

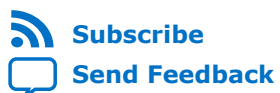


# 40Gbps Ethernet Accelerator Functional Unit (AFU) Design Example User Guide

---

## Intel® Programmable Acceleration Card with Intel® Arria® 10 GX FPGA

Updated for Intel® Acceleration Stack for Intel® Xeon® CPU with FPGAs: **1.2**



**UG-20163 | 2019.04.30**

Latest document on the web: [PDF](#) | [HTML](#)



## Contents

---

<b>1. About this Document.....</b>	<b>3</b>
1.1. Intended Audience.....	3
1.2. Conventions.....	3
1.3. Acronym List.....	3
1.4. Acceleration Glossary.....	4
<b>2. Overview.....</b>	<b>5</b>
2.1. 40GbE Design Example AFU Hardware.....	6
2.2. 40GbE Design Example AFU Software.....	6
<b>3. Running the Design Example Tests.....</b>	<b>7</b>
3.1. Setup Prerequisites.....	7
3.2. Running 40GbE Internal Loopback Test in Single Intel PAC System.....	8
3.3. Running 40GbE External Loopback Test in Single Intel PAC System.....	9
3.4. Running 40GbE Intel PAC-to-PAC Test between two connected Intel PACs.....	11
<b>4. Using the Design Example as a Platform for Further Evaluation.....</b>	<b>15</b>
4.1. Prerequisite while Evaluating with the Intel FPGA MAC/PHY IP.....	15
4.2. Evaluation with an Alternate MAC/PHY IP.....	16
<b>5. 40Gbps Ethernet AFU Design Example User Guide Archives.....</b>	<b>17</b>
<b>6. Document Revision History for 40Gbps Ethernet AFU Design Example User Guide.....</b>	<b>18</b>

## 1. About this Document

This document provides an overview of the 40Gbps Ethernet Accelerator Functional Unit (AFU) design example included in the Intel® Acceleration Stack for Intel Xeon® CPU with FPGAs and instructions to quickly evaluate the network port capability of the Intel Programmable Acceleration Card with Intel Arria® 10 GX FPGA.

### 1.1. Intended Audience

This document is intended for AFU developers and systems engineers to use as a quick start guide for evaluating AFU design and system integration of the network port feature on the Intel PAC with Intel Arria 10 GX FPGA.

### 1.2. Conventions

**Table 1. Document Conventions**

Convention	Description
#	If this symbol precedes a command, enter the command as a root.
\$	If this symbol precedes a command, enter the command as a user.
This font	Indicates file names, commands, and keywords. The font also indicates long command lines. For long command lines, press Enter only if the next line starts a new command, where the # or \$ character denotes the start of the next command.
<variable_name>	Indicates placeholder text that you must replace with appropriate values. Do not include the angle brackets.

### 1.3. Acronym List

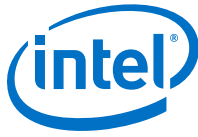
**Table 2. Acronyms List**

Acronyms	Expansion	Description
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	Accelerator Function	Compiled Hardware Accelerator image implemented in FPGA logic that accelerates an application. An AFU and associated AFs are also referred as GBS (Green-Bits, Green BitStream) in the Acceleration Stack installation directory tree and in source code comments.
API	Application Programming Interface	A set of subroutine definitions, protocols, and tools for building software applications.
ASE	AFU Simulation Environment	Co-simulation environment that allows you to use the same host application and AF in a simulation environment. ASE is part of the Intel Acceleration Stack for FPGAs.

*continued...*

Intel Corporation. All rights reserved. Agilix, Altera, Arria, Cyclone, Enpirion, Intel, the Intel logo, MAX, Nios, Quartus and Stratix words and logos are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries. Intel warrants performance of its FPGA and semiconductor products to current specifications in accordance with Intel's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Intel assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Intel. Intel customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.

\*Other names and brands may be claimed as the property of others.



Acronyms	Expansion	Description
CCI-P	Core Cache Interface	CCI-P is the standard interface that AFUs use to communicate with the host.
FIU	FPGA Interface Unit	FIU is a platform interface layer that acts as a bridge between platform interfaces like PCIe*, UPI, and AFU-side interfaces such as CCI-P.
FIM	FPGA Interface Manager	The FPGA hardware containing the FPGA Interface Unit (FIU) and external interfaces such as interfaces for memory, and networking. The FIM is also referred as BBS (Blue-Bits, Blue BitStream) in the Acceleration Stack installation directory tree and in source code comments. The AF interfaces with the FIM at run time.
NLB	Native Loopback	The NLB performs reads and writes to the CCI-P link to test connectivity and throughput.
OPAE	Open Programmable Acceleration Engine	The OPAE is a software framework for managing and accessing AFs.
HSSI	High Speed Serial Interface	This is a reference to the multi-gigabit serial transceiver I/O in the FIM and the corresponding interface to the AFU.
PR	Partial Reconfiguration	The ability to dynamically reconfigure a portion of an FPGA while the remaining FPGA design continues to function.

