



# Intel<sup>®</sup> FPGA RTE for OpenCL Pro Edition<sup>™</sup>

## Getting Started Guide

---

Updated for Intel<sup>®</sup> Quartus<sup>®</sup> Prime Design Suite: **18.0**



**UG-OCL005 | 2018.05.04**

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



## Contents

---

<b>1. Intel® FPGA RTE for OpenCL™ Pro Edition Getting Started Guide.....</b>	<b>4</b>
1.1. Prerequisites for the Intel FPGA RTE for OpenCL Pro Edition.....	4
1.2. Contents of the Intel FPGA RTE for OpenCL Pro 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 Utility Command Options.....	7
1.3.3. Managing an FPGA Board.....	8
1.3.4. Managing Host Application.....	8
1.4. Overview of the Intel FPGA RTE for OpenCL Pro Edition Setup Process.....	9
<b>2. Getting Started with the Intel FPGA RTE for OpenCL Pro Edition for 64-Bit Windows....</b>	<b>11</b>
2.1. Downloading the Intel FPGA RTE for OpenCL Pro Edition .....	11
2.2. Installing the Intel FPGA RTE for OpenCL Pro Edition .....	12
2.3. Setting the Intel FPGA RTE for OpenCL Pro 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 an FPGA Board .....	20
<b>3. Getting Started with the Intel FPGA RTE for OpenCL Pro Edition for x86_64 Linux Systems.....</b>	<b>21</b>
3.1. Downloading the Intel FPGA RTE for OpenCL Pro Edition.....	21
3.2. Installing the Intel FPGA RTE for OpenCL Pro Edition.....	21
3.2.1. Setting the Intel FPGA RTE for OpenCL Pro Edition User Environment Variables.....	23
3.3. Verifying Software Installation.....	23
3.4. Installing an FPGA Board.....	24
3.5. Updating the Hardware Image on the FPGA.....	25
3.5.1. Querying the Device Name of Your FPGA Board.....	26
3.5.2. Programming the Flash Memory of an FPGA.....	26
3.6. Executing an OpenCL Kernel on an FPGA.....	27
3.6.1. Building the Host Application.....	27
3.6.2. Running the Host Application.....	28
3.6.3. Output from Successful Kernel Execution.....	28
3.7. Uninstalling the Software.....	29
3.8. Uninstalling an FPGA Board.....	29
<b>4. Getting Started with the Intel FPGA RTE for OpenCL Pro Edition for Intel ARMv7-A SoC FPGA.....</b>	<b>30</b>
4.1. Getting Started with the Intel FPGA RTE for OpenCL Pro Edition for SoC FPGA on Windows.....	31



- 4.1.1. Downloading the Intel FPGA SDK for OpenCL and the SoC EDS..... 32
- 4.1.2. Installing the Intel FPGA SDK for OpenCL Pro Edition for SoC FPGA.....33
- 4.1.3. Installing the Intel SoC FPGA Embedded Development Suite Pro Edition..... 33
- 4.1.4. Recompiling the Linux Kernel Driver..... 34
- 4.1.5. Installing the Intel FPGA RTE for OpenCL Pro Edition onto the SoC FPGA Board..... 34
- 4.1.6. Installing the Intel Arria 10 SoC Development Kit..... 35
- 4.1.7. Executing an OpenCL Kernel on an SoC FPGA..... 35
- 4.1.8. Uninstalling the Intel FPGA RTE for OpenCL..... 36
- 4.2. Getting Started with the Intel FPGA RTE for OpenCL Pro Edition for SoC FPGA on Linux..... 37
  - 4.2.1. Downloading the Intel FPGA SDK for OpenCL and the SoC EDS..... 37
  - 4.2.2. Installing the Intel FPGA SDK for OpenCL Pro Edition for SoC FPGA.....38
  - 4.2.3. Installing the Intel SoC FPGA Embedded Development Suite Pro Edition..... 39
  - 4.2.4. Recompiling the Linux Kernel Driver..... 39
  - 4.2.5. Installing the Intel FPGA RTE for OpenCL Pro Edition onto the SoC FPGA Board..... 40
  - 4.2.6. Installing the SoC Development Kit..... 41
  - 4.2.7. Executing an OpenCL Kernel on an SoC FPGA..... 41
  - 4.2.8. Uninstalling the Intel FPGA RTE for OpenCL..... 42
- A. Document Revision History of the Intel FPGA RTE for OpenCL Pro Edition Getting Started Guide..... 43**



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

---

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

The RTE is a subset of the Intel FPGA Software Development Kit (SDK) for OpenCL Pro Edition<sup>(2)</sup>. 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*.

**Attention:** If you require OpenCL kernel development and deployment functionalities, download the Intel FPGA SDK for OpenCL Pro Edition. Refer to the *Intel FPGA SDK for OpenCL Pro Edition Getting Started Guide* for more information.

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

## Related Information

- [OpenCL Reference Pages](#)
- [OpenCL Specification version 1.0](#)
- [Intel FPGA SDK for OpenCL Getting Started Guide](#)
- [Intel FPGA SDK for OpenCL Cyclone V SoC Getting Started Guide](#)

## 1.1. Prerequisites for the Intel FPGA RTE for OpenCL Pro Edition

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

---

(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](http://www.khronos.org/conformance).



