INTEL® MEDIA SDK & PERFORMANCE ANALYSIS

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 or 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. Refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice Revision #20110804.

LEGAL NOTICES AND DISCLAIMERS (1 OF 2)

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software, or service activation. Performance varies depending on system configuration. No computer system can be absolutely secure. Check with your system manufacturer or retailer or learn more at www.intel.com.

Performance estimates were obtained prior to implementation of recent software patches and firmware updates intended to address exploits referred to as "Spectre" and "Meltdown." Implementation of these updates may make these results inapplicable to your device or system.

Cost reduction scenarios described are intended as examples of how a given Intel-based product, in the specified circumstances and configurations, may affect future costs and provide cost savings. Circumstances will vary. Intel does not guarantee any costs or cost reduction.

This document contains information on products, services, and processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest forecast, schedule, specifications and roadmaps.

Any forecasts of goods and services needed for Intel's operations are provided for discussion purposes only. Intel will have no liability to make any purchase in connection with forecasts published in this document.

Arduino* 101 and the Arduino infinity logo are trademarks or registered trademarks of Arduino, LLC.

Altera, Arria, the Arria logo, Intel, the Intel logo, Intel Atom, Intel Core, Intel Nervana, Intel Xeon Phi, Movidius, Saffron, and Xeon are trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

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

LEGAL NOTICES AND DISCLAIMERS (2 OF 2)

This document contains information on products, services, and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest forecast, schedule, specifications, and roadmaps. Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software, or service activation. Learn more at intel.com, or from the OEM or retailer.

No computer system can be absolutely secure.

Tests document performance of components on a particular test, in specific systems. Differences in hardware, software, or configuration will affect actual performance. Consult other sources of information to evaluate performance as you consider your purchase. For more complete information about performance and benchmark results, visit www.intel.com/performance.

Cost-reduction scenarios described are intended as examples of how a given Intel-based product, in the specified circumstances and configurations, may affect future costs and provide cost savings. Circumstances will vary. Intel does not guarantee any costs or cost reduction.

Statements in this document that refer to Intel's plans and expectations for the quarter, the year, and the future are forward-looking statements that involve a number of risks and uncertainties.

A detailed discussion of the factors that could affect Intel's results and plans is included in Intel's SEC filings, including the annual report on Form 10-K.

The products described 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.

Performance estimates were obtained prior to implementation of recent software patches and firmware updates intended to address exploits referred to as "Spectre" and "Meltdown." Implementation of these updates may make these results inapplicable to your device or system.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel does not control or audit third-party benchmark data or the web sites referenced in this document. You should visit the referenced web site and confirm whether referenced data are accurate.

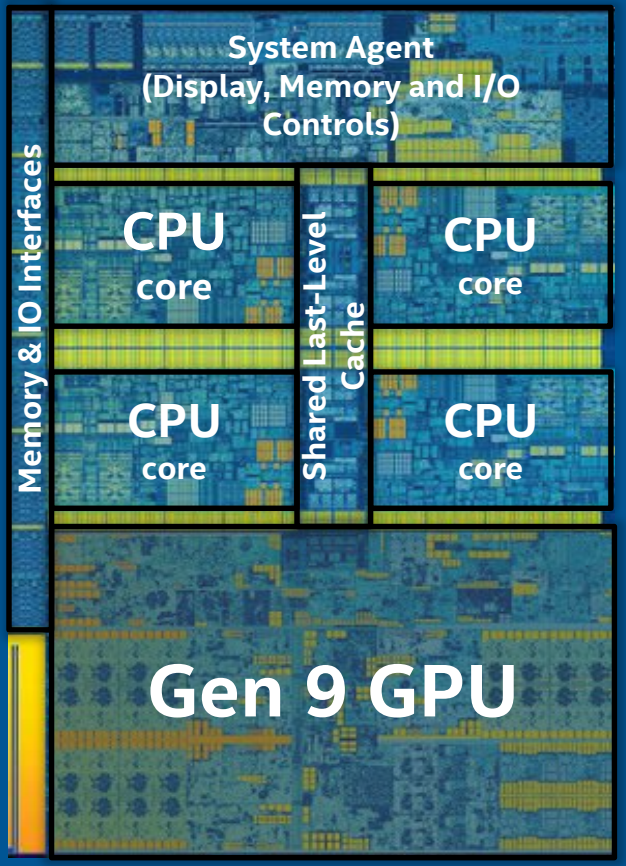
Intel, the Intel logo, Pentium, Celeron, Atom, Core, Xeon, Movidius, Saffron, and others are trademarks of Intel Corporation in the United States and other countries.

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

INTEL® INTEGRATED GRAPHICS

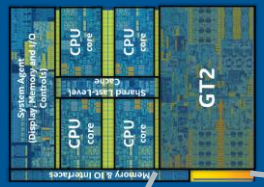
Gen is the internal name for Intel's on-die GPU solution. It's a HW ingredient with various configurations

- Intel® Core® Processors include Gen hardware
- Gen GPU can be used for graphics, and also as a general compute resource
- Libraries contained in Intel® Distribution of OpenVINO™ (and many others) support Gen offload using OpenCL*



