OpenCL™ and Microsoft DirectX* Video Acceleration Surface Sharing

Sample User's Guide

OpenCL™ Code Builder - Samples

Contents

Legal Information

By using this document, in addition to any agreements you have with Intel, you accept the terms set forth below.

You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Intel products described herein. You agree to grant Intel a non-exclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein.

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

A "Mission Critical Application" is any application in which failure of the Intel Product could result, directly or indirectly, in personal injury or death. SHOULD YOU PURCHASE OR USE INTEL'S PRODUCTS FOR ANY SUCH MISSION CRITICAL APPLICATION, YOU SHALL INDEMNIFY AND HOLD INTEL AND ITS SUBSIDIARIES, SUBCONTRACTORS AND AFFILIATES, AND THE DIRECTORS, OFFICERS, AND EMPLOYEES OF EACH, HARMLESS AGAINST ALL CLAIMS COSTS, DAMAGES, AND EXPENSES AND REASONABLE ATTORNEYS' FEES ARISING OUT OF, DIRECTLY OR INDIRECTLY, ANY CLAIM OF PRODUCT LIABILITY, PERSONAL INJURY, OR DEATH ARISING IN ANY WAY OUT OF SUCH MISSION CRITICAL APPLICATION, WHETHER OR NOT INTEL OR ITS SUBCONTRACTOR WAS NEGLIGENT IN THE DESIGN, MANUFACTURE, OR WARNING OF THE INTEL PRODUCT OR ANY OF ITS PARTS.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined". Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or go to:

http://www.intel.com/design/literature.htm.

Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families. Go to: http://www.intel.com/products/processor_number/.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

Intel, Intel logo, Intel Core, VTune, Xeon are trademarks of Intel Corporation in the U.S. and other countries.

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

OpenCL and the OpenCL logo are trademarks of Apple Inc. used by permission from Khronos.

Microsoft product screen shot(s) reprinted with permission from Microsoft Corporation.

Optimization Notice

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804

About OpenCL™ and Microsoft DirectX* Video Acceleration Surface Sharing

The OpenCL™ and Microsoft DirectX* Video Acceleration (DXVA) Surface Sharing sample demonstrates how to use Microsoft DXVA and OpenCL Code Builder together for video post processing and real-time rendering.

Software Prerequisites

The following software is required to successfully build the code sample:

- OpenCL™ Code Builder, available with Intel® INDE
- Microsoft DirectX SDK, refer to http://msdn.microsoft.com/en-us/library/windows/desktop/ee663275(v=vs.85).aspx
- Microsoft Visual Studio 2010 and higher

Introduction

Microsoft DXVA provides a hardware-accelerated environment used in video decoding and rendering applications. The OpenCL and Microsoft DirectX Video Acceleration (DXVA) Surface Sharing sample uses OpenCL for video post processing and DXVA to render the video in real-time.

Motivation

The OpenCL and Microsoft DirectX Video Acceleration (DXVA) Surface Sharing sample demonstrates an OpenCL implementation of video post processing on a shared DXVA surface, showing how to:

- Share a DXVA surface with OpenCL
- Perform basic image post processing filters

Algorithm

The algorithm consists of the following stages:

1. Create a simulated video frame and a shared DXVA surface.
2. Use OpenCL to write simulated video frame into DXVA surface; convert from BGRA source frame to `NV12` format.
3. Use OpenCL to apply selected video filters on shared DXVA surface.
4. Render the frame to screen via DXVA.

Implementation Details

The OpenCL and Microsoft DirectX Video Acceleration (DXVA) Surface Sharing sample demonstrates creating a shared DXVA surface and then using OpenCL to perform video post processing on a simulated video frame before rendering to the screen with DXVA. It uses the `cl_khr_dx9_media_sharing` extension to avoid extra memory copying from the DXVA frame into Intel SDK for OpenCL Applications memory object.

OpenCL™ Implementation

The first part of the application reads from a BMP image which is used as a simulated video frame. This image is in BGRA format and is passed to an OpenCL™ kernel along with a shared DXVA surface. This kernel does a conversion from BGRA to NV12 image format and writes the results to the DXVA surface. The next stage of the application runs any image filter kernels that have been set to **Enabled**. The included image filters are invert, which inverts the image colors and Gaussian which applies a simple Gaussian blur to the image. After the image filter kernels are finished executing the DXVA surface is released back to DXVA and it is then rendered to the screen.

- `simple_write_shared.cl` – Accepts a buffer which is assumed 32bit BGRA image. As well as the Y and UV planes of a NV12 format shared DXVA surface. Kernel reads a pixel from buffer, then converts it from RGB to YUV format and then writes the resultant values out to the Y and UV surfaces.
- `simple_invert_shared.cl` – Accepts a pair of 2 image2D planes as read only for Y and UV planes, also takes 2 image2D planes as write only for output Y and UV planes. This kernel reads in the pixel values, inverts them via subtracting the pixel values from 1.0, and then writing out the result.
- `simple_gauss_shared.cl` – Accepts a pair of 2 image2D planes as read only for Y and UV planes, also takes 2 image2D planes as write only for output Y and UV planes. This kernel implements a simple version of a Gaussian image blur filter.

Controlling the Sample

The sample can be controlled in the following ways

- F1: enables/disables invert filter
- F2: enables/disables Gaussian filter
- F7: Enables/Disables simple frames per second calculation

References

OpenCL™ 1.2 Specification at http://www.khronos.org/registry/cl/specs/opencl-1.2.pdf