## 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.

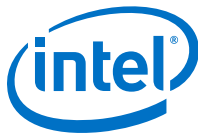
## Software Prerequisites

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 Pro Edition Getting Started Guide](#)

**1.2. Contents of the Intel FPGA RTE for OpenCL Pro Edition**

The Intel FPGA RTE for OpenCL Pro 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 of the Intel FPGA SDK for OpenCL Pro 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 *INTELFPGAOCSDKROOT* environment variable.

**Table 1. Contents of the RTE Installation Directory**

Windows Folder	Linux Directory	ARM Directory	Description
bin	bin	bin	High-level utilities. Include this directory in your <i>PATH</i> environment variable setting.
board	board	board	The Reference Platform available with the RTE. <i>Important:</i> The Reference Platform for the RTE does not include the hardware subdirectory.
host	host	host	Files necessary for compiling your host program.
host \include	host/ include	host/ include	OpenCL Specification version 1.0 header files and software interface files necessary for compiling and linking your host application. The <i>host/include/CL</i> subdirectory also includes the C++ header file <i>cl.hpp</i> . 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. <i>Important:</i> The OpenCL version 1.1 C++ bindings are compatible with OpenCL Specification versions 1.0 and 1.1. Add this path to the <i>include</i> file search path in your development environment.
host \windows64 4\lib	host/ linux64/ lib	host/ arm32/ lib	OpenCL host runtime libraries for the given target platform that provide the OpenCL platform and runtime APIs. These libraries are necessary for linking and running your host application.

*continued...*



Windows Folder	Linux Directory	ARM Directory	Description
			Prior to running your host application, include this directory in the library search path. <ul style="list-style-type: none"> <li>For Linux and ARM®, add the path to the <code>LD_LIBRARY_PATH</code> environment variable setting.</li> <li>For Windows, add the path to the <code>PATH</code> environment variable setting.</li> </ul>
host \windows64\bin	host/ linux64/bin	host/ arm32/bin	Platform-specific binary for the RTE Utility, runtime commands, and DLLs (for Windows) necessary for running your host application, wherever applicable. Include this directory in your <code>PATH</code> environment variable setting.
share\lib \perl	share/lib /perl	share/lib/ perl	Perl scripts and support libraries for the RTE Utility.

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

[Displaying the Software Version](#) on page 7

[Listing the Intel FPGA RTE for OpenCL Utility Command Options](#) on page 7

[Managing an FPGA Board](#) on page 8

[Managing Host Application](#) on page 8

### 1.3.1. Displaying the Software Version

To display the version of the Intel FPGA RTE for OpenCL, 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 Utility Command Options

To display information on the Intel FPGA RTE for OpenCL 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 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 RTE for OpenCL Programming Guide*.

#### Related Information

[Managing an FPGA Board](#)

### 1.3.4. Managing Host Application

The Intel FPGA RTE for OpenCL 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 Programming Guide*:

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

#### Related Information

[Managing Host Application](#)





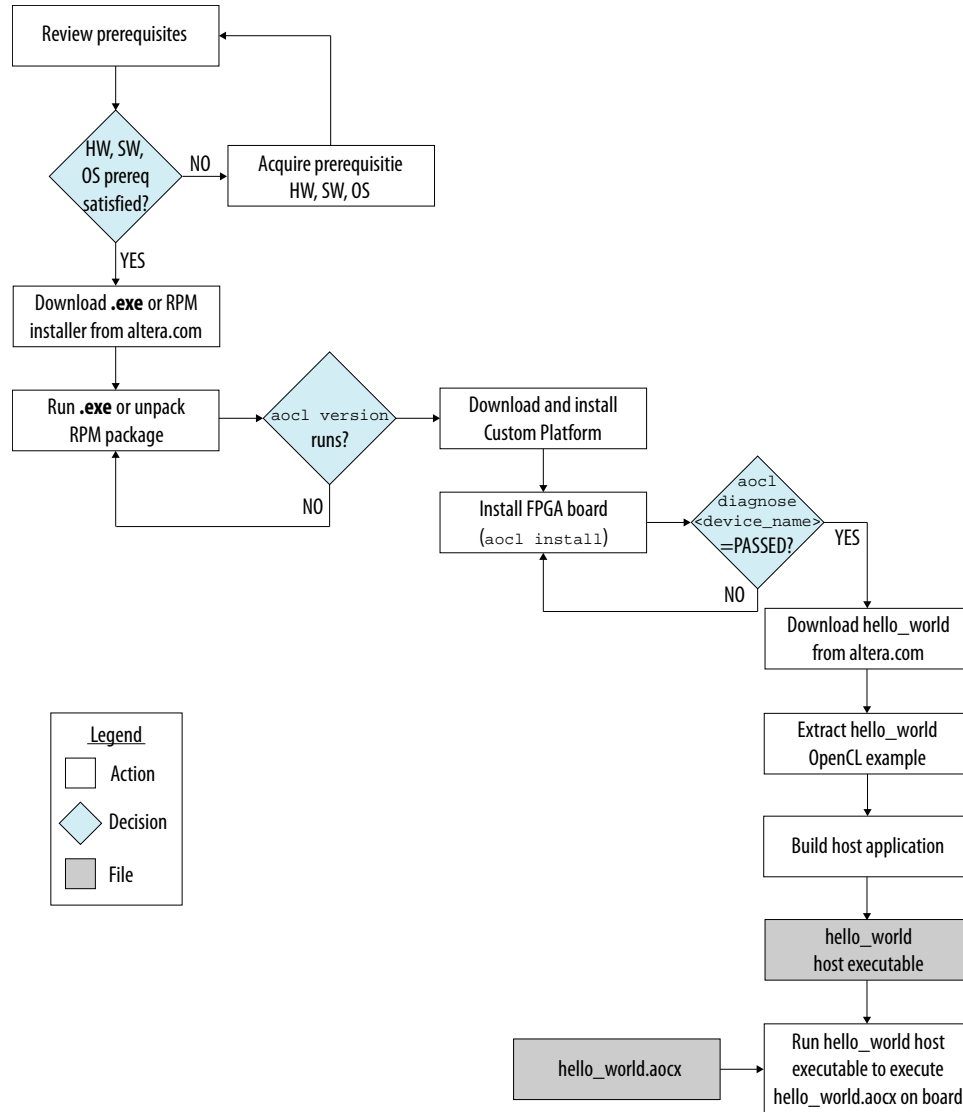
## 1.4. Overview of the Intel FPGA RTE for OpenCL Pro Edition Setup Process

