Intel® Acceleration Stack User Guide

Intel FPGA Programmable Acceleration Card N3000

Updated for Intel® Acceleration Stack for Intel® Xeon® CPU with FPGAs: **1.1**

Online Version
Send Feedback

UG-20244

ID: 683040
Version: 2021.06.14

Contents

1. About this Document **4**
1.1. Acronym List 5
2. System Requirements **7**
2.1. Cooling Requirements 7
3. Hardware Installation **11**
3.1. Installing the Intel FPGA PAC N3000 11
4. Installing the OPAE Software **16**
4.1. Install Additional Packages 17
4.2. Install the Release Package 17
4.2.1. Remove Previous OPAE Packages 18
4.2.2. Install the Acceleration Stack for Runtime 19
4.2.3. Install the Acceleration Stack for Development 19
4.2.4. Verify the OPAE Installation 21
4.2.5. Install the Configuration Files 22
4.3. Identify the Intel MAX 10 Version on your Intel FPGA PAC N3000 23
4.3.1. FPGA Factory Image Overview 24
5. OPAE Tools **26**
5.1. Using fpgasupdate 26
5.2. Using fpgainfo 27
5.3. Test PCIe and External Memories with fpgabist 30
5.4. Test Network Loopback using fpgadiag 31
6. Sample Test: Native Loopback **32**
7. Installing the Intel XL710 Driver **34**
7.1. Updating the Intel XL710 Firmware 34
8. Configuring Ethernet Interfaces **38**
8.1. Modifying the Interface Maximum Transmission Unit (MTU) Size 40
8.2. Setting Forward Error Correction (FEC) Mode 41
8.3. Ethernet Pause Flow Control 42
8.4. Get Link Status and Statistics 44
9. Testing Network Loopback Using Data Plane Development Kit (DPDK) **46**
9.1. Test Using an External Traffic Generator 49
9.2. Test Using a Packet Generator 53
9.3. Troubleshooting in DPDK 55
9.4. Revert Back from DPDK to OPAE 56
10. Graceful Shutdown **57**
10.1. Background 57
10.2. Using OPAE 57
10.3. Using DPDK 60
11. Single Event Upset (SEU) **61**
11.1. OPAE Handling of SEU 61

11.2. DPDK Handling of SEU..... 63

12. Document Revision History for Intel Acceleration Stack User Guide: Intel FPGA PAC N3000..... 65

A. Troubleshooting.....66

A.1. If OPAE installation verification fails, how to install OPAE manually?.....66

B. Upgrade your Intel FPGA PAC N3000 with Production Version of BMC and Intel Arria 10 Image..... 67

B.1. Upgrading from 1.1 Beta to Production Version..... 67

B.1.1. Root Entry Hash Programmed.....68

B.1.2. Root Entry Hash Not Programmed..... 69

B.2. Upgrading from 1.1 Alpha-2 or Older to Production Version..... 70

C. Configure the 4.19 Kernel 76

D. fpgabist Sample Output..... 77

D.1. For 2x2x25G Configuration..... 77

D.2. For 8x10G Configuration.....85

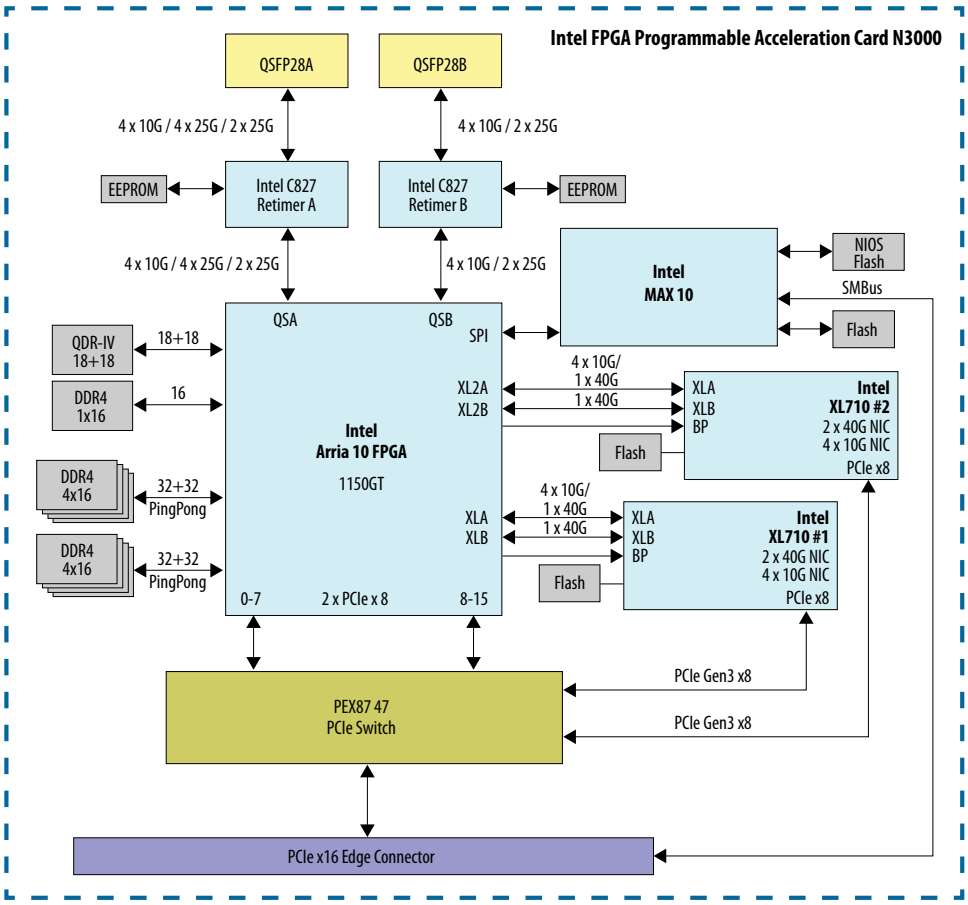
1. About this Document

This document provides:

- Instructions and requirements for installing the Intel FPGA Programmable Acceleration Card N3000 into a server.
- Instructions for installing the Open Programmable Acceleration Engine (OPAE) software on host Intel® Xeon® processor for managing and accessing the Intel FPGA PAC N3000.
- Instructions for installing the OPAE tools for validation of the Intel FPGA PAC N3000.
- Instructions for running built-in self-tests using the FPGA factory image.
- Instructions for installing the Intel XL710 driver and network testing tools for testing the ethernet capabilities.
- Information about Graceful Shutdown and Single Event Upset (SEU) handling.

The Intel Acceleration Stack for Intel Xeon CPU with FPGAs is a collection of software, firmware, and tools that allows both software and RTL developers to take advantage of the power of Intel FPGA PAC N3000. By offloading computationally intensive networking tasks to the FPGA, the acceleration platform allows the Intel Xeon processor to execute other critical processing tasks.

Figure 1. Block Diagram



1.1. Acronym List

Acronym	Expansion	Description
Intel FPGA PAC	Intel FPGA Programmable Acceleration Card	Intel FPGA PAC N3000 is a full-duplex 100 Gbps in-system re-programmable acceleration card for multi-workload networking application acceleration.
AFU	Accelerator Functional Unit	Hardware Accelerator implemented in FPGA logic which offloads a computational operation for an application from the CPU to improve performance.
AF	Acceleration Function	Compiled Hardware Accelerator image implemented in FPGA logic that accelerates an application.
API	Application Programming Interface	A set of subroutine definitions, protocols, and tools for building software applications.
FIU	FPGA Interface Unit	FIU is a platform interface layer that acts as a bridge between platform interfaces like PCIe* and AFU-side interfaces such as CCI-P.
OPAE	Open Programmable Acceleration Engine	The OPAE is a set of drivers, utilities, and API's for managing and accessing AFs.

continued...

Acronym	Expansion	Description
FME	FPGA Management Engine	The FME provides information about the FPGA platform including the OPAE version.
OTSU	One-Time Secure Update	Updates the Intel MAX® 10 Board Management Controller (BMC) with new image files to enable the Root of Trust (RoT) features.
RSU	Remote System Update	An RSU operation sends an instruction to the device to trigger a power cycle of the Intel FPGA PAC N3000 only. This will force reconfiguration from flash for either Intel MAX 10 BMC image (on devices that support it) or the FPGA image.

2. System Requirements

- Operation System:
  - CentOS Linux version 7.6 kernel 3.10 or kernel 4.19(1)
  - Red Hat* Enterprise Linux* (RHEL) version 7.6 kernel 3.10

  For information about the latest OS support, refer to the Getting Started page.
- Connectivity hardware for testing the Ethernet interfaces:
  - 10 Gbps QSFP+
  - 25 Gbps QSFP28

Refer to the Intel FPGA Programmable Acceleration Card N3000 Data Sheet for the following:

- List of supported QSFP+ and QSFP28 modules.
- Information on Intel FPGA PAC N3000 ordering part numbers (OPNs) that support Intel Acceleration Stack 1.1 production version.

While selecting a server, ensure that the server meets the following requirements (per slot):

- Power
- Cooling air flow
- Physical dimension

For more information, refer to the *Intel FPGA Programmable Acceleration Card N3000 Data Sheet*.

Note: To compile an AFU using the Intel Quartus® Prime Pro Edition software, your server must have at least 48 GB of system RAM.

2.1. Cooling Requirements

The high performance devices installed on the Intel FPGA PAC N3000 require server based forced air cooling to maintain proper operating temperature. This section provides air flow requirements for the Intel FPGA PAC N3000. Sufficient airflow keeps the Intel Arria® 10 GT junction temperature below 95 °C. Each data point corresponds to minimum airflow (Y-axis) through the card for a corresponding card inlet temperature (X-axis).

(1) Refer to Configure the 4.19 Kernel on page 76 to support OPAE installation.

Figure 2. Air Flow Pattern

Key parameters:

- Required Cooling Airflow (CFM): Volumetric flow rate, in cubic feet per minute, of air passing through the PCIe faceplate of the Intel FPGA PAC N3000.
- T_J FPGA Junction Temperature
- T_{LA} Local Ambient Temperature(2): Temperature of forced air as it enters the Intel FPGA PAC N3000.

  Note: In many systems, T_{LA} is higher than the room ambient due to heating affects of chassis components.

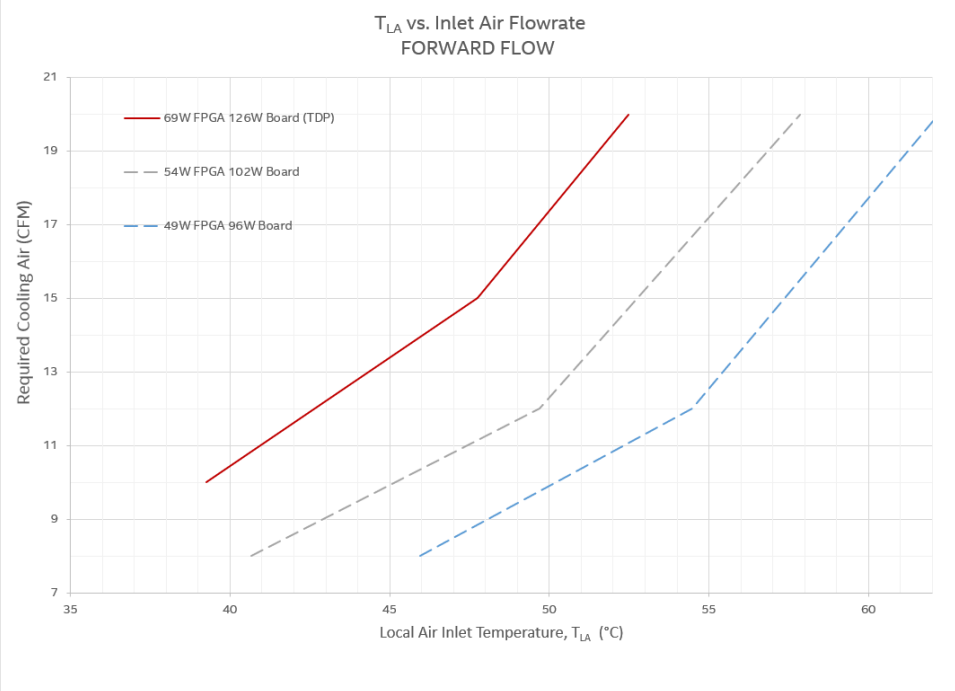
(2) T_{LA} values shown in the tables and charts of this section are informational, and may represent conditions outside of the supported operating range.

Table 1. Power and Thermal Requirements

The Intel FPGA PAC N3000 uses a passive heatsink. The server must provide sufficient forced airflow to keep the FPGA within the following operating conditions:

Parameters	Maximum Values
Intel Arria 10 FPGA Thermal Design Power (TDP)(3)	≤ 69 W
Intel FPGA PAC N3000 Thermal Design Power (TDP)	≤ 126 W
Recommended FPGA Maximum Operating Temperature	95°C
FPGA $T_{J\text{-}MAX}$ or Thermal Protection Shutdown	100°C
Maximum T_{LA} (forward airflow)	51°C
Maximum T_{LA} (reverse airflow)	45°C

Figure 3. Forward Flow Cooling Curve



(3) Intel Arria 10 FPGA TDP cannot be obtained from on-board BMC sensors. Use the Intel Quartus Prime Power Analyzer to verify compliance with this value for your design.

Table 2. Local Air Inlet Temperature and Air Flow Values for Forward Cooling

To maintain T_J = 95°C					
FPGA Power < 49 W Board Power < 96 W		FPGA Power < 54 W Board Power < 102 W		FPGA Power < 69 W Board Power < 126 W (TDP)	
Max. T_{LA} (°C)	Air Flow (CFM)	Max. T_{LA} (°C)	Air Flow (CFM)	Max. T_{LA} (°C)	Air Flow (CFM)
46	8	40.7	8	39.2	10
54.5	12	49.8	12	47.8	15
62	20	57.9	20	52.3	20

Figure 4. Reverse Flow Cooling Curve

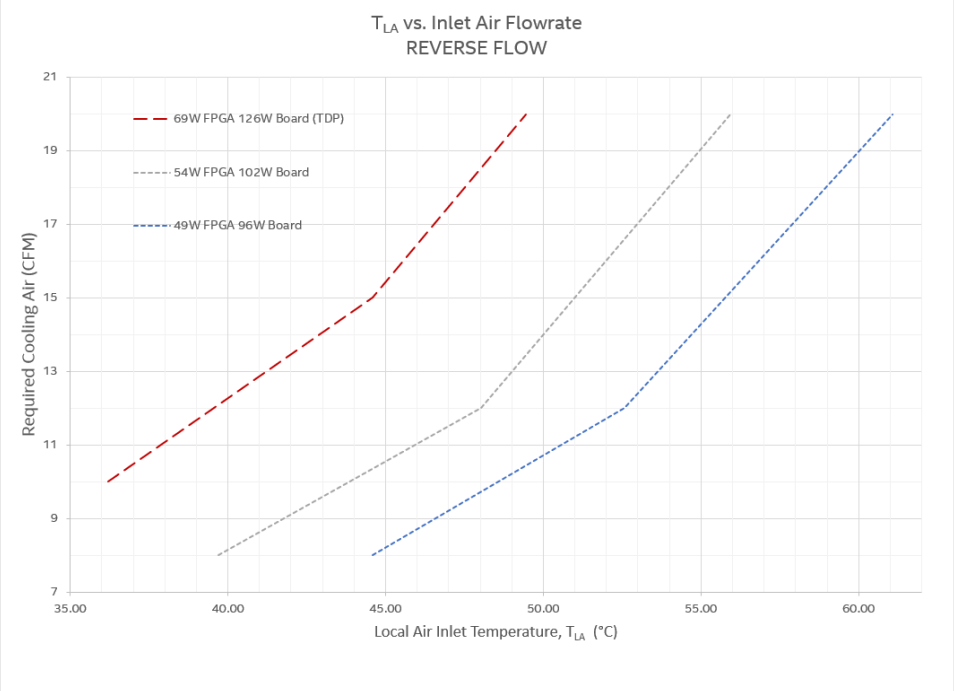


Table 3. Local Air Inlet Temperature and Air Flow Values for Reverse Cooling

To maintain T_J = 95°C					
FPGA Power < 49 W Board Power < 96 W		FPGA Power < 54 W Board Power < 102 W		FPGA Power < 69 W Board Power < 126 W (TDP)	
Max. T_{LA} (°C)	Air Flow (CFM)	Max. T_{LA} (°C)	Air Flow (CFM)	Max. T_{LA} (°C)	Air Flow (CFM)
44.6	8	39.8	8	36.1	10
52.6	12	48	12	44.7	15
61	20	56	20	49.4	20

Note: These flow curves are based on numerical analysis and should be used as a starting point for thermal design estimation. Thermal performance of host systems may vary. Therefore, you should perform in-system thermal validation.

3. Hardware Installation

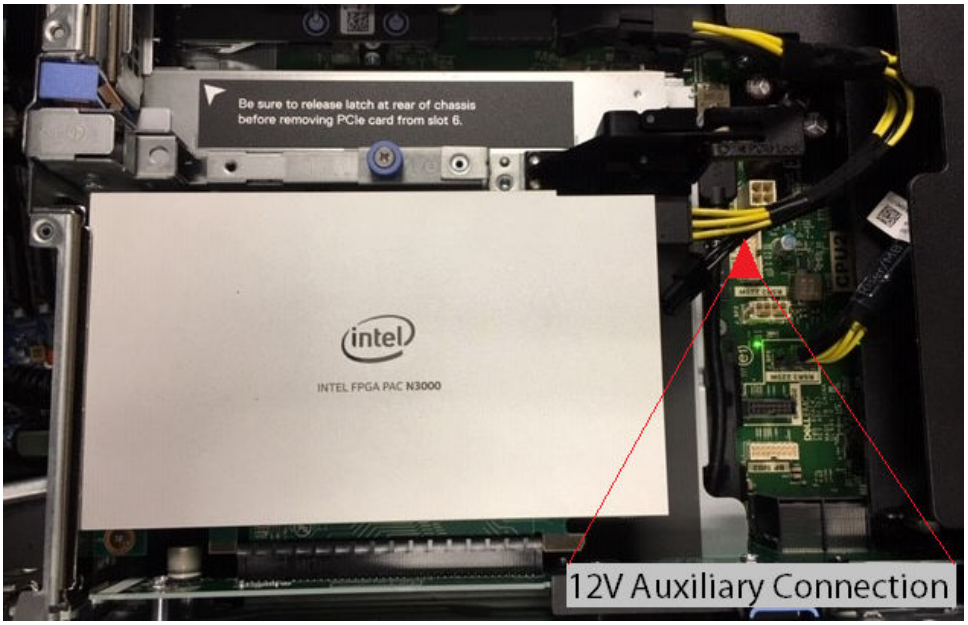
To operate the Intel FPGA PAC N3000 in your server, you must have the following:

- PCI Express* Gen3x16 slot with physical space for a full height half length PCIe form factor board
- Auxiliary 12 V 6-pin power connector
- Server that provides sufficient airflow for a given air inlet temperature

Note: The Intel FPGA PAC N3000 cannot operate without the 6-pin auxiliary power connector. Internal board circuitry prevents operation without the auxiliary power source and PCIe connector power source.

Note: If the Intel FPGA PAC N3000 is not integrated into a server closed loop fan control system, you must set the fan speed to 100%. The fan speed (100%) setting must be applied to avoid overheating when the server turns on. When the BMC detects that the card has overheated, it powers down the card to prevent damage.

Figure 5. Typical Intel FPGA PAC N3000 Installation in a Server



3.1. Installing the Intel FPGA PAC N3000

Complete these steps to install the Intel FPGA PAC N3000:

1. Power down the system.
2. Plug the Intel FPGA PAC N3000 into a PCIe x16 physical and x16 electrical slot on the motherboard.
3. Connect the auxiliary power to the 12 V 6-pin connector using an applicable cable.

   Note: Make sure the auxiliary power cable does not block airflow to the Intel FPGA PAC N3000.
4. Enable the following options in the BIOS:
   - Intel VT-x (Intel Virtualization Technology for IA-32 and Intel 64 Processors)
   - Intel VT-d (Intel Virtualization Technology for Directed I/O)
5. For network testing, you can insert a loopback module into each QSFP28 port.

   Note: ESD protection is required while handling the Intel FPGA PAC N3000.

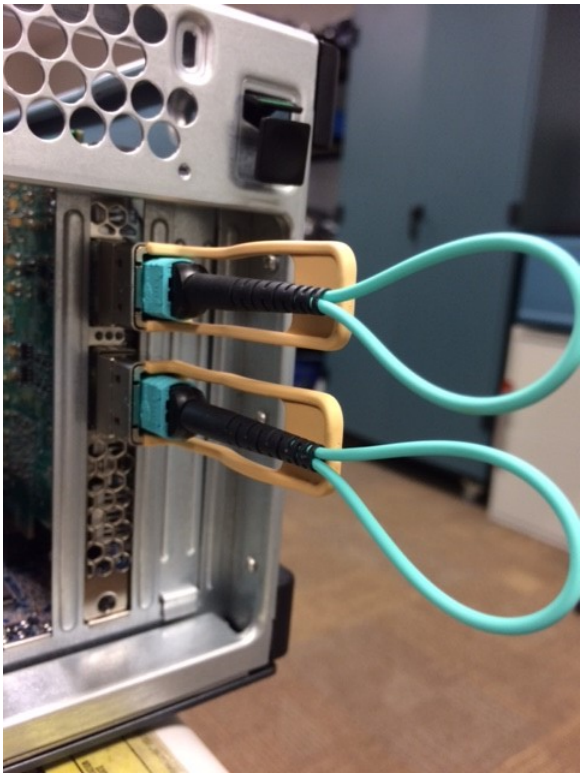
   Warning: Take caution when you are inserting and removing the cards into PCIe slots. The bottom side of the card has capacitors and resistors that can be knocked off if the cards scrapes against edges or corners of the slot in the chassis.

Figure 6. Example of QSFP Non-optical Modules

This is a 25 GbE QSFP setup.

Figure 7. Example of QSFP Loopback Optical Modules

This figure shows the correct orientation of the QSFP module for optical loopback testing. The loopback configuration consists of two Intel 40GBASE-SR4 QSFP+ optical modules and two 10Gtek 12-Core MPO OM3 Fiber Optic Loopback Cables.



6. Power on the system and observe the Ethernet status LEDs which are located between the QSFP connectors. The LED operation is described in the table below:

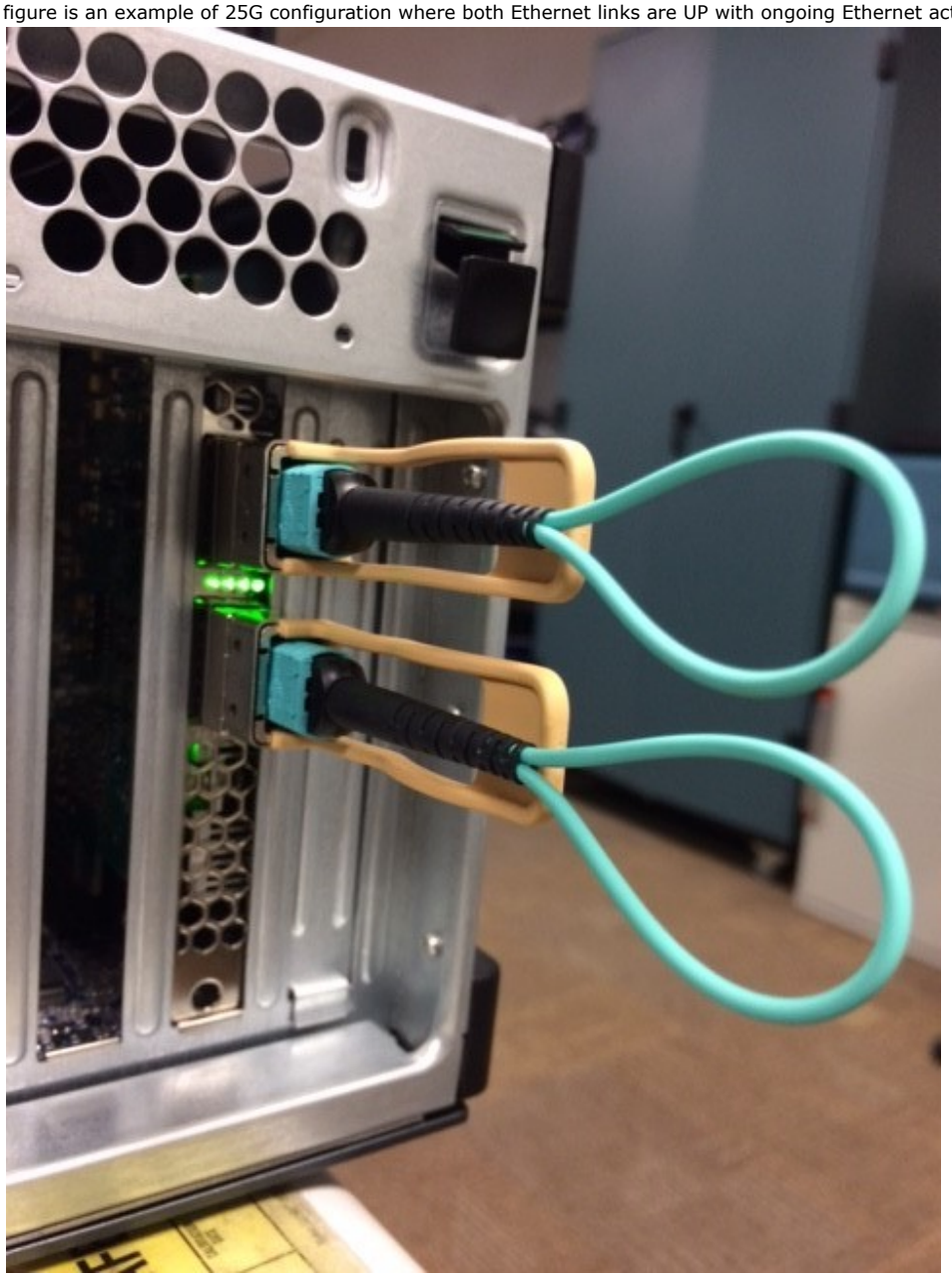
Table 4. LED Behavior

LED Type	Description
Connectivity LED	Yellow means link is up with link speed of 10G.
	Green means link is up with link speed of 25G.
	Off means link is down.
Activity LED	Green blinking at 1 Hz means link activity present.
	Off means link is down or no activity.
All LEDs blinking yellow	It means either: • 12 V Auxiliary or PCIe edge supply voltage is below 10.46 V or • FPGA core temperature reaches 100°C or • Board temperature reaches 85°C You should check the following: • Card insertion in PCIe slot. • 12 V Auxiliary connection on the Intel FPGA PAC N3000 and on the server motherboard. • Fan setting for cooling air flow. Sufficient airflow is required whenever the Intel FPGA PAC N3000 is powered on.

