Intel® FPGA RTE for OpenCL™ Standard Edition

Getting Started Guide

Updated for Intel® Quartus® Prime Design Suite: 18.1
# Contents

1. Intel® FPGA RTE for OpenCL™ Standard Edition Getting Started Guide ........................................ 4
   1.1. Prerequisites for the Intel FPGA RTE for OpenCL Standard Edition .................................... 5
   1.2. Contents of the Intel FPGA RTE for OpenCL Standard Edition ........................................... 6
   1.3. RTE Utility ......................................................................................................................... 7
      1.3.1. Displaying the Software Version ........................................................................... 7
      1.3.2. Listing the Intel FPGA RTE for OpenCL Standard Edition Utility Command Options ......................................... 7
      1.3.3. Managing an FPGA Board ................................................................................... 8
      1.3.4. Managing Host Application ............................................................................... 8

2. Getting Started with the Intel FPGA RTE for OpenCL Standard Edition for 64-Bit Windows ................................................................................................................................. 11
   2.1. Downloading the Intel FPGA RTE for OpenCL Standard Edition ...................................... 11
   2.2. Installing the Intel FPGA RTE for OpenCL ......................................................................... 12
   2.3. Setting the Intel FPGA RTE for OpenCL Standard Edition User Environment Variables.... 12
   2.4. Verifying Software Installation ........................................................................................ 13
   2.5. Installing an FPGA Board ............................................................................................... 14
   2.6. Updating the Hardware Image on the FPGA .................................................................... 15
      2.6.1. Querying the Device Name of Your FPGA Board ................................................ 16
      2.6.2. Programming the Flash Memory of an FPGA ....................................................... 16
   2.7. Executing an OpenCL Kernel on an FPGA ..................................................................... 18
      2.7.1. Building the Host Application ............................................................................. 18
      2.7.2. Running the Host Application ............................................................................ 19
      2.7.3. Output from Successful Kernel Execution......................................................... 19
   2.8. Uninstalling the Software .............................................................................................. 20
   2.9. Uninstalling the FPGA Board ......................................................................................... 20

3. Getting Started with the Intel FPGA RTE for OpenCL Standard Edition for x86_64 Linux Systems ............................................................................................................................................. 21
   3.1. Downloading the Intel FPGA RTE for OpenCL Standard Edition .................................. 21
   3.2. Installing the Intel FPGA RTE for OpenCL ....................................................................... 22
   3.3. Setting the Intel FPGA RTE for OpenCL Standard Edition User Environment Variables.... 23
   3.4. Verifying Software Installation ........................................................................................ 23
   3.5. Installing an FPGA Board ............................................................................................... 24
   3.6. Updating the Hardware Image on the FPGA .................................................................... 25
      3.6.1. Querying the Device Name of Your FPGA Board ................................................ 26
      3.6.2. Programming the Flash Memory of an FPGA ....................................................... 26
   3.7. Executing an OpenCL Kernel on an FPGA ..................................................................... 27
      3.7.1. Building the Host Application ............................................................................. 27
      3.7.2. Running the Host Application ............................................................................ 28
      3.7.3. Output from Successful Kernel Execution......................................................... 28
   3.8. Uninstalling the Software .............................................................................................. 29
   3.9. Uninstalling the FPGA Board ......................................................................................... 29
4. Getting Started with the Intel FPGA RTE for OpenCL Standard Edition for Intel ARMv7-A SoC FPGA

4.1. Getting Started with the Intel FPGA RTE for OpenCL Standard Edition for SoC FPGA on Windows


4.1.2. Installing the Intel FPGA SDK for OpenCL Standard Edition for SoC FPGA

4.1.3. Installing the Intel SoC FPGA Embedded Development Suite Standard Edition

4.1.4. Recompiling the Linux Kernel Driver

4.1.5. Installing the Intel FPGA RTE for OpenCL Standard Edition onto the SoC FPGA Board

4.1.6. Installing the Cyclone V SoC Development Kit

4.1.7. Executing an OpenCL Kernel on an SoC FPGA

4.1.8. Uninstalling the Intel FPGA RTE for OpenCL Standard Edition

4.2. Getting Started with the Intel FPGA RTE for OpenCL Standard Edition for SoC FPGA on Linux


4.2.2. Installing the Intel FPGA SDK for OpenCL Standard Edition for SoC FPGA

4.2.3. Installing the Intel SoC FPGA Embedded Development Suite Standard Edition

4.2.4. Recompiling the Linux Kernel Driver

4.2.5. Installing the Intel FPGA RTE for OpenCL Standard Edition onto the SoC FPGA Board

4.2.6. Installing the Cyclone V SoC Development Kit

4.2.7. Executing an OpenCL Kernel on an SoC FPGA

4.2.8. Uninstalling the Intel FPGA RTE for OpenCL Standard Edition


1. Intel® FPGA RTE for OpenCL™ Standard Edition Getting Started Guide

The Intel® FPGA RTE for OpenCL™ Standard Edition Getting Started Guide describes the procedures you follow to install the Intel FPGA Runtime Environment (RTE) for OpenCL. This document also contains instructions on how to deploy an OpenCL (1) application with the RTE.

The RTE is a subset of the Intel FPGA Software Development Kit (SDK) for OpenCL(2). Unlike the SDK, which provides an environment that enables the development and deployment of OpenCL kernel programs, the RTE provides tools and runtime components that enable you to build and execute a host program, and execute precompiled OpenCL kernel programs on target accelerator boards.

OpenCL is a C-based open standard for the programming of heterogeneous parallel devices. For more information on the OpenCL Specification version 1.0, refer to the OpenCL Reference Pages. For detailed information on the OpenCL application programming interface (API) and programming language, refer to the OpenCL Specification version 1.0.


Do not install the RTE and the SDK on the same host system.

Attention: If you require OpenCL kernel development and deployment functionalities that target the Cyclone® V SoC Development Kit, refer to the Intel FPGA SDK for OpenCL Standard Edition Cyclone V SoC Getting Started Guide for more information.

Related Information

- OpenCL Reference Pages
- OpenCL Specification version 1.0
- Intel FPGA SDK for OpenCL Standard Edition Getting Started Guide
- Intel FPGA SDK for OpenCL Cyclone V SoC Getting Started Guide

(1) OpenCL and the OpenCL logo are trademarks of Apple Inc. used by permission of the Khronos Group™.

(2) The Intel FPGA SDK for OpenCL is based on a published Khronos Specification, and has passed the Khronos Conformance Testing Process. Current conformance status can be found at www.khronos.org/conformance.
1.1. Prerequisites for the Intel FPGA RTE for OpenCL Standard Edition

To install the Intel FPGA RTE for OpenCL Standard Edition and deploy an application on an Intel preferred accelerator board, your system must meet certain hardware, target platform, and software requirements.

**Hardware Requirements**

Accelerator boards requirements:
- Acquire a Reference Platform from Intel, or a Custom Platform from an Intel preferred board vendor.
  
  For more information, refer to the Intel FPGA SDK for OpenCL FPGA Platforms page on the Intel FPGA website.

Deployment system requirements:
- You must have administrator privileges on the development system to install the necessary packages and drivers.
- The deployment system has at least 20 megabytes (MB) of free disk space for software installation.
- The deployment system has at least 128 MB of RAM.
  
  *Tip:* Refer to board vendor’s documentation on the recommended system storage size.

The host system must be running one of the following supported operating systems:
- For a list of supported Windows and Linux operating systems, refer to the Operating System Support page on the Intel FPGA website.
- Linux versions as supported on Intel SoC FPGA products on the Arm* ARMv7-A architecture.

*Important:* For x86_64 Linux systems, install the Linux OS kernel source and headers (for example, kernel-devel.x86_64 and kernel-headers.x86_64), and the GNU Compiler Collection (GCC) (gcc.x86_64).

To install the Linux kernel source or header package, invoke the `yum install <kernel_package_name>` command.

You must have administrator privileges on the host system to install the necessary packages and drivers.

**Software Prerequisites**

Develop your host application using one of the following RTE-compatible C compiler or software development environment:
- For Windows systems, use Microsoft Visual Studio Professional version 2010 or later.
- For Linux systems, use the C compiler included with the GCC.
- For SoC applications, use the GCC cross-compiler available with the Intel SoC FPGA Embedded Development Suite (EDS).
Linux systems require the Perl command version 5 or later. Include the path to the Perl command in your PATH system environment variable setting.

For Intel FPGA RTE for OpenCL packages that include Intel Code Builder, Intel Code Builder requires Java SE version 1.8.71 or later to run.

**Related Information**
- OpenCL Platforms
- Intel FPGA SDK for OpenCL Standard Edition Getting Started Guide

### 1.2. Contents of the Intel FPGA RTE for OpenCL Standard Edition

The Intel FPGA RTE for OpenCL Standard Edition provides utilities, host runtime libraries, drivers, and RTE-specific libraries and files.

**Utilities and Host Runtime Libraries**
- The RTE Utility includes commands you can invoke to perform high-level tasks. The RTE utilities are a subset of the Intel FPGA SDK for OpenCL Standard Edition utilities.
- The host runtime provides the OpenCL host platform API and runtime API for your OpenCL host application.

The host runtime consists of the following libraries:
- *Statically-linked libraries* provide OpenCL host APIs, hardware abstractions and helper libraries.
- *Dynamic link libraries* (DLLs) provide hardware abstractions and helper libraries.

**Drivers, Libraries and Files**

The RTE installation process installs the RTE into a directory that you own. The path to the software installation directory is referenced by the `INTELFPGAOCLSDKROOT` environment variable.

<table>
<thead>
<tr>
<th>Windows Folder</th>
<th>Linux Directory</th>
<th>ARM Directory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bin</td>
<td>bin</td>
<td>bin</td>
<td>High-level utilities. Include this directory in your PATH environment variable setting.</td>
</tr>
<tr>
<td>board</td>
<td>board</td>
<td>board</td>
<td>The Reference Platform available with the RTE. Important: The Reference Platform for the RTE does not include the hardware subdirectory.</td>
</tr>
<tr>
<td>host</td>
<td>host</td>
<td>host</td>
<td>Files necessary for compiling your host program.</td>
</tr>
<tr>
<td>host \include</td>
<td>host/</td>
<td>host/</td>
<td>OpenCL Specification version 1.0 header files and software interface files necessary for compiling and linking your host application. The host/include/CL subdirectory also includes the C++ header file cl.hpp. The file contains an OpenCL version 1.1 C++ wrapper API. These C++ bindings enable a C++ host program to access the OpenCL runtime APIs using native C++ classes and methods.</td>
</tr>
</tbody>
</table>

*continued...*
Example OpenCL Applications

You can download example OpenCL applications from the OpenCL Design Examples page.

Related Information
OpenCL Design Examples

1.3. RTE Utility

The Intel FPGA RTE for OpenCL Standard Edition utility is a subset of the Intel FPGA SDK for OpenCL Standard Edition utility. It provides you with tools and information to perform high-level tasks such as configuring the host application development flow.

1.3.1. Displaying the Software Version

To display the version of the Intel FPGA RTE for OpenCL Standard Edition, invoke the version utility command.

Note: The ARM processor on an Intel SoC FPGA board does not support this utility.

- At the command prompt, invoke the aocl version command. Example output:
  
aocl <version>.<build> (Intel(R) Runtime Environment for OpenCL(TM), Version <version> Build <build>, Copyright (C) <year> Intel Corporation)

1.3.2. Listing the Intel FPGA RTE for OpenCL Standard Edition Utility Command Options

To display information on the Intel FPGA RTE for OpenCL Standard Edition utility command options, invoke the help utility command.
Attention: The ARM processor on an Intel SoC board does not support this utility.

- At a command prompt, invoke the aocl help command. The RTE categorizes the utility command options based on their functions. It also provides a description for each option.

1.3.3. Managing an FPGA Board

The Intel FPGA RTE for OpenCL Standard Edition includes utility commands you can invoke to install, uninstall, diagnose, and program your FPGA board.

For more information about the install, uninstall, diagnose, program and flash utility commands, refer to the Managing an FPGA Board section of the Intel FPGA SDK for OpenCL Standard Edition Programming Guide.

Related Information
Managing an FPGA Board

1.3.4. Managing Host Application

The Intel FPGA RTE for OpenCL Standard Edition includes utility commands you can invoke to obtain information on flags and libraries necessary for compiling and linking your host application.

Attention: To cross-compile your host application to an SoC FPGA board, include the -arm option in your utility command.

Caution: For Linux systems, if you debug your host application using the GNU Project Debugger (GDB), invoke the following command prior to running the host application:

handle SIG44 nostop

Without this command, the GDB debugging process terminates with the following error message:

Program received signal SIG44, Real-time event 44.

For information on the following utility command options, refer to the Managing Host Application section of the Intel FPGA SDK for OpenCL Standard Edition Programming Guide:

- example-makefile or makefile
- compile-config
- ldflags
- ldlibs
- link-config or linkflags

Related Information
Managing Host Application
1.4. Overview of the Intel FPGA RTE for OpenCL Standard Edition Setup Process

The Intel FPGA RTE for OpenCL Standard Edition Getting Started Guide outlines the procedures for installing the Intel FPGA RTE for OpenCL Standard Edition and deploying an OpenCL example design onto your device.

