Cornelis™ Omni-Path Express™ Host Fabric Interface Platform Configuration

Reference Guide

November 2023
Doc. No. 562586, Rev. 5.0

You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Cornelis Networks products described herein. You agree to grant Cornelis Networks a non-exclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

All product plans and roadmaps are subject to change without notice.

The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Cornelis Networks technologies may require enabled hardware, software, or service activation.

No product or component can be absolutely secure.

Your costs and results may vary.

Cornelis, Cornelis Networks, Omni-Path, Omni-Path Express, and the Cornelis Networks logo belong to Cornelis Networks, Inc. Other names and brands may be claimed as the property of others.

Copyright © 2023 Cornelis Networks, Inc.

Table of Contents

Revision History

Date	Revision	Description
Nov 2023	5.0	Update branding and minor corrections throughout.
Jan 2017	4.0	Updates to Section 4 “Creating a Configuration File” section.
Dec 2016	3.0	Document updated for Revision 3.0.
Feb 2016	2.0	Added programming EEPROM information to “Loading the Platform Configuration”.
Nov 2015	1.0	Document updated for Revision 1.0.

1. Preface

This guide is part of the documentation set for the Cornelis Omni-Path Express (OPX) Fabric, which is an end-to-end solution consisting of Cornelis Omni-Path Express Host Fabric Interface Adapters (HFIs), Cornelis Omni-Path Express Edge Switches, Cornelis Omni-Path Express Director Class Switches, and fabric management and development tools.

The Cornelis Omni-Path Express Fabric delivers the next generation, High Performance Computing (HPC) network solution that is designed to cost-effectively meet the growth, density, and reliability requirements of large-scale HPC clusters.

Both the OPX Fabric and standard InfiniBand (IB) can send Internet Protocol (IP) traffic over the fabric, or *IPoFabric*. In this document it may also be referred to as *IP over IB* or *IPoIB*. From a software point of view, IPoFabric behaves the same way as IPoIB, and in fact uses an ib_ipoib driver to send IP traffic over the ib0/ib1 ports.

1.1. Intended Audience

The intended audience for the OPX document set is network administrators and other qualified personnel.

1.2. Documentation Library

All Cornelis Networks product documentation may be found in the Release Library located in the Cornelis Customer Center .

To help guide you with the identity of the documents related to the task you are wanting to accomplish, refer to the Document Library table in the *Cornelis Omni-Path Express Fabric Quick Start Guide*.

1.3. Document Conventions

The following conventions are standard for Cornelis Omni-Path Express documentation:

- **Note:** provides additional information.
- **Caution:** indicates the presence of a hazard that has the potential of causing damage to data or equipment.
- **Warning:** indicates the presence of a hazard that has the potential of causing personal injury.
- Text in blue font indicates a hyperlink (jump) to a figure, table, or section in this guide. Links to websites are also shown in blue. For example:

  See Section 1.6 “License Agreements” for more information.

  For more information, visit www.cornelisnetworks.com.
- Text in **bold** font indicates user interface elements such as menu items, buttons, check boxes, key names, keystrokes, or column headings. For example:

Click the **Start** button, point to **Programs**, point to **Accessories**, and then click **Command Prompt**.

Press **CTRL+P** and then press the **UP ARROW** key.

- Text in `Courier` font indicates a file name, directory path, or command line text. For example:

  Enter the following command: `sh ./install.bin`

- Text in *italics* indicates terms, emphasis, variables, or document titles. For example:

  Refer to *Cornelis Omni-Path Express Fabric Software Installation Guide* for details.

  In this document, the term *chassis* refers to a managed switch.

Procedures and information may be marked with one of the following qualifiers:

- **(Linux)** – Tasks are only applicable when Linux is being used.
- **(Host)** – Tasks are only applicable when OPX Host Software or OPXS is being used on the hosts.
- **(Switch)** – Tasks are applicable only when OPX Switches or Chassis are being used.
- Tasks that are generally applicable to all environments are not marked.

1.4. Best Practices

Note the following Cornelis recommendations:

- To obtain the most recent functional and security updates, please download and install the latest versions of the OPX software and firmware. These are found in the Cornelis Customer Center.
- To improve security:
  - Administrators should log out users and disable multi-user logins prior to performing provisioning and similar tasks.
  - Update the default HTTPS certificate (refer to the *Cornelis Omni-Path Express Fabric Switches GUI User Guide*, "Updating the Certificate" for details).

1.5. Cornelis Omni-Path Express Fabric Design Generator for Cornelis Omni-Path Express Fabric

The Fabric Design Generator generates sample cluster configurations based on key cluster attributes, including a side-by-side comparison of up to four cluster configurations. The tool also generates parts lists and cluster diagrams.

To access the Fabric Design Generator for OPX Fabric, go to Cornelis™ Omni-Path Express™ Fabric Design Generator.

1.6. License Agreements

Cornelis software and firmware is provided under one or more license agreements. Please refer to the license agreement(s) provided with the software for specific detail. Do not install or use the software until you have carefully read and agree to the terms and conditions of the license agreement(s). By loading or using the software, you agree to the terms of the license agreement(s). If you do not wish to so agree, do not install or use the software.

1.7. Technical Support

Technical support for Cornelis products is available 24 hours a day, 365 days a year. Please contact Cornelis Networks Customer Support by visiting www.cornelisnetworks.com or send your request directly to support@cornelisnetworks.com.

2. Introduction

The configuration file is a communication vehicle between the platform implementer and the Cornelis Omni-Path Express Host Fabric Interface Adapter (OPX HFI) driver, to identify key features of the platform and signal path. These features include identification of the port type, whether the link is using a QSFP cable or backplane, for example, the amount of attenuation in the signal path, and signal integrity settings for optical cables.

Some of the settings in the configuration file, such as Active Optical Cable (AOC) signal integrity settings, are provided by Cornelis and have been determined through validation testing of OPX silicon and compliant channels. Other settings are updated by the platform implementer, based on their platform design and verification testing. The accuracy of these configuration file settings can have an impact on the duration and success rate of link negotiation and initialization. For example, missing settings can result in an inability to link up with Active Optical Cables.