For details on the location of Connectivity and Activity LEDs, refer to the Intel FPGA Programmable Acceleration Card N3000 Data Sheet.

Figure 8. Ethernet Status LEDs Power-On Indication

This figure is an example of 25G configuration where both Ethernet links are UP with ongoing Ethernet activity.



Note: If your server is set up with secure boot, the OPAE Remote Setup (RSU) command will not function. This RSU limitation is due to secure boot having the following limitations:

- Using kexec to start an unsigned kernel image.
- Hibernation and resume from hibernation.
- User-space access to physical memory and I/O ports.
- Module parameters that allow setting memory and I/O port addresses.
- Writing to MSRs through /dev/cpu/*/msr.
- Use of custom ACPI methods and tables.
- ACPI APEI error injection.

While you implement secure boot in your server, you must power cycle your server to load a new FPGA image rather than using the RSU command.

4. Installing the OPAE Software

The OPAE is a software framework delivered as part of the Intel Acceleration Stack for managing and accessing the Intel FPGA PAC.

The following section describes the installation of OPAE on a freshly imaged server with supported OS and kernel. The host must have internet connectivity to retrieve additional software packages. The installation steps require `sudo` or `root` privileges on your host.

Note: The OPAE version created for Intel FPGA PAC N3000 is not compatible with any other Intel FPGA PAC.

To verify that you have the correct kernel, kernel source and header, perform the following steps:

1. Check the kernel version running on the server:

   ```
   $ uname -a
   ```

   Sample output:

   ```
   Linux rae-xxx 3.10.0-957.el7.x86_64
   ```

2. List the kernel source on the system:

   ```
   $ ls -l /usr/src/kernels/
   ```

   Sample output:

   ```
   drwxr-xr-x. 22 root root 4096 Jun 21 13:05 3.10.0-957.el7.x86_64
   ```

3. List the installed kernel header:

   ```
   $ rpm -qa | grep kernel-header

   kernel-headers-3.10.0-957.el7.x86_64
   ```

   If the kernel source and header do not match the kernel version running on the server, there can be issues with installing OPAE driver.

To mitigate this issue:

1. Remove the incompatible kernel header:

   ```
   $ sudo yum remove kernel-headers.x86_64
   ```

2. Install the correct kernel source:

   ```
   $ sudo yum install "kernel-devel-uname-r == $(uname -r)"
   ```

3. Install the correct kernel header:

   ```
   $ sudo yum install kernel-headers-`uname -r`
   ```

4.1. Install Additional Packages

Before you install and build the OPAE software, you must install the required packages. Run the following commands:

```
$ sudo yum install gcc gcc-c++ \
cmake make autoconf automake libxml2 \
libxml2-devel json-c-devel boost ncurses ncurses-devel \
ncurses-libs boost-devel libuuid libuuid-devel python2-jsonschema \
doxygen hwloc-devel libpng12 rsync openssl-devel \
bc python-devel python-libs python-sphinx openssl python2-pip
```

```
$ sudo pip install intelhex
```

For Red Hat Enterprise Linux (EPEL) system, run the following commands to install the addtional packages:

```
$ sudo subscription-manager repos --enable "rhel-*-optional-rpms" \
--enable "rhel-*-extras-rpms" --enable "rhel-ha-for-rhel-*-server-rpms"
```

```
$ sudo yum install https://dl.fedoraproject.org/pub/epel/epel-release-\
latest-7.noarch.rpm
```

For CentOS system, run the following command to install the extra repository:

```
$ sudo yum install epel-release
```

4.2. Install the Release Package

Before installing the release package, ensure that the Intel FPGA PAC N3000 is installed properly as mentioned in the Hardware Installation on page 11.

The installers for Intel FPGA PAC N3000 allow easy installation of the release package.

- Acceleration Stack Installers

  To install the Acceleration Stack, select either the **Acceleration Stack for Runtime** (`n3000_ias_1_1_pv_rte_installer.tar.gz`) or the **Acceleration Stack for Development** (`n3000_ias_1_1_pv_dev_installer.tar.gz`). Each package includes three components:

  1. Runtime (rte) or Development (dev) Acceleration Stack installer script (`n3000-1.3.6-*-setup.sh`)
  2. Firmware files (`N3000_XL710_firmware.zip`) — To update the Intel XL710 Firmware.
  3. Supplemental files (`N3000_supplemental_files.zip`) — Includes:
     - A sample `hello_afu` example that executes on loaded factory image.
     - A helper script to find the PCIe Root port for Intel FPGA PAC N3000.

The following table explains the differences between the two versions of the Acceleration Stack:

Details	Acceleration Stack for Runtime	Acceleration Stack for Development
	runtime (rte) installer	development (dev) installer
Purpose	Provides necessary environment to execute the AFUs as well as allows for software development of host application.	Provides necessary environment to execute the AFUs as well as allows for software development of host application. Additionally, it also includes development environment for Intel Arria 10 GT FPGA.
OPAE Software Development Kit (SDK) Version	1.3.6-4	
Intel Quartus Prime Pro Edition	Not included or required	Included: Intel Quartus Prime Pro Edition software version 19.2 with IP licenses required to create a programmable Intel Arria 10 GT FPGA image.
Default installation location	N/A	`/home/<username>/inteldevstack`

- Configuration installers (2x2x25G, 4x25G or 8x10G):

  You can pick one of the desired configuration installer for your Intel FPGA PAC N3000.

  Note: The XL710 devices are configured in different modes to support either 10G or 25G traffic. The XL710 devices cannot be configured to switch between 10G and 25G, and thus Intel recommends you to download the valid configuration installer.

  Depending on the configuration on your Intel FPGA PAC N3000, the XL710 has one of the following device ID's:

XL710 Device ID	Valid Configuration
0x0d58(25G)	2x2x25G 4x25G
0x0cf8(10G)	8x10G

  To identify the XL710 device ID on the Intel FPGA PAC N3000:

```
$ lspci -d :0d58
```

```
$ lspci -d :0cf8
```

  Install the necessary files to update the Intel Arria 10 FPGA , Intel MAX 10 RTL and Firmware to a Root of Trust state. The files are installed at location:

  — `/usr/share/opae/n3000/one-time-update/<config directory>/`

  where `<config directory>` = 25G or 10G

  — `/usr/share/opae/super-rsu/<config directory>/`

  where `<config directory>` = 2x2x25G or 4x25G or 8x10G

4.2.1. Remove Previous OPAE Packages

Remove any previous installation of OPAE or FPGA and Intel MAX 10 update package using the following:

```
$ sudo yum remove opae*
```

4.2.2. Install the Acceleration Stack for Runtime

1. Download, extract, and change permission for the RTE installer:

```
$ tar xvzf n3000_ias_1_1_pv_rte_installer.tar.gz
```

```
$ cd n3000_ias_1_1_pv_rte_installer
```

```
$ chmod +x n3000-1.3.6-rte-setup.sh
```

2. Run the script:

```
$ sudo ./n3000-1.3.6-rte-setup.sh -y --owner <user[:group]>
```

The `--owner` argument allows you to change the ownership of directories to a given user. Running interactively without the -y option is not supported. For example:

```
$ sudo ./n3000-1.3.6-rte-setup.sh -y --owner john:john
```

The installation will take a few minutes to complete.

Sample output:

```
Running setup
Beginning installation
Processing group "OPAE Software"
Analyzing dependencies...
Analyzing packages to install...
error running: ['yum', 'info', 'opae-intel-fpga-driver.x86_64']
error running: ['yum', 'info', 'opae.admin.noarch']
error running: ['yum', 'info', 'opae-libs.x86_64']
error running: ['yum', 'info', 'opae-tools.x86_64']
error running: ['yum', 'info', 'opae-tools-extra.x86_64']
error running: ['yum', 'info', 'opae-devel.x86_64']
        Installing OPAE Software packages...
                opae-intel-fpga-driver-2.0.1-6.x86_64.rpm
                opae.admin-1.0.2-3.noarch.rpm
                opae-libs-1.3.6-4.x86_64.rpm
                opae-tools-1.3.6-4.x86_64.rpm
                opae-tools-extra-1.3.6-4.x86_64.rpm
                opae-devel-1.3.6-4.x86_64.rpm
Processing group "OPAE PACSign"
Analyzing dependencies...
Analyzing packages to install...
error running: ['yum', 'info', 'opae.pac_sign.x86_64']
        Installing OPAE PACSign packages...
                opae.pac_sign-1.0.2-3.x86_64.rpm
Extracting opae-1.3.6-4.tar.gz
```

The message *error running: ['yum' , …]* in the above output is expected and can be ignored.

4.2.3. Install the Acceleration Stack for Development

1. Download, extract, and change permission for the DEV installer:

```
$ tar xvzf n3000_ias_1_1_pv_dev_installer.tar.gz
```

```
$ cd n3000_ias_1_1_pv_dev_installer
```

```
$ chmod +x  n3000-1.3.6-dev-setup.sh
```

2. Run the script:

```
$ sudo ./n3000-1.3.6-dev-setup.sh -y --owner <user[:group]>
```

The `--owner` argument allows you to change the ownership of directories to a given user. Running interactively without the -y option is not supported. For example:

```
$ sudo ./n3000-1.3.6-dev-setup.sh -y --owner john:john
```

The installation will take approximately 20 minutes to complete.

Sample output:

```
Running setup
Beginning installation
Processing group "OPAE Software"
Analyzing dependencies...
Analyzing packages to install...
error running: ['yum', 'info', 'opae-intel-fpga-driver.x86_64']
error running: ['yum', 'info', 'opae.admin.noarch']
error running: ['yum', 'info', 'opae-libs.x86_64']
error running: ['yum', 'info', 'opae-tools.x86_64']
error running: ['yum', 'info', 'opae-tools-extra.x86_64']
error running: ['yum', 'info', 'opae-devel.x86_64']
        Installing OPAE Software packages...
                opae-intel-fpga-driver-2.0.1-6.x86_64.rpm
                opae.admin-1.0.2-3.noarch.rpm
                opae-libs-1.3.6-4.x86_64.rpm
                opae-tools-1.3.6-4.x86_64.rpm
                opae-tools-extra-1.3.6-4.x86_64.rpm
                opae-devel-1.3.6-4.x86_64.rpm
Processing group "OPAE PACSign"
Analyzing dependencies...
Analyzing packages to install...
error running: ['yum', 'info', 'opae.pac_sign.x86_64']
        Installing OPAE PACSign packages...
                opae.pac_sign-1.0.2-3.x86_64.rpm
Extracting opae-1.3.6-4.tar.gz
Extracting pac_n3000_rtl_1.3.6.tar.gz
Installing main Quartus package: QuartusProSetup-19.2.0.57-linux.run
                Installing update: quartus-19.2-0.01vc-linux.run
Source /home/<user>/inteldevstack/bin/init_env.sh to setup your environment.
Changing ownership on /home/<user>/inteldevstack
Installation done
```

The message *error running: ['yum' , …]* in the above output is expected and can be ignored.

3. Add the Intel Quartus Prime development tool to the path. This is required for AFU compilation:

```
$ source /home/<username>/inteldevstack/bin/init_env.sh
```

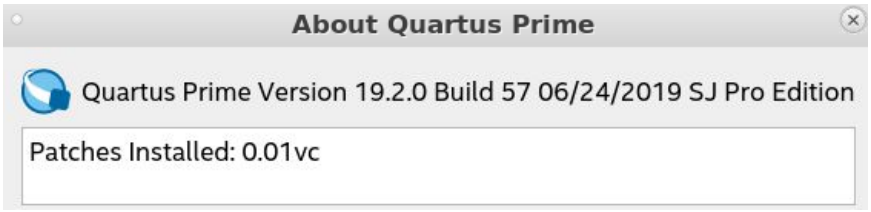
```
Adding /home/<username>/inteldevstack/intelFPGA_pro/quartus/bin to PATH
Adding /home/<username>/inteldevstack/intelFPGA_pro/
nios2eds/bin/gnu/H-x86_64-pc-linux-gnu/bin to PATH
Adding /home/<username>/inteldevstack/bin to PATH
```

4. Verify Intel Quartus Prime installation by bringing up Intel Quartus Prime GUI and verifying the version and IP licenses:

```
$ quartus
```

a. Click **Help** > **About Quartus Prime** to verify Intel Quartus Prime version:

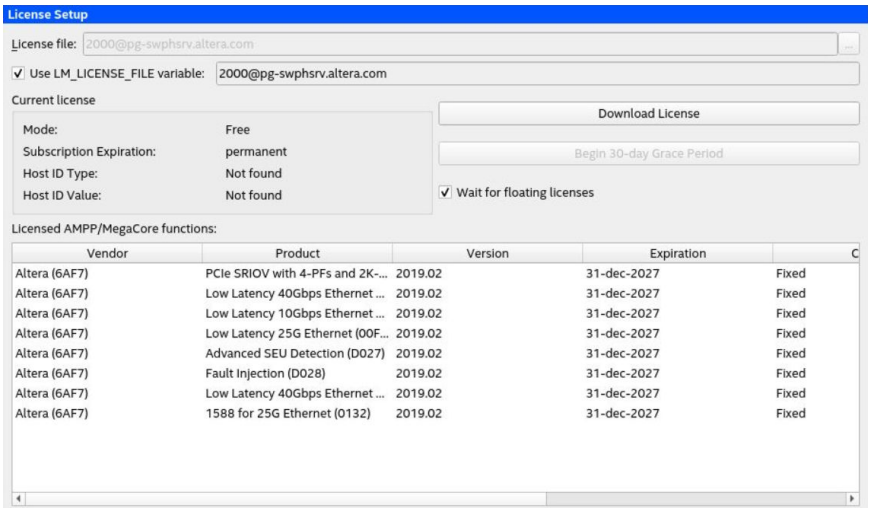
Figure 9. Intel Quartus Prime Version



If **patch .01vc** is not installed, then verify that `init_env.sh` ran successfully.

b. Click **Tools** > **License Setup** to verify IP licenses.

Figure 10. IP Licenses



4.2.4. Verify the OPAE Installation

After executing the Acceleration Stack for Runtime or Development installer, ensure that you have successfully installed OPAE.

- Verify the OPAE package installation:

```
$ rpm -qa | grep opae

opae-intel-fpga-driver-2.0.1-6.x86_64
opae-devel-1.3.6-4.x86_64
opae-libs-1.3.6-4.x86_64
opae.admin-1.0.2-3.noarch
opae.pac_sign-1.0.2-3.x86_64
opae-tools-1.3.6-4.x86_64
opae-tools-extra-1.3.6-4.x86_64
```

- Verify the OPAE driver installation:

```
$ lsmod | grep fpga

ifpga_sec_mgr          13757  1 intel_max10
intel_fpga_fme         71678  0
```

```
intel_fpga_afu         36165  0
fpga_mgr_mod           14812  1 intel_fpga_fme
intel_fpga_pci         26500  2 intel_fpga_afu,intel_fpga_fme
```

```
$ lsmod | grep pac_n3000_net
pac_n3000_net          28483  1 c827_retimer
```

- Verify if the Linux has enumerated the Intel FPGA PAC N3000 FPGA Management Engine Device (FME):

```
$ lspci | grep 0b30

61:00.0 Processing accelerators: Intel Corporation Device 0b30
```

The above output provides the PCIe BDF (Bus : Device : Function) value for the Intel FPGA PAC. Here, the `61:00.0` indicates the bus is 61, device is 00 and function is 0. The PCIe BDF value varies based on systems, therefore your values can be different. Record this value for future use.

If the verification of OPAE installation fails, refer to Troubleshooting on page 66.

4.2.5. Install the Configuration Files

The configuration (cfg) installer installs all the binary ingredients (FPGA, Intel MAX 10) required to upgrade the Intel FPGA PAC N3000 to current release version.

1. Install the appropriate configuration setup file (2x2x25G, 4x25G, or 8x10G).
   - Filename: `n3000-1.3.6-cfg-*-setup.sh`
   - To change permission:

     ```
     $ sudo chmod +x n3000-1.3.6-cfg-*-setup.sh
     ```

2. Run the script:

   ```
   $ sudo ./n3000-1.3.6-cfg-*-setup.sh -y --install-dir /tmp/tmp_cfg \
   --owner  <user[:group]>
   ```

   Important: You must use the `--install-dir option` with the configuration script to prevent overwriting of the `/inteldevstack` or `/intelrtestack` directory.

   Sample output:

   ```
   Running setup
   Beginning installation
   Processing group "OPAE FPGA Image Files for 4x25G"
   Analyzing dependencies...
   Analyzing packages to install...
   error running: ['yum', 'info', 'opae-one-time-update-n3000-25G.noarch']
   error running: ['yum', 'info', 'opae-super-rsu-n3000-4x25G.noarch']
                  Installing OPAE FPGA Image Files for 4x25G packages...
                                 opae-one-time-update-
   n3000-25G-1.3.6-6.noarch.rpm
                                 opae-super-rsu-n3000-4x25G-1.3.6-6.noarch.rpm
   Changing ownership on /tmp/tmp_cfg
   Installation done
   ```

   The message *error running: ['yum' , …]* in the above output is expected and can be ignored.

3. Verify proper installation of the one-time update (OTSU) and super-rsu files.

- If you install 2x2x25G or 4x25G configuration installer, the installed files are located at

```
ls -l /usr/share/opae/n3000/one-time-update/25G/
```

```
ls -l /usr/share/opae/n3000/super-rsu/<config directory>/
```

where <config directory> = 2x2x25G or 4x25G.

- If you install 8x10G configuration installer, the installed files are located at:

```
ls -l /usr/share/opae/n3000/one-time-update/10G/
```

```
ls -l /usr/share/opae/n3000/super-rsu/<config directory>/
```

where <config directory> = 8x10G.

4.3. Identify the Intel MAX 10 Version on your Intel FPGA PAC N3000

Run the following command and verify the output with the table below:

```
$sudo fpgainfo fme
```

Table 5. Intel MAX 10 Versions

Acceleration Stack 1.1 for Intel FPGA PAC N3000	Intel MAX 10 NIOS Firmware (FW)	Intel MAX 10 Build
Production	D.2.0.19	D.2.0.6
Beta	D.2.0.12	D.2.0.5
Alpha-2	D.1.0.13	D.1.0.14
Alpha-1	D.1.0.13	D.1.0.13

If your Intel FPGA PAC N3000 does not have the production version of Intel MAX 10 NIOS Firmware (FW) and Build, refer to the section: Upgrade your Intel FPGA PAC N3000 with Production Version of BMC and Intel Arria 10 Image on page 67. When you upgrade your Intel FPGA PAC N3000 from Alpha-2/Beta version to Production version, refer to the following table for Bitstream ID and PR Interface ID:

Table 6. Bitstream ID After Upgrade

Intel FPGA PAC N3000	Factory Partition Image			User Partition Image		
	Configuration	Bitstream ID	PR Interface ID	Configuration	Bitstream ID	PR Interface ID
BD-NVV-N3000-2	2x2x25G	0x2300041001 0309	a5d72a3c-c8b0-4939-912c-f715e5dc10ca	4x25G	0x2300011001 0309	f3c99413-5081-4aad-bced-07eb84a6d0bb
BD-NFV-N3000-1	8x10G	0x2300001001 0309	901dd697-ca79-4b05-b843-8138cefa2846	8x10G	0x2300001001 0309	901dd697-ca79-4b05-b843-8138cefa2846

If your Intel FPGA PAC N3000 has the production version of Intel MAX 10 Firmware and Build, install the PV 1.1 Patch. When your Intel FPGA PAC N3000 is shipped with the production version, refer to the following table for Bitstream ID and PR Interface ID:

Table 7. Bitstream ID for Intel FPGA PAC N3000 shipped with Production version

Intel FPGA PAC N3000	Factory Partition Image			User Partition Image		
	Configuration	Bitstream ID	PR Interface ID	Configuration	Bitstream ID	PR Interface ID
BD-NVV-N3000-2	2x2x25G	0x2300041001030F	a5d72a3c-c8b0-4939-912c-f715e5dc10ca	4x25G	0x2300011001030F	f3c99413-5081-4aad-bced-07eb84a6d0bb
BD-NFV-N3000-1	8x10G	0x2300001001030F	901dd697-ca79-4b05-b843-8138cefa2846	8x10G	0x2300001001030F	901dd697-ca79-4b05-b843-8138cefa2846

4.3.1. FPGA Factory Image Overview

The Intel FPGA PAC N3000 has an on-board flash with two partitions (user and factory) for storing two FPGA image files known as user image and factory image. A new Intel FPGA PAC N3000 is provided with the factory image in both factory and user partition of the flash.

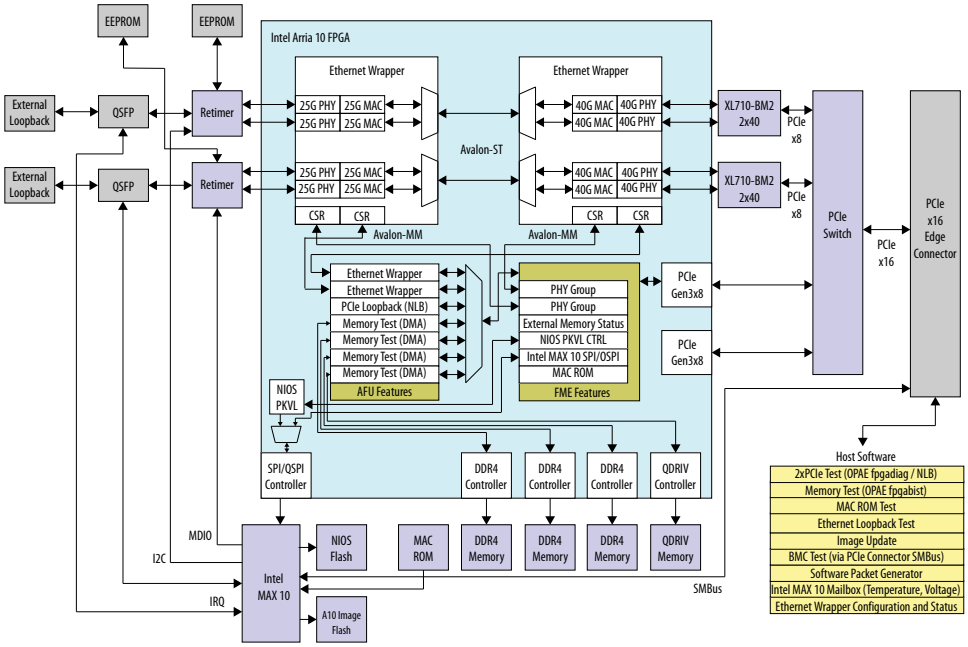
After an OTSU, in 8x10G configuration, the Intel FPGA PAC N3000 is loaded with 8x10G image in both factory and user partition. While in 4x25G and 2x2x25G configurations, the Intel FPGA PAC N3000 is loaded with 2x2x25G in factory partition and 4x25G in user partition.

When the image in the user partition fails to load, the Intel FPGA PAC N3000 reverts back and boots from factory partition. This factory image loaded into the user partition provides basic functionality to demonstrate all the interfaces including Ethernet and external memory interfaces.

The factory image includes the following Intellectual Property (IP) to support in the development of Accelerator Function (AF):

- The PCIe IP core
- The Core Cache Interface protocol (CCI-P) fabric
- DDR4 memory interface controller IP
- QDR4 memory interface controller IP
- 10 or 25 GbE physical interface and MACs with pass-through connectivity between Intel Ethernet Connection C827 Retimer and Intel Ethernet Controller XL710-BM2
- FPGA Management Engine (FME)
- Nios® core to configure the Intel Ethernet Connection C827 Retimers

Figure 11. Example: Factory Image for 2x2x25G



5. OPAE Tools

The following OPAE tools are provided:

- **fpgasupdate**: This tool updates Intel MAX 10 BMC image and firmware, root entry hash, and FPGA static region (SR) user image in the user partition..
- **fpgainfo**: This tool displays FPGA information derived from sysfs files.
- **fpgabist**: This tool performs board self-diagnostic tests for PCIe and external memories.
- **fpgadiag**: This tool performs board self-diagnostic tests for network loopback.
- **fpgad**: This service allows for monitoring critical sensors and protect against server crashing.
- **fpgastats**: This tool provides the MAC statistics for both line side and host side.

5.1. Using fpgasupdate

The fpgasupdate tool updates board firmware including BMC and FPGA SR user image. This section describes how to update the FPGA SR user image. While in the upgrade process, the fpgasupdate tool securely updates the Intel MAX 10 BMC with an Intel provided signed file.

The new Intel FPGA PAC N3000 is shipped with either the 4x25G or the 8x10G factory image for 25G cards and 10G cards respectively. The following steps describe how to load FPGA images into the FPGA flash user partition. You may follow the steps to load or re-load the factory image if required.

Note: Do not switch between 8x10G to 2x2x25G or 4x25G Intel Arria 10 images. The XL710 devices are configured in different modes to support either 10G or 25G traffic. The XL710 devices cannot be configured to switch between 10G and 25G, and thus Intel recommends you to not switch Intel Arria 10 image supporting different speed configurations.

Depending on the configuration on your Intel FPGA PAC N3000, the XL710 has one of the following device ID's:

XL710 Device ID	Valid Configuration
0x0d58(25G)	2x2x25G 4x25G
0x0cf8(10G)	8x10G

To identify the XL710 device ID on the Intel FPGA PAC N3000:

```
$ lspci -d :0d58
```

```
$ lspci -d :0cf8
```

1. Run the `fpgasupdate` command:

```
$ sudo fpgasupdate <FPGA Bitstream> <PCIe B:D.F>
```

Note: Running `fpgasupdate` involves binary file verification and writing the FPGA flash, as a result the `fpgasupdate` command takes approximately 40 minutes to complete.