The *Intel FPGA RTE for OpenCL Pro Edition Getting Started Guide* outlines the procedures for installing the Intel FPGA RTE for OpenCL Pro 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 Pro 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**

- [Getting Started with the Intel FPGA RTE for OpenCL Pro Edition for Intel ARMv7-A SoC FPGA](#) on page 30
- [Intel FPGA SDK for OpenCL Pro Edition Getting Started Guide](#)



## 2. Getting Started with the Intel FPGA RTE for OpenCL Pro 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 Pro Edition](#) on page 11
2. [Installing the Intel FPGA RTE for OpenCL Pro Edition](#) on page 12
3. [Setting the Intel FPGA RTE for OpenCL Pro Edition User Environment Variables](#) 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 an FPGA Board](#) on page 20

### 2.1. Downloading the Intel FPGA RTE for OpenCL Pro Edition

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

1. Go to the Intel FPGA SDK for OpenCL Download Center at the following URL:  
<http://dl.altera.com/opencl/>
2. Select the Pro edition.
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 Pro Edition

Install the Windows version of the Intel FPGA RTE for OpenCL Pro 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 .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>\altera\<version>\aclrt-windows64`).

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

Verify that `INTELFPGAOCCLSDKROOT` points to the current version of the software. Open a Windows command window and then type `echo %INTELFPGAOCCLSDKROOT%` at the command prompt.

If the returned path does not point to the location of the RTE installation, edit the `INTELFPGAOCCLSDKROOT` setting.

For instructions on modifying environment variable settings, refer to *Setting the Intel FPGA RTE for OpenCL Pro Edition User Environment Variables*.

### Related Information

[Setting the Intel FPGA RTE for OpenCL Pro Edition User Environment Variables](#) on page 12

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

You have the option to set the Intel FPGA RTE for OpenCL Pro 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**

Environment Variable	Path to Include
<code>PATH</code>	1. <code>%INTELFPGAOCCLSDKROOT%\bin</code> 2. <code>%INTELFPGAOCCLSDKROOT%\windows64\bin</code> 3. <code>%INTELFPGAOCCLSDKROOT%\host\windows64\bin</code> where <code>INTELFPGAOCCLSDKROOT</code> points to the path of the software installation



- 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 `%INTELFPGAOCCLSDKROOT%\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 %INTELFPGAOCCLSDKROOT%\bin to PATH
Adding %INTELFPGAOCCLSDKROOT%\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 accelerator board or SoC device, 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 `a10_ref`, set `AOCL_BOARD_PACKAGE_ROOT` to point to the `<path_to_a10_ref>/a10_ref` directory.



*Note:* If you ran the `$INTELFPGAOCLESDKROOT/init_opencl.sh` script to set the SDK user environment variables, the script has set `AOCL_BOARD_PACKAGE_ROOT` to point to `$INTELFPGAOCLESDKROOT/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 Pro 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 `acl127`.

**Attention:** For possible errors after implementing the `aocl diagnose` utility, refer to [Possible Errors After Running the diagnose Utility](#) section in the *Intel Arria® 10 GX FPGA Development Kit Reference Platform Porting Guide*. 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 Pro Edition User Environment Variables](#) on page 12
- [Querying the Device Name of Your FPGA Board](#) on page 16
- [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 `acl127`) 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 INTELFGAOCLSDKROOT/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>

DIAGNOSTIC_PASSED
```

### 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 II 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 Pro 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 `acl127`) 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.

### 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 Pro 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 `alteraocl.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>
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 %INTELFPGAOCCLSDKROOT%\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
CL_DEVICE_MAX_COMPUTE_UNITS = 1
CL_DEVICE_MAX_CONSTANT_ARGS = 8
CL_DEVICE_MAX_CONSTANT_BUFFER_SIZE = 2147483648
CL_DEVICE_MAX_WORK_ITEM_DIMENSIONS = 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 AOCC: 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.
```

