Intel® Media Software Development Kit 2014 for Linux* Servers Release Notes

(Version 5.0.1603321.86136)

Overview

The **Intel® Media Software Development Kit for Linux* Servers** (Intel Media SDK) is a software development library that exposes the media acceleration capabilities of Intel platforms for decoding, encoding and video preprocessing. The library covers a wide range of Intel platforms. Intel Media SDK targets general application developers who want to integrate encoding and decoding into their applications.

Please see "`<install-folder>/doc/MediaSDK Sample Guide.pdf`" for information on sample source code provided with the Intel Media SDK.

New in This Release

New in 2014 version:

- Support for `mfxVideoCodecPlugin` API which is a new API extension for the user defined functions, specialized for Decode, Encode and VPP. It provides ease of use interfaces for integration of user-defined Decode, Encode and VPP functions into Intel Media SDK pipelines.
- Intra Refresh (aka "Rolling I frame") feature was enabled.
- Stability of multi-process encoding was improved.
- Stability of Look Ahead BRC on bottom field first (BFF) interlaced content was improved.
- VPP Composition:
  - `VPP::Reset` support for dynamic change of composition layout.
  - Limitation on the first/primary stream to be non-scalable was removed.

    - Limitation that only up to 8 input streams could be resized was removed.
    - **Preview** of Alpha Blending feature: luma key (with NV12 color format), global alpha (NV12 or RGB32) and per-pixel alpha (RGB32). **"Preview" status of a feature means that API is subject to change and implementation is of Beta quality.**
- New samples: Splitters and Muxers Sample and Full Transcoding Sample.

Detailed description of the new features can be found in Intel® Media SDK Reference Manual (`mediasdk-man.pdf`).

For information on the USER class please see
`"<install-folder>\doc\mediasdkusr-man.pdf"`

Features

Hardware accelerated Intel® Media SDK Library included in this package implements Intel Media SDK API 1.8 and contains the following components:

Component	Supported features	Limitations
H.264 decoder	Supported Profiles: • Baseline • Main • High	Maximum supported resolution is 4096x2304
H.264 encoder	Supported Profiles: • Baseline • Main • High Supported BRC methods: • Constant QP (CQP) • Constant Bit Rate (CBR) • Variable Bit Rate (VBR) • Look Ahead (LA)	Maximum supported resolution is 4096x2304
MPEG-2 decoder	Supported Profiles: • Simple • Main • High	Maximum supported resolution is 1920x1088
MPEG-2 encoder	Supported Profiles: • Simple • Main	Maximum supported resolution is 1920x1088

	• High Supported BRC methods: • Constant QP (CQP) • Constant Bit Rate (CBR) • Variable Bit Rate (VBR)	
VC1 decoder	Supported Profiles: • Simple • Main • Advanced	Maximum supported resolution is 1920x1088
Video Pre Processing (VPP)	Supported Algorithms: • Color Conversion • Scaling • De-Interlacing • De-noising • Frame Rate Conversion • Composition	Maximum supported resolution is 4096x2304

Common for all components: minimum supported resolution is 32x32, frame width must be a multiple of 16, frame height must be a multiple of 16 for progressive frames and a multiple of 32 otherwise.

Please see the Intel Media SDK Reference Manual for details "`<install-folder>/doc/mediasdk-man.pdf`"

Alpha blending (preview)

This feature has not yet been included in the official Intel Media SDK API and thus is not covered in the Intel Media SDK Reference Manual. Feature description is given below. There are 3 distinct methods to overlay one frame surface over another, they can be used separately or combined:

- **Luma key**. Can be used to overlay one NV12 surface over another. See `mfxVPPCompInputStream::LumaKeyEnable, LumaKeyMin,LumaKeyMax` fields below.
- **Global alpha**. Can be used to overlay either NV12 surface over NV12 surface, or RGB32 surface over RGB32 surface. See `mfxVPPCompInputStream:: AlphaEnable, Alpha` fields below.
- **Per-pixel alpha.** Can be used only with RGB32 surfaces. Set `mfxVPPCompInputStream::reserved2[0] = 1` to enable per-pixel alpha.