**Important:** The RTE does not include the Intel FPGA SDK for OpenCL Offline Compiler; therefore, you cannot use the RTE to compile an OpenCL kernel. You must use the Intel FPGA SDK for OpenCL on a separate development machine to create an executable file (.aocx) from the .cl kernel source file. Refer to the Intel FPGA SDK for OpenCL Standard Edition Getting Started Guide for instructions on setting up the SDK and compiling an OpenCL kernel.
Figure 1. RTE Setup Process for x86-64 Systems

The figure below summarizes the steps for installing the RTE and the FPGA board, and in executing an OpenCL kernel on the board.

For an overview of the RTE setup process for SoC, refer to Getting Started with the Intel FPGA RTE for OpenCL for Intel ARMv7-A SoC.

Related Information
- Intel FPGA SDK for OpenCL Standard Edition Getting Started Guide
- Getting Started with the Intel FPGA RTE for OpenCL Standard Edition for Intel ARMv7-A SoC FPGA on page 30
2. Getting Started with the Intel FPGA RTE for OpenCL Standard Edition for 64-Bit Windows

Figure 1 on page 10 outlines the RTE setup process for 64-bit Windows systems.

1. Downloading the Intel FPGA RTE for OpenCL Standard Edition on page 11
2. Installing the Intel FPGA RTE for OpenCL on page 12
4. Verifying Software Installation on page 13
5. Installing an FPGA Board on page 14
6. Updating the Hardware Image on the FPGA on page 15
7. Executing an OpenCL Kernel on an FPGA on page 18
8. Uninstalling the Software on page 20
9. Uninstalling the FPGA Board on page 20

2.1. Downloading the Intel FPGA RTE for OpenCL Standard Edition

Download the Intel FPGA RTE for OpenCL Standard Edition for Windows from the Intel FPGA RTE for OpenCL Download Center.

1. Go to the Intel FPGA RTE for OpenCL Download Center at the following URL: http://fpgasoftware.intel.com/opencl/
3. Select the software version. The default selection is the current version.
4. Select one of the following download methods:
   — Akamai DLM3 Download Manager
   — Direct Download
5. Click the RTE tab and select Intel FPGA Runtime Environment for OpenCL Windows x86-64. Click More beside Download and install instructions to view the download and installation procedure.
6. Click the download button to start the download process.
7. Perform the steps outlined in the download and installation instructions on the download page.

Related Information
Intel FPGA website
2.2. Installing the Intel FPGA RTE for OpenCL

Install the Windows version of the Intel FPGA RTE for OpenCL Standard Edition in a folder that you own.

You must have administrator privileges to execute these instructions.

To install the Intel FPGA RTE for OpenCL, perform the following tasks:

1. Run the `setup.bat` file to install the SDK with the Intel Quartus® Prime Standard Edition software.

2. Run the `.exe` installer. Direct the installer to extract the software to an empty folder that you own (that is, not a system folder).
   
   Note: The installation path must not contain any spaces (for example, `<home_directory>\intelfpga\<version>\ aclrt-windows64`).

3. Note: The installer sets the user environment variable `INTELFPGAOCLSDKROOT` to point to the path of the software installation.
   
   Verify that `INTELFPGAOCLSDKROOT` points to the current version of the software.
   
   Open a Windows command window and then type `echo %INTELFPGAOCLSDKROOT%` at the command prompt.
   
   If the returned path does not point to the location of the Intel FPGA RTE for OpenCL installation, edit the `INTELFPGAOCLSDKROOT` setting.

Related Information

Setting the Intel FPGA RTE for OpenCL Standard Edition User Environment Variables on page 12

2.3. Setting the Intel FPGA RTE for OpenCL Standard Edition User Environment Variables

You have the option to set the Intel FPGA RTE for OpenCL Standard Edition Windows user environment variables permanently or transiently. The environment variable settings describe the FPGA board and the host runtime to the software.

Attention: If you set the environment variables permanently, you apply the settings once during installation. If you set the environment variables transiently, you must apply the settings during installation and during every subsequent session you run.

Table 2. Intel FPGA RTE for OpenCL Windows User Environment Variable Settings

<table>
<thead>
<tr>
<th>Environment Variable</th>
<th>Path to Include</th>
</tr>
</thead>
<tbody>
<tr>
<td>PATH</td>
<td>1. <code>\bin</code> 2. <code>\windows\bin</code> 3. <code>\host\bin</code> where <code>INTELFPGAOCLSDKROOT</code> points to the path of the software installation</td>
</tr>
</tbody>
</table>
• To apply permanent environment variable settings, perform the following tasks:
  a. Click **Windows Start menu ➤ Control Panel** (or search for and then open the Control Panel application in Windows 8.1 and Windows 10).
  b. Click **System and Security ➤ System**.
  c. In the **System** window, click **Advanced system settings**.
  d. Click the **Advanced** tab in the **System Properties** dialog box.
  e. Click **Environment Variables**.
     The **Environment Variables** dialog box appears.
  f. To modify an existing environment variable setting, select the variable under **User variables for <user_name>** and then click **Edit**. In the **Edit User Variable** dialog box, type the environment variable setting in the **Variable value** field.
  g. If you add a new environment variable, click **New** under **User variables for <user_name>**. In the **New User Variable** dialog box, type the environment variable name and setting in the **Variable name** and **Variable value** fields, respectively.

For an environment variable with multiple settings, add semicolons to separate the settings.

• To apply transient environment variable settings, open a command window and run the `%INTELFPGAOCLSDKROOT%\init_opencl.bat` script.

Example script output:

```
AOCL_BOARD_PACKAGE_ROOT path is not set in environment
Setting to default s5_ref board.
If you want to target another board, do
  set AOCL_BOARD_PACKAGE_ROOT=board_pkg_dir
and re-run this script
Adding %INTELFPGAOCLSDKROOT%\bin to PATH
Adding %INTELFPGAOCLSDKROOT%\host\windows64\bin to PATH
Adding %AOCL_BOARD_PACKAGE_ROOT%\windows64\bin to PATH
```

where **AOCL_BOARD_PACKAGE_ROOT** points to the path of the Custom or Reference Platform.

Running the `init_opencl.bat` script only affects the current command window. The script performs the following tasks:

- Finds the Microsoft Visual Studio installation
- Imports the Microsoft Visual Studio environment to properly set the **LIB** environment variable
- Ensures that the **PATH** environment variable includes the path to the Microsoft **LINK.EXE** file and the **aocl.exe** file

### 2.4. Verifying Software Installation

Invoke the `version` utility command and verify that the correct version of the OpenCL software is installed.

**Attention:** Intel FPGA RTE for OpenCL-supported Intel SoC boards do not support the `version` utility.
At a command prompt, invoke the `aocl version` utility command. An output similar to the one below notifies you of a successful installation:

```
aocl <version>.(<build> Intel(R) FPGA Runtime Environment for OpenCL(TM), Version <version> Build <build>, Copyright (C) <year> Intel Corporation)
```

If installation was unsuccessful, reinstall the software. You can also refer to the `Intel FPGA Software Installation and Licensing` manual and the Intel FPGA Knowledge Base for more information.

**Related Information**
- Intel FPGA Software Installation and Licensing
- Knowledge Base

### 2.5. Installing an FPGA Board

Before creating an OpenCL application for an FPGA board, you must first download and install the Custom Platform from your board vendor. Most Custom Platform installers require administrator privileges. To install your board into a Windows host system, invoke the `install <path_to_customplatform>` utility command.

The steps below outline the board installation procedure. Some Custom Platforms require additional installation tasks. Consult your board vendor's documentation for further information on board installation.

1. Follow your board vendor's instructions to connect the FPGA board to your system.
2. Download the Custom Platform for your FPGA board from your board vendor's website. To download an Intel FPGA SDK for OpenCL Reference Platform, refer to the Intel FPGA SDK for OpenCL FPGA Platforms page.
3. Install the Custom Platform in a folder that you own (that is, not a system folder).
   
   You can install multiple Custom Platforms simultaneously on the same system using the RTE utilities, such as `aocl diagnose` with multiple Custom Platforms, you must set the `AOCL_BOARD_PACKAGE_ROOT` environment variable to point to the location of the Custom Platform subdirectory of the board on which you wish to run the utility. The Custom Platform subdirectory contains the `board_env.xml` file. To run the RTE utilities on a different Custom Platform, you must update the `AOCL_BOARD_PACKAGE_ROOT` environment variable to point to the location of the Custom Platform subdirectory of that specific board.

   In a system with multiple Custom Platforms, ensure that the host program uses the FPGA Client Driver (FCD) to discover the boards rather than linking to the Custom Platforms' memory-mapped device (MMD) libraries directly. As long as FCD is correctly set up for Custom Platform, FCD finds all the installed boards at runtime.

4. Set the user environment variable `AOCL_BOARD_PACKAGE_ROOT` to point to the location of the Custom Platform subdirectory containing the `board_env.xml` file.

   For example, for `s5_net`, set `AOCL_BOARD_PACKAGE_ROOT` to point to the `<path_to_s5_net>/s5_net` directory.
Note: If you ran the $INTELFPGAOCLSDKROOT/init_opencl.sh script to set the SDK user environment variables, the script has set AOCL_BOARD_PACKAGE_ROOT to point to $INTELFPGAOCLSDKROOT/board/s5_ref, by default.

5. Add the paths to the Custom Platform libraries (for example, the memory-mapped (MMD) library) to the PATH environment variable setting.

For example, if you use an Intel FPGA SDK for OpenCL Reference Platform, the Windows PATH environment variable setting is %AOCL_BOARD_PACKAGE_ROOT%\windows64\bin.

For information on setting user environment variables and running the init_opencl script, refer to the Setting the Intel FPGA RTE for OpenCL Standard Edition User Environment Variables section.

6. Invoke the command aocl install <path_to_customplatform> at a command prompt.

Invoking aocl install <path_to_customplatform> also installs a board driver that allows communication between host applications and hardware kernel programs.

Remember: You need administrative rights to install a board. To run a Windows command prompt as an administrator, click Start ➤ All Programs ➤ Accessories. Under Accessories, right click Command Prompt, In the right-click menu, click Run as Administrator.

7. To query a list of FPGA devices installed in your machine, invoke the aocl diagnose command.

The software generates an output that includes the <device_name>, which is an acl number that ranges from acl0 to acl31.

Attention: For more information on querying the <device_name> of your accelerator board, refer to the Querying the Device Name of Your FPGA Board section.

8. To verify the successful installation of the FPGA board, invoke the command aocl diagnose <device_name> to run any board vendor-recommended diagnostic test.

Related Information
- Setting the Intel FPGA RTE for OpenCL Standard Edition User Environment Variables on page 12
- Querying the Device Name of Your FPGA Board on page 16
- Installing the Cyclone V SoC Development Kit on page 36
- Intel FPGA SDK for OpenCL FPGA Platforms

2.6. Updating the Hardware Image on the FPGA

If applicable, before you execute an OpenCL kernel program on the FPGA, ensure that the flash memory of the FPGA contains a hardware image created using a current version of the OpenCL software.
Remember: • If your Custom Platform requires that you preload a valid OpenCL image into the flash memory, for every major release of the Intel Quartus Prime Design Suite, program the flash memory of the FPGA with a hardware image compatible with the current version of the software.

2.6.1. Querying the Device Name of Your FPGA Board

Some OpenCL software utility commands require you to specify the device name (<device_name>). The <device_name> refers to the acl number (e.g. acl0 to acl31) that corresponds to the FPGA device. When you query a list of accelerator boards, the OpenCL software produces a list of installed devices on your machine in the order of their device names.

• To query a list of installed devices on your machine, type aocl diagnose at a command prompt. The software generates an output that resembles the example shown below:

```
aocl diagnose: Running diagnostic from INTELFPGAOCLSDKROOT/board/<board_name>/platform(libexec)
Verified that the kernel mode driver is installed on the host machine.
Using board package from vendor: <board_vendor_name>
Querying information for all supported devices that are installed on the host machine ...

device_name  Status  Information
acl0          Passed  <descriptive_board_name>
              PCIe dev_id = <device_ID>, bus:slot.func = 02:00.00, at Gen 2 with 8 lanes.
              FPGA temperature = 43.0 degrees C.

acl1          Passed  <descriptive_board_name>
              PCIe dev_id = <device_ID>, bus:slot.func = 03:00.00, at Gen 2 with 8 lanes.
              FPGA temperature = 35.0 degrees C.

Found 2 active device(s) installed on the host machine, to perform a full diagnostic on a specific device, please run aocl diagnose <device_name>
```

2.6.2. Programming the Flash Memory of an FPGA

Configure the FPGA by loading the hardware image of an Intel FPGA RTE for OpenCL design example into the flash memory of the device. When there is no power, the FPGA retains the hardware configuration file in the flash memory. When you power up the system, it configures the FPGA circuitry based on this hardware image in the flash memory. Therefore, it is imperative that an OpenCL-compatible hardware configuration file is loaded into the flash memory of your FPGA.

