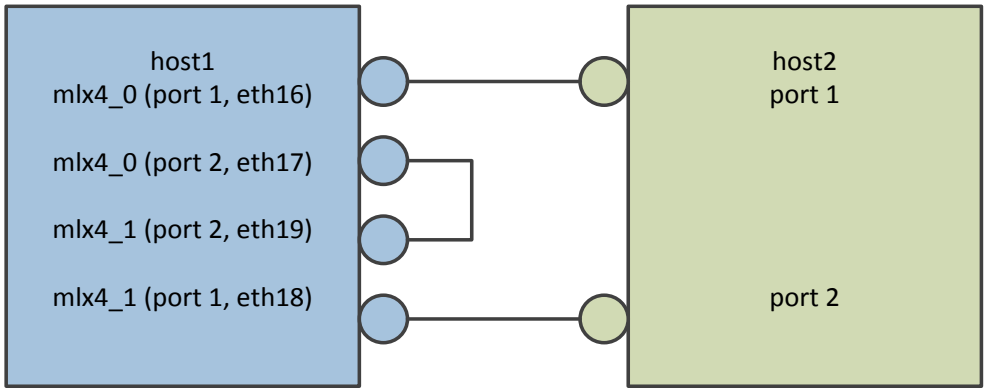
NOTE: `--rss-udp testpmd` option should be used to achieve best spreading of UDP flows.

5 Sanity Check

Provided that all software components have been successfully installed and at least one ConnectX® adapter is present in the host system, run testpmd to test PMD.

These examples assume that there is a host with two dual port adapters that:

- First port of each NIC is linked to another similar host
- Second port of each NIC is linked with each other



