Western Digital.

Configuration Guide

Lossless Networking Guide for Intel® RNIC with Arista 7050CX3 Switch on OpenFlex™ Data24 NVMe-oF Storage Platform

Abstract

This configuration guide provides an overview of how to configure lossless Ethernet settings on Intel based Ethernet network adapter with Arista 7050CX3 switch on Western Digital® OpenFlex Data24 NVMe-oF platform..

April 2025

Table of Contents

Introduction

NVMe-oF™ based storage offers the promise of low latency shared storage. To obtain the performance potential of this technology, Ethernet Network Adapters in initiator hosts must be configured for lossless networking using standard Data Center Bridging (DCB) technologies.

The lossless settings are applied to the host with an Intel E810 ethernet network adapter and an Arista 7050CX3 switch to access storage from OpenFlex Data24 NVMe-oF storage platform.

Hardware and Software Specifications

Terminal output shown in this guide may vary based on your product and firmware version.

Hardware and Software Specifications

	Variable
Storage product	OpenFlex Data24 NVMe-oF Storage Platform
Storage interface	Six QSFP28 (100Gbps) fabric ports
Number of NVMe Disks and Capacity	24 x 15.36 TB SN840 Disks
Switch Name and Model	Arista DCS-7050CX3-32S-R
Switch Hardware and Software image version	HW: 11.01 / SW: 4.27.0F
Host OS	Debian GNU/Linux 11 (bullseye)
Host Kernel	5.10.0-20-amd64
Host CPU	Intel(R) Xeon(R) Platinum 8360Y CPU @ 2.40GHz
Host CPU core details	Dual socket server with 36 core CPU each. 144 logical cores in total with HT enabled
Host Memory	256GB
Host NIC	1 x Intel(R) Ethernet Network Adapter - E810-2CQDA2 (100Gbps)
Host NIC firmware version	4.30 0x8001af25 1.3429.0
Host NIC Driver version	Package 28.2.1, ice 1.12.7, irdma 1.12.55, rdma-core 44.0
Number of volumes connected to host	24

Lossless Configuration Process Summary

The process of enabling lossless networking functionality on Intel® based Ethernet Network Adapters can be broken down into the following steps:

1. PCIe Bifurcation of Intel RNIC in BIOS
2. Download and Install Software Drivers for Intel RNIC
3. Configure RoCEv2 with Global Pause
4. Configure RoCEv2 with Priority Flow Control (PFC) & Explicit Congestion Notification (DQCN)
5. Configure Boot Time Scripts

