Mellanox ConnectX®-3 and ConnectX®-3 Pro Poll-Mode Driver

Quick Start Guide

1.7-8_2.8.4

www.mellanox.com

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 PRODUCTO(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

Mellanox Technologies, Ltd.
Beit Mellanox
PO Box 586 Yokneam 20692
Israel
www.mellanox.com
Tel: +972 (0)74 723 7200
Fax: +972 (0)4 959 3245

Table of Contents

List of Tables

Document Revision History

Table 1: Document Revision History

Revision	Description

1.7-8_2.8.4	• Added the following sections:
	  • VMware vSphere 5.5
	  • ESX Hypervisor
	  • Virtual Machine
	• Updated the following sections:
	  • Bare Metal
	  • Hypervisor
	  • Compilation Script
	  • Configuration Script
	  • General Settings

2.7.4	• Added the following Performance Configuration sections:
	  • General Settings
	  • KVM Settings
	• Updated the following sections:
	  • Bare Metal
	  • Hypervisor

2.6.9	• Added section Sending and Receiving Jumbo Frames
	• Updated the following sections:
	  • Bare Metal
	  • Hypervisor
	  • Compilation Script
	  • Running DPDK Application with Mellanox ConnectX®-3 Poll-Mode Driver

2.6.8	• Added the following sections:
	  • Compilation Script
	  • Configuration Script
	• Updated the following sections:
	  • Mellanox ConnectX®-3 Poll-Mode Driver (PMD) Overview
	  • Bare Metal
	  • Hypervisor
	  • Running DPDK Application with Mellanox ConnectX®-3 Poll-Mode Driver
	  • Sanity Check
	• Removed the VMware section

2.5	• The Quick Start Guide was completely re-structured and re-written

2.0	• Updated the following sections:
	  • DPDK Poll-Mode Driver Configuration
	  • Sanity Check

Revision	Description

1.1	• Updated the following sections:
	  • DPDK Poll-Mode Driver (PMD) Overview
	  • Internal Compilation
	  • External Compilation
	  • Sanity Check
	• Added the following sections:
	  • Automatic Installation
	  • mlnx-ofa_kernel-2.0
	  • libibverbs-1.1.6mlnx1
	  • libmlx4-1.0.4mlnx1
	  • DPDK Poll-Mode Driver Configuration

1.0	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 OFED Driver for VMware vSphere 5.5 User Manual	Provides general information on the installation, configuration, management and maintenance of the software and hardware of VPI (InfiniBand, Ethernet) adapter cards on VMware vSphere 5.5.
Mellanox OFED Driver for VMware vSphere 5.5 Release Notes	Describes the new features and changes of the latest VMware vSphere 5.5 MLNX_OFED release.
Mellanox ConnectX-3 DPDK PMD Release Notes	Describes the new features and changes of the latest DPDK PMD

1 DPDK Poll-Mode Driver (PMD) Overview

librte_pmd_mlx4 is the DPDK Poll-Mode Driver (PMD) for Mellanox ConnectX®-3 / ConnectX®-3 Pro Ethernet adapters. This driver is based on libibverbs and currently supports:

- Scattering/gathering RX/TX packets
- Multiple RX (with RSS/RCA) and TX queues
- Multiple MAC addresses
- VLAN filtering
- Link state information
- Software counters/statistics
- Start/stop/close operations
- Multiple physical ports host adapter
- Hardware TX and RX checksum offloading
- Hardware VXLAN TX and RX checksum offloading
- DPDK 1.7 and DPDK 1.8 from dpdk.org <http://dpdk.org/>

2 DPDK Poll-Mode Driver Installation and Configuration

2.1 Bare Metal

1. Install MLNX_OFED v3.0-x.x.x.

   MLNX_OFED 3.0-x.x.x can be downloaded from Mellanox site:

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

2. Verify that ConnectX-3/ConnectX-3 Pro firmware is 2.34.5000 (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. Download DPDK 1.7.1 or DPDK 1.8.0 from http://dpdk.org/.