Note: If you have programmed the static region root entry hash, then the `sr_vista_rot_*_unsigned.bin` must be signed with appropriate root key and code signing key using the appropriate Hardware Security Module (HSM). For more information, refer to the Security User Guide: Intel FPGA Programmable Acceleration Card N3000.

If you want to reload the Intel provided factory image in the FPGA flash user partition, perform the following:

```
$ sudo fpgasupdate /usr/share/opae/n3000/super-rsu/<config>/\
sr_vista_rot_*_unsigned.bin [PCIe B:D.F]
```

where `<config>` = `2x2x25G` or `4x25G` or `8x10G`, based on the installed configuration.

2. Perform remote system update to power cycle the Intel FPGA PAC N3000 so that the updated images are loaded into FPGAs:

```
$ sudo rsu bmcimg B:D.F
```

Note: As a result of using the `rsu` command, the host rescans the PCI bus and may assign a different Bus/Device/Function (B/D/F) value than the originally assigned value. The Intel XL710 Ethernet controller should be considered unavailable during the operation. Intel recommends you to stop or pause any applications until the update is complete.

5.2. Using fpgainfo

Command synopsis: `fpgainfo <command> [<args>]`

Table 8. fpgainfo Commands

command	args (optional)	Description
	--help, -h	Prints help information and exit
	--bus, -B	Provides PCIe bus number of resource
	--device, -D	Provides PCIe device number of resource
	--function, -F	Provides PCIe function number of resource
errors fme	--clear, -c	Provides/clear errors of FME
errors port	--clear, -c	Provides/clear errors of port
errors all	--clear, -c	Provides/clear errors of both FME and port
power		Provides total power in watts that the FPGA hardware consumes
temp		Provides FPGA temperature values in degrees Celsius
port		Provides information about the port
fme		Provides information about the FME

continued...

command	args (optional)	Description
bmc		Provides BMC sensors information
mac		Provides information about MAC ROM connected to FPGA
phy	-G <group>	Provides information about ethernet PHYs in FPGA. <group> can be 0, 1, all.

Example 1. fpgainfo fme

```
$ sudo fpgainfo fme
```

Sample output based on 2x2x25G:

```
Board Management Controller, Intel MAX 10 NIOS FW version D.2.0.19
Board Management Controller, Intel MAX 10 Build version D.2.0.6
//****** FME ******//
Object Id                        : 0xEF00000
PCIe s:b:d.f                     : 0000:b2:00.0
Device Id                        : 0x0b30
Numa Node                        : 1
Ports Num                        : 01
Bitstream Id                     : 0x23000410010309
Bitstream Version                : 0.2.3
Pr Interface Id                  : a5d72a3c-c8b0-4939-912c-f715e5dc10ca
Boot Page                        : user
```

Sample output based on 8x10G:

```
Board Management Controller, Intel MAX 10 NIOS FW version D.2.0.19
Board Management Controller, Intel MAX 10 Build version D.2.0.6
//****** FME ******//
Object Id                        : 0xEF00000
PCIe s:b:d.f                     : 0000:89:00.0
Device Id                        : 0x0b30
Numa Node                        : 1
Ports Num                        : 01
Bitstream Id                     : 0x23000010010309
Bitstream Version                : 0.2.3
Pr Interface Id                  : 901dd697-ca79-4b05-b843-8138cefa2846
Boot Page                        : user
```

Example 2. fpgainfo bmc

```
$ sudo fpgainfo bmc
```

Sample output based on 2x2x25G:

```
Board Management Controller, Intel MAX 10 NIOS FW version D.2.0.19
Board Management Controller, Intel MAX 10 Build version D.2.0.6
//****** BMC SENSORS ******//
Object Id                        : 0xEF00000
PCIe s:b:d.f                     : 0000:b2:00.0
Device Id                        : 0x0b30
Numa Node                        : 1
Ports Num                        : 01
Bitstream Id                     : 0x23000410010309
Bitstream Version                : 0.2.3
Pr Interface Id                  : a5d72a3c-c8b0-4939-912c-f715e5dc10ca
( 1) Board Power                 : 74.18 Watts
( 2) 12V Backplane Current       : 3.24 Amps
( 3) 12V Backplane Voltage       : 12.16 Volts
( 4) 1.2V Voltage                : 1.19 Volts
( 6) 1.8V Voltage                : 1.80 Volts
```

```
( 8) 3.3V Voltage             : 3.25 Volts
(10) FPGA Core Voltage        : 0.90 Volts
(11) FPGA Core Current        : 18.45 Amps
(12) FPGA Die Temperature     : 73.50 Celsius
(13) Board Temperature        : 46.50 Celsius
(14) QSFP0 Supply Voltage     : 0.00 Volts
(15) QSFP0 Temperature        : N/A
(24) 12V AUX Current          : 2.85 Amps
(25) 12V AUX Voltage          : 12.19 Volts
(37) QSFP1 Supply Voltage     : 0.00 Volts
(38) QSFP1 Temperature        : N/A
(44) PKVL0 Core Temperature   : 73.50 Celsius
(45) PKVL0 SerDes Temperature : 73.50 Celsius
(46) PKVL1 Core Temperature   : 74.00 Celsius
(47) PKVL1 SerDes Temperature : 74.50 Celsius
```

Note: When the copper modules are connected to QSFP, you get N/A output for the QSFP temperature values as the QSFP EEPROM does not support copper modules. But, the QSFP EEPROM does support optical modules.

Sample output based on 8x10G:

```
Board Management Controller, Intel MAX 10 NIOS FW version D.2.0.19
Board Management Controller, Intel MAX 10 Build version D.2.0.6
//****** BMC SENSORS ******//
Object Id                     : 0xEF00000
PCIe s:b:d.f                  : 0000:89:00.0
Device Id                     : 0x0b30
Numa Node                     : 1
Ports Num                     : 01
Bitstream Id                  : 0x23000010010309
Bitstream Version             : 0.2.3
Pr Interface Id               : 901dd697-ca79-4b05-b843-8138cefa2846
( 1) Board Power              : 70.01 Watts
( 2) 12V Backplane Current    : 3.06 Amps
( 3) 12V Backplane Voltage    : 12.17 Volts
( 4) 1.2V Voltage             : 1.19 Volts
( 6) 1.8V Voltage             : 1.80 Volts
( 8) 3.3V Voltage             : 3.25 Volts
(10) FPGA Core Voltage        : 0.90 Volts
(11) FPGA Core Current        : 17.18 Amps
(12) FPGA Die Temperature     : 79.00 Celsius
(13) Board Temperature        : 43.50 Celsius
(14) QSFP0 Supply Voltage     : 0.00 Volts
(15) QSFP0 Temperature        : N/A
(24) 12V AUX Current          : 2.69 Amps
(25) 12V AUX Voltage          : 12.19 Volts
(37) QSFP1 Supply Voltage     : N/A
(44) PKVL0 Core Temperature   : 76.00 Celsius
(45) PKVL0 SerDes Temperature : 76.50 Celsius
(46) PKVL1 Core Temperature   : 76.50 Celsius
(47) PKVL1 SerDes Temperature : 77.00 Celsius
```

Example 3. fpgainfo phy

```
$ sudo fpgainfo phy -B 0xb2
```

Sample output based on 2x2x25G:

```
Board Management Controller, Intel MAX 10 NIOS FW version D.2.0.19
Board Management Controller, Intel MAX 10 Build version D.2.0.6
//****** PHY ******//
Object Id                     : 0xEF00000
PCIe s:b:d.f                  : 0000:b2:00.0
Device Id                     : 0x0b30
Numa Node                     : 1
Ports Num                     : 01
```

```
Bitstream Id                : 0x23000410010309
Bitstream Version           : 0.2.3
Pr Interface Id             : a5d72a3c-c8b0-4939-912c-f715e5dc10ca
//****** PHY GROUP 0 ******//
Direction                   : Line side
Speed                       : 25 Gbps
Number of PHYs              : 4
//****** PHY GROUP 1 ******//
Direction                   : Host side
Speed                       : 40 Gbps
Number of PHYs              : 4
//****** Intel C827 Retimer ******//
Port0 25G                   : Up
Port1 25G                   : Up
Port2 25G                   : Up
Port3 25G                   : Up
Retimer A Version           : 101c.1064
Retimer B Version           : 101c.1064
```

Sample output based on 8x10G:

```
Board Management Controller, Intel MAX 10 NIOS FW version D.2.0.19
Board Management Controller, Intel MAX 10 Build version D.2.0.6
//****** PHY ******//
Object Id                   : 0xEF00000
PCIe s:b:d.f                : 0000:89:00.0
Device Id                   : 0x0b30
Numa Node                   : 1
Ports Num                   : 01
Bitstream Id                : 0x23000010010309
Bitstream Version           : 0.2.3
Pr Interface Id             : 901dd697-ca79-4b05-b843-8138cefa2846
//****** PHY GROUP 0 ******//
Direction                   : Line side
Speed                       : 10 Gbps
Number of PHYs              : 8
//****** PHY GROUP 1 ******//
Direction                   : Host side
Speed                       : 10 Gbps
Number of PHYs              : 8
//****** Intel C827 Retimer ******//
Port0 10G                   : Up
Port1 10G                   : Up
Port2 10G                   : Up
Port3 10G                   : Up
Port4 10G                   : Up
Port5 10G                   : Up
Port6 10G                   : Up
Port7 10G                   : Up
Retimer A Version           : 101c.1064
Retimer B Version           : 101c.1064
```

Note: When using the 4x25G configuration, the Retimer B is held in reset to reduce power. Therefore, Retimer B firmware version is not listed (0000.0000) in the `fpgainfo phy` output based on 4x25G.

5.3. Test PCIe and External Memories with fpgabist

Tests are included to demonstrate the performance of PCIe and external memories.

Requirements:

- OPAE tool fpgabist requires that hugepage to be set.
  - For CentOS:

    ```
    # echo 200 > /sys/kernel/mm/hugepages/hugepages-2048kB/nr_hugepages
    ```

  - For Red Hat:

    ```
    # echo 200 > /proc/sys/vm/nr_hugepages
    ```

- Commands must be run as root.

Note: The fpgabist diagnostic tool only works when the Intel supplied factory images are programmed into the Intel FPGA PAC N3000. For more information about how to reload the factory image, refer to Using fpgasupdate on page 26.

Example 4. Using fpgabist

```
# fpgabist -B 0x8a -i 0x0b30
```

Note: Your bus (-B) value may be different.

Related Information

fpgabist Sample Output on page 77

5.4. Test Network Loopback using fpgadiag

The test requires use of an external traffic generator to send traffic through QSFP ports, the `--side host` enables loopback on the host side of the Intel Arria 10 FPGA.

```
$    sudo fpgadiag -B 0x3e -m fpgalpbk --side host –-direction remote –-enable
```

To disable:

```
$    sudo fpgadiag -B 0x3e -m fpgalpbk --side host –-direction remote --disable
```

Note: Since all four Ethernet channels in the 4x25G configuration are in QSFP0, only one QSFP port indicates linkup status through LEDs.

6. Sample Test: Native Loopback

This section describes how to run a memory copy test using the Intel provided FPGA factory image and `hello_fpga.c` host program. The FPGA factory image includes logic to support this test and an internal register with the expected **AFU UUID**. The `hello_fpga.c` only works with an FPGA image with this **AFU UUID**. The acceleration logic (NLB) in the FPGA is programmed to copy `CSR_NUM_LINES` (cache lines) from source to destination buffer on the host system. For more information refer to the Native Loopback Accelerator Functional Unit (AFU) User Guide for Intel FPGA Programmable Acceleration Card N3000.

Make sure the hugepage is allocated:

- For CentOS:

```
$ sudo sh -c "echo 200 > /sys/kernel/mm/hugepages/hugepages-2048kB/\
nr_hugepages"
```

- For Red Hat:

```
# echo 200 > /proc/sys/vm/nr_hugepages
```

Note: Commands must be run as root.

Extract the package: `N3000_supplemental_files.zip` which is provided as part of the Acceleration Stack installer.

```
$    unzip N3000_supplemental_files.zip
```

```
$    cd N3000_supplemental_files
```

```
$    gcc -o hello_fpga -std=gnu99 -rdynamic -ljson-c -luuid -lpthread \
-lopae-c -lm -Wl,-rpath -lopae-c hello_fpga.c
```

```
$    sudo ./hello_fpga
```

Sample output:

```
Using OPAE C library version '1.3.6' build '99fa5de'
Running Test
Running on bus 0x15.
dfh = 1000000080000001f
id[0] = c000c9660d824272
id[1] = 9aeffe5f84570612
dfh = 2000000080000000
id[0] = a9149a35bace01ea
id[1] = ef82def7f6ec40fc
dfh = 2000000080000000
id[0] = a9149a35bace01ea
id[1] = ef82def7f6ec40fc
dfh = 2000000080000000
id[0] = a9149a35bace01ea
id[1] = ef82def7f6ec40fc
dfh = 2000000080000000
```

```
id[0] = a9149a35bace01ea
id[1] = ef82def7f6ec40fc
dfh = 1000010080001070
id[0] = f89e433683f9040b
id[1] = d8424dc4a4a3c413
Found NLB0 at offset 0x28000
Done Running Test
```

Note: On a multi card system, pass PCIe bus argument -B 0x<xx>

7. Installing the Intel XL710 Driver

1. Download the i40e driver (2.9.21 version) from Intel Resource and Design Center.
2. Install driver as root.

```
$ tar xvzf i40e-2.9.21.tar.gz ; cd i40e-2.9.21
```

```
$ cd src
```

```
$ sudo make install
```

```
$ sudo rmmod i40e
```

```
$ sudo insmod i40e.ko
```

3. Download the i40e virtual function driver (3.7.53 version) from Intel Resource and Design Center.
4. Install driver as root.

```
$ tar xzvf iavf-3.7.53.tar.gz ; cd iavf-3.7.53
```

```
$ cd src
```

```
$ sudo make install
```

Insert the iavf kernel module to add support for virtual functions for Intel XL710:

```
$ sudo insmod iavf.ko
```

7.1. Updating the Intel XL710 Firmware

To update the Intel XL710 firmware, follow these steps:

1. Extract the `N3000_XL710_firmware.zip` which is provided as part of the Acceleration Stack installer:

```
$ unzip N3000_XL710_firmware.zip
```

2. Change directory:

```
cd N3000_XL710_firmware/
```

```
$ export N3000_XL710_FIRMWARE=$PWD
```

3. The XL710 devices on each Intel FPGA PAC N3000 are configured during board manufacturing to support either 10G or 25G operation. You cannot change the Ethernet network operation (10G or 25G). The XL710 PCIe device ID identifies whether 10G or 25G support is configured. The device ID for 25G is 0x0d58 and for 10G is 0x0cf8 respectively.

Determine the XL710 device ID on your Intel FPGA PAC N3000 using the following commands:

```
$ lspci -d :0d58
$ lspci -d :0cf8
```

Based on the result, you will run nvmupdate with the appropriate config file.

4. Download the NVM Update Package version 7.0: NVMUpdatePackage_700_Series.zip.
5. Extract the nvmupdate64e tool:

```
$ unzip NVMUpdatePackage_700_Series.zip
```

```
$ cd  NVMUpdatePackage_700_Series/
```

```
$ tar xvzf 700Series_NVMUpdatePackage_v7_00_Linux.tar.gz
```

```
$ chmod +x 700Series/Linux_x64/nvmupdate64e
```

```
$ sudo cp 700Series/Linux_x64/nvmupdate64e $N3000_XL710_FIRMWARE/
```

6. Update the XL710:

```
$ cd  $N3000_XL710_FIRMWARE/
```

   - For device ID: 0cf8

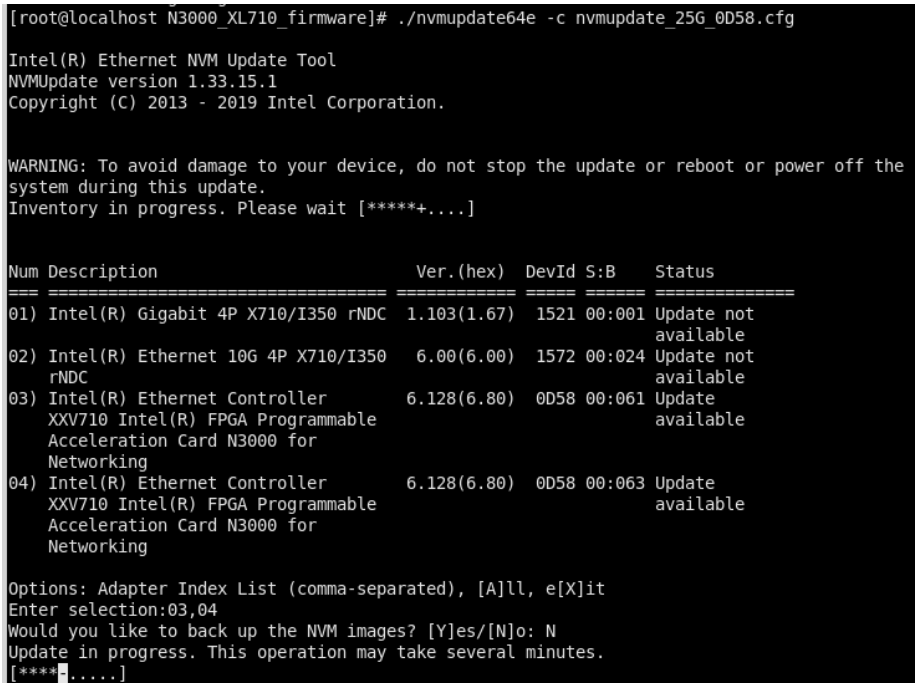
     ```
     $ sudo ./nvmupdate64e -c nvmupdate_10G_0CF8.cfg
     ```

   - For device ID: 0d58

     ```
     $ sudo ./nvmupdate64e -c nvmupdate_25G_0D58.cfg
     ```

7. A menu lists all the XL710 devices and specifies which ones have updates (only 0D58 or 0CF8 device should show **Update Available**).
   a. Enter the **Num** values (use comma to separate multiple devices)
   b. Hit **Enter** and wait. An on-screen will prompt to hit any key when programming is completed.

Figure 12. Step Illustration



8. Power cycle the card using:

```
$ sudo rsu bmcimg <FPGA PCIe B:D.F>
```

In the future, you can choose to upgrade to Intel XL710 Firmware 7.3 or later and install the corresponding i40e and iavf driver. The following steps are instructive and provide guidance on how to perform firmware updates for future versions:

1. Check the version compatibility between firmware and driver. Refer to the Feature Support Matrix.
2. Download the NVM Update package version 7.3 or above from download center.
3. Extract the `nvmupdate64e` tool:

```
$ unzip NVMUpdatePackage_*_Series.zip
$ cd  NVMUpdatePackage_*_Series/
$ tar xvzf 700Series_NVMUpdatePackage_*_Linux.tar.gz
$ chmod +x 700Series/Linux_x64/nvmupdate64e
$ cd 700Series/Linux_x64/
```

4. Upgrade Intel XL710 firmware:

```
# ./nvmupdate64e
```

   For more details, refer to the corresponding `Readme` file.

5. The NVMUpdate utility returns an exit code of zero after successful completion of the update.
6. Power cycle the card:

```
$ sudo rsu bmcimg [PCIe B:D.F]
```

Note: You must not use firmware version 7.1 and 7.2, since these versions do not support Intel XL710 device ID 0d58.

8. Configuring Ethernet Interfaces

The Intel FPGA PAC N3000 contains multiple Ethernet MAC points where each point has specific naming, monitoring and configuration operations. The following figures illustrate the Ethernet data path for each network configuration.

Figure 13. 8x10G Configuration

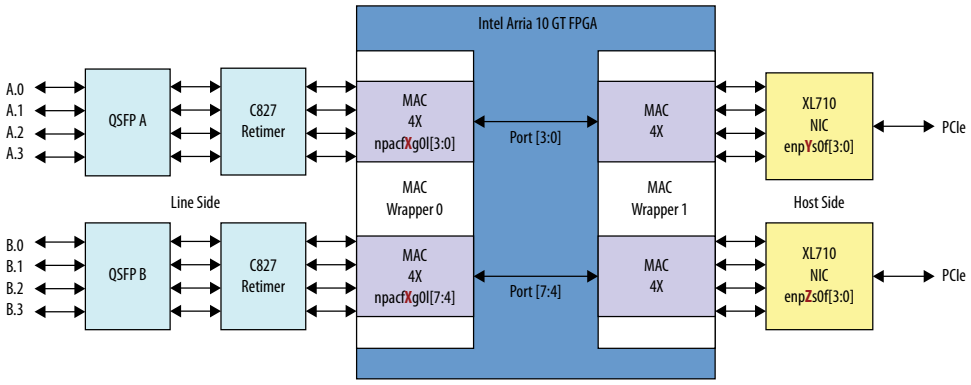


Figure 14. 2x2x25G Configuration

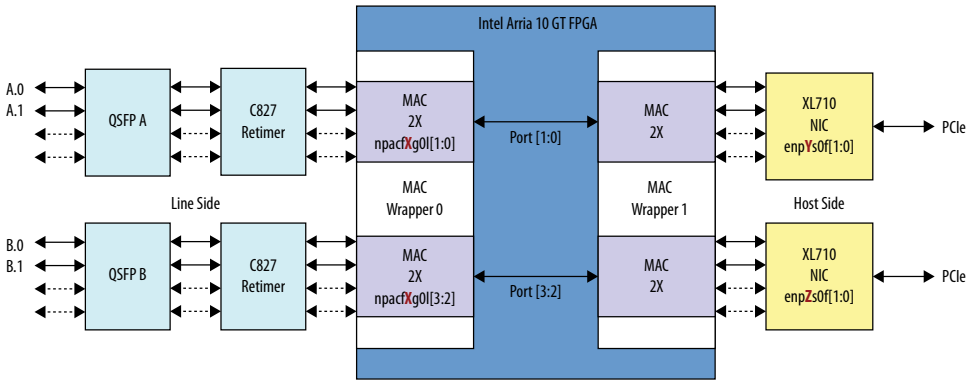
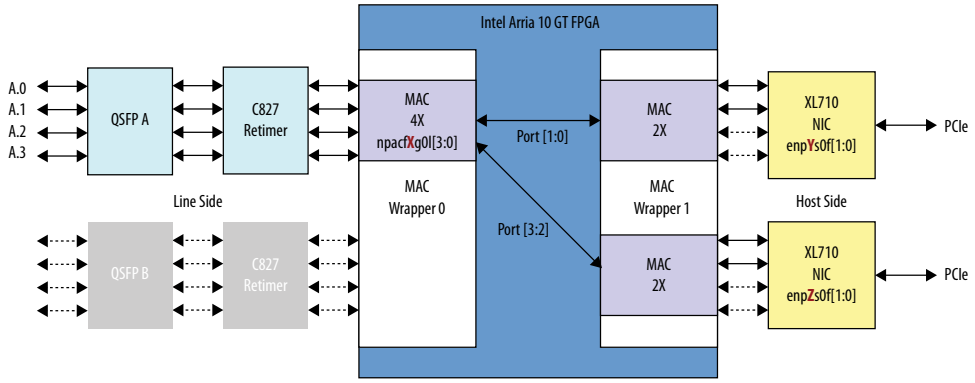


Figure 15. 4x25G Configuration



The above figures show example device naming conventions for the XL710 `enp[Y:Z]s0f[3:0]`. Your server may have a different naming convention and numbering scheme. You will need the network logical names to use Linux tools for link configuration and monitoring. To find the network logical names of a specific Intel FPGA PAC N3000 in your server, perform the following steps:

1. List the available Intel FPGA PAC N3000 in your server using:

```
$ sudo fpgainfo fme
```

Sample output:

```
Board Management Controller, Intel MAX 10 NIOS FW version D.2.0.19
Board Management Controller, Intel MAX 10 Build version D.2.0.6
//****** FME ******//
Object Id                        : 0xEF00000
PCIe s:b:d.f                     : 0000:15:00.0
Device Id                        : 0x0b30
Numa Node                        : 0
Ports Num                        : 01
Bitstream Id                     : 0x23000410010309
Bitstream Version                : 0.2.3
Pr Interface Id                  : a5d72a3c-c8b0-4939-912c-f715e5dc10ca
Boot Page                        : user
```

2. Use the following command to find the logical name(s) of the Ethernet interfaces on that Intel FPGA PAC N3000:

```
ls -la /sys/class/net
```

Sample output:

```
lrwxrwxrwx. 1 root root 0 Nov 20 06:07 enp20s0f0 -> ../../devices/
pci0000:11/0000:11:00.0/0000:12:00.0/0000:13:08.0/0000:14:00.0/net/enp20s0f0
lrwxrwxrwx. 1 root root 0 Nov 20 06:07 enp20s0f1 -> ../../devices/
pci0000:11/0000:11:00.0/0000:12:00.0/0000:13:08.0/0000:14:00.1/net/enp20s0f1
lrwxrwxrwx. 1 root root 0 Nov 20 06:07 enp22s0f0 -> ../../devices/
pci0000:11/0000:11:00.0/0000:12:00.0/0000:13:10.0/0000:16:00.0/net/enp22s0f0
lrwxrwxrwx. 1 root root 0 Nov 20 06:07 enp22s0f1 -> ../../devices/
pci0000:11/0000:11:00.0/0000:12:00.0/0000:13:10.0/0000:16:00.1/net/enp22s0f1
lrwxrwxrwx. 1 root root 0 Nov 20 02:34 lo -> ../../devices/virtual/net/lo
lrwxrwxrwx. 1 root root 0 Nov 20 22:44 npacf0g0l0 -> ../../devices/
pci0000:11/0000:11:00.0/0000:12:00.0/0000:13:09.0/0000:15:00.0/fpga/intel-
fpga-dev.0/intel-fpga-fme.0/pac_n3000_net.2.auto/net/npacf0g0l0
lrwxrwxrwx. 1 root root 0 Nov 20 22:44 npacf0g0l1 -> ../../devices/
pci0000:11/0000:11:00.0/0000:12:00.0/0000:13:09.0/0000:15:00.0/fpga/intel-
fpga-dev.0/intel-fpga-fme.0/pac_n3000_net.2.auto/net/npacf0g0l1
lrwxrwxrwx. 1 root root 0 Nov 20 22:44 npacf0g0l2 -> ../../devices/
pci0000:11/0000:11:00.0/0000:12:00.0/0000:13:09.0/0000:15:00.0/fpga/intel-
```

```
fpga-dev.0/intel-fpga-fme.0/pac_n3000_net.2.auto/net/npacf0g0l2
lrwxrwxrwx. 1 root root 0 Nov 20 22:44 npacf0g0l3 -> ../../devices/
pci0000:11/0000:11:00.0/0000:12:00.0/0000:13:09.0/0000:15:00.0/fpga/intel-
fpga-dev.0/intel-fpga-fme.0/pac_n3000_net.2.auto/net/npacf0g0l3
```

For example:

This listing is example of the 8x10G network configuration. The logical device names `npacf0g0l[3:0]` represents the Ethernet MAC wrapper 0 on the line side Intel Arria 10 FPGA. The logical device names `enp[Y:Z]s0f[1:0]` are the XL710 Ethernet ports.