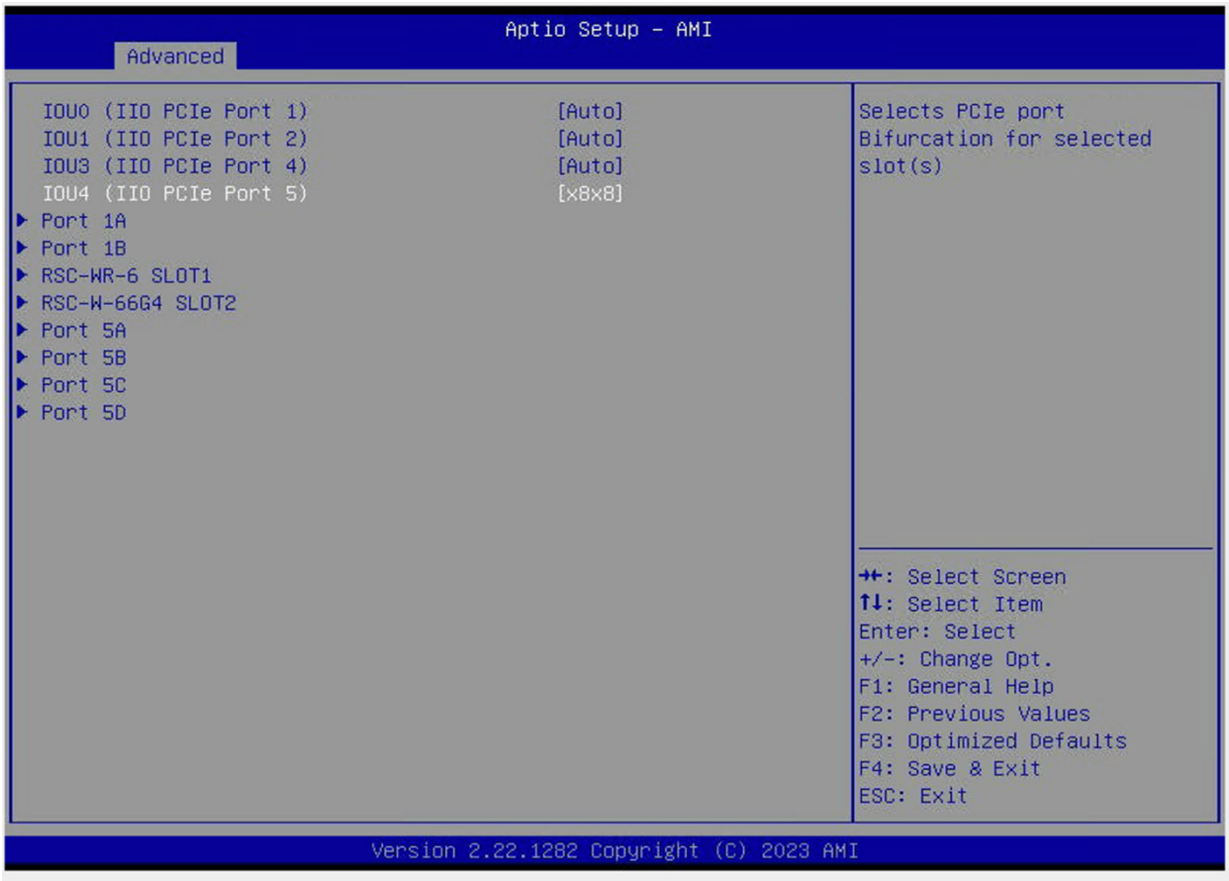
Lossless Configuration for Intel RNIC

PCIe Bifurcation of Intel RNIC in BIOS

After connecting the Intel Network card in the system, only one of the ports appears to be functional. It is important to enable bifurcation settings in BIOS and ensure the card is working with PCIe x16.

Please find example configuration on Supermicro system to enable PCIe Bifurcation settings on Intel Ethernet Adapter card which is connected on IOU4.

```
$ lspci | grep -i ethernet | grep -i e810
4b:00.0 Ethernet controller: Intel Corporation Ethernet Controller E810-C for QSFP (rev 02)
```



Install/Update NVMUpdatePackage

1. First verify that the system has an Intel adapter installed. After the host has boot into the operating system (Linux®), run the following command to verify the Intel Network Adapter shows up on the PCIe® bus.

```
$ lspci | grep -i Intel | grep -i ethernet
```

2. Visit Intel's website and use the "Search" dialog to locate "Ethernet Adapter Complete Driver Pack".

3. Select "Intel® Ethernet Adapter Complete Driver Pack" in the output, select "Download" and accept the EULA.

4. Copy the downloaded package file to the host.

5. Install the pre-requisite packages.

```
$ apt install linux-headers-5.10.0-20-amd64 ethtool
```

6. Create directory to extract package file into

```
$ mkdir ~/Release_28.2.1
$ cd ~/Release_28.2.1
```

7. Extract the package "zip" file on the host server

```
$ unzip ~/Release_28.2.1.zip
```

8. Change directory into the firmware utility folder

```
$ cd Release _ 28.2.1/NVMUpdatePackage/E810/
```

9. Extract the firmware package for the E810 series adapter.

```
$ tar -zxf E810 _ NVMUpdatePackage _ v4 _ 30 _ Linux.tar.gz
```

10. Change directory into the E810 firmware package

```
$ cd E810/Linux _ x64/
```

11. Update NVM version using NVMupdate script. Use a script to perform an inventory of all the Intel Ethernet devices in the system or update the Intel Ethernet devices in the system.

Update script example:

```
$ ./nvmupdate64e -u -l -o results.xml -b -c nvmupdate.cfg
```

This causes the NVMUpdate utility to check the installed Intel Ethernet devices against those contained in nvmupdate.cfg. If a device contains an NVM version older than that specified in the config file, the utility will update the device's NVM. It will create an xml file containing the results of the update. Note that -b is optional. Specifying -b will create a backup of the current NVM image(s). This may add about 30% to the tools execution time.