## 2.8. Uninstalling the Software

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

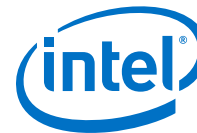
1. In Windows Explorer, navigate to the `altera\<version> <edition>` folder.
2. Delete the `aocl rte-windows64` folder.
3. Remove the following paths from the `PATH` environment variable:
  - a. `%INTELFPGAOCCLSDKROOT%\bin`
  - b. `%INTELFPGAOCCLSDKROOT%\host\windows64\bin`
4. Remove the `INTELFPGAOCCLSDKROOT`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

## 2.9. Uninstalling an 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 Pro 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 Pro Edition](#) on page 21
2. [Installing the Intel FPGA RTE for OpenCL Pro Edition](#) on page 21
3. [Verifying Software Installation](#) on page 23
4. [Installing an FPGA Board](#) on page 24
5. [Updating the Hardware Image on the FPGA](#) on page 25
6. [Executing an OpenCL Kernel on an FPGA](#) on page 27
7. [Uninstalling the Software](#) on page 29
8. [Uninstalling an FPGA Board](#) on page 29

### 3.1. Downloading the Intel FPGA RTE for OpenCL Pro Edition

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

1. Go to the Intel FPGA RTE for OpenCL Download Center at the following URL:  
<http://dl.altera.com/opencl/>
2. Select the Pro edition.
3. Select the software version. The default selection is the current version.
4. Select **Direct Download**.
5. 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.
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](#)

### 3.2. Installing the Intel FPGA RTE for OpenCL Pro Edition

Install the Linux version of the Intel FPGA RTE for OpenCL Pro 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 `INTELFPGAOCCLSDKROOT` 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. Run the `setup_pro.sh` file to install the SDK with the Intel Quartus Prime Pro Edition software.
2. 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/altera/<version>/aclrte_linux64`).

— To install the software using the Red Hat Package Manager (RPM), at the command prompt, type the `rpm -i aocl-rte-<version>.x86_64.rpm` command.

The RPM installs the software in the default location (for example, `opt/altera/aclrte-linux64`).

— To install the software in the default location with verbose progress reporting, type `rpm -ivh aocl-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> aocl-rte-<version>.x86_64.rpm`

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

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

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

For instructions on modifying environment variable settings, refer to *Setting the Intel FPGA RTE for OpenCL Pro Edition User Environment Variables*.

### Related Information

[Setting the Intel FPGA RTE for OpenCL Pro Edition User Environment Variables](#) on page 23



### 3.2.1. Setting the Intel FPGA RTE for OpenCL Pro Edition User Environment Variables

You have the option to set the Intel FPGA RTE for OpenCL Pro 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**

Environment Variable	Path to Include
<i>PATH</i>	<code>\$INTELFPGAOCCLSDKROOT/bin</code> where <i>INTELFPGAOCCLSDKROOT</i> points to the path of the software installation
<i>LD_LIBRARY_PATH</i>	<code>\$INTELFPGAOCCLSDKROOT/host/linux64/lib</code> <code>\$AOCL_BOARD_PACKAGE_ROOT/linux64/lib</code> where <i>AOCL_BOARD_PACKAGE_ROOT</i> points to the path of the Custom or Reference Platform

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

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

- To apply transient environment variable settings, open a bash-shell command-line terminal and run the `source $INTELFPGAOCCLSDKROOT/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 $INTELFPGAOCCLSDKROOT/bin to PATH
Adding $INTELFPGAOCCLSDKROOT/host/linux64/lib to LD_LIBRARY_PATH
Adding $AOCL_BOARD_PACKAGE_ROOT/linux64/lib to LD_LIBRARY_PATH
```

### 3.3. Verifying Software Installation

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

**Attention:** FPGA RTE for OpenCL-supported 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> 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](#)

### 3.4. 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 `a10_ref`, set `AOCL_BOARD_PACKAGE_ROOT` to point to the `<path_to_a10_ref>/a10_ref` directory.





*Note:* If you ran the `$INTELFPGAOCCLSDKROOT/init_opencl.sh` script to set the SDK user environment variables, the script has set `AOCL_BOARD_PACKAGE_ROOT` to point to `$INTELFPGAOCCLSDKROOT/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 Pro 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 possible errors after implementing the `aocl diagnose` utility, refer to [Possible Errors After Running the diagnose Utility](#) section. 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

- [Querying the Device Name of Your FPGA Board](#) on page 26
- [Setting the Intel FPGA RTE for OpenCL Pro Edition User Environment Variables](#) on page 23
- [Intel FPGA SDK for OpenCL FPGA Platforms](#)

### 3.5. 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.



### 3.5.1. 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 `acl127`) 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 `aoctl diagnose` at a command prompt.

The software generates an output that resembles the example shown below:

```
aoctl diagnose: Running diagnostic from INTELFGAOCLSDKROOT/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 aoctl diagnose <device_name>

DIAGNOSTIC_PASSED
```

### 3.5.2. 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 Pro 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 `acl127`) 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](#)

## 3.6. 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.

### 3.6.1. 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_openc1_hello_world_x64_linux_<version>>/hello_world/bin` directory.

### 3.6.2. 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 `$INTELFPGAOCSDKROOT/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_openc1_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.

### 3.6.3. 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_ITEM_DIMENSIONS = 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.
```

### 3.7. Uninstalling the Software

To uninstall the Intel FPGA RTE for OpenCL 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 aocl-rte` command.
2. Remove `$INTELFPGAOCCLSDKROOT/bin` from the `PATH` environment variable.
3. Remove `$INTELFPGAOCCLSDKROOT/host/linux64/lib` from the `LD_LIBRARY_PATH` environment variable.
4. Remove the `INTELFPGAOCCLSDKROOT` environment variable.