Preloading an OpenCL image into the flash memory is necessary for the proper functioning of many Custom Platforms. For example, most PCIe®-based boards require a valid OpenCL image in flash memory so that hardware on the board can use the image to configure the FPGA device when the host system powers up for the first time. If the FPGA is not configured with a valid OpenCL image, the system will fail to enumerate the PCIe endpoint, or the driver will not function.
Before running any designs, ensure that the flash memory of your board has a valid OpenCL image that is compatible with the current OpenCL software version. Consult your board vendor’s documentation for board-specific requirements.

Caution:
When you load the hardware configuration file into the flash memory of the FPGA, maintain system power for the entire loading process, which might take a few minutes. Also, do not launch any host code that calls OpenCL kernels or might otherwise communicate with the FPGA board.

To load your hardware configuration file into the flash memory of your FPGA board, perform the following tasks:

1. Install any drivers or utilities that your Custom Platform requires.
   For example, some Custom Platforms require you to install the Intel FPGA Download Cable driver to load your hardware configuration file into the flash memory. For installation instructions, refer to the Intel FPGA Download Cable User Guide.

2. Verify that you set the AOCL_BOARD_PACKAGE_ROOT environment variable to point to the subfolder in your Custom Platform that contains the board_env.xml file. Open a Windows command window and type echo %AOCL_BOARD_PACKAGE_ROOT% at the command prompt. If the returned path does not point to the location of the board_env.xml file within your Custom Platform, follow the instructions in Setting the Intel FPGA RTE for OpenCL Standard Edition User Environment Variables to modify the environment variable setting.

3. Download a design example for your Custom Platform.
   Remember: Download design examples from the OpenCL Design Examples page, and extract the example to a location that you have write access for. Ensure that the location name does not contain spaces.

4. To load the hardware configuration file into the flash memory, invoke the aocl flash <device_name> <design_example_filename>.aocx command, where <device_name> refers to the acl number (e.g. acl0 to acl31) that corresponds to your FPGA device, and <design_example_filename>.aocx is the hardware configuration file you create from the <design_example_filename>.cl file in the design example package.

5. Power down your device or computer and then power it up again.
   Power cycling ensures that the FPGA configuration device retrieves the hardware configuration file from the flash memory and configures it into the FPGA.

   Warning: Some Custom Platforms require you to power cycle the entire host system after programming the flash memory. For example, PCIe-based Custom Platforms might require a host system restart to reenumerate the PCIe endpoint. Intel recommends that you power cycle the complete host system after programming the flash memory.

Related Information
- Intel FPGA Download Cable II User Guide
- OpenCL Design Examples
2.7. Executing an OpenCL Kernel on an FPGA

Build your OpenCL host application in Microsoft Visual Studio, and run the application by invoking the `hello_world.exe` executable. The Intel FPGA RTE for OpenCL is compatible with 64-bit host binaries only.

**Related Information**
- Creating the Hardware Configuration File of an OpenCL Kernel for SoC FPGA
- OpenCL Design Examples

### 2.7.1. Building the Host Application

The `<path_to_exm_opencl_hello_world_x64_windows_<version>>\hello_world\hello_world.sln` file contains the host solution. After you open this `.sln` file in Microsoft Visual Studio, you can build the OpenCL host application in the `main.cpp` file.

If you are using Microsoft Visual Studio, you need the FCD, and the Installable Client Driver (ICD) from Khronos. To set up Microsoft Visual Studio with FCD and ICD, perform the following tasks prior to building the host application:

1. Verify that FCD and ICD are set up correctly. You must set up FCD and ICD manually if invoking the `aocl install <path_to_customplatform>` utility command fails to set them up. For instructions, refer to the Accessing Custom Platform-Specific Functions and Linking to the ICD Loader Library on Windows sections of the Intel FPGA RTE for OpenCL Standard Edition Programming Guide for more information.

2. Link the host application to the `OpenCL.lib` library.
   a. Under the solution properties, select **Configuration Properties ➤ Linker ➤ Input**.
   b. In the Additional Dependencies field, enter `OpenCL.lib`.
      
      **Attention:** Because you are using FCD and ICD, do not link the host program to `alteracl.lib` or to your Custom Platform’s MMD libraries directly.

To build the `hello_world` host application, perform the following tasks:

1. Open the `<path_to_exm_opencl_hello_world_x64_windows_<version>>\hello_world\hello_world.sln` file in Microsoft Visual Studio.

2. Verify that the build configuration is correct. The default build configuration is **Debug**, but you can use **Release**. You must select the appropriate option as the solution platform (for example, for x64 architecture, select **x64**).

3. Build the solution by selecting the **Build ➤ Build Solution** menu option, or by pressing the F7 key.

   The `hello_world.exe` executable will be in the `<path_to_exm_opencl_hello_world_x64_windows_<version>>\hello_world\bin` folder.

4. Verify that the build is correct. An output ending with a message similar to the one shown below notifies you of a successful build:
1> Build succeeded.
1> Time Elapsed 00:00:03:29
--------------- Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped --------------

**Attention:** You can ignore the LNK4009: PDB 'vc90.pdb' was not found with... warnings because they have no effect on the build. The compiler might issue this type of warning messages if you have built your Windows libraries using a previous version of Microsoft Visual Studio.

**Related Information**
- Accessing Custom Platform-Specific Functions
- Linking to the ICD Loader Library on Windows

### 2.7.2. Running the Host Application

To execute the OpenCL kernel on the FPGA, run the Windows host application that you built from the `.sln` file.

1. Add the path `%INTELFPAGOCLSDKROOT%\host\windows64\bin` to the `PATH` environment variable.

2. At a command prompt, navigate to the host executable within the `<path_to_exm_opencl_hello_world_x64_windows_<version>>\hello_world\bin` folder.

3. Invoke the `hello_world.exe` executable. The `hello_world` executable executes the kernel code on the FPGA.

### 2.7.3. Output from Successful Kernel Execution

When you run the host application to execute your OpenCL kernel on the target FPGA, the OpenCL software notifies you of a successful kernel execution.

Example output:

```
Reprogramming device [0] with handle 1
Querying platform for info:
---------------------------
CL_PLATFORM_NAME = Intel(R) FPGA SDK for OpenCL(TM)
CL_PLATFORM_VENDOR = Intel Corporation
CL_PLATFORM_VERSION = OpenCL 1.0 Intel(R) FPGA SDK for OpenCL(TM), <version>

Querying device for info:
------------------------
CL_DEVICE_NAME = <board name>: <descriptive board name>
CL_DEVICE_VENDOR = <board vendor name>
CL_DEVICE_VENDOR_ID = <board vendor ID>
CL_DEVICE_VERSION = OpenCL 1.0 Intel(R) FPGA SDK for OpenCL(TM), <version>
CL_DRIVER_VERSION = <version>
CL_DEVICE_ADDRESS_BITS = 64
CL_DEVICE_AVAILABLE = true
CL_DEVICE_ENDIAN_LITTLE = true
CL_DEVICE_GLOBAL_MEM_CACHE_SIZE = 32768
CL_DEVICE_GLOBAL_MEM_CACHELINE_SIZE = 0
CL_DEVICE_GLOBAL_MEM_SIZE = 8589934592
CL_DEVICE_IMAGE_SUPPORT = true
CL_DEVICE_LOCAL_MEM_SIZE = 16384
CL_DEVICE_MAX_CLOCK_FREQUENCY = 1000
```
2.8. Uninstalling the Software

To uninstall the Intel FPGA RTE for OpenCL Standard Edition for Windows, delete the RTE folder and restore all modified environment variables to their previous settings.

1. In Windows Explorer, navigate to the altera\<version>\<edition> folder.
2. Delete the aclrte-windows64 folder.
3. Remove the following paths from the PATH environment variable:
   a. %INTELFPGAOCLSDKROOT%\bin
   b. %INTELFPGAOCLSDKROOT%\host\windows64\bin
4. Remove the INTELFPGAOCLSDKROOT environment variable.

2.9. Uninstalling the FPGA Board

To uninstall an FPGA board for Windows, invoke the uninstall utility command, uninstall the Custom Platform, and unset the relevant environment variables. You must uninstall the existing FPGA board if you migrate your OpenCL application to another FPGA board that belongs to a different Custom Platform.

To uninstall your FPGA board, perform the following tasks:

1. Following your board vendor’s instructions to disconnect the board from your machine.
2. Invoke the aocl uninstall <path_to_customplatform> utility command to remove the current host computer drivers (for example, PCIe drivers). The Intel FPGA RTE for OpenCL uses these drivers to communicate with the FPGA board.
3. Uninstall the Custom Platform.
4. Unset the PATH environment variable.
5. Unset the AOCL_BOARD_PACKAGE_ROOT environment variable.
3. Getting Started with the Intel FPGA RTE for OpenCL Standard Edition for x86_64 Linux Systems

Figure 1 on page 10 outlines the RTE setup process for x86_64 Linux systems.

1. Downloading the Intel FPGA RTE for OpenCL Standard Edition on page 21
2. Installing the Intel FPGA RTE for OpenCL on page 22
4. Verifying Software Installation on page 23
5. Installing an FPGA Board on page 24
6. Updating the Hardware Image on the FPGA on page 25
7. Executing an OpenCL Kernel on an FPGA on page 27
8. Uninstalling the Software on page 29
9. Uninstalling the FPGA Board on page 29

3.1. Downloading the Intel FPGA RTE for OpenCL Standard Edition

Download the Intel FPGA RTE for OpenCL Standard Edition for Linux from the Download Center.

1. Go to the Intel FPGA RTE for OpenCL Download Center at the following URL:
   http://fpgasoftware.intel.com/opencl/
3. Select the software version. The default selection is the current version.
4. Select one of the following download methods:
   — Akamai DLM3 Download Manager
   — Direct Download
5. Click More beside Download and install instructions if you want to see the download and installation instructions.
6. Click the RTE tab and select the installation package you want to download. Click More beside Download and install instructions to view the download and installation procedure.
7. Click the download button to start the download process.
8. Perform the steps outlined in the download and installation instructions on the download page.

Related Information

Intel FPGA website
3.2. Installing the Intel FPGA RTE for OpenCL

Install the Linux version of the Intel FPGA RTE for OpenCL Standard Edition in a directory that you own.

- You must have `sudo` or `root` privileges.
- You must install the Linux OS kernel source and headers (for example, `kernel-devel.x86_64` and `kernel-headers.x86_64`), and the GNU Compiler Collection (GCC) (`gcc.x86_64`).
- If you are installing a package that includes Intel Code Builder, you must have Java SE 1.8.71 or later installed to run Intel Code Builder. If you have an earlier version of Java SE installed, you can still complete the installation of Intel Code Builder. However, you must meet the Java version prerequisite to run Intel Code Builder.

**Attention:** If you install the software on a system that does not contain any C Shell Run Commands file (`.cshrc`) or Bash Run Commands file (`.bashrc`) in your directory, you must set the environment variables `INTELFPGAOCLSDKROOT` and `PATH` manually. Alternatively, you may create the `.cshrc` and `.bashrc` files, and then append the environment variables to them. To ensure that the updates take effect, restart your terminal after you set the environment variables.

To install the Intel FPGA RTE for OpenCL, perform the following tasks:

1. At the command prompt, type the RPM command to install the downloaded RPM package.
   
   **Note:** The installation path must not contain any spaces (for example, `/usr/intelfpga/<version>/aclrte_linux64`).
   
   - To install the software using the Red Hat Package Manager (RPM), at the command prompt, type the `rpm -i accl-rte-<version>.x86_64.rpm` command.
   
   The RPM installs the software in the default location (for example, `opt/intelfpga/aclrte-linux64`).
   
   - To install the software in the default location with verbose progress reporting, type` rpm -ivh accl-rte-<version>.x86_64.rpm`
   
   - To install the software in an alternate directory that you own (that is, not a system directory), type` rpm -i --prefix <rte_destination_directory> accl-rte-<version>.x86_64.rpm`

2. **Note:** The installer sets the environment variable `INTELFPGAOCLSDKROOT` to point to the path of the software installation.

   Verify that `INTELFPGAOCLSDKROOT` points to the current version of the software. Open a shell and then type `echo $INTELFPGAOCLSDKROOT` at the command prompt.

   If the returned path does not point to the location of the Intel FPGA RTE for OpenCL installation, edit the `INTELFPGAOCLSDKROOT` setting.

**Related Information**

- Setting the Intel FPGA RTE for OpenCL Standard Edition User Environment Variables on page 23
3.3. Setting the Intel FPGA RTE for OpenCL Standard Edition User Environment Variables

You have the option to set the Intel FPGA RTE for OpenCL Standard Edition Linux user environment variables permanently or transiently. The environment variable settings describe the FPGA board and the host runtime to the software.

Attention: If you set the environment variables permanently, you apply the settings once during installation. If you set the environment variables transiently, you must apply the settings during installation and during every subsequent session you run.

