# Intel® Media SDK
# RAW Media Accelerator Sample



## Overview

The **Intel® Media SDK RAW Media Accelerator Sample** demonstrates how to use the Intel® Media SDK RAW Media Accelerator to create a simple console application that performs processing of RAW data in Bayer formats from camera and outputs data in monitor compatible formats.

## Features

The Intel Media SDK RAW Media Accelerator Sample supports the following video formats:

| | |
|---|---|
| input (uncompressed) | 16-bit BGGR, RGGB, GBRG, GRBG Bayer formats |
| output (uncompressed) | ARGB32, ARGB64 |

The sample supports the following image processing algorithms: demosaicing (always on), and forward gamma correction (switchable). The input image should be 8-pixel padded to be correctly processed by Intel® Media SDK RAW Media Accelerator pipeline. The sample application can either do the padding itself (-padding command line option), or pass the unpadded image to the Accelerator, in which case the padding is performed inside the Accelerator pipeline.

## Software requirements

See `<install-folder>\Media SDK Sample Guide.pdf`

- Microsoft* Windows* 7, Microsoft Windows 8.1.
- For Microsoft DirectX* 11 functionality - Microsoft Windows 8.1.
- Microsoft Visual C++* 2012.
- Intel® Iris™ graphics and HD Graphics Driver for Windows 7/8/8.1 version 15.33.22.(64.)3621 or higher.

## Hardware requirements

See `<install-folder>\Media_Samples_Guide.pdf`

- Preferable: 4th Generation Intel® Core™ processors with Intel® Iris™ Pro graphics 5200 and Solid State Drive storage.
- Hardware acceleration is available on platforms with:
    - 4th Generation Intel® Core™ processors,
    - Selected SKUs of Intel® Celeron™ and Intel® Pentium™ processors with Intel HD Graphics which support Intel® Quick Sync Video.
- On other supported platforms only software fallback available:
    - 3rd Generation Intel® Core™ processors,
    - Selected SKUs of Intel® Celeron™ and Intel® Pentium™ processors with Intel HD Graphics which support Intel® Quick Sync Video.
    - Selected SKUs of Intel® Atom™ processors with Intel® HD Graphics which support Intel Quick Sync Video.

## How to Build the Application

See `<install-folder>\Media_Samples_Guide.pdf`

For building sample_camera an additional file named `mfxcamera.h` is required:

You may find it in **Intel® Media SDK RAW Media Accelerator** package.

You may set INTELMEDIASDKROOT variable to the location of **Intel® Media SDK RAW Media Accelerator**, no additional actions are required in this case. But if INTELMEDIASDKROOT should is set to another location, you may need to copy mfxcamera.h to `sample_camera\include folder` from **Intel® Media SDK RAW Media Accelerator** package.

## Running the Software

The executable is to be run from the folder where the Intel RAW Media Accelerator plugin is located.

The executable file `sample_camera.exe` expects the following command-line arguments for proper function:

| | |
|---|---|
| `-i <InputFile>` | Input RAW video file path and name base: the sample will consecutively look for files named <InputFile>00000000.<ext>, <InputFile>00000001.<ext>, etc., where <ext> is bg16 (bggr), rg16 (rggb), gb16 (gbrg), or |

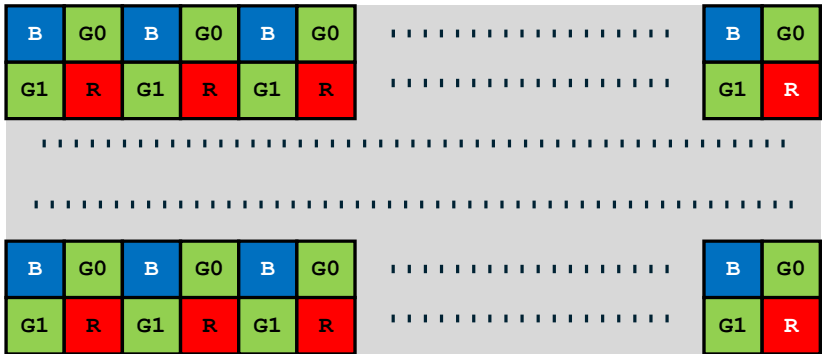
| | |
|---|---|
| | gr16 (grbg), depending on the -format argument. |
| -o <OutputFile> <num> | Output video file path and name base: the sample will consecutively dump output files named <OutputFile>0.<ext>, <OutputFile>1.<ext>, etc., where <ext> is either bmp of argb16 depending on the -outFormat argument.<br><num> - number of files to dump, 20 by default |
| -plugin_verision <ver> | Define Intel® Media SDK RAW Media Accelerator Plug-in version |
| -asyncDepth <asyncDepth><br>-a <asyncDepth> | Set async depth for the pipeline, default 4 |
| -w <Width> | Width of input Bayer and output images in samples |
| -h <Height> | Height of input Bayer and output images in lines |
| -format <BayerFormat><br>-f <BayerFormat> | Input Bayer format: bggr, rggb, gbrg, grbg |
| -outFormat <OutputFormat><br>-of <OutputFormat> | Output image format, argb16 or 16 meaning 16-bit ARGB (ARGB64), ARGB32 otherwise, NV12 |
| -n <num><br>-numFramesToProcess <num> | Number of frames to process |
| -r<br>-render | render output in a separate window |
| -imem | Input memory type (sys\|video). Default is system. |
| -omem | Output memory type (sys\|video). Default is system. |
| -accel | Type of acceleration device (d3d9\|d3d11). Default is d3d9. |
| -bitDepth <bitDepth><br>-b <bitDepth> | Bit depth of the input images, default 10 |
| -pd<br>-padding | The source is padded by the sample application – no padding to be done by Accelerator pipeline |
| resetInterval <resetInt> | Reset every <resetInt> frames, default 7 |
| -reset –i … -o … -f … -w… -h… | Parameters to be used after next reset. Only these 5 parameters are supported, if a parameter is not set here, the originally set value is used. There can be any number of resets, applied in order of appearance in the command line, after <resetInt> frames are processed with the current parameters. |
| -ng | No gamma correction to be done |