This document describes:

- The features of the platform firmware
- The format of text-based INI files
- The tables and fields in INI files
- How to convert text-based INI files into binary configuration files, and vice-versa
- The interaction of the driver with the platform firmware
- The topics covered include:
  - Access to the configuration file in discrete adapters
  - Access to the configuration file in integrated processors
  - Methods to enable OEMs to provide the configuration file at manufacturing
  - Methods to enable OEMs to update the configuration in the field, if needed

The host fabric software design aims to ensure that the platform configuration data travels with the platform hardware it describes. This avoids a host of problems regarding inventory and support of different configuration files meant for various platforms as well as versioning of the contents. It also avoids Cornelis becoming the single point of contact to inventory and support configuration files for platforms created by OEMs and custom design partners.

Assumptions for the design are documented below:

- Host software stack is expected to support only UEFI BIOS and Legacy Boot Mode.
- An environment (UEFI BIOS) capable of executing, and always executing, the HfiPcieGen3 driver regardless of boot mode. This requires that the OPX HfiPcieGen3 driver not be subjected to Optimized Boot (and other similar features) of platform BIOS that would prevent it from being executed as part of the boot process.

3. Platform Configuration Features

The platform configuration provides a set of features whose behavior is configured using OPX ASIC pins and INI settings.

3.1. Port Types

The platform configuration supports the concept of ports having different types.

A port's type is defined in the text INI file. The link and signal integrity management of a port is highly dependent on its port type. The supported port types are described in the following table.

Table 1. Port Types

Port Type	Description
DISCONNECTED	The OEM design does not use this port
FIXED	A FIXED port is normally associated with static link configuration, where the signal integrity characteristics are a known fixed value. An example of a FIXED port is a backplane port. These ports are referred to as 'Proprietary Channels' in the Cornelis Omni-Path Express Channel Electrical Specification.
VARIABLE	A VARIABLE port is associated with a user-modified link configuration, where the signal integrity settings are a limited set of fixed values. These ports are referred to as 'Proprietary Channels' in the Cornelis Omni-Path Express Channel Electrical Specification.
QSFP	A QSFP port is associated with a QSFP28 connector, where QSFP modules can be dynamically inserted, dynamically removed, and different types of QSFP modules can be installed, depending on end- user fabric considerations. QSFP ports are typically implemented as 'Interoperable Channels', as described in the Cornelis Omni-Path Express Channel Electrical Specification.

3.2. Power Management

There are several platform configuration settings associated with QSFP power management, so that an OEM platform designer can limit the total amount of power consumed by all the installed QSFPs.

The LPMode pin on the QSFP modules should be left disconnected, which will cause the QSFP modules to stay in low power mode, until the driver explicitly enables high power mode through the QSFP's I2C (Inter-Integrated Controller protocol) management interface.

The driver will not put any QSFP into high power mode whose POWER_CLASS (see Byte 129 in SFF-8636) exceeds the SYSTEM.QSFP_POWER_CLASS_MAX setting. This provides a limit on the maximum amount of power that a QSFP can consume.

When a QSFP module is plugged in, the driver will interrogate its POWER_CLASS and TYPE information to determine how much power the QSFP module will actually need.

3.3. QSFP Management

The driver supports QSFP management and takes responsibility for:

- Recognizing that QSFP modules have been inserted/removed
- Performing the necessary QSFP communications needed to initialize, query, and ready the QSFP modules for fabric link operation
- Monitoring QSFP temperatures
- Adjusting QSFP SI related settings based on requests by the local "channel"

Each QSFP module has a set of low speed management signals:

ModSelL — This pin is ignored by the driver.

ResetL — The driver requires a unique ResetL signal for each QSFP port. The driver asserts the ResetL signal to reset a QSFP module which has been inserted, in order to get the QSFP module into a well-defined state.

LPMode — The driver assumes that the platform designer leaves the LPMode pin on the QSFP modules disconnected, which will cause the QSFP modules to stay in low power mode, until the driver explicitly enables high power mode through the QSFP's I2C management interface.

ModPrsL — The driver requires a unique ModPrsL signal for each QSFP port and uses this signal to detect the insertion and removal of the QSFP module in that port.

IntL — The driver requires a unique IntL signal for each QSFP port to detect an interrupt condition from the QSFP module in that port.

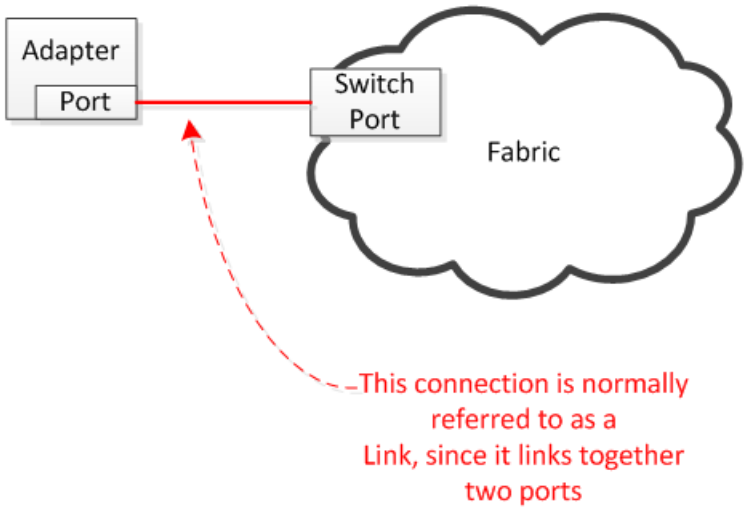
The driver also requires I2C communication to each QSFP module.