Table 3. Intel FPGA RTE for OpenCL Linux User Environment Variable Settings

<table>
<thead>
<tr>
<th>Environment Variable</th>
<th>Path to Include</th>
</tr>
</thead>
<tbody>
<tr>
<td>PATH</td>
<td>$INTELFPGAOCLSDKROOT/bin where INTELFPGAOCLSDKROOT points to the path of the software installation</td>
</tr>
<tr>
<td>LD_LIBRARY_PATH</td>
<td>$INTELFPGAOCLSDKROOT/host/linux64/lib $AOCL_BOARD_PACKAGE_ROOT/linux64/lib where AOCL_BOARD_PACKAGE_ROOT points to the path of the Custom or Reference Platform</td>
</tr>
</tbody>
</table>

- To apply permanent environment variable settings, open a shell and then type the `export <variable_name>="<variable_setting>"` command.

  For example, the command `export PATH="$INTELFPGAOCLSDKROOT/bin":$PATH` adds $INTELFPGAOCLSDKROOT/bin to the list of PATH settings.

- To apply transient environment variable settings, open a bash-shell command-line terminal and run the `source $INTELFPGAOCLSDKROOT/init_opencl.sh` command. This command does not work in other shells.

Example script output:

```
AOCL_BOARD_PACKAGE_ROOT path is not set in environment
Setting to default s5_ref board.
If you want to target another board, do
    set AOCL_BOARD_PACKAGE_ROOT=board_pkg_dir
Adding $INTELFPGAOCLSDKROOT/bin to PATH
Adding $INTELFPGAOCLSDKROOT/host/linux64/lib to LD_LIBRARY_PATH
Adding $AOCL_BOARD_PACKAGE_ROOT/linux64/lib to LD_LIBRARY_PATH
```

where AOCL_BOARD_PACKAGE_ROOT points to the path of the Custom or Reference Platform.

3.4. Verifying Software Installation

Invoke the `version` utility command and verify that the correct version of the OpenCL software is installed.

Attention: Intel FPGA RTE for OpenCL-supported Intel SoC boards do not support the `version` utility.
• At a command prompt, invoke the `aocl version` utility command. An output similar to the one below notifies you of a successful installation:

```
aocl <version>.<build> (Intel(R) FPGA Runtime Environment for OpenCL(TM), Version <version> Build <build>, Copyright (C) <year> Intel Corporation)
```

• If installation was unsuccessful, reinstall the software. You can also refer to the *Intel FPGA Software Installation and Licensing* manual and the Intel FPGA Knowledge Base for more information.

### Related Information
- Intel FPGA Software Installation and Licensing
- Knowledge Base

### 3.5. Installing an FPGA Board

Before creating an OpenCL application for an FPGA board on Linux, you must first download and install the Custom Platform from your board vendor. Most Custom Platform installers require administrator privileges. To install your board into a Linux host system, invoke the `install` utility command.

The steps below outline the board installation procedure. Some Custom Platforms require additional installation tasks. Consult your board vendor's documentation for further information on board installation.

1. Follow your board vendor's instructions to connect the FPGA board to your system.
2. Download the Custom Platform for your FPGA board from your board vendor's website. To download an Intel FPGA RTE for OpenCL Reference Platform, refer to the Intel FPGA RTE for OpenCL FPGA Platforms page.
3. Install the Custom Platform in a directory that you own (that is, not a system directory).

You can install multiple Custom Platforms simultaneously on the same system. To use the RTE utilities, such as `aocl diagnose` with multiple Custom Platforms, you must set the `AOCL_BOARD_PACKAGE_ROOT` environment variable to point to the location of the Custom Platform subdirectory of the board on which you wish to run the utility. The Custom Platform subdirectory contains the `board_env.xml` file. To run the RTE utilities on a different Custom Platform, you must update the `AOCL_BOARD_PACKAGE_ROOT` environment variable to point to the location of the Custom Platform subdirectory of that specific board.

In a system with multiple Custom Platforms, ensure that the host program uses the FPGA Client Drivers (FCD) to discover the boards rather than linking to the Custom Platforms' memory-mapped device (MMD) libraries directly. If FCD is correctly set up for Custom Platform, FCD finds all the installed boards at runtime.

4. Set the user environment variable `AOCL_BOARD_PACKAGE_ROOT` to point to the location of the Custom Platform subdirectory containing the `board_env.xml` file.

For example, for s5_net, set `AOCL_BOARD_PACKAGE_ROOT` to point to the `<path_to_s5_net>/s5_net` directory.

**Note:** If you ran the `$INTELFPGAOCLSDKROOT/init_opencl.sh` script to set the SDK user environment variables, the script has set `AOCL_BOARD_PACKAGE_ROOT` to point to `$INTELFPGAOCLSDKROOT/board/s5_ref`, by default.
5. Add the paths to the Custom Platform libraries (for example, memory-mapped (MMD) library) to the `LD_LIBRARY_PATH` environment variable setting.

For example, if you use an Intel FPGA RTE for OpenCL Reference Platform, the Linux `LD_LIBRARY_PATH` setting is 

```
$AOCL_BOARD_PACKAGE_ROOT/linux64/lib
```

For information on setting Linux user environment variables and running the `init_opencl` script, refer to the Setting the Intel FPGA RTE for OpenCL Standard Edition User Environment Variables section.

6. **Remember:** You need `sudo` or `root` privileges to install a board.

Invoke the command `aocl install <path_to_customplatform>` at a command prompt.

Invoking `aocl install <path_to_customplatform>` also installs a board driver that allows communication between host applications and hardware kernel programs.

7. To query a list of FPGA devices installed in your machine, invoke the `aocl diagnose` command.

The software generates an output that includes the `<device_name>`, which is an acl number that ranges from acl0 to acl31.

**Attention:** For more information on querying the `<device_name>` of your accelerator board, refer to the Querying the Device Name of Your FPGA Board section.

8. To verify the successful installation of the FPGA board, invoke the command `aocl diagnose <device_name>` to run any board vendor-recommended diagnostic test.

**Related Information**
- Installing the Cyclone V SoC Development Kit on page 47
- Querying the Device Name of Your FPGA Board on page 26
- Setting the Intel FPGA RTE for OpenCL Standard Edition User Environment Variables on page 23
- Intel FPGA SDK for OpenCL FPGA Platforms

### 3.6. Updating the Hardware Image on the FPGA

If applicable, before you execute an OpenCL kernel program on the FPGA, ensure that the flash memory of the FPGA contains a hardware image created using a current version of the OpenCL software.

**Remember:**
- If your Custom Platform requires that you preload a valid OpenCL image into the flash memory, for every major release of the Intel Quartus Prime Design Suite, program the flash memory of the FPGA with a hardware image compatible with the current version of the software.
**3.6.1. Querying the Device Name of Your FPGA Board**

Some OpenCL software utility commands require you to specify the device name \(<device\_name>\). The \(<device\_name>\) refers to the acl number (e.g. acl0 to acl31) that corresponds to the FPGA device. When you query a list of accelerator boards, the OpenCL software produces a list of installed devices on your machine in the order of their device names.

- To query a list of installed devices on your machine, type `aocl diagnose` at a command prompt.
  
The software generates an output that resembles the example shown below:

```plaintext
aocl diagnose: Running diagnostic from INTELFPGAOCSDKROOT/board/<board_name>/platform/libexec
Verified that the kernel mode driver is installed on the host machine.
Using board package from vendor: <board\_vendor\_name>
Querying information for all supported devices that are installed on the host machine ...

<table>
<thead>
<tr>
<th>device_name</th>
<th>Status</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>acl0</td>
<td>Passed</td>
<td>PCIe dev_id = &lt;device_ID&gt;, bus:slot.func = 02:00.00, at Gen 2 with 8 lanes. FPGA temperature = 43.0 degrees C.</td>
</tr>
<tr>
<td>acl1</td>
<td>Passed</td>
<td>PCIe dev_id = &lt;device_ID&gt;, bus:slot.func = 03:00.00, at Gen 2 with 8 lanes. FPGA temperature = 35.0 degrees C.</td>
</tr>
</tbody>
</table>

Found 2 active device(s) installed on the host machine, to perform a full diagnostic on a specific device, please run aocl diagnose <device\_name>

DIAGNOSTIC\_PASSED
```

**3.6.2. Programming the Flash Memory of an FPGA**

Configure the FPGA by loading the hardware image of an Intel FPGA RTE for OpenCL design example into the flash memory of the device. When there is no power, the FPGA retains the hardware configuration file in the flash memory. When you power up the system, it configures the FPGA circuitry based on this hardware image in the flash memory. Therefore, it is imperative that an OpenCL-compatible hardware configuration file is loaded into the flash memory of your FPGA.

Preloading an OpenCL image into the flash memory is necessary for the proper functioning of many Custom Platforms. For example, most PCIe-based boards require a valid OpenCL image in flash memory so that hardware on the board can use the image to configure the FPGA device when the host system powers up for the first time. If the FPGA is not configured with a valid OpenCL image, the system will fail to enumerate the PCIe endpoint, or the driver will not function.

Before running any designs, ensure that the flash memory of your board has a valid OpenCL image that is compatible with the current OpenCL software version. Consult your board vendor’s documentation for board-specific requirements.
Caution: When you load the hardware configuration file into the flash memory of the FPGA, maintain system power for the entire loading process, which might take a few minutes. Also, do not launch any host code that calls OpenCL kernels or might otherwise communicate with the FPGA board.

To load your hardware configuration file into the flash memory of your FPGA board, perform the following tasks:

1. Install any drivers or utilities that your Custom Platform requires.
2. Verify that you set the `AOCL_BOARD_PACKAGE_ROOT` environment variable to point to the subfolder in your Custom Platform that contains the `board_env.xml` file. Open a shell and then type `echo $AOCL_BOARD_PACKAGE_ROOT` at the command prompt.
   If the returned path does not point to the location of the `board_env.xml` file within your Custom Platform, follow the instructions in Setting the Intel FPGA RTE for OpenCL Standard Edition User Environment Variables to modify the environment variable setting.
3. Download a design example for your Custom Platform.
   *Remember:* Download design examples from the OpenCL Design Examples page, and extract the example to a location to which you have write access. Ensure that the location name does not contain spaces.
4. To load the hardware configuration file into the flash memory, invoke the `aocl flash <device_name> <design_example_filename>.aocx` command, where `<device_name>` refers to the acl number (e.g. acl0 to acl31) that corresponds to your FPGA device, and `<design_example_filename>.aocx` is the hardware configuration file you create from the `<design_example_filename>.cl` file in the example design package.
5. Power down your device or computer and then power it up again.
   Power cycling ensures that the FPGA configuration device retrieves the hardware configuration file from the flash memory and configures it into the FPGA.

   **Warning:** Some Custom Platforms require you to power cycle the entire host system after programming the flash memory. For example, PCIe-based Custom Platforms might require a host system restart to reenumerate the PCIe endpoint. Intel recommends that you power cycle the complete host system after programming the flash memory.

**Related Information**

OpenCL Design Examples

### 3.7. Executing an OpenCL Kernel on an FPGA

You must build your OpenCL host application with the `Makefile` file, and run the application by invoking the `hello_world` executable. You need GNU development tools such as `gcc` and `make` to build the OpenCL application.

### 3.7.1. Building the Host Application

Build the host executable with the
`<path_to_exm_opencl_hello_world_x64_linux_<version>>/hello_world/Makefile` file.
To build the host application, perform the following tasks:

1. Navigate to the hello_world directory.

2. Invoke the `make -f Makefile` command. Alternatively, you can simply invoke the `make` command.
   
   The hello_world executable will be in the `<path_to_exm_opencl_hello_world_x64_linux_<version>>/hello_world/bin` directory.

### 3.7.2. Running the Host Application

To execute the OpenCL kernel on the FPGA, run the Linux host application that you built from the Makefile.

1. Add the path `$INTELFPGAOCLSDKROOT/host/linux64/lib` to the `LD_LIBRARY_PATH` environment variable.

2. At a command prompt, navigate to the host executable within the `<path_to_exm_opencl_hello_world_x64_linux_<version>>/hello_world/bin` directory.

3. Invoke the hello_world executable.
   
   The hello_world executable executes the kernel code on the FPGA.

### 3.7.3. Output from Successful Kernel Execution

When you run the host application to execute your OpenCL kernel on the target FPGA, the OpenCL software notifies you of a successful kernel execution.

