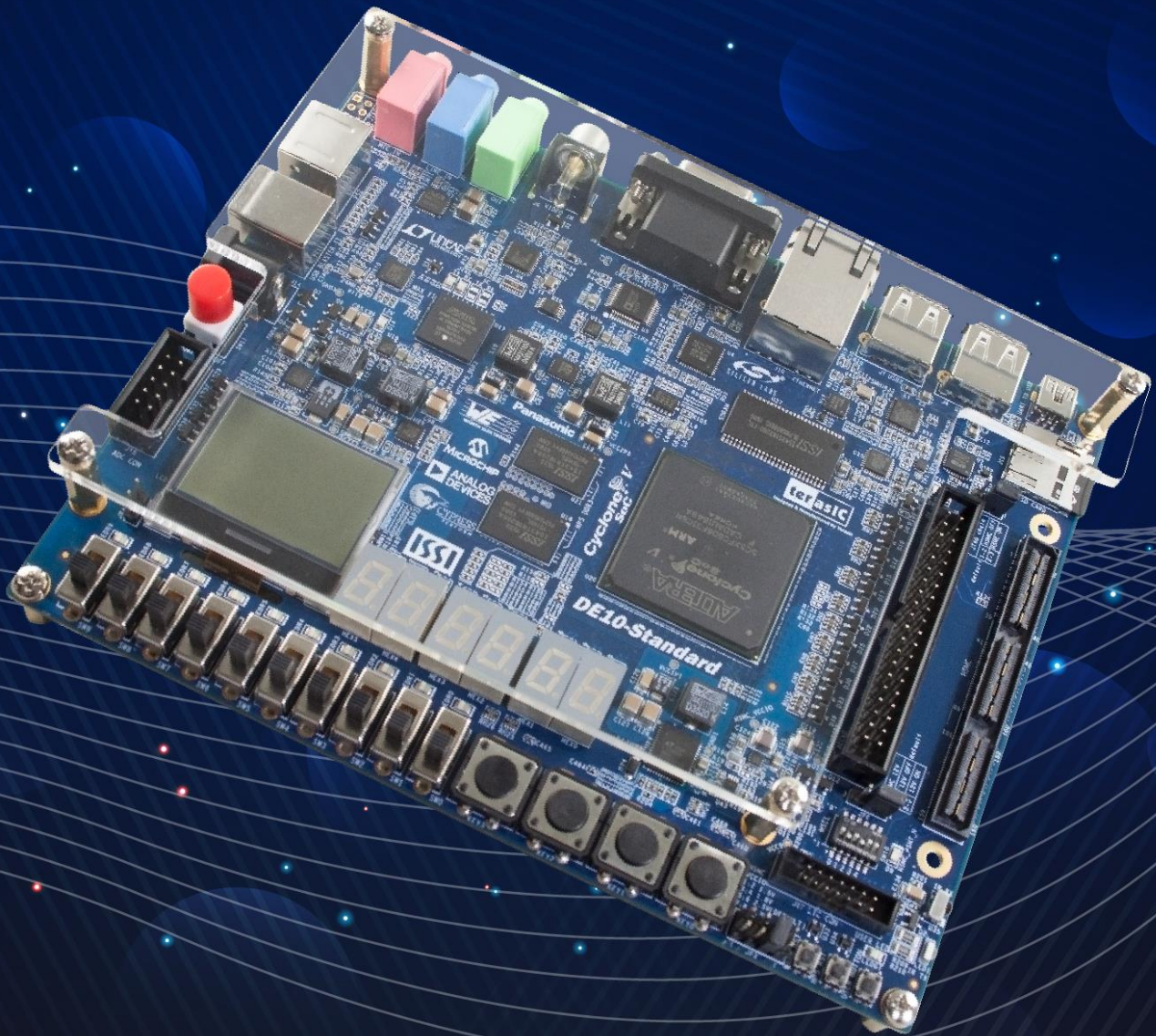


# ***DE10-Standard OpenCL***



Copyright © 2003-2017 Terasic Inc. All Rights Reserved.

<b>CHAPTER 1</b>	<b><i>INTRODUCTION</i></b>	<b>3</b>
1.1	DE10-STANDARD OPENCL BSP	3
1.2	SYSTEM REQUIREMENTS	4
1.3	OPENCL ARCHITECTURE	4
1.4	EXECUTE OPENCL DEMO ON DE10-STANDARD	5
<b>CHAPTER 2</b>	<b><i>OPENCL ON WINDOWS</i></b>	<b>10</b>
2.1	SOFTWARE INSTALLATION	10
2.2	OPENCL LICENSE INSTALLATION	11
2.3	CONFIGURATION OF OPENCL AND DE10-STANDARD BSP	12
2.4	OPENCL ENVIRONMENT VERIFICATION	14
2.5	COMPILE AND EXECUTE OPENCL PROJECT	15
<b>CHAPTER 3</b>	<b><i>OPENCL ON LINUX</i></b>	<b>20</b>
3.1	SOFTWARE INSTALLATION	20
3.2	OPENCL LICENSE INSTALLATION	22
3.3	CONFIGURATION OF ENVIRONMENT VARIABLES	22
3.4	VERIFICATION OF OPENCL ENVIRONMENT	23
3.5	BUILD AND EXECUTE OPENCL PROJECT	24

# Chapter 1

## *Introduction*

DE10-Standard is a robust hardware design platform built with Intel System-on-Chip (SoC) FPGA. It is designed for Intel University Program. This document gives introduction on how to setup OpenCL development environment, compile, and execute example projects for DE10-Standard. Users can refer to Intel SDK for OpenCL Programming Guide for more details about OpenCL coding instruction.

[http://www.altera.com/literature/hb/opencl-sdk/aocl\\_programming\\_guide.pdf](http://www.altera.com/literature/hb/opencl-sdk/aocl_programming_guide.pdf)

### 1.1 DE10-Standard OpenCL BSP

The DE10-Standard OpenCL Board Support Package (BSP) contains required resources for users to develop OpenCL project based on DE10-Standard Board. The BSP is available from the website:

<http://de10-standard.terasic.com/cd>

For Windows Host, please download DE10-Standard\_openCL\_BSP.zip. For Linux, please download the DE10-Standard\_openCL\_BSP.tar.gz. The CentOS 7.0 Linux distribution is recommended for the OpenCL application. These two compressed files are different in the compression type only and their contents are the same. **Figure 1-1** shows the contents of OpenCL BSP for DE10-Standard.








 arm32	2017/1/24 9:06
 de10_standard_sharedonly	2017/1/18 11:16
 driver	2017/1/17 9:27
 examples	2017/1/24 10:09
 board_env.xml	2017/1/17 9:30
 de10_standard_openc1.zip	2017/1/24 15:29
 README.txt	2017/1/20 16:07



Figure 1-1 Contents of OpenCL BSP for DE10-Standard

## 1.2 System Requirements

The following items are required to setup OpenCL for DE10-Standard board:

- Terasic DE10-Standard board
- microSD card with at least 4GB capacity
- microSD card reader
- USB cable (type A to mini-B)
- Ethernet cable or USB-Storage
- Host PC with
  - USB host port
  - 32GB memory is recommended
  - 64-bit Windows 7 or Linux
  - Win32 Disk Imager
  - PuTTY or Minicom(Linux) utility
  - Intel Quartus Prime v16.1 installed with valid license
  - Intel OpenCL v16.1 installed with valid license
  - Intel SoC EDS v16.1 installed

## 1.3 OpenCL Architecture

An OpenCL project consists of OpenCL Kernel and Host Program, as shown in **Figure 1-2**. The Kernel is realized on the FPGA part of SoC FPGA. The Host Program is not the ARM part of the SoC FPGA. It is cross-compiled by Intel SoC EDS installed on Windows or Linux. The Kernel is developed in Quartus and OpenCL SDK is installed on Windows or Linux.