3.4. Signal Integrity Tuning

3.4.1. Background and Nomenclature

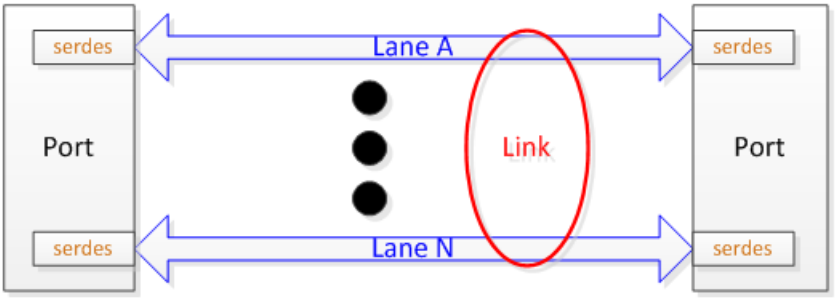
Adapters and switches have ports. Ports on switches and adapters are numbered starting with '1'. The ASIC supports up to 1 port per HFI for a maximum of 2 ports for a dual-ported CHF card. The connection between two ports is referred to as a "link" as shown below.

Figure 1. Link



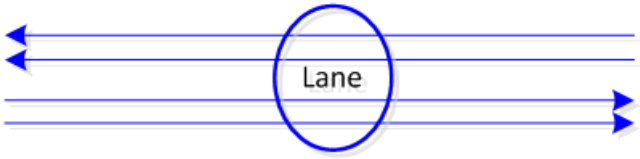
A link supports 1 or more "lanes". Each lane is a bi-directional communication channel. Lanes are typically lettered starting with 'A'. The ASIC supports 4 lanes per port.

Figure 2. Lanes



A lane is normally made up of 4 wires, 2 wires in each direction (an RX pair and a TX pair). Each set is called a differential pair.

Figure 3. Differential Pair



A SerDes (serializer – deserializer) is used per lane. The serializer converts a digital 0/1 into a voltage presented onto the TX differential pair. The deserializer converts the voltage appearing on the RX differential pairs into digital 0/1. As multiple bits are sent, waveforms are produced as each "wire" transitions between voltages that represent digital 0s and digital 1s.

The SerDes transmitter and receiver each use analog waveform shaping to better 'tune' the waveform being sent and received, to counteract the signal-degrading effects of the signal path through the system. Platform implementers may use the INI file to provide hints about the channel to the SerDes and to decrease the duration of the SerDes optimization process during link negotiation and initialization (LNI). These ballpark tunings are referred to as presets. Additionally, in cases where external active devices are present in the signal path, such as an active optical cable (AOC), the INI file is used to specify the initial signal integrity settings for the external active device.

3.4.2. INI Based Tuning Tables

The platform configuration supports TxPreset and RxPreset tables for specifying signal integrity tunings for active external QSFP cables. For backplane or passive copper QSFP cables, the INI file defines the attenuation of the channel, as a starting point hint for the SerDes in its link tuning algorithm.

The TxPresets relate to the ASIC TX electrical channel, and the RxPresets relate to the ASIC RX electrical channel. Each TxPreset entry includes settings for the ASIC SerDes TX logic, as well as the "electrical input side" settings for the local QSFP device that is receiving this TX signal. Each RxPreset entry specifies the electrical output side settings for the local QSFP device that is transmitting a signal that is received by the ASIC RX pins.

The platform configuration supports TxPreset and RxPresets at the port level, not at the lane level, so the same preset values are applied to all lanes (of the same direction) on the port.

Normally the TxPreset Table, RxPreset Table, and QSFP_ATTENUATION table are preset and cover the range of OEM platform signal integrity requirements. When using preset files, TX_PRESET_IDX for each port should be set to the same value as LOCAL_ATTEN_25G. RX_PRESET_IDX should be set to 0.

Fields not applicable to the implemented port type may be omitted from the INI file. The signal integrity information provided in the per-port INI settings shall be consistent with the electrical specifications and compliance points as defined in the *Cornelis Omni-Path Express Channel Electrical Specification*.

Table 2. Per Port INI Settings

Port Type	Applicable Fields	Description
DISCONNECTED	None	
FIXED	LOCAL_ATTEN_25G	Typical insertion loss of the entire channel, from TP_PAD to TP_PAD, at 12.89 GHz
	REMOTE_ATTEN_25G	Estimated typical insertion loss of remote platform port, from TP_PAD to TP_IO_W, at 12.89 GHz. In most cases, the preset value should be used.
VARIABLE	Variable table	This table is unused in the OPX Generation 1 HFI.
QSFP	LOCAL_ATTEN_25G	Typical insertion loss of the platform port, from TP_PAD to TP_IO_W, at 12.89 GHz

Port Type	Applicable Fields	Description
	REMOTE_ATTEN_25G	Estimated typical insertion loss of remote platform port, from TP_PAD to TP_IO_W, at 12.89 GHz. In most cases, the preset value should be used.

4. Creating a Configuration File

Text-based INI files are simple ASCII-based text files.

These text files contain zero or more input lines. Lines greater than 2048 characters in length are truncated. Lines can be terminated with either a LF (line feed) or a CR/LF (carriage return/line feed) combination.

Tabs and spaces are treated as delimiters between fields in the input line. Multiple tabs and spaces occurring together are treated as common whitespace.

If the first non-whitespace character on a line is a semicolon (;) the remainder of the line is treated as a comment.

An `#include <filename>` syntax is supported. Include files help to simplify the management of the information going into an INI configuration. Include files can also be used by an OEM to support sharing pieces of INI configurations between different OEM platform designs. Include statements may be nested (i.e., an include file can also include files). When the end-of-file has been reached in an include file, input parsing returns to the file that had included that file. The path to the `#include <filename>` is relative to the directory where the OEM runs the tool from (i.e., not necessarily the directory where the tool binary is located).

4.1. High-Level Process Steps

The creation and application of the platform configuration file can be summarized using the following high-level process steps.

1. Obtain a sample OPX HFI configuration file set in text format.
2. Update fields in the text files to match platform implementation specifics.
3. Compile the set of text files into a single binary image.
4. Program the binary platform configuration file into non-volatile storage on your platform.

Each of these steps is described in detail below.

