OpenCL™ Runtime 14.1.1 for Intel® Core™ and Xeon® Processors, and Intel® Xeon Phi™ Coprocessors

What's New

In this release (14.1):

- Support for OpenCL Standard Portable Intermediate Representation (SPIR) 1.2 consumption.
- Intel® Manycore Platform Software Stack (Intel® MPSS) 3.2 is required.

NOTE: Using OpenCL Runtime 14.1 with MPSS 3.2.1 is not recommended, as this combination introduces stability issues.

- Performance improvements:
  - Faster execution of code dominated by statically diverging dynamically uniform branches
  - More efficient event traversing algorithm
  - `NO_DMA` mode is default, which improves buffer creation speed (not a preview feature anymore)
  - Improved device side memory pool control
- CPU only: Starting with this release, kernel binary is the very final machine code. This enables creating the kernel binary offline and distributing it with the application machine code binary. This also eliminates the compilation time at the end-use product (`clCreateProgramWithBinary`)
- Bug fixed (for Intel® Xeon Phi™ coprocessors only): Compilation crash when a struct is defined globally in the CL file.
- New performance-related environment variables on Intel Xeon Phi —see the user guide for details
  - `CL_CONFIG_MIC_DEVICE_FORCE_BUFFERS_PINNING_ON_HOST`
  - `CL_CONFIG_MIC_DEVICE_2MB_POOL_FINI_SIZE_MB`
  - `CL_CONFIG_MIC_DEVICE_2MB_POOL_INIT_SIZE_MB`

- Added 32-bit version of the runtime for Windows OS.
- Added OpenCL CPU device support on Intel Core™ processors.

System Requirements

Supported Hardware

This runtime is tested on the following processors:

- Intel® Xeon Phi™ coprocessor Product Family
- Intel Xeon® Processor E5 Product Family
- Intel Xeon Processor E7 Product Family

The OpenCL Runtime 14.1 provides only OpenCL CPU device support on the following processors:

- Intel® Core™ Processors
- Intel Xeon Processor E3 Family
- Intel® Celeron® Processor J1000/N2000 Series and
  Intel Pentium Processor J2000/N3000 Series
- Intel® Atom™ Processor Family code named Bay Trail

To enable GPU device support on the aforementioned processors, install the Intel Graphics driver.

NOTE: Incompatible or proprietary instructions in non-Intel processors may cause the analysis capabilities of this product to function incorrectly. Any attempt to analyze code not supported by Intel® processors may lead to failures in this product.

Supported Operating Systems

The following is the list of supported operating systems:

Linux* Operating Systems:

- Red Hat Enterprise Linux* OS 6.1 or higher (64-bit version)
- SUSE Linux Enterprise Server* 11.2 or higher (64-bit version)

Windows* Operating Systems (32- and 64-bit):

- Microsoft Windows* 7 SP1
- Microsoft Windows 8
- Microsoft Windows Server 2008 R2
- Microsoft Windows Server 2012

Due to possible Intel® AVX issues with the default `glibc` 2.11.1 implementation, the product libraries might require `glibc-2.12-1.47` or higher. Refer to the OS documentation for more information.

Installation Notes

Installation on Microsoft Windows* OS

To install the OpenCL™ Runtime 14.1 on Windows operating systems, download and install the Runtime package and follow the installer prompts.

To remove the OpenCL Runtime 14.1, use the **Control Panel** > **Programs and Features** > **OpenCL™ Runtime 14.1** > **Uninstall**.

The uninstaller removes all originally installed files, leaving any temporary or newly created files. To ensure a clean uninstallation, verify that the `INTELOCLSDKROOT`, `INTELOCLSAMPLESROOT`, and `PATH` environment variables are in their preinstall state.

NOTE: For Intel Xeon Phi coprocessor device support, you must install the 3.2.1 version of Intel MPSS available at http://software.intel.com/en-us/articles/intel-manycore-platform-software-stack-mpss-archive

Installation in on RedHat* and SUSE* Linux Enterprise OS

All RPM packages of the OpenCL Runtime 14.1 are digitally signed.

Verify signature of each RPM package after downloading.

To verify signatures, do the following:

1. Download the public key from the download page of the product.
2. Import public key into rpm database by use of the following command:
   ```
   # sudo rpm --import Intel-E901-172E-EF96-900F-B8E1-4184-D7BE-0E73-F789-186F.pub
   ```
3. Verify signature of RPM package:
   ```
   # rpm --checksig <rpm name>.rpm
   ```
   where `<rpm name>` is the name of the RPM package.
   Expected output for RPM with a valid digital signature is:
   ```
   <rpm name>.rpm: rsa sha1 (md5) pgp md5 OK
   ```

NOTE: For Intel Xeon Phi coprocessor device support, you must install the 3.2.1 version of Intel MPSS available at http://software.intel.com/en-us/articles/intel-manycore-platform-software-stack-mpss-archive

Installing the Product Using RPM Package Manager

To install the OpenCL Runtime for CPU on Linux* OS, download and unpack the `opencl_runtime_14.1_x64_4.4.0.117.tgz` package, and run the following commands:

On Red Hat Enterprise Linux* OS:

```
# sudo yum install *base*.rpm *intel-cpu*.rpm
```

On SUSE* Linux Enterprise Server OS:

```
# sudo zypper install *base*.rpm *intel-cpu*.rpm
```

To install the CPU and Intel Xeon Phi coprocessor runtime, run following commands:

On Red Hat Enterprise Linux OS:

```
# sudo yum install *base*.rpm *intel-cpu*.rpm *intel-mic*.rpm
```

On SUSE Linux Enterprise Server OS:

```
# sudo zypper install *base*.rpm *intel-cpu*.rpm *intel-mic*.rpm
```

NOTE: If the CPU-only runtime is already installed, the installation of the CPU and the Intel Xeon Phi coprocessor runtime adds the Intel Xeon Phi coprocessor support without removing the CPU runtime.