mfxExtVPPComposite

Definition

```
typedef struct mfxVPPCompInputStream {
        mfxU32  DstX;
        mfxU32  DstY;
        mfxU32  DstW;
        mfxU32  DstH;
        mfxU16  LumaKeyEnable;
        mfxU16  LumaKeyMin;
        mfxU16  LumaKeyMax;

        mfxU16  AlphaEnable;
        mfxU16  Alpha;

        mfxU16  reserved2[19];
} mfxVPPCompInputStream;

typedef struct {
    mfxExtBuffer    Header;

    /* background color*/
    union {
        mfxU16   Y;
        mfxU16   R;
    };
    union {
        mfxU16   U;
        mfxU16   G;
    };
    union {
        mfxU16   V;
        mfxU16   B;
    };

    mfxU16      reserved1[24];

    mfxU16      NumInputStream;
    mfxVPPCompInputStream *InputStream;
} mfxExtVPPComposite;
```

Description

The `mfxExtVPPComposite` structure is used to control composition of several input surfaces in the one output. In this mode, the VPP skips any other filters. The VPP returns error if any mandatory filter is specified and filter skipped warning for optional filter. The only supported filters are CSC (RGB, YUY2, NV12)->(RGB, NV12), deinterlacing and interlaced scaling.

The VPP returns **`MFX_ERR_MORE_DATA`** for additional input until an output is ready. When the output is ready, VPP returns **`MFX_ERR_NONE`**. The application must process the output frame after synchronization.

Composition process is controlled by:

- **mfxFrameInfo::CropXYWH** in input surface- defines location of picture in the input frame,
- **InputStream[i].DstXYWH** - defines location of the cropped input picture in the output frame,
- **mfxFrameInfo::CropXYWH** in output surface - defines actual part of output frame. All pixels in output frame outside this region will be filled by specified color.

If the application uses composition process on video streams with different frame sizes, the application should provide maximum frame size in **mfxVideoParam** during initialization, reset or query operations.

If the application uses composition process, **MFXVideoVPP_QueryIOSurf** function returns cumulative number of input surfaces, i.e. number required to process all input video streams. The function sets frame size in the **mfxFrameAllocRequest** equal to the size provided by application in the **mfxVideoParam**.

Composition process supports all types of surfaces, but opaque type has next limitations:

- all input surfaces should have the same size,
- all input surfaces should be described in one **mfxExtOpaqueSurfaceAlloc** structure.

Members

`Header.BufferId`

Must be **MFX_EXTBUFF_VPP_COMPOSITE**

`Y, U, V`

`R, G, B`

background color, may be changed dynamically through Reset. No default value. YUV black is (0;128;128) or (16;128;128) depending on the sample range. The SDK uses YUV or RGB triple depending on output color format.

`NumInputStream`

Number of input surfaces to compose one output. May be changed dynamically at runtime through Reset. Number of surfaces can be decreased or increased, but should not exceed number specified during initialization. Query mode 2 should be used to find maximum supported number.

`InputStream`

This array of mfxVPPCompInputStream structures describes composition of input video streams. It should consist of exactly `NumInputStream` elements.

`DstX,`
`DstY,`
`DstW,`
`DstH`

Location of input stream in output surface.

`LumaKeyEnable`

None zero value enables luma keying for the input stream. Luma keying is used to mark some of the areas of the frame with specified

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

luma values as transparent. It may be used for closed captioning, for example.

`LumaKeyMin`
`LumaKeyMax` Minimum and maximum values of luma key, inclusive. Pixels whose luma values fit in this range are rendered transparent.

`AlphaEnable` None zero value enables global alpha blending for this input stream.

`Alpha` Alpha value for this stream in [0..255] range. 0 – transparent, 255 – opaque.

System Requirements

Hardware

The following processor models are supported:

- Intel® Xeon® Processor E3-1285 v3 and E3-1285L v3 (Intel C226 Chipset) with Intel HD Graphics P4700
- 4th Generation Intel Core™ Processors with Intel Iris™ Pro Graphics, Intel Iris Graphics or Intel HD Graphics 4200+ Series
- 3rd Generation Intel Core Processors with Intel HD Graphics 4000/2500

The following processor models are supported but not recommended for new designs:

- Intel Xeon Processor E3-1285 v2 and E3-1285L v2 (Intel C216 Chipset) with Intel HD Graphics P4000

Please note: Intel Xeon processors are only supported with the chipsets listed. Intel Xeon configurations with other chipsets are not supported. Previous generations of Intel Core processors are not supported. Intel Celeron® and Intel Atom™ processors are also not supported.