5. 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

6. Extract the PMD package mlx4_pmd_v1.7-8_2.8.4.tar.gz.

7. Run the compilation script.

   ```
   ./compile_mlx4_pmd.sh -s <RTE_SDK> -t <RTE_TARGET>
   ```

   For example:

   ```
   ./compile_mlx4_pmd.sh -s /var/soft/dpdk-1.7.1 -t x86_64-native-linuxapp-gcc
   ```

   See section 2.4 for all compilation script options.

8. Run the configuration script.

   ```
   ./configure_mlx4_pmd.sh -s #NUMA node
   ```

   See section 2.5 for all configuration script options.

   NOTE: If hugepage reconfiguration is needed after rebooting the machine, you can re-run the configuration script.

9. In case that VLAN filtering is used, run:

   ```
   ./configure_mlx4_pmd.sh -s #NUMA node -v
   ```

   NOTE: Please be aware, in this case there can be performance penalty

10. Restart MLNX_OFED.

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

2.2 KVM

2.2.1 Hypervisor

1. Install MLNX_OFED v3.0-x.x.x.

   MLNX_OFED 3.0-x.x.x can be downloaded from Mellanox site:

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

   Please follow MLNX_OFED User Manual instructions how to enable SR-IOV.
2. Verify that ConnectX-3/ConnectX-3 Pro firmware is v2.34.5000 (Use ibstat command).
3. 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, please set log_num_mgm_entry_size=-1

   NOTE: Please be aware, in this case there can be perfromance penalty in this case.
4. Restart MLNX_OFED.

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

2.2.2 Virtual Machine

1. Install MLNX_OFED v3.0-x.x.x.

   MLNX_OFED 3.0-x.x.x can be downloaded from Mellanox site:

   http://www.mellanox.com/page/products_dyn?product_family=26&mtag=linux
2. Download DPDK 1.7 or DPDK 1.8 from http://dpdk.org/.
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/linux_gsg/index.html
4. Extract the PMD package mlx4_pmd_v1.7-8_2.8.tar.gz
5. Run the compilation script. See section 2.4 for all compilation script options.

   ```
   ./compile_mlx4_pmd.sh  -s <RTE_SDK> -t <RTE_TARGET>
   ```

   For example:

   ```
   ./compile_mlx4_pmd.sh -s /var/soft/dpdk-1.7.1 -t x86_64-native-linuxapp-gcc
   ```
6. Run the configuration script. See section 2.5 for all configuration script options.

   ```
   ./configure_mlx4_pmd.sh -s 0
   ```

   NOTE: If hugepage reconfiguration is needed after rebooting the machine, you can re-run the configuration script.

2.3 VMware vSphere 5.5

2.3.1 ESX Hypervisor

1. Install Mellanox OFED Driver for VMware vSphere 5.5 version v2.3.1.

   Please contact Mellanox Support to receive this driver.

   For instructions on how to install MLNX_OFED, please refer to section 2.2 in the Mellanox OFED Driver for VMware vSphere 5.5 Use Manual.

2. Install ConnectX-3 firmware v2.33.5000.

   For instructions on how to install firmware, please refer to section 2.5 (Firmware Programming) in the Mellanox OFED Driver for VMware vSphere 5.5 User Manual.

3. Verify that the installed firmware is configured as follow:

```
opt/mellanox/bin/mlxconfig -d /dev/mt4099_pci_cr0 q

Device #1:
----------

Device type:    ConnectX3
PCI device:     /dev/mt4099_pci_cr0

Configurations:          Current
         SRIOV_EN        1
         NUM_OF_VFS      16
         LINK_TYPE_P1    3
         LINK_TYPE_P2    3
LOG_BAR_SIZE    3
```

   For further information, please refer to section “Setting UP SR-IOV” (step 4) in Mellanox OFED Driver for VMware vSphere 5.5 User Manual.

4. Configure the required number of VFs for each port.

```
esxcli system module parameters set -m mlx4_core -p 'num_vfs=<VFs over
Port1, VFs over port2, 0> port_type_array=2'
```

   Example:

```
esxcli system module parameters set -m mlx4_core -p "num_vfs=2,2,0
port_type_array=2"
```

   For further information, please refer to section “Setting UP SR-IOV” (step 5) in Mellanox OFED Driver for VMware vSphere 5.5 User Manual.