NOTE

Separate files are required for each HFI ASIC present in the platform. For example, a dual-socket Purley platform will require 2 platform configuration files, one per socket. See Section 6.2 “Integrated Processors” for more details.

4.2. Obtain a Sample OPX HFI Configuration File Set in Text Format

A sample OPX HFI configuration file set in text format can be obtained through your Cornelis Platform Application Engineer or Cornelis Networks Customer Support.

Alternatively, for Intel Xeon Phi x200 Product Family and Intel Xeon Scalable Processors with Intel C620 Series Chipsets systems that utilize the embedded fabric processors, the files can be obtained via the BIOS releases for those platforms. The BIOS release versions have been optimized for the reference design implemented by Intel.

4.3. Update Fields in the Text Files to Match Platform Implementation Specifics

For ease of use and readability, the text version of the platform configuration file is split into multiple files, by function. The 'original' file set consisted of eight text files. An alternate 'reduced' file set consists of four text files. Both file sets contain acceptable content to create a configuration binary. The only difference is the organization of the content across the text files. If both file sets are available, platform implementers are encouraged to use the 'reduced' file structure, due to its smaller learning curve.

4.4. Sample File Structure

- **Sample\HFI1_IniMain_Sample.ini** – Parent file, includes all other text configuration files
- **Sample\HFI1_IniPlatform_Sample.txt** – Platform-specific information that might need to be edited by the platform implementers depending on how closely the design is to reference designs.
- **Common\HFI1_IniInternal.txt** – Used for configuration file internals, should not be edited by platform implementers
- **Common\HFI1_IniDefaults.txt** – Default values for system and port information, should not be edited by platform implementers

4.5. Fields to Update

There are several key fields to be updated, based on the platform implementation. These fields are described in the following sections. Note that the location of these fields may vary, depending on if the starting text files are from the 'original' file structure or the 'reduced' file structure.

4.5.1. NODE_STRING

This field is located at line 14 in the `Sample\HFI1_IniPlatform_Sample.txt` file.

The node string is a 64-byte ASCII string used to version the data of the configuration file. The text in this field should be in the form:

```
HFI_Type<NN>_V<HFI Generation>.<Major Version XX>.<Minor Version YY>.<Patch Version ZZ>
```

Example: `NODE_STRING="HFI_Sample v1.0.0.0"`

For HFI ASIC, the HFI Generation value is 1. Please work with your Cornelis account owner to be allocated a TYPE value for your HFI product.

4.5.2. QSFP_POWER_CLASS_MAX

This field is located at line 15 in the `Sample\HFI1_IniPlatform_Sample.txt` file.

This field identifies the highest QSFP power class that the OPX software will allow to attempt link up. Cables with higher power classes remain in `Offline` and report an `OfflineDisabledReason` of `Offline_PowerPolicy`. Typically, this field is used to limit the power draw or thermal cooling requirements based on platform capabilities.

Permitted values: 1 through 7 (refer to SFF-8636 for corresponding power levels).

4.5.3. PORT_TYPE

This field is located at lines 36 and 41 in the `Sample\HFI1_IniPlatform_Sample.txt` file.

The port type tells the OPX driver if it should use the `ModPrspin` to determine if the link should be attempted, as well as whether the port is a QSFP port or a backplane port.

Table 3. Permitted Values

Port Type	ModPrs monitored?	Description
QSFP	Y	The port is a QSFP cable or transceiver.
FIXED	N	The port is a backplane port. Link up will be attempted regardless of state of port ModPrs.
VARIABLE	Y	The port is a backplane port, but an external entity will assert ModPrs when a link should be attempted.
DISCONNECTED	N	The port is not implemented on the platform.

4.5.4. LOCAL_ATTEN_25G

This field is located at lines 37 and 42 in the `Sample\HFI1_IniPlatform_Sample.txt` file.

The local attenuation field is used by OPX software to configure the SerDes equalization. This field is particularly important for links containing optical transceivers and cables. The local attenuation value should be the insertion loss of the channel between the HFI die pad (`TP_PAD`) and the beginning of the QSFP cable or transceiver paddlecard (`TP_IO`), at 12.89 GHz. Note that this channel includes the CPU and/or ASIC package.

Since there are four transmit and four receive lanes, the local attenuation field should be the average across all transmit lanes. To obtain the appropriate equalization across all operating and manufacturing conditions, this attenuation should be the nominal insertion loss design point. This field is optional for links that will not incorporate optical cables or transceivers. If the platform implementer chooses not to provide the attenuation for a copper-only channel, this field should be set to zero (0).

If the platform implementer has followed the reference designs provided by Cornelis, this field can be calculated by the following formulas:

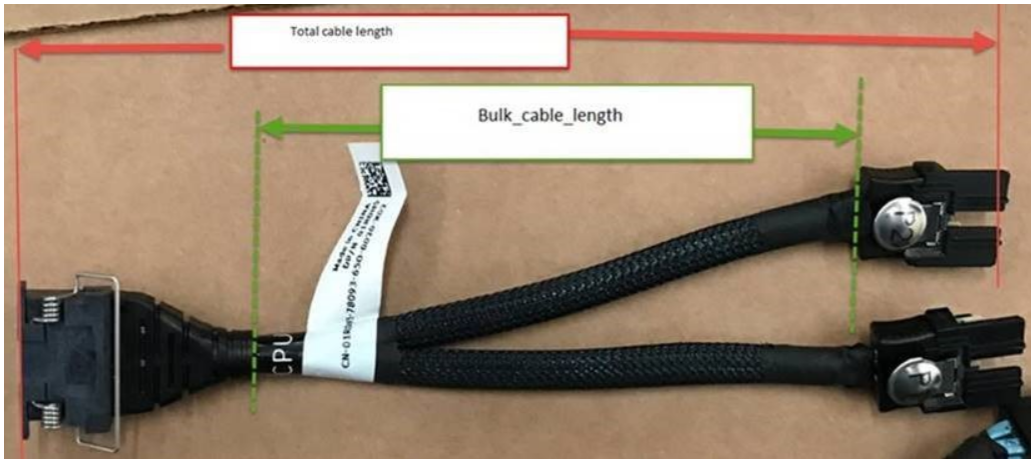
```
IFP-A (for KNL-F systems):
LOCAL_ATTEN_25G = 4.69 + (0.2615 * Bulk_Cable_Length)

IFP-B (for SKX-F systems):
LOCAL_ATTEN_25G = 3.655 + (0.2165 * Bulk_Cable_Length)
```

NOTE

Bulk_Cable_Length is in inches. Results from the formula should be rounded to the nearest whole number.