Software

- Ubuntu* 12.04 LTS for 64-bit architecture (currently 12.04.3) or SUSE* Linux* Enterprise Server 11 for 64-bit architecture
- Xf86-video-intel driver (needed only for local rendering with LibVA X11 backend support). Recommended version: 2.20.10, http://cgit.freedesktop.org/xorg/driver/xf86-video-intel/snapshot/xf86-video-intel-2.20.10.tar.gz
- Additional platform-specific software requirements (X indicates a supported combination – **only these hardware/kernel combinations are supported**):

OS	Ubuntu* 12.04 LTS		SUSE* Linux* Enterprise Server 11
Kernel version	3.2.0-41	3.8.0-23	SP3 3.0.76-11

Intel® Xeon® E3-1285 v3 / 4th Generation Intel Core™		X	X
Intel Xeon E3-1285 v2 / 3rd Generation Intel Core	X		X

Installed Package Contents

Intel® Media SDK package is installed to `/opt/intel/mediasdk`. All components described below are relative to this installation folder:

Component	Description
`./lib64/8086/{DEV_ID}/libmfxhw64-p.so.1.8`	Intel® Media SDK Dynamic Library, default place to find by Intel Media SDK dispatcher.
`./lib64/8086/{DEV_ID}/libmfxhw64.so`	Symbolic link to Intel Media SDK Dynamic Library
`./bin/x64/libmfxhw64-p.so.1.8`	Intel Media SDK Dynamic Library, for user-defined placement.
`./bin/x64/libmfxhw64.so`	Symbolic link to Intel Media SDK Dynamic Library
`./doc`	Intel Media SDK documentation: • Intel Media SDK Reference Manual `mediasdk-man.pdf` • Samples Overview `MediaSDK Sample Guide.pdf`
`./include`	External Intel Media SDK headers: • Type definitions in `mfxdefs.h` • Structure definitions in `mfxstructures.h` • Structure definitions in `mfxstructures.h`, `mfxastructures.h`, `mfxvstructures.h` and `mfxcommon.h` • Function definitions in C in `mfxvideo.h` • C++ wrapper of the SDK

	functions in `mfxvideo++.h` • Audio function definitions in C in `mfxaudio.h` • C++ wrapper of the SDK audio functions in `mfxaudio++.h` • Extensions for Multi-view Video Coding options `mfxmvc.h` • Extensions for User-Defined Functions `mfxplugin.h` • C++ wrapper for User-Defined Functions `mfxplugin++.h` • Extensions for JPEG*/Motion JPEG Video coding options `mfxjpeg.h (NA for Linux)`
`./lib/lin_x64`	• Static Dispatcher Library `libmfx.a`
`./samples`	• Samples build script `build.pl` • CMake* configuration file CmakeLists.txt • Contains the following source code samples: o Intel Media SDK Encoding Sample in folder `sample_encode` o Intel Media SDK Decoding Sample in folder `sample_decode.` o Intel Media SDK Transcoding Sample in folder `sample_multi_transcode` o Intel Media SDK Video-conferencing Sample in folder `sample_videoconf` o Intel Media SDK Video Processing Sample in folder `sample_vpp` o Intel Media SDK Full Transcoding Sample in folder `sample_full_transcode` o Intel Media SDK Splitters

	and Muxers Sample in folder sample_spl_mux o Intel Media SDK User VPP Plug-in for rotation in folder sample_user_modules
./samples/_bin/x64	Pre-built binaries of installed sample applications • Console sample application binaries (DRM): sample_encode_drm sample_decode_drm sample_vpp_drm sample_multi_transcode_drm sample_videoconf_drm sample_full_transcode_drm • Console sample application binaries (X11): sample_encode_x11 sample_decode_x11 sample_vpp_x11 sample_videoconf_x11 sample_multi_transcode_x11 sample_full_transcode_x11
./builder	CMake helper scripts.
./opensource	Source code for the Intel Media SDK Dispatcher

Installation Prerequisites

Installation is performed by the install_media.sh script. However, these prerequisites must be taken care of before running the script:

1. Double check hardware, OS, and kernel level. Only the configurations described above are supported.

   For SUSE* Linux* Enterprise Server 11, the default kernel is correct. For Ubuntu* 12.04 please install the appropriate kernel as below. This specific kernel is required to match the drm/i915 module in the release. While it is possible to run with other kernel+drm/i915 combinations only the patched drm/i915 module for this release with the kernels/headers below are supported.