Run *testpmd* interactively from the DPDK build tree (for more information about its command-line options, please refer to its documentation: http://www.dpdk.org/doc/guides/testpmd_app_ug/index.html):

```
 root# ~/DPDK/x86_64-native-linuxapp-gcc/app/test-pmd/testpmd -c
0xf000f000 -n 4 -d -- -i
 EAL: Detected lcore 0 as core 0 on socket 0
 EAL: Detected lcore 1 as core 1 on socket 0
 EAL: Detected lcore 2 as core 2 on socket 0
 EAL: Detected lcore 3 as core 3 on socket 0
 EAL: Detected lcore 4 as core 4 on socket 0
 [...]
 EAL: Detected lcore 27 as core 3 on socket 1
 EAL: Detected lcore 28 as core 4 on socket 1
 EAL: Detected lcore 29 as core 5 on socket 1
 EAL: Detected lcore 30 as core 6 on socket 1
 EAL: Detected lcore 31 as core 7 on socket 1
 EAL: Support maximum 64 logical core(s) by configuration.
 EAL: Detected 32 lcore(s)
 EAL:   cannot open VFIO container, error 2 (No such file or directory)
 EAL: VFIO support could not be initialized
 EAL: Setting up memory...
 EAL: Ask a virtual area of 0x6400000 bytes
 EAL: Virtual area found at 0x7f15fd600000 (size = 0x6400000)
 EAL: Ask a virtual area of 0x200000 bytes
[...]
 EAL: PCI device 0000:83:00.0 on NUMA socket 1
 EAL:   probe driver: 15b3:1007 librte_pmd_mlx4
 PMD: librte_pmd_mlx4: PCI information matches, using device "mlx4_0"
(VF: false)
 PMD: librte_pmd_mlx4: 2 port(s) detected
 PMD: librte_pmd_mlx4: bad state for port 1: "down" (1)
 PMD: librte_pmd_mlx4: port 1 MAC address is 00:02:c9:b5:b7:50
 PMD: librte_pmd_mlx4: bad state for port 2: "down" (1)
 PMD: librte_pmd_mlx4: port 2 MAC address is 00:02:c9:b5:b7:51
 EAL: PCI device 0000:84:00.0 on NUMA socket 1
 EAL:   probe driver: 15b3:1007 librte_pmd_mlx4
```

```
 PMD: librte_pmd_mlx4: PCI information matches, using device "mlx4_1"
(VF: false)
 PMD: librte_pmd_mlx4: 2 port(s) detected
 PMD: librte_pmd_mlx4: bad state for port 1: "down" (1)
 PMD: librte_pmd_mlx4: port 1 MAC address is 00:02:c9:b5:ba:b0
 PMD: librte_pmd_mlx4: bad state for port 2: "down" (1)
 PMD: librte_pmd_mlx4: port 2 MAC address is 00:02:c9:b5:ba:b1
 Interactive-mode selected
 Configuring Port 0 (socket 0)
 PMD: librte_pmd_mlx4: 0x7f35e0: TX queues number update: 0 -> 1
 PMD: librte_pmd_mlx4: 0x7f35e0: RX queues number update: 0 -> 1
 Port 0: 00:02:C9:B5:B7:50
 Configuring Port 1 (socket 0)
 PMD: librte_pmd_mlx4: 0x7f3620: TX queues number update: 0 -> 1
 PMD: librte_pmd_mlx4: 0x7f3620: RX queues number update: 0 -> 1
 Port 1: 00:02:C9:B5:B7:51
 Configuring Port 2 (socket 0)
 PMD: librte_pmd_mlx4: 0x7f3660: TX queues number update: 0 -> 1
 PMD: librte_pmd_mlx4: 0x7f3660: RX queues number update: 0 -> 1
 Port 2: 00:02:C9:B5:BA:B0
 Configuring Port 3 (socket 0)
 PMD: librte_pmd_mlx4: 0x7f36a0: TX queues number update: 0 -> 1
 PMD: librte_pmd_mlx4: 0x7f36a0: RX queues number update: 0 -> 1
 Port 3: 00:02:C9:B5:BA:B1
 Checking link statuses...
 Port 0 Link Up - speed 10000 Mbps - full-duplex
 Port 1 Link Up - speed 40000 Mbps - full-duplex
 Port 2 Link Up - speed 10000 Mbps - full-duplex
 Port 3 Link Up - speed 40000 Mbps - full-duplex
 Done
 testpmd>
```

The following commands are typed from the *testpmd* interactive prompt.

Check port status:

```
testpmd> show port info all
******************** Infos for port 0  ********************
   MAC address: 00:02:C9:B5:B7:50
   Connect to socket: 0
   memory allocation on the socket: 0
   Link status: up
   Link speed: 10000 Mbps
   Link duplex: full-duplex
   Promiscuous mode: enabled
   Allmulticast mode: disabled
   Maximum number of MAC addresses: 128
   Maximum number of MAC addresses of hash filtering: 0
   VLAN offload:
     strip on
     filter on
     qinq(extend) off

   ******************** Infos for port 1  ********************
   MAC address: 00:02:C9:B5:B7:51
   Connect to socket: 0
   memory allocation on the socket: 0
   Link status: up
   Link speed: 40000 Mbps
   Link duplex: full-duplex
   Promiscuous mode: enabled
   Allmulticast mode: disabled
   Maximum number of MAC addresses: 128
   Maximum number of MAC addresses of hash filtering: 0
   VLAN offload:
     strip on
     filter on
```

```
    qinq(extend) off

  ******************** Infos for port 2  ********************
  MAC address: 00:02:C9:B5:BA:B0
  Connect to socket: 0
  memory allocation on the socket: 0
  Link status: up
  Link speed: 10000 Mbps
  Link duplex: full-duplex
  Promiscuous mode: enabled
  Allmulticast mode: disabled
  Maximum number of MAC addresses: 128
  Maximum number of MAC addresses of hash filtering: 0
  VLAN offload:
    strip on
    filter on
    qinq(extend) off

  ******************** Infos for port 3  ********************
  MAC address: 00:02:C9:B5:BA:B1
  Connect to socket: 0
  memory allocation on the socket: 0
  Link status: up
  Link speed: 40000 Mbps
  Link duplex: full-duplex
  Promiscuous mode: enabled
  Allmulticast mode: disabled
  Maximum number of MAC addresses: 128
  Maximum number of MAC addresses of hash filtering: 0
  VLAN offload:
    strip on
    filter on
    qinq(extend) off
  testpmd>
```