The `pac_n3000_net` platform device driver creates the standard Linux network device interfaces for each Intel Arria 10 FPGA Ethernet MAC pair. It provides C827 re-timer information for unified network status reporting. It enables use of standard Linux tools for both link configuration and monitoring.

```
$ lsmod | grep pac_n3000_net
pac_n3000_net          28483  1 c827_retimer
```

8.1. Modifying the Interface Maximum Transmission Unit (MTU) Size

The default Maximum Transmission Unit (MTU) size for 10G FPGA MAC Wrapper 0 and Wrapper 1 is 1518 bytes. The default MTU size for 25G FPGA MAC Wrapper 0 and Wrapper 1 is 9600. The default MTU size for XL710 is 1500 bytes. You must configure FPGA MAC wrappers and XL710 to have the same MTU setting to ensure each MAC will allow your desired maximum packet size.

Command for configuring MTU of FPGA MAC wrapper 0:

```
$ ip link set dev npacfXgYlZ mtu <#>

<#> = desired MTU setting
```

Command for configuring MTU of XL710:

```
$ ip link set dev <XL710 interface name> mtu <#>

<#> = desired MTU setting
```

Example of current settings:

```
$ ip link show npacf0g0l0

48: npacf0g0l0: <LOWER_UP> mtu 9600 qdisc noop state UNKNOWN mode DEFAULT group
default qlen 1000
    link/generic
```

```
$ ip link show enp20s0f0

52: enp20s0f0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP mode
DEFAULT group default qlen 1000
    link/ether 64:4c:36:00:17:28 brd ff:ff:ff:ff:ff:ff
```

Example: Set MTU to 9600 for both FPGA and XL710

```
$ sudo ip link set dev npacf0g0l0 mtu 9600
```

```
$ sudo ip link set dev enp20s0f0 mtu 9600
```

```
$ ip link show npacf0g0l0

48: npacf0g0l0: <LOWER_UP> mtu 9600 qdisc noop state UNKNOWN mode DEFAULT group
default qlen 1000
    link/generic
```

```
$ ip link show enp20s0f0
52: enp20s0f0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9600 qdisc mq state UP mode
DEFAULT group default qlen 1000
    link/ether 64:4c:36:00:17:28 brd ff:ff:ff:ff:ff:ff
```

To set the FPGA MAC Wrapper 1 MTU, use the following command:

```
$ fpgadiag -B <bus> -m fpgamac --side=host --mtu <#>

<bus> = PCIe bus of FPGA in 0xYZ format
<#> = desired MTU setting
```

To check FPGA MAC Wrapper 1 MTU:

```
$ sudo fpgadiag -b <bus> -m fpgamac --side=host --mtu
```

Sample output:

```
============================================================
maximum frame length     | transmit         | receive          |
mac 0                    | 9600             | 9600             |
mac 1                    | 9600             | 9600             |
mac 2                    | 9600             | 9600             |
mac 3                    | 9600             | 9600             |
```

8.2. Setting Forward Error Correction (FEC) Mode

The Intel FPGA PAC N3000 supports different forward error correction (FEC) modes to enhance data reliability for 25GbE network configurations. The default FEC mode is Reed Solomon FEC. The following FEC modes are available:

Table 9. FEC Modes

fec_mode	Mode
no	No FEC
kr	Fire Code Forward Error Correction (IEEE 802.3 Clause 74)
rs	Reed Solomon Forward Error Correction (IEEE 802.3 Clause 108)

Note: The configurable FEC is only supported for 2x2x25G and 4x25G network configurations. For 8x10G, the FEC setting has no impact and this operation is invalid.

To set FEC mode:

```
$ sudo fecmode -B <bus> <mode>

<mode> = ‘no’, ‘kr’, ‘rs’
<bus> = PCIe bus of FPGA in the format “0xYZ”
```

Note: The `fecmode` command causes a board level rsu event while changing FEC modes. The rsu event causes a board level reset which causes previously configured Ethernet settings to revert back to default settings. Intel recommends you first set FEC mode, then configure Ethernet and other board level settings. The rsu event may also cause the PCIe bus number to change.

To get FEC mode:

```
$ fecmode -B <bus>

<bus> = PCIe bus of FPGA in the format “0xYZ”
```

For example:

```
$ fecmode -B 0xb3
FEC mode in current driver: rs
FEC mode in current hardware: rs
```

```
$ sudo fecmode -B 0xb3 kr
reloading driver with new parameter 'kr'
performing remote system update
2019-11-14 10:00:34,121 - [[pci_address(0000:b3:00.0), pci_id(0x8086, 0x0b30)]]
performing RSU operation
2019-11-14 10:00:34,123 - [[pci_address(0000:ae:00.0), pci_id(0x8086, 0x2030)]]
removing device from PCIe bus
2019-11-14 10:00:34,124 - waiting 10 seconds for boot
2019-11-14 10:00:44,135 - rescanning PCIe bus: /sys/devices/pci0000:ae/pci_bus/
0000:ae
2019-11-14 10:00:49,119 - RSU operation complete
Done
```

```
$ fecmode -B 0xb3
FEC mode in configuration: kr
FEC mode in current driver: kr
FEC mode in current hardware: kr
```

8.3. Ethernet Pause Flow Control

The Intel FPGA PAC N3000 supports pause frame operation for:

- 25G as described in Flow Control section of *25G Ethernet Intel Arria 10 FPGA IP User Guide*.
- 10G as described in Flow Control section of *Low Latency Ethernet 10G MAC Intel FPGA IP User Guide*.

The pause frame generation and response to pause frame reception is disabled by default. You can read the current setting of pause frame generation using:

```
$ ethtool --show-pause npacf0g0l0
```

```
Pause parameters for npacf0g0l0:
Autonegotiate:    off
RX:         off
TX:         off
```

To turn on the pause frame generation:

1. Set Pause Quanta: Configurable register used to set the desired delay time embedded in the pause frame packet requesting peer to stop transmitting for a period defined by the quanta. One quanta equals 512-bit times.
2. Hold-off Quanta: Configurable register used to set the desired delay time between consecutive pause frames packets in quanta or 512-bit times.
3. Pause Frame Enable: Allows you to enable or disable pause frame behavior using the `ethtool`.

In a multicard system, if you want to configure pause frame on only a single PCIe device, run the following command to find the device mapping for the specific FPGA PCIe:

```
ls -l /sys/class/fpga/intel-fpga-dev.*
```

Sample output:

```
/sys/class/fpga/intel-fpga-dev.0 -> ../../devices/
pci0000:11/0000:11:00.0/0000:12:00.0/0000:13:09.0/0000:15:00.0/fpga/intel-fpga-
dev.0
```

```
/sys/class/fpga/intel-fpga-dev.1 -> ../../devices/pci0000:ae/
0000:ae:00.0/0000:af:00.0/0000:b0:09.0/0000:b2:00.0/fpga/intel-fpga-dev.1
```

To turn on the pause frame generation for only FPGA PCIe 15:00.0, replace the `intel-fpga-dev.*` with device instance id `intel-fpga-dev.0` in below commands.

Example of setting pause frame generation:

```
# echo 200 > /sys/class/fpga/intel-fpga-dev.*/intel-fpga-fme.*/
pac_n3000_net.*.auto/\
net/npacf0g0l0/tx_pause_frame_quanta
```

```
# cat /sys/class/fpga/intel-fpga-dev.*/intel-fpga-fme.*/pac_n3000_net.*.auto/\
net/npacf0g0l0/tx_pause_frame_quanta
0xc8
```

```
# echo 200 > /sys/class/fpga/intel-fpga-dev.*/intel-fpga-fme.*/
pac_n3000_net.*.auto/\
net/npacf0g0l0/tx_pause_frame_holdoff
```

```
# cat /sys/class/fpga/intel-fpga-dev.*/intel-fpga-fme.*/pac_n3000_net.*.auto/\
net/npacf0g0l0/tx_pause_frame_holdoff
0xc8
```

```
# ethtool --pause npacf0g0l0 tx on
```

```
# ethtool --show-pause npacf0g0l0
```

```
Pause parameters for npacf0g0l0:
Autonegotiate:    off
RX:         off
TX:         on
```

Note: All Ethernet settings listed in this section are not persistent across power cycles or server reboots or rsu. After power cycle or server reboot or rsu, the Intel FPGA PAC N3000 returns to default settings. The rsu command causes change in the PCIe B:D.F value.

8.4. Get Link Status and Statistics

To retrieve the line side link status, use the Linux `ethtool` and get link statistics using OPAE `fpgastats` command:

```
$ ethtool npacf0g0l0
```

Sample output:

```
Settings for npacf0g0l0:
Link detected: yes
```

```
$ ethtool -S npacf0g0l1
This command lists FPGA Ethernet counters
```

```
$ ethtool -S p7p1
This command lists XL710 Ethernet counters
```

Note: The Ethernet links between the FPGA and XL710 controllers is always up when the FPGA is being programmed.

The OPAE fpgastats command lists all FPGA Ethernet MAC counters on the Intel FPGA PAC N3000 specified by bus number. The fpgastats command is useful for detecting packet drops inside the FPGA because it provides both Ethernet wrapper 0 and 1 in an easily read format.

```
$ sudo fpgastats [-h] [--segment SEGMENT] [--bus BUS] [--device DEVICE] [--
function FUNCTION] [--clear] [--debug]

optional arguments:
  -h, --help            show this help message and exit
  --segment SEGMENT, -S SEGMENT
                        Segment number of PCIe device
  --bus BUS, -B BUS     Bus number of PCIe device
  --device DEVICE, -D DEVICE
                        Device number of PCIe device
  --function FUNCTION, -F FUNCTION
                        Function number of PCIe device
  --clear, -c           Clear statistics
  --debug, -d           Output debug information
```

You can clear Ethernet counts with the clear option:

```
$ sudo fpgastats -B 0x8a -c
```

Note: The XL710 controller does not support the OPAE `fpgastats` command.

Figure 16. Sample Output

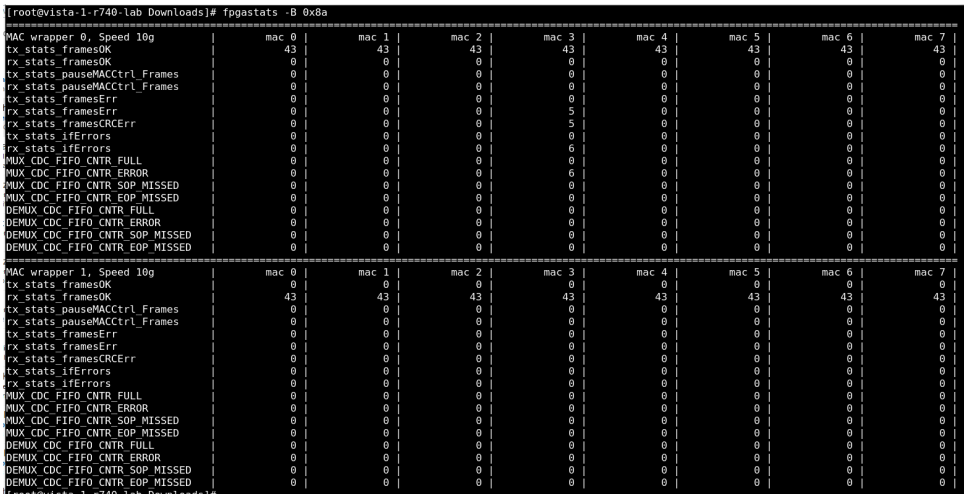


Table 10. MUX and DEMUX Counters

Name	Description
`tx_stats_*`	Refer to Low Latency Ethernet 10G MAC Intel FPGA IP User Guide.
`rx_stats_*`	
`CNTR_TX_*`	Refer to 25G Ethernet Intel Arria 10 FPGA IP User Guide or Low Latency 40- and 100-Gbps Ethernet MAC and PHY MegaCore Function User Guide.
`CNTR_RX_*`	
`MUX_CDC_FIFO_CNTR_FULL`	Drop counter on MUX. Increments when `mux_cdc_fifo` is full and there is no space for current packet.
`MUX_CDC_FIFO_CNTR_ERROR`	Counts packets marked with error on RX Avalon® streaming interface of MUX.
`MUX_CDC_FIFO_CNTR_SOP_MISSED`	Counts missed SOP on RX Avalon streaming interface of MUX.
`MUX_CDC_FIFO_CNTR_EOP_MISSED`	Counts missed EOP on RX Avalon streaming interface of MUX.
`DEMUX_CDC_FIFO_CNTR_FULL`	Drop counter on DEMUX. Increments when `demux_cdc_fifo` is full and there are no space for current packet.
`DEMUX_CDC_FIFO_CNTR_ERROR`	Counts packets marked with error on RX Avalon streaming interface of DEMUX.
`DEMUX_CDC_FIFO_CNTR_SOP_MISSED`	Counts missed SOP on RX Avalon streaming interface of DEMUX
`DEMUX_CDC_FIFO_CNTR_EOP_MISSED`	Counts missed EOP on RX Avalon streaming interface of DEMUX

For more information about MUX and DEMUX counters, refer to the Ethernet Interface.

9. Testing Network Loopback Using Data Plane Development Kit (DPDK)

Before starting Data Plane Development Kit (DPDK), you must perform configuration steps described in Configuring Ethernet Interfaces on page 38 as it relies on the FPGA being bound to OPAE driver (`pac_n3000_net`). While using DPDK, FPGA is unbound from this driver and bound to the `vfio-pci` driver.

Follow these steps to install DPDK for testing network loopback:

1. You must enable the Intel IOMMU driver on the host. Complete the following steps to enable the Intel IOMMU driver:
   a. Add **iommu=pt intel_iommu=on** to the GRUB_CMDLINE_LINUX entry by editing `/etc/default/grub`. For example:

   ```
   GRUB_CMDLINE_LINUX="crashkernel=auto rd.lvm.lv=rhel/root rd.lvm.lv=rhel/\
   swap rhgb quiet pci=realloc intel_iommu=pt"
   ```

   For RHEL: Additionally, add **pci=realloc** to GRUB_CMDLINE_LINUX entry.
   b. GRUB reads its configuration from either the */boot/grub2/grub.cfg* file on traditional BIOS-based machines or from the */boot/efi/EFI/redhat/ grub.cfg* file on UEFI machines. Depending on your system, execute one of the instructions below as root:
      - BIOS based machine:

        ```
        grub2-mkconfig -o /boot/grub2/grub.cfg
        ```

      - UEFI based machine:

        ```
        grub2-mkconfig -o /boot/efi/EFI/redhat/ grub.cfg
        ```

        ```
        # grub2-mkconfig -o /boot/efi/EFI/redhat/
        Generating grub configuration file ...
        Found linux image: /boot/vmlinuz-3.10.0-957.el7.x86_64
        Found initrd image: /boot/initramfs-3.10.0-957.el7.x86_64.img
        Found linux image: /boot/vmlinuz-0-
        rescue-594cabaaf9a84c6ea0a5167c89ad916d
        Found initrd image: /boot/initramfs-0-
        rescue-594cabaaf9a84c6ea0a5167c89ad916d.img
        /usr/sbin/grub2-mkconfig: line 290: /boot/efi/EFI/redhat/: Is a
        directory
        ```

   c. Reboot the server to apply the new GRUB configuration file.
   d. To verify the GRUB update, run the following command:

   ```
   $ cat /proc/cmdline
   ```

The sample output below shows **intel_iommu=on** on the kernel command line.

```
BOOT_IMAGE=/vmlinuz-3.10.0-957.el7.x86_64 root=/dev/mapper/rhel-root ro
default_hugepagesz=1G hugepagesz=1G hugepages=64 hugepagesz=2M
hugepages=2048 nosoftlockup mce=ignore_ce audit=0
isolcpus=1-11,24-35,13-23,36-47 nohz_full=1-11,24-35,13-23,36-47
rcu_nocbs=1-11,24-35,13-23,36-47 pci=realloc intel_iommu=on iommu=pt
enforcing=0 crashkernel=auto rd.lvm.lv=rhel/root rd.lvm.lv=rhel/swap
rhgb quiet skew_tick=1
```

2. Install the required packages:

```
$    sudo yum install readline-devel libpcap libpcap-devel numactl-devel
```

You need to download these two extra Fedora packages:

- `libfdt-1.4.7-3.fc30.x86_64.rpm`
- `libfdt-devel-1.4.7-3.fc30.x86_64.rpm`

To install:

```
$ sudo rpm -i <RPM file>
```

To check installation:

```
$ rpm -qa | grep libfdt
```

3. Compile and bind drivers:
   a. Download DPDK code from DPDK community and checkout release 19.08.

```
$ git clone https://github.com/DPDK/dpdk.git
```

```
$ cd dpdk
```

```
$ git pull
```

```
Already up-to-date.
```

```
$ git reset --hard 31b798a6f08e9b333b94b8bb26910209aa810b73
```

```
HEAD is now at 31b798a build: avoid overlinking
```

```
$ wget https://patches.dpdk.org/series/6821/mbox/
```

```
Length: 179795 (176K) [text/plain]
Saving to: \u2018index.html\u2019

100%[==============>] 179,795      554KB/s   in 0.3s

(554 KB/s) - \u2018index.html\u2019 saved [179795/179795]
```

```
$ git am index.html
```

```
Applying: net/i40e: i40e support ipn3ke FPGA port bonding
Applying: raw/ifpga/base: add irq support
Applying: raw/ifpga/base: clear pending bit
Applying: raw/ifpga/base: add SEU error support
Applying: raw/ifpga/base: add device tree support
Applying: raw/ifpga/base: align the send buffer for SPI
Applying: raw/ifpga/base: add sensor support
Applying: raw/ifpga/base: introducing sensor APIs
Applying: raw/ifpga/base: update SEU register definition
Applying: raw/ifpga: add SEU error handler
```

```
Applying: raw/ifpga: add PCIe BDF devices tree scan
Applying: net/ipn3ke: remove configuration for i40e port bonding
Applying: raw/ifpga/base: add secure support
Applying: raw/ifpga/base: configure FEC mode
Applying: raw/ifpga/base: clean fme errors
Applying: raw/ifpga/base: add new API get board info
Applying: raw/ifpga: add lightweight fpga image support
Applying: raw/ifpga/base: add multiple cards support
```

```
$ vim config/common_linux
// modify CONFIG_RTE_LIBRTE_PMD_IFPGA_RAWDEV=n to
CONFIG_RTE_LIBRTE_PMD_IFPGA_RAWDEV=y
```

```
$ vim config/common_base
// modify CONFIG_RTE_LIBRTE_IPN3KE_PMD=n to
CONFIG_RTE_LIBRTE_IPN3KE_PMD=y
```

b. Build DPDK and export RTE_SDK path to point to dpdk directory:

Note: Ignore the message "*Build complete [x86_64-native-linuxapp-gcc] Installation cannot run with T defined and DESTDIR undefined*"

```
$    export RTE_SDK=$PWD
```

```
$    export RTE_TARGET=x86_64-native-linuxapp-gcc
```

```
$    make config T=x86_64-native-linuxapp-gcc
```

```
$    make install -j8 T=x86_64-native-linuxapp-gcc
```

Note: $RTE_SDK points to the extracted dpdk source location.

c. Bind FPGA and NIC to `vfio-pci` driver as shown below. The sample output below shows result for 2x2x25G configuration. The 8x10G configuration produces similar output with following exception:

Table 11. Output Differences

Configuration	Ports per Intel Ethernet Controller XL710-BM2 NIC	Device ID for Intel Ethernet Controller XL710-BM2 NIC Ports
2x2x25G	2	0d58
8x10G	4	0cf8

```
$    cd $RTE_SDK
```

Check the binding between driver and device with the following command:

```
$    ./usertools/dpdk-devbind.py --status-dev net
```

```
Sample output:
Network devices using DPDK-compatible driver
============================================
<none>
Network devices using kernel driver
===================================
0000:18:00.0 'NetXtreme BCM5720 Gigabit Ethernet PCIe 165f' if=em1
drv=tg3 unused=
0000:18:00.1 'NetXtreme BCM5720 Gigabit Ethernet PCIe 165f' if=em2
drv=tg3 unused= *Active*
0000:19:00.0 'NetXtreme BCM5720 Gigabit Ethernet PCIe 165f' if=em3
drv=tg3 unused=
0000:19:00.1 'NetXtreme BCM5720 Gigabit Ethernet PCIe 165f' if=em4
drv=tg3 unused=
0000:14:00.0 'Device 0d58' if=p1p1 drv=i40e unused=
```

```
0000:14:00.1 'Device 0d58' if=p1p2 drv=i40e unused=
0000:16:00.0 'Device 0d58' if=p1p3 drv=i40e unused=
0000:16:00.1 'Device 0d58' if=p1p4 drv=i40e unused=
Other Network devices
====================
0000:15:00.0 'Device 0b30' unused=intel_fpga_pci
```

Install the vfio-pci kernel driver:

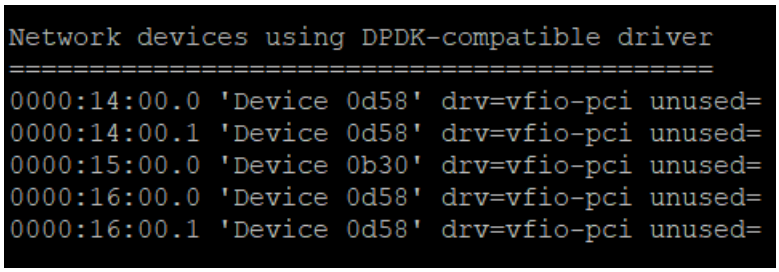
```
$    sudo modprobe vfio-pci
```

Bind the FPGA and FVL PFs to DPDK driver. Replace Bus:Device.Function with your output from above step. In case of 8x10G image, bind all 8 FVL B:D.F and the FPGA to vfio_pci:

```
$    sudo ./usertools/dpdk-devbind.py -b vfio-pci 14:00.0 \
14:00.1 15:00.0 16:00.0 16:00.1
```

Rerun `./usertools/dpdk-devbind.py --status-dev net` to check that FPGA and FVL PF's are bound to `vfio-pci` driver.

Figure 17. Sample Output



d. Reserve hugepages:

```
$ sudo mkdir -p /mnt/huge
```

```
$ sudo mount -t hugetlbfs nodev /mnt/huge
```

```
$ sudo sh -c "echo 2048 > /sys/kernel/mm/hugepages/\
hugepages-2048kB/nr_hugepages"
```

For more information on how the FPGA support is enabled in DPDK, refer to Data Plane Development Kit Reference Manual: Intel FPGA Programmable Acceleration Card N3000.

9.1. Test Using an External Traffic Generator

This test can be performed on 2x2x25G , 8x10G and 4x25G configuration.

1. Hardware Setup: Connect cable between QSFP port of Intel FPGA PAC N3000 and external traffic generator.

   Note: In 4x25G configuration, only one QSFP port is active per the configuration.