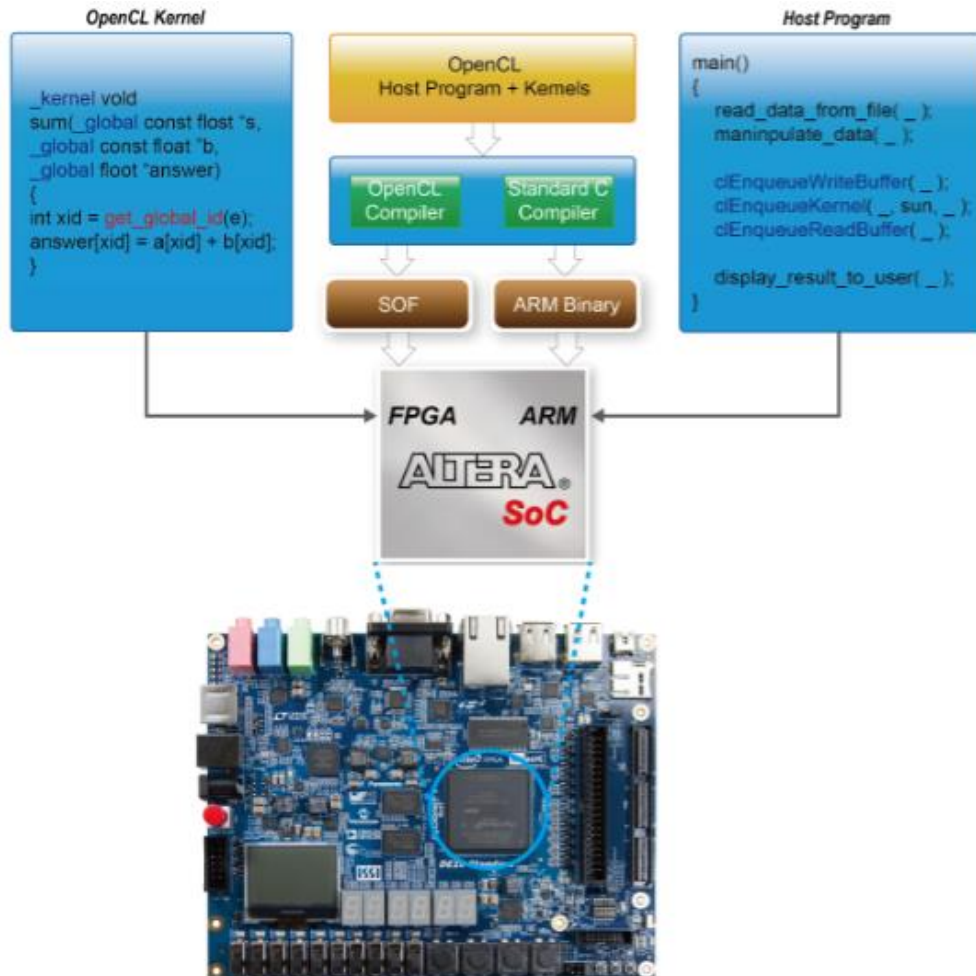


Figure 1-2 Intel SoC FPGA OpenCL architecture

## 1.4 Execute OpenCL Demo on DE10-Standard

This section describes how to execute OpenCL demo on DE10-Standard from the Linux image file included in DE10-Standard Board Support Package (BSP). Windows or Linux Host is required to setup the demo.

### ■ Using Windows Host PC

The following software should be installed on the Windows host PC to complete the setup.

- Disk Imager - available from <http://sourceforge.net/projects/win32diskimager>
- PuTTY- available from <https://the.earth.li/~sgtatham/putty/latest/w32/putty.exe>

The procedures to execute the hello\_world and vector\_add demos are:

1. Download DE10-Standard BSP from <http://de10-standard.terasic.com/cd> and extract the Linux image file **de10\_standard\_openc1.img** from **de10\_standard\_openc1.zip**
2. Write the Linux image file **de10\_standard\_openc1.img** into a microSD card with Disk Imager Utility.
3. Insert the microSD card into the microSD card socket(J11).
4. Make sure the DIP switch (SW10) MSEL[4:0] = 01010.
5. Connect your host PC to the UART-to-USB port (J4) on DE10-Standard via an USB cable. Users need to install the UART-to-USB device driver as described in the DE10-Standard Getting Started Guide.
6. Launch PuTTY utility on your host PC. Make sure the baud rate is set to 115200.
7. Power on DE10-Standard to boot Linux. Login Linux with username 'root' (password is not required).
8. Type "source ./init\_openc1.sh" to load the OpenCL Linux kernel driver and setup environment variable for OpenCL Run-Time library, which is already installed on the microSD card.
9. Launch hello-world demo:
  - Type "cd hello\_world/" to go to the hello\_world folder.
  - Type "aocl program /dev/acl0 hello\_world.aocx" to reconfigure the FPGA with the hello\_world kernel.
  - Type "./hello\_world" to launch the hello\_world host application, as shown in **Figure 1-3**.
10. Launch vectorAdd demo:
  - Type "cd .." to return to the home directory.
  - Type "cd vector\_add" to go to the vector\_add folder.
  - Type "aocl program /dev/acl0 vectorAdd.aocx" to reconfigure the FPGA with the vectorADD kernel.
  - Type "./vector\_add" to launch the vector\_add host application, as shown in **Figure 1-4**.

```

root@socfpga:~# source ./init_openc1.sh
root@socfpga:~# cd hello_world/
root@socfpga:~/hello_world# aocl program /dev/acl0 hello_world.aocx
aocl program: Running reprogram from /home/root/openc1_arm32_rte/board/c5soc/arm
32/bin
Reprogramming was successful
root@socfpga:~/hello_world# ./hello_world
Querying platform for info
=====
CL_PLATFORM_NAME           = Altera SDK for OpenCL
CL_PLATFORM_VENDOR         = Altera Corporation
CL_PLATFORM_VERSION        = OpenCL 1.0 Altera SDK for OpenCL, Ver
sion 16.0

Querying device for info:
=====
CL_DEVICE_NAME             = de10_standard_sharedonlyCyclone V SoC
Development Kit
CL_DEVICE_VENDOR           = Altera Corporation
CL_DEVICE_VENDOR_ID        = 4466
CL_DEVICE_VERSION          = OpenCL 1.0 Altera SDK for OpenCL, Ver
sion 16.0
CL_DRIVER_VERSION          = 16.0
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  = 536870912
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 = 134217728
CL_DEVICE_MAX_WORK_ITEM_DIMENSIONS = 3
CL_DEVICE_MAX_WORK_ITEM_DIMENSIONS = 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
Reprogramming device with handle 1

Kernel initialization is complete.
Launching the kernel...

```

Figure 1-3 Hello-world demo

```

root@socfpga:~/hello_world# cd ..
root@socfpga:~# cd vector_add/
root@socfpga:~/vector_add# aocl program /dev/ocl0 vectorAdd.aocx
aocl program: Running reprogram from /home/root/opencl_arm32_rte/board/c5soc/arm
32/bin
Reprogramming was successful!
root@socfpga:~/vector_add# ./vector_add
Initializing OpenCL
Platform: Altera SDK for OpenCL
Using 1 device(s)
  de10_standard_sharedonlyCyclone V SoC Development Kit
Using AOCX: vectorAdd.aocx
Reprogramming device with handle 1
Launching for device 0 (1000000 elements)

Time: 166.662 ms
Kernel time (device 0): 7.054 ms

Verification: PASS
root@socfpga:~/vector_add#

```

Figure 1-4 vector\_add demo

## ■ Using Linux Host PC with Root Privilege

The following software should be installed on the Linux host PC to complete the setup.