| | |
|---|---|
| -noGamma | |
| -gamma_points | Set specific gamma points (64 points expected) |
| -gamma_corrected | Set specific gamma corrected values (64 values expected) |
| -bbl B G0 G1 R<br>-bayerBlackLevel B G0 G1 R | Bayer black level correction coefficients |
| -bwb B G0 G1 R<br>-bayerWhiteBalance B G0 G1 R | Bayer white balance correction coefficients |
| -tcc R G B C M Y<br>-totalColorControl R G B C M Y | Total color control |
| -ccm n00 n01 ... n33 | Color correction 3x3 matrix coefficients |
| -lens a b c d | Enable lens geometry distortion correction |
| -vignette maskfile | Enable vignette correction using mask from specified file |
| -chroma_aberration aR bR cR dR aG bG cG dG aB bB cB dB | Enable chroma aberration correction |
| -alpha alpha | Write value to alpha channel of output surface |
| -perf_opt | Buffered reading of input (support: Bayer sequence) |
| -r<br>-render | Render output in a separate window |
| -wall w h n m f t tmo | Same as -r, and positioned rendering window in a particular cell on specific monitor<br>w - number of columns of video windows on selected monitor<br>h - number of rows of video windows on selected monitor<br>n(0,.,w*h-1) - order of video window in table that will be rendered<br>m(0,1..) - monitor id<br>f - rendering framerate<br>t(0/1) - enable/disable window's title<br>tmo - timeout for -wall option |

Below is an example of a command-line to execute the Intel® Media SDK RAW Media Accelerator Sample:

```
sample_camera –i c:\content\Film01\film01_ -w 4096 -h 2160 -f rggb –of 16 –d3d11 -r
–b 12 –a 3 -o out 50 –n 100
```

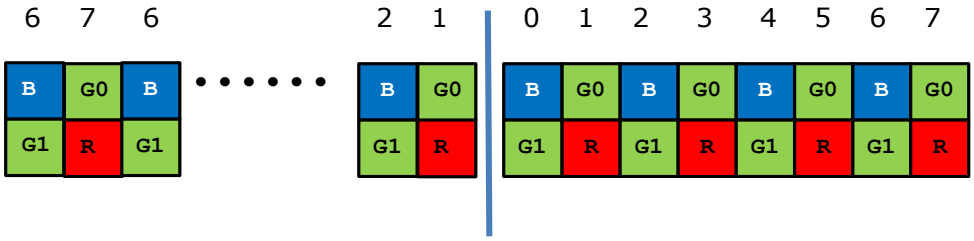
The structure of a Bayer image for bggr Bayer pattern is shown below.




The four Bayer patterns differ in the position of the colors in the 2x2 unit element:

bggr rggb gbrg grbg


The mirror padding on the left image boundary is demonstrated below:




The same procedure is applied to the right boundary, and after that the padding at the top and bottom boundaries is done in the same manner.

Note: the outermost (leftmost in the case shown above) columns/rows are only used by Bayer denoise algorithm which is going to be part of furute versions of the RAW Media Accelerator pipeline; for the current implementation, the boundary pixels can be set to arbitrary values.

Input bit depths of 8, 10, 12, and 14 are currently supported, the data should be shifted to the most significant bit (MSB) side.

If the output is 16-bit ARGB (ARGB64), the output bit depth is equal to the input bit depth, the output data being MSB shifted as well. By defining `CONVERT_TO_LSB` identifier in the sample source code, the sample can be configured to shift the output data to the least significant bit (LSB) side.

When the output is set to 8-bit ARGB (ARGB32), the final stage of the Accelerator pipeline converts (right shifts) the data to 8 bits. Rendering in the sample is only available with 8-bit output.

Sample performance measured on an Intel Core i7 4950HQ 2.4 GHz machine with Intel® Iris™ Pro graphics, running at 1300 MHz, Intel SSD drive, Windows 8.1:
4096x2160 unpadded Bayer input
27 fps pure pipeline
25 fps with rendering

## Known Limitations

- On some configurations rendering is observed slow down the sample overall performance significantly – by up to 10 fps, which can probably be improved by decoupling the rendering from the main accelerator pipeline.
- -p option is not in use any more – camera plugin UID is specified inside the sample's code

## Legal Information

**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 SSE3 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