Figure 18. External Tester with 8x10G Network Configuration

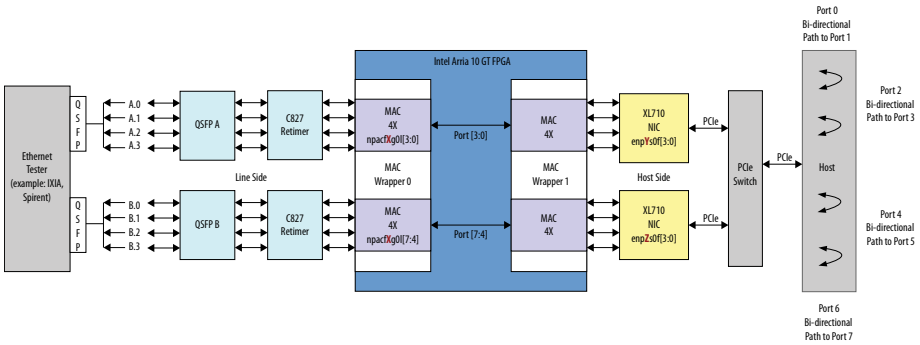


Figure 19. External Tester with 2x2x25G Network Configuration

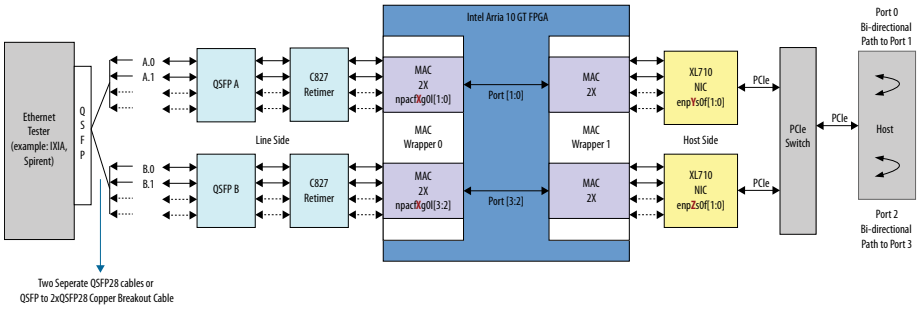
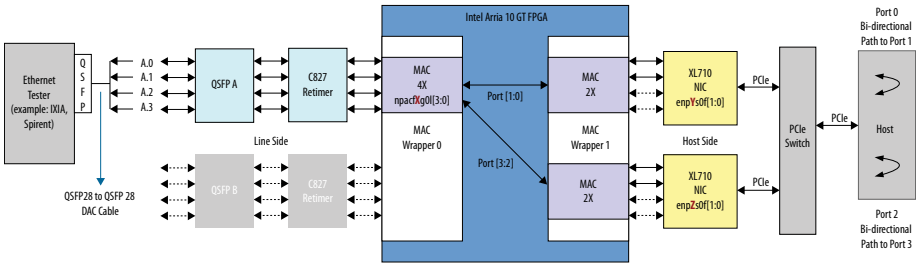


Figure 20. External Tester with 4x25G Network Configuration



2. Start the DPDK testpmd application.

   Note: Replace the FPGA B:D.F with values specific to your system.

```
$    cd $RTE_SDK
```

```
$    sudo ./x86_64-native-linuxapp-gcc/app/testpmd -l 0,1,2,3,4,5,6,7 -n 4 \
--vdev 'ifpga_rawdev_cfg0,ifpga=15:00.0,port=0' -- -i --no-numa
```

   Now, start traffic from external traffic generator:

```
testpmd>    start
```

```
testpmd>    show port stats all
```

   Sample output for 2x2x25G configuration:

```
  ######################## NIC statistics for port 0
########################
  RX-packets: 0          RX-missed: 0          RX-bytes:  0
  RX-errors: 0
```

```
  RX-nombuf:  0
  TX-packets: 0          TX-errors: 0          TX-bytes:  0

  Throughput (since last show)
  Rx-pps:            0
  Tx-pps:            0

###############################################################################

  ######################## NIC statistics for port 1
########################
  RX-packets: 0          RX-missed: 0          RX-bytes:  0
  RX-errors: 0
  RX-nombuf:  0
  TX-packets: 0          TX-errors: 0          TX-bytes:  0

  Throughput (since last show)
  Rx-pps:            0
  Tx-pps:            0

###############################################################################

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

  Throughput (since last show)
  Rx-pps:            0
  Tx-pps:            0

###############################################################################

  ######################## NIC statistics for port 3
########################
  RX-packets: 0          RX-missed: 0          RX-bytes:  0
  RX-errors: 0
  RX-nombuf:  0
  TX-packets: 0          TX-errors: 0          TX-bytes:  0

  Throughput (since last show)
  Rx-pps:            0
  Tx-pps:            0

###############################################################################

  ######################## NIC statistics for port 4
########################
  RX-packets: 0          RX-missed: 0          RX-bytes:  0
  RX-errors: 0
  RX-nombuf:  0
  TX-packets: 0          TX-errors: 0          TX-bytes:  0

  Throughput (since last show)
  Rx-pps:            0
  Tx-pps:            0

###############################################################################

  ######################## NIC statistics for port 5
########################
  RX-packets: 0          RX-missed: 0          RX-bytes:  0
  RX-errors: 0
  RX-nombuf:  0
  TX-packets: 0          TX-errors: 0          TX-bytes:  0

  Throughput (since last show)
  Rx-pps:            0
  Tx-pps:            0
```

```
##############################################################################

  ######################## NIC statistics for port 6
########################
  RX-packets: 0          RX-missed: 0          RX-bytes:  0
  RX-errors: 0
  RX-nombuf:  0
  TX-packets: 0          TX-errors: 0          TX-bytes:  0

  Throughput (since last show)
  Rx-pps:            0
  Tx-pps:            0

##############################################################################

  ######################## NIC statistics for port 7
########################
  RX-packets: 0          RX-missed: 0          RX-bytes:  0
  RX-errors: 0
  RX-nombuf:  0
  TX-packets: 0          TX-errors: 0          TX-bytes:  0

  Throughput (since last show)
  Rx-pps:            0
  Tx-pps:            0

##############################################################################
```

Expected result: testpmd by default works in paired mode. In this mode, the the packet forwarding is between pairs of ports, for example: (0,1), (2,3).

- In 2x2x25G configuration: Traffic forwarding is between ports (0,1) and (2,3)
- In 8x10G configuration: Traffic forwarding is between (0,1), (2,3), (4,5), and (6,7)

List of the cores to run on:

In case of 2x2x25G or 4x25G, we have 4 XL710 ports and hence we assign 4 cores. While in case of 8x10G, we have 8 XL710 ports and hence we assign 8 cores:

```
-l <core list>
```

Optionally, you can choose to send traffic to specific ports of the NIC rather than all ports. For this, the ports have to be explicitly white listed using the `-w <XL710 Port BDF>`. For example: Below command shows that only XL710 ports 14:00.0,14:00.1 are white listed. The FPGA BDF must also be explicitly white listed in this case:

```
sudo ./x86_64-native-linuxapp-gcc/app/testpmd -l 1,3 -n 4 -w \
0000:14:00.0 -w \
0000:14:00.1 -w \
0000:15:00.0 --vdev 'ifpga_rawdev_cfg0,ifpga=15:00.0,port=0' \
-- -i --no-numa
```

Refer to the Testpmd Application User Guide to understand the Environment Abstraction Layer arguments to testpmd.

Table 12. Specific Arguments

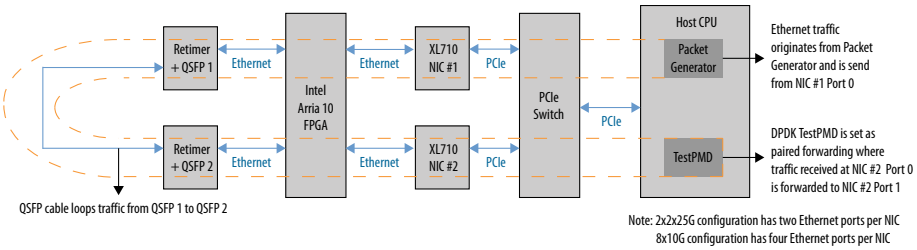
Arguments	Description	Capability Enabled
`-w <"FPGA BDF"> --vdev 'ifpga_rawdev_cfg0,ifpga=<"FPGA BDF">,port=0'`	The AFU name format is FPGA BDF|Port. Each FPGA can be divided into four blocks at most. Port identifies which FPGA block the AFU bitstream belongs to, but currently only Port 0 (Ethernet) is supported.	This triggers rawdev PMD driver to hotplug AFU to the IFPGA BUS.
`-w <XL710 port BDF>`	Whitelists the Intel XL710 NIC PF.	

9.2. Test Using a Packet Generator

This test can be performed on 2x2x25G and 8x10G configuration.

1. Hardware Setup: Connect loopback cable between the two QSFP ports.

Figure 21. Test Diagram



2. Download pktgen-3.7.1.zip into $RTE_SDK directory. Extract and build Pktgen. Run the following steps as root user:

```
$    cd $RTE_SDK
```

```
$    unzip pktgen-3.7.1.zip
```

```
$    cd pktgen-3.7.1
```

Note: If you are using RHEL OS, follow the additional setup instruction *Setup Prerequisites on Red Hat-based Systems* documented in `INSTALL.md file`.

```
$    export RTE_SDK=<DPDK Source PATH>
```

```
$    export RTE_TARGET=x86_64-native-linuxapp-gcc
```

```
$    export C_INCLUDE_PATH=/usr/local/src/lua-5.3.5/src
```

Note: Pktgen relies on all the three-environment variable defined above.

```
$    sudo -E make
```

3. Start a new terminal window. Use this new terminal window for running the DPDK testpmd application. Two ports from Intel XL710 #2 are assigned to testpmd. Port topology is set to paired. Thus, the forwarding is between pairs of ports. For example: (0,1); meaning anything received on port 0 will be forwarded to port 1.

Note: Replace XL710 B:D.F and FPGA B:D.F with values specific to your system.

```
$    cd $RTE_SDK
```

```
$    sudo  ./x86_64-native-linuxapp-gcc/app/testpmd -l 1,3 -n 4 -w  \
0000:14:00.0,switch_mode=IPN3KE_0@15:00.0_0 -w \
0000:14:00.1,switch_mode=IPN3KE_0@15:00.0_1 -w \
0000:15:00.0 --vdev 'ifpga_rawdev_cfg0,ifpga=15:00.0,port=0'  \
-- -i --no-numa --port-topology=paired
```

```
testpmd>    start
```

Now start traffic on pktgen (step 4).

Run the following command after starting pktgen (step 4):

```
testpmd>    show port stats all
```

Sample output:

```
You should see 10000 pkts received on port 0 and then transmitted from port
1 of XL7102. The “--port-topology=paired” causes forwarding between pairs of
ports (0,1) ie Traffic received on 0000:14:00.0 is forwarded to 0000:14:00.1
and then transmitted.

######## NIC statistics for port 0 ########
  RX-packets: 10000      RX-missed: 0          RX-bytes:  10200000
  RX-errors: 0
  RX-nombuf:  0
  TX-packets: 0          TX-errors: 0          TX-bytes:  0
  Throughput (since last show)
  Rx-pps:          496
  Tx-pps:            0
############################################
##########    NIC statistics for port 1 ###########
  RX-packets: 0          RX-missed: 0          RX-bytes:  0
  RX-errors: 0
  RX-nombuf:  0
  TX-packets: 10000      TX-errors: 0          TX-bytes:  10200000
  Throughput (since last show)
  Rx-pps:            0
  Tx-pps:          496
############################################
```

4. Start pktgen on terminal 1. Two ports from XL710 1 are assigned to pktgen. To understand the arguments, refer to the EAL Commandline Options.

   Note: Replace XL710 B:D.F with values specific to your system.

   Refer to Troubleshooting in DPDK on page 55 in case of an error.

   ```
   $    cd $RTE_SDK/pktgen_3.7.1
   ```

   ```
   $    sudo ./app/x86_64-native-linuxapp-gcc/pktgen -l 0,2,4 -n 4 \
   --proc-type primary --log-level 7 --file-prefix pg -w 0000:16:00.0 \
   -w 0000:16:00.1 -- -T -P -m 2.0 -m 4.1 -f themes/black-yellow.theme
   ```

   Configure the pkt size and pkt count:

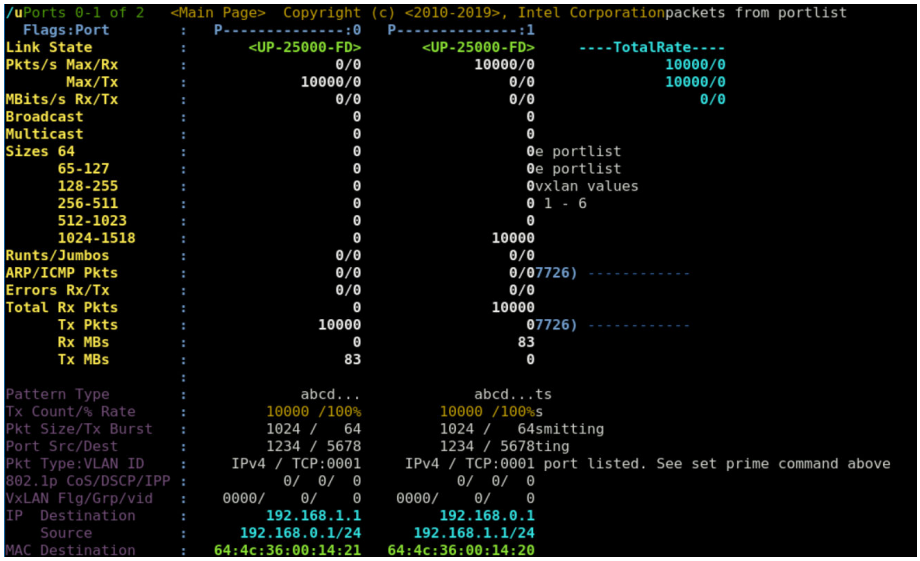
   ```
   $    set all size 1024
   ```

   ```
   $    set all count 10000
   ```

   ```
   $    start 0
   ```

   Sample output: You should see 10000 pkt transmitted from port 0 and received on port 1 of XL7101.

Figure 22. Sample Output



Note: Link status in pktgen tool may show down, this can be ignored.

Refer to the Testpmd Application User Guide to understand the Environment Abstraction Layer arguments to testpmd.

Table 13. Specific Arguments

Arguments	Description	Capability Enabled
`-w <"FPGA BDF"> --vdev 'ifpga_rawdev_cfg0,ifpga=<"FPGA BDF">,port=0'`	The AFU name format is FPGA BDF|`Port`. Each FPGA can be divided into four blocks at most. `Port` identifies which FPGA block the AFU bitstream belongs to, but currently only Port 0 (Ethernet) is supported.	This triggers rawdev PMD driver to hotplug AFU to the IFPGA BUS.
`-w <XL710 port BDF>`	Whitelists the Intel XL710 NIC PF.	

9.3. Troubleshooting in DPDK

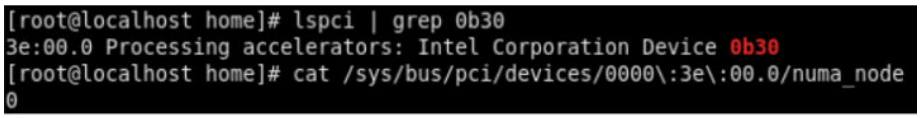
If you see error as below while starting pktgen, follow the listed steps:

```
Port  0: Link Up - speed 25000 Mbps - full-duplex <Enable promiscuous mode>
Port  1: Link Up - speed 25000 Mbps - full-duplex <Enable promiscuous mode>

RX processing lcore:   1 rx:  1 tx:  0
PANIC in pktgen_main_rx_loop():
*** port 0 socket ID 0 has different socket ID for lcore 1 socket ID 1
7:  [/lib64/libc.so.6(clone+0x6d) [0x7ff79c354ead]]
6: [/lib64/libpthread.so.0(+0x7dd5) [0x7ff79c62bdd5]]
5: [./app/x86_64-native-linuxapp-gcc/pktgen(eal_thread_loop+0x1d4) [0x583594]]
4: [./app/x86_64-native-linuxapp-gcc/pktgen(pktgen_launch_one_lcore+0xa7)
[0x4a8997]]
3: [./app/x86_64-native-linuxapp-gcc/pktgen() [0x4a1b1f]]
2: [./app/x86_64-native-linuxapp-gcc/pktgen(__rte_panic+0xb8) [0x46b941]]
1: [./app/x86_64-native-linuxapp-gcc/pktgen(rte_dump_stack+0x1a) [0x58935a]]
        Aborted (core dumped)
```

1. Find the CPU (NUMA node/socket) connected to the Intel FPGA PAC N3000.

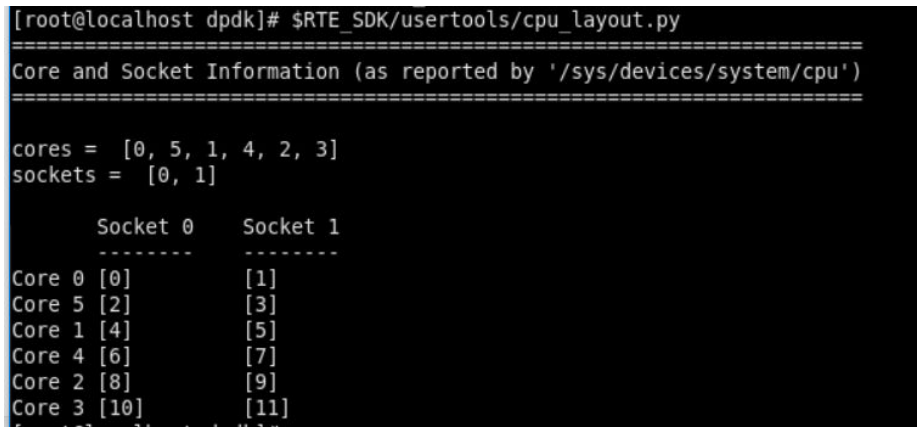
Figure 23. Example



2. Run command to find the CPU to core mapping.

```
$    RTE_SDK/usertools/cpu_layout.py
```

Figure 24. Output



This shows that all even cores [0, 2, 4, 6, 8, 10] are bound to socket 0 and all odd cores [1, 3, 5, 7, 9, 11] are bound to socket 1.

Note: The NUMA node is same as the socket.

Since ports in this setup are bound to socket 0, even cores are used as lcores in the below command. Update the lcore in the command as per your system configuration.

```
$  ./app/x86_64-native-linuxapp-gcc/pktgen -l 0,2,4 -n 4 --proc-type primary \
--log-level 7 --file-prefix pg -w 0000:16:00.0 -w 0000:16:00.1 -- \
-T -P -m 2.0 -m 4.1 -f themes/black-yellow.theme
```

Note: First core is always used by the management thread. lcore 2 is used to handle rx and tx for port 0. lcore 4 is used to handle rx and tx for port 1.

9.4. Revert Back from DPDK to OPAE

In the following command, replace 15:00.0 with the appropriate B:D:F value that corresponds to the FPGA:

```
$ sudo rmmod vfio-pci
```

```
$ echo 0000:15:00.0 | sudo tee /sys/bus/pci/drivers/intel-fpga-pci/bind
```

```
$ sudo modprobe i40e
```

Bind the XL710 interfaces to i40e driver:

```
$ sudo ./usertools/dpdk-devbind.py -b i40e 14:00.0 \
14:00.1 16:00.0 16:00.1
```

10. Graceful Shutdown

10.1. Background

The Intel FPGA PAC N3000 provides protective circuitry that automatically shuts down key board power supplies in the event of critical board sensors surpassing the fatal thresholds. The critical board sensors are listed below:

Table 14. Critical Sensors

Sensor ID	Sensor	Upper Fatal Threshold	Upper Warning Threshold	Lower Fatal Threshold	Lower Warning Threshold
12	FPGA Core Temperature	100°C	90°C	X	X
13	Board Temperature	85°C	75°C	X	X
25	12V Aux Voltage	X	X	10.56 V	11.40 V
3	12 V Backplane Voltage	X	X	10.56 V	11.40 V

For more information about sensors, refer to the Intel FPGA Programmable Acceleration Card N3000 Board Management Controller User Guide.

Surpassing the fatal thresholds of the above four critical sensors causes the Intel FPGA PAC PCIe buses to shut off, which could lead to a server Fatal PCIe Surprise Link Down event.

Intel provides two methods to prevent the server Fatal PCIe Surprise Link Down event. These methods mask PCIe Advanced Error Reporting (AER) registers for the Intel FPGA PAC N3000 to avoid Surprise Link Down. If you are using DPDK and have unbound the OPAE FPGA driver, follow the method described under *Using DPDK*.

Note: Once the Intel FPGA PAC N3000 surpasses the fatal threshold of a critical sensor, a server power cycle is required to recover operation of the Intel FPGA PAC N3000.

10.2. Using OPAE

The **fpgad** is a service that can help you protect the server from crashing when the hardware reaches an upper non-recoverable or lower non-recoverable sensor threshold (also called as fatal threshold). The **fpgad** is capable of monitoring each of the 20 sensors reported by the Board Management Controller.

```
$ sudo fpgainfo bmc
```

For more information about sensors, refer to the Intel FPGA Programmable Acceleration Card N3000 Board Management Controller (BMC) User Guide.

Note: Qualified OEM server systems should provide the required cooling for your workloads. Therefore, using **fpgad** may be optional.

When the `opae-tools-extra-1.3.6-4.x86_64.rpm` package is installed, **fpgad** is placed in the OPAE binaries directory (default: `/usr/bin`). The configuration file `fpgad.cfg` is located at `/etc/opae`. The log file `fpgad.log` which monitors **fpgad** actions is located at `/var/lib/opae/`.

The **fpgad** periodically reads the sensor values and if the values exceed the warning threshold stated in the `fpgad.conf` or the hardware defined warning threshold, it masks the PCIe Advanced Error Reporting (AER) registers for the Intel FPGA PAC to avoid system reset.

Use the following command to start the **fpgad** service:

```
$ sudo systemctl start fpgad
```

The configuration file only includes the threshold setting for critical sensor 12V Aux Voltage (sensor 25) and 12 V Backplane Voltage (sensor 3). These sensors do not have a hardware defined warning threshold and hence **fpgad** relies on the configuration file. The other two critical sensor FPGA Core Temperature (sensor 12) and Board Temperature (sensor 13) have a hardware defined warning threshold and fatal threshold set to values mentioned in the above table. The **fpgad** uses this information to mask the PCIe AER register when the sensor reaches the warning threshold.

Snapshot of the `fpgad.cfg` file located at `/etc/opae/` which configures the sensor 12V Aux Voltage (sensor 25) is shown below:

```
"fpgad-vc": {
                        "configuration": {
                                "cool-down": 30,
                                "config-sensors-enabled": true,
                                "sensors": [
                                        {
                                                "id": 25,
                                                "low-warn": 11.40,
                                                "low-fatal": 10.56
                                        },
                                ]
                        },
                        "enabled": true,
                        "plugin": "libfpgad-vc.so",
                        "devices": [
                                [ "0x8086", "0x0b30" ],
                                [ "0x8086", "0x0b31" ]
                        ]
                }
```

You must create another entry below the 12V Aux Voltage entry for 12V Backplane Voltage (sensor 3). The updated configuration file should have the following entry:

```
"fpgad-vc": {
                        "configuration": {
                                "cool-down": 30,
                                "config-sensors-enabled": true,
                                "sensors": [
                                        {
                                                "id": 25,
                                                "low-warn": 11.40,
                                                "low-fatal": 10.56
                                        }
                                ]
                        },
                        "enabled": true,
                        "plugin": "libfpgad-vc.so",
```

```
                        "devices": [
                                [ "0x8086", "0x0b30" ],
                                [ "0x8086", "0x0b31" ]
                        ]
                },

"fpgad-vc": {
                        "configuration": {
                                "cool-down": 30,
                                "config-sensors-enabled": true,
                                "sensors": [
                                        {
                                                "id": 3,
                                                "low-warn": 11.40,
                                                "low-fatal": 10.56
                                        }
                                ]
                        },
                        "enabled": true,
                        "plugin": "libfpgad-vc.so",
                        "devices": [
                                [ "0x8086", "0x0b30" ],
                                [ "0x8086", "0x0b31" ]
                        ]
                }
```

You can monitor the log file to see if upper or lower warning threshold levels are hit. For example:

```
tail -f /var/lib/opae/fpgad.log | grep “sensor.*warning”
fpgad-vc: sensor 'FPGA Die Temperature' warning
```

You must take appropriate action to recover from this warning before the sensor value reaches upper or lower fatal limits. On reaching the warning threshold limit, the daemon masks the AER registers and the log file will indicate that the sensor is tripped.

Sample output: Warning message when the FPGA Core Temperature exceeds the upper warning threshold limit.

```
Ex: tail -f /var/lib/opae/fpgad.log
fpgad-vc: saving previous ECAP_AER+0x08 value 0x003ff030 for 0000:5d:00.0
fpgad-vc: saving previous ECAP_AER+0x14 value 0x000031c1 for 0000:5d:00.0
fpgad-vc: sensor 'FPGA Die Temperature' still tripped.
```

Sample output: Warning message when the voltage exceeds the lower warning threshold limit.:

```
fpgad-vc: sensor '12V AUX Voltage' warning.
fpgad-vc: saving previous ECAP_AER+0x08 value 0x00100000 for 0000:ae:00.0
fpgad-vc: saving previous ECAP_AER+0x14 value 0x00002000 for 0000:ae:00.0
fpgad-vc: sensor '12V AUX Voltage' still tripped.
fpgad-vc: sensor '12V AUX Voltage' still tripped.
```

If the upper or lower fatal threshold limit is reached, then a power cycle of server is required to recover the Intel FPGA PAC N3000. AER is unmasked by the **fpgad** after the sensor values are within the normal range which is above the lower warning or below the upper warning threshold.

Sample output when upper or lower fatal threshold is reached:

```
fpgad-vc: failed to read sensor xx
```

To stop **fpgad**:

```
$ sudo systemctl stop fpgad.service
```

To check status of **fpgad**:

```
$ sudo systemctl status fpgad.service
```

Optional: To enable **fpgad** to re-start on boot, execute

```
$ sudo systemctl enable fpgad.service
```

For a full list of `systemctl` commands, run the following command:

```
$ systemctl -h
```

10.3. Using DPDK

The Ifpga Rawdev driver continually monitors the critical sensors mentioned in Table 14 on page 57. If the critical sensor thresholds are exceeded while the DPDK software is running, then the DPDK software stops with an error message. For example: Run the testpmd with the server fan turned off.

```
 sudo ./x86_64-native-linuxapp-gcc/app/testpmd -l 0,1,2,3,4,5,6,7 -n 4 \
--vdev 'ifpga_rawdev_cfg0,ifpga=15:00.0,port=0' -- -i --no-numa

testpmd > set_surprise_link_check_aer(): Set AER, pls graceful shutdown
>>>>>>Set AER 0,0 3ff030,31c1
EAL: Error - exiting with code: 1
Cause: >>>>>>Graceful Shutdown
```

You must power cycle the server to ensure complete functionality of the board is restored.

11. Single Event Upset (SEU)

The Intel Manufacturing Single Event Upset (SEU) testing of Intel FPGA PAC N3000 provides the following results:

- SEU events do not induce latch-up in Intel FPGA PAC N3000.
- No SEU errors have been observed in hard CRC circuits and I/O registers.
- The cyclic redundancy check (CRC) circuit can detect all single-bit and multi-bit errors within the configuration memory.

SEU errors may occur in either of the two primary devices of the Intel FPGA PAC N3000:

- **Intel MAX 10 SEU**: An SEU event is detected by Error Detection CRC (EDCRC) circuitry in Intel MAX 10. The CRC function implemented in Intel FPGA PAC N3000 enables CRC status to be reported to FPGA via a dedicated CRC_ERROR pin. The CRC error output is continually polled in an interval between 5.5 and 13.6 seconds.

  If a CRC error was detected, it is not permanently logged if subsequent polls do not detect an error.

  When FPGA detects a CRC_ERROR assertion, it is logged in the FPGA internal register RAS_CATFAT_ERR. The system register bits are not reliable after an SEU event, therefore a power cycle is required.
- **FPGA SEU**: In FPGA device, the contents of the configuration RAM (CRAM) bits can be affected by soft SEU errors. The hardened on-chip EDCRC circuitry auto-detects CRC errors. Corrections of CRAM upsets are not supported. Therefore, if SEU errors are detected, FPGA reset is required.

11.1. OPAE Handling of SEU

An OPAE tool `fpgad` monitors for SEU events and records any such occurrence in the log file `/var/lib/opae/fpgad.log`

To start `fpgad`:

```
sudo systemctl start fpgad
```

- Intel MAX 10 SEU:

  The `fpgad.log` file would show the below output:

  ```
  tail -f /var/lib/opae/fpgad.log
  fpgad-vc: failed to get value object for sensor38.
  fpgad-vc: poll count = 1
  ```

```
fpgad-vc: SEU error occurred on bmc @ 0000:b2:00.0
fpgad-vc: failed to get value object for sensor15.
fpgad-vc: failed to get value object for sensor38.
```

Ignore the message: *failed to get value object for sensor*. Sensor 15 and sensor 38 indicate QSFP temperature. This failure indicates that the QSFP cable was not plugged in.