6th Generation Core i7 (Skylake) Processor

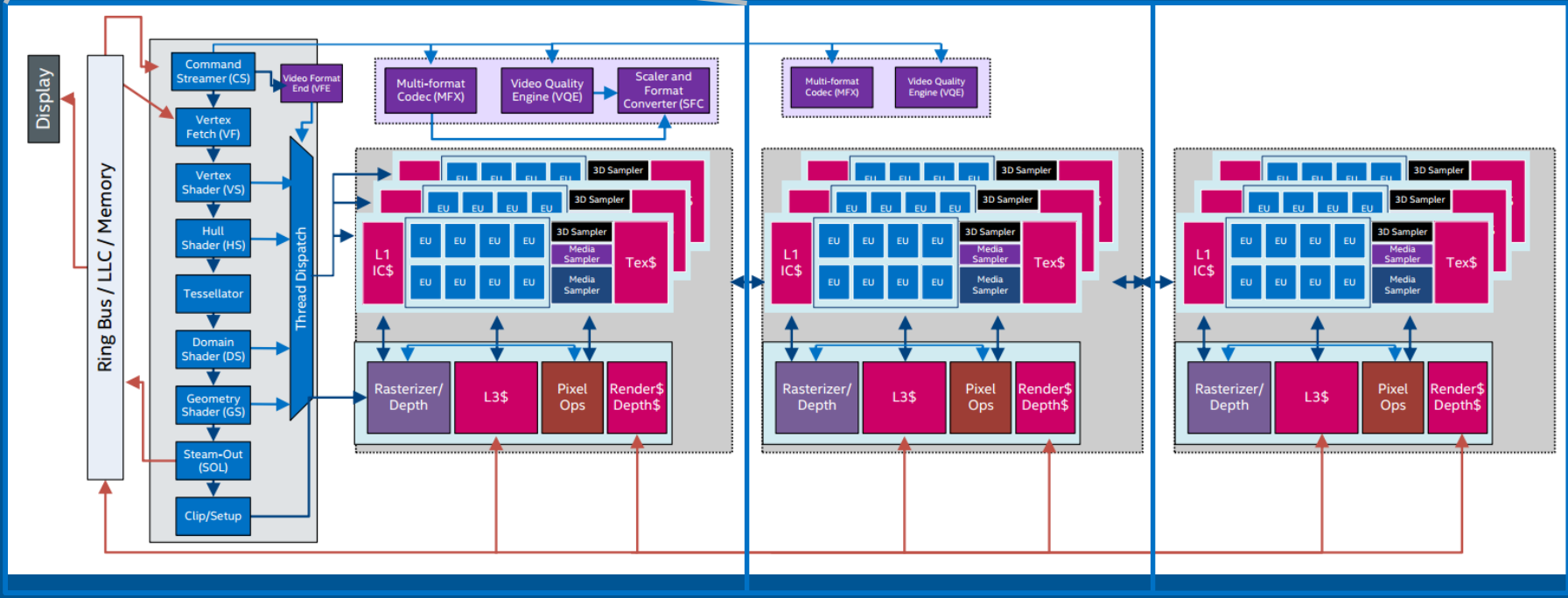
INTEL® GPU CONFIGURATIONS

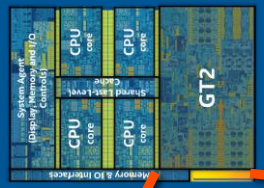


GT2
Intel® HD Graphics
24 EUs, 1 MFX

GT3
Intel® Iris™ Graphics
48 EUs, 2 MFX

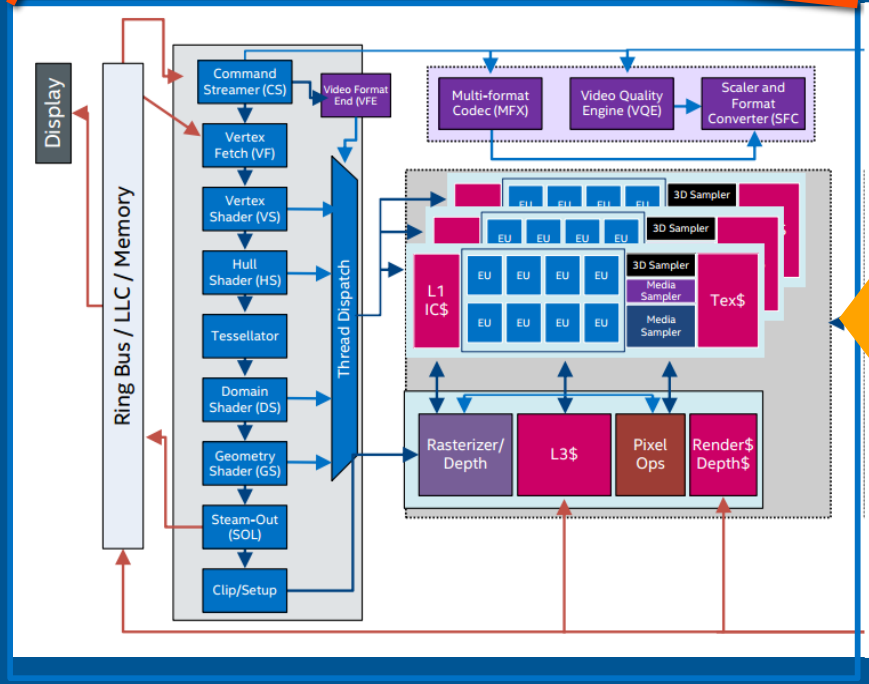
GT4
Intel® Iris™ Pro Graphics
72 EUs, 2 MFX

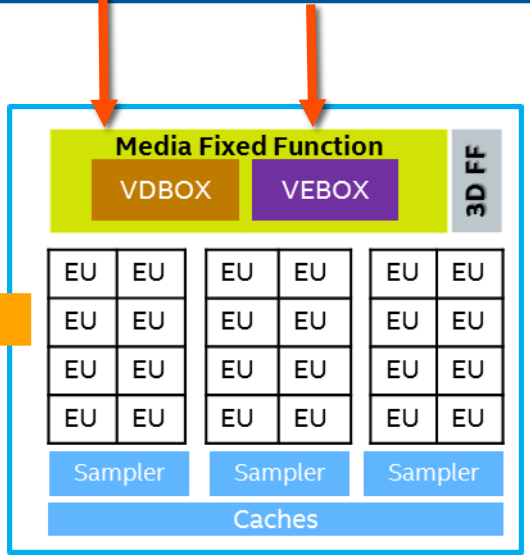




VDBox : A BitStream decoder
Encoding acceleration

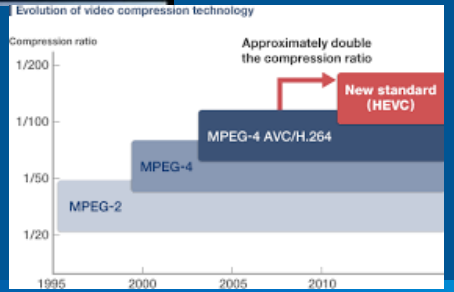
VEBox : HW acceleration for video enhancement/frame processing operations





WHY NEED ENCODERS/DECODERS?