- Minicom – a terminal which can be installed via command “yum install minicom” or “apt-get install minicom”
1. Download DE10-Standard BSP from <http://de10-standard.terasic.com/cd> and extract the linux image file **de10\_standard\_openc1.img** from **de10\_standard\_openc1.zip**.
  2. Write the Linux image file **de10\_standard\_openc1.img** into the microSD card with Disk Imager.
    - Insert the microSD card into a card reader and connect it to the host PC. If the microSD card already contains an image, existing partitions will be mounted automatically. Please unmounts all the partitions.
    - Type `dmesg | tail` command to check which device name is assigned to the microSD card. It's likely to be `/dev/sdb` (change `/dev/sdb` to the device name found in the previous step).
    - Run “`sudo dd if=de10_standard_openc1.img of=/dev/sdb bs=1M`”



- Run “sync”
3. Insert the microSD card into the microSD card slot (J11) of DE10-Standard.
  4. The DIP switch (SW10) on DE10-Standard for MSEL[4:0] must be set to 01010.
  5. Connect the host PC to the UART-to-USB port (J4) on DE10-Standard with an USB cable. Users should install the UART-to-USB device Linux driver as described in the FTDI driver download web page <http://www.ftdichip.com/Drivers/VCP.htm>.
  6. Launch Minicom utility (“minicom -s” for the configuration when it’s launched the first time) on the host PC. The baud rate should be set to 115200. Shutdown the hardware flow control.
  7. Power on DE10-Standard to boot Linux and log in as root. There’s no password required.
  8. Type “source ./init\_openc1.sh” to load the OpenCL Linux kernel driver and setup environment variables for OpenCL Run-Time Environment that is already installed on the microSD card.
  9. Launch hello world demo:
    - Type “cd hello\_world/” to change the current directory to the hello\_world folder.
    - Type “aocl program /dev/acl0 hello\_world.aocx” to reconfigure the FPGA with the hello\_world kernel.
    - Type “./hello\_world” to launch the hello\_world host application, as shown in **Figure 1-3**.
  10. Launch vector Add demo:
    - Type “cd ..” to return to the home directory.
    - Type “cd vector\_add” to change the directory to the vector\_add folder.
    - Type “aocl program /dev/acl0 vectorAdd.aocx” to reconfigure the FPGA with the vectorADD kernel.
    - Type “./vector\_add” to launch the vector\_add host application, as shown in **Figure 1-4**.

## Chapter 2

# *OpenCL on Windows*

This chapter describes how to setup DE10-Standard OpenCL development environment in Windows 64-bit and then build and execute OpenCL project on DE10-Standard. For more details about getting started with Intel OpenCL for Cyclone V SoC, please refer to:

[http://www.altera.com/literature/hb/opencl-sdk/aocl\\_c5soc\\_getting\\_started.pdf](http://www.altera.com/literature/hb/opencl-sdk/aocl_c5soc_getting_started.pdf)

## 2.1 Software Installation

This section describes how to install the software required for developing OpenCL project on DE10-Standard.

### ■ Install Intel Quartus Prime and OpenCL SDK

Intel Quartus Prime and OpenCL SDK are available from the website:

<http://dl.altera.com/opencl>

For Quartus Prime installation, please make sure the Cyclone V device package is selected.

### ■ Install Intel SoC EDS

Intel SoC EDS tool is required to cross-compile the Host Program for ARM processor. The software is available from the website:

<http://dl.altera.com/soceds>

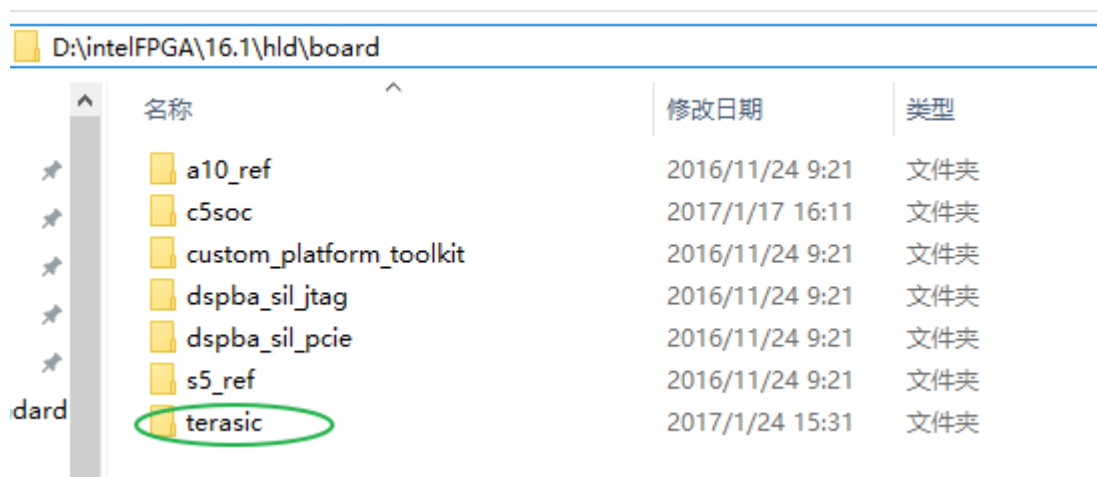
Please make sure the DS-5 is installed during the installation of SoC EDS.

## ■ Install DE10-Standard OpenCL Board Support Package (BSP)

After Quartus Prime and OpenCL SDK are installed, please create a new folder “terasic” under the folder “D:\intelFPGA\16.1\hld\board”, assuming Intel Quartus Prime is installed under the folder “D:\intelFPGA\16.1”. Please download the DE10-Standard BSP file **DE10-Standard.zip** from

<http://de10-standard.terasic.com/cd>

Please uncompress the zip file and copy the “de10\_standard” folder to the “terasic” folder created previously, as shown in **Figure 2-1**.



名称	修改日期	类型
a10_ref	2016/11/24 9:21	文件夹
c5soc	2017/1/17 16:11	文件夹
custom_platform_toolkit	2016/11/24 9:21	文件夹
dspba_sil_jtag	2016/11/24 9:21	文件夹
dspba_sil_pcie	2016/11/24 9:21	文件夹
s5_ref	2016/11/24 9:21	文件夹
terasic	2017/1/24 15:31	文件夹

Figure 2-1 Location of “D:/intelFPGA/16.1/hld/board/terasic” folder

## 2.2 OpenCL License Installation

An OpenCL license is required for Intel OpenCL SDK to compile OpenCL project. Users can purchase the OpenCL license from Intel. A file named “license.dat” will be given upon purchasing OpenCL SDK license. For license installation, please place the file “license.dat” in the local disk drive “c:\” and create a Windows environment variable **LM\_LICENSE\_FILE** with value “c:\license.dat”. The value of this environment variable needs to match the actual location of “license.dat” file.

The procedures below show how to create the **LM\_LICENSE\_FILE** environment variable in

Windows 7:

1. Open the Start Menu and right click on **Computer**. Select **Properties**.
2. Select **Advanced system settings**.
3. Select **Environment Variables** from the **Advanced** tab.
4. Select **New**.
5. In the **New User Variable** dialog shown in **Figure 2-2**, type “**LM\_LICENSE\_FILE**” in the **Variable name** field. Then if you use a license file type “**c:\license.dat**” in the **Variable value** field, else a floating license on a license server type “port number @server IP address”.