- FPGA SEU:

  The `fpgad.log` file would show the below output:

  ```
  tail -f /var/lib/opae/fpgad.log
  fpgad-vc: failed to get value object for sensor38.
  fpgad-vc: poll count = 1
  fpgad-vc: SEU error occurred on fpga @ 0000:b2:00.0
  fpgad-vc: failed to get value object for sensor15.
  fpgad-vc: failed to get value object for sensor38.
  ```

  Ignore the message: *failed to get value object for sensor*. Sensor 15 and sensor 38 indicate QSFP temperature. This failure indicates that the QSFP cable was not plugged in.

To recover from both Intel MAX 10 and FPGA SEU event, reset the Intel FPGA PAC N3000 using the following command:

```
$ rsu bmcimg <PCI BDF>
```

For testing your system's response to an SEU event, Intel provides a mechanism to inject an error which will be logged by `fpgad` similar to the way an SEU event is logged.

1. Start `fpgad`

   ```
   $ sudo systemctl start fpgad
   ```

2. Terminal 2: monitor fpgad.log

   ```
   $ sudo tail -f /var/lib/opae/fpgad.log
   ```

3. Terminal 1: Inject error

   ```
   $ sudo sh -c "echo 1 > /sys/class/fpga/intel-fpga-dev.0/\
   intel-fpga-fme.0/errors/inject_error"
   ```

   Sample output:

   ```
   fpgad-vc: error interrupt event received.
   fpgad-vc: poll count = 1.
   fpgad-vc: detect inject_error 0x1 @ 0000:15:00.0
   fpgad-vc: detect catfatal_errors 0x800 @ 0000:15:00.0
   ```

   Note: `poll count =1`: indicates an error was detected.

4. To clear the error injection:

   ```
   $ sudo sh -c "echo 0 > /sys/class/fpga/intel-fpga-dev.0/intel-fpga-fme.0/
   errors/inject_error"
   ```

11.2. DPDK Handling of SEU

When the SEU errors occur, an interrupt is generated in DPDK IFPGA Rawdev driver which supports FPGA management. The interrupt handler is defined in the file $RTE_SDK/drivers/raw/ifpga/ifpga_rawdev.c as:

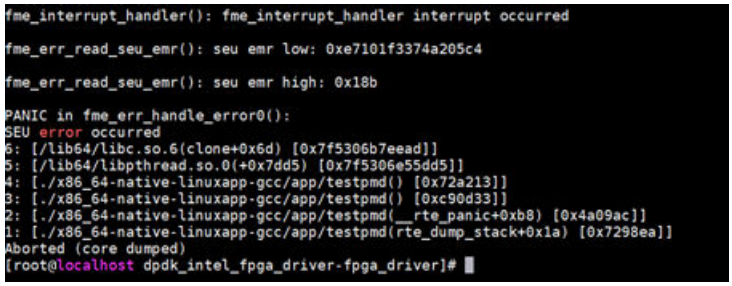
```
static void
fme_interrupt_handler(void *param)
{
        struct opae_manager *mgr = (struct opae_manager *)param;
        IFPGA_RAWDEV_PMD_INFO("%s interrupt occurred\n", __func__);
        fme_err_handle_error0(mgr);
        fme_err_handle_nonfaterror(mgr);
        fme_err_handle_catfatal_error(mgr);
}
```

The function `fme_err_handle_catfatal_error(mgr)` handles the SEU error by causing a panic through `rte_panic()` function call. The implementation of this interrupt handler is provided as reference code and can be customized.

When a SEU event occurs, you receive a panic message while running the DPDK application using the following command:

```
$ sudo ./x86_64-native-linuxapp-gcc/app/testpmd -l 0,1,2,3,4,5,6,7 -n 4 \
--vdev 'ifpga_rawdev_cfg0,ifpga=15:00.0,port=0' -- -i --no-numa
```

Figure 25. Panic Message



To recover from the SEU panic which is caused by the reference SEU event interrupt handler, follow these steps:

1. Unbind from vfio driver:

   ```
   sudo rmmod vfio-pci
   ```

2. Rebind to OPAE driver:

   ```
   modprobe intel-fpga-pci
   echo 0000:BB:DD.F > /sys/bus/pci/drivers/intel-fpga-pci/bind
   ```

3. Re-configure the FPGA:

   a. Extract the `N3000_supplemental_files.zip` which is provided as part of the Acceleration Stack Installer:

   ```
   $ unzip N3000_supplemental_files.zip
   ```

   ```
   $ cd N3000_supplemental_files/
   ```

b. Find PCIe Root Port:

```
$ chmod +x find_RP.sh
```

Sample output:

```
0000:ae:00.0
0000:af:00.0
0000:b0:09.0
0000:b2:00.0 -> intel-fpga-dev.0
```

The first entry in the list is the PCIe Root port. The last entry is the Intel FPGA PAC N3000.

c. Record the RP AER value:

```
$ sudo setpci -s ae:00.0 ECAP_AER+0x08.L
00210000
```

```
$ sudo setpci -s ae:00.0 ECAP_AER+0x14.L
000031c1
```

Note: Your AER values may differ from the above responses.

d. Disable AER:

```
$ sudo setpci -s <RP BDF> ECAP_AER+0x08.L=0xffffffff
```

```
$ sudo setpci -s <RP BDF> ECAP_AER+0x14.L=0xffffffff
```

e. Trigger re-configure FPGA:

```
$ sudo rsu fpga b2:00.0
```

f. Enable AER , use the values obtained from step 3b:

```
$ sudo setpci -s <RP BDF> ECAP_AER+0x08.L=0x00210000
```

```
$ sudo setpci -s <RP BDF> ECAP_AER+0x14.L=0x000031c1
```

4. Install `vfio-pci` driver.

```
$ sudo modprobe vfio-pci
```

5. Bind all ports to the `vfio-pci` driver and restart the DPDK application.

12. Document Revision History for Intel Acceleration Stack User Guide: Intel FPGA PAC N3000

Document Version	Intel Acceleration Stack Version	Changes
2021.06.14	1.1	Added a note about the RSU functionality in section: *Installing the Intel FPGA PAC N3000*.
2020.08.17	1.1	Clarified the Bitstream ID for Intel FPGA PAC N3000 in section: *Identify the Intel MAX 10 Version on your Intel FPGA PAC N3000*.
2020.05.28	1.1	• Updated the following sections: — *Using fpgasupdate* — *Update the Intel XL710 Firmware* — *Get Link Status and Statistics* • Added a note in the following sections: — *Using fpgainfo* — *Upgrade your Intel FPGA PAC N3000 with Production Version of BMC and Intel Arria 10 Image*
2020.02.12	1.1	Updated the following sections: • *Identify the Intel MAX 10 Version on your Intel FPGA PAC N3000* • *Installing the Intel XL710 Driver* • *Graceful Shutdown* • *Upgrade your Intel FPGA PAC N3000 with Production Version of BMC and Intel Arria 10 Image*
2019.11.25	1.1	Initial release.

A. Troubleshooting

A.1. If OPAE installation verification fails, how to install OPAE manually?

1. Download either Acceleration Stack runtime (rte) or development (dev) installer.
2. Extract the OPAE packages:

```
$ ./n3000-1.3.6-*-setup.sh extract
```

```
$ cp opae-intel-fpga*.rpm n3000-1.3.6-rte/opae/
```

```
$ cd n3000-1.3.6-rte/opae/
```

3. Remove any previously installed OPAE:

```
$ sudo yum remove opae*
```

4. Manually install OPAE driver:

```
$ sudo yum install opae-intel-fpga*.rpm
```

5. Verify installation:

```
$ lsmod | grep fpga
```

```
ifpga_sec_mgr 16384 1 intel_max10
intel_fpga_fme 65536 0
intel_fpga_afu 32768 0
fpga_mgr_mod 16384 1 intel_fpga_fme
intel_fpga_pci 24576 2 intel_fpga_fme,intel_fpga_afu
```

If the verification fails, analyze the message log of kernel installation for hints to fix the issue.

6. Manually install OPAE libraries and tools:

```
$ sudo yum install opae*.rpm
```

7. Verify installation pf OPAE libraries and tools:

```
opae-tools-extra-1.3.6-4.x86_64
opae-one-time-update-n3000-25G-1.3.6-6.noarch
opae.admin-1.0.2-3.noarch
opae-tools-1.3.6-4.x86_64
opae-intel-fpga-driver-2.0.1-6.x86_64
opae-devel-1.3.6-4.x86_64
opae-super-rsu-n3000-2x2x25G-1.3.6-6.noarch
opae-libs-1.3.6-4.x86_64
opae.pac_sign-1.0.2-3.x86_64
```

B. Upgrade your Intel FPGA PAC N3000 with Production Version of BMC and Intel Arria 10 Image

Based on the current version loaded on your Intel FPGA PAC N3000, follow the appropriate section of this appendix to upgrade your Intel FPGA PAC N3000.

In the commands below, replace <25G or 10G> with appropriate directory which means use 25G or 10G based on the configuration installer selected and installed.

```
ls -l /usr/share/opae/n3000/one-time-update/
```

Note: The upgrade process updates all Intel FPGA PAC N3000 on the server simultaneously. Ensure all Intel FPGA PAC N3000 on the server have the same XL710 device ID. Also, ensure that the listed steps are not interrupted.

Delete all old remnant files:

```
$ sudo rm -rf /usr/share/opae/n3000/one-time-update/2x2x25G/
```

```
$ sudo rm -rf /usr/share/opae/n3000/one-time-update/8x10G/
```

Note: These upgrades erase the Static Region (SR) root entry hash and any CSK cancellation IDs previously programmed in the flash of the Intel FPGA PAC N3000.

Remember:

- Stop any service or daemon accessing the FPGA or XL710 before updating the Intel FPGA PAC N3000 such as `fpgad`.
- PLDM requests may return stale data. Avoid Host PLDM requests.
- Ensure cooling requirements are met. The server can reboot if the FPGA Core temperature exceeds 95°C. For more information, refer to Cooling Requirements on page 7.

Tip: Before you proceed with upgrade, ensure that the FPGA Die Temperature is below 80°C using the following command:

```
sudo fpgainfo bmc
```

If it is higher than the threshold value, increase the fan speed to improve thermal condition.

B.1. Upgrading from 1.1 Beta to Production Version

The production version of BMC is signed with a release key which is different from the key used in beta version. Hence, the upgrade process involves performing a rollback to non-ROT BMC and Intel Arria 10 image and there after running OTSU to upgrade the BMC and Intel Arria 10 on the board to production version.

There are two scenarios while updating your Intel FPGA PAC N3000 from Acceleration Stack 1.1 Beta to Production version:

- SR Root entry hash programmed on the Intel FPGA PAC N3000
- SR Root entry hash not programmed on the Intel FPGA PAC N3000

B.1.1. Root Entry Hash Programmed

1. Sign the provided Intel Arria 10 image located at `/usr/share/opae/n3000/one-time-update/<25G or 10G>/chip_rsu-*-user-rollback-unsigned.bin` with valid code signing key and SR root key using the PACSign and name the output file as `chip_rsu-*-user-rollback-signed.bin`.
2. Copy the signed Intel Arria 10 image to location: `/usr/share/opae/n3000/one-time-update/*/`

   ```
   $ cp chip_rsu-*-user-rollback-signed.bin /usr/share/opae/n3000/\
   one-time-update/<25G or 10G>/
   ```

3. Reflect the signed Intel Arria 10 image name in the rollback manifest file:

   ```
   $ vim /usr/share/opae/n3000/one-time-update/<25G or 10G>/rollback-*.json
   ```

   Change the `filename` field under flash to reflect the signed bitstream name, for example:

   ```
   "flash": [
           {
               "enabled": true,
               "filename": "chip_rsu-8x10G-user-rollback-signed.bin",
               "force": false,
               "secure": true,
               "timeout": "45m",
               "type": "user",
               "version": "0x0021064001020134"
           },
   ```

4. Rollback the Intel MAX 10 and Intel Arria 10 image to allow upgrade:

   ```
   $ sudo super-rsu /usr/share/opae/n3000/one-time-update/<25G or 10G>/\
   rollback-*.json --with-rsu
   ```

   ```
   $ sudo fpgainfo fme
   ```

   Note: The Intel MAX 10 build version will be 111.2.13 after performing rollback.

   ```
   $ ls -l /sys/class/fpga/intel-fpga-dev.*/intel-fpga-fme.*/\
   spi-altera.*.auto/spi_master/spi*/spi*/intel-generic-qspi.*.auto/
   ```

   This sysfs entry must exist.

5. Run the One-Time Secure Update (OTSU):

   ```
   $ sudo fpgaotsu /usr/share/opae/n3000/one-time-update/<25G or 10G>/\
   otsu-*.json --rsu
   ```

   If OTSU fails, run the `fpgainfo fme` to find out the Intel MAX 10 build version and take appropriate action as stated:

Intel MAX 10 Build Version	Action
D.111.2.13	Repeat OTSU
D.2.0.6	Run command: `sudo super-rsu /usr/share/opae/n3000/super-rsu/<2x2x25G or 8x10G or 4x25G>/super-rsu-*.json --with-rsu`

6. To verify successful OTSU:

```
sudo fpgaotsu /usr/share/opae/n3000/one-time-update/<25G or 10G>/\
otsu-*.json --verify
```

At this point, the Intel FPGA PAC N3000 will have the following Intel Arria 10 image, Intel MAX 10 NIOS FW and Intel MAX 10 Build versions.

Configuration	User Partition Image	Bitstream ID	PR interface ID	Intel MAX 10 NIOS FW	Intel MAX 10 BUILD
2x2x25G	4x25G	0x2300110010309	f3c99413-5081-4aad-bced-07eb84a6d0bb	D.2.0.19	D.2.0.6
4x25G					
8x10G	8x10G	0x2300010010309	901dd697-ca79-4b05-b843-8138cefa2846	D.2.0.19	D.2.0.6

Important: The 2x2x25G or 4x25G Configuration Installer loads the FPGA flash user partition with 4x25G Intel provided factory test image and loads factory partition with 2x2x25G Intel provided factory test image. The 8x10 Configuration Installer loads both the user and factory FPGA flash partitions with 8x10G Intel provided factory test image.

7. Install the PV 1.1 Patch.

B.1.2. Root Entry Hash Not Programmed

1. Rollback the Intel MAX 10 and Intel Arria 10 image to allow update:

```
$ sudo super-rsu /usr/share/opae/n3000/one-time-update/<25G or 10G>/\
rollback-*.json --with-rsu
```

```
$ sudo fpgainfo fme
```

Note: The Intel MAX 10 build version will be 111.2.13 after performing rollback.

```
$ ls -l /sys/class/fpga/intel-fpga-dev.*/intel-fpga-fme.*/\
spi-altera.*.auto/spi_master/spi*/spi*/intel-generic-qspi.*.auto/
```

This sysfs entry must exist.

2. Run the One-Time Secure Update (OTSU):

```
$ sudo fpgaotsu /usr/share/opae/n3000/one-time-update/*/otsu-*.json --rsu
```

If OTSU fails, run the `fpgainfo fme` to find out the Intel MAX 10 build version and take appropriate action as stated:

Intel MAX 10 Build Version	Action
D.111.2.13	Repeat OTSU
D.2.0.6	Run command: `sudo super-rsu /usr/share/opae/n3000/super-rsu/<2x2x25G, 4x25G or 8x10G>/super-rsu-*.json --with-rsu`

3. To verify successful OTSU:

```
sudo fpgaotsu /usr/share/opae/n3000/one-time-update/<25G or 10G>/\
otsu-*.json --verify
```

At this point, the Intel FPGA PAC N3000 will have the following Intel Arria 10 image, Intel MAX 10 NIOS FW and Intel MAX 10 Build versions.

Configuration	User Partition Image	Bitstream ID	PR interface ID	Intel MAX 10 NIOS FW	Intel MAX 10 BUILD
2x2x25G	4x25G	0x2300110010309	f3c99413-5081-4aad-bced-07eb84a6d0bb	D.2.0.19	D.2.0.6
4x25G					
8x10G	8x10G	0x2300010010309	901dd697-ca79-4b05-b843-8138cefa2846	D.2.0.19	D.2.0.6

Important: The 2x2x25G or 4x25G Configuration Installer loads the FPGA flash user partition with 4x25G Intel provided factory test image and loads factory partition with 2x2x25G Intel provided factory test image. The 8x10 Configuration Installer loads both the user and factory FPGA flash partitions with 8x10G Intel provided factory test image.

4. Install the PV 1.1 Patch.

B.2. Upgrading from 1.1 Alpha-2 or Older to Production Version

1. Load temporary Intel MAX 10 image to allow update:

```
$ sudo super-rsu /usr/share/opae/n3000/one-time-update/\
<25G or 10G>/super-rsu.json --with-rsu
```

```
[2019-11-04 11:11:29,455] [DEBUG   ] [MAINTHREAD  ] - FOUND FPGA OBJECTS:
['/SYS/CLASS/FPGA/INTEL-FPGA-DEV.0']
[2019-11-04 11:11:29,458] [DEBUG   ] [MAINTHREAD  ] - FOUND DEVICE AT
0000:08:00.0 -TREE IS
 [PCI_ADDRESS(0000:00:03.0), PCI_ID(0X8086, 0X2F08)]
    [PCI_ADDRESS(0000:03:00.0), PCI_ID(0X10B5, 0X8747)]
        [PCI_ADDRESS(0000:04:08.0), PCI_ID(0X10B5, 0X8747)]
            [PCI_ADDRESS(0000:05:00.0), PCI_ID(0X8086, 0X0D58)]
            [PCI_ADDRESS(0000:05:00.1), PCI_ID(0X8086, 0X0D58)]
        [PCI_ADDRESS(0000:04:09.0), PCI_ID(0X10B5, 0X8747)]
            [PCI_ADDRESS(0000:08:00.0), PCI_ID(0X8086, 0X0B30)]
        [PCI_ADDRESS(0000:04:10.0), PCI_ID(0X10B5, 0X8747)]
            [PCI_ADDRESS(0000:0A:00.0), PCI_ID(0X8086, 0X0D58)]
            [PCI_ADDRESS(0000:0A:00.1), PCI_ID(0X8086, 0X0D58)]
        [PCI_ADDRESS(0000:04:11.0), PCI_ID(0X10B5, 0X8747)]
            [PCI_ADDRESS(0000:0D:00.0), PCI_ID(0X8086, 0X0B32)]

[2019-11-04 11:11:29,462] [WARNING ] [MAINTHREAD  ] - UPDATE STARTING.
PLEASE DO NOT INTERRUPT.
[2019-11-04 11:11:29,462] [DEBUG   ] [MAINTHREAD  ] - BMC_IMG - CURRENT_REV:
"D", FLASH_REV: "D"
[2019-11-04 11:11:29,463] [DEBUG   ] [MAINTHREAD  ] - BMC_IMG IS BEING FORCE
FLASHED
```

```
[2019-11-04 11:11:29,464] [DEBUG   ] [MAINTHREAD  ] - BMC_IMG VERSIONS NOT
EQUAL (SYSTEM:1.0.13 != MANIFEST:111.2.13)
[2019-11-04 11:11:29,464] [DEBUG   ] [MAINTHREAD  ] - [08:00.0] UPDATE
TIMEOUT SET TO: 1200.0
[2019-11-04 11:11:29,464] [DEBUG   ] [08:00.0     ] - UPDATE OF BOARD AT
[PCI_ADDRESS(0000:08:00.0), PCI_ID(0X8086, 0X0B30)] STARTED
[2019-11-04 11:11:29,464] [DEBUG   ] [MAINTHREAD  ] - MAX TIMEOUT SET TO:
0:20:00
[2019-11-04 11:11:29,464] [DEBUG   ] [08:00.0     ] - STARTING TASK:
FPGAFLASH BMC_IMG /USR/SHARE/OPAE/N3000/ONE-TIME-UPDATE/25G/Intel MAX
10_SYSTEM_REVD_DUAL_V111.2.13_TEMPORARY_DEFAULT_FPGA_DIE_CFM0_AUTO.RPD
0000:08:00.0
USING /DEV/MTD0
2019-11-04 11:11:30.651141 REVERSING BITS
2019-11-04 11:11:31.227369 ERASING 0X000A8000 BYTES STARTING AT 0X000B8000
2019-11-04 11:11:31.233072 WRITING 0X000A8000 BYTES TO 0X000B8000
2019-11-04 11:12:20.285000 ACTUAL BYTES WRITTEN 0X11B000 - 0XB8000 = 0X63000
2019-11-04 11:12:20.285701 READING 0X00063000 BYTES FROM 0X000B8000
2019-11-04 11:12:21.473206 VERIFYING FLASH
2019-11-04 11:12:21.473641 FLASH SUCCESSFULLY VERIFIED
[2019-11-04 11:12:21,553] [DEBUG   ] [08:00.0     ] - TASK COMPLETED IN
0:00:52.087794
[2019-11-04 11:12:22,579] [INFO    ] [MAINTHREAD  ] -
[[PCI_ADDRESS(0000:08:00.0), PCI_ID(0X8086, 0X0B30)]] PERFORMING RSU
OPERATION
[2019-11-04 11:12:22,634] [INFO    ] [MAINTHREAD  ] -
[[PCI_ADDRESS(0000:00:03.0), PCI_ID(0X8086, 0X2F08)]] REMOVING DEVICE FROM
PCIE BUS
[2019-11-04 11:12:22,634] [DEBUG   ] [MAINTHREAD  ] - REMOVING DEVICE AT
0000:00:03.0
[2019-11-04 11:12:22,634] [INFO    ] [MAINTHREAD  ] - WAITING 10 SECONDS FOR
BOOT
[2019-11-04 11:12:32,644] [INFO    ] [MAINTHREAD  ] - RESCANNING PCIE
BUS: /SYS/DEVICES/PCI0000:00/PCI_BUS/0000:00
[2019-11-04 11:12:32,763] [INFO    ] [MAINTHREAD  ] - REDISCOVERING BOARDS
TO VERIFY AFTER RSU
[2019-11-04 11:12:32,764] [DEBUG   ] [MAINTHREAD  ] - FOUND FPGA OBJECTS:
['/SYS/CLASS/FPGA/INTEL-FPGA-DEV.0']
[2019-11-04 11:12:32,766] [DEBUG   ] [MAINTHREAD  ] - FOUND DEVICE AT
0000:06:00.0 -TREE IS
 [PCI_ADDRESS(0000:00:03.0), PCI_ID(0X8086, 0X2F08)]
    [PCI_ADDRESS(0000:03:00.0), PCI_ID(0X10B5, 0X8747)]
        [PCI_ADDRESS(0000:04:08.0), PCI_ID(0X10B5, 0X8747)]
            [PCI_ADDRESS(0000:05:00.0), PCI_ID(0X8086, 0X0D58)]
            [PCI_ADDRESS(0000:05:00.1), PCI_ID(0X8086, 0X0D58)]
        [PCI_ADDRESS(0000:04:09.0), PCI_ID(0X10B5, 0X8747)]
            [PCI_ADDRESS(0000:06:00.0), PCI_ID(0X8086, 0X0B30)]
        [PCI_ADDRESS(0000:04:10.0), PCI_ID(0X10B5, 0X8747)]
            [PCI_ADDRESS(0000:07:00.0), PCI_ID(0X8086, 0X0D58)]
            [PCI_ADDRESS(0000:07:00.1), PCI_ID(0X8086, 0X0D58)]
        [PCI_ADDRESS(0000:04:11.0), PCI_ID(0X10B5, 0X8747)]
            [PCI_ADDRESS(0000:08:00.0), PCI_ID(0X8086, 0X0B32)]

[2019-11-04 11:12:32,767] [DEBUG   ] [MAINTHREAD  ] - BMC_IMG - CURRENT_REV:
"D", FLASH_REV: "D"
[2019-11-04 11:12:32,767] [DEBUG   ] [MAINTHREAD  ] - SELF-TEST DISABLED IN
CONFIGURATION
[2019-11-04 11:12:32,767] [INFO    ] [MAINTHREAD  ] - SUPER_RSU.PYC UPDATE
COMPLETED IN: 0:01:03.305579
[2019-11-04 11:12:32,768] [INFO    ] [MAINTHREAD  ] - SUPER-RSU EXITING WITH
CODE '0'
```

```
$ sudo fpgainfo fme
```

The Intel MAX 10 build version will be 111.2.13.

```
$ ls -l /sys/class/fpga/intel-fpga-dev.*/intel-fpga-fme.*/\
spi-altera.*.auto/spi_master/spi*/spi*/intel-generic-qspi.*.auto
```

This sysfs entry must exist.