- 1 Minute of HD video (60 sec x 25 FPs x 1920 x 1080 x 3/2 bytes) = 4.6 GB !!
- Encode before send/store, decode before using.
  - The good news is that Videos usually have spatial and temporal redundancy we could use
  - Intel GPU can accelerate these compute intensive tasks



RECAP VIDEO ANALYTICS PIPELINE

Completed: Integrate OpenVINO™ for Inference instead of OpenCV*, **what else**?

OpenCV* has been used all the purposes along-side inference:

- Read Video from Source File/Stream
- Demux from Video Container
- Decode Compressed Bitstream to Frame data (RGB, YUV etc.)
- Re-size
- Normalize Objects
- Draw Rectangles
- Re-size back to original size
- Save, Write to File

RECAP VIDEO ANALYTICS PIPELINE

Framework	Requirement/Function
OpenCV* (FFMpeg, Gstreamer)	Read Video from Source File/Stream
OpenCV* (FFMpeg, Gstreamer)	Demux
Intel® Media SDK	Decode Compressed Bitstream
Intel® Media SDK	Re-size, Crop etc.
OpenCV*	Normalize Image Data (Mean Substraction)
OpenCV*	Add Text/Shapes etc.
Intel® Media SDK	Resize, change color space
Intel® Media SDK	Encode Frame to binary format or Transcode
OpenCV* (Gstreamer*)	Save, Write to File

INTEL® MEDIA SDK OVERVIEW

- Intel® Media SDK equips developers with a standard API to create high-performance video solutions for consumer and professional uses.
- Intel® Media SDK provides easy access to hardware acceleration with Intel-optimized software fallback.
- Developers can use their own software codecs with the Intel® Media SDK plugin mechanism.
- Development teams can shift resources from performance optimization for each individual hardware platform to focusing on feature innovation and application capabilities in their video solutions.

INTEL® MEDIA SDK IMPLEMENTATION

Smart Cameras

Apollo Lake

INTEL® MEDIA SDK
for Embedded Linux

(Intel® Atom™ x7 E3950)

Encode: 6 AVC or 4 HEVC 1080p@30fps

Decode: 20 AVC 1080p@30fps

NVR, Data Center
Clod

Linux (CentOS)
or Windows Server

INTEL® MEDIA SERVER STUDIO

(Intel® Xeon® E3 15xx v5)

Encode: 18 AVC 1080p@30fps

1 HEVC 4K@60fps

Transcode: 4 AVC (4K) →HEVC (4K)

Client

INTEL® MEDIA SDK
for Windows & Embedded Linux

WHAT IS NEW IN INTEL® MEDIA SOFTWARE TOOLS – 2018 RELEASE?

Intel® Media Server Studio for Linux*

- **Enhances AVC compression & video quality** features
- **Improves HEVC video encode quality** & **CPU performance** for multiple simultaneous media sessions
- **Expands API parameters** for more control over the codecs
- **Increases performance** significantly for **Sessions Joining API**
- Supports **CentOS 7.4**

More details

Intel® Media SDK for Embedded Linux*

- **Supports new 8th gen Intel® Core™, Celeron® & Pentium® processors for IOT solutions**, which includes a fully validated media stack for building robust solutions
- Common usages: digital surveillance, retail, smart cities, industrial, health care, & more

More details: intel.ly/2q6s09a

Intel® Media SDK for Windows*

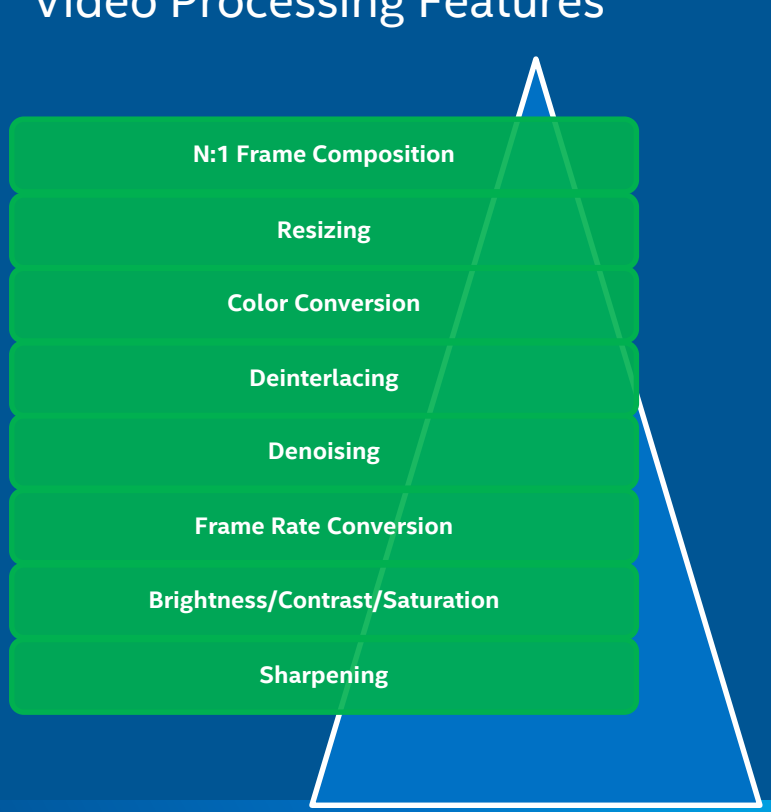
- Provides **encoding features** from Intel® Media Server Studio Professional Edition **now FREE**[1], adds support for **data center, visual cloud, broadcasting** & **embedded**[2]
- Includes **Video Quality Caliper**[1], an easy to use tool to view **PSNR, SS**IM
- Supports **HEVC** within codec components[1]
- **AVC:** Supports weighted predictions for P-frames & B-frames
- **HEVC:** Provides max frame size bitrate control