8. Start IO forwarding between ports 1 and 3. The *tx_first* argument tells
9. *testpmd* to send a single packet burst which will be always forwarded by both ports:

```
testpmd> set fwd io
  Set io packet forwarding mode
  testpmd> set portlist 1,3
  previous number of forwarding ports 4 - changed to number of
configured ports 2
  testpmd> start tx_first
    io packet forwarding - CRC stripping disabled - packets/burst=32
    nb forwarding cores=1 - nb forwarding ports=2
    RX queues=1 - RX desc=128 - RX free threshold=0
    RX threshold registers: pthresh=8 hthresh=8 wthresh=0
    TX queues=1 - TX desc=512 - TX free threshold=0
    TX threshold registers: pthresh=32 hthresh=0 wthresh=0
    TX RS bit threshold=0 - TXQ flags=0x0
  testpmd>
```

10. Display *testpmd* port statistics:

```
testpmd> show port stats all

    ######################## NIC statistics for port 0  ########################
    RX-packets: 0          RX-missed: 0          RX-bytes:  0
    RX-badcrc:  0          RX-badlen: 0          RX-errors: 0
    RX-nombuf:  0
    TX-packets: 0          TX-errors: 0          TX-bytes:  0
    ############################################################################

    ######################## NIC statistics for port 1  ########################
    RX-packets: 60800584   RX-missed: 0          RX-bytes:  3891239534
    RX-badcrc:  0          RX-badlen: 0          RX-errors: 0
    RX-nombuf:  0
```

```
  TX-packets: 61146609   TX-errors: 0          TX-bytes:  3913382976
  ############################################################################

  ######################## NIC statistics for port 2  ########################
  RX-packets: 0          RX-missed: 0          RX-bytes:  0
  RX-badcrc:  0          RX-badlen: 0          RX-errors: 0
  RX-nombuf:  0
  TX-packets: 0          TX-errors: 0          TX-bytes:  0
  ############################################################################

  ######################## NIC statistics for port 3  ########################
  RX-packets: 61146920   RX-missed: 0          RX-bytes:  3913402990
  RX-badcrc:  0          RX-badlen: 0          RX-errors: 0
  RX-nombuf:  0
  TX-packets: 60800953   TX-errors: 0          TX-bytes:  3891262080
  ############################################################################
testpmd>
```

11. Stop forwarding:

```
 testpmd> stop
Telling cores to stop...
Waiting for lcores to finish...

  ---------------------- Forward statistics for port 1  ----------------------
  RX-packets: 78238689       RX-dropped: 0             RX-total: 78238689
  TX-packets: 78681769       TX-dropped: 0             TX-total: 78681769
  ----------------------------------------------------------------------------

  ---------------------- Forward statistics for port 3  ----------------------
  RX-packets: 78681737       RX-dropped: 0             RX-total: 78681737
  TX-packets: 78238721       TX-dropped: 0             TX-total: 78238721
  ----------------------------------------------------------------------------

  ++++++++++++++++ Accumulated forward statistics for all ports++++++++++++++++
  RX-packets: 156920426      RX-dropped: 0             RX-total: 156920426
  TX-packets: 156920490      TX-dropped: 0             TX-total: 156920490
  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

Done.
testpmd>
```

12. Exit testpmd.

```
testpmd> quit
Stopping port 0...done
Stopping port 1...done
Stopping port 2...done
Stopping port 3...done
bye...
root#
```