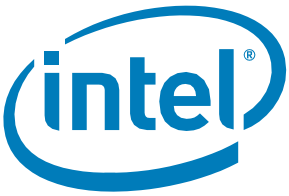


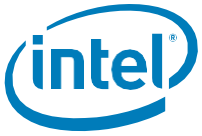
Intel® Xeon® + FPGA

Intel® Software Development Kit (SDK) for OpenCL™ /Board Support Package (BSP) Installation Guide – NDA

Revision 0.5

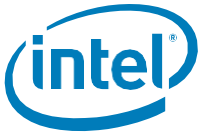
December 2016

Intel Confidential



Contents

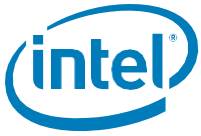
Figures



Revision History

Document Number	Revision Number	Description	Date
VIP Release	0.5	• BSP v1.0	December 2016
VIP Release	0.2	• BSP v0.2	July 2016

§



1 Introduction

1.1 Terminology

Table 1-1. Reference Documents

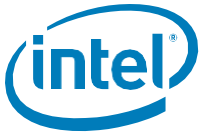
Term	Description
BSP	Board Support Package
MCP	Multi-Core Processor
RTE	Runtime Environment

1.2 Reference Documents

Table 1-2. Reference Documents

Title	CDI Document Location
Xeon® + FPGA Broadwell SDP Startup Guide Video	568255
Startup Video 4.15.15	568227

§



2 Version Release History

v0.1.0 (April 2016)

- First release

v0.1.1 (May 2016)

Changes:

- Enabled PR between OpenCL™ kernels
- Enabled BSP to interleave memory accesses across all Virtual Channels
- Changed naming of BSP
- Functional fixes

v0.1.2 (June 2016)

Changes:

- Increased clock frequency of MPF and BSP Avalon->CCI Translation layers for better memory bandwidth
- Several functional fixes
- Workaround to smaller than cache-line stores, may cause performance degradation for some kernels, but should be removed next drop

v0.1.3 (June 2016)

Changes:

- Moved BSP to ACL and ACDS 16.0

v0.1.4 (July 2016)

Changes:

- FMAX improvements
- Fix to 'aocl diagnose'v0.1.3 (June 2016)

v0.2 (July 2016)

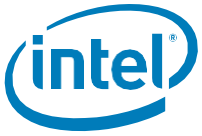
Changes:

- Timing closure of BSP logic

v0.3.1 (Sept 2016)

Changes:

- Re-architecture of Avalon-to-CCI Bridge
- FMAX optimizations
- Board name changed to bdw_fgpa_v031

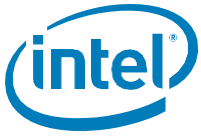


v1.0 (Dec 2016)

Changes:

- Moved to ACDS 16.0.2
- Higher memory bandwidth, as a result of:
  - Support for multi-cacheline transactions over CCI
  - BSP logic running at higher clock speed
- New floorplan, more FPGA space available for the kernel
- Clean-up of MMD error messages
- Board name changed to bdw_fpga_v1.0

§



3 Software Requirements

It is recommended that you keep the development host separate from the Multi-Core Processor (MCP) host. If you choose to use the MCP host as your development host as well, do not install the Runtime Environment (RTE) because you already have the Intel® SDK for OpenCL™ on your machine.

3.1 Development host separate from the MCP host (Recommended)

Development Host

- Intel® SDK for OpenCL™ with update and patches
- OpenCL™ Board Support Package (BSP) v1.0

MCP

- Altera RTE for OpenCL™ Version 16.0
- OpenCL™ BSP v1.0
- SR-5.0.2 (AALSDK and AALKERNEL)

Note: Further details about how to install the software when the Development host is separate from the MCP host may be found in the next chapter.

Note: Further details about how to update the Intel® SDK for OpenCL™ v16.0 to v16.0.2 and how to apply the patches may be found in the next chapter.

Note: Patches are distributed as a part of the OpenCL™ BSP v1.0 release

3.2 MCP as development host

MCP

- Intel® SDK for OpenCL™ with update and patches
- OpenCL™ BSP v1.0
- SR-5.0.2 (AALSDK and AALKERNEL)

Note: Further details about how to install the software when the MCP is used as a development host as well may be found in the next chapter.

Note: Further details about how to update the Intel® SDK for OpenCL™ v16.0 to v16.0.2 and how to apply the patches may be found in the next chapter.