More details

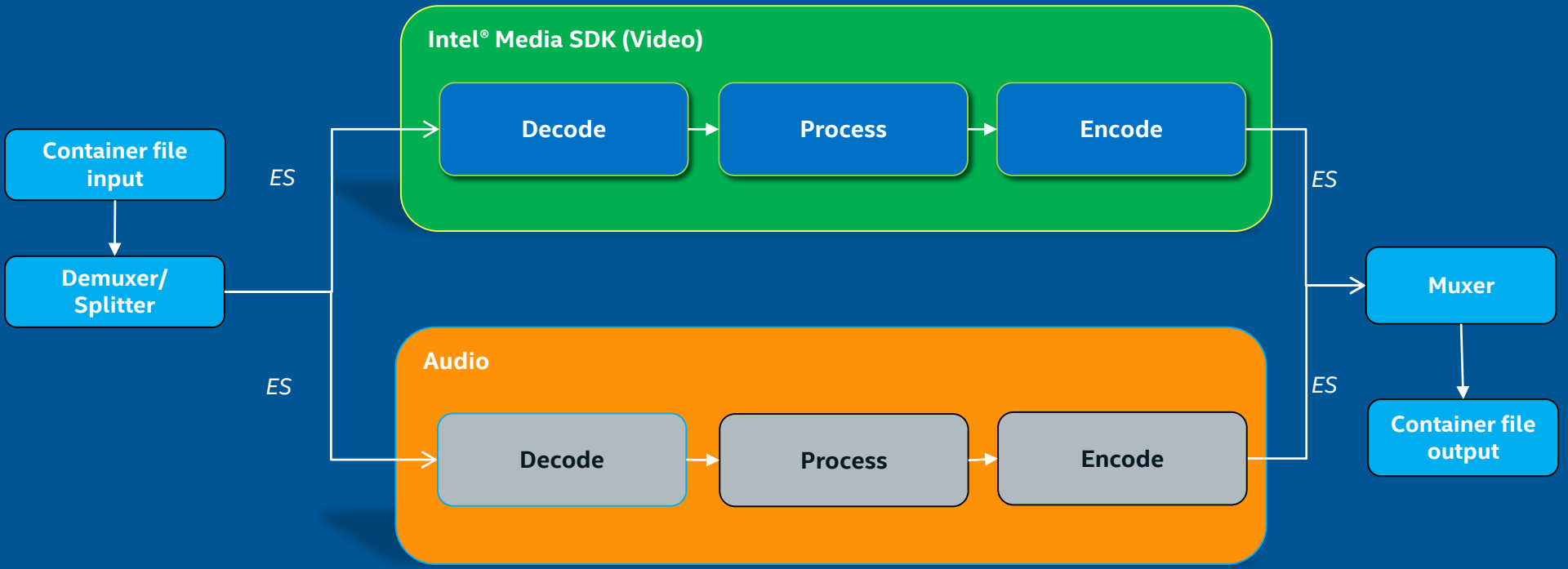
[1]Previously available only with a paid license – **now available for FREE!**
[2]In addition to current desktop, client, mobile support.

INTEL® MEDIA SDK IMPLEMENTATION

- Supports multiple formats
  - H.265 (HEVC)
  - H.264 (AVC)
  - MPEG-2
  - MJPEG
  - VP8
  - VP9
  - VC-1
  - More support can be added with plugin extensions
- Cross OS and Cross-Platform API
  - Embedded Linux, CentOS, Windows
  - C/C++ APIs
- Open Sourced
- https://github.com/Intel-Media-SDK



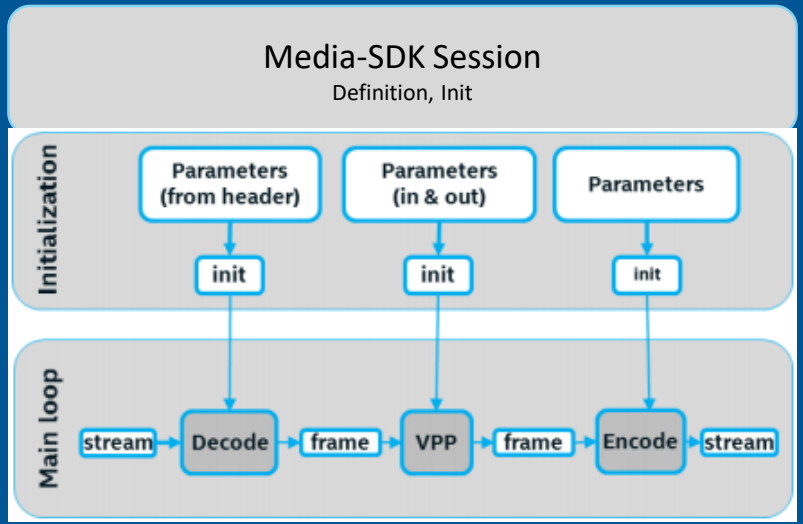
INTEL® MEDIA SDK IMPLEMENTATION



ES = Elementary stream

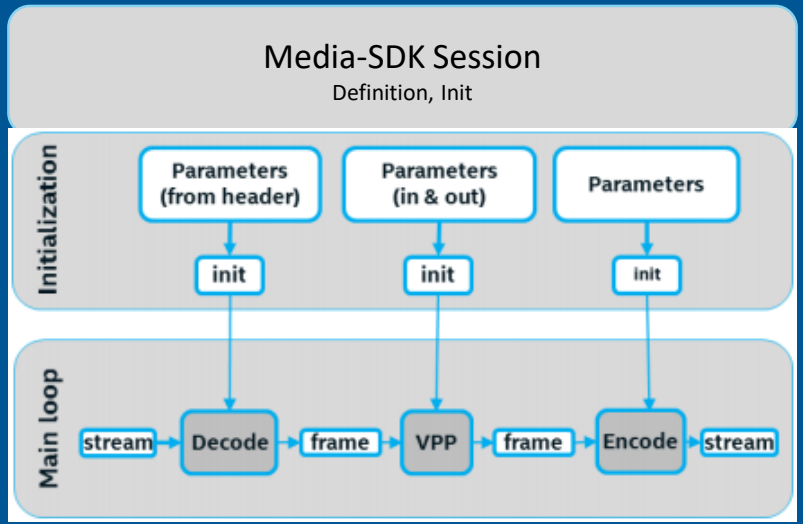
INTEL® MEDIA SDK CONCEPTS

- Media SDK functions fall into these categories :
  - Decode
  - Encode
  - VPP (Video Processing Pipeline)
  - (and Core, Aux and Misc)
  - Definition, Initialization
- Media-SDK “Session” has to be defined and initialized
- Each session can contain 3 “Stages”: “decode”, “encode” and “VPP”
  - After the session is initialized, each stage also has to be defined and initialized
  - A session (all stages) can run on ONE hardware device (CPU/GPU)
- And can use system (CPU) or Video (GPU) memory.