## 1.4. Acceleration Glossary

**Table 3. Acceleration Stack for Intel Xeon CPU with FPGAs Glossary**

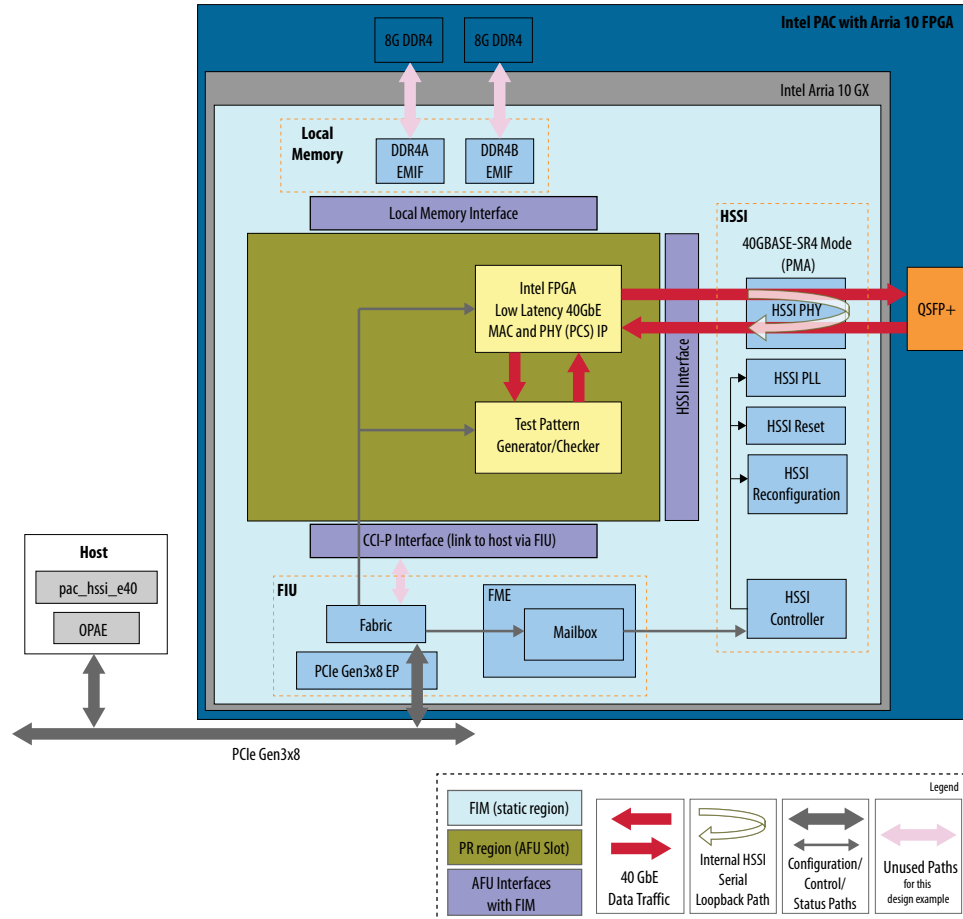
Term	Abbreviation	Description
Intel Acceleration Stack for Intel Xeon CPU with FPGAs	Acceleration Stack	A collection of software, firmware and tools that provides performance-optimized connectivity between an Intel FPGA and an Intel Xeon processor.
Intel Programmable Acceleration Card with Intel Arria 10 GX FPGA	Intel PAC with Intel Arria 10 GX FPGA	PCIe accelerator card with an Intel Arria 10 FPGA. Programmable Acceleration Card is abbreviated PAC. Contains an FPGA Interface Manager (FIM) that pairs with an Intel Xeon processor over PCIe bus.
Intel Xeon Scalable Platform with Integrated FPGA	Integrated FPGA Platform	Intel Xeon plus FPGA platform with the Intel Xeon and an FPGA in a single package and sharing a coherent view of memory via the Ultra Path Interconnect (UPI).
OPAE_PLATFORM_ROOT		A Linux shell environment variable set up during the process of installing the OPAE SDK delivered with the Acceleration Stack.

## 2. Overview

The 40Gbps Ethernet (40GbE) AFU design example in the Acceleration Stack installation allows you to evaluate the network port capabilities of the Intel PAC with Intel Arria 10 GX FPGA. The 40GbE AFU design example contains a 40GbE MAC instance with traffic generation and checking logic to send and receive ethernet packets on the QSFP+ network port. The Acceleration Stack installation includes OPAE tools, APIs and a sample host application to initialize and start packet transfers from the host, and subsequently retrieve port statistics.

This design example supports internal HSSI transceiver loopback, external QSFP+ port loopback, and Intel PAC-to-PAC modes of operation.