For Intel® Xeon® E3-1285 v2 / 3rd Generation Intel Core™ Processors, install kernel and headers

```
apt-get install linux-image-3.2.0-41-generic
linux-headers-3.2.0-41-generic
```

For Intel Xeon E3-1285 v3 / 4th Generation Intel Core Processors, install kernel and headers:

```
apt-get install linux-image-3.8.0-23-generic
linux-headers-3.8.0-23-generic
```

Check that an Ivybridge or Haswell level VGA adapter can be found with `lshw` or `lspci –nn`

```
$ lspci -nn
...
00:02.0 VGA compatible controller [0300]: Intel Corporation Haswell
Integrated Graphics Controller [8086:0416] (rev 02)
...
```

2. Intel Media SDK for Linux Servers now includes a customized libdrm. To avoid conflicts please remove any other versions of libdrm from the system. This can be accomplished on the command line via

```
$ sudo find /usr –name ‘libdrm*’ (check files to remove)
$ sudo find /usr –name ‘libdrm* -exec rm –rf {} \;
```

Installation

After the prerequisites above have been completed, create a directory for the installation, extract the package contents, and run the installer.

```
$ export MEDIASDK_INSTALL_FOLDER=(some folder)
$ mkdir $MEDIASDK_INSTALL_FOLDER
$ mv intel-linux-media_{release}.tar.gz $MEDIASDK_INSTALL_FOLDER
$ cd $MEDIASDK_INSTALL_FOLDER
$ tar -xvzf intel-linux-media_{release}.tar.gz
$ sudo ./install_media.sh
```

The installer output should be similar to the text below. For a supported configuration using the specific kernel appropriate to your processor architecture press y to install the kernel-mode driver (KMD), which is the drm.ko, drm_kms_helper.ko and i915.ko patched for this specific configuration.

```
INFO... Install on Ubuntu ...
INFO... Installing New Driver...
INFO... MediaSDK installed successfully in /opt/intel/mediasdk!
INFO... Do you want to install KMD?
press 'y' to confirm, otherwise cancelled.y
INFO... Original i915.ko backuped in kmd_backup/i915.ko.2014-03-03_082642
INFO... Original drm.ko backuped in kmd_backup/drm.ko.2014-03-03_082642
INFO... Original drm_kms_helper.ko backuped in
kmd_backup/drm_kms_helper.ko.2014-03-03_082642
INFO... Trying to install  kmd...
INFO... ./kmd/binary/xcode-ubuntu-12.04-k3.8-rel/i915.ko installed
successufully.
INFO... ./kmd/binary/xcode-ubuntu-12.04-k3.8-rel/drm.ko installed
successufully.
INFO... ./kmd/binary/xcode-ubuntu-12.04-k3.8-rel/drm_kms_helper.ko installed
successufully.
INFO... After reboot, you can 'lsmod' to identify whether i915.ko drm.ko
drm_kms_helper.ko loaded. if not, you have to rebuild kernel by yourself with
patched files (kdm/source) in this package.
update-initramfs: Generating /boot/initrd.img-3.8.0-23-generic
INFO... Kernel module updated successfully!
INFO... Package installation Done.
```

After rebooting, double check that the i915 module is loaded correctly

```
$lsmod | grep i915
i915               617480  2
drm_kms_helper        49196  1 i915
drm                285862  3 i915,drm_kms_helper
i2c_algo_bit         13564  1 i915
video               19652  1 i915
```

Ensure that the Intel® Media SDK library can be found. By default, the dispatcher searches in `/opt/intel/mediasdk/lib64/8086/<device_id>/`. The libmfxhw-p.so.{version> and libmfxhw64.so files can also be located in convenient Linux library locations like `/usr/local/lib` or anywhere else according to standard Linux library search rules. For example, the library search path can be adjusted by the `LD_LIBRARY_PATH` variable:

```
$ export LD_LIBRARY_PATH=$MEDIASDK_INSTALL_FOLDER/bin/x64
```

Communication with the DRM library occurs via /dev/dri/cardx handles. Usually /dev/dri/card0, though there can be more entries if there are more graphics adapters. Quite often permissions are set so that root access is required for users not in the video group. Please add Media SDK application users to the video group and/or ensure users have permissions to work as a regular user with `sudo chmod 666 /dev/dri/card0`.