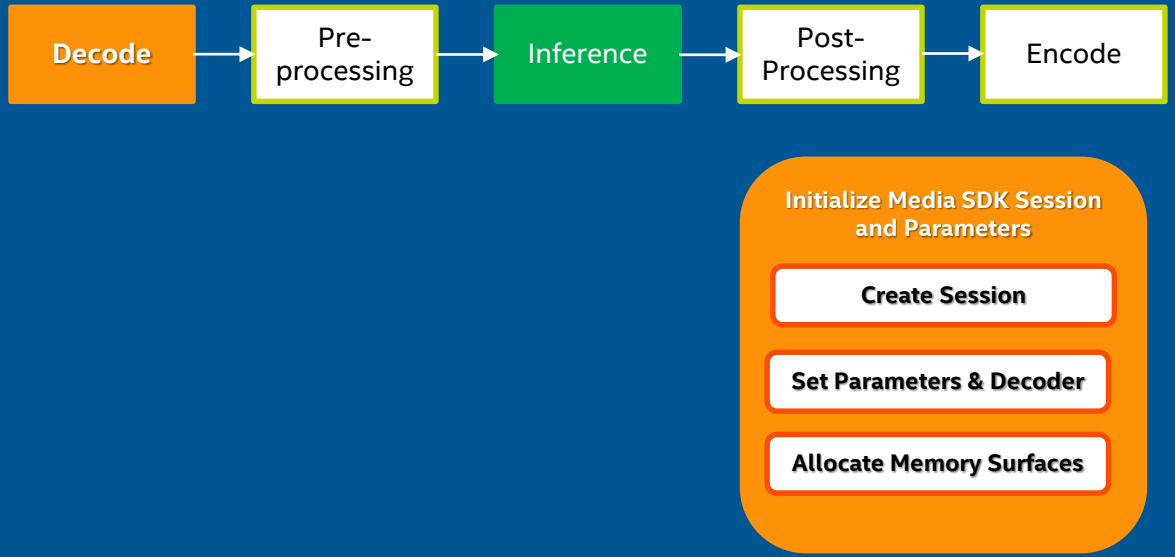


INTEL® MEDIA SDK CONCEPTS

- Session/pipeline-based:
  - A session is created for a pipeline of steps.
  - Surfaces may be shared internally between pipeline stages.
- Asynchronous:
  - Each stage can have multiple frames “in flight”
  - Frame surface synchronization status needs to be checked.
  - Frames may be locked while a session is working on them.
- Based on the NV12 color format:
  - NV12 is the ‘native’ format, (the architecture can usually provides better performance)
- Designed to minimize copies:
  - Arrange pipeline steps to reuse surfaces in the same location instead of copying them between CPU and GPU.