Note: Patches are distributed as a part of the OpenCL™ BSP v1.0 release

§

4 Intel® SDK for OpenCL™ Installation

4.1 Setting up the Development Host Software

The following instructions describe the installation of both the Intel® SDK for OpenCL™ v16.0.2 and BSP v1.0. Prior to downloading the software, refer to the README.txt files in order to verify the software versions are compatible with the Board Support Package (BSP).

Note: It is recommended that you keep the development host separate from the MCP host. If you choose to use the MCP host as your development host as well, do not install the RTE because you already have the Intel® SDK for OpenCL™ on your machine.

4.1.1 Installing Intel® SDK for OpenCL™ and MCP Licensing

The latest installation files for Intel® SDK for OpenCL™ (includes Quartus Prime software and devices) may be found at http://dl.altera.com/opencl/

The steps to request and install a Multi-Chip Package (MCP) license may be found at https://cloud.altera.com/bdw_fpga/

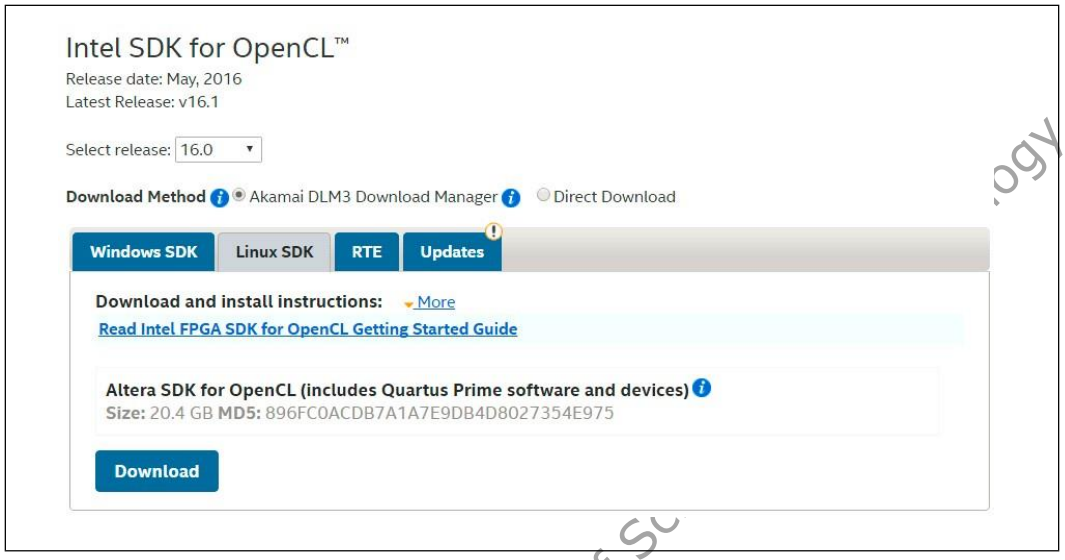
Note: The Intel® SDK for OpenCL™ is based on a published Khronos* Specification, and has passed the Khronos Conformance Testing Process. Current conformance status can be found at www.khronos.org/conformance.

Note: OpenCL™, and the OpenCL™ logo are trademarks of Apple* used by permission of the Khronos Group.

4.1.2 Installing the Intel® SDK for OpenCL™ v16.0 (Linux*)

1. Go to http://dl.altera.com/opencl/
2. Select the **16.0** release.
3. Click the **Linux SDK** tab.
4. Click the **Download** button to download the Intel SDK for OpenCL™ (Quartus Prime and Device support included) - 20.4GB (.tar).
5. Unpack the **AOCL.16.0.0.211-linux.tar** setup file and follow the instructions in the installer