Verify the firmware was installed correctly on the host server using the following command.

Example Output:

```
$ ethtool -i ens2
driver: ice
version: 1.12.7
firmware-version: 4.30 0x8001af25 1.3429.0
expansion-rom-version:
bus-info: 0000:4b:00.0
supports-statistics: yes
supports-test: yes
supports-eeprom-access: yes
supports-register-dump: yes
supports-priv-flags: yes
```

Note: Refer the Release_28.2.1/NVMUpdatePackage/E810/E810/Linux_x64/readme.txt for more details.

Install Intel Ethernet Adapter (Ice) Base Network Driver

Install ice-1.12.7.tar.gz Linux* Base Drivers for Intel(R) Ethernet Network Connections from complete driver pack.

The ice driver requires the Dynamic Device Personalization (DDP) package file to enable advanced features (such as dynamic tunneling, Intel(R) Ethernet Flow Director, RSS, and ADQ, or others). The driver installation process installs the default DDP package file and creates a soft link ice.pkg to the physical package ice-x.x.x.x.pkg in the firmware root directory (typically /lib/firmware/ or /lib/firmware/updates/). The driver install process also puts both the driver module and the DDP file in the initramfs/initrd image.

Note: When the driver loads, it looks for intel/ice/ddp/ice.pkg in the firmware root. If this file exists, the driver will download it into the device. If not, the driver will go into Safe Mode where it will use the configuration contained in the device's NVM. This is NOT a supported configuration and many advanced features will not be functional. See "Dynamic Device Personalization" later for more information.

1. Install prerequisite packages for package build in system.

```
$ apt install build-essential cmake gcc libudev-dev libnl-3-dev libnl-route-3-dev ninja-build pkg-config
valgrind python3-dev cython3 python3-docutils pandoc make
$ apt install libsystemd-dev
```

2. Go to the directory where ice driver present and untar the archive.

```
$ cd Release _ 28.2.1/PROCGB/Linux/
$ tar -zxf ice-1.12.7.tar.gz
```

3. Go to the ice driver src directory:

```
$ cd ice-1.12.7/src/
```

4. Compile the driver module:

```
$ make install
```

Note: The binary will be installed as: /lib/modules/5.10.0-20-amd64/updates/drivers/net/ethernet/intel/ice/ice.ko. The install location listed above is the default location. This may differ for various Linux distributions. You may see warnings from depmod related to unknown RDMA symbols during the make of the OOT base driver. These warnings are normal and appear because the in-tree RDMA driver will not work with the OOT base driver. To address the issue, you need to install the latest OOT versions of the base and RDMA drivers.

5. Load the module using the modprobe command. To check the version of the driver and then load it:

```
$ modinfo ice
$ modprobe ice
```

Note: Ensure that any older ice drivers are removed from the kernel before loading the new module:

```
$ rmmod ice; modprobe ice
```

Note: To enable verbose debug messages in the kernel log, use the dynamic debug feature (dyndbg). See “Dynamic Debug” later in this README for more information.

6. Assign an IP address to the interface by entering the following, where <ethX> is the interface name that was shown in dmesg after modprobe:

```
$ ip addr add 192.168.0.60/21 dev enp77s0
```

7. Verify that the interface works. Enter the following, where IP_address is the IP address for another machine on the same subnet as the interface that is being tested:

```
$ ping 192.168.0.60
```

Note: Refer the Release_28.2.1/PROCGB/Linux/ice-1.12.7/README for more details on ice driver.

Install irdma Driver

To manually build and install the irdma driver and supporting rdma-core libraries:

1. Decompress the irdma driver archive:

```
$ cd Release _ 28.2.1/RDMA/Linux/
$ tar -zxf irdma-1.12.55.tgz
```

2. Build and install the RDMA driver:

```
$ cd irdma-1.12.55/
$ ./build.sh
```

Note: By default, the irdma driver is built using in-distro RDMA libraries and modules. Optionally, irdma may also be built using OFED modules. Please refer README_irdma.txt for more details.