Example output:

```
Reprogramming device [0] with handle 1
Querying platform for info:
--------------------------
CL_PLATFORM_NAME          = Intel(R) FPGA SDK for OpenCL(TM)
CL_PLATFORM_VENDOR       = Intel Corporation
CL_PLATFORM_VERSION      = OpenCL 1.0 Intel(R) FPGA SDK for OpenCL(TM), <version>
--------------------------
Querying device for info:
------------------------
CL_DEVICE_NAME            = <board name> : <descriptive board name>
CL_DEVICE_VENDOR         = <board vendor name>
CL_DEVICE_VENDOR_ID      = <board vendor ID>
CL_DEVICE_VERSION        = OpenCL 1.0 Intel(R) FPGA SDK for OpenCL(TM), <version>
CL_DRIVER_VERSION        = <version>
CL_DEVICE_ADDRESS_BITS   = 64
CL_DEVICE_AVAILABLE      = true
CL_DEVICE_ENDIAN_LITTLE  = true
CL_DEVICE_GLOBAL_MEM_CACHE_SIZE = 32768
CL_DEVICE_GLOBAL_MEM_CACHELINE_SIZE = 0
CL_DEVICE_GLOBAL_MEM_SIZE = 8589934592
CL_DEVICE_IMAGE_SUPPORT  = true
CL_DEVICE_LOCAL_MEM_SIZE = 16384
CL_DEVICE_MAX_CLOCK_FREQUENCY = 1000
CL_DEVICE_MAX_COMPUTE_UNITS = 1
CL_DEVICE_MAX_CONSTANT_ARGS = 8
CL_DEVICE_MAX_CONSTANT_BUFFER_SIZE = 2147483648
CL_DEVICE_MAX_WORK_ITEMDIMENSIONS = 3
CL_DEVICE_MEM_BASE_ADDR_ALIGN = 8192
CL_DEVICE_MIN_DATA_TYPE_ALIGN_SIZE = 1024
CL_DEVICE_PREFERRED_VECTOR_WIDTH_CHAR = 4
CL_DEVICE_PREFERRED_VECTOR_WIDTH_SHORT = 2
```
CL_DEVICE_PREFERRED_VECTOR_WIDTH_INT = 1
CL_DEVICE_PREFERRED_VECTOR_WIDTH_LONG = 1
CL_DEVICE_PREFERRED_VECTOR_WIDTH_FLOAT = 1
CL_DEVICE_PREFERRED_VECTOR_WIDTH_DOUBLE = 0
Command queue out of order? = false
Command queue profiling enabled? = true
Using AOCX: hello_world.aocx
Kernel initialization is complete.
Launching the kernel...
Thread #2: Hello from the Intel(R) FPGA OpenCL(TM) compiler!
Kernel execution is complete.

3.8. Uninstalling the Software

To uninstall the Intel FPGA RTE for OpenCL Standard Edition for Linux, remove the software package via the RPM uninstaller, then delete the software directory and restore all modified environment variables to their previous settings.

1. Remove the software package by performing one of the following tasks:
   a. To uninstall the RTE, type the rpm -e acl-rte command.
2. Remove $INTELFPGAOCLSDKROOT/bin from the PATH environment variable.
3. Remove $INTELFPGAOCLSDKROOT/host/linux64/lib from the LD_LIBRARY_PATH environment variable.
4. Remove the INTELFPGAOCLSDKROOT environment variable.

3.9. Uninstalling the FPGA Board

To uninstall an FPGA board for Linux, invoke the uninstall utility command, uninstall the Custom Platform, and unset the relevant environment variables. You must uninstall the existing FPGA board if you migrate your OpenCL application to another FPGA board that belongs to a different Custom Platform.

To uninstall your FPGA board, perform the following tasks:
1. Disconnect the board from your machine by following the instructions provided by your board vendor.
2. Invoke the acl uninstall <path_to_customplatform> utility command to remove the current host computer drivers (for example, PCIe drivers). The Intel FPGA RTE for OpenCL uses these drivers to communicate with the FPGA board.
3. Uninstall the Custom Platform.
4. Unset the LD_LIBRARY_PATH environment variable.
5. Unset the AOCL_BOARD_PACKAGE_ROOT environment variable.
4. Getting Started with the Intel FPGA RTE for OpenCL Standard Edition for Intel ARMv7-A SoC FPGA

The following sections provide instructions for setting up Windows and Linux versions of the RTE for use with the Cyclone V SoC Development Kit.

Figure 2. Key Components of the Cyclone V SoC Development Kit
Figure 3. RTE Setup Process for SoC FPGA

The figure below outlines the steps for installing the software and the SoC FPGA board, and in executing an OpenCL kernel on the SoC FPGA board.

4.1. Getting Started with the Intel FPGA RTE for OpenCL Standard Edition for Intel ARMv7-A SoC FPGA on Windows

To execute an OpenCL kernel onto an SoC FPGA, install Windows versions of the Intel FPGA SDK for OpenCL Standard Edition and the Intel SoC FPGA Embedded Development Suite (EDS) Standard Edition on your host system, and install the RTE on your SoC FPGA board. You must also build your host application using an ARM-specific Makefile.

1. **Downloading the Intel FPGA SDK for OpenCL Standard Edition and the SoC EDS Standard Edition** on page 32
4. Getting Started with the Intel FPGA RTE for OpenCL Standard Edition for Intel ARMv7-A SoC FPGA


The SDK includes the SD card image you need to recompile the OpenCL Linux kernel driver. If you wish to recompile the Linux kernel driver and write the SD card image on your own, download the RTE Standard Edition for SoC instead.
To download the SDK, follow the instructions outlined in the Downloading the Intel FPGA SDK for OpenCL Standard Edition section of the Intel FPGA SDK for OpenCL Standard Edition Getting Started Guide.

To download the RTE, perform the following tasks:
   a. Go to the Intel FPGA SDK for OpenCL Download Center at the following URL: http://fpgassoftware.intel.com/opencl/
   c. Select the software version. The default selection is the current version.
   d. Click the RTE tab. Click More beside Download and install instructions to view the download and installation procedure.
   e. Click the download button beside Intel FPGA Runtime Environment for OpenCL Linux Cyclone V SoC TGZ to start the download process.
   f. Perform the steps outlined in the download and installation instructions on the download page.

Download the SoC EDS by performing the following steps:
   a. Go to the Intel FPGA Download Center at the following URL: http://dl.altera.com,
   b. Click Embedded Software ➤ SoC EDS to enter the download page for the subscription edition of the SoC EDS.
   d. Select the software version. The default selection is the current version.
   e. Select Windows as the operating system.
   f. Select Akamai DLM3 Download Manager or Direct Download as the download method.
   g. If you select Akamai DLM3 Download Manager as the download method, click Download.
   h. If you select Direct Download as the download method, click Intel SoC FPGA Embedded Development Suite (EDS).
   i. Perform the steps outlined in the download and installation instructions on the download page.

Related Information
   • Intel FPGA website
   • Downloading the Intel FPGA SDK for OpenCL Standard Edition

4.1.2. Installing the Intel FPGA SDK for OpenCL Standard Edition for SoC FPGA

The Intel FPGA SDK for OpenCL Cyclone V SoC Development Kit Reference Platform (c5soc) includes an SD flash card image necessary for running OpenCL applications on the board. The SD flash card image includes the recompiled Linux kernel driver, preinstalled version of the Intel FPGA RTE for OpenCL Standard Edition, and a script for setting environment variables.
To get started with the RTE on the Cyclone V SoC Development Kit using the SD flash card image that comes with the SDK, install the SDK Standard Edition for Windows. If you want to create your own SD card image, install the RTE Standard Edition.

You must have administrator privileges.

- To install the SDK, perform the following tasks:
  a. Unpack the downloaded tar file into a folder that you own.
     The installation path must not contain any spaces (for example, `<home_directory>\intelfpga\<version>\hld`).
  b. Run the `setup.bat` file to install the SDK and device support.

- To install the RTE, unpack the `.tgz` file and install the RTE in a folder that you own.

**Note:** The installer sets the environment variable `INTELFPGAOCLSDKROOT` to point to the path of the software installation.

Verify that the installer sets the user environment variable `INTELFPGAOCLSDKROOT` to point to the current version of the software. Open a Windows command window and then type `echo %INTELFPGAOCLSDKROOT%` at the command prompt. If the returned path does not point to the location of the current SDK installation, or if the path is not set, modify the `INTELFPGAOCLSDKROOT` setting.

**Related Information**
Setting the Intel FPGA RTE for OpenCL Standard Edition User Environment Variables on page 12

### 4.1.3. Installing the Intel SoC FPGA Embedded Development Suite Standard Edition

Install the SoC EDS Standard Edition for Windows to build your host application for OpenCL kernel deployment on an SoC board.

1. Run the installer. Follow the installation instructions in the `SoCEDSSetup-<version>-windows.exe` executable. For more information, refer to the [Installing the SoC EDS section of the Intel SoC FPGA Embedded Development Suite User Guide](#).


   For more information on the Arm DS-5 Intel SoC FPGA Edition Toolkit, refer to the Arm DS-5 Intel SoC FPGA Edition page of the ARM website.


**Related Information**
- [Installing the ARM DS-5 Intel SoC FPGA Edition Toolkit](#)
- [SoC FPGA EDS Licensing](#)
- [Installing the SoC FPGA EDS](#)
4.1.4. Recompiling the Linux Kernel Driver

**Attention:** If you download and install the Intel FPGA SDK for OpenCL Standard Edition, the Cyclone V SoC Development Kit Reference Platform (INTELFPGAOCLSDKROOT/board/c5soc) includes an SD card image (linux_sd_card_image.tgz) that contains the recompiled Linux kernel driver.

If you need to rebuild the Linux kernel driver, recompile the aclsoc Linux kernel driver to the exact version of the Linux kernel running on the SoC FPGA.

**Important:** You must recompile the aclsoc Linux kernel driver on your Linux development machine.

1. Unpack the aocl-rte-<version>.arm32.tgz tarball to a temporary directory on your development machine by typing the `tar -xvfz aocl-rte-<version>.arm32.tgz` command.
2. Navigate to the INTELFPGAOCLSDKROOT/board/c5soc/driver subdirectory of the unpacked aclrte-arm32 package.
3. Perform the tasks outlined in the Compiling the Linux Kernel for Cyclone V SoC section of the Intel FPGA SDK for OpenCL Cyclone V SoC Development Kit Reference Platform Porting Guide.

**Related Information**
Compiling the Linux Kernel for Cyclone V SoC FPGA

4.1.5. Installing the Intel FPGA RTE for OpenCL Standard Edition onto the SoC FPGA Board

The RTE Standard Edition installation package for Intel SoC FPGAs with 32-bit ARM processor is available in tar format. To install the software, you must install it in a directory that you own, and set all the necessary environment variables.

1. Create an RTE directory on the board’s file system by typing the `mkdir <rte_destination_directory>` command.
2. Move the downloaded installation package aclrte-arm32.tgz to the RTE directory by typing the `mv aclrte-arm32.tgz <rte_destination_directory>` command.
3. Type `cd <rte_destination_directory>` to navigate to the RTE directory.
4. To unpack the tarball, type `tar -xvfz aclrte-arm32.tgz` at the command prompt.
5. Transfer the aclsoc_drv.ko file you built on your development machine into the `<rte_destination_directory>/board/c5soc/driver` directory on the SoC FPGA board.
6. Set the environment variables, as shown below.
Intel recommends that you consolidate the settings of the environment variables into a file called `init_opencl.sh`. Then, run the command `source ./init_opencl.sh` to load all the environment variables and the OpenCL Linux kernel driver simultaneously.

```bash
export INTELFPGAOCLESRTROOT=<rte_destination_directory>
export AOCL_BOARD_PACKAGE_ROOT=$INTELFPGAOCLESRTROOT/board/c5soc
export PATH=$INTELFPGAOCLESRTROOT/bin:$PATH
export LD_LIBRARY_PATH=$INTELFPGAOCLESRTROOT/host/arm32/lib:$LD_LIBRARY_PATH
insmod $AOCL_BOARD_PACKAGE_ROOT/driver/aclsoc_drv.ko
```

### 4.1.6. Installing the Cyclone V SoC Development Kit

To execute an OpenCL kernel on a Cyclone V SoC FPGA, first install the Cyclone V SoC Development Kit and then apply the Intel FPGA SDK for OpenCL-specific configurations.

For the key components of a Cyclone V SoC Development Kit, refer to [Getting Started with the Intel FPGA SDK for OpenCL Standard Edition for Intel ARMv7-A SoC FPGA](#).

1. [Writing an SD Card Image onto the Micro SD Flash Card](#) on page 36
2. [Configuring the SW3 Switches](#) on page 37
3. [Setting Up Terminal Connection](#) on page 38
4. [Setting Environment Variables and Loading OpenCL Linux Kernel Driver](#) on page 39
5. [Connecting the Board to Network via Ethernet](#) on page 39

#### Related Information

- [Getting Started with the Intel FPGA RTE for OpenCL Standard Edition for Intel ARMv7-A SoC FPGA](#)

### 4.1.6.1. Writing an SD Card Image onto the Micro SD Flash Card

To write an Intel FPGA SDK for OpenCL-compatible SD card image onto the micro SD flash card on Windows, download and install the Win32 Disk Imager, and then write the SD card image onto the micro SD flash card. The SD card image contains everything you need to start using OpenCL on the board.

The SD card image `linux_sd_card_image.tgz` is included in the Cyclone V SoC Development Kit Reference Platform, available with the SDK.

You must have administrator privileges.

1. Extract the files from the `%INTELFPGAOCLESRTROOT%\board\c5soc\linux_sd_card_image.tgz` archive.
   You can use tools such as 7zip or WinZip to extract the SD card image file from the .tgz archive.