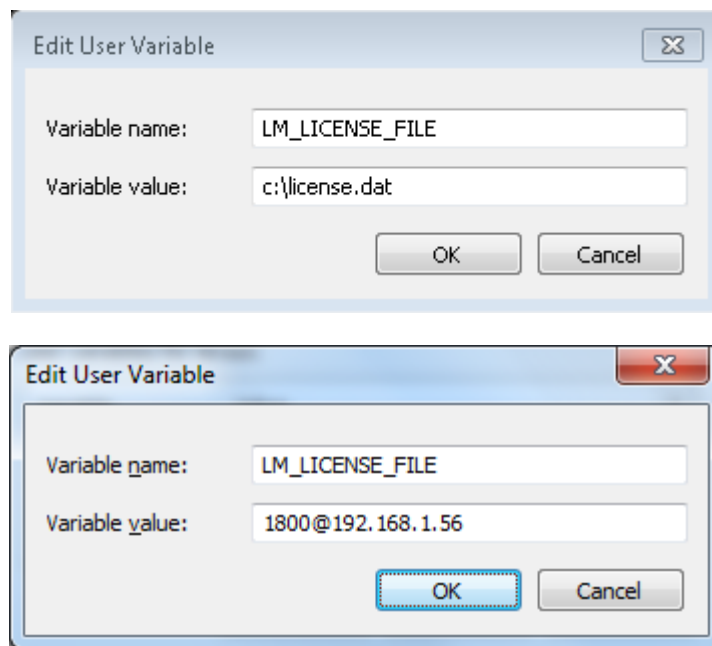


Figure 2-2 Setup LM\_LICENSE\_FILE environment variable

## 2.3 OpenCL Environment Configuration

Please add the following paths into the PATH environment variable:

### ■ OpenCL Configuration

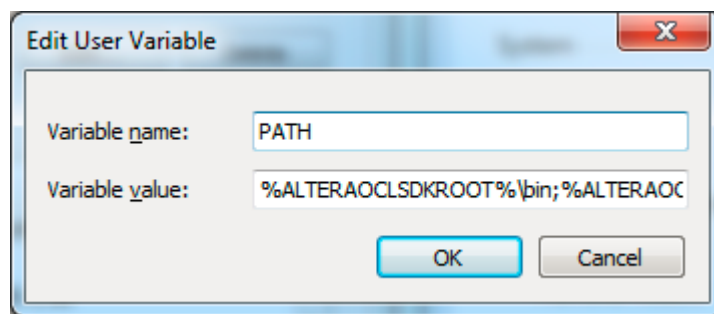
For the operating system to find the OpenCL utilities correctly, users need to add the following paths into the PATH environment variable:

1. %ALTERAOCLSDKROOT%\bin

2. %ALTEAOCLSDKROOT%\host\windows64\bin

Here are the procedures to add these two paths to the **PATH** environment variable on Windows 7:

1. Open the Start Menu and right click on **Computer**. Select **Properties**.
  2. Select **Advanced system settings**.
  3. Select **Environment Variables** from the **Advanced** tab
  4. Select **PATH** item and click the **Edit** button.
  5. In the **Edit User Variable** dialog shown in **Figure 2-3**, add the following two strings into the **Value** edit box. The strings should be separated by the symbol “;”.
- %ALTEAOCLSDKROOT%\bin
  - %ALTEAOCLSDKROOT%\host\windows64\bin



**Figure 2-3 Modify PATH Environment Variable**

## ■ DE10-Standard BSP Configuration

For Intel OpenCL SDK to find the kit location of DE10-Standard correctly, Users need to create an environment variable **AOCL\_BOARD\_PACKAGE\_ROOT** and set its value as:

“%ALTEAOCLSDKROOT%\board\de10\_standard”

The procedures to create the required **AOCL\_BOARD\_PACKAGE\_ROOT** environment variable in Windows 7 are:

1. Open the Start Menu and right click on **Computer**. Select **Properties**.
2. Select **Advanced system settings**.
3. Select **Environment Variables** from the **Advanced** tab.
4. Select **New**.
5. In the **New User Variable** dialog shown in **Figure 2-4**, type “**AOCL\_BOARD\_PACKAGE\_ROOT**” in the **Variable name** field and type “%ALTEAOCLSDKROOT%\board\de10\_standard” in the **Variable value** field.



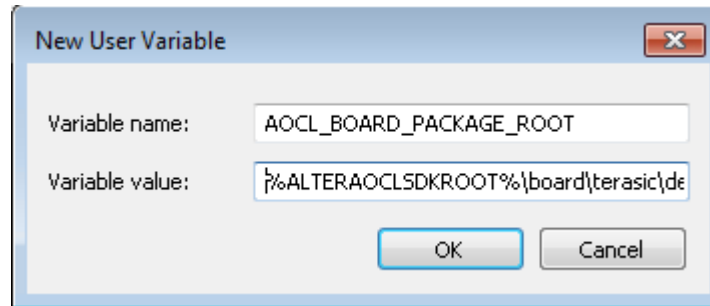


Figure 2-4 Setup AOCL\_BOARD\_PACKAGE\_ROOT environment variable

## 2.4 OpenCL Environment Verification

This section shows how to confirm the OpenCL environment is setup correctly. Please open **Command Prompt** window by clicking Windows **Start** button and click **All Programs**. Click **Accessories** and then click **Command Prompt**.

### ■ Verify Utility

Type “aocl version” command in the command prompt window and see if the version displayed matches the number shown in **Figure 2-5**. If the ‘aocl’ command cannot be found, please check if the “%ALTERAOCLSDKROOT%\bin” path is added to the **PATH** environment variable correctly.

```
C:\Users\matthew>aocl version
aocl 16.1.0.196 (Intel(R) FPGA SDK for OpenCL(TM), Version 16.1.0 Build 196, Copyright (C) 2016 Intel Corporation)
C:\Users\matthew>_
```

Figure 2-5 aocl version of Intel FPGA SDK for OpenCL in the command prompt

### ■ Verify Target Board

Type “aoc --list-boards” command in the command prompt window and make sure “de10\_standard\_sharedonly” is listed in the **Board list**, as shown in **Figure 2-6**. If “de10\_standard\_sharedonly” is not listed, please check if the **AOCL\_BOARD\_PACKAGE\_ROOT** environment

variable is assigned correctly.

```
C:\Users\matthew>aocl version
aocl 16.1.0.196 (Intel(R) FPGA SDK for OpenCL(TM), Version 16.1.0 Build 196, Copyright (C) 2016 Intel Corporation)

C:\Users\matthew>aoc --list-boards
Board list:
  de10_standard_sharedonly

C:\Users\matthew>_
```

Figure 2-6 'de10\_standard\_sharedonly' is listed in the Board list

## ■ How to Check Environment Variables

The value of environment variables can be retrieved by typing 'echo' in the command prompt window. For example, type "echo %AOCL\_BOARD\_PACKAGE\_ROOT%" can retrieve the value of environment variable **AOCL\_BOARD\_PACKAGE\_ROOT**.

```
C:\Users\matthew>echo %AOCL_BOARD_PACKAGE_ROOT%
D:\intelFPGA\16.1\hld\board\terasic\de10_standard

C:\Users\matthew>_
```

Figure 2-7 The value of AOCL\_BOARD\_PACKAGE\_ROOT environment variable

## 2.5 Compile and Execute OpenCL Project

This section shows how to compile and execute OpenCL kernel and OpenCL Host Program provided in the DE10-Standard BSP. Users can follow the same procedures to compile and execute other OpenCL examples for DE10-Standard.