Figure 1. System Block Diagram



Intel Corporation. All rights reserved. Agilx, Altera, Arria, Cyclone, Enpirion, Intel, the Intel logo, MAX, Nios, Quartus and Stratix words and logos are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries. Intel warrants performance of its FPGA and semiconductor products to current specifications in accordance with Intel's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Intel assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Intel. Intel customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.

\*Other names and brands may be claimed as the property of others.

## 2.1. 40GbE Design Example AFU Hardware

The design example uses the Intel FPGA 40G Ethernet MAC and PHY IP core to send and receive 40GbE ethernet packets on the Intel PAC's QSFP+ network port. The design example supports generating and checking all network traffic data on the Intel PAC only through the implemented traffic generation and checking module in the AFU.

- The MAC IP instance connects to the HSSI PHY using the HSSI device class interface defined by OPAE. For more information about HSSI interface and 40G Ethernet MAC and PHY IP core connection, refer to the *Networking Interface for Open Programmable Acceleration Engine: Intel Programmable Acceleration Card with Intel Arria 10 GX FPGA*.
- The HSSI PHY implemented in the FIM connects to the FPGA's transceiver I/O.

The HSSI Controller in the FIM utilizes transceiver reconfiguration to set the desired mode of the HSSI PHY. The design example requires that the host set the HSSI PHY mode to 40GBASE-SR4 (PMA only).

The design example utilizes the PR Management Interface ports on the `hssi` interface for internal purpose. Intel recommends you to terminate these ports in your AFU designs, refer to the *Networking Interface for Open Programmable Acceleration Engine: Intel Programmable Acceleration Card with Intel Arria 10 GX FPGA* for more details.

**Note:** The Intel Acceleration Stack version 1.1 supports PHY modes of 4x10GBASE-SR (PCS/PMA) and 40GBASE-SR4 (PMA only).

Use OPAE tools and APIs from the host to initialize and control packet transfers, and collect port statistics.

### Related Information

- [Low Latency 40-Gbps Ethernet IP Core User Guide](#)
- [Networking Interface for Open Programmable Acceleration Engine: Intel Programmable Acceleration Card with Intel Arria 10 GX FPGA](#)

## 2.2. 40GbE Design Example AFU Software

Use the OPAE driver, tools, APIs, and sample host application to configure the HSSI PHY mode, initialize and control packet transfers, and collect port statistics with the design example AFU. For more information, refer to the following README file in the OPAE SDK installation:

```
$OPAE_PLATFORM_ROOT/hw/samples/eth_e2e_e40/sw/README.md
```

For more information about managing the network port feature from the host using the OPAE driver, refer to the *Networking Interface for Open Programmable Acceleration Engine: Intel Programmable Acceleration Card with Intel Arria 10 GX FPGA*.

### Related Information

- [Open Programmable Acceleration Engine \(OPAE\) Tools Guide](#)
- [Networking Interface for Open Programmable Acceleration Engine: Intel Programmable Acceleration Card with Intel Arria 10 GX FPGA](#)

## 3. Running the Design Example Tests

In the `$OPAE_PLATFORM_ROOT/hw/samples` directory, there are two reference AFUs containing packet generation—`eth_e2e_e10` (10G Ethernet), and `eth_e2e_e40` (40G Ethernet). These AFUs contain packet generators and can be exercised by the sample OPAE host application located in the `sw` subdirectory.

### 3.1. Setup Prerequisites

To install the Intel PAC and OPAE SDK on a supported platform, follow the *Intel Acceleration Stack Quick Start Guide for Intel Programmable Acceleration Card with Intel Arria 10 GX FPGA*. If you only want to evaluate network port operation using the pre-compiled AFs from the OPAE SDK installation, you do not need the Intel Quartus® Prime Pro Edition software.

The `OPAE_PLATFORM_ROOT` environment variable points to the location where you installed the OPAE SDK, which is delivered as part of the Acceleration Stack for Intel PAC with Intel Arria 10 GX FPGA.

In a multi-card system, use the following command to find the device sysfs entry associated with the desired PCIe device:

```
$ ls -lrt /sys/class/fpga/intel-fpga-dev.*/
```

Sample output:

```
/sys/class/fpga/intel-fpga-dev.1/:
total 0
-rw-r--r--. 1 root root 4096 Apr  7 21:46 uevent
drwxr-xr-x. 4 root root    0 Apr  7 21:46 intel-fpga-port.1
drwxr-xr-x. 12 root root    0 Apr  7 21:46 intel-fpga-fme.1
lrwxrwxrwx. 1 root root    0 Apr  7 21:46 subsystem -> ../../../../../../class/
fpga
drwxr-xr-x. 2 root root    0 Apr  7 21:46 power
lrwxrwxrwx. 1 root root    0 Apr  7 21:49 device -> ../../../../0000:af:00.0

/sys/class/fpga/intel-fpga-dev.0/:
total 0
lrwxrwxrwx. 1 root root    0 Apr  7 21:46 subsystem -> ../../../../../../class/
fpga
lrwxrwxrwx. 1 root root    0 Apr  7 21:49 device -> ../../../../0000:86:00.0
```



```
drwxr-xr-x. 12 root root    0 Apr  7 23:27 intel-fpga-fme.0
-rw-r--r--.  1 root root 4096 Apr  7 23:27 uevent
drwxr-xr-x.  2 root root    0 Apr  7 23:27 power
drwxr-xr-x.  4 root root    0 Apr  7 23:27 intel-fpga-port.0
```

This shows that the dev1 PCIe B:D:F is af:00.0 and dev0 PCIe B:D:F is 86:00.0.