2. Download the Win32 Disk Imager from the [SourceForge](#) website.
3. Unzip the Win32 Disk Imager and the SD card image to a directory that you own.
4. Insert the micro SD card into the card reader and connect it to your PC.
5. Launch the Win32 Disk Imager. In the dialog box, under **Image File**, browse to the SD card image file.

6. From the **Device** pull-down menu, select the destination drive of the micro SD card.

   **Warning:** Specifying the wrong device name might cause the SD card image to overwrite all existing data.

7. Click **Write**.

8. After you write the image onto the micro SD flash card, insert the card into the micro SD card slot on the Cyclone V SoC Development Kit.

9. Power up the board.

   If the LEDs on the FPGA flash in a counter pattern, the image is written onto the micro SD card successfully. A section of OpenCL logic on the FPGA drives these LEDs.

### 4.1.6.2. Configuring the SW3 Switches

Configure the SW3 dual in-line package (DIP) switches on the Cyclone V SoC Development Kit for use with the Intel FPGA SDK for OpenCL. The switch bank is located next to the micro SD card slot.
- Set the SW3 DIP switches to the following positions:

<table>
<thead>
<tr>
<th>Switch</th>
<th>Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ON</td>
</tr>
<tr>
<td>2</td>
<td>OFF</td>
</tr>
<tr>
<td>3</td>
<td>ON</td>
</tr>
<tr>
<td>4</td>
<td>OFF</td>
</tr>
<tr>
<td>5</td>
<td>ON</td>
</tr>
<tr>
<td>6</td>
<td>ON</td>
</tr>
</tbody>
</table>

The figure below illustrates the physical configuration of the SW switches on the Cyclone V SoC Development Kit:

![SW Switch Configuration](image)

### 4.1.6.3. Setting Up Terminal Connection

To set up the terminal connection on the Cyclone V SoC Development Kit for use with the Windows version of the Intel FPGA SDK for OpenCL, specify the USB virtual COM port settings.

1. Connect the board to your development machine via the micro-USB port that is closest to the power supply connector on the board.
2. Connect the board to the power supply and power it up.
3. Download the Virtual COM port (VCP) driver from the VCP driver download page on the Future Technology Devices International (FTDI) Ltd. website.
4. Determine the COM port in use.
   a. From the Windows Start menu, click Control Panel ➤ Hardware and Sound.
   b. Under Devices and Printers, click Device Manager.
   c. In the Device Manager window, under Ports, click USB Serial Port (COM<X>).
5. Connect either the Tera Term or PuTTY open-source terminal emulator to the COM port that the FDTI driver creates.
6. Set the port settings to 115200, 8N1, with parity and control flow set to none.
7. For Tera Term, select **Setup ➤ Terminal**, and then change Code Page to 1250.
8. Without powering down, restart the board.

### 4.1.6.4. Setting Environment Variables and Loading OpenCL Linux Kernel Driver

After you turn on the board and establish terminal connection, log into the Cyclone V SoC Development Kit as user **root** with no password. Then, before you run your host application, set the environment variables and load the OpenCL Linux kernel driver.

- After logging into the SoC FPGA board, run the `source ./init_opencl.sh` command, which performs the following tasks:
  - Set the `PATH`, `LD_LIBRARY_PATH`, and `AOCL_BOARD_PACKAGE_ROOT` environment variables.
  - Load the OpenCL Linux kernel driver.

The `init_opencl.sh` file is available in the SD card image that you write onto the micro SD flash card. It contains the commands shown below:

```bash
export INTELFPGAOCLSDKROOT=<aocl_destination_directory>
export AOCL_BOARD_PACKAGE_ROOT=$INTELFPGAOCLSDKROOT/board/c5soc
export PATH=$INTELFPGAOCLSDKROOT/bin:$PATH
export LD_LIBRARY_PATH=$INTELFPGAOCLSDKROOT/host/arm32/lib:
  $LD_LIBRARY_PATH
insmod $AOCL_BOARD_PACKAGE_ROOT/driver/aclsoc_drv.ko
```

### 4.1.6.5. Connecting the Board to Network via Ethernet

Connecting the Cyclone V SoC Development Kit to the host network allows you to transfer files to and from your SoC FPGA.

1. Connect the HPS Ethernet port of the board to your network.
2. Reboot the board.

The boards acquires an IP address, allowing you to initiate a Secure Shell (SSH) connection and runs a Secure Copy (SCP) program to login and transfer files.

Alternatively, you can mount a network drive to your board and access the files directly.

#### 4.1.6.5.1. Ensuring IP Address Acquisition

After you connect the HPS Ethernet port on the Cyclone V SoC Development Kit to your network and reboot the board, ensure that the board acquires an IP address successfully.

After you connect the HPS Ethernet port to your network and power up your board, you should see a solid orange light and a blinking green light. If not, check the connection of the Ethernet cable to the Ethernet port on your network.

1. To check if your board has an IP address, search for the IP address in boot messages such as the one shown below:

```plaintext
Sending discover...
libphy: stmmac-0:04 - Link is Up - 1000/Full
Sending discover...
```
The message **Lease of <board_IP_address> obtained, lease time 86400** identifies the IP address of the board.

2. If you receive the following output, perform a warm reboot of the board by pressing the WARM button next to the LED lights.

   Sending discover...
   libphy: stmmac-0:04 - Link is Up - 1000/Full
   Sending discover...
   Sending discover...
   No lease, failing

The board uses the dynamic host configuration protocol (DHCP) to acquire an IP address. If the session times out waiting for an IP assignment, reboot the CPU to restart the IP acquisition process. To reboot the CPU, press the Warm reset button next to the four HPS LEDs on the board.

3. If you are unable to acquire the IP address, ensure that the Ethernet cable is in good working condition and the Ethernet port on your network is enabled.

4.1.6.5.2. Using SSH and SCP

Instead of connecting the Cyclone V SoC Development Kit to the host system using UART over USB and transferring files using a network drive, you can initiate a Secure Shell connection and transfer files using a Secure Copy program.

1. To establish a connection between the Cyclone V SoC Development Kit and the host system via SSH, invoke the `ssh root@<board_ip_address>` command from your development machine.

   For instructions on how to identify `<board_ip_address>`, refer to the **Ensuring IP Address Acquisition** section.

2. To transfer files, one at a time, from the host system to the board via SCP, invoke the `scp <source_filename> root@<board_ip_address>:<target_filename>` command from your development machine.

4.1.7. Executing an OpenCL Kernel on an SoC FPGA

The procedures outlined in this document are for building and running the host application for the hello_world example design. To execute the hello_world OpenCL kernel on your SoC FPGA, you must first create an `hello_world.aocx` file. For instructions on obtaining the hello_world example design and creating the `hello_world.aocx` file, refer to the **Creating the FPGA Hardware Configuration File of an OpenCL Kernel** section of the Intel FPGA SDK for OpenCL Standard Edition Cyclone V SoC Getting Started Guide.

Build your host application using the GCC cross-compiler available with the SoC EDS.
4.1.7.1. Building the Host Application

Build your SoC FPGA-specific OpenCL host application using the GCC cross-compiler available with the Windows version of the SoC EDS.

1. Perform the following tasks to download the hello_world design example.
   a. Download the SoC FPGA-specific hello_world design example (<version>Arm32 Linux package (.tgz)) from the Hello World Design Example page.
   b. Extract exm_opencl_hello_world_arm32_linux_<version>.tar to a location to which you have write access.
      Important: Ensure that the location name does not contain spaces.
   c. Verify that the AOCL_BOARD_PACKAGE_ROOT environment variable setting points to the Cyclone V SoC Development Kit Reference Platform. Open a Windows command window and then type echo %AOCL_BOARD_PACKAGE_ROOT% at the command prompt.
      If the returned path is not %INTELFPGAOCLSDKROOT%\board\c5soc, or if AOCL_BOARD_PACKAGE_ROOT is not set, modify the environment variable setting.

2. At a command prompt, invoke the following command to set the PATH environment variable:
   SET PATH=%PATH%;<path_to_SoCEDS_installation_dir>\ds-5\sw\gcc\bin

3. Navigate to the <path_to_exm_opencl_hello_world_arm32_linux_<version>>\hello_world folder.

4. Invoke the make -f Makefile command. Alternatively, you can simply invoke the make command.
   The hello_world executable will be in the <path_to_exm_opencl_hello_world_arm32_linux_<version>>\hello_world\bin folder.

4.1.7.2. Running the Host Application

For Windows systems, execute the hello_world.aocx executable file on the SoC FPGA by running the host application you built from the ARM-specific Makefile.

1. Log into your SoC FPGA board.

2. Copy the hello_world.aocx hardware configuration file and the hello_world host executable from their current folders to the board.

3. Verify that the LD_LIBRARY_PATH environment variable setting includes %INTELFPGAOCLSDKROOT%\host\arm32\lib. Run the command echo $LD_LIBRARY_PATH.
   If you ran the init_opencl.sh script, the LD_LIBRARY_PATH setting should point to %INTELFPGAOCLSDKROOT%\host\arm32\lib.

4. To execute the kernel on the SoC FPGA, at a command prompt, navigate to the host executable folder and run the hello_world host executable.
4.1.7.3. Output from Successful Kernel Execution on the Cyclone V SoC Development Kit

When you run the host application to execute your OpenCL kernel on the Cyclone V SoC Development Kit, the software notifies you of a successful kernel execution.

Example output:

```
Reprogramming device [0] with handle 1
Querying platform for info:
--------------------------
CL_PLATFORM_NAME = Intel(R) FPGA SDK for OpenCL(TM)
CL_PLATFORM_VENDOR = Intel Corporation
CL_PLATFORM_VERSION = OpenCL 1.0 Intel(R) FPGA SDK for OpenCL(TM), <version>

Querying device for info:
------------------------
CL_DEVICE_NAME = c5soc : Cyclone V SoC Development Kit
CL_DEVICE_VENDOR = Intel(R) Corporation
CL_DEVICE_VENDOR_ID = 4466
CL_DEVICE_VERSION = OpenCL 1.0 Intel(R) FPGA SDK for OpenCL(TM), <version>
CL_DRIVER_VERSION = <version>
CL_DEVICE_ADDRESS_BITS = 64
CL_DEVICE_AVAILABLE = true
CL_DEVICE_ENDIAN_LITTLE = true
CL_DEVICE_GLOBAL_MEM_CACHE_SIZE = 32768
CL_DEVICE_GLOBAL_MEM_CACHELINE_SIZE = 0
CL_DEVICE_GLOBAL_MEM_SIZE = 2147483648
CL_DEVICE_IMAGE_SUPPORT = false
CL_DEVICE_LOCAL_MEM_SIZE = 16384
CL_DEVICE_MAX_CLOCK_FREQUENCY = 1000
CL_DEVICE_MAX_COMPUTE_UNITS = 1
CL_DEVICE_MAX_CONSTANT_ARGS = 8
CL_DEVICE_MAX_CONSTANT_BUFFER_SIZE = 3758096384
CL_DEVICE_MAX_WORK_ITEM_DIMENSIONS = 3
CL_DEVICE_MAX_WORK_ITEM_DIMENSIONS = 1024
CL_DEVICE_MIN_DATA_TYPE_ALIGN_SIZE = 128
CL_DEVICE_PREFERRED_VECTOR_WIDTH_CHAR = 4
CL DEVICE_PREFERRED_VECTOR_WIDTH_SHORT = 2
CL_DEVICE_PREFERRED_VECTOR_WIDTH_INT = 1
CL_DEVICE_PREFERRED_VECTOR_WIDTH_LONG = 1
CL DEVICE_PREFERRED_VECTOR_WIDTH_FLOAT = 1
CL_DEVICE_PREFERRED_VECTOR_WIDTH_DOUBLE = 0
Command queue out of order? = false
Command queue profiling enabled? = true

Kernel initialization is complete.
Launching the kernel...

Thread #2: Hello from the Intel(R) FPGA SDK for OpenCL(TM) Compiler!

Kernel execution is complete.
```
unset INTELFPGAOCLSDKROOT
unset PATH
unset LD_LIBRARY_PATH

4. Uninstall the Intel FPGA SDK for OpenCL Standard Edition on your host system and unset the corresponding environment variables.

**Related Information**

Uninstalling the Intel FPGA SDK for OpenCL

---

**4.2. Getting Started with the Intel FPGA RTE for OpenCL Standard Edition for SoC FPGA on Linux**


2. Installing the Intel FPGA SDK for OpenCL Standard Edition for SoC FPGA on page 44
3. Installing the Intel SoC FPGA Embedded Development Suite Standard Edition on page 45
4. Recompiling the Linux Kernel Driver on page 45
5. Installing the Intel FPGA RTE for OpenCL Standard Edition onto the SoC FPGA Board on page 46
6. Installing the Cyclone V SoC Development Kit on page 47
7. Executing an OpenCL Kernel on an SoC FPGA on page 51
8. Uninstalling the Intel FPGA RTE for OpenCL Standard Edition on page 53

---

**4.2.1. Downloading the Intel FPGA SDK for OpenCL Standard Edition and the SoC EDS Standard Edition**


The SDK includes the SD card image you need to recompile the OpenCL Linux kernel driver. If you wish to recompile the Linux kernel driver and write the SD card image on your own, download the RTE Standard Edition for SoC instead.
The Intel FPGA SDK for OpenCL Cyclone V SoC Development Kit Reference Platform includes an SD flash card image necessary for running OpenCL applications on the board. The SD flash card image includes the recompiled Linux kernel driver, a preinstalled version of the Intel FPGA RTE for OpenCL Standard Edition, and a script for setting environment variables.