Figure 4-1. Intel® SDK for OpenCL™ Download Page

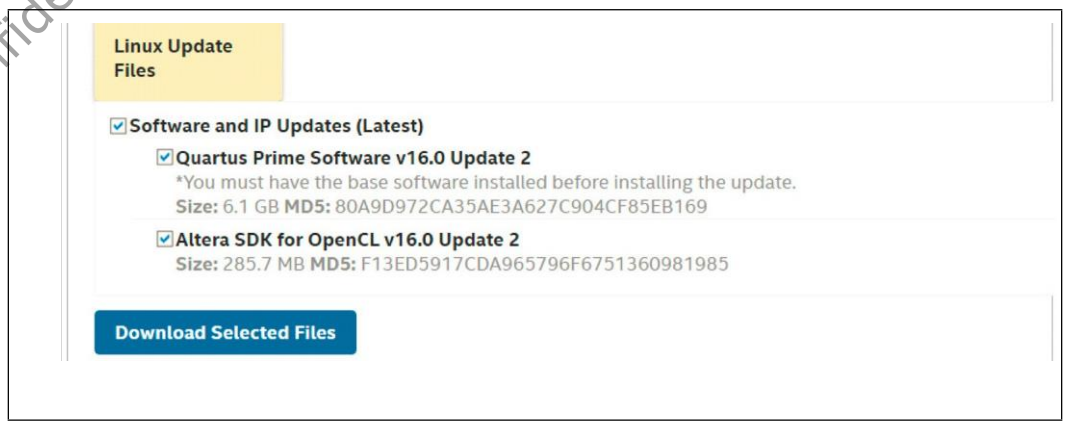


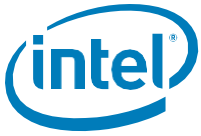
4.1.3 Updating the Intel® SDK for OpenCL™ to v16.0.2 (Linux*)

1. Go to http://dl.altera.com/opencl/
2. Select the **16.0** release.
3. Click the **Updates** tab.
4. Scroll down, check both the **Quartus Prime Software v16.0 Update 2** - 6.1GB (.run) and the **Altera SDK for OpenCL™ v16.0 Update 2** - 285.7MB (.run), and click the **Download Selected Files** button to download the Linux Update Files.
5. After download the .run files, change file permission by running the command: chmod +x *.run
6. Run the **QuartusSetup-16.0.2.222-linux.run** file to begin installation
7. Run the **AOCLSetup-16.0.2.222-linux.run** file to begin installation

Note: During installation, please specify installation to update: <home directory>/altera_pro/16.0

Figure 4-2. Intel® SDK for OpenCL™ Update Download Page





4.1.4 Installing the Intel® SDK for OpenCL™ v16.0.2 Patches (Linux*)

The patches, **quartus-pro-16.0.2-patch-2.06-linux.tgz** and **opencl-16.0.2-patch-2.01cl-linux.tgz**,are distributed as a part of the BSP v1.0 release.

1. Unpack the **quartus-pro-16.0.2-patch-2.06-linux.tgz** file
2. Change file permission by running the command: chmod +x *.run
3. Run the **quartus-16.0.2-2.06-linux.run** file to perform the installation

Note: You must either have previously installed the Quartus Prime 16.0.2 software or must install the Quartus Prime 16.0.2 software before installing this patch. Otherwise, the patch will not be installed correctly and the Quartus Prime software will not run properly.

1. Unpack the **opencl-16.0.2-patch-2.01cl-linux.tgz** file
2. Change file permission by running the command: chmod +x *.run
3. Run the **opencl-16.0.2-patch-2.01cl-linux.run** file to perform the installation

Note: You must either have previously installed the Intel® SDK for OpenCL™ 16.0.2 software or must install the Intel® SDK for OpenCL™ 16.0.2 software before installing this patch. Otherwise, the patch will not be installed correctly and the Intel® SDK for OpenCL™ software will not run properly.

4.2 Setting up the MCP (Intel® Xeon® +FPGA) Host Software

The following instructions describe the installation of version 16.0 of RTE software and the BSP for OpenCL™. Prior to downloading the software, refer to the README.txt files in order to verify the software versions are compatible with the BSP.