### Related Information

[Intel Acceleration Stack Quick Start Guide for Intel Programmable Acceleration Card with Intel Arria 10 GX FPGA](#)

For more information about installation of the software and licensing requirements.

## 3.2. Running 40GbE Internal Loopback Test in Single Intel PAC System

1. Load the AF for the 40GbE AFU example.

```
$ cd $OPAE_PLATFORM_ROOT
$ sudo fpgaconf hw/samples/eth_e2e_e40/bin/eth_e2e_e40.gbs
```

2. `cd $OPAE_PLATFORM_ROOT/hw/samples/eth_e2e_e40/sw`
3. Run the following steps on your Intel PAC:

- a. Compile the library and application using the command:

```
$ make
```

- b. To configure the transceiver channel into 40G mode, write **40** to the following `sysfs` entry:

```
$ sudo sh -c "echo 40 > /sys/class/fpga/intel-fpga-dev.<instance_id>\
/intel-fpga-fme.<instance_id>/intel-pac-hssi.<instance_id>.\
auto/hssi_mgmt/config"
```

*<instance\_id>* represents the consecutive numbering of device, fme, and hssi instances.

For example:

```
sudo sh -c "echo 40 > /sys/class/fpga/intel-fpga-dev.0\
/intel-fpga-fme.0/intel-pac-hssi.2.auto/hssi_mgmt/config"
```

- c. To allow non-root users to access the 40GbE AFU instance, you can provide read and write privileges to the port (`/dev/intel-fpga-port.\*`) where `\*` denotes the respective socket. For example, to provide read and write privileges on Port 0:

```
$ sudo chmod 666 /dev/intel-fpga-port.0
```

- d. To resolve library dependency:

```
export LD_LIBRARY_PATH=`pwd`:LD_LIBRARY_PATH
```

- e. To enable the internal loopback on B:D:F - 00:0a:0b,

```
$ ./pac_hssi_e40 -b 00 -d 0a -f 0b --action=loopback_enable
```





- f. To clear PHY, transmit, and receive statistics:

```
$ ./pac_hssi_e40 -b 00 -d 0a -f 0b --action=stat_clear
```

Sample output:

```
Cleared TX stats on channel 0  
Cleared RX stats on channel 0
```

- g. To transmit 0x1000 packets:

```
$ ./pac_hssi_e40 -b 00 -d 0a -f 0b --action=pkt_send
```

Sample output:

```
Sent 0x10000 packets on channel 0
```

*Note:* After programming the eth\_e2e\_e40 AFU, the initial send of packets may drop the first packet. Subsequent packet sends do not drop any packets.

- h. To get PHY, transmit and receive statistics:

```
$ ./pac_hssi_e40 -b 00 -d 0a -f 0b --action=stat
```

To find the instance id associated with your device:

```
$ ls /sys/class/fpga/
```

For more details, refer to the README file located in the sw subdirectory to:

```
$OPAE_PLATFORM_ROOT/hw/samples/eth_e2e_e40/sw/README.md
```

To run this example on a virtual machine:

- Program the eth\_e2e\_e40 AFU and configure the transceiver channel to 10G mode from the Host machine by referencing the previous substeps.
- Follow the steps in the [Running the OPAE in a Virtualized Environment](#) section of the *Intel Acceleration Stack Quick Start Guide for Intel Programmable Acceleration Card with Intel Arria 10 GX FPGA* to create a virtual function and attach the virtual function to a virtual machine.
- Run the internal loopback test on the virtual machine.

### 3.3. Running 40GbE External Loopback Test in Single Intel PAC System

The setup and output from the commands in the external loopback test are similar to the internal loopback test. The only difference is that the traffic loopback is established after the Intel PAC's QSFP+ network port.

1. Loopback the generated network traffic at the Intel PAC's external QSFP+ network port. You can accomplish this loopback in several ways:



- installing a QSFP+ optical module loopback adapter, or
  - installing a QSFP+ optical module with MPO connection and looping back through:
    - an inserted fiber loopback plug, or
    - external network equipment
2. Load the AF for the 40GbE AFU example (if the AF is not already loaded).

```
$ cd $OPAE_PLATFORM_ROOT
$ sudo fpgaconf hw/samples/eth_e2e_e40/bin/eth_e2e_e40.gbs
```

3. `cd $OPAE_PLATFORM_ROOT/hw/samples/eth_e2e_e40/sw`
4. Run the following steps on your Intel PAC:

- a. Compile the library and application using the command:

```
$ make
```

- b. To configure the transceiver channel into 40G mode, write **40** to the following `sysfs` entry:

```
$ sudo sh -c "echo 40 > /sys/class/fpga/intel-fpga-dev.<instance_id>\
/intel-fpga-fme.<instance_id>/intel-pac-hssi.<instance_id>.\
auto/hssi_mgmt/config"
```

*<instance\_id>* represents the consecutive numbering of device, fme, and hssi instances.