### ■ Compile OpenCL Kernel

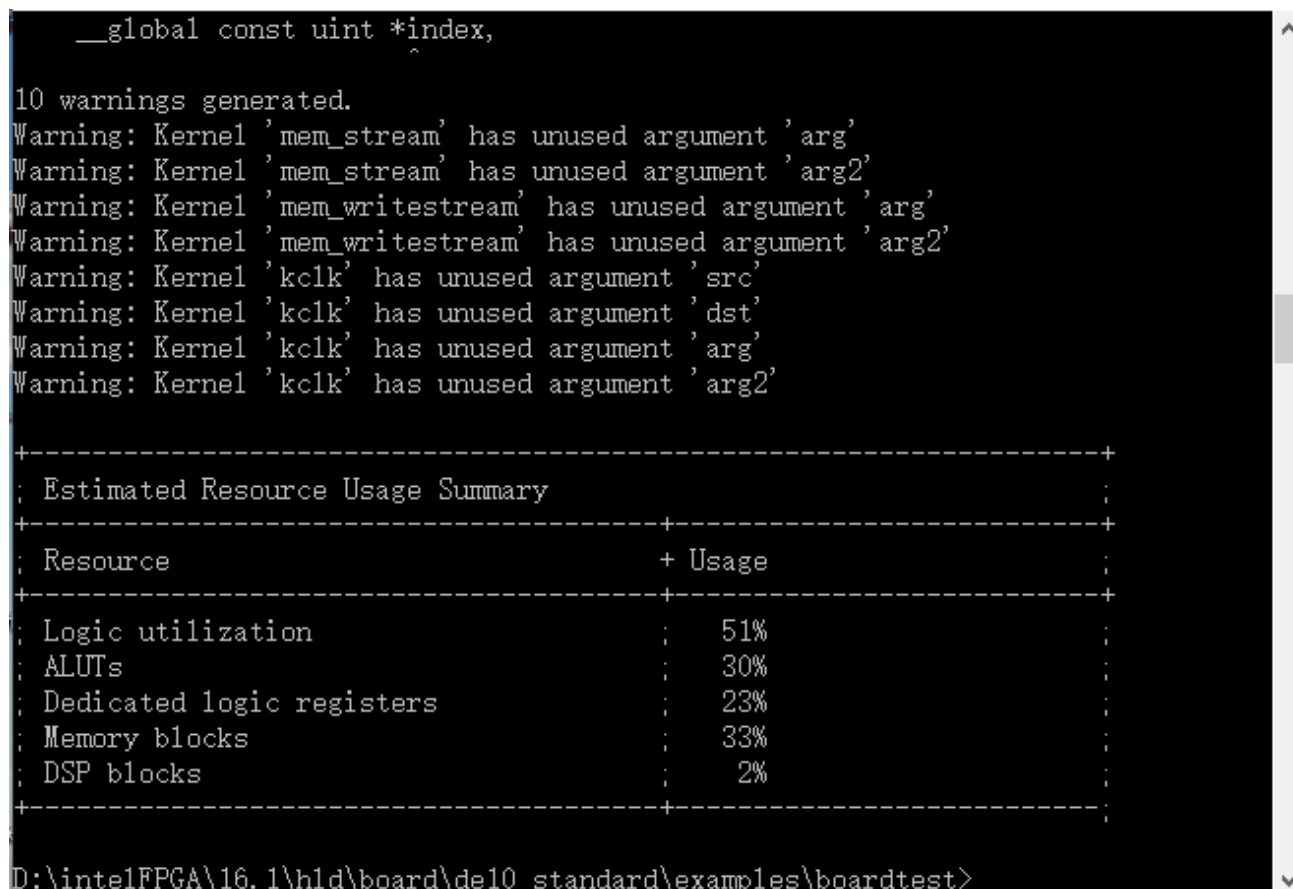
The utility **aoc** (Intel SDK for OpenCL Kernel Compiler) is used to compile OpenCL kernel. Type

“cd D:\intelFPGA\16.1\hld\board\terasic\de10\_standard\examples\boardtest” in the command prompt window to change the current directory to the folder **boardtest** and then type:

```
“aoc device/boardtest.cl -o bin/boardtest.aocx --sw-dimm-partition -board \
    de10_standard_sharedonly --report”
```

to compile the OpenCL kernel. It will take approximately an hour to complete the compilation. When the compilation process is complete, an OpenCL image file **boardtest.aocx** is generated under the **bin** folder. **Figure 2-8** shows the OpenCL kernel is compiled successfully. For more details about the usage of **aoc**, please refer to the **Intel SDK for OpenCL Programming Guide**:

[http://www.altera.com/literature/hb/opencl-sdk/aocl\\_programming\\_guide.pdf](http://www.altera.com/literature/hb/opencl-sdk/aocl_programming_guide.pdf)



```
__global const uint *index,
^
10 warnings generated.
Warning: Kernel 'mem_stream' has unused argument 'arg'
Warning: Kernel 'mem_stream' has unused argument 'arg2'
Warning: Kernel 'mem_writestream' has unused argument 'arg'
Warning: Kernel 'mem_writestream' has unused argument 'arg2'
Warning: Kernel 'kclk' has unused argument 'src'
Warning: Kernel 'kclk' has unused argument 'dst'
Warning: Kernel 'kclk' has unused argument 'arg'
Warning: Kernel 'kclk' has unused argument 'arg2'

+-----+
; Estimated Resource Usage Summary
+-----+
; Resource                                + Usage
+-----+
; Logic utilization                       : 51%
; ALUTs                                  : 30%
; Dedicated logic registers               : 23%
; Memory blocks                          : 33%
; DSP blocks                             : 2%
+-----+

D:\intelFPGA\16.1\hld\board\de10_standard\examples\boardtest>
```

**Figure 2-8 Screenshot of OpenCL kernel compiled successfully by typing “aoc boardtest.cl”**

## ■ Compile Host Program

The Host Program is compiled in Intel SoC EDS. Please launch embedded command shell by executing the “Embedded\_Command\_Shell.bat”, as shown in **Figure 2-9**, under the folder “D:\intelFPGA\16.1\embedded”, assuming Intel SoC EDS is installed under the directory “D:\intelFPGA\16.1”

D:\intelFPGA\16.1\embedded				
	名称	修改日期	类型	大小
	drivers	2017/1/4 14:53	文件夹	
	ds-5	2017/1/4 15:08	文件夹	
	ds-5_installer	2017/1/4 14:54	文件夹	
	embeddedsw	2017/1/4 14:53	文件夹	
	examples	2017/1/4 16:55	文件夹	
	host_tools	2017/1/4 14:53	文件夹	
	ip	2017/1/4 14:53	文件夹	
	Embedded_Command_Shell.bat	2016/10/25 17:41	Windows 批处理...	2 KB
	embedded_command_shell.sh	2016/10/25 17:41	SH 文件	2 KB
	env.sh	2016/10/25 17:41	SH 文件	10 KB
	version.txt	2016/10/25 17:41	文本文档	1 KB

Figure 2-9 Location of embedded\_command\_Shell.bat

Type the following in the command shell:

“cd /cygdrive/D/intelFPGA/16.1/hld/board/terasic/de10\_standard/examples/boardtest/”

to change the current directory to the folder where the board test project is located. Type “make” to build the host project shown in **Figure 2-10**. If the compilation is successful, a binary file **boardtest\_host** will be generated under the boardtest folder.

```
$ ls
boardtest boardtest.aoco boardtest.aocx device host Makefile README.txt

matthew@matthew-PC /cygdrive/d/intelFPGA/16.1/hld/board/de10_standard/examples/b
oardtest
$ make
arm-linux-gnueabi-g++ -fPIC host/src/main.cpp host/src/memspeed.cpp host/sr
c/reorder.cpp host/src/reorder_ocl.cpp host/src/hostspeed.cpp host/src/hostsp
eed_ocl.cpp host/src/aclutil.cpp host/src/timer.cpp host/src/rwtest.cpp -o bo
ardtest_host -DLINUX -ID:/intelFPGA/16.1/hld/host/include -LD:/intelFPGA/16.1\h
ld\board\de10_standard\arm32\lib -LD:/intelFPGA/16.1/hld/host/arm32/lib -Wl,--no
-as-needed -lalteracl -lalterammdpcie -lelf -lstdc++
d:/intelFPGA/16.1/embedded/ds-5/sw/gcc/bin/./lib/gcc/arm-linux-gnueabi/4.8.3/
../../../../arm-linux-gnueabi/bin/ld.exe: warning: libacl_emulator_kernel_rt.s
o, needed by D:\intelFPGA\16.1\hld\board\de10_standard\arm32\lib\libalteracl.so,
not found (try using -rpath or -rpath-link)

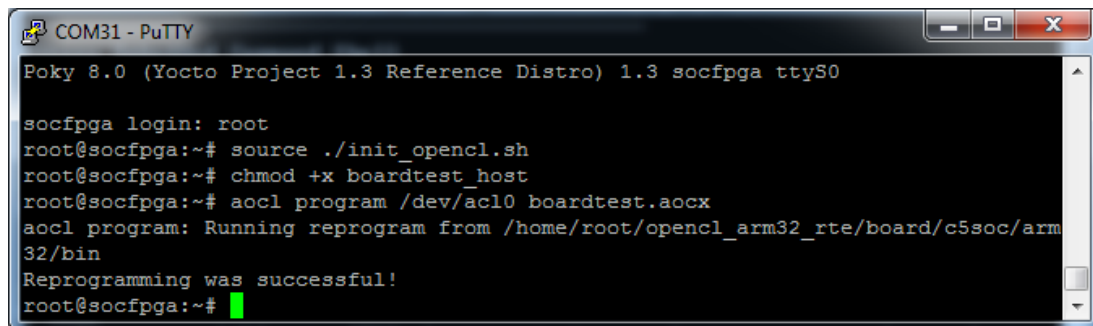
matthew@matthew-PC /cygdrive/d/intelFPGA/16.1/hld/board/de10_standard/examples/b
oardtest
$
```

Figure 2-10 Type ‘make’ to build the boardtest host project

## ■ Execute Board Test Project

Please boot DE10-Standard with the Linux image generated from Chapter 1.4 for DE10-Standard OpenCL. Users need to copy both the kernel file **boardtest.aocx** and host file **boardtest\_host** generated from the previous section from the host PC to Linux running on DE10-Standard. Users can copy the file by typing Linux “scp” command through Ethernet or USB storage.