5. Configure Flow Steering.

```
esxcli system module parameters set -m mlx4_core –p
“log_num_mgm_entry_size=-7”
```

   Note: If VLAN filtering is required, run::

```
esxcli system module parameters set -m mlx4_core –p
“log_num_mgm_entry_size=-1”
```

6. Check what module parameters are used.

```
esxcfg-module mlx4_core -g
```

7. Restart the driver.

```
#> /opt/mellanox/bin/openibd.sh restart
```

8. Assign a Virtual Function to a Virtual Machine in the vSphere Web Client.

For further information, please refer to section “Assigning a Virtual Function to a Virtual Machine in the vSphere Web Client” in Mellanox OFED Driver for VMware vSphere 5.5 User Manual.

9. Configure the Passthrough Device for a Virtual Function in the vSphere Web Client.

   For further information, please refer to section “Configuring the Passthrough Device for a Virtual Function in the vSphere Web Client” in Mellanox OFED Driver for VMware vSphere 5.5 User Manual.

2.3.2 Virtual Machine

1. Install MLNX_OFED.v2.4-x.x.x.

   MLNX_OFED v2.4-x.x.x can be downloaded from Mellanox site:

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

2. Download DPDK 1.7 or DPDK 1.8 from http://dpdk.org/.
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/linux_gsg/index.html

4. Extract the PMD package mlx4_pmd_v1.7-8_2.8.tar.gz
5. Run the compilation script. See section 2.4 for all compilation script options.

   ```
   ./compile_mlx4_pmd.sh  -s <RTE_SDK> -t <RTE_TARGET>
   ```

   For example:

   ```
   ./compile_mlx4_pmd.sh -s /var/soft/dpdk-1.7.1 -t x86_64-native-linuxapp-gcc
   ```

6. Run the configuration script. See section 2.5 for all configuration script options.

   ```
   ./configure_mlx4_pmd.sh –s 0
   ```

NOTE: If hugepage reconfiguration is needed after rebooting the machine, you can re-run the configuration script.

2.4 Compilation Script

```
./compile_mlx4_pmd.sh  -s <RTE_SDK>  -t <RTE_TARGET>  [ -r (enable receive inline) ] [ -c (enable HW checksum offloading) ] [ -m (enable multi-segment send / recv messages) ]  [ -d (debug) ]
```

Parameter	Description
-s <RTE_SDK>	This parameter is mandatory, need to specify full path to DPDK compiled sources
-t <RTE_TARGET>	This parameter is mandatory and needs to be specified according to the RTE_TARGET that DPDK was compiled with
-r	Optional parameter which enables inline receive.
-d	Optional parameter which enables PMD debug prints
-m	When compiled with this option, multi-segment send and receive can be used
-c	Enable hardware checksum offloading **Note:** Hardware checksum offloading is supported only in ConnectX-3 Pro. **Note:** Hardware checksum offloading is currently not supported in VMware VM.

NOTE: Multi-segment send/recv is not required. Please do not use this flag as it reduces PMD performance.

NOTE: Please do not use "–c" flag (hardware checksum offloading) if it is not required as it can reduce PMD performaces.

2.5 Configuration Script

```
./configure_mlx4_pmd.sh -s <numa node #> [-p <PAGE_SIZE>] [-n <HUGE PAGES NUM>] [-v <enable vlan filtering>]
```

Parameter	Description
-s <NUMA node #>	[Mandatory]: Please specify the NUMA node that will be used. Make sure that this NUMA node is connected to the NIC to be used.
-p <PAGE SIZE>	[Optional]: If not specified, the default page size is 2K
-n <HUGE PAGES NUM>	[Optional]: If not specified, the default number of pages that will be configured is 2048
-v	[Optional]: Should be used in case VLAN filtering is needed. If not specified optimized steering mode will be configured

2.6 Sending and Receiving Jumbo Frames

- DPDK 1.6:
  a. Configure the ConnectX-3/ ConnectX-3 Pro MTU.