For example:

```
sudo sh -c "echo 40 > /sys/class/fpga/intel-fpga-dev.0\
/intel-fpga-fme.0/intel-pac-hssi.2.auto/hssi_mgmt/config"
```

- c. To allow non-root users to access the 40GbE AFU instance, you can provide read and write privileges to the port (`/dev/intel-fpga-port.\*`) where `\*` denotes the respective socket. For example, to provide read and write privileges on Port 0:

```
$ sudo chmod 666 /dev/intel-fpga-port.0
```

- d. To resolve library dependency:

```
export LD_LIBRARY_PATH=`pwd`:LD_LIBRARY_PATH
```

- e. To disable the internal loopback on B:D:F - 00:0a:0b,

```
$ ./pac_hssi_e40 -b 00 -d 0a -f 0b --action=loopback_disable
```

- f. To trigger the DFE (Decision feedback equalizer):

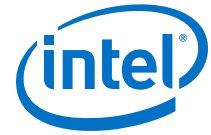
```
sudo sh -c "echo 1 > /sys/class/fpga/intel-fpga-dev.0/intel-fpga-fme.0/\
intel-pac-hssi.2.auto/hssi_mgmt/dfc_kickstart"
```

Verify that some of the DFE tap values are non-zero. This ensures that the script run is successful.

```
cat /sys/class/fpga/intel-fpga-dev.0/intel-fpga-fme.0/intel-pac-
hssi.2.auto/hssi_mgmt/dfc_kickstart
```

- g. To clear PHY, transmit, and receive statistics:

```
$ ./pac_hssi_e40 -b 00 -d 0a -f 0b --action=stat_clear
```



Sample output:

```
Cleared TX stats on channel 0  
Cleared RX stats on channel 0
```

- h. To transmit 0x1000 packets:

```
$ ./pac_hssi_e40 -b 00 -d 0a -f 0b --action=pkt_send
```

Sample output:

```
Sent 0x10000 packets on channel 0
```

*Note:* After programming the eth\_e2e\_e40 AFU, the initial send of packets may drop the first packet. Subsequent packet sends do not drop any packets.

- i. To get PHY, transmit and receive statistics:

```
$ ./pac_hssi_e40 -b 00 -d 0a -f 0b --action=stat
```

*Note:* After every hot plug/unplug of the cables, you must trigger the DFE as discussed above after disabling internal loopback.

For more details, refer to the README file located in the sw subdirectory to:

```
$OPAE_PLATFORM_ROOT/hw/samples/eth_e2e_e40/sw/README.md
```

To run this example on a virtual machine:

- Program the eth\_e2e\_e40 AFU and configure the transceiver channel to 40G mode from the Host machine by referencing the previous substeps.
- Disable internal loopback and trigger DFE from the Host by referencing the previous substeps.
- Follow the steps in the [Running the OPAE in a Virtualized Environment](#) section of the *Intel Acceleration Stack Quick Start Guide for Intel Programmable Acceleration Card with Intel Arria 10 GX FPGA* to create a virtual function and attach the virtual function to a virtual machine.
- Run the external loopback test on the virtual machine.

### 3.4. Running 40GbE Intel PAC-to-PAC Test between two connected Intel PACs

In this procedure, you can install the Intel PACs in the same system or two separate systems with the Acceleration Stack. Unless shown otherwise, you can expect the commands to return similar outputs as the internal loopback test.