### 3.8. Uninstalling an FPGA Board

To uninstall an FPGA board for Linux, invoke the `uninstall` utility command, uninstall the Custom Platform, and unset the relevant environment variables.

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 `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 `LD_LIBRARY_PATH` environment variable.
5. Unset the `AOCL_BOARD_PACKAGE_ROOT` environment variable.



## 4. Getting Started with the Intel FPGA RTE for OpenCL Pro Edition for Intel ARMv7-A SoC FPGA

---

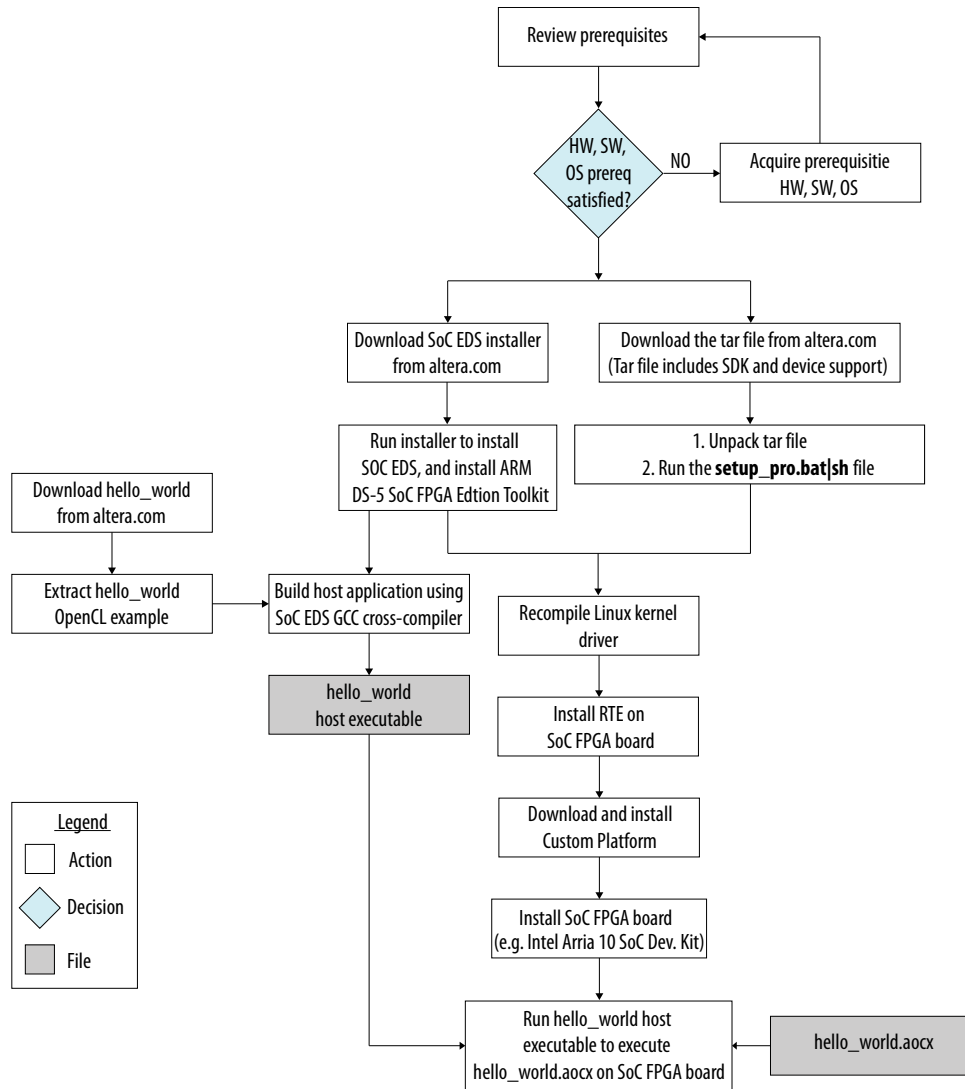
The following sections provide instructions for setting up Windows and Linux versions of the RTE for use with the Intel Arria 10 SoC Development Kit.

For information on the key components of the Intel Arria 10 SoC Development Kit, refer to the Intel Arria 10 SoC Development Kit product page on the Intel FPGA website.



**Figure 2. 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.



**Related Information**

[Intel Arria 10 SoC Development Kit product page](#)

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

To execute an OpenCL kernel onto an SoC FPGA, install Windows versions of the Intel FPGA SDK for OpenCL Pro Edition and the Intel SoC FPGA Embedded Development Suite (EDS) Pro 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.