INTEL® MEDIA SDK IMPLEMENTATION DETAILS



INTEL® MEDIA SDK IMPLEMENTATION DETAILS

ENCODED STREAM

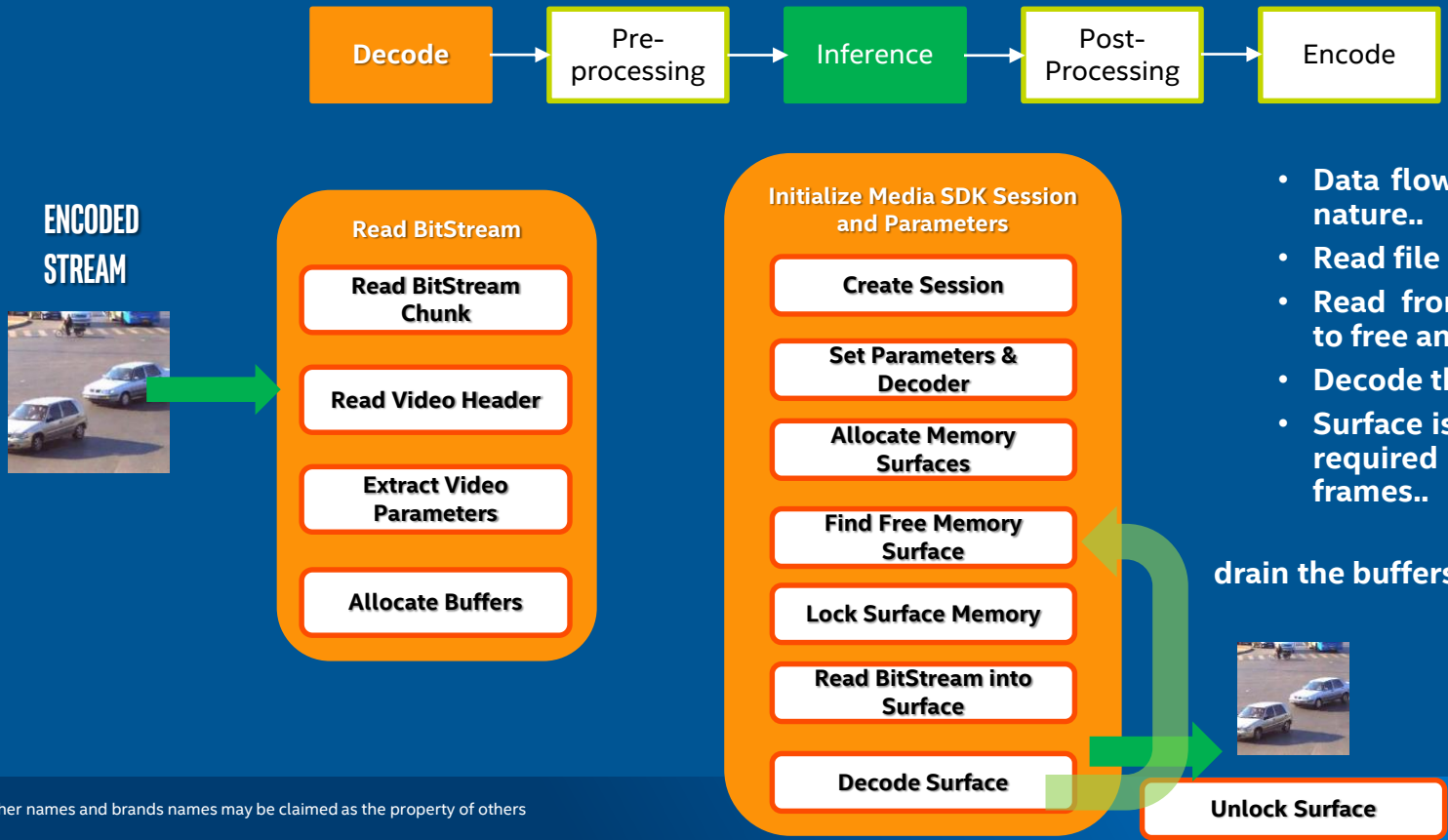
Read BitStream

- Read BitStream Chunk
- Read Video Header
- Extract Video Parameters
- Allocate Buffers

Initialize Media SDK Session and Parameters

- Create Session
- Set Parameters & Decoder
- Allocate Memory Surfaces

INTEL® MEDIA SDK IMPLEMENTATION DETAILS



- Data flow is asynchronous in nature..
- Read file into Bit-Stream
- Read from Bit-stream buffer to free and locked surface
- Decode the frame
- Surface is unlocked when not required anymore by other frames..

INTEL® MEDIA SDK IMPLEMENTATION DETAILS

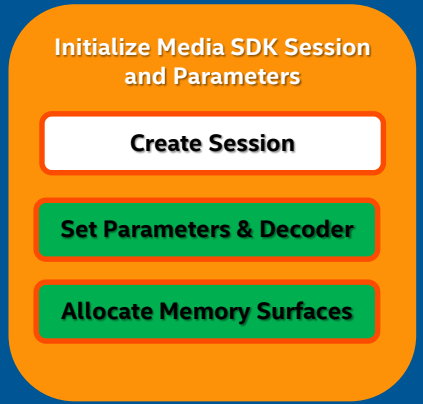

```
mfxStatus sts; //Media-SDK session type, Version, Session..
mfxIMPL impl;  //Implementation type, HW/SW acceleration..
mfxVersion ver;
MFXVideoSession mfxSession;  //Our media-SDK Session
mfxBitstream mfxBS;//Bit Stream Buffer (contain encoded stream..)
mfxFrameSurface1** pmfxSurfaces;          //meta data to frame buffer
mfxU8* surfaceBuffers;       //Output buffer
sts = Initialize(impl, ver, &mfxSession, NULL);//Initialize M-SDK Session
```

Initialize Media SDK Session and Parameters

- **Create Session**
- **Set Parameters & Decoder**
- **Allocate Memory Surfaces**

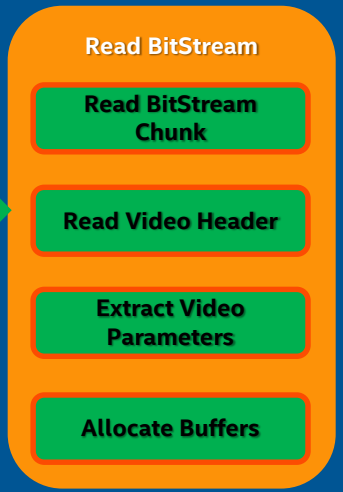
INTEL® MEDIA SDK IMPLEMENTATION DETAILS


```
MFXVideoDECODE mfxDEC(mfxSession); // Create a decode stage
mfxVideoParam mfxVideoParams;    //Prepare Decoder parameters
                                 //And set them to AVC, using system memory
memset(&mfxVideoParams, 0, sizeof(mfxVideoParams));
mfxVideoParams.mfx.CodecId = MFX_CODEC_AVC;              // h264
mfxVideoParams.IOPattern = MFX_IOPATTERN_OUT_SYSTEM_MEMORY; // CPU memory

mfxBitstream mfxBS; //Media-SDK bit-stream buffer
memset(&mfxBS, 0, sizeof(mfxBS)); //Allocate the bit-stream buffer
mfxBS.MaxLength = 1024 * 1024;
mfxBS.Data = new mfxU8[mfxBS.MaxLength];
```



INTEL® MEDIA SDK IMPLEMENTATION DETAILS

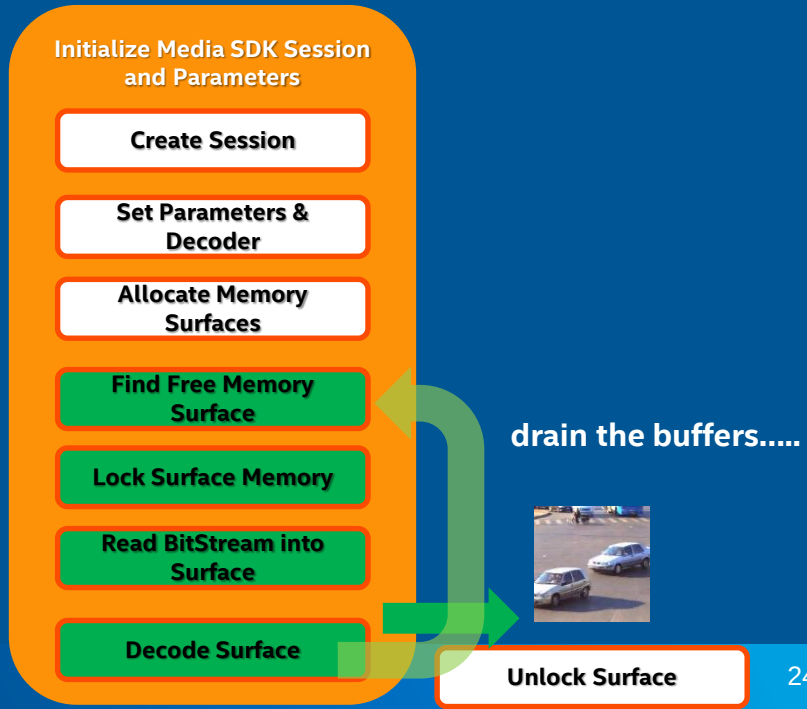
ENCODED STREAM




```
sts = ReadBitStreamData(&mfxBS, f_i); //Read data chunk from bit-stream buffer
sts = mfxDEC.DecodeHeader(&mfxBS, &mfxVideoParams); //look for video header
mfxFrameAllocRequest DecRequest; //what number of surfaces required?
memset(&DecRequest, 0, sizeof(DecRequest)); //Allocate them..
sts = mfxDEC.QueryIOSurf(&mfxVideoParams, &DecRequest);

sts = mfxDEC.Init(&mfxVideoParams); //Initialize the decoder
```