Permitted values: The `LOCAL_ATTEN_25G` value must be in whole decibels (dBs).

Figure 4. How to Measure bulk_cable_length



4.5.5. TX_PRESET_IDX_ACTIVE_EQ

This field is located at lines 38 and 43 in the `Sample\HFI1_IniPlatform_Sample.txt` file.

This field is a legacy field that should contain the same value as the `LOCAL_ATTEN_25G` field.

4.6. Common Files

There are several fields contained in two files that should not be edited by most platform implementers. These files are located in the Common folder. The fields are documented here for the rare case where they do need to be changed by the platform implementers. `HFI1_IniInternal.txt` is used for configuration file internals and should not be edited by platform implementers. `HFI1_IniDefaults.txt` contains default values for system and port information and should not be edited by platform implementers.

4.7. QSFP Attenuation Table Fields

This table is unused for STL Generation 1 HFIs.

5. Converting INI Files

For the Linux version of the tool, the following command line is used to convert a "text-based INI file" into a binary configuration file.

```
wfr_oem_tool wfr ini text2bin <inputFile> <outputFile>
```

For the Linux version of the tool, the following command line is used to convert a binary configuration file back into "text-based INI files".

```
wfr_oem_tool wfr ini bin2text <inputFile> <outputDataFile> <outputMetaFile>
```

Note that when converting a binary configuration file back into "text-based INI files", the tool is not able to exactly reproduce the original "text-based INI files". The non-reproducible information includes: user comments, original filenames, and #include directives.

6. Loading the Platform Configuration

The platform configuration is intended to be distributed as part of the hardware in the discrete case, and as part of the BIOS in the integrated case. For the discrete cards, the UEFI HfiPcieGen3 driver and the Linux driver read the platform configuration file from the optionROM EEPROM.

The location of the AOC configuration file will vary based on whether the WFR solution is integrated or discrete. The Option ROM will need to know this difference and find the image, from the EPROM in the case of the discrete solution, or the BIOS filesystem in the integrated solution.

As a fallback for debugging and testing, the driver will also load the platform configuration via the **`request_firmware(…)`** mechanism provided by the Linux kernel if a platform configuration file is not found in the locations defined above. The fall back file needs to describe the entire platform, regardless of the number of discrete or integrated solutions present in the platform. This fallback file is named **`hfi1_platform.dat`** and should be placed in /lib/firmware/updates on an RHEL system or equivalent location on a SLES system. The file should also conform to the design described in this document.

To program the EEPROM for the discrete HFI ASIC, the binary configuration file output of the **`wfr_oem_tool`** described above is provided to a utility named **`hfi1_eprom`**, which is invoked as:

```
hfi1_eprom -w -c <path to binary configuration file output by wfr_oem_tool>
```

To acquire the wfr_oem_tool, visit Cornelis Networks Customer Support

6.1. Discrete Adapters

6.1.1. UEFI BIOS Booting in UEFI Mode/Legacy Mode

The following mechanism is proposed:

The option ROM implemented as SPI flash contains the following contents:

- A PCIe optionROM driver named HfiPcieGen3Loader with a maximum size limit of 128k
- A platform configuration file up to 4k in size
- A HfiPcieGen3 driver that performs the actual HFI initialization described in DN_HFI_PCIe_Gen3 and DN_HFI1_UNDI

In the discrete adapter case, the HfiPcieGen3 driver and Linux driver read the platform configuration file partition via the WFR SPI access registers.

6.2. Integrated Processors

6.2.1. UEFI BIOS Booting in UEFI Mode/Legacy Mode

The platform configuration file is stored on a per-socket basis as part of the default UEFI variable store. This is accomplished at manufacturing time by including the platform configuration files within the BIOS image with the naming convention `hfi1-socket<N>-configuration`, where *N* is the socket ID.

The HfiPcieGen3 driver identifies the socket ID of the HFI in a CPU-specific manner. The UEFI driver will have to use the cpuid instruction to recognize the CPU the HFI is integrated into and identify the socket ID accordingly.

For Intel Xeon Scalable Processors with Intel C620 Series Chipsets, the mechanism to identify the socket ID is to use a UEFI Hand-Off- Block (HOB) published by the BIOS. For Intel Xeon Scalable Processors with Intel C620 Series Chipsets, the GUID to be used to access the HOB is `829d41d2-6ca5-485b-a1a2-d1b79627abcd`.

The HOB structure for Intel Xeon Scalable Processors with Intel C620 Series Chipsets is defined below:

```
typedef struct {
   UINT8 Valid;
   UINT8 SocketId;
   UINT8 Segment;
   UINT8 BusBase;
   UINT8 BusLimit;
   UINT8 Reserved;
   UINT8 Reserved2;
   UINT8 Reserved3;
} EFI_OPA_SOCKET_MAP;


typedef struct {
   EFI_HOB_GUID_TYPE    Hob;
   EFI_OPA_SOCKET_MAP Info[0];
} OPA_SOCKET_MAP_HOB;
```

For Intel Xeon Phi x200 Product Family, since there is only one socket available due to QPI/KTI limitations, the socket ID is hard coded to zero.