### 4.1.1. Downloading the Intel FPGA SDK for OpenCL and the SoC EDS

To get started with the Intel FPGA RTE for OpenCL Pro Edition on the Intel Arria 10 SoC Development Kit, download the Intel FPGA SDK for OpenCL Pro Edition and the SoC EDS Pro Edition for Windows from the Intel FPGA Download Center.

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 for SoC instead.

- To download the SDK, follow the instructions outlined in the *Downloading the Intel FPGA SDK for OpenCL Pro Edition* section of the *Intel FPGA SDK for OpenCL Pro 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://dl.altera.com/opencl/>
  - b. Select the Pro edition.
  - 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. From the Download Center, click **Embedded Software** ► **SoC EDS** to enter the download page for the subscription edition of the SoC EDS.
  - b. Select the Pro Edition.
  - c. Select the software version.
  - d. Select **Windows** as the operating system.
  - e. Select **Akamai DLM3 Download Manager** or **Direct Download** as the download method.
  - f. If you select Akamai DLM3 Download Manager as the download method, click **Download**.
  - g. If you select Direct Download as the download method, click **Intel SoC FPGA Embedded Development Suite Pro Edition**.
  - h. 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](#)





## 4.1.2. Installing the Intel FPGA SDK for OpenCL Pro Edition for SoC FPGA

The Intel FPGA SDK for OpenCL Intel Arria 10 SoC Development Kit Reference Platform (a10soc) 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, and a script for setting environment variables.

To get started with the RTE on the Intel Arria 10 SoC Development Kit using the SD flash card image that comes with the SDK, install the SDK for Windows. If you want to create your own SD card image, install the RTE.

You must have administrator privileges.

- To install the SDK, perform the following tasks:
  - a. Unpack the downloaded `AOCL-<version>-<build>-windows.tar` file into a folder that you own.  
The installation path must not contain any spaces (for example, `<home_directory>\altera\<version>\hld`).
  - b. Run the `setup_pro.bat` file to install the SDK and device support.
- To install the RTE, unpack the `.tgz` file 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 Pro Edition User Environment Variables](#) on page 12

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

Install the SoC EDS 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 Design Suite User Guide*.
2. Perform the tasks outlined in the *Installing the Arm DS-5\* Intel SoC FPGA Edition Toolkit* section of the *Intel SoC FPGA Embedded Design Suite User Guide* to install the Arm Development Studio 5\* (DS-5) Intel SoC FPGA Edition Toolkit for your operating system.



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.

3. Consult the *Licensing* section of the *Intel SoC FPGA Embedded Design Suite User Guide* for licensing instructions for the SoC EDS and the Arm DS-5 Intel SoC FPGA Edition Toolkit.

#### Related Information

- [Installing the SoC FPGA EDS on Windows](#)
- [Installing the ARM DS-5 Intel SoC FPGA Edition Toolkit](#)
- [SoC FPGA EDS Licensing](#)

### 4.1.4. Recompiling the Linux Kernel Driver

**Attention:** If you download and install the Intel FPGA SDK for OpenCL Pro Edition, the Intel Arria 10 SoC Development Kit Reference Platform (`INTELFPGAOCLESDKROOT/board/a10soc`) includes an SD card image (`a10soc_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 `INTELFPGAOCLESDKROOT/board/a10soc/arm32/driver` subdirectory of the unpacked `aclrte-arm32` package.
3. Perform the tasks outlined in the *Recompiling the Linux Kernel for the Intel Arria 10 SoC Development Kit* and the *Compiling and Installing the OpenCL Linux Kernel Driver* sections of the *Intel FPGA SDK for OpenCL Intel Arria 10 SoC Development Kit Reference Platform Porting Guide*.

#### Related Information

- [Recompiling the Linux Kernel for the Intel Arria 10 SoC Development Kit](#)
- [Compiling and Installing the OpenCL Linux Kernel Driver](#)

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

The RTE 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.

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

#### 4.1.6. Installing the Intel Arria 10 SoC Development Kit

For information on the setup of the Intel Arria 10 SoC Development Kit, refer to the *Arria 10 SoC Development Kit User Guide*.

##### Related Information

[Arria 10 SoC Development Kit User Guide](#)

#### 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.

##### Related Information

[Creating the Hardware Configuration File of an OpenCL Kernel for SoC FPGA](#)

##### 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 Intel Arria 10 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 `%INTELFPGAOCCLSDKROOT%\board\al10soc`, 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.

### Related Information

[OpenCL Design Examples](#)

#### 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 `%INTELFPGAOCCLSDKROOT%\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 `%INTELFPGAOCCLSDKROOT%\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.

#### 4.1.8. Uninstalling the Intel FPGA RTE for OpenCL

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 INTELFPGAOCSDKROOT
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](#)

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

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

### 4.2.1. Downloading the Intel FPGA SDK for OpenCL and the SoC EDS

To get started with the Intel FPGA RTE for OpenCL Pro Edition on the Intel Arria 10 SoC Development Kit, download the Intel FPGA SDK for OpenCL Pro Edition and the SoC EDS Pro Edition for Linux from the Intel FPGA Download Center.

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 for SoC instead.