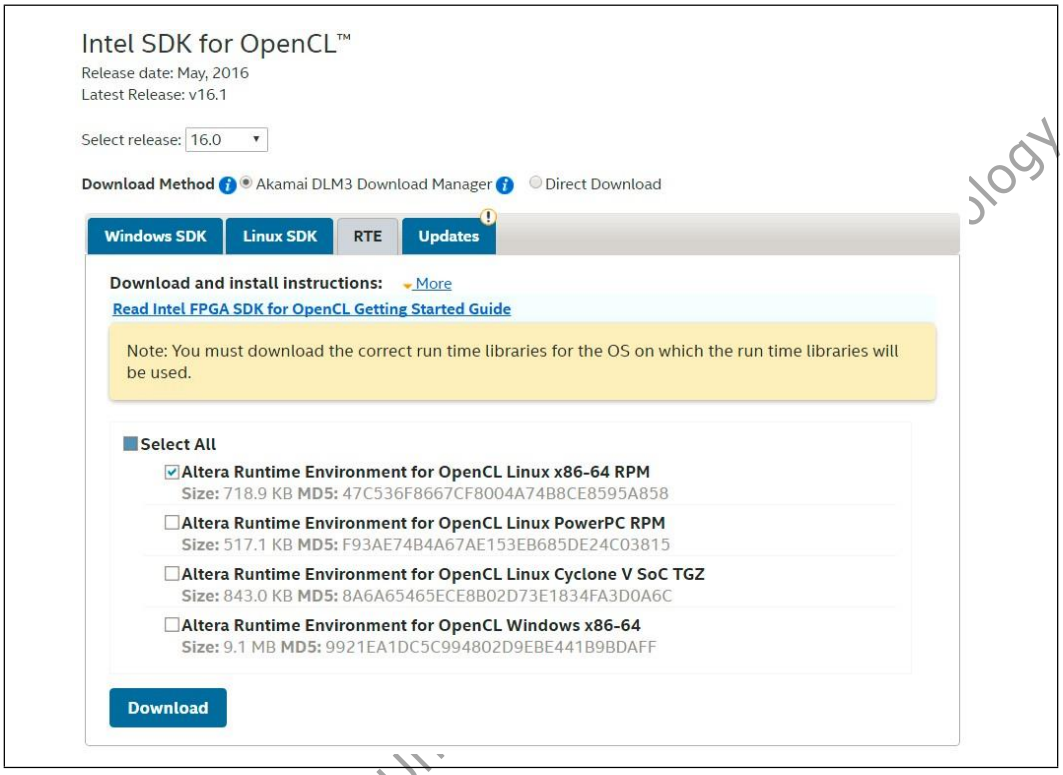
Note: It is recommended that you keep the development host separate from the MCP host. If you choose to use the MCP host as your development host, do not install the RTE because you already have the Intel® SDK for Open CL™ on your machine.

4.2.1 Installing the Altera RTE for OpenCL™ Version 16.0 (Linux*)

The RTE is delivered in RPM format. However, the SDP host (Xeon+FPGA Server) runs on Ubuntu so you will need to convert the RPM to Debian format or install the Intel FPGA SDK for OpenCL™ in the MCP host instead of the RTE.

1. Go to http://dl.altera.com/opencl/
2. Select the **16.0** release.
3. Click the **RTE** tab.
4. Select **Altera Runtime Environment for OpenCL™ Linux x86-64 RPM** file.
5. Click **Download**.
6. Install the downloaded RPM

Figure 4-3. Altera RTE for OpenCL™ Download Page



4.2.2 Installing AALSDK and AALKERNEL (SR-5.0.2)

The Intel SDP will have the AALSDK and AALKERNEL already installed and located in the **/opt/aalsdk/** directory. Make sure that both AALSDK and AALKERNEL are SR-5.0.2. The latest version of the AAL, SR-5.0.3, is not compatible with BSP v1.0.

Note: For detailed installation instructions refer to the following IBL doc #568255 and doc #568227.

Note: SR-5.0.2 is available as VIP Kit #115218

4.3 Installing the Intel® Xeon®+FPGA MCP BSP

1. Invoke the following command to unpack the BSP on both the Development Host and the MCP Host. In our case the MCP Host will be the Xeon+FPGA SDP.

```
$ tar -xvzf bdw_fpga_pilot_opencl_v1.0_part1.tar.gz

$ tar -xvzf bdw_fpga_pilot_opencl_v1.0_part2.tar.gz
```

2. Set the environment variable AOCL_BOARD_PACKAGE_ROOT to point to the location of the **board_env.xml** file within the BSP

```
$ export AOCL_BOARD_PACKAGE_ROOT=<path to the BSP contents>
```

4.4 Configuring the MCP Host to Load the FPGA at Startup (Initialization)

The **fpga1.rbf** that is currently provided for the MCP will not work for this version of the BSP. The **fpga2.rbf** that comes with the Intel® Xeon® +FPGA BSP must be used. The image is located in **AOCL_BOARD_PACKAGE_ROOT/images** directory, where the AOCL_BOARD_PACKAGE_ROOT points to the location of the **board_env.xml** file within the BSP.