After these two files are copied to Linux running on DE10-Standard, please go to the terminal and type “source ./init\_openc1.sh” to setup OpneCL environment. Type “chmod +x boardtest\_host” to add execution attribute to the host file and type “aocl program /dev/acl0 boardtest.aocx” to configure the FPGA, as shown in **Figure 2-11**. Type “./boardtest\_host” to launch the host application, as shown in **Figure 2-12**.



```
COM31 - PuTTY
Poky 8.0 (Yocto Project 1.3 Reference Distro) 1.3 socfpga ttyS0

socfpga login: root
root@socfpga:~# source ./init_openc1.sh
root@socfpga:~# chmod +x boardtest_host
root@socfpga:~# aocl program /dev/acl0 boardtest.aocx
aocl program: Running reprogram from /home/root/openc1_arm32_rte/board/c5soc/arm
32/bin
Reprogramming was successful!
root@socfpga:~#
```

**Figure 2-11 “aocl program /dev/acl0 boardtest.aocx” configures FPGA successfully**



```

It is assumed that all memory interfaces have equal widths.

BOARD BANDWIDTH UTILIZATION = 42.06%
Warning : Board bandwidth utilization is less than 90%

Kernel mem bandwidth assuming ideal memory: 15953 MB/s
    * If this is lower than your board's peak memory
    * bandwidth then your kernel's clock isn't fast enough
    * to saturate memory
    * approx. fmax = 125

Kernel mem bandwidth assuming ideal memory is greater than board's peak memory b
andwidth. Success.

KERNEL-TO-MEMORY BANDWIDTH = 2692 MB/s/bank

*****
***** TEST COMPLETED FOR DEVICE 0 *****
*****

BOARDTEST PASSED
root@socfpga:~/boardtest# █

```

Figure 2-12 “boardtest\_host” is executed successfully

## Chapter 3

# *OpenCL on Linux*

This chapter describes how to setup the environment for the development of OpenCL on Linux, build OpenCL project including kernel and host application, followed by execution and verification of OpenCL project.

For more details about OpenCL on Linux, please refer to the Getting Started Guide of Intel OpenCL Cyclone V SoC:

[http://www.altera.com/literature/hb/opencl-sdk/aocl\\_c5soc\\_getting\\_started.pdf](http://www.altera.com/literature/hb/opencl-sdk/aocl_c5soc_getting_started.pdf)

### 3.1 Software Installation

This section describes where to download and how to install the software required for OpenCL on DE10-Standard.

- **Download Intel Quartus Prime and OpenCL SDK**

Intel Quartus Prime and OpenCL SDK are available from the website of Intel PSG:

<http://dl.altera.com/opencl/>

Follow the link and select Linux operation system with version 16.1, as shown in **Figure 3-1**.

## Intel FPGA SDK for OpenCL™

Release date: November, 2016

Latest Release: v16.1

Select release: 16.1

**Download Method**  ☐ Akamai DLM3 Download Manager  ☒ Direct Download

Windows SDK

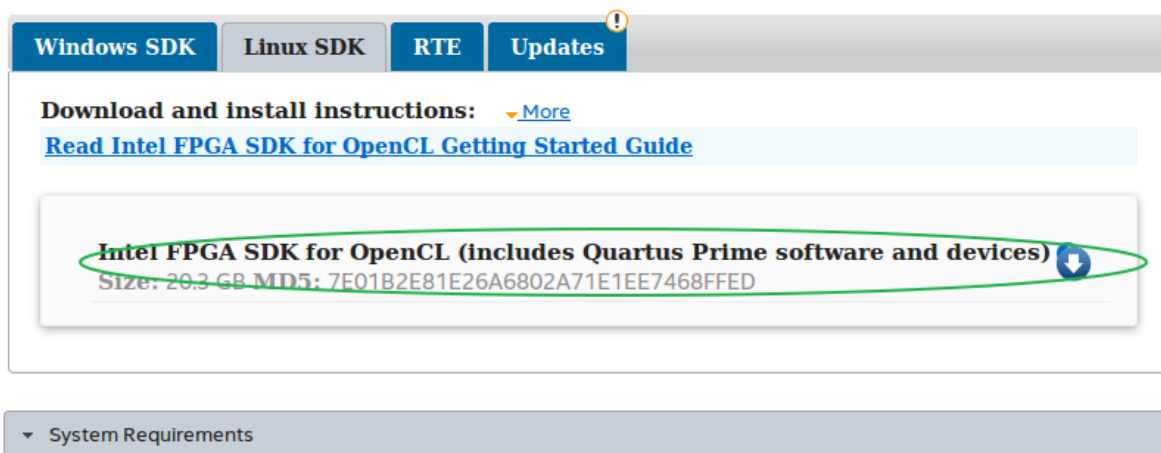
Linux SDK

RTE

Updates 

**Figure 3-1 Download the Linux version of OpenCL SDK v16.1 from the website of Intel PSG**

Choose Direct Download as the download method since the download manager is for Windows only. Click the arrow to download Intel FPGA Design Software, as shown in [Figure 3-2](#), and make sure the Cyclone V device is included. Users can download Intel SDK for OpenCL as standalone installer or RPM package.



**Figure 3-2 Download Intel OpenCL SDK Linux version with Cyclone V device support**

- **Install Intel SoC EDS**

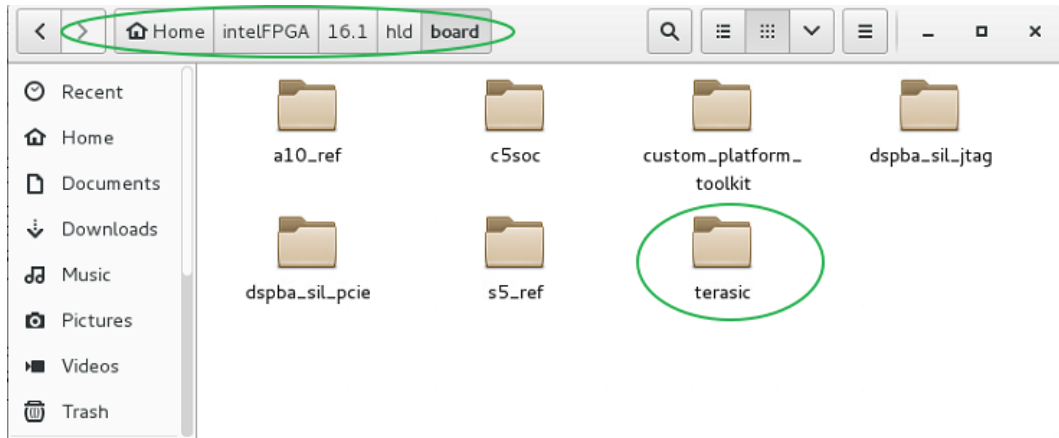
Intel SoC EDS tool is required to cross-compile the host program for ARM processor. The software is available from the website:

<http://dl.altera.com/soceds>

Please make sure DS-5 is installed during the installation of SoC EDS.