Knowing this socket ID, the HfiPcieGen3 driver reads the configuration files stored per socket from the default UEFI variable store via the UEFI APIs GetVariable/SetVariable and saves selected values, based on the port type, into the HFI ASIC scratch registers following the bitmap(v1) below. The GUID used to access the EFI variables are shown below:

Purley: `56d23d33-07d7-48f9-81d6-7ed5991ece88`

Groveport: `58952180-01d5-4106-8955-fe1f5b596f09`

Table 4. ASIC_CFG_SCRATCH 0

Start	End	Length	Name
32	43	12	Unused
44	47	4	Bit map version
48	63	16	Bottom 16 bits of checksum over all platform configuration fields described below and bitmap version

Table 5. ASIC_CFG_SCRATCH 1

Start	End	Length	Name
0	3	4	HFI0 Port.Port Type
4	9	6	HFI0 Port.Local Attenuation 25Gbps
10	15	6	HFI0 Port.Remote Estimated Attenuation 25 Gbps
16	19	4	HFI1 Port.Port Type
20	25	6	HFI1 Port.Local Attenuation 25Gbps
26	31	6	HFI1 Port.Remote Estimated Attenuation 25 Gbps
32	39	8	HFI0 System.QSFP Default Attenuation 25 Gbps
40	47	8	HFI1 System.QSFP Default Attenuation 25 Gbps
48	63	16	Reserved

Table 6. ASIC_CFG_SCRATCH 2

Start	End	Length	Name
0	3	4	HFI0 System.QSFP Power Class Max
4	24	21	HFI0 TX Preset.TX Preset NO EQ
25	45	21	HFI0 TX Preset.TX Preset EQ
46	61	16	HFI0 RX Preset.RX Preset
62	63	2	Reserved

Table 7. ASIC_CFG_SCRATCH 3

Start	End	Length	Name
0	3	4	HFI1 System.QSFP Power Class Max
4	24	21	HFI1 TX Preset.TX Preset NO EQ
25	45	21	HFI1 TX Preset.TX Preset EQ
46	61	16	HFI1 RX Preset.RX Preset
62	63	2	Reserved

The bitmap of the TX Preset described above is adapted from the TX Preset bitmap described in the WFR Platform Configuration Guide by stripping the most significant reserved 11 bits of the TX Preset DWORD:

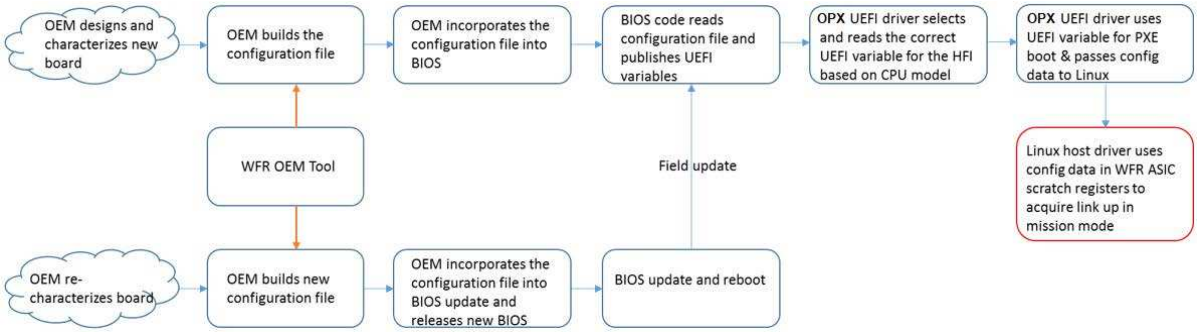
POSTCUR (5b)	QSFP TX EQ (4b)	[i] (1b)	[h] (1b)	[g] (1b)	ATTN (5b)	PRECUR (4b)

Similarly, the bitmap of the RX Preset described above is adapted from the RX Preset bitmap described in the WFR Platform Configuration Guide by stripping the most significant reserved 22 bits of the RX Preset DWORD:

QSFP RX AMP (2b)	QSFP RX EQ (4b)	[f] (1b)	[e] (1b)	[d] (1b)	[c] (1b)

This section is summarized in the figure below.

Figure 5. Design and Flow for Integrated Processors



7. Field Definition Reference

The following sections describe the details of all configuration file fields. Most platform implementers can create their platform configuration file by following the simplified methodology described in Section 4 “Creating a Configuration File”.

7.1. Data Tables

An INI configuration contains information for up to 6 unique data tables as described below:

Table 8. Data Tables

Data Table Name	Needed	Purpose
System	Yes	Global settings
Port	Yes	Per port settings
TxPreset	Yes	TX Presets Table
RxPreset	Yes	RX Presets Table
QsfpAttenuation	If OEM platform uses QSFPs	QSFP Attenuation Table
VariableSettings	If OEM platform uses Variable ports	Variable Port Table

Within a DATA section of an INI file, a FIELD value can be specified more than once, for instance, if an OEM has a default value in an #include file that they want to override. When a FIELD value is specified more than once, then the last specified value is the one that is used.

By default if a value isn't given for a FIELD, it defaults to 0.

7.2. INI MetaData

In addition to the 6 data tables, each INI file must include MetaData.

Although an OEM doesn't modify the MetaData, they may often refer to the WFRIniMeta.txt file for reference, and so a brief description of the format of the WFRIniMeta.txt file is given here.

Normally an OEM includes the MetaData by having a '#include "WFRIniMeta.txt" statement as the first non-comment, non-blank line in their top level INI text file.

The supplied WFRIniMeta.txt file should not be changed or updated, except as instructed by Cornelis as platform configuration updates and/or tool changes occur.

The MetaData file is used to map the information from the text based INI file into the binary configuration file. The MetaData file allows Cornelis to add new fields or to resize existing fields and maintain some forwards/backwards compatibility between different INI binary files that use newer or older MetaData field descriptions.

Within the MetaData file, there is a section that describes meta-data for each of the Data tables. Each of these sections follows the same basic format. Each

section starts with its name, which is one of the following: [SystemMetaData], [PortMetaData], [RxPresetsMetaData], [TxPresetsMetaData], [QsfpAttenuationMetaData], or [VariableSettingsMetaData].

After the section name in the MetaData there are a series of input lines that define a record in the corresponding data table.