- To download the SDK, follow the instructions outlined in the *Downloading the Intel FPGA SDK for OpenCL Pro Edition* section of the *Intel FPGA SDK for OpenCL Pro 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://dl.altera.com/openc/>
  - b. Select the Pro edition.
  - 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. From the Download Center, click **Embedded Software > SoC EDS** to enter the download page for the subscription edition of the SoC EDS.
  - b. Select the Pro edition.
  - c. Select the software version.
  - d. Select **Linux** as the operating system.
  - e. Select **Direct Download** as the download method.
  - f. Click **Intel SoC FPGA Embedded Development Suite Pro Edition**. Download will begin immediately.
  - g. 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](#)

### 4.2.2. Installing the Intel FPGA SDK for OpenCL Pro Edition for SoC FPGA

The Intel FPGA SDK for OpenCL Intel Arria 10 SoC Development Kit Reference Platform (a10soc) 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, and a script for setting environment variables.

To get started with the RTE on the Intel Arria 10 SoC Development Kit using the SD flash card image that comes with the SDK, install the SDK for Linux. If you want to create your own SD card image, install the RTE.

You must have sudo or root privileges.



- To install the SDK, perform the following tasks:
  - a. Unpack the downloaded `AOCL-<version>-<build>-linux.tar` file.
  - b. Run the `setup_pro.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 `INTELFPGAOCCLSDKROOT` to point to the path of the software installation.

Verify that the installer sets the user environment variable `INTELFPGAOCCLSDKROOT` to point to the current version of the software. Open a Windows command window and then type `echo %INTELFPGAOCCLSDKROOT%` 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 `INTELFPGAOCCLSDKROOT` setting.

#### Related Information

[Setting the Intel FPGA RTE for OpenCL Pro Edition User Environment Variables](#) on page 23

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

Install the Intel SoC EDS 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*.
2. Perform the tasks outlined in the *Installing the Arm DS-5 Intel SoC FPGA Edition Toolkit* section of the *Intel SoC FPGA Embedded Development Suite User Guide* to install the Arm Development Studio 5 (DS-5) Intel SoC FPGA Edition Toolkit for your operating system.

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.

3. Consult the *Licensing* section of the *Intel SoC FPGA Embedded Development Suite User Guide* for licensing instructions for the SoC EDS and the Arm DS-5 Intel SoC FPGA Edition Toolkit.

#### Related Information

- [Installing the SoC FPGA EDS on Windows](#)
- [Installing the ARM DS-5 Intel SoC FPGA Edition Toolkit](#)
- [SoC FPGA EDS Licensing](#)

### 4.2.4. Recompiling the Linux Kernel Driver

**Attention:** If you download and install the Intel FPGA SDK for OpenCL Pro Edition, the SoC Development Kit Reference Platform (`INTELFPGAOCCLSDKROOT/board/a10soc`) includes an SD card image (`a10soc_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 `INTELFPGAOCCLSDKROOT/board/a10soc/arm32/driver` subdirectory of the unpacked `aclrte-arm32` package.
3. Perform the tasks outlined in the *Recompiling the Linux Kernel for the SoC Development Kit* and the *Compiling and Installing the OpenCL Linux Kernel Driver* sections of the *Intel FPGA SDK for OpenCL SoC Development Kit Reference Platform Porting Guide*.

#### Related Information

- [Recompiling the Linux Kernel for the Intel Arria 10 SoC Development Kit](#)
- [Compiling and Installing the OpenCL Linux Kernel Driver](#)

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

The RTE installation package for 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.

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.

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





## 4.2.6. Installing the SoC Development Kit

For information on the setup of the SoC Development Kit, refer to the *Arria 10 SoC Development Kit User Guide*.

### Related Information

[Arria 10 SoC Development Kit User Guide](#)

## 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](#)

### 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 Intel Arria 10 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_openc1_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_openc1_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 `$INTELFPGAOCCLSDKROOT/host/arm32/lib`. Run the command `echo $LD_LIBRARY_PATH`.  
If you ran the `init_opencl.sh` script, the `LD_LIBRARY_PATH` setting should point to `$INTELFPGAOCCLSDKROOT/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.

#### 4.2.8. Uninstalling the Intel FPGA RTE for OpenCL

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 INTELFPGAOCCLSDKROOT
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](#)



## A. Document Revision History of the Intel FPGA RTE for OpenCL Pro Edition Getting Started Guide

Document Version	Intel Quartus Prime Version	Changes
2018.05.04	18.0	<ul style="list-style-type: none"> <li>Removed information related to the Intel FPGA RTE for OpenCL Standard Edition, the Intel FPGA SDK for OpenCL Standard Edition, and the Intel SoC FPGA Embedded Development Suite Standard Edition.</li> <li>Updated the <i>Getting Started with the Intel FPGA RTE for OpenCL Pro Edition for ARMv7-A SoC FPGA</i> chapter to include Intel Arria 10 SoC information.</li> </ul>