2. Run the One-Time Secure Update (OTSU). This process takes approximately 40 minutes.

```
$ sudo fpgaotsu /usr/share/opae/n3000/one-time-update/<25G or 10G>/\
otsu-*.json --rsu
```

```
[2019-11-04 11:17:30,445] [INFO    ] [MAINTHREAD] INTEL FPGA PAC N3000
0000:06:00.0 IS NOT SECURE.
[2019-11-04 11:17:30,446] [WARNING ] [MAINTHREAD] UPDATE STARTING. PLEASE DO
NOT INTERRUPT.
[2019-11-04 11:17:30,446] [INFO    ] [0000:06:00.0] UPDATING INTEL FPGA PAC
N3000 : 0000:06:00.0
[2019-11-04 11:17:30,456] [INFO    ] [0000:06:00.0] READING FPGA@0X10000 FOR
256 BYTES FOR VERIFICATION
(100%) [####################] [256/256 BYTES]
[TIME:0:00:00.012339]
[2019-11-04 11:17:30,469] [INFO    ] [0000:06:00.0] READ/MODIFY/WRITING
FPGA@0X10000 FOR 256 BYTES (VC_OPTION_BITS_REVERSED)
[2019-11-04 11:17:30,584] [INFO    ] [0000:06:00.0] READING FPGA@0X10000 FOR
256 BYTES FOR VERIFICATION
(100%) [####################] [256/256 BYTES]
[TIME:0:00:00.012686]
[2019-11-04 11:17:30,601] [INFO    ] [0000:06:00.0] VERIFIED FPGA@0X10000
FOR 256 BYTES (VC_OPTION_BITS_REVERSED)
[2019-11-04 11:17:30,602] [INFO    ] [0000:06:00.0] ERASING FLASH@0X0 FOR
134217728 BYTES
[2019-11-04 11:17:42,122] [INFO    ] [0000:06:00.0] WRITING FLASH@0X3800000
FOR 179748 BYTES (VISTA_ROT_FACTORY_V254.255.16.BIN)
(100%) [####################] [179748/179748 BYTES]
[TIME:0:00:02.767896]
[2019-11-04 11:17:44,890] [INFO    ] [0000:06:00.0] READING FLASH@0X3800000
FOR 179748 BYTES FOR VERIFICATION
(100%) [####################] [179748/179748 BYTES]
[TIME:0:00:00.550265]
[2019-11-04 11:17:45,499] [INFO    ] [0000:06:00.0] VERIFIED FLASH@0X3800000
FOR 179748 BYTES (VISTA_ROT_FACTORY_V254.255.16.BIN)
[2019-11-04 11:17:45,499] [INFO    ] [0000:06:00.0] WRITING FLASH@0X3A00FF0
FOR 16 BYTES (VISTA_ROT_FACTORY_V254.255.16_HEADER.BIN)
(100%) [####################] [16/16 BYTES]
[TIME:0:00:00.000288]
[2019-11-04 11:17:45,512] [INFO    ] [0000:06:00.0] READING FLASH@0X3A00FF0
FOR 16 BYTES FOR VERIFICATION
(100%) [####################] [16/16 BYTES]
[TIME:0:00:00.000042]
[2019-11-04 11:17:45,525] [INFO    ] [0000:06:00.0] VERIFIED FLASH@0X3A00FF0
FOR 16 BYTES (VISTA_ROT_FACTORY_V254.255.16_HEADER.BIN)
[2019-11-04 11:17:45,529] [INFO    ] [0000:06:00.0] ERASING FPGA@0X7800000
FOR 8388608 BYTES
[2019-11-04 11:17:46,249] [INFO    ] [0000:06:00.0] WRITING FPGA@0X7FFC004
FOR 32 BYTES (VISTA_DEV_BMC_ROOT_HASH.RAW32)
(100%) [####################] [32/32 BYTES]
[TIME:0:00:00.000280]
[2019-11-04 11:17:46,250] [INFO    ] [0000:06:00.0] READING FPGA@0X7FFC004
FOR 32 BYTES FOR VERIFICATION
(100%) [####################] [32/32 BYTES]
[TIME:0:00:00.000043]
[2019-11-04 11:17:46,263] [INFO    ] [0000:06:00.0] VERIFIED FPGA@0X7FFC004
FOR 32 BYTES (VISTA_DEV_BMC_ROOT_HASH.RAW32)
[2019-11-04 11:17:46,263] [INFO    ] [0000:06:00.0] WRITING FPGA@0X7FFC000
FOR 4 BYTES (BMC_KEY_PROGRAMMED)
(100%) [####################] [4/4 BYTES]
[TIME:0:00:00.010881]
[2019-11-04 11:17:46,274] [INFO    ] [0000:06:00.0] READING FPGA@0X7FFC000
FOR 4 BYTES FOR VERIFICATION
(100%) [####################] [4/4 BYTES]
[TIME:0:00:00.011830]
[2019-11-04 11:17:46,287] [INFO    ] [0000:06:00.0] VERIFIED FPGA@0X7FFC000
FOR 4 BYTES (BMC_KEY_PROGRAMMED)
[2019-11-04 11:17:46,287] [INFO    ] [0000:06:00.0] ERASING FPGA@0X3820000
FOR 8257536 BYTES
```

```
[2019-11-04 11:17:47,118] [INFO    ] [0000:06:00.0] WRITING FPGA@0X3820000
FOR 8192 BYTES (INTEL-PAC-N3000.DTB)
(100%) [####################] [8192/8192 BYTES]
[TIME:0:00:00.134668]
[2019-11-04 11:17:47,253] [INFO    ] [0000:06:00.0] READING FPGA@0X3820000
FOR 8192 BYTES FOR VERIFICATION
(100%) [####################] [8192/8192 BYTES]
[TIME:0:00:00.025550]
[2019-11-04 11:17:47,279] [INFO    ] [0000:06:00.0] VERIFIED FPGA@0X3820000
FOR 8192 BYTES (INTEL-PAC-N3000.DTB)
[2019-11-04 11:17:47,279] [INFO    ] [0000:06:00.0] ERASING FPGA@0X20000 FOR
58720256 BYTES
[2019-11-04 11:19:07,148] [INFO    ] [0000:06:00.0] WRITING FPGA@0X20000 FOR
44589056 BYTES (VISTA_ROT_FACTORY_2X2X25G_REVERSE.BIN)
(100%) [####################] [44589056/44589056 BYTES]
[TIME:0:11:55.337529]
[2019-11-04 11:31:02,512] [INFO    ] [0000:06:00.0] READING FPGA@0X20000 FOR
44589056 BYTES FOR VERIFICATION
(100%) [####################] [44589056/44589056 BYTES]
[TIME:0:02:13.014834]
[2019-11-04 11:33:15,559] [INFO    ] [0000:06:00.0] VERIFIED FPGA@0X20000
FOR 44589056 BYTES (VISTA_ROT_FACTORY_2X2X25G_REVERSE.BIN)
[2019-11-04 11:33:15,566] [INFO    ] [0000:06:00.0] ERASING BMCIMG@0X0 FOR
32768 BYTES
[2019-11-04 11:33:15,569] [INFO    ] [0000:06:00.0] WRITING BMCIMG@0X0 FOR
32768 BYTES (Intel MAX
10_SYSTEM_REVD_ROT_DUAL_V2.0.6_UFM0_MDIOFILTER_REVERSED.RPD)
(100%) [####################] [32768/32768 BYTES]
[TIME:0:00:04.057484]
[2019-11-04 11:33:19,627] [INFO    ] [0000:06:00.0] READING BMCIMG@0X0 FOR
32768 BYTES FOR VERIFICATION
(100%) [####################] [32768/32768 BYTES]
[TIME:0:00:00.099048]
[2019-11-04 11:33:19,726] [INFO    ] [0000:06:00.0] VERIFIED BMCIMG@0X0 FOR
32768 BYTES (Intel MAX
10_SYSTEM_REVD_ROT_DUAL_V2.0.6_UFM0_MDIOFILTER_REVERSED.RPD)
[2019-11-04 11:33:19,727] [INFO    ] [0000:06:00.0] ERASING BMCIMG@0X8000
FOR 32768 BYTES
[2019-11-04 11:33:19,730] [INFO    ] [0000:06:00.0] WRITING BMCIMG@0X8000
FOR 32768 BYTES (Intel MAX
10_SYSTEM_REVD_ROT_DUAL_V2.0.6_UFM1_BOOTLOADER_REVERSED.RPD)
(100%) [####################] [32768/32768 BYTES]
[TIME:0:00:04.024239]
[2019-11-04 11:33:23,754] [INFO    ] [0000:06:00.0] READING BMCIMG@0X8000
FOR 32768 BYTES FOR VERIFICATION
(100%) [####################] [32768/32768 BYTES]
[TIME:0:00:00.102573]
[2019-11-04 11:33:23,857] [INFO    ] [0000:06:00.0] VERIFIED BMCIMG@0X8000
FOR 32768 BYTES (Intel MAX
10_SYSTEM_REVD_ROT_DUAL_V2.0.6_UFM1_BOOTLOADER_REVERSED.RPD)
[2019-11-04 11:33:23,857] [INFO    ] [0000:06:00.0] ERASING BMCIMG@0X70000
FOR 294912 BYTES
[2019-11-04 11:33:23,861] [INFO    ] [0000:06:00.0] ERASING BMCIMG@0X10000
FOR 393216 BYTES
[2019-11-04 11:33:23,866] [INFO    ] [0000:06:00.0] WRITING BMCIMG@0X10000
FOR 688128 BYTES (Intel MAX
10_SYSTEM_REVD_ROT_DUAL_V2.0.6_CFM1_FACTORY_REVERSED.RPD)
(100%) [####################] [688128/688128 BYTES]
[TIME:0:01:23.544640]
[2019-11-04 11:34:47,411] [INFO    ] [0000:06:00.0] READING BMCIMG@0X10000
FOR 688128 BYTES FOR VERIFICATION
(100%) [####################] [688128/688128 BYTES]
[TIME:0:00:02.038583]
[2019-11-04 11:34:49,451] [INFO    ] [0000:06:00.0] VERIFIED BMCIMG@0X10000
FOR 688128 BYTES (Intel MAX
10_SYSTEM_REVD_ROT_DUAL_V2.0.6_CFM1_FACTORY_REVERSED.RPD)
[2019-11-04 11:34:49,451] [INFO    ] [0000:06:00.0] ERASING BMCIMG@0XB8000
FOR 688128 BYTES
[2019-11-04 11:34:49,457] [INFO    ] [0000:06:00.0] WRITING BMCIMG@0XB8000
FOR 688128 BYTES (Intel MAX
10_SYSTEM_REVD_ROT_DUAL_V2.0.6_CFM0_USER_REVERSED.RPD)
```

```
(100%) [####################] [688128/688128 BYTES]
[TIME:0:01:23.489277]
[2019-11-04 11:36:12,947] [INFO    ] [0000:06:00.0] READING BMCIMG@0XB8000
FOR 688128 BYTES FOR VERIFICATION
(100%) [####################] [688128/688128 BYTES]
[TIME:0:00:02.065609]
[2019-11-04 11:36:15,013] [INFO    ] [0000:06:00.0] VERIFIED BMCIMG@0XB8000
FOR 688128 BYTES (Intel MAX
10_SYSTEM_REVD_ROT_DUAL_V2.0.6_CFM0_USER_REVERSED.RPD)
[2019-11-04 11:36:20,045] [INFO    ] [0000:06:00.0] ERASING BMCFW@0X0 FOR
8388608 BYTES
[2019-11-04 11:36:46,602] [INFO    ] [0000:06:00.0] WRITING BMCFW@0X0 FOR
179748 BYTES (VISTA_ROT_UPDATE_V2.0.16.BIN)
(100%) [####################] [179748/179748 BYTES]
[TIME:0:00:02.692600]
[2019-11-04 11:36:49,295] [INFO    ] [0000:06:00.0] READING BMCFW@0X0 FOR
179748 BYTES FOR VERIFICATION
(100%) [####################] [179748/179748 BYTES]
[TIME:0:00:00.535994]
[2019-11-04 11:36:49,888] [INFO    ] [0000:06:00.0] VERIFIED BMCFW@0X0 FOR
179748 BYTES (VISTA_ROT_UPDATE_V2.0.16.BIN)
[2019-11-04 11:36:49,888] [INFO    ] [0000:06:00.0] WRITING BMCFW@0X7F0000
FOR 16 BYTES (VISTA_ROT_UPDATE_V2.0.16_HEADER.BIN)
(100%) [####################] [16/16 BYTES]
[TIME:0:00:00.000291]
[2019-11-04 11:36:49,889] [INFO    ] [0000:06:00.0] READING BMCFW@0X7F0000
FOR 16 BYTES FOR VERIFICATION
(100%) [####################] [16/16 BYTES]
[TIME:0:00:00.011831]
[2019-11-04 11:36:49,902] [INFO    ] [0000:06:00.0] VERIFIED BMCFW@0X7F0000
FOR 16 BYTES (VISTA_ROT_UPDATE_V2.0.16_HEADER.BIN)
[2019-11-04 11:36:49,906] [INFO    ] [0000:06:00.0] ERASING FPGA@0X4000000
FOR 58720256 BYTES
[2019-11-04 11:38:09,042] [INFO    ] [0000:06:00.0] WRITING FPGA@0X4000000
FOR 44589056 BYTES (VISTA_ROT_FACTORY_4X25G_REVERSE.BIN)
(100%) [####################] [44589056/44589056 BYTES]
[TIME:0:11:54.859506]
[2019-11-04 11:50:03,926] [INFO    ] [0000:06:00.0] READING FPGA@0X4000000
FOR 44589056 BYTES FOR VERIFICATION
(100%) [####################] [44589056/44589056 BYTES]
[TIME:0:02:12.432333]
[2019-11-04 11:52:16,390] [INFO    ] [0000:06:00.0] VERIFIED FPGA@0X4000000
FOR 44589056 BYTES (VISTA_ROT_FACTORY_4X25G_REVERSE.BIN)
[2019-11-04 11:52:16,463] [INFO    ] [MAINTHREAD]
[[PCI_ADDRESS(0000:06:00.0), PCI_ID(0X8086, 0X0B30)]] PERFORMING RSU
OPERATION
[2019-11-04 11:52:16,518] [INFO    ] [MAINTHREAD]
[[PCI_ADDRESS(0000:00:03.0), PCI_ID(0X8086, 0X2F08)]] REMOVING DEVICE FROM
PCIE BUS
[2019-11-04 11:52:16,519] [INFO    ] [MAINTHREAD] WAITING 10 SECONDS FOR BOOT
[2019-11-04 11:52:26,529] [INFO    ] [MAINTHREAD] RESCANNING PCIE BUS: /SYS/
DEVICES/PCI0000:00/PCI_BUS/0000:00
[2019-11-04 11:52:30,312] [INFO    ] [MAINTHREAD] TOTAL TIME: 0:34:59.865373
[2019-11-04 11:52:30,312] [INFO    ] [MAINTHREAD] ONE-TIME SECURE UPDATE OK
```

If OTSU fails, run the `fpgainfo fme` to find out the Intel MAX 10 build version and take appropriate action as stated:

Intel MAX 10 Build Version	Action
D.111.2.13	Repeat OTSU
D.2.0.6	Run command: `sudo super-rsu /usr/share/opae/n3000/super-rsu/<2x2x25G or 8x10G or 4x25G>/super-rsu-*.json --with-rsu`

3. To verify successful OTSU:

```
sudo fpgaotsu /usr/share/opae/n3000/one-time-update/<25G or 10G>/\
otsu-*.json --verify
```

```
[2019-11-04 12:10:00,844] [INFO    ] [MAINTHREAD] INTEL FPGA PAC N3000
0000:06:00.0 IS ALREADY SECURE.
[2019-11-04 12:10:00,845] [WARNING ] [MAINTHREAD] UPDATE STARTING. PLEASE DO
NOT INTERRUPT.
[2019-11-04 12:10:00,845] [INFO    ] [MAINTHREAD] TOTAL TIME: 0:00:00.000012
[2019-11-04 12:10:00,845] [INFO    ] [MAINTHREAD] ONE-TIME SECURE UPDATE OK
```

At this point, the Intel FPGA PAC N3000 will have the following Intel Arria 10 image, Intel MAX 10 NIOS FW and Intel MAX 10 Build versions.

Configuration	User Partition Image	Bitstream ID	PR interface ID	Intel MAX 10 NIOS FW	Intel MAX 10 BUILD
2x2x25G	4x25G	0x2300110010309	f3c99413-5081-4aad-bced-07eb84a6d0bb	D.2.0.19	D.2.0.6
4x25G					
8x10G	8x10G	0x2300010010309	901dd697-ca79-4b05-b843-8138cefa2846	D.2.0.19	D.2.0.6

Important: The 2x2x25G or 4x25G Configuration Installer loads the FPGA flash user partition with 4x25G Intel provided factory test image and loads factory partition with 2x2x25G Intel provided factory test image. The 8x10 Configuration Installer loads both the user and factory FPGA flash partitions with 8x10G Intel provided factory test image.

4. Install the PV 1.1 Patch.

C. Configure the 4.19 Kernel

Before you proceed ahead, ensure that you have installed Centos 7.6 in your system.

1. When you configure 4.19 kernel, make sure the following minimum parameters are set as recommended for the compatible 4.19 kernel:

Table 15. Minimum Parameters

Parameter	Value
CONFIG_MTD=m	Build for kernel module
CONFIG_SPI_MASTER=y	Build in kernel
CONFIG_SPI_BITBANG=m	Build for kernel module
CONFIG_REGMAP_MMIO=y	Build in kernel
CONFIG_SPI_BITBANG=m	Build for kernel module
CONFIG_EEPROM_AT24=m	Build for kernel module
# CONFIG_MTD_SPI_NOR is not set	Do not build in kernel, use OPAE driver instead
# CONFIG_FPGA is not set	Do not build in kernel, use OPAE driver instead

2. Download the 4.19 kernel source from The Linux Kernel Archives and build it.
3. Ensure that the header files from kernel build are found during the build process of OPAE driver. Also, ensure that your kernel source version and kernel header version are the same. If the kernel source and header do not match the kernel version running on the server, OPAE driver will not be installed.
4. Update the grub configuration to choose to boot from 4.19 version before you reboot.

D. fpgabist Sample Output

Note: For pre-production boards, the MAC addresses in fpgabist output shows FF:FF:FF:FF:FF:FF. For more information, refer to the Intel Acceleration Stack for Intel Xeon CPU with FPGAs Version 1.1 Release Notes: Intel FPGA Programmable Acceleration Card N3000.

Related Information

Test PCIe and External Memories with fpgabist on page 30

D.1. For 2x2x25G Configuration

```
Board Management Controller, Intel MAX 10 NIOS FW version D.2.0.19
Board Management Controller, Intel MAX 10 Build version D.2.0.6
//****** FME ******//
Object Id                    : 0xEF00000
PCIe s:b:d.f                 : 0000:b2:00.0
Device Id                    : 0x0b30
Numa Node                    : 1
Ports Num                    : 01
Bitstream Id                 : 0x23000410010309
Bitstream Version            : 0.2.3
Pr Interface Id              : a5d72a3c-c8b0-4939-912c-f715e5dc10ca
Boot Page                    : user

Board Management Controller, Intel MAX 10 NIOS FW version D.2.0.19
Board Management Controller, Intel MAX 10 Build version D.2.0.6
//****** PORT ******//
Object Id                    : 0xEE00000
PCIe s:b:d.f                 : 0000:b2:00.0
Device Id                    : 0x0b30
Numa Node                    : 1
Ports Num                    : 01
Bitstream Id                 : 0x23000410010309
Bitstream Version            : 0.2.3
Pr Interface Id              : a5d72a3c-c8b0-4939-912c-f715e5dc10ca
Accelerator Id               : 9aeffe5f-8457-0612-c000-c9660d824272

Board Management Controller, Intel MAX 10 NIOS FW version D.2.0.19
Board Management Controller, Intel MAX 10 Build version D.2.0.6
//****** TEMP ******//
Object Id                    : 0xEF00000
PCIe s:b:d.f                 : 0000:b2:00.0
Device Id                    : 0x0b30
Numa Node                    : 1
Ports Num                    : 01
Bitstream Id                 : 0x23000410010309
Bitstream Version            : 0.2.3
Pr Interface Id              : a5d72a3c-c8b0-4939-912c-f715e5dc10ca
(12) FPGA Die Temperature    : 73.50 Celsius
(13) Board Temperature       : 46.50 Celsius
(15) QSFP0 Temperature       : N/A
(38) QSFP1 Temperature       : N/A
(44) PKVL0 Core Temperature  : 73.50 Celsius
(45) PKVL0 SerDes Temperature : 73.50 Celsius
(46) PKVL1 Core Temperature  : 74.00 Celsius
```

```
(47) PKVL1 SerDes Temperature : 74.50 Celsius

Board Management Controller, Intel MAX 10 NIOS FW version D.2.0.19
Board Management Controller, Intel MAX 10 Build version D.2.0.6
//****** POWER ******//
Object Id                     : 0xEF00000
PCIe s:b:d.f                  : 0000:b2:00.0
Device Id                     : 0x0b30
Numa Node                     : 1
Ports Num                     : 01
Bitstream Id                  : 0x23000410010309
Bitstream Version             : 0.2.3
Pr Interface Id               : a5d72a3c-c8b0-4939-912c-f715e5dc10ca
( 1) Board Power              : 74.81 Watts
( 2) 12V Backplane Current    : 3.27 Amps
( 3) 12V Backplane Voltage    : 12.16 Volts
( 4) 1.2V Voltage             : 1.19 Volts
( 6) 1.8V Voltage             : 1.80 Volts
( 8) 3.3V Voltage             : 3.25 Volts
(10) FPGA Core Voltage        : 0.90 Volts
(11) FPGA Core Current        : 18.51 Amps
(14) QSFP0 Supply Voltage     : 0.00 Volts
(24) 12V AUX Current          : 2.88 Amps
(25) 12V AUX Voltage          : 12.19 Volts
(37) QSFP1 Supply Voltage     : 0.00 Volts

Board Management Controller, Intel MAX 10 NIOS FW version D.2.0.19
Board Management Controller, Intel MAX 10 Build version D.2.0.6
//****** FME ERRORS ******//
Object Id                     : 0xEF00000
PCIe s:b:d.f                  : 0000:b2:00.0
Device Id                     : 0x0b30
Numa Node                     : 1
Ports Num                     : 01
Bitstream Id                  : 0x23000410010309
Bitstream Version             : 0.2.3
Pr Interface Id               : a5d72a3c-c8b0-4939-912c-f715e5dc10ca
PCIe0 Errors                  : 0x0
PCIe1 Errors                  : 0x0
Catfatal Errors               : 0x0
Seu Emr                       : 0x0
Inject Error                  : 0x0
Nonfatal Errors               : 0x0
Next Error                    : 0x0
First Error                   : 0x0
Errors                        : 0x0

Board Management Controller, Intel MAX 10 NIOS FW version D.2.0.19
Board Management Controller, Intel MAX 10 Build version D.2.0.6
//****** PORT ERRORS ******//
Object Id                     : 0xEE00000
PCIe s:b:d.f                  : 0000:b2:00.0
Device Id                     : 0x0b30
Numa Node                     : 1
Ports Num                     : 01
Bitstream Id                  : 0x23000410010309
Bitstream Version             : 0.2.3
Pr Interface Id               : a5d72a3c-c8b0-4939-912c-f715e5dc10ca
Accelerator Id                : 9aeffe5f-8457-0612-c000-c9660d824272
First Malformed Req           : 0x0
First Error                   : 0x0
Errors                        : 0x0

Board Management Controller, Intel MAX 10 NIOS FW version D.2.0.19
Board Management Controller, Intel MAX 10 Build version D.2.0.6
//****** PHY ******//
Object Id                     : 0xEF00000
PCIe s:b:d.f                  : 0000:b2:00.0
Device Id                     : 0x0b30
Numa Node                     : 1
Ports Num                     : 01
```

```
Bitstream Id                    : 0x23000410010309
Bitstream Version               : 0.2.3
Pr Interface Id                 : a5d72a3c-c8b0-4939-912c-f715e5dc10ca
//****** PHY GROUP 0 ******//
Direction                       : Line side
Speed                           : 25 Gbps
Number of PHYs                  : 4
//****** PHY GROUP 1 ******//
Direction                       : Host side
Speed                           : 40 Gbps
Number of PHYs                  : 4
//****** Intel C827 Retimer ******//
Port0 25G                       : Up
Port1 25G                       : Up
Port2 25G                       : Up
Port3 25G                       : Up
Retimer A Version               : 101c.1064
Retimer B Version               : 101c.1064

Board Management Controller, Intel MAX 10 NIOS FW version D.2.0.19
Board Management Controller, Intel MAX 10 Build version D.2.0.6
//****** MAC ******//
Object Id                       : 0xEF00000
PCIe s:b:d.f                    : 0000:b2:00.0
Device Id                       : 0x0b30
Numa Node                       : 1
Ports Num                       : 01
Bitstream Id                    : 0x23000410010309
Bitstream Version               : 0.2.3
Pr Interface Id                 : a5d72a3c-c8b0-4939-912c-f715e5dc10ca
Number of MACs                  : 8
MAC address 0                   : 64:4c:36:00:16:e0
MAC address 1                   : 64:4c:36:00:16:e1
MAC address 2                   : 64:4c:36:00:16:e2
MAC address 3                   : 64:4c:36:00:16:e3
MAC address 4                   : 64:4c:36:00:16:e4
MAC address 5                   : 64:4c:36:00:16:e5
MAC address 6                   : 64:4c:36:00:16:e6
MAC address 7                   : 64:4c:36:00:16:e7
found the NLB offset=0x28000

Cachelines Read_Count Write_Count Cache_Rd_Hit Cache_Wr_Hit Cache_Rd_Miss
Cache_Wr_Miss   Eviction 'Clocks(@200 MHz)'   Rd_Bandwidth   Wr_Bandwidth
      1024   97676676    97675844           0            0
0             0          0        200028728     6.250 GB/s     6.250 GB/s

VH0_Rd_Count VH0_Wr_Count VH1_Rd_Count VH1_Wr_Count VL0_Rd_Count VL0_Wr_Count
    97676676     97675845            0            0            0            0

found the NLB offset=0x28000

Cachelines Read_Count Write_Count Cache_Rd_Hit Cache_Wr_Hit Cache_Rd_Miss
Cache_Wr_Miss   Eviction 'Clocks(@200 MHz)'   Rd_Bandwidth   Wr_Bandwidth
      1024   97665536    97664692           0            0
0             0          0        200027985     6.250 GB/s     6.250 GB/s

VH0_Rd_Count VH0_Wr_Count VH1_Rd_Count VH1_Wr_Count VL0_Rd_Count VL0_Wr_Count
    97665536     97664693            0            0            0            0

found the NLB offset=0x28000

Cachelines Read_Count Write_Count Cache_Rd_Hit Cache_Wr_Hit Cache_Rd_Miss
Cache_Wr_Miss   Eviction 'Clocks(@200 MHz)'   Rd_Bandwidth   Wr_Bandwidth
      1024   97661776    97660924           0            0
0             0          0        200030859     6.249 GB/s     6.249 GB/s

VH0_Rd_Count VH0_Wr_Count VH1_Rd_Count VH1_Wr_Count VL0_Rd_Count VL0_Wr_Count
    97661780     97660925            0            0            0            0

found the NLB offset=0x28000
```

```
Cachelines Read_Count Write_Count Cache_Rd_Hit Cache_Wr_Hit Cache_Rd_Miss
Cache_Wr_Miss   Eviction 'Clocks(@200 MHz)'   Rd_Bandwidth   Wr_Bandwidth
      1024   97641100    97640260            0            0
0             0          0        200027828     6.248 GB/s     6.248 GB/s

VH0_Rd_Count VH0_Wr_Count VH1_Rd_Count VH1_Wr_Count VL0_Rd_Count VL0_Wr_Count
    97641100     97640261            0            0            0            0

Running test in HW mode
Buffer Verification Success!
Buffer Verification Success!
Running DDR sweep test
Buffer pointer = 0x7f1204dfc000, size = 0x100000000 (0x7f1204dfc000 through
0x7f1304dfc000)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA

Measured bandwidth = 6079.116050 Megabytes/sec

Clear buffer

DDR Sweep FPGA to Host

Measured bandwidth = 6648.719007 Megabytes/sec

Verifying buffer..
Buffer Verification Success!
DDR sweep with unaligned pointer and size
Buffer pointer = 0x7f12057fd03d, size = 0xfffffbe (0x7f12057fd03d through
0x7f13057fcffb)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA

Measured bandwidth = 6077.252495 Megabytes/sec

Clear buffer

DDR Sweep FPGA to Host

Measured bandwidth = 6606.536954 Megabytes/sec

Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7f12057fd003, size = 0xfffffffd (0x7f12057fd003 through
0x7f13057fd000)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA

Measured bandwidth = 6077.917447 Megabytes/sec

Clear buffer

DDR Sweep FPGA to Host

Measured bandwidth = 6600.669144 Megabytes/sec

Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7f12057fd007, size = 0xfffffff6 (0x7f12057fd007 through
0x7f13057fcffd)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA

Measured bandwidth = 6077.249565 Megabytes/sec

Clear buffer
```

```
DDR Sweep FPGA to Host

Measured bandwidth = 6599.379119 Megabytes/sec

Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7f12057fd000, size = 0xfffffffd (0x7f12057fd000 through
0x7f13057fcffd)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA

Measured bandwidth = 6079.961435 Megabytes/sec

Clear buffer

DDR Sweep FPGA to Host

Measured bandwidth = 6639.524496 Megabytes/sec

Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7f12057fd000, size = 0xffffffc3 (0x7f12057fd000 through
0x7f13057fcfc3)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA

Measured bandwidth = 6078.064773 Megabytes/sec

Clear buffer

DDR Sweep FPGA to Host

Measured bandwidth = 6728.900022 Megabytes/sec

Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7f12057fd000, size = 0xfffffff9 (0x7f12057fd000 through
0x7f13057fcff9)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA

Measured bandwidth = 6082.346431 Megabytes/sec

Clear buffer

DDR Sweep FPGA to Host

Measured bandwidth = 6722.463627 Megabytes/sec

Verifying buffer..
Buffer Verification Success!
Running test in HW mode
Buffer Verification Success!
Buffer Verification Success!
Running DDR sweep test
Buffer pointer = 0x7f7a5abfc000, size = 0x100000000 (0x7f7a5abfc000 through
0x7f7b5abfc000)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA

Measured bandwidth = 6079.671455 Megabytes/sec

Clear buffer

DDR Sweep FPGA to Host

Measured bandwidth = 6654.373365 Megabytes/sec
```

```
Verifying buffer..
Buffer Verification Success!
DDR sweep with unaligned pointer and size
Buffer pointer = 0x7f7a5b5fd03d, size = 0xfffffffbe (0x7f7a5b5fd03d through
0x7f7b5b5fcffb)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA

Measured bandwidth = 6078.450083 Megabytes/sec

Clear buffer

DDR Sweep FPGA to Host

Measured bandwidth = 6676.330310 Megabytes/sec

Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7f7a5b5fd003, size = 0xfffffffd (0x7f7a5b5fd003 through
0x7f7b5b5fd000)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA

Measured bandwidth = 6079.223431 Megabytes/sec

Clear buffer

DDR Sweep FPGA to Host

Measured bandwidth = 6673.156991 Megabytes/sec

Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7f7a5b5fd007, size = 0xfffffff6 (0x7f7a5b5fd007 through
0x7f7b5b5fcffd)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA

Measured bandwidth = 6082.227028 Megabytes/sec

Clear buffer

DDR Sweep FPGA to Host

Measured bandwidth = 6664.409925 Megabytes/sec

Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7f7a5b5fd000, size = 0xfffffffd (0x7f7a5b5fd000 through
0x7f7b5b5fcffd)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA

Measured bandwidth = 6082.114956 Megabytes/sec

Clear buffer

DDR Sweep FPGA to Host

Measured bandwidth = 6719.411365 Megabytes/sec

Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7f7a5b5fd000, size = 0xfffffffc3 (0x7f7a5b5fd000 through
0x7f7b5b5fcfc3)
Allocated test buffer
```

```
Fill test buffer
DDR Sweep Host to FPGA

Measured bandwidth = 6083.076700 Megabytes/sec

Clear buffer

DDR Sweep FPGA to Host

Measured bandwidth = 6712.587358 Megabytes/sec

Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7f7a5b5fd000, size = 0xfffffff9 (0x7f7a5b5fd000 through
0x7f7b5b5fcff9)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA

Measured bandwidth = 6083.303806 Megabytes/sec

Clear buffer

DDR Sweep FPGA to Host

Measured bandwidth = 6716.590674 Megabytes/sec

Verifying buffer..
Buffer Verification Success!
Running test in HW mode
Buffer Verification Success!
Buffer Verification Success!
Running DDR sweep test
Buffer pointer = 0x7fc286bfc000, size = 0x40000000 (0x7fc286bfc000 through
0x7fc2c6bfc000)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA

Measured bandwidth = 2502.886807 Megabytes/sec

Clear buffer

DDR Sweep FPGA to Host

Measured bandwidth = 2581.081446 Megabytes/sec

Verifying buffer..
Buffer Verification Success!
DDR sweep with unaligned pointer and size
Buffer pointer = 0x7fc2875fd03d, size = 0x3fffffbe (0x7fc2875fd03d through
0x7fc2c75fcffb)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA

Measured bandwidth = 2502.966117 Megabytes/sec

Clear buffer

DDR Sweep FPGA to Host

Measured bandwidth = 2581.055594 Megabytes/sec

Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7fc2875fd003, size = 0x3ffffffd (0x7fc2875fd003 through
0x7fc2c75fd000)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA
```

```
Measured bandwidth = 2502.960558 Megabytes/sec

Clear buffer

DDR Sweep FPGA to Host

Measured bandwidth = 2581.020016 Megabytes/sec

Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7fc2875fd007, size = 0x3ffffff6 (0x7fc2875fd007 through
0x7fc2c75fcffd)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA

Measured bandwidth = 2503.483825 Megabytes/sec

Clear buffer

DDR Sweep FPGA to Host

Measured bandwidth = 2582.389741 Megabytes/sec

Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7fc2875fd000, size = 0x3fffffd (0x7fc2875fd000 through
0x7fc2c75fcffd)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA

Measured bandwidth = 2502.961054 Megabytes/sec

Clear buffer

DDR Sweep FPGA to Host

Measured bandwidth = 2580.327886 Megabytes/sec

Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7fc2875fd000, size = 0x3fffffc3 (0x7fc2875fd000 through
0x7fc2c75fcfc3)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA

Measured bandwidth = 2479.009733 Megabytes/sec

Clear buffer

DDR Sweep FPGA to Host

Measured bandwidth = 2581.453682 Megabytes/sec

Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7fc2875fd000, size = 0x3ffffff9 (0x7fc2875fd000 through
0x7fc2c75fcff9)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA

Measured bandwidth = 2476.627081 Megabytes/sec

Clear buffer

DDR Sweep FPGA to Host
```

```
Measured bandwidth = 2581.714146 Megabytes/sec

Verifying buffer..
Buffer Verification Success!
Running test in HW mode
Buffer Verification Success!
Buffer Verification Success!
Running DDR sweep test
Buffer pointer = 0x7f18ce9fc000, size = 0x1000000 (0x7f18ce9fc000 through
0x7f18cf9fc000)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA

Measured bandwidth = 933.216916 Megabytes/sec

Clear buffer

DDR Sweep FPGA to Host

Measured bandwidth = 910.980582 Megabytes/sec

Verifying buffer..
Buffer Verification Success!
============================================================
Beginning FPGA Built-In Self-Test
============================================================

Running mode: nlb
Running fpgadiag lpbk1 vh0-vh0 test...
Running fpgadiag lpbk1 vh0-vh1 test...
Running fpgadiag lpbk1 vh1-vh0 test...
Running fpgadiag lpbk1 vh1-vh1 test...
Finished Executing NLB (FPGA DIAG) Tests

Running mode: dma_afu
Running fpga_dma_test test on DDR4_A...
Running fpga_dma_test test on DDR4_B...
Running fpga_dma_test test on DDR4_C...
Running fpga_dma_test test on QDR...
Finished Executing DMA Tests

Built-in Self-Test Completed.
```