Table 9. Record Items

RECORD ITEM	PURPOSE
IDX	A unique value (unique for this table type) that the driver uses to identify this particular field.
LEN	The number of bits associated with the FIELD in the binary configuration file.
START	The start bit of the lowest numbered bit of the FIELD in the binary configuration file.
FIELD	The text based name of the field. This can optionally include "ENCODINGS" that describe how the WFR driver interprets the unique values that this FIELD can be set to.

The conversion tool enforces some constraints on the MetaData records including:

- If a field is greater than or equal to 32 bits in length, it must start on an even 32-bit boundary
- Fields that are 32 bits or shorter must not cross 32-bit boundaries
- Fields cannot overlap (i.e., the bit field location defined by START and LEN) must be unique.

The START and LEN bits from a MetaData record are written into the binary INI file for use by the WFR driver to locate the corresponding FIELD values in the data tables of the binary configuration file.

7.3. System Table Fields

The System Table defines configurable global platform configuration settings.

Table 10. System Table Fields

FIELD NAME	Used By	Description
META_VERSION	F/W Config	16 bits. This value is defined by the supplied INI files and should not be changed by the OEM designer. It is intended to provide some forward/backward interoperability in the event that Cornelis chooses to change the format of the INI META data in the binary configuration file.

FIELD NAME	Used By	Description
NODE_STRING	SMA	64-byte ASCII string used to version the data of the configuration file. The text in this field should be of the form `HFI_Type<NN>_V<HFI Generation>.<Major Version XX>.<Minor Version YY>.<Patch Version ZZ>`. For WFR, the HFI Generation value is 1. Please work with your Cornelis account owner to be allocated a TYPE value for your HFI product.
QSFP_POWER_CLASS_MAX	Power Management	This defines the maximum QSFP power class of a QSFP module that the driver will move into a high power state. See the "Power Management" description above for more details about this field. 1 = Power Class 1 (1.5W max) 2 = Power Class 2 (2.0W max) 3 = Power Class 3 (2.5W max)
QSFP_ATTENUATION_DEFAULT_25G	Signal Integrity Tuning	8 bits - This value is used as the QSFP attenuation (in dB) for 25.7825 Gb/s signaling if the 12 GHz QSFP attenuation value from the QSFP device (B189 in SFF-8636) is not considered valid (invalid being 0 or greater than 31).

7.4. Port Table

The Port Table defines per-port configurable firmware settings.

7.4.1. [Port=Common]

As a convenience to OEM designers, the INI conversion tool supports the notion of a "Common" section for ports. This allows an OEM designer to specify a common set of settings to apply to all ports first, before specifying port specific settings. The use of "[Port=Common]" is optional.

If a [Port=Common] section is used, it is recommended that it is #include(d) before any specific Port settings are specified.

7.4.2. [Port=<logical port number>]

Port Table information for each logical port should be specified in the INI file.

Table 11. Port Table Information

FIELD NAME	Used By	Description
PORT_TYPE	Various	This is the Port Type. See Section 3.1 "Port Types" for more information. Choices are: DISCONNECTED, FIXED, VARIABLE, and QSFP.
LOCAL_ATTEN_12G	Signal Integrity	A value between 0 and 36 inclusive. See the Section 3.4 "Signal Integrity Tuning" for how this value is used.
LOCAL_ATTEN_25G	Signal Integrity	A value between 0 and 36 inclusive. See Section 3.4 "Signal Integrity Tuning" for how this value is used.
REMOTE_ATTEN_12G	Signal Integrity	6 bits See Section 3.4 "Signal Integrity Tuning" for how this value is used.
REMOTE_ATTEN_25G	Signal Integrity	6 bits See Section 3.4 "Signal Integrity Tuning" for how this value is used.
TX_PRESET_IDX_ACTIVE_NO_EQ	Signal Integrity	12 bits See Section 3.4 "Signal Integrity Tuning" for how this value is used.
TX_PRESET_IDX_ACTIVE_EQ	Signal Integrity	12 bits See Section 3.4 "Signal Integrity Tuning" for how this value is used.
RX_PRESET_IDX	Signal Integrity	12 bits See Section 3.4 "Signal Integrity Tuning" for how this value is used.

7.4.3. RX Preset Table Field

The RX Preset Table Entries are used to configure the RX electrical channel. An entry in this table is selected based on the Signal Integrity algorithms described earlier in this document. Typically, the RX Preset Table is provided by Cornelis and does not need to be edited by the platform implementer.

Each unique entry is started with "[RxPreset=<index>]".

Table 12. RX Preset Table

FIELD NAME	Used By	Description
QSFP_RX_OUTPUT_CDR_APPLY	Signal Integrity	If the electrical channel includes a QSFP module that supports RX_OUTPUT_CDR (clock data recovery), then this field indicates whether or not the QSFP_RX_OUTPUT_CDR setting from this Rx Present Table entry is applied. This field can be set to "NO" or "YES". If set to "NO" then no RX_OUTPUT_CDR setting is applied.
QSFP_RX_OUTPUT_EMP_APPLY	Signal Integrity	If the electrical channel includes a QSFP module that supports RX_OUTPUT_EMP (emphasis), then this field indicates whether or not the QSFP_RX_OUTPUT_EMP setting from this Rx Present Table entry is applied. This field can be set to "NO" or "YES". If set to "NO" then no RX_OUTPUT_EMP setting is applied.
QSFP_RX_OUTPUT_AMP_APPLY	Signal Integrity	If the electrical channel includes a QSFP module that supports RX_OUTPUT_AMP (amplitude), then this field indicates whether or not the QSFP_RX_OUTPUT_AMP setting from this Rx Present Table entry is applied. This field can be set to "NO" or "YES". If set to "NO" then no RX_OUTPUT_AMP setting is applied.
QSFP_RX_OUTPUT_CDR	Signal Integrity	If the electrical channel includes a QSFP module that supports RX_OUTPUT_CDR, and the QSFP_RX_OUTPUT_CDR_APPLY setting is set to "YES", then this setting is applied to the RX_OUTPUT_CDR control in the QSFP module. This field can be set to "DISABLED" or "ENABLED". If set to "ENABLED" then the RX_OUTPUT_CDR in the module is explicitly enabled, otherwise it is explicitly disabled. **Implementer's note:** In the current version of the driver, this configuration file setting is ignored. Instead, all QSFP cables Power Class 3 and below will always enable the CDR and all QSFP cables Power Class 4 and above will always disable the CDR.
QSFP_RX_OUTPUT_EMP	Signal Integrity	If the electrical channel includes a QSFP device that supports RX_OUTPUT_EMP, and the QSFP_RX_OUTPUT_EMP_APPLY setting is set to "YES", then this EMP setting (4 bit value) is applied to the module.