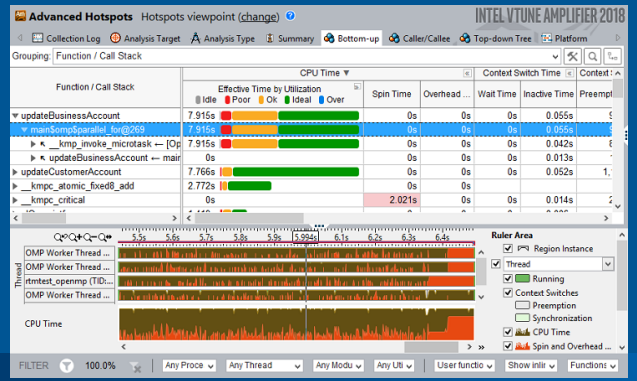
INTEL® MEDIA SDK IMPLEMENTATION DETAILS


```
while (MFX_ERR_NONE <= sts || MFX_ERR_MORE_DATA) {//Main decoding loop
    sts = ReadBitStreamData(&mfxBS, f_i);  //Read encoded stream from
file

    nIndex = GetFreeSurfaceIndex(pmfxSurfaces, numSurfaces); //Find
free frame

    sts = mfxDEC.DecodeFrameAsync(&mfxBS, pmfxSurfaces[nIndex],
&pmfxOutSurface, &syncp);
              // Decode the frame (asynchronous)

    sts = mfxSession.SyncOperation(syncp, 60000);
              // Synchronize. Wait until decoded frame is ready

    mfxFrameData* pData = &pmfxOutSurface->Data;
              // Taking the decoding surface for further analysis
}
```



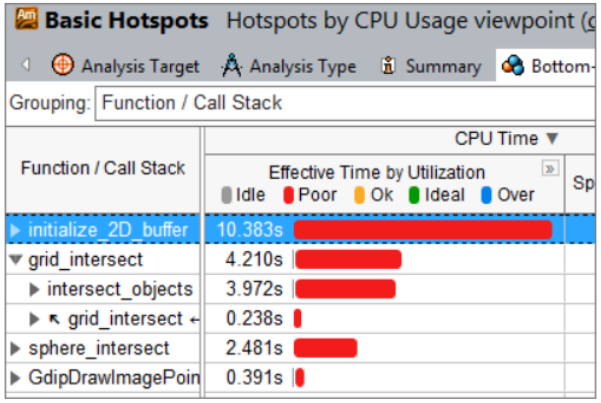
INTEL® VTUNE™ AMPLIFIER

- High-performing modern code must be:
  - **Threaded and scalable** to utilize multiple CPUs
  - **Vectorized** for efficient use of multiple FPUs
  - **Tuned** to take advantage of non-uniform memory architectu
- Real time, Heterogeneous workloads are even more complicate
  - vTune is a great place to start the performance analysis.

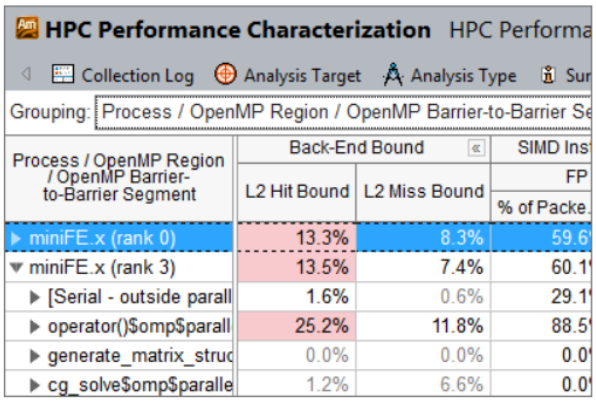


Processors	Intel® and compatible processors and coprocessors including Intel Xeon Phi processors.
Languages	C, C++, C#, Fortran, Java*, Python*, Go*, assembly, and more.
Compilers	Works with compilers from Microsoft, GCC, Intel, and others that follow the same standards.
Development Environments	Integrate with Microsoft Visual Studio* or run as a stand-alone product.
Host Operating Systems	Windows, Linux, and macOS (optional download[1])
Target Operating Systems	Windows, Linux, FreeBSD*, Android*, Tizen*, Wind River Linux*, and Yocto Project*
Basic Threading Analysis **Full threading information**	OpenMP*, Intel TBB, and native threads.
Extended Threading Performance Analysis	OpenMP and Intel TBB
MPI parallelism	Integration with Intel Trace Analyzer and Collector MPI profiler
GPU	OpenCL and media application tuning on newer Intel processors.