3. Load the driver:

```
$ modprobe irdma
```

Notes: This modprobe step is required only during installation. Normally, irdma is autoloaded via a udev rule when ice or i40e is loaded: /usr/lib/udev/rules.d/90-rdma-hw-modules.rules.

4. Uninstall any previous versions of rdma-core user-space libraries. In Ubuntu system run below commands to remove older version of rdma-core if installed.

```
$ apt remove rdma-core
$ apt purge rdma-core
```

5. Patch, build, and install rdma-core user space libraries. On Ubuntu system, to create Debian packages from rdma-core:

- Install following packages before installing rdma-core.

```
$ apt install libsystemd-dev dh-python dh-make
```

- Download rdma-core-44.0.tar.gz from GitHub

```
$ wget https://github.com/linux-rdma/rdma-core/releases/download/v44.0/rdma-core-44.0.tar.gz
```

- Apply patch libirdma-44.0.patch to rdma-core

```
$ tar -xzvf rdma-core-44.0.tar.gz
$ cd rdma-core-44.0
$ patch -p2 < ~/Release_28.2.1/RDMA/Linux/irdma-1.12.55/libirdma-44.0.patch
```

- Change the below change in file rdma-core-44.0/debian/ibverbs-providers.install.

Change the line from **usr/lib/*/libmana.so*** to **usr/lib/*/libmana.so.***

- Build rdma-core

```
$ dh clean --with python3,systemd --builddirectory=build-deb
$ dh build --with systemd --builddirectory=build-deb
$ dh binary --with systemd --builddirectory=build-deb
```

- This creates .deb packages in the parent directory. Install the .deb packages using below command.

```
$ sudo dpkg -i ../*.deb
```

6. Add the following to /etc/security/limits.conf:

```
* soft memlock unlimited
* hard memlock unlimited
* soft nofile 1048000
* hard nofile 1048000
```

7. In addition, the files /etc/systemd/user.conf and /etc/systemd/system.conf may need to have the following line:

```
$ DefaultLimitMEMLOCK=1073741824
```

Note: This will change the Max locked memory for all process to 1G.

8. Restart the active session so new values will take effect. This avoids any limits on user mode applications as far as pinned memory and number of open files used.

9. The Intel Ethernet 800 Series supports both iWARP and RoCEv2 transports. By default, the irdma driver is loaded in iWARP mode. RoCEv2 may be selected globally (for all ports) using the module parameter “roce_ena=1”.

Note: To automatically enable RoCEv2 mode for all ports when the irdma driver is loaded, add the following line to /etc/modprobe.d/irdma.conf:

```
$ options irdma roce_ena=1
```

If the irdma driver is currently loaded, first unload it:

```
$ rmmod irdma
```

Reload the driver with appropriate roce_ena value:

```
$ modprobe irdma roce_ena=1
```

10. Updated irdma config file

```
$ cat /etc/modprobe.d/irdma.conf
blacklist i40iw
alias i40iw irdma
options irdma roce_ena=1 dcqcn_enable=1
```

11. Reboot the host if any RDMA applications were running during the rdma-core reinstall.

12. After successful installation, RDMA devices are listed in the output of “ibv_devices”. Install ibverbs-utils for utils commands.

```
$ apt install ibverbs-utils
$ ibv _ devices
device                 node GUID
------              ----------------
rocep75s0           42a6b7fffe7b4730
rocep77s0           42a6b7fffe7b4734
```

13. Verify the new modules versions after system is rebooted.

```
$ cat /sys/module/irdma/version
$ cat /sys/module/ice/version
```

Example Output:

```
$ cat /sys/module/irdma/version
1.12.55
$ cat /sys/module/ice/version
1.12.7
```

Note: Please refer Release_28.2.1/RDMA/Linux/irdma-1.12.55/README_irdma.txt for more configurable parameters.