Figure 2. System Setup with Intel PACs Installed on Same Host

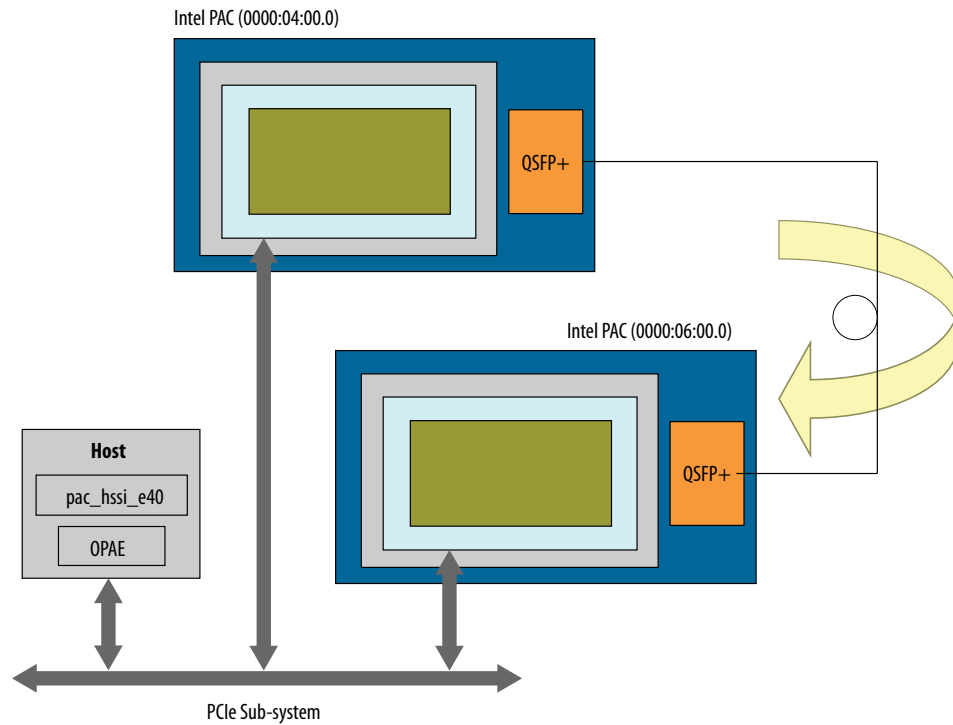
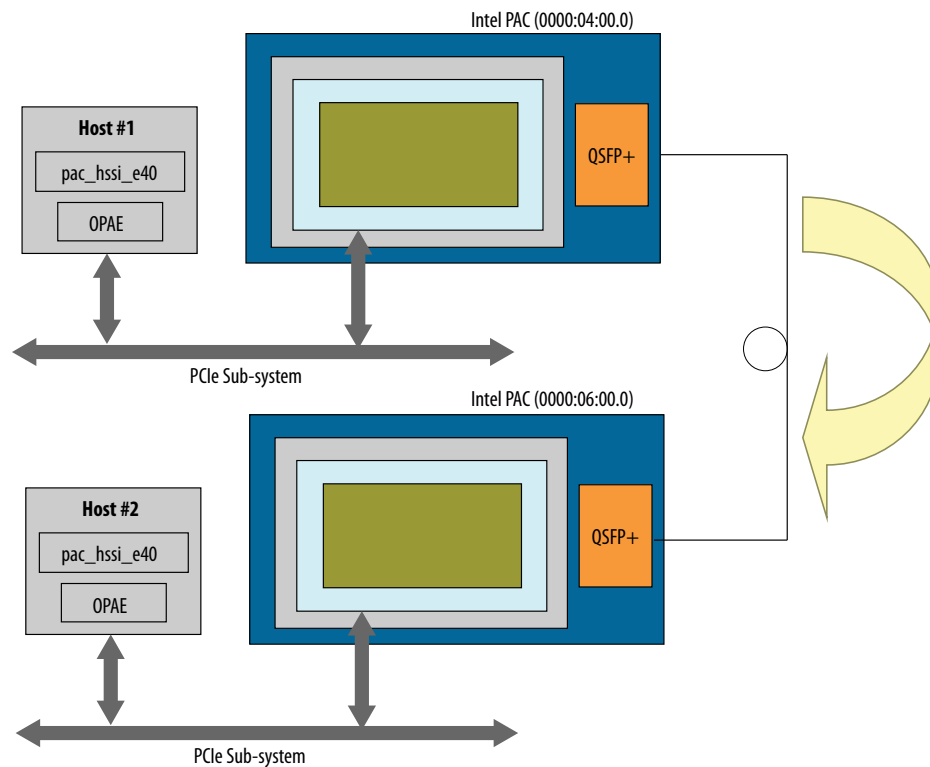
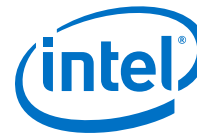


Figure 3. System Setup with Intel PACs Installed on Separate Hosts





1. Install a QSFP+ optical module in each Intel PAC and connect the QSFP+ ports with an optical cable.
2. Assuming the two Intel PACs are installed in the same system, find their PCI **Bus:Device:Function** mappings.

```
$ lspci | grep 09c4
```

Sample output:

```
04:00.0 Processing accelerators: Intel Corporation Device 09c4
06:00.0 Processing accelerators: Intel Corporation Device 09c4
```

3. Load the AF for the 40GbE AFU example on both Intel PACs.

```
$ cd $OPAE_PLATFORM_ROOT
$ sudo fpgaconf hw/samples/eth_e2e_e40/bin/eth_e2e_e40.gbs -b 0x04
$ sudo fpgaconf hw/samples/eth_e2e_e40/bin/eth_e2e_e40.gbs -b 0x06
```

4. `cd $OPAE_PLATFORM_ROOT/hw/samples/eth_e2e_e40/sw`

5. Run the following steps on your both Intel PAC:

- a. Compile the library and application using the command:

```
$ make
```

- b. To configure the transceiver channel into 40G mode, write **40** to the following `sysfs` entry:

```
$ sudo sh -c "echo 40 > /sys/class/fpga/intel-fpga-dev.<instance_id>\
/intel-fpga-fme.<instance_id>/intel-pac-hssi.<instance_id>.\
auto/hssi_mgmt/config"
```

`<instance_id>` represents the consecutive numbering of device, fme, and hssi instances.