- **Install DE10-Standard OpenCL Board Support Package (BSP)**

After Quartus Prime and OpenCL SDK are installed, please copy the terasic folder to the Intel OpenCL SDK folder “/root/intelFPGA/16.1/”, as shown in **Figure 3-3**.



**Figure 3-3** Copy the terasic folder to the /root/intelFPGA/16.1/hld/bolard folder

## 3.2 OpenCL License Installation

A license for OpenCL is required to compile OpenCL project with Intel OpenCL SDK. Users can purchase the OpenCL license from Intel. After users have obtained a license file named “license.dat”, it needs to be placed in the local disk such as “/root/intelFPGA/16.1/hld/license.dat”. Users also need to create an environment variable **LM\_LICENSE\_FILE** and set its value as “/root/intelFPGA/16.1/hld/license.dat”, which corresponds to the actual location of the license file. If you use a floating license on a license server, please set its value as “port number@server IP address”

The next section will describe how to setup the license environment.

## 3.3 Configuration of Environment Variables

If users install the Intel FPGA development software and OpenCL SDK on a system that does not contain any .cshrc or bash resource file (.bashrc) in the directory, the ALTERALOCSDKROOT

and PATH environment variables must be set manually. Users also need to create an environment variable **AOCL\_BOARD\_PACKAGE\_ROOT** for Intel OpenCL SDK to find the kit location of DE10-Standard correctly. The value of this environment variable needs to be set as:

“\$ALTERAOCLSDKROOT/board/terasic/de10\_standard”

Alternatively, users can edit the “/etc/profile”, and append the environment variables to it. It can be done by typing “*gedit /etc/profile*” command in the terminal to open the **profile** file with the **gedit** editor tool and append the following settings to the **profile** file. After the edit is complete, save the file and type “*source /etc/profile*” command in the terminal to apply the settings.

```
export QUARTUS_ROOTDIR=/root/intelFPGA/16.1/quartus
export ALTERAOCLSDKROOT=/root/intelFPGA/16.1/hld
export
PATH=$PATH:$QUARTUS_ROOTDIR/bin:/root/intelFPGA/16.1/embedded/ds-5/bin: \
/root/intelFPGA/16.1/embedded/ds-5/sw/gcc/bin:$ALTERAOCLSDKROOT/bin: \
$ALTERAOCLSDKROOT/linux64/bin:
export LD_LIBRARY_PATH=$ALTERAOCLSDKROOT/linux64/lib
export AOCL_BOARD_PACKAGE_ROOT=$ALTERAOCLSDKROOT/board/terasic/de10_standard
export QUARTUS_64BIT=1
export LM_LICENSE_FILE=1800@192.168.1.56
```

## 3.4 Verification of OpenCL Environment

This section shows how to make sure the OpenCL environment is setup correctly. Please open the terminal window in Linux.

- **Verify Utility**

Type “aocl version” command in the terminal and make sure the aocl version reported matches the information shown in **Figure 3-4**. If the ‘aocl’ command cannot be found, please check if the “\$ALTERAOCLSDKROOT/bin” path is added to the **PATH** environment variable correctly.

```
[root@localhost de10_standard]# aocl version
aocl 16.1.0.196 (Intel(R) FPGA SDK for OpenCL(TM), Version 16.1.0 Build 196, Copyright (C) 2016 Intel Corporation)
[root@localhost de10_standard]#
```

**Figure 3-4 The information about aocl version**



- **Verify Target Board**

Type “aoc --list-boards” command in the terminal and make sure “de10\_standard\_sharedonly” is displayed in the Board list, as shown in [Figure 3-5](#). If “de10\_standard\_sharedonly” is not listed, please check if the **AOCL\_BOARD\_PACKAGE\_ROOT** environment variable is assigned correctly.

```
[root@localhost de10_standard]# aoc --list-boards
Board list:
  de10_standard_sharedonly
[root@localhost de10_standard]#
```

Figure 3-5 ‘de10\_standard\_sharedonly’ is shown in the Board list

- **Check Environment Variables**

The values of environment variables can be retrieved by typing ‘echo’ command in the terminal. For example, type “echo \$AOCL\_BOARD\_PACKAGE\_ROOT” can dump the value of environment variable **AOCL\_BOARD\_PACKAGE\_ROOT**, as shown in [Figure 3-6](#).

```
[root@localhost de10_standard]# echo $AOCL_BOARD_PACKAGE_ROOT
/root/intelFPGA/16.1/hld/board/terasic/de10_standard
[root@localhost de10_standard]#
[root@localhost de10_standard]#
[root@localhost de10_standard]#
```

Figure 3-6 Check the environment variable AOCL\_BOARD\_PACKAGE\_ROOT

## 3.5 Build and Execute OpenCL Project

This section shows how to compile and test OpenCL example project and OpenCL host program provided in DE10-Standard BSP. Users can follow the same procedures to compile and test other OpenCL projects for DE10-Standard.

- **Compile OpenCL Kernel**

The utility **aoc** (Intel SDK for OpenCL Kernel Compiler) is used to compile OpenCL kernel. Type “cd /root/intelFPGA/16.1/hld/board/terasic/de10\_standard/examples/boardtest” in the terminal to change the current directory to the **boardtest** project folder. Type

```
“aoc device/boardtest.cl -o bin/boardtest.aocx --sw-dimm-partition --board \
    de10_standard_sharedonly --report”
```

to compile the OpenCL kernel. The compilation process will take about one hour. When the

compilation process is complete, an OpenCL image file **boardtest.aocx** is generated under the **bin** folder. **Figure 3-7** shows the screenshot after the OpenCL kernel is compiled successfully. For more information about the usage of **aoc**, please refer to the **Intel SDK for OpenCL Programming Guide**:

[http://www.altera.com/literature/hb/opencl-sdk/aocl\\_programming\\_guide.pdf](http://www.altera.com/literature/hb/opencl-sdk/aocl_programming_guide.pdf)

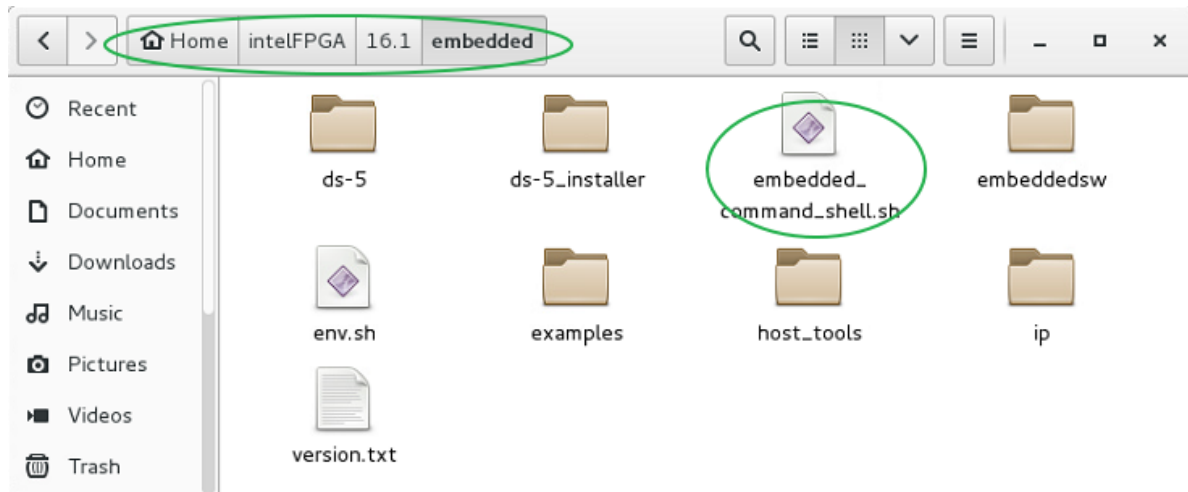
```
[root@localhost boardtest]# aoc device/boardtest.cl -o bin/boardtest.aocx --sw-d
imm-partition --board del0_standard_sharedonly --report
aoc: Selected target board del0_standard_sharedonly
/root/intelFPGA/16.1/hld/board/terasic/del0_standard/examples/boardtest/device/b
oardtest.cl:76:20: warning: declaring kernel argument with no 'restrict' may lea
d to low kernel performance
    __global uint *dst,
                ^
/root/intelFPGA/16.1/hld/board/terasic/del0_standard/examples/boardtest/device/b
oardtest.cl:77:26: warning: declaring kernel argument with no 'restrict' may lea
d to low kernel performance
    __global const uint *index,
                        ^
2 warnings generated.
Warning: Kernel 'mem_writestream' has unused argument 'arg'
Warning: Kernel 'mem_writestream' has unused argument 'arg2'
Warning: Kernel 'mem_readstream' has unused argument 'arg2'
Warning: Kernel 'mem_read_writestream' has unused argument 'arg'

+-----+
; Estimated Resource Usage Summary
+-----+
; Resource                                + Usage
+-----+
; Logic utilization                        ; 50%
; ALUTs                                   ; 30%
; Dedicated logic registers               ; 23%
; Memory blocks                           ; 31%
; DSP blocks                              ; 0%
+-----+
[root@localhost boardtest]#
```