D.2. For 8x10G Configuration

```
============================================================
Beginning FPGA Built-In Self-Test
============================================================
Board Management Controller, Intel MAX 10 NIOS FW version D.2.0.19
Board Management Controller, Intel MAX 10 Build version D.2.0.6
//****** FME ******//
Object Id                        : 0xF000000
PCIe s:b:d.f                     : 0000:8a:00.0
Device Id                        : 0x0b30
Numa Node                        : 1
Ports Num                        : 01
Bitstream Id                     : 0x23000010010309
Bitstream Version                : 0.2.3
Pr Interface Id                  : 901dd697-ca79-4b05-b843-8138cefa2846
Boot Page                        : user

Board Management Controller, Intel MAX 10 NIOS FW version D.2.0.19
Board Management Controller, Intel MAX 10 Build version D.2.0.6
//****** PORT ******//
Object Id                        : 0xEF00000
PCIe s:b:d.f                     : 0000:8a:00.0
Device Id                        : 0x0b30
```

```
Numa Node                     : 1
Ports Num                     : 01
Bitstream Id                  : 0x23000010010309
Bitstream Version             : 0.2.3
Pr Interface Id               : 901dd697-ca79-4b05-b843-8138cefa2846
Accelerator Id                : 9aeffe5f-8457-0612-c000-c9660d824272

Board Management Controller, Intel MAX 10 NIOS FW version D.2.0.19
Board Management Controller, Intel MAX 10 Build version D.2.0.6
//****** TEMP ******//
Object Id                     : 0xF000000
PCIe s:b:d.f                  : 0000:8a:00.0
Device Id                     : 0x0b30
Numa Node                     : 1
Ports Num                     : 01
Bitstream Id                  : 0x23000010010309
Bitstream Version             : 0.2.3
Pr Interface Id               : 901dd697-ca79-4b05-b843-8138cefa2846
(12) FPGA Die Temperature     : 51.00 Celsius
(13) Board Temperature        : 32.00 Celsius
(15) QSFP0 Temperature        : N/A
(38) QSFP1 Temperature        : N/A
(44) PKVL0 Core Temperature   : 49.00 Celsius
(45) PKVL0 SerDes Temperature : 49.50 Celsius
(46) PKVL1 Core Temperature   : 49.50 Celsius
(47) PKVL1 SerDes Temperature : 50.50 Celsius

Board Management Controller, Intel MAX 10 NIOS FW version D.2.0.19
Board Management Controller, Intel MAX 10 Build version D.2.0.6
//****** POWER ******//
Object Id                     : 0xF000000
PCIe s:b:d.f                  : 0000:8a:00.0
Device Id                     : 0x0b30
Numa Node                     : 1
Ports Num                     : 01
Bitstream Id                  : 0x23000010010309
Bitstream Version             : 0.2.3
Pr Interface Id               : 901dd697-ca79-4b05-b843-8138cefa2846
( 1) Board Power              : 60.10 Watts
( 2) 12V Backplane Current    : 2.77 Amps
( 3) 12V Backplane Voltage    : 12.14 Volts
( 4) 1.2V Voltage             : 1.19 Volts
( 6) 1.8V Voltage             : 1.80 Volts
( 8) 3.3V Voltage             : 3.27 Volts
(10) FPGA Core Voltage        : 0.90 Volts
(11) FPGA Core Current        : 12.18 Amps
(14) QSFP0 Supply Voltage     : N/A
(24) 12V AUX Current          : 2.17 Amps
(25) 12V AUX Voltage          : 12.18 Volts
(37) QSFP1 Supply Voltage     : N/A

Board Management Controller, Intel MAX 10 NIOS FW version D.2.0.19
Board Management Controller, Intel MAX 10 Build version D.2.0.6
//****** PORT ERRORS ******//
Object Id                     : 0xEF00000
PCIe s:b:d.f                  : 0000:8a:00.0
Device Id                     : 0x0b30
Numa Node                     : 1
Ports Num                     : 01
Bitstream Id                  : 0x23000010010309
Bitstream Version             : 0.2.3
Pr Interface Id               : 901dd697-ca79-4b05-b843-8138cefa2846
Accelerator Id                : 9aeffe5f-8457-0612-c000-c9660d824272
First Error                   : 0x0
First Malformed Req           : 0x0
Errors                        : 0x0
Board Management Controller, Intel MAX 10 NIOS FW version D.2.0.19
Board Management Controller, Intel MAX 10 Build version D.2.0.6
//****** FME ERRORS ******//
Object Id                     : 0xF000000
PCIe s:b:d.f                  : 0000:8a:00.0
```

```
Device Id                     : 0x0b30
Numa Node                     : 1
Ports Num                     : 01
Bitstream Id                  : 0x23000010010309
Bitstream Version             : 0.2.3
Pr Interface Id               : 901dd697-ca79-4b05-b843-8138cefa2846
Seu Emr                       : 0x0
First Error                   : 0x0
Next Error                    : 0x0
Errors                        : 0x0
PCIe1 Errors                  : 0x0
Nonfatal Errors               : 0x0
Inject Error                  : 0x0
Catfatal Errors               : 0x0
PCIe0 Errors                  : 0x0

Board Management Controller, Intel MAX 10 NIOS FW version D.2.0.19
Board Management Controller, Intel MAX 10 Build version D.2.0.6
//****** PHY ******//
Object Id                     : 0xF000000
PCIe s:b:d.f                  : 0000:8a:00.0
Device Id                     : 0x0b30
Numa Node                     : 1
Ports Num                     : 01
Bitstream Id                  : 0x23000010010309
Bitstream Version             : 0.2.3
Pr Interface Id               : 901dd697-ca79-4b05-b843-8138cefa2846
//****** PHY GROUP 0 ******//
Direction                     : Line side
Speed                         : 10 Gbps
Number of PHYs                : 8
//****** PHY GROUP 1 ******//
Direction                     : Host side
Speed                         : 10 Gbps
Number of PHYs                : 8
//****** Intel C827 Retimer ******//
Port0 10G                     : Down
Port1 10G                     : Down
Port2 10G                     : Down
Port3 10G                     : Down
Port4 10G                     : Down
Port5 10G                     : Down
Port6 10G                     : Down
Port7 10G                     : Down
Retimer A Version             : 101c.1064
Retimer B Version             : 101c.1064

Board Management Controller, Intel MAX 10 NIOS FW version D.2.0.19
Board Management Controller, Intel MAX 10 Build version D.2.0.6
//****** MAC ******//
Object Id                     : 0xF000000
PCIe s:b:d.f                  : 0000:8a:00.0
Device Id                     : 0x0b30
Numa Node                     : 1
Ports Num                     : 01
Bitstream Id                  : 0x23000010010309
Bitstream Version             : 0.2.3
Pr Interface Id               : 901dd697-ca79-4b05-b843-8138cefa2846
Number of MACs                : 8
MAC address 0                 : 64:4c:36:00:16:e0
MAC address 1                 : 64:4c:36:00:16:e1
MAC address 2                 : 64:4c:36:00:16:e2
MAC address 3                 : 64:4c:36:00:16:e3
MAC address 4                 : 64:4c:36:00:16:e4
MAC address 5                 : 64:4c:36:00:16:e5
MAC address 6                 : 64:4c:36:00:16:e6
MAC address 7                 : 64:4c:36:00:16:e7

Running mode: nlb
Running fpgadiag lpbk1 vh0-vh0 test...
found the NLB offset=0x28000
```

```
Cachelines Read_Count Write_Count Cache_Rd_Hit Cache_Wr_Hit Cache_Rd_Miss
Cache_Wr_Miss   Eviction 'Clocks(@200 MHz)'   Rd_Bandwidth   Wr_Bandwidth
      1024   97724300    97723396           0            0
0             0          0        200174911     6.249 GB/s     6.249 GB/s

VH0_Rd_Count VH0_Wr_Count VH1_Rd_Count VH1_Wr_Count VL0_Rd_Count VL0_Wr_Count
    97724300     97723397            0            0            0            0

Running fpgadiag lpbk1 vh0-vh1 test...
found the NLB offset=0x28000

Cachelines Read_Count Write_Count Cache_Rd_Hit Cache_Wr_Hit Cache_Rd_Miss
Cache_Wr_Miss   Eviction 'Clocks(@200 MHz)'   Rd_Bandwidth   Wr_Bandwidth
      1024   97717928    97717048           0            0
0             0          0        200161726     6.249 GB/s     6.249 GB/s

VH0_Rd_Count VH0_Wr_Count VH1_Rd_Count VH1_Wr_Count VL0_Rd_Count VL0_Wr_Count
    97717932     97717049            0            0            0            0

Running fpgadiag lpbk1 vh1-vh0 test...
found the NLB offset=0x28000

Cachelines Read_Count Write_Count Cache_Rd_Hit Cache_Wr_Hit Cache_Rd_Miss
Cache_Wr_Miss   Eviction 'Clocks(@200 MHz)'   Rd_Bandwidth   Wr_Bandwidth
      1024   97909184    97908288           0            0
0             0          0        200496312     6.251 GB/s     6.251 GB/s

VH0_Rd_Count VH0_Wr_Count VH1_Rd_Count VH1_Wr_Count VL0_Rd_Count VL0_Wr_Count
    97909184     97908289            0            0            0            0

Running fpgadiag lpbk1 vh1-vh1 test...
found the NLB offset=0x28000

Cachelines Read_Count Write_Count Cache_Rd_Hit Cache_Wr_Hit Cache_Rd_Miss
Cache_Wr_Miss   Eviction 'Clocks(@200 MHz)'   Rd_Bandwidth   Wr_Bandwidth
      1024   97911048    97910212           0            0
0             0          0        200494947     6.251 GB/s     6.251 GB/s

VH0_Rd_Count VH0_Wr_Count VH1_Rd_Count VH1_Wr_Count VL0_Rd_Count VL0_Wr_Count
    97911048     97910213            0            0            0            0

Finished Executing NLB (FPGA DIAG) Tests

Running mode: dma_afu
Running fpga_dma_test test on DDR4_A...

Running test in HW mode
Buffer Verification Success!
Buffer Verification Success!
Running DDR sweep test
Buffer pointer = 0x7f3c4aede000, size = 0x100000000 (0x7f3c4aede000 through
0x7f3d4aede000)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA
Measured bandwidth = 6047.228482 Megabytes/sec
Clear buffer
DDR Sweep FPGA to Host
Measured bandwidth = 6818.460343 Megabytes/sec
Verifying buffer..
Buffer Verification Success!
DDR sweep with unaligned pointer and size
Buffer pointer = 0x7f3c4b8df03d, size = 0xfffffffe (0x7f3c4b8df03d through
0x7f3d4b8deffb)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA
Measured bandwidth = 6043.044307 Megabytes/sec
Clear buffer
DDR Sweep FPGA to Host
```

```
Measured bandwidth = 6684.135801 Megabytes/sec
Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7f3c4b8df003, size = 0xfffffffd (0x7f3c4b8df003 through
0x7f3d4b8df000)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA
Measured bandwidth = 6047.563358 Megabytes/sec
Clear buffer
DDR Sweep FPGA to Host
Measured bandwidth = 6663.338529 Megabytes/sec
Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7f3c4b8df007, size = 0xfffffff6 (0x7f3c4b8df007 through
0x7f3d4b8deffd)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA
Measured bandwidth = 6045.153009 Megabytes/sec
Clear buffer
DDR Sweep FPGA to Host
Measured bandwidth = 6673.172429 Megabytes/sec
Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7f3c4b8df000, size = 0xfffffffd (0x7f3c4b8df000 through
0x7f3d4b8deffd)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA
Measured bandwidth = 6043.433517 Megabytes/sec
Clear buffer
DDR Sweep FPGA to Host
Measured bandwidth = 6751.913703 Megabytes/sec
Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7f3c4b8df000, size = 0xffffffc3 (0x7f3c4b8df000 through
0x7f3d4b8defc3)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA
Measured bandwidth = 6047.493452 Megabytes/sec
Clear buffer
DDR Sweep FPGA to Host
Measured bandwidth = 6787.509760 Megabytes/sec
Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7f3c4b8df000, size = 0xfffffff9 (0x7f3c4b8df000 through
0x7f3d4b8deff9)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA
Measured bandwidth = 6048.778852 Megabytes/sec
Clear buffer
DDR Sweep FPGA to Host
Measured bandwidth = 6746.716503 Megabytes/sec
Verifying buffer..
Buffer Verification Success!
Running fpga_dma_test test on DDR4_B...

Running test in HW mode
Buffer Verification Success!
Buffer Verification Success!
Running DDR sweep test
Buffer pointer = 0x7f734649e000, size = 0x100000000 (0x7f734649e000 through
0x7f744649e000)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA
Measured bandwidth = 5998.758802 Megabytes/sec
Clear buffer
```

```
DDR Sweep FPGA to Host
Measured bandwidth = 6475.479900 Megabytes/sec
Verifying buffer..
Buffer Verification Success!
DDR sweep with unaligned pointer and size
Buffer pointer = 0x7f7346e9f03d, size = 0xfffffffe (0x7f7346e9f03d through
0x7f7446e9effb)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA
Measured bandwidth = 5993.467581 Megabytes/sec
Clear buffer
DDR Sweep FPGA to Host
Measured bandwidth = 6445.968784 Megabytes/sec
Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7f7346e9f003, size = 0xfffffffd (0x7f7346e9f003 through
0x7f7446e9f000)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA
Measured bandwidth = 5995.302867 Megabytes/sec
Clear buffer
DDR Sweep FPGA to Host
Measured bandwidth = 6411.985356 Megabytes/sec
Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7f7346e9f007, size = 0xfffffff6 (0x7f7346e9f007 through
0x7f7446e9effd)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA
Measured bandwidth = 5997.001248 Megabytes/sec
Clear buffer
DDR Sweep FPGA to Host
Measured bandwidth = 6427.978131 Megabytes/sec
Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7f7346e9f000, size = 0xfffffffd (0x7f7346e9f000 through
0x7f7446e9effd)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA
Measured bandwidth = 6001.461969 Megabytes/sec
Clear buffer
DDR Sweep FPGA to Host
Measured bandwidth = 6451.031043 Megabytes/sec
Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7f7346e9f000, size = 0xfffffffc3 (0x7f7346e9f000 through
0x7f7446e9efc3)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA
Measured bandwidth = 5979.483548 Megabytes/sec
Clear buffer
DDR Sweep FPGA to Host
Measured bandwidth = 6482.630084 Megabytes/sec
Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7f7346e9f000, size = 0xfffffff9 (0x7f7346e9f000 through
0x7f7446e9eff9)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA
Measured bandwidth = 5993.585033 Megabytes/sec
Clear buffer
DDR Sweep FPGA to Host
Measured bandwidth = 6394.897393 Megabytes/sec
Verifying buffer..
Buffer Verification Success!
```

```
Running fpga_dma_test test on DDR4_C...

Running test in HW mode
Buffer Verification Success!
Buffer Verification Success!
Running DDR sweep test
Buffer pointer = 0x7f836ebd7000, size = 0x40000000 (0x7f836ebd7000 through
0x7f83aebd7000)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA
Measured bandwidth = 2501.897592 Megabytes/sec
Clear buffer
DDR Sweep FPGA to Host
Measured bandwidth = 2582.127891 Megabytes/sec
Verifying buffer..
Buffer Verification Success!
DDR sweep with unaligned pointer and size
Buffer pointer = 0x7f836f5d803d, size = 0x3ffffbe (0x7f836f5d803d through
0x7f83af5d7ffb)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA
Measured bandwidth = 2501.362552 Megabytes/sec
Clear buffer
DDR Sweep FPGA to Host
Measured bandwidth = 2582.615815 Megabytes/sec
Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7f836f5d8003, size = 0x3fffffd (0x7f836f5d8003 through
0x7f83af5d8000)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA
Measured bandwidth = 2500.993490 Megabytes/sec
Clear buffer
DDR Sweep FPGA to Host
Measured bandwidth = 2582.841768 Megabytes/sec
Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7f836f5d8007, size = 0x3ffffff6 (0x7f836f5d8007 through
0x7f83af5d7ffd)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA
Measured bandwidth = 2502.907991 Megabytes/sec
Clear buffer
DDR Sweep FPGA to Host
Measured bandwidth = 2581.905891 Megabytes/sec
Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7f836f5d8000, size = 0x3fffffd (0x7f836f5d8000 through
0x7f83af5d7ffd)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA
Measured bandwidth = 2500.809100 Megabytes/sec
Clear buffer
DDR Sweep FPGA to Host
Measured bandwidth = 2582.694133 Megabytes/sec
Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7f836f5d8000, size = 0x3fffffc3 (0x7f836f5d8000 through
0x7f83af5d7fc3)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA
Measured bandwidth = 2500.802500 Megabytes/sec
Clear buffer
DDR Sweep FPGA to Host
Measured bandwidth = 2583.019243 Megabytes/sec
```

```
Verifying buffer..
Buffer Verification Success!
Buffer pointer = 0x7f836f5d8000, size = 0x3ffffff9 (0x7f836f5d8000 through
0x7f83af5d7ff9)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA
Measured bandwidth = 2501.453270 Megabytes/sec
Clear buffer
DDR Sweep FPGA to Host
Measured bandwidth = 2583.127684 Megabytes/sec
Verifying buffer..
Buffer Verification Success!
Running fpga_dma_test test on QDR...

Running test in HW mode
Buffer Verification Success!
Buffer Verification Success!
Running DDR sweep test
Buffer pointer = 0x7ff375fb3000, size = 0x1000000 (0x7ff375fb3000 through
0x7ff376fb3000)
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA
Measured bandwidth = 936.309401 Megabytes/sec
Clear buffer
DDR Sweep FPGA to Host
Measured bandwidth = 915.727217 Megabytes/sec
Verifying buffer..
Buffer Verification Success!
Finished Executing DMA Tests

Built-in Self-Test Completed.
```