For example:

```
sudo sh -c "echo 40 > /sys/class/fpga/intel-fpga-dev.0\
/intel-fpga-fme.0/intel-pac-hssi.2.auto/hssi_mgmt/config"
```

- c. To allow non-root users to access the 40GbE AFU instance, you can provide read and write privileges to the port (`/dev/intel-fpga-port.*`) where `*` denotes the respective socket. For example, to provide read and write privileges on Port 0:

```
$ sudo chmod 666 /dev/intel-fpga-port.0
```

- d. To resolve library dependency:

```
export LD_LIBRARY_PATH=`pwd`:${LD_LIBRARY_PATH}
```

- e. To disable the internal loopback on B:D:F - 00:0a:0b,

```
$ ./pac_hssi_e40 -b 00 -d 0a -f 0b --action=loopback_disable
```

- f. To trigger the DFE:

```
sudo sh -c "echo 1 > /sys/class/fpga/intel-fpga-dev.0/intel-fpga-fme.0/\
intel-pac-hssi.2.auto/hssi_mgmt/dfe_kickstart"
```



Verify that some of the DFE tap values are non-zero. This ensures that the script run is successful.

```
cat /sys/class/fpga/intel-fpga-dev.0/intel-fpga-fme.0/intel-pac-hssi.2.auto/hssi_mgmt/dfc_kickstart
```

- g. To clear PHY, transmit, and receive statistics:

```
$ ./pac_hssi_e40 -b 00 -d 0a -f 0b --action=stat_clear
```

Sample output:

```
Cleared TX stats on channel 0  
Cleared RX stats on channel 0
```

- h. To transmit 0x1000 packets:

```
$ ./pac_hssi_e40 -b 00 -d 0a -f 0b --action=pkt_send
```

Sample output:

```
Sent 0x10000 packets on channel 0
```

**Note:** After programming the eth\_e2e\_e40 AFU, the initial send of packets may drop the first packet. Subsequent packet sends do not drop any packets.

**Note:** In `pac_hssi_e40 [-h] [-b <bus>] [-d <device>] [-f <function>] [-m Dest. MAC] -a action`, the Dest (destination) MAC address is user configurable. By default, the broadcast address is used as the destination MAC address.

- i. To get PHY, transmit and receive statistics:

```
$ ./pac_hssi_e40 -b 00 -d 0a -f 0b --action=stat
```

**Note:** After every hot plug/unplug of the cables, you must trigger the DFE as discussed above after disabling internal loopback.

For more details, refer to the README file located in the `sw` subdirectory to:

```
$OPAE_PLATFORM_ROOT/hw/samples/eth_e2e_e40/sw/README.md
```

To run this example on a virtual machine:

- Program the eth\_e2e\_e40 AFU and configure the transceiver channel to 40G mode from the Host machine by referencing the previous substeps.
- Disable internal loopback and trigger DFE from the Host by referecing the previous substeps.
- Follow the steps in the [Running the OPAE in a Virtualized Environment](#) section of the *Intel Acceleration Stack Quick Start Guide for Intel Programmable Acceleration Card with Intel Arria 10 GX FPGA* to create a virtual function and attach the virtual function to a virtual machine.
- Run the external loopback test on the virtual machine.

## 4. Using the Design Example as a Platform for Further Evaluation

---

Use the 40GbE AFU design example to perform further evaluation with the Intel FPGA MAC/PHY IP, third party IP, or your own MAC/PHY IP. The example design source is located in the same location as the sample AFUs included in the OPAE SDK installation:

```
$OPAE_PLATFORM_ROOT/hw/samples/eth_e2e_e40
```

The RTL source for the example is at the following location:

```
$OPAE_PLATFORM_ROOT/hw/samples/eth_e2e_e40/hw/rtl
```

While recompiling the example AFU to regenerate an AF (.gbs), you require an installed version of the Intel Quartus Prime Pro Edition (version 17.1.1) software.

OPAE version 1.0.2 does not support the ASE flow for HSSI interfaces.

### Related Information

- [Intel Acceleration Stack Quick Start Guide for Intel Programmable Acceleration Card with Intel Arria 10 GX FPGA](#)  
For more information about installation of the software and licensing requirements.
- [Accelerator Functional Unit \(AFU\) Developer's Guide](#)  
For more information about the OPAE SDK design flow for AFUs that target the Intel PAC with Intel Arria 10 GX FPGA.