**Figure 3-7 OpenCL kernel is compiled successfully**

- **Compile Host Program**

Intel SoC EDS is used to compile the host program. Please launch the embedded command shell by executing the “Embedded\_Command\_Shell.sh”, as shown in **Figure 3-8**, under the folder “/root/intelFPGA/16.1/embedded”, assuming Intel SoC EDS is installed under the directory “/root/intelFPGA/16.1”.



**Figure 3-8 Location of Embedded\_Command\_Shell.sh**

In the command shell, please type:

`“cd /root/intelFPGA/16.1/hld/board/terasic/de10_standard/examples/boardtest/”`

and the current directory will be changed to the folder where the board test project is located. Type “make” to build the host project, as shown in **Figure 3-9**. If the host project is compiled successfully, a host binary file **boardtes\_host** will be generated under the boardtest folder.

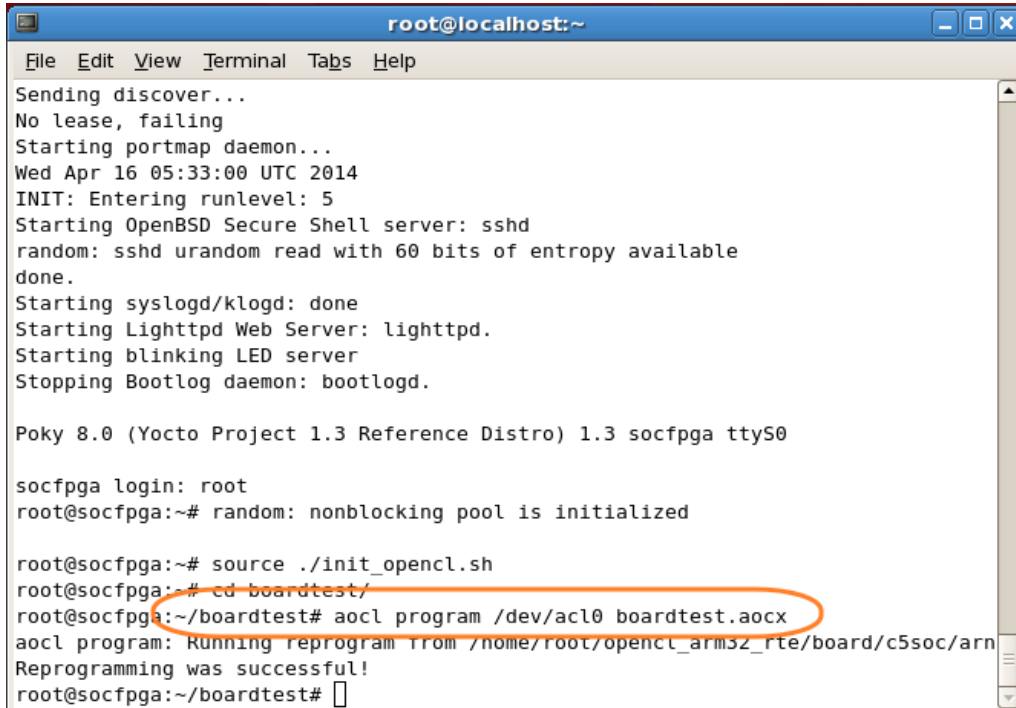
```
[root@localhost boardtest]# make
arm-linux-gnueabi-g++ -fPIC host/main.cpp host/memspeed.cpp host/reorder.cpp
host/reorder_ocl.cpp host/hostspeed.cpp host/hostspeed_ocl.cpp host/aclutil.cpp
host/timer.cpp host/rwtest.cpp host/kernel_launch.cpp host/kernel_rw.cpp -o boardtest_host
-DLINUX -I/root/intelFPGA/16.1/hld/host/include -L/root/intelFPGA/16.1/hld/host/arm32/lib
-L/root/intelFPGA/16.1/hld/host/arm32/lib -Wl,--no-as-needed -lalteracl -lalterammdpcie
-lstdc++ -lelf -lstdc++ -lrt /root/intelFPGA/16.1/embedded/ds-5/sw/gcc/bin/./lib/gcc/arm-linux-gnueabi/4.8.3/../../../../arm-linux-gnueabi/bin/ld: warning: libacl_emulator_kernel_rt.so,
needed by /root/intelFPGA/16.1/hld/board/terasic/de10_standard/arm32/lib/libalteracl.so, not found (try using -rpath or -rpath-link)
[root@localhost boardtest]#
```

**Figure 3-9 Type ‘make’ to build the boardtest\_host project**

- **Test boardtest project**

Please boot DE10-Standard with the Linux image for DE10-Standard OpenCL described in Chapter 1.4. Users need to copy the generated kernel file **boardtest.aocx** and the host file **boardtest\_host** from the host PC to the Linux system running on DE10-Standard. This can be done by establishing SSH connection via Ethernet with “scp” command or via usb-storage with “mount” command.

After these two files are copied to the Linux system running on DE10-Standard, please add executable attribution with “chmod” command and type “aocl program /dev/acl0 boardtest.aocx” to configure the FPGA, as shown in **Figure 3-10**. Type “./boardtest\_host” to launch the host application, as shown in **Figure 3-11**.



```

root@localhost:~
File Edit View Terminal Tabs Help
Sending discover...
No lease, failing
Starting portmap daemon...
Wed Apr 16 05:33:00 UTC 2014
INIT: Entering runlevel: 5
Starting OpenBSD Secure Shell server: sshd
random: sshd urandom read with 60 bits of entropy available
done.
Starting syslogd/klogd: done
Starting Lighttpd Web Server: lighttpd.
Starting blinking LED server
Stopping Bootlog daemon: bootlogd.

Poky 8.0 (Yocto Project 1.3 Reference Distro) 1.3 socfpga ttyS0

socfpga login: root
root@socfpga:~# random: nonblocking pool is initialized

root@socfpga:~# source ./init_openc1.sh
root@socfpga:~# cd boardtest/
root@socfpga:~/boardtest# aocl program /dev/acl0 boardtest.aocx
aocl program: Running reprogram from /home/root/openc1_arm32_rte/board/c5soc/arm
Reprogramming was successful!
root@socfpga:~/boardtest#
  
```

**Figure 3-10 “aocl program /dev/acl0 boardtest.aocx” configures the FPGA successfully**

```
root@localhost:~  
File Edit View Terminal Tabs Help  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
OpenCL Notification Callback: Requested memory object size exceeds device limits  
Min time:      185520  
Max time:     390332290  
Avg time:       294915  
Finished 4000 iterations with 0 errors  
  
SNOOP TEST PASSED  
root@socfpga:~/boardtest#  
CTRL-A Z for help |115200 8N1 | NOR | Minicom 2.1 | VT102 | Offline
```

**Figure 3-11 “boardtest host” test is successful**