1. Invoke the following command to locate the internal USB flash drive:

```
$ ls -l /dev/disk/by-id
```

2. Identify the device **usb-Verbatim_Store_n_Go_Drive** in the output, similar to the example below:

Figure 4-4. USB Disk ID

3. Mount the USB drive

```
$ sudo mount /dev/disk/by-id/
usbVerbatim_Store_n_Go_Drive_15062499003464-0:0-part1 /mnt
```

4. Invoke the following commands to go to the **/mnt** directory and check the name of the <name>.rbf that is already loaded (for example, **fpga1.rbf**).

```
$ cd /mnt

$ ls

fpga1.rbf
```

5. If the name does not match the loaded image, rename the **.rbf** to match the name in the mount directory and then copy the **.rbf** to the **/mnt** directory.

```
$ cd $AOCL_BOARD_PACKAGE_ROOT/images/

//If name is different then rename, otherwise not needed.

$ mv fpga2.rbf fpga1.rbf

$ sudo cp fpga1.rbf /mnt/fpga1.rbf
```

6. Once completed power cycle the SDP.
7. After booting up the SDP check that the bitstream is loaded by invoking the following command and check to see the below output:

```
$ lspci |grep bcbc

80:10.0 System peripheral: Intel Corporation Device bcbc
```

§

5 Compile a Design Example

Compile an OpenCL™ design example and try running it on the MCP. Read the steps in the Intel® SDK for OpenCL™ Getting Started Guide to learn more about the SDK for OpenCL™ programming Flow: https://www.altera.com/en_US/pdfs/literature/hb/openclsdk/aocl_getting_started.pdf

5.1 Example Description (mem_bandwidth)

This example runs a basic OpenCL™ kernel that tests the bandwidth of a memory to memory copy, as well as read and write only.

5.2 Compiling a Design on the Development Host

1. Verify that the environment variable ALTERAOCLSDKROOT points to the location of the Intel® SDK for OpenCL™ installation. Set the variable accordingly if needed.

   ```
   $ export ALTERAOCLSDKROOT=<path>
   ```

2. Verify that the environment variable **AOCL_BOARD_PACKAGE_ROOT** points to the location of the board_env.xml file within the BSP. Set the variable accordingly if needed.

   ```
   $ export AOCL_BOARD_PACKAGE_ROOT=<path to the BSP contents>
   ```

3. Invoke the following commands to run the init_opencl.sh script:

   ```
   $ source $ALTERAOCLSDKROOT/init_opencl.sh
   ```

4. Navigate to the design example directory and compile the kernel. Prior to specifying <board_name> check the BSP directory for the correct name (for example, "bdw_fpga").

   ```
   $ cd AOCL_BOARD_PACKAGE_ROOT/example_design/mem_bandwidth

   $ aoc device/mem_bandwidth.cl -o bin/mem_bandwidth.aocx --board <board>
   ```

where <board> matches the board you have in your system. If you are unsure of the board name, use the following command to list available boards:

```
$ aoc --list-boards
```

This compilation command can also be used to target the emulator by adding the expression `-march=emulator` flag.

If the board already has a AOCX file, be sure to either replace or relocate that AOCX file.

5.2.1 Running a Design on the Intel® Xeon®+FPGA MCP Host

1. Invoke the following commands to set all the variables correctly. It is recommended that you create an **init.sh** script to set the variables.

```
//Contents of Init.sh

export ALTERAOCLSDKROOT=<OpenCL RTE Root>

export AALSDK=<Path to AALSDK>

export AOCL_BOARD_PACKAGE_ROOT=<path to the BSP contents>

export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$ALTERAOCLSDKROOT/
host/linux64/lib:$AOCL_BOARD_PACKAGE_ROOT/host/linux64/lib

export PATH=$PATH:$ALTERAOCLSDKROOT/bin

export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/user/local/lib

$ export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$AALSDK/lib
```

2. Invoke the following command to run the **init.sh** file

```
$ source init.sh
```

3. Navigate to the design example directory and then type make to compile the host program.

```
$ cd <download_path>/example_design/mem_bandwidth

$ make
```

4. Copy the generated **mem_bandwidth.aocx** file from the development host (that is where the kernel was compiled) to the mem_bandwidth/bin directory of the MCP host

```
$ cp mem_bandwidth.aocx /bin
```

5. Invoke the following commands to install the driver.

```
$ cd <package dir>/aalkernel-5.0.2/mybuild

$ sudo ./insdrv cci
```

6. Run the executable

```
$ cd /bin

$ ./mem_bandwidth <optional number of 64 byte lines to
transfer>
```

The output will include a wall-clock time of the OpenCL™ execution time and the kernel time as reported by the OpenCL™ event profiling API, as well as the calculated bandwidth. The host program includes verification against the host CPU.

§