FIELD NAME	Used By	Description
QSFP_RX_OUTPUT_AMP	Signal Integrity	If the electrical channel includes a QSFP device that supports RX_OUTPUT_AMP, and the QSFP_RX_OUTPUT_AMP_APPLY setting is set to "YES", then this AMP setting (2 bit value) is applied to the QSFP module. **NOTE:** This field defines the AMP Code value to use. A module may not support the full set of AMP Code values. In the cases where this setting exceeds the largest supported value of the module, that largest supported value (above this value) is used instead.

7.4.4. TX Preset Table Field

The TX Preset Table Entries are used to configure the TX electrical channel. An entry in this table is selected based on the Signal Integrity algorithms described earlier in this document. Typically, the TX Preset Table is provided by Cornelis and does not need to be edited by the platform implementer.

Each unique entry is started with "[TxPreset=<index>]".

Table 13. TX Preset Table

FIELD NAME	Used By	Description
PRECUR	Signal Integrity	This is a 4-bit value used to configure the precursor (emphasis) on the HFI ASIC TX pin.
ATTN	Signal Integrity	This is a 5-bit value used to configure the attenuation (amplitude) on the HFI ASIC TX pin.
POSTCUR	Signal Integrity	This is a 5-bit value used to configure the post cursor on the HFI ASIC TX pin.
QSFP_TX_INPUT_CDR_APPLY	Signal Integrity	If the electrical channel includes a QSFP module that supports TX_INPUT_CDR (clock data recovery), then this field indicates whether or not the QSFP_TX_INPUT_CDR setting from this TX Present Table entry is applied. This field can be set to "NO" or "YES". If set to "NO" then no TX_INPUT_CDR setting is applied.
QSFP_TX_INPUT_EQ_APPLY	Signal Integrity	If the electrical channel includes a QSFP module that supports TX_INPUT_EQ, then this field indicates whether or not the QSFP_TX_INPUT_EQ setting from this Tx Present Table entry is applied. This field can be set to "NO" or "YES". If set to "NO" then no TX_INPUT_EQ setting is applied.

FIELD NAME	Used By	Description
QSFP_TX_INPUT_CDR	Signal Integrity	If the electrical channel includes a QSFP module that supports TX_INPUT_CDR, and the QSFP_TX_INPUT_CDR_APPLY setting is set to "YES", then this setting is applied to the TX_INPUT_CDR control in the QSFP module. This field can be set to "DISABLED" or "ENABLED". If set to "ENABLED" then the TX_INPUT_CDR in the module is explicitly enabled, otherwise it is explicitly disabled. **Implementer's note:** In the current version of the driver, this configuration file setting is ignored. Instead, all QSFP cables Power Class 3 and below will always enable the CDR and all QSFP cables Power Class 4 and above will always disable the CDR.
QSFP_TX_INPUT_EQ	Signal Integrity	If the electrical channel includes a QSFP module that supports TX_INPUT_EQ, and the QSFP_TX_INPUT_EQ_APPLY setting is set to "YES", then this EQ setting (4 bit value) is applied to the module.

8. Glossary of Acronyms

The following acronyms are used throughout the OPX Documentation Set.

AOC	Active Optic Cable - an optic cable used to connect modules in the OPX fabric
API	Application Programming Interface - how an application interacts with another application or a user
ASCII	American Standard Code for Information Interchange - a character encoding standard
ASIC	Application Specific Integrated Circuit - a customized chip designed for a particular purpose
BIOS	Basic Input Output System - firmware used to initialize and test system hardware components, and load a boot loader from a mass storage device which then initializes a kernel
CHF	Cornelis Omni-Path Express Host Fabric Interface Adapter
CPU	Central Processing Unit - primary chip in a computer used to execute the instructions comprising a computer program
EEPROM	Electrically Erasable Programmable Read-only Memory - user-modifiable read-only memory
GUID	Globally Unique Identifier - a 128-bit text string generated when a unique reference number is needed to identify a component (e.g., hardware, software, account, node, etc.) on a computer or network
HFI	Host Fabric Interface - the hardware and software that allows one module to talk to another in an OPX Fabric.
HFI ASIC	Host Fabric Interface Application Specific Integrated Circuit
HTTPS	Hypertext Transfer Protocol Secure - an extension of HTTP, it is used for secure communication over a computer network, where the communication protocol is encrypted using Secure Sockets Layer
I2C	Inter-Integrated Controller - a 2-wire bus with a protocol that allows multiple peripheral devices to communicate with one or more "controller" chips

OEM	Original Equipment Manufacturer - a company whose goods are used as components in the products of another company
OPX	Omni-Path Express
OPX HFI	Omni-Path Express Host Fabric Interface
PCIe	Peripheral Component Interconnect Express - a high-speed serial computer expansion bus standard for connecting a computer to one or more peripheral devices
QSFP	Quad Small Form-factor Pluggable - a compact, hot-pluggable transceiver used for data communications applications
RHEL	Red Hat Enterprise Linux
SLES	SUSE Linux Enterprise Server
SMA	Subnet Management Agent
SPI	Serial Peripheral Interconnect - a synchronous serial communication interface specification used for short-distance communication
UEFI	Unified Extensible Firmware Interface - a publicly available specification that defines a software interface between an operating system and platform firmware