NOTE: If you encounter a message saying that `libcoi_host.so.0()(64bit)` or `libcoi_host.so.0(COI_1.0)(64bit)` is needed, install the Intel MPSS.

Installing the Product Using Shell Scripts

To install only the CPU runtime, run the following command:

```
# sudo ./install-cpu.sh
```
To install the CPU and Intel Xeon Phi coprocessor runtime, run the following command:

```
# sudo ./install-cpu+mic.sh
```

Installing the CPU and Intel Xeon Phi coprocessor runtime using scripts without prior to uninstallation of the CPU-only runtime is not supported.

Uninstalling Intel SDK for OpenCL™ Applications from RedHat* and SUSE Linux Enterprise* OS

To uninstall the product using the uninstallation script, do the following:

1. Go to the folder to which you extracted the TGZ archive content.
2. Run the `uninstall.sh` script.

You can use the OS-specific command to remove all the packages, starting with "`opencl-1.2-`". To do so, run the following commands:

For Red Hat Enterprise Linux OS:

```
# sudo yum remove "opencl-1.2-*"
```

For SUSE Linux Enterprise OS:

```
# sudo zypper remove "opencl-1.2-*"
```

Known Installation and Configuration Issues

OpenCL™ Runtime package installer adds the target installation folder of the CPU runtime to the end of the system `PATH` environment variable. If the variable is too long, the application might not be able to load the CPU runtime `DLL` files. To solve the problem, move the folders to the beginning of the `PATH` variable or delete unnecessary folders from the `PATH`.

Known Issues on Intel Xeon Phi Coprocessors

- Device transfer from and to non-64B aligned host `ptr`, results in low throughput.
- `clEnqueueCopyBuffer` is suboptimal in case the source and the target buffers are already located on the same Intel Xeon Phi coprocessor device. In this case, doing the same with an optimized kernel might provide performance improvements.
- `clEnqueueCopyBufferRec` is suboptimal on Intel Xeon Phi coprocessor in the cases, with small buffer regions, higher and narrow regions (many rows and few columns). In these cases, you may like to copy the entire source buffer or write your own optimized kernel for this operation.
- JIT (kernel) profiling works with MPSS 3.2 and with update 16 and later versions of Intel VTune Amplifier XE 2013.
- Intel VTune Amplifier XE 2013 source-view: Function call stack might contain more call instances than expected, for example: a function was only called once but it appears as if it was called more than once. To mitigate this issue, use the "Source Function Stack" grouping in the top-down view.
- No support for OpenCL images object type.
- No support for sub devices (also known as device fission).
- 2MB page size support – OpenCL runtime allocates buffers on 2MB pages heap automatically, whenever the allocated size is greater than one KB. You can modify this threshold be setting the following environment variable with your preference number of KBs: `CL_CONFIG_MIC_DEVICE_2MB_BUF_MINSIZE_`KB. Setting this variable to zero, disables 2MB page allocation.
- `clEnqueueFillBuffer` crashes on device side when used in out-of-order queue.

NOTE:	Intel Xeon Phi coprocessor support is available with 64-bit version of the runtime. 32-bit package includes CPU runtime only.

Known Issues on Intel CPUs

- OpenCL Runtime requires Intel® Threading Building Blocks (Intel® TBB) version 4.2.1, which is included in the OpenCL Runtime package installation folder. Make sure there is no Intel® TBB version conflict in your system upon runtime installation.
  When an OpenCL application is invoked, the **OpenCL** path should be the first path in `LD_LIBRARY_PATH`.
  For example:
  CSH
  `setenv LD_LIBRARY_PATH ${OCL_INSTALL_DIR}\bin:${LD_LIBRARY_PATH}`

  BASH

  `export LD_LIBRARY_PATH=${OCL_INSTALL_DIR}\bin:${LD_LIBRARY_PATH}`

- Device fission extension is not supported. Only device fission core feature is supported.
- Known issues with Intel® VTune™ Amplifier XE 2013 source-view support:
  - Source-level profiling does not work properly when functions are included (using `#include`) from other files.
  - Note that due to function inlining, the profiling data is aggregated into the top-level kernel.
  - Source information for code built with "`-g`" does not appear on kernels, only on user functions.
- The current TBB version used with OpenCL/CPU is 4.2.1 (4.2 update 1).
  - Any standalone TBB package loaded by the OpenCL host-code should be of higher version that the OpenCL/TBB version.
  - The standalone TBB package must use the default TBB configuration, which is also used by the OpenCL runtime.
  - Make sure you use and load the right TBB libraries. For example, if you plan to use new features on a standalone TBB version higher than OpenCL, ensure that the corresponding standalone TBB libraries are correctly loaded (`LD_LIBRARY_PATH` in Linux or `PATH` in Windows are correct).