### 4.1. Prerequisite while Evaluating with the Intel FPGA MAC/PHY IP

In addition to the Intel licensing requirements for Intel Quartus Prime Pro Edition and Intel FPGA IP specified in the *Intel Acceleration Stack Quick Start Guide for Intel Programmable Acceleration Card with Intel Arria 10 GX FPGA*, the regeneration of AFs for the 40GbE design example with the Intel FPGA MAC IP/PHY also requires the following license:

*IP-40GEUMACPHY Low Latency 40G MAC/PHY*

### Related Information

[Intel Acceleration Stack Quick Start Guide for Intel Programmable Acceleration Card with Intel Arria 10 GX FPGA](#)

For more information about installation of the software and licensing requirements.

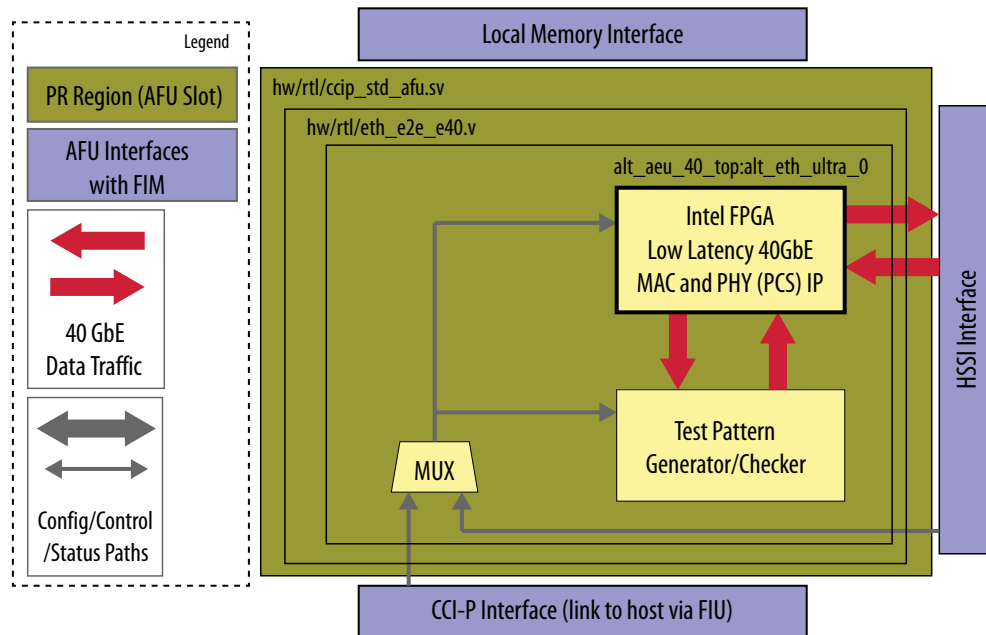
## 4.2. Evaluation with an Alternate MAC/PHY IP

You can use the 40GbE AFU example as a framework to evaluate MAC/PHY IP from third parties or your own IP. The design example instantiates the Intel FPGA MAC/PHY IP in the following AFU RTL source file:

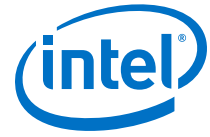
```
OPAE_PLATFORM_ROOT/hw/samples/eth_e2e_e40/hw/rtl/eth_e2e_e40.v
```

Replace the `alt_eth_ultra_0` instance shown in this figure with your own MAC/PHY IP instance. You must provide any necessary wrapper shim logic to integrate your IP within the AFU design example framework.

**Figure 4. 40GbE AFU Design Example Hierarchy**







## 5. 40Gbps Ethernet AFU Design Example User Guide Archives

---

Intel Acceleration Stack Version	User Guide (PDF)
1.1	<a href="#">40Gbps Ethernet Accelerator Functional Unit (AFU) Design Example User Guide: For Intel Programmable Acceleration Card with Intel Arria 10 GX FPGA</a>



## 6. Document Revision History for 40Gbps Ethernet AFU Design Example User Guide

---

Document Version	Intel Acceleration Stack Version	Changes
2019.04.30	1.2 (supported with Intel Quartus Prime Pro Edition 17.1.1)	Added information about how to find the instance id in section: <i>Setup Prerequisites</i> .
2019.01.02	1.2 (supported with Intel Quartus Prime Pro Edition 17.1.1)	Minor edits.
2018.12.04	1.2 (supported with Intel Quartus Prime Pro Edition 17.1.1)	<ul style="list-style-type: none"> <li>• Updated steps in: <ul style="list-style-type: none"> <li>– <i>Running 40GbE Internal Loopback Test in Single PAC System</i></li> <li>– <i>Running 40GbE External Loopback Test in Single PAC System</i></li> <li>– <i>Running 40GbE PAC-to-PAC Test between two connected PACs</i></li> </ul> </li> <li>• Added a new chapter: <i>40Gbps Ethernet AFU Design Example User Guide Archives</i></li> </ul>
2018.08.06	1.1 (supported with Intel Quartus Prime Pro Edition 17.1.1)	Initial release.