Configure RoCEv2 with Global Pause

Lossless settings on OpenFlex Data24

1. Flow control GET command

```
$ curl -i -u admin:admin -X GET http://<IP>/Storage/Devices/<Storage-Device-ID>/Ports/<port-ID>/
```

2. Flow control SET command

```
$ curl -i -u admin:admin -H “Content-Type: application/json” -H “If-Match:<e-tag>” -X PUT --data
‘{“FlowControl”:{“ID”:1,”Name”:”Global Pause”}}’ http://<IP>/Storage/Devices/<Storage-Device -ID>/Ports/<port-
ID>/
```

Configuration on Arista 7050CX3 Switch Ports

All the below commands need to be executed on all the switch ports connecting Host RNIC and OpenFlex Data24.

```
$ interface ethernet 14/1
$ switchport mode access
$ no lldp transmit
$ no lldp receive
$ no priority-flow-control
$ flowcontrol receive on
$ flowcontrol send on
$ show interface ethernet 14/1 flow-control
```

Configuration on Host

The following is a list of things to know before following this process:

- All of these commands must be executed for each port on the RDMA Network Adapter that needs to be configured in host.
- The majority of these commands are not persistent through reboot and will have to be scripted to run at every boot.

1. Load the below modules in order.

```
$ modprobe -v configfs
$ modprobe -v nvme-core multipath=yes
$ modprobe -v nvme
$ modprobe -v nvme-rdma
$ modprobe -v irdma
$ rdma link
$ ulimit -n 4096
$ ulimit -u 51296400
```

2. Enable link level flow control on both directions.

```
$ ethtool -A ens2 rx on tx on
$ ethtool -a ens2
```

3. Modify the default ring buffer settings.

```
$ ethtool -G ens2 rx 2048 tx 2048
$ ethtool -g ens2
```

Example command outputs for lossless config on one port.

```
$ ethtool -A ens2 rx on tx on
$ ethtool -a ens2
```

Pause parameters for ens2:

```
Autonegotiate:  on
RX:             on
TX:             on
$ ethtool -G ens2 rx 2048 tx 2048
$ ethtool -g ens2
Ring parameters for ens2:
Pre-set maximums:
RX:             8160
RX Mini: n/a
RX Jumbo:       n/a
TX:             8160
Current hardware settings:
RX:             2048
RX Mini: n/a
RX Jumbo:       n/a
TX:             2048
```

Configure RoCEv2 with Priority Flow Control & Explicit Congestion Notification

Lossless settings on OpenFlex Data24

1. Flow control GET command

```
curl -i -u admin:admin -X GET http://<IP>/Storage/Devices/<Storage-Device- ID>/Ports/<port-ID>/
```

2. Flow control SET command

```
curl -i -u admin:admin -H "Content-Type: application/json" -H "If-Match:<e-tag>" -X PUT --data
'{"FlowControl":{"ID":2,"Name":"PFC"}, "PFCLevel":3}' http://<IP>/Storage/Devices/<Storage-Device-ID>/
Ports/<port-ID>/
```

Lossless settings on Arista 7050CX3 Switch

All the below commands need to be executed on all the switch ports connecting Host RNIC and OpenFlex Data24.

```
$ interface Ethernet14/1
   mtu 5000
   speed forced 100gfull
   switchport access vlan 30
   no lldp transmit
   no lldp receive
   qos trust dscp
   priority-flow-control on
   priority-flow-control priority 3 no-drop
   !
   tx-queue 3
      random-detect ecn minimum-threshold 1150 segments maximum-threshold 1500 segments max-mark-probability 100 weight 0
```

Configurations on Host

The following is a list of things to know before following this process:

- All of these commands must be executed for each port on the RDMA Network Adapter that needs to be configured.
- "lldptool" is the primary tool used to configure the Intel RNICs.
- The majority of these commands are not persistent through reboot and will have to be scripted to run at every boot.

1. Install lldptool which is used for lossless configuration.

```
$ apt install lldptool
```

2. Start lldp as daemon if it is not started already

```
$ lldpad -d
```

3. Load the below modules in order.

```
$ modprobe -v configfs
$ modprobe -v nvme-core multipath=yes
$ modprobe -v nvme
$ modprobe -v nvme-rdma
$ modprobe -v irdma
$ rdma link
$ ulimit -n 4096
$ ulimit -u 51296400
```

4. Use ethtool to disable default link flow control and firmware DCB mode.

```
$ ethtool -A ens2 rx off tx off
$ ethtool --set-priv-flags ens2 fw-lldp-agent off
```