Date	Version	Changes
November 2017	2017.11.04	<ul style="list-style-type: none"> <li>Rebranded the following: <ul style="list-style-type: none"> <li>ALTERAOCLSDKROOT to INTELFPGAOCLSDKROOT</li> <li>CL_CONTEXT_EMULATOR_DEVICE_ALTERA to CL_CONTEXT_EMULATOR_DEVICE_INTELFPGA</li> </ul> </li> <li>In <a href="#">Prerequisites for the Intel FPGA RTE for OpenCL Pro Edition</a> on page 4, changed Microsoft Visual Studio version 2010 Professional as Microsoft Visual Studio Professional version 2010 or later.</li> <li>In <a href="#">Building the Host Application</a> on page 18, updated references to Microsoft Visual Studio 2015 as Microsoft Visual Studio.</li> </ul>
May 2017	2017.05.05	<ul style="list-style-type: none"> <li>Rebranded the Altera Client Driver (ACD) to FPGA Client Driver (FCD).</li> <li>Updated the download instructions in <i>Downloading the Intel FPGA RTE for OpenCL</i> for Windows and Linux.</li> <li>Added reminders that folder names where you uncompress downloaded OpenCL design examples must not contain spaces.</li> </ul>
October 2016	2016.10.31	<ul style="list-style-type: none"> <li>Rebranded Altera RTE for OpenCL to Intel FPGA RTE for OpenCL.</li> <li>Rebranded Altera SoC Embedded Design Suite to Intel SoC FPGA Embedded Design Suite.</li> <li>Rebranded ARM DS-5 Altera Edition Toolkit to Arm DS-5 Intel SoC FPGA Edition Toolkit.</li> <li>Deprecated and removed support for big-endian system, resulting in the following documentation changes: <ul style="list-style-type: none"> <li>In <i>Prerequisites for the Altera RTE for OpenCL</i>, removed "Red Hat Enterprise Linux 6 on big-endian system" from the list of supported operating systems.</li> <li>In <i>Contents of the Altera RTE for OpenCL</i>, removed the Big-Endian System Directory column from the table of contents in the RTE installation directory.</li> <li>In <i>Downloading the Altera RTE for OpenCL</i> for Linux, removed from Step 6 the choice to select the OpenCL PowerPC RPM installation package.</li> <li>Removed the topics <i>Installing the Altera RTE for OpenCL on Big-Endian Systems</i> and <i>Setting the Environment Variables on Big-Endian Systems</i>.</li> <li>In <i>Installing and FPGA Board</i> for Linux, removed information on running the <code>init_opencl</code> script on big-endian systems, and removed related link to <i>Setting the Environment Variables on Big-Endian Systems</i>.</li> </ul> </li> </ul>

continued...

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

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

ISO  
9001:2008  
Registered



Date	Version	Changes
		<ul style="list-style-type: none"> <li>• In <i>Installing an FPGA Board</i> for 64-bit Windows and Linux, provided the following updates:               <ul style="list-style-type: none"> <li>— Noted that the SDK supports installation of multiple Custom Platforms. To use the SDK utilities on each board in a system with multiple Custom Platforms, the <code>AOCL_BOARD_PACKAGE_ROOT</code> environment variable setting must correspond to the Custom Platform subdirectory of the associated board.</li> <li>— 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.</li> </ul> </li> <li>• In <i>Building the Host Application</i> 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.</li> <li>• Updated all RTE output for a successful kernel execution.</li> </ul>
May 2016	2016.05.02	<ul style="list-style-type: none"> <li>• Replaced the lists of supported Windows and Linux versions to a link to the Operating System Support page on the Altera website.</li> <li>• Added the <code>%ALTERAOCLSDKROOT%\windows64\bin</code> setting to the list of Windows environment variables.</li> </ul>
November 2015	2015.11.02	<ul style="list-style-type: none"> <li>• Added Windows 8.1 to supported Windows versions.</li> <li>• Added the following figures to illustrate the RTE setup processes for x86-64, big-endian, and SoC systems:               <ul style="list-style-type: none"> <li>— <i>RTE Setup Process for x86-64 and Big-Endian Systems</i></li> <li>— <i>RTE Setup Process for SoC</i></li> </ul> </li> <li>• Modified software download and installation instructions for SoC to include the new tar file installation package.</li> <li>• Modified instructions for executing the hello_world OpenCL example design onto a device. You must create your own <code>.aocx</code> file from the <code>hello_world.cl</code> file on a separate development machine, and then use the RTE to deploy the <code>.aocx</code> file onto the device.</li> <li>• Removed licensing sections because an AOCL license is not necessary to run the RTE.</li> </ul>
May 2015	15.0.0	<ul style="list-style-type: none"> <li>• Reorganized instructions into the following sections:               <ul style="list-style-type: none"> <li>— Introduction to the RTE</li> <li>— Getting Started with the RTE for Windows</li> <li>— Getting Started with the RTE for Linux and Big-Endian Systems</li> <li>— Getting Started with the RTE for Altera ARMv7-A SoC, which is further divided into Windows and Linux instructions</li> </ul> </li> </ul>
December 2014	14.1.0	<ul style="list-style-type: none"> <li>• Reorganized information flow.</li> <li>• Updated Red Hat Enterprise Linux (RHEL) version support.</li> <li>• Added licensing information in the <i>Licensing the Software</i> section.</li> <li>• Included information on the <code>init_openc1</code> script for setting environment variables.</li> <li>• Updated the board uninstallation instructions to include the invocation of the <code>aocl uninstall</code> utility command.</li> </ul>
June 2014	14.0.0	Initial release.