To get started with the RTE on the Cyclone V SoC Development Kit using the SD flash card image that comes with the SDK, install the SDK Standard Edition for Linux. If you want to create your own SD card image, install the RTE Standard Edition.

You must have sudo or root privileges.
To install the SDK, perform the following tasks:

a. Unpack the downloaded tar file.

b. Run the `setup.sh` file to install the SDK and device support.

To install the RTE, unpack the `.tgz` file install the RTE in a directory that you own.

Note: The installer sets the environment variable `INTELFPGAOCLSDKROOT` to point to the path of the software installation.

Verify that the installer sets the user environment variable `INTELFPGAOCLSDKROOT` to point to the current version of the software. Open a Windows command window and then type `echo $INTELFPGAOCLSDKROOT` at the command prompt.

If the returned path does not point to the location of the current SDK installation, or if the path is not set, modify the `INTELFPGAOCLSDKROOT` setting.

**Related Information**

Setting the Intel FPGA RTE for OpenCL Standard Edition User Environment Variables on page 23

### 4.2.3. Installing the Intel SoC FPGA Embedded Development Suite Standard Edition

Install the Intel SoC EDS Standard Edition for Linux to build your host application for OpenCL kernel deployment on an SoC FPGA board.

The GCC tool chain is part of the SoC EDS installation package.

1. Run the `SoCEDSSetup--<version>--linux.run` installer. For more information, refer to the Installing the SoC EDS section of the Intel SoC FPGA Embedded Development Suite User Guide.


For more information on the Arm DS-5 Intel SoC FPGA Edition Toolkit, refer to the Arm DS-5 Intel SoC FPGA Edition page of the ARM website.


**Related Information**

- Installing the ARM DS-5 Intel SoC FPGA Edition Toolkit
- SoC FPGA EDS Licensing
- Installing the SoC FPGA EDS

### 4.2.4. Recompiling the Linux Kernel Driver

**Attention:** If you download and install the Intel FPGA SDK for OpenCL Standard Edition, the Cyclone V SoC Development Kit Reference Platform (`INTELFPGAOCLSDKROOT/board/c5soc`) includes an SD card image (`linux_sd_card_image.tgz`) that contains the recompiled Linux kernel driver.
If you need to rebuild the Linux kernel driver, recompile the aclsoc Linux kernel driver to the exact version of the Linux kernel running on the SoC FPGA.

Important: You must recompile the aclsoc Linux kernel driver on your Linux development machine.

1. Unpack the aocl-rte-<version>.arm32.tgz tarball to a temporary directory on your development machine by typing the tar -xvfz aocl-rte-<version>.arm32.tgz command.

2. Navigate to the INTELFPGAOCLSDKROOT/board/c5soc/driver subdirectory of the unpacked aclrte-arm32 package.

3. Perform the tasks outlined in the Compiling the Linux Kernel for Cyclone V SoC section of the Intel FPGA SDK for OpenCL Cyclone V SoC Development Kit Reference Platform Porting Guide.

Related Information
Compiling the Linux Kernel for Cyclone V SoC FPGA

4.2.5. Installing the Intel FPGA RTE for OpenCL Standard Edition onto the SoC FPGA Board

The RTE Standard Edition installation package for Intel SoC FPGAs with 32-bit ARM processor is available in tar format. To install the software, you must install it in a directory that you own, and set all the necessary environment variables.

1. Create an RTE directory on the board's file system by typing the mkdir <rte_destination_directory> command.

2. Move the downloaded installation package aclrte-arm32.tgz to the RTE directory by typing the mv aclrte-arm32.tgz <rte_destination_directory> command.

3. Type cd <rte_destination_directory> to navigate to the RTE directory.

4. To unpack the tarball, type tar -xvfz aclrte-arm32.tgz at the command prompt.

5. Transfer the aclsoc_drv.ko file you built on your development machine into the <rte_destination_directory>/board/c5soc/driver directory on the SoC FPGA board.

6. Set the environment variables, as shown below.

Intel recommends that you consolidate the settings of the environment variables into a file called init_opencl.sh. Then, run the command source ./init_opencl.sh to load all the environment variables and the OpenCL Linux kernel driver simultaneously.

```bash
export INTELFPGAOCLSDKROOT=<rte_destination_directory>
export AOCL_BOARD_PACKAGE_ROOT=$INTELFPGAOCLSDKROOT/board/c5soc
export PATH=$INTELFPGAOCLSDKROOT/bin:$PATH
export LD_LIBRARY_PATH=$INTELFPGAOCLSDKROOT/host/arm32/lib:$LD_LIBRARY_PATH
insmod $AOCL_BOARD_PACKAGE_ROOT/driver/aclsoc_drv.ko
```
4.2.6. Installing the Cyclone V SoC Development Kit

To execute an OpenCL kernel on a Cyclone V SoC FPGA, first install the Cyclone V SoC Development Kit and then apply the Intel FPGA SDK for OpenCL-specific configurations.

For the key components of a Cyclone V SoC Development Kit, refer to *Getting Started with the Intel FPGA SDK for OpenCL Standard Edition for Intel ARMv7-A SoC FPGA*.

4.2.6.1. Writing an SD Card Image onto the Micro SD Flash Card

To write an Intel FPGA SDK for OpenCL-compatible SD card image onto the micro SD flash card on Linux, extract the SD card image from the Cyclone V SoC Development Kit Reference Platform, and then write the image onto the micro SD flash card. The SD card image contains everything you need to start using OpenCL on the board.

The SD card image `linux_sd_card_image.tgz` is included in the Cyclone V SoC Development Kit Reference Platform, available with the SDK. Ensure that the environment variable `AOCL_BOARD_PACKAGE_ROOT` points to the location of the `board_env.xml` file in the Reference Platform.

You must have sudo or root privileges.

1. To decompress the $INTEL_FPGA_OCL_SDK_ROOT/board/c5soc/
linux_sd_card_image.tgz file, run the `tar xvfz` 
linux_sd_card_image.tgz command.

2. Insert the micro SD flash card into a card reader, and connect the reader to your PC.
   a. If the flash card already contains an image, partitions will exist automatically in the micro SD card. Unmount or eject all these partitions.

3. Run the `dmesg | tail` command to verify the device name of the flash card (for example, `/dev/sde`).

4. Write the SD card image onto the micro SD flash card by running the following commands:
   
   sudo dd if=linux_sd_card_image of=/dev/sde bs=1M
   sync
   
   **Attention:** If the device name of your micro SD flash card is not `/dev/sde,
   replace `/dev/sde` in the above command with the device name you obtain from Step 3.

   **Warning:** Specifying the wrong device name might cause the SD card image to overwrite all existing data.

5. After you write the image onto the micro SD flash card, insert the card into the micro SD card slot on the Cyclone V SoC Development Kit.

4.2.6.2. Configuring the SW3 Switches

Configure the SW3 dual in-line package (DIP) switches on the Cyclone V SoC Development Kit for use with the Intel FPGA SDK for OpenCL. The switch bank is located next to the micro SD card slot.
- Set the SW3 DIP switches to the following positions:

<table>
<thead>
<tr>
<th>Switch</th>
<th>Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ON</td>
</tr>
<tr>
<td>2</td>
<td>OFF</td>
</tr>
<tr>
<td>3</td>
<td>ON</td>
</tr>
<tr>
<td>4</td>
<td>OFF</td>
</tr>
<tr>
<td>5</td>
<td>ON</td>
</tr>
<tr>
<td>6</td>
<td>ON</td>
</tr>
</tbody>
</table>

The figure below illustrates the physical configuration of the SW switches on the Cyclone V SoC Development Kit:

![Cyclone V SoC Development Kit SW Switches](image)

### 4.2.6.3. Setting Up Terminal Connection

To set up the terminal connection on the Cyclone V SoC Development Kit for use with the Linux version of the Intel FPGA SDK for OpenCL, specify the USB virtual COM port settings.

1. Connect the board to your development machine via the micro-USB port that is closest to the power supply connector on the board.
2. Connect the board to the power supply and power it up.
3. Run the command `dmesg | tail` to determine which device the Future Technology Devices International (FTDI) driver assigns for the connection (e.g. `/dev/ttyUSB0`).
4. Setup the minicom as follows:
   a. Ensure that minicom is installed on your system. If not, invoke the `yum install minicom` command.
   b. Run `minicom -s` as root to enter the minicom setup mode.
   c. Select **Serial port setup** and then press Enter.
d. Press A to change **Serial Device** to `/dev/ttyUSB0` and then press Enter.

e. Press E to change the port settings. Press E again to select **115200** for **Speed**, and then press Q to set **Data/Parity/Stopbits** to **8-N-1** configuration.

f. Press Enter twice to return to the main minicom setup menu.

g. Select **Save setup as dfl** and then press Enter to save the minicom settings as defaults.

h. Select **Exit**.

5. Without powering down, restart the board. You should see Linux boot messages appear on the terminal command of your choice.

**4.2.6.4. Setting Environment Variables and Loading OpenCL Linux Kernel Driver**

After you turn on the board and establish terminal connection, log into the Cyclone V SoC Development Kit as user **root** with no password. Then, before you run your host application, set the environment variables and load the OpenCL Linux kernel driver.

- After logging into the SoC FPGA board, run the `source ./init_opencl.sh` command, which performs the following tasks:
  
  a. Set the `PATH`, `LD_LIBRARY_PATH`, and `AOCL_BOARD_PACKAGE_ROOT` environment variables.
  
  b. Load the OpenCL Linux kernel driver.

The `init_opencl.sh` file is available in the SD card image that you write onto the micro SD flash card. It contains the commands shown below:

```bash
export INTELFPGAOCPLSDKROOT=<aocl_destination_directory>
export AOCL_BOARD_PACKAGE_ROOT=$INTELFPGAOCPLSDKROOT/board/c5soc
export PATH=$INTELFPGAOCPLSDKROOT/bin:$PATH
export LD_LIBRARY_PATH=$INTELFPGAOCPLSDKROOT/host/arm32/lib:
$LD_LIBRARY_PATH
insmod $AOCL_BOARD_PACKAGE_ROOT/driver/aclsoc_drv.ko
```

**4.2.6.5. Connecting the Board to Network via Ethernet**

Connecting the Cyclone V SoC Development Kit to the host network allows you to transfer files to and from your SoC FPGA.

1. Connect the HPS Ethernet port of the board to your network.

2. Reboot the board.

   The boards acquires an IP address, allowing you to initiate a Secure Shell (SSH) connection and runs a Secure Copy (SCP) program to login and transfer files.

   Alternatively, you can mount a network drive to your board and access the files directly.

**4.2.6.5.1. Ensuring IP Address Acquisition**

After you connect the HPS Ethernet port on the Cyclone V SoC Development Kit to your network and reboot the board, ensure that the board acquires an IP address successfully.
After you connect the HPS Ethernet port to your network and power up your board, you should see a solid orange light and a blinking green light. If not, check the connection of the Ethernet cable to the Ethernet port on your network.

1. To check if your board has an IP address, search for the IP address in boot messages such as the one shown below:

```
Sending discover...
libphy: stmmac-0:04 - Link is Up - 1000/Full
Sending discover...
Sending select for 137.57.175.148...
Lease of 137.57.175.148 obtained, lease time 86400
/etc/udhcpc.d/50default: Adding DNS 137.57.142.218
/etc/udhcpc.d/50default: Adding DNS 137.57.109.10
/etc/udhcpc.d/50default: Adding DNS 137.57.64.1
done.
```

The message `Lease of <board_IP_address> obtained, lease time 86400` identifies the IP address of the board.

2. If you receive the following output, perform a warm reboot of the board by pressing the WARM button next to the LED lights.

```
Sending discover...
libphy: stmmac-0:04 - Link is Up - 1000/Full
Sending discover...
Sending discover...
No lease, failing
```

The board uses the dynamic host configuration protocol (DHCP) to acquire an IP address. If the session times out waiting for an IP assignment, reboot the CPU to restart the IP acquisition process. To reboot the CPU, press the Warm reset button next to the four HPS LEDs on the board.

3. If you are unable to acquire the IP address, ensure that the Ethernet cable is in good working condition and the Ethernet port on your network is enabled.

### 4.2.6.5.2. Mounting a Shared Drive

The most convenient way to share files between a development PC and the Cyclone V SoC FPGA development board is to mount a network drive.

1. Check the `/etc/fstab` file systems table file on your development PC for the line that describes the mounting of the drive you want to use on the board.