5. Enable PFC for priority level 3.

```
$ lldptool -Ti ens2 -V PFC willing=no enabled=3
```

6. Enable DSCP of 24 (Priority 3) on TC3

```
$ lldptool -Ti ens2 -V APP app=3,5,24
```

7. Add DSCP mapping of 48 on TC6

```
$ lldptool -Ti ens2 -V APP app=6,5,48
```

8. Enable ETS to map DSCP priority/TC3 and the ECN to priority/TC6, and to allocate 80% bandwidth to TC3 and 20% bandwidth to TC6.

```
$ lldptool -Ti ens2 -V ETS-CFG willing=no tsa=0:ets,1:ets,2:ets,3:strict,4:ets,5:ets,6:strict,7:ets
tcbw=0,0,0,80,0,0,20,0 willing=no
```

9. Verify the new settings.

```
$ lldptool -ti ens2
```

9. Verify the new settings.

```
$ lldptool -ti ens2
```

10. Set the default_roce_tos. DSCP is set to 24 (0x18) and priority 3 so default_roce_tos is 96 (4 x DSCP value).

```
$ mkdir /sys/kernel/config/rdma_cm/rocep75s0
$ echo 96 > /sys/kernel/config/rdma_cm/rocep75s0/ports/1/default_roce_tos
```

11. Configure ECN and add DSCP mapping for ECN. Here ECN has a default value of 48.

```
$ mkdir /sys/kernel/config/irdma/rocep75s0
$ cd /sys/kernel/config/irdma/rocep75s0/
$ echo 0x706050418020100 > up_up_map
$ echo 1 > roce_dcqcn_enable
$ echo 48 > cnp_up_override
$ echo 1 > up_map_enable
```

Example command outputs for lossless config on one port.

```
$ ethtool -A ens2 rx off tx off
$ ethtool --set-priv-flags ens2 fw-lldp-agent off
$ lldptool -Ti ens2 -V PFC willing=no enabled=3
willing = no
prio = 3
$ lldptool -Ti ens2 -V APP app=3,5,24
0:(3,5,24) local hw (set)
$ lldptool -Ti ens2 -V APP app=6,5,48
0:(3,5,24) local hw (set)
1:(6,5,48) local hw (set)
$ lldptool -Ti ens2 -V ETS-CFG willing=no tsa=0:ets,1:ets,2:ets,3:strict,4:ets,5:ets,6:strict,7:ets
tcbw=0,0,0,80,0,0,20,0
willing = no
TSA = 0:ets 1:ets 2:ets 3:strict 4:ets 5:ets 6:strict 7:ets
tcbw = 0% 0% 0% 80% 0% 0% 20% 0%
$ lldptool -ti ens2
Chassis ID TLV
MAC: 40:a6:b7:7b:47:30
Port ID TLV
MAC: 40:a6:b7:7b:47:30
Time to Live TLV
120
IEEE 8021QAZ ETS Configuration TLV
Willing: no
CBS: not supported
MAX_TCS: 8
PRIO_MAP: 0:0 1:0 2:0 3:0 4:0 5:0 6:0 7:0
TC Bandwidth: 0% 0% 0% 80% 0% 0% 20% 0%
TSA_MAP: 0:ets 1:ets 2:ets 3:strict 4:ets 5:ets 6:strict 7:ets
IEEE 8021QAZ PFC TLV
Willing: no
MACsec Bypass Capable: no
PFC capable traffic classes: 8
PFC enabled: 3
End of LLDPDU TLV
$ rdma link
link rocep75s0/1 state ACTIVE physical_state LINK_UP netdev ens2
link rocep77s0/1 state ACTIVE physical_state LINK_UP netdev enp77s0
```

```
$ mkdir /sys/kernel/config/rdma_cm/rocep75s0
$ echo 96 > /sys/kernel/config/rdma_cm/rocep75s0/ports/1/default_roce_tos
$ mkdir /sys/kernel/config/irdma/rocep75s0
$ cd /sys/kernel/config/rdma_cm/rocep75s0/
$ echo 0x706050418020100 > up_up_map
$ echo 1 > roce_dcqcn_enable
$ echo 48 > cnp_up_override
$ echo 1 > up_map_enable
```