     Example: `ifconfig eth0 MTU 4160`

  b. Compile PMD with the `"-m"` option, if the mbuf size is smaller than the MTU size and you need to use scattered mbuf.
- DPDK 1.7 & DPDK 1.8:

  Use the `rte_eth_dev_set_mtu` API to set the appropriate MTU.

2.7 RX VLAN Filter

1. Make sure that regular steering mode is configured (on Bare Metal and KVM).

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

2. Modify the /etc/modprob.d/mlnx.conf file if required

   or run

```
./configure_mlx4_pmd.sh with –v parameter
```

3. Configure VLAN interface on the port using standard Linux tools.
4. Use rte_eth_dev_vlan_filter() DPDK API to add or remove VLAN.

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.
- Verify the optimized steering mode is configured

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

  If not please run the `./configure_mlx4_pmd.sh` script

- **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
```

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**

```
lldpad stop
```

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

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

- Using 1Gb huge pages results in better performance.
- **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)
- Configure boot with "`iommu=pt`"
- Use 1G huge pages
- Make sure to allocate 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 ConnectX®-3/ ConnectX®-3 Pro Poll-Mode Driver

The output of the PMD compilation is a shared object, `librte_pmd_mlx4.so`.

It is created under the

`mlx4_pmd_v1.7-8_2.8.4/librte_pmd_mlx4/librte_pmd_mlx4.so`

The `librte_pmd_mlx4.so` should be dynamically loaded during the DPDK application run with the `–d` flag.

For example:

```
app/testpmd -c 0xe000 -n 4 -b 0000:01:00.0 -b 0000:01:00.1 -d /tmp/
mlx4_pmd_v1.7-8_2.8.4/librte_pmd_mlx4/librte_pmd_mlx4.so  -- --i --numa --
burst=64 --txd=256 --rxd=256 --mbcache=256 --coremask=0xc000 --rxq=1 --txq=1
--portmask 0xA
```

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

Make sure you compile PMD with –r parameter, see section Compilation Script.

Example: for 64B messages

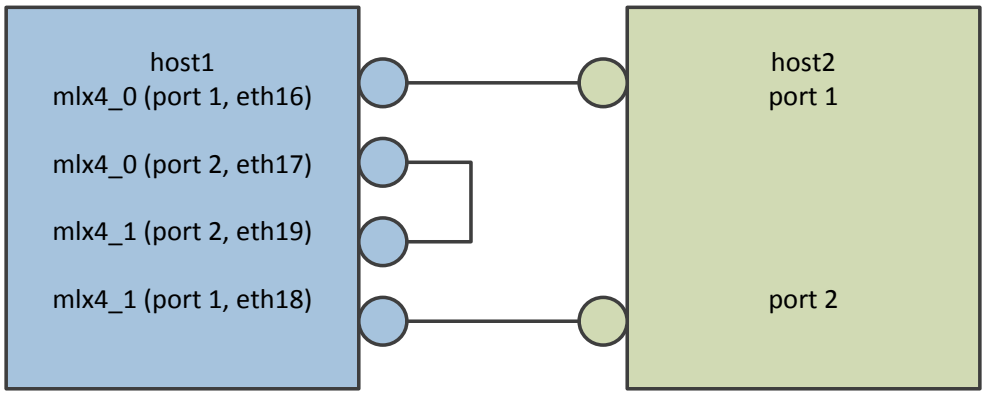
```
MLX4_INLINE_RECV_SIZE=64 app/testpmd -c 0xe000 -n 4 -b 0000:01:00.0 -b
0000:01:00.1 -d /tmp/mlx4_pmd_v2.7.4/librte_pmd_mlx4/librte_pmd_mlx4.so  --
--i --numa --burst=64 --txd=256 --rxd=256 --mbcache=256 --coremask=0xc000 --
rxq=1 --txq=1 --portmask 0xA
```

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



1. 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 ./librte_pmd_mlx4.so -- -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.

1. 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>
```

2. Start IO forwarding between ports 1 and 3. The *tx_first* argument tells
3. *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>
```

4. 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>
```

5. 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>
```

6. Exit testpmd.

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