   The following example `/etc/fstab` entry indicates that the `/data` folder on the my_nas server is mounted to the `/data` folder on the development PC:

   ```
   my_nas:/data /data nfs
   exec,dev,suid,rw,tcp,hard,intr,vers=3,rsize=32768,wsize=32768,
timeo=600,retrans=200
   ```

2. Add the `/etc/fstab` entry described above to the `/etc/fstab` file on the Cyclone V SoC development board.

3. Run the `sync` command to save the `/etc/fstab` file to the micro SD flash card.

4. Create an empty folder on the board that serves as the mounting point for the network drive.

   For example: `type mkdir /data`, where `/data` is the name of the folder.

5. Invoke the `busybox mount -a` command.
If the mounting operation fails, rerun the command.

### 4.2.6.5.3. Using SSH and SCP

Instead of connecting the Cyclone V SoC Development Kit to the host system using UART over USB and transferring files using a network drive, you can initiate a Secure Shell connection and transfer files using a Secure Copy program.

1. To establish a connection between the Cyclone V SoC Development Kit and the host system via SSH, invoke the `ssh root@<board_ip_address>` command from your development machine.
   
   For instructions on how to identify `<board_ip_address>`, refer to the *Ensuring IP Address Acquisition* section.

2. To transfer files, one at a time, from the host system to the board via SCP, invoke the `scp <source_filename> root@<board_ip_address>:<target_filename>` command from your development machine.

### 4.2.7. Executing an OpenCL Kernel on an SoC FPGA

The procedures outlined in this document are for building and running the host application for the hello_world example design. To execute the hello_world OpenCL kernel on your SoC FPGA, you must first create an `hello_world.aocx` file. For instructions on obtaining the hello_world example design and creating the `hello_world.aocx` file, refer to the *Creating the FPGA Hardware Configuration File of an OpenCL Kernel* section of the *Intel FPGA SDK for OpenCL Standard Edition Cyclone V SoC Getting Started Guide*.

Build your host application using the GCC cross-compiler available with the SoC EDS.

**Related Information**

- Creating the Hardware Configuration File of an OpenCL Kernel for SoC FPGA
- OpenCL Design Examples

### 4.2.7.1. Building the Host Application

Build your SoC FPGA-specific OpenCL host application using the GCC cross-compiler available with the Linux version of the SoC EDS.

- To build your host application for emulation, modify the `AOCL_BOARD_PACKAGE_ROOT` environment variable setting to point to a non-SoC Reference or Custom Platform. Verify the setting by opening a shell and then typing `echo $AOCL_BOARD_PACKAGE_ROOT` at the command prompt.

- To build your host application for kernel execution, verify that the `AOCL_BOARD_PACKAGE_ROOT` environment variable setting points to the Cyclone V SoC Development Kit Reference Platform.

1. At a command prompt, invoke the following command to set the `PATH` environment variable:
export PATH=<path_to_SoCEDS_installation_dir>/ds-5/sw/gcc/bin: $PATH

2. Navigate to the
<path_to_exm_opencl_hello_world_arm32_linux_<version>>/hello_world directory.

3. Invoke the make -f Makefile command. Alternatively, you can simply invoke
the make command. The hello_world executable will be in the
<path_to_exm_opencl_hello_world_arm32_linux_<version>>/hello_world/bin directory.

4.2.7.2. Running the Host Application

For Linux systems, execute the hello_world.aocx file on the SoC FPGA by running
the host application you built from the ARM-specific Makefile.

1. Log into your SoC FPGA board.

2. Copy the hello_world.aocx hardware configuration file and the hello_world
host executable from their current directories to the board.

3. Verify that the LD_LIBRARY_PATH environment variable setting includes
$INTELFPGAOCLSDKROOT/host/arm32/lib. Run the command echo
$LD_LIBRARY_PATH.
If you ran the init_opencl.sh script, the LD_LIBRARY_PATH setting should
point to $INTELFPGAOCLSDKROOT/host/arm32/lib.

4. To execute the kernel on the SoC FPGA, at a command prompt, navigate to the
host executable directory and run the hello_world host executable.

4.2.7.3. Output from Successful Kernel Execution on the Cyclone V SoC
Development Kit

When you run the host application to execute your OpenCL kernel on the Cyclone V
SoC Development Kit, the software notifies you of a successful kernel execution.

Example output:

```
Reprogramming device [0] with handle 1
Querying platform for info:
--------------------------
CL_PLATFORM_NAME = Intel(R) FPGA SDK for OpenCL(TM)
CL_PLATFORM_VENDOR = Intel Corporation
CL_PLATFORM_VERSION = OpenCL 1.0 Intel(R) FPGA SDK for OpenCL(TM), <version>
--------------------------
Querying device for info:
--------------------------
CL_DEVICE_NAME = c5soc : Cyclone V SoC Development Kit
CL_DEVICE_VENDOR = Intel(R) Corporation
CL_DEVICE_VENDOR_ID = 4466
CL_DEVICE_VERSION = OpenCL 1.0 Intel(R) FPGA SDK for OpenCL(TM), <version>
CL_DRIVER_VERSION = <version>
CL_DRIVER_ADDRESS_BITS = 64
CL_DRIVER_AVAILABLE = true
CL_DRIVER_ENDIAN_LITTLE = true
CL_DEVICE_GLOBAL_MEM_CACHE_SIZE = 32768
CL_DEVICE_GLOBAL_MEM_CACHELINE_SIZE = 0
CL_DEVICE_GLOBAL_MEM_SIZE = 2147483648
CL_DEVICE_IMAGE_SUPPORT = false
CL_DEVICE_LOCAL_MEM_SIZE = 16384
CL_DEVICE_MAX_CLOCK_FREQUENCY = 1000
```
4. Getting Started with the Intel FPGA RTE for OpenCL Standard Edition for Intel ARMv7-A SoC FPGA

UG-20149 | 2018.09.24

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CL_DEVICE_MAX_COMPUTE_UNITS</td>
<td>1</td>
</tr>
<tr>
<td>CL_DEVICE_MAX_CONSTANT_ARGS</td>
<td>8</td>
</tr>
<tr>
<td>CL_DEVICE_MAX_CONSTANT_BUFFER_SIZE</td>
<td>3758096384</td>
</tr>
<tr>
<td>CL_DEVICE_MAX_WORK_ITEM_DIMENSIONS</td>
<td>3</td>
</tr>
<tr>
<td>CL_DEVICE_MAX_WORK_ITEM_DIMENSIONS</td>
<td>1024</td>
</tr>
<tr>
<td>CL_DEVICE_MIN_DATA_TYPE_ALIGN_SIZE</td>
<td>128</td>
</tr>
<tr>
<td>CL_DEVICE_PREFERRED_VECTOR_WIDTH_CHAR</td>
<td>4</td>
</tr>
<tr>
<td>CL_DEVICE_PREFERRED_VECTOR_WIDTH_SHORT</td>
<td>2</td>
</tr>
<tr>
<td>CL_DEVICE_PREFERRED_VECTOR_WIDTH_INT</td>
<td>1</td>
</tr>
<tr>
<td>CL_DEVICE_PREFERRED_VECTOR_WIDTH_LONG</td>
<td>1</td>
</tr>
<tr>
<td>CL_DEVICE_PREFERRED_VECTOR_WIDTH_FLOAT</td>
<td>1</td>
</tr>
<tr>
<td>CL_DEVICE_PREFERRED_VECTOR_WIDTH_DOUBLE</td>
<td>0</td>
</tr>
</tbody>
</table>

Command queue out of order? = false
Command queue profiling enabled? = true

Kernel initialization is complete.
Launching the kernel...

Thread #2: Hello from the Intel(R) FPGA SDK for OpenCL(TM) Compiler!

Kernel execution is complete.

4.2.8. Uninstalling the Intel FPGA RTE for OpenCL Standard Edition

To uninstall the RTE from the SoC FPGA board, delete the RTE directory and restore all modified environment variables to their previous settings.

1. Navigate to the root directory in the SoC FPGA board’s file system that contains the `<rte_destination_directory>` directory.
2. Type `rm -rf <rte_destination_directory>` to remove the RTE directory.
3. Remove the environment variable settings by typing the following commands:
   
   ```
   unset AOCL_BOARD_PACKAGE_ROOT
   unset INTELFPGAOCLSDKROOT
   unset PATH
   unset LD_LIBRARY_PATH
   ```

4. Uninstall the Intel FPGA SDK for OpenCL on your host system and unset the corresponding environment variables.

Related Information

Uninstalling the Intel FPGA SDK for OpenCL

<table>
<thead>
<tr>
<th>Date</th>
<th>Version</th>
<th>Changes</th>
</tr>
</thead>
</table>
| November 2017  | 2017.11.04  | • Rebranded the following:<br>  — ALTERAOCLSDKROOT to INTELFPGAOCLSDKROOT  <br>  — CL_CONTEXT_EMULATOR_DEVICE_ALTERA to CL_CONTEXT_EMULATORDEVICE_INTELFPGA  
|                |             | • In Prerequisites of the Intel FPGA RTE for OpenCL, changed Microsoft Visual Studio version 2010 Professional as Microsoft Visual Studio Professional version 2010 or later.  
|                |             | • In Building the Host Application (Windows), updated references to Microsoft Visual Studio 2015 as Microsoft Visual Studio.  
| May 2017       | 2017.05.05  | • Rebranded the Altera Client Driver (ACD) to FPGA Client Driver (FCD).  
|                |             | • Updated the download instructions in Downloading the Intel FPGA RTE for OpenCL for Windows and Linux.  
|                |             | • Added reminders that folder names where you uncompress downloaded OpenCL design examples must not contain spaces.  
| October 2016   | 2016.10.31  | • Rebranded Altera RTE for OpenCL to Intel FPGA RTE for OpenCL.  
|                |             | • Rebranded Altera SoC Embedded Design Suite to Intel SoC FPGA Embedded Design Suite.  
|                |             | • Deprecated and removed support for big-endian system, resulting in the following documentation changes:  
|                |             |  — In Prerequisites for the Altera RTE for OpenCL, removed "Red Hat Enterprise Linux 6 on big-endian system" from the list of supported operating systems.  
|                |             |  — In Contents of the Altera RTE for OpenCL, removed the Big-Endian System Directory column from the table of contents in the RTE installation directory.  
|                |             |  — In Downloading the Altera RTE for OpenCL for Linux, removed from Step 6 the choice to select the OpenCL PowerPC RPM installation package.  
|                |             |  — Removed the topics Installing the Altera RTE for OpenCL on Big-Endian Systems and Setting the Environment Variables on Big-Endian Systems.  
|                |             |  — In Installing and FPGA Board for Linux, removed information on running the init_opencl script on big-endian systems, and removed related link to Setting the Environment Variables on Big-Endian Systems.  

continued...
<table>
<thead>
<tr>
<th>Date</th>
<th>Version</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>May 2016</td>
<td>2016.05.02</td>
<td>• In Installing an FPGA Board for 64-bit Windows and Linux, provided the following updates:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>— Noted that the SDK supports installation of multiple Custom Platforms.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>— To use the SDK utilities on each board in a system with multiple Custom Platforms, the AOCL_BOARD_PACKAGE_ROOT environment variable setting must correspond to the Custom Platform subdirectory of the associated board.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>— Noted that in a system with multiple Custom Platforms, the host program should use ACD to discover the boards instead of directly linking to the MMD libraries.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• In Building the Host Application for 64-bit Windows, outlined the prerequisite tasks for setting up ACD and ICD for use with Microsoft Visual Studio 2015 prior to building the host application.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Updated all RTE output for a successful kernel execution.</td>
</tr>
<tr>
<td>November 2015</td>
<td>2015.11.02</td>
<td>• Replaced the lists of supported Windows and Linux versions to a link to the Operating System Support page on the Altera website.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Added the %ALTERAOCLSDKROOT%/windows64/bin setting to the list of Windows environment variables.</td>
</tr>
<tr>
<td>May 2015</td>
<td>15.0.0</td>
<td>• Added Windows 8.1 to supported Windows versions.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Added the following figures to illustrate the RTE setup processes for x86-64, big-endian, and SoC systems:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>— RTE Setup Process for x86-64 and Big-Endian Systems</td>
</tr>
<tr>
<td></td>
<td></td>
<td>— RTE Setup Process for SoC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Modified software download and installation instructions for SoC to include the new tar file installation package.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Modified instructions for executing the hello_world OpenCL example design onto a device. You must create your own .aocx file from the hello_world.cl file on a separate development machine, and then use the RTE to deploy the .aocx file onto the device.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Removed licensing sections because an AOCL license is not necessary to run the RTE.</td>
</tr>
<tr>
<td>June 2014</td>
<td>14.0.0</td>
<td>Initial release.</td>
</tr>
<tr>
<td>December 2014</td>
<td>14.1.0</td>
<td>• Reorganized information flow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Updated Red Hat Enterprise Linux (RHEL) version support.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Added licensing information in the Licensing the Software section.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Included information on the init_opencl script for setting environment variables.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Updated the board uninstallation instructions to include the invocation of the aoc1 uninstall utility command.</td>
</tr>
</tbody>
</table>