Example Scripts

- Setup ip using script.

```
$ cat set_ip.sh
ip addr add 192.168.1.60/24 dev ens2
ip link set dev ens2 mtu 5000
ip link set dev ens2 up
ip addr add 192.168.0.60/21 dev enp77s0
ip link set dev enp77s0 mtu 5000
ip link set dev enp77s0 up
```

- Setup lossless configuration Global Pause.

```
$ cat lossless_global_pause.sh
modprobe -v configfs
modprobe -v nvme-core multipath=yes
modprobe -v nvme
modprobe -v nvme_rdma
modprobe irdma roce_port_cfg=3
rdma link
ulimit -n 4096
ulimit -u 51296400
#To enable Global Pause on rnics
ethtool -A ens2 rx on tx on
ethtool -A enp77s0 rx on tx on
ethtool -a ens2
ethtool -a enp77s0
ethtool -G ens2 rx 2048 tx 2048
ethtool -G enp77s0 rx 2048 tx 2048
ethtool -g ens2
ethtool -g enp77s0
```

- Setup lossless configuration PFC with DCQCN enabled.

```
$ cat lossless_pfc.sh
#!/usr/bin/bash -x
modprobe -v configfs
modprobe -v nvme-core multipath=yes
modprobe -v nvme
modprobe -v nvme-rdma
modprobe -v irdma
rdma link
ulimit -n 4096
ulimit -u 51296400
```

```
#Port1:
ethtool -A ens2 rx off tx off
ethtool --set-priv-flags ens2 fw-lldp-agent off
#lldpad -d
lldptool -ti ens2
lldptool -Ti ens2 -V PFC willing=no enabled=3
#to remove the previous app entries
lldptool -Ti ens2 -V APP -d app=3,5,24
sleep 10
lldptool -Ti ens2 -V APP -d app=6,5,48
sleep 10
lldptool -Ti ens2 -V APP app=3,5,24
sleep 10
lldptool -Ti ens2 -V APP app=6,5,48
sleep 10
lldptool -Ti ens2 -V ETS-CFG willing=no tsa=0:ets,1:ets,2:ets,3:strict,4:ets,5:ets,6:strict,7:ets
tcbw=0,0,0,80,0,0,20,0 willing=no
lldptool -ti ens2
rdma link
ibv _ devices
mkdir /sys/kernel/config/rdma _ cm/rocep75s0
echo 96 > /sys/kernel/config/rdma _ cm/rocep75s0/ports/1/default _ roce _ tos

mkdir /sys/kernel/config/irdma/rocep75s0
cd /sys/kernel/config/irdma/rocep75s0/
echo 0x706050418020100 > up _ up _ map
echo 1 > roce _ dcqcn _ enable
echo 48 > cnp _ up _ override
echo 1 > up _ map _ enable

#Port2:
ethtool -A enp77s0 rx off tx off
ethtool --set-priv-flags enp77s0 fw-lldp-agent off
lldptool -ti enp77s0
lldptool -Ti enp77s0 -V PFC willing=no enabled=3
#to remove the previous app entries
lldptool -Ti enp77s0 -V APP -d app=3,5,24
sleep 10
lldptool -Ti enp77s0 -V APP -d app=6,5,48
sleep 10
lldptool -Ti enp77s0 -V APP app=3,5,24
sleep 10
lldptool -Ti enp77s0 -V APP app=6,5,48
sleep 10
lldptool -Ti enp77s0 -V ETS-CFG willing=no tsa=0:ets,1:ets,2:ets,3:strict,4:ets,5:ets,6:strict,7:ets
tcbw=0,0,0,80,0,0,20,0 willing=no
lldptool -ti enp77s0
rdma link
ibv _ devices
mkdir /sys/kernel/config/rdma _ cm/rocep77s0
echo 96 > /sys/kernel/config/rdma _ cm/rocep77s0/ports/1/default _ roce _ tos
```

```
mkdir /sys/kernel/config/irdma/rocep77s0
cd /sys/kernel/config/irdma/rocep77s0
echo 0x706050418020100 > up_up_map
echo 1 > roce_dcqcn_enable
echo 48 > cnp_up_override
echo 1 > up_map_enable
echo
```

5601 Great Oaks Parkway
San Jose, CA 95119, USA
www.westerndigital.com