Testing with pre-built samples

Pre-built samples are included in `l_MSDK/samples/bin/x64`.

In case you don't have easy access to raw elementary streams, utilities like FFmpeg* can generate them for you:

```
$ ffmpeg -i input.mp4 -an -vcodec copy -bsf h264_mp4toannexb -f h264 out.h264
```

This can work with a wide variety of .mp4 content, including creative commons clips like Sintel or Big Buck Bunny.

To test decode:

```
$ sample_decode_drm h264 -i in.264 -o out.yuv -hw
```

To test encode:

```
$ sample_encode_drm h264 -i in.yuv -o out.264 -hw -w <in.yuv width> -h <in.yuv height>
```

To test transcode:

```
$ sample_multi_transcode_drm h264 -i::h264 in.264 -o::h264 out.264 -hw
```

Note: For Intel Media SDK for Linux Servers there is no software implementation, so for these samples to work -hw must be specified on the command line.

Compiling samples

The Intel® Media SDK samples are built with a version of CMake* which is newer than the one available by default via the Ubuntu* or SUSE* Linux* Enterprise Server package management system.   Please install the latest version from www.cmake.org.

To build, make sure $MFX_HOME is set to the directory corresponding to your build then type

```
perl build.pl --cmake=intel64.make.release -build
```

in the samples directory.

Please note: the build system will only build samples if the prerequisites can be found. For most cases only libdrm is needed. If X11 is not installed the _x11 samples will not be built.

For more information see `MediaSDK Sample Guide.pdf` in the doc directory.

Installing for non-supported configurations

There is a good chance Intel® Media SDK will work in other configurations. For full support using one of the two supported configurations is required. However, you are free use the Intel Media SDK in other settings if you are willing to go through an extra step in reporting issues. **If an issue can be reproduced in one of the supported configurations it can be addressed, otherwise you are on your own.**

Since Intel Media SDK is based on hardware access via the video driver, the main concern with alternative installations will be making sure that all device IDs and other changes to the kernel are available. Usually patches required to enable working with the hardware are submitted to the kernel repository tip relatively quickly. If you use an advanced kernel or compile from close to the tip you are likely to get most, if not all, changes required for Intel Media SDK to work.

Unfortunately there are no guarantees at this point with this approach. Enabling Intel Media SDK to work on a wider set of configurations is a work in progress.

Versions Information

To obtain Intel® Media SDK Library version run the following command (output should be as provided below):

```
$ strings libmfxhw64.so | grep mediasdk
mediasdk_product_version: 5.0.1603321.86136
mediasdk_file_version: 5.14.3.12
mediasdk_copyright: Copyright(c) 2007-2014 Intel Corporation
mediasdk_product_name: Intel(r) Media SDK 2014 for Linux* Servers
```

To obtain versions of the prebuilt samples provided within Intel Media SDK package run them without any command-line options. For example:

```
$ ./sample_decode_drm
Intel(R) Media SDK Decoding Sample Version 5.0.1603321.86136
$ ./sample_encode_drm
Intel(R) Media SDK Encoding Sample Version 5.0.1603321.86136
$ ./sample_vpp_drm
Intel(R) Media SDK VPP Sample Version 5.0.1603321.86136
$ ./sample_multi_transcode_drm
Intel(R) Media SDK Multi Transcoding Sample Version 5.0.1603321.86136
```

Known Limitations

This release is subject to the following known limitations:

- **API:**

  Intel® Media SDK API is designed for a range of products. A particular product release may support only a subset of the features of the declared API version. This release has the following API limitations:

  - Only the following features among those introduced in API 1.7 are supported:
    - `RateControlMethod::MFX_RATECONTROL_LA`

        - mfxExtCodingOption2::LookAheadDepth
        - mfxExtCodingOption2::MBBRC
        - mfxExtCodingOption2::Trellis
    - Only the following features among those introduced in API 1.8 are supported:
        - mfxVideoCodecPlugin
        - mfxExtVPPComposite
        - mfxExtVPPDeinterlacing
        - mfxExtCodingOption2::LookAheadDS
        - mfxHandleType::MFX_HANDLE_VA_DISPLAY
        - mfxImpl::MFX_IMPL_VIA_VAAPI

  Additionally, all the APIs listed above, except for mfxVideoCodecPlugin, mfxHandleType::MFX_HANDLE_VA_DISPLAY and mfxImpl::MFX_IMPL_VIA_VAAPI, are supported only on Intel Xeon® E3-1285 v3 / 4th Generation Intel Core™ platforms. Make sure to call Query functions to check the actual support on particular platform in runtime.

- **Performance:**
    - Advanced De-Interlacing provides better quality but might be slower than BOB DI in some cases. API control mfxExtVPPDeinterlacing is provided to choose the desired de-interlacing method.
    - Transcoding to H.264 on lower target usages might show worse performance as compared to Intel Media SDK for Linux Servers 2013 R2 on Intel Xeon E3-1285 v3 / 4th Generation Intel Core platforms. This effect is due to several encoding features improving subjective and objective visual quality being enabled by default (multiple reference frames, MBBRC, Trellis).
- **H.264 encoder:**
    - Look Ahead BRC may generate non HRD-compliant streams.
    - Careful memory/resource planning is needed when using Look Ahead BRC due to storage of pre-analyzed frames. 1:N and N:N transcoding use cases are especially demanding for memory.
    - Trellis option can be enabled only on lower target usages, on some of those it is enabled by default but can be switched off. Exact implementation details are hidden and may change with time and between platforms, so using Query function to retrieve actual support is strongly recommended.
    - MBBRC option is enabled by default on lower target usages but can be switched off. Exact implementation details are hidden and may change with time and between platforms, so using Query function to retrieve actual support is strongly recommended.
- **MPEG-2 encoder:**

    - The MPEG-2 encoder is hardware accelerated only on Intel Xeon E3-1285 v3 / 4th Generation Intel Core platforms. On Intel Xeon E3-1285 v2 / 3rd Generation Intel Core software fallback will be used.
    - The MPEG-2 encoder may produce output that under-runs the MPEG-2 video buffer verifier hypothetical reference decoder model (VBV HRD) on particular streams.
- **Decoders:**
    - H.264/VC-1 decoders might be unstable in multi-process environment on Ubuntu* 12.04 on some streams.
    - Decoders may not handle some corrupted streams correctly: they may not recover from corruption and may not provide information on corruption in `mfxFrameSurface::Data::Corrupted` field.
- **VPP**:
    - VPP Composition suffers from a per-frame memory leak when used for more than 8 input streams. A suggested workaround is to perform multistage VPP Composition on application level: compose 8 streams using one VPP component, then feed the output and 7 more streams to the next VPP in a chain, etc., until the desired number of composed streams is reached.
    - VPP Composition produces slightly blurred output when used for more than 8 streams.
    - Alpha Blending API is not final and is subject to change. The feature has the following known limitations:
        - Luma key works only for one stream
        - Corner cases with alpha=0 (fully transparent) and alpha=255 (fully opaque) might be processed incorrectly
- **USER/Plug-ins**
    - General Plug-in API included as experimental feature, stability issues may exist.
- Limitations related to source code samples are discussed in their corresponding readme files. See "`<install-folder>/doc/MediaSDK Sample Guide.pdf`" for an overview of the samples and additional documentation.

Legal Information

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.

UNLESS OTHERWISE AGREED IN WRITING BY INTEL, THE INTEL PRODUCTS ARE NOT DESIGNED NOR INTENDED FOR ANY APPLICATION IN WHICH THE FAILURE OF THE INTEL PRODUCT COULD CREATE A SITUATION WHERE PERSONAL INJURY OR DEATH MAY OCCUR.

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 by visiting Intel's Web Site.

MPEG is an international standard for video compression/decompression promoted by ISO. Implementations of MPEG CODECs, or MPEG enabled platforms may require licenses from various entities, including Intel Corporation.

VP8 video codec is a high quality royalty free, open source codec deployed on millions of computers and devices worldwide. Implementations of VP8 CODECs, or VP8 enabled platforms may require licenses from various entities, including Intel Corporation.

Intel, the Intel logo, Intel Core are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

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