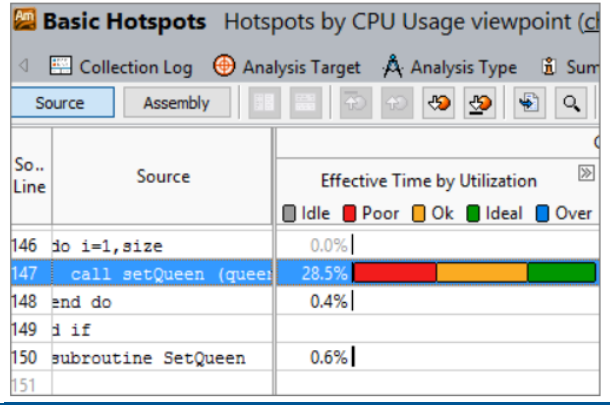
ALGORITHM ANALYSIS



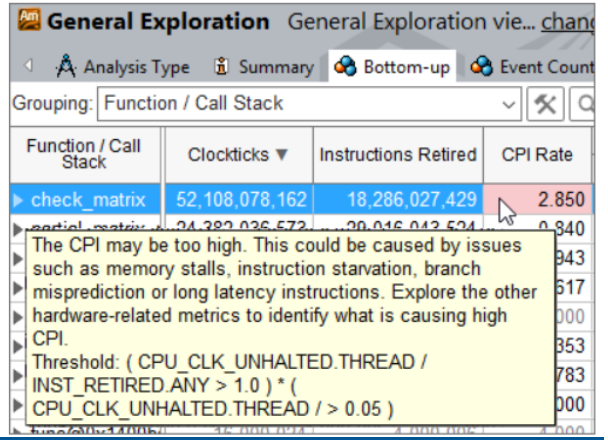
COMPUTE-INTENSIVE APPLICATIONS ANALYSIS



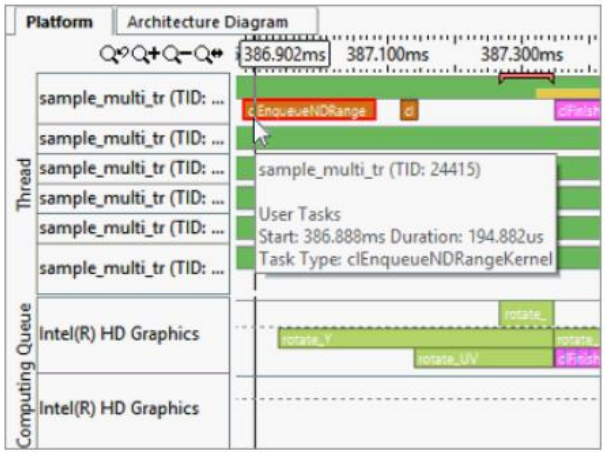
SOURCE ANALYSIS



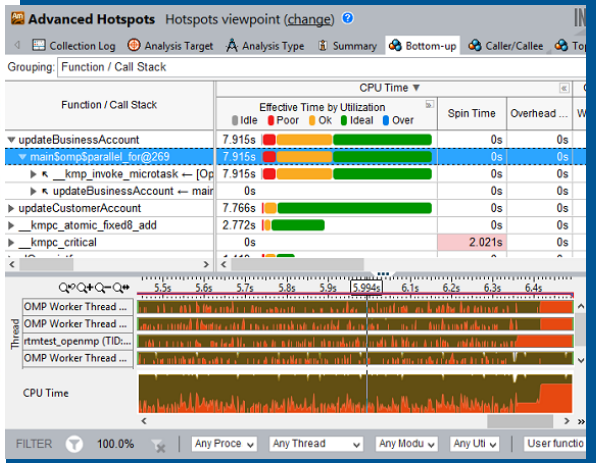
MICROARCHITECTURE ANALYSIS



PLATFORM ANALYSIS

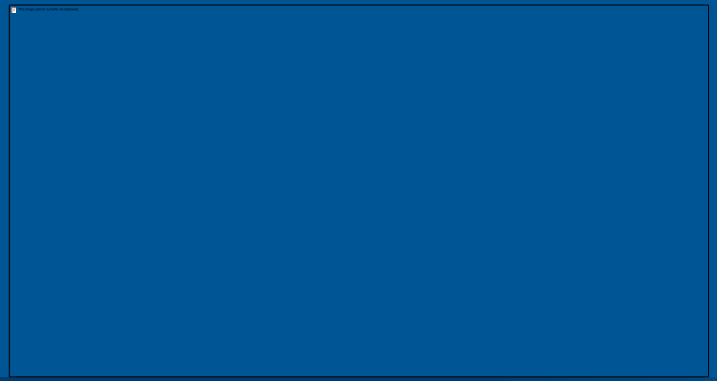


ADVANCED HOTSPOTS



INTEL® VTUNE™ AMPLIFIER

- ITT - Instrumentation and Tracing Technology APIs
  - VTune™ Amplifier enable your application to generate and control the collection of trace data during its execution
  - All major performance calls of the Inference Engine are instrumented with ITT
  - This allows viewing the Inference Engine calls on the VTune™ timelines and aggregations plus correlating them to the underlying APIs like OpenCL.



```
#include "ittnotify.h"
#include <ctime>
#include <windows.h>

#define NUM_THREADS 4

__itt_domain* domain = __itt_domain_create(L"Task Domain");
__itt_string_handle* UserTask = __itt_string_handle_create(L"UserTask");
__itt_string_handle* UserSubTask = __itt_string_handle_create(L"UserSubTask");
```

1. Include ittnotify.h

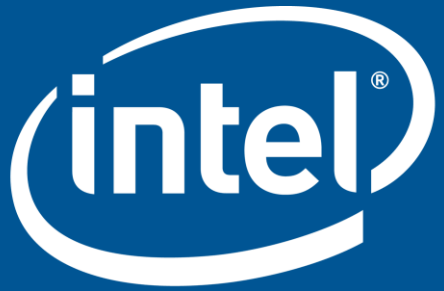
2. Create __itt_* handles

```
...
__itt_task_begin (domain, __itt_null, __itt_null, UserSubTask);
do_foo(1);
__itt_task_end (domain);
...
```

3. Insert task begin and end marks in your code

INTEL® VTUNE™ AMPLIFIER

- Analyze source code for performance bottlenecks
- Characterize the amount of parallelism in an application
- Determine which synchronization locks or APIs are limiting the parallelism in an application
- Balance the load on the hetero devices



Software

HANDS-ON LAB 3

Hands-on lab

1. Log-in to your lab PC (intel/**P@ssw0rd**)
2. Open Firefox and **goto: localhost:8888**, run Jupyter Lab interface
3. Navigate to: **`/home/intel/Workshop`**
4. Click on "**Intel Media SDK – Lab3.ipynb"**

Jupyter Notebook:

- Jupyter notebook is an interactive scripting environment with Markdown support.
- Code part is active and runs on its own environment settings.

What we will cover?

- Running tutorials implementing Intel® Media-SDK and Intel® Distribution of OpenVINO™
- Idea is to review Intel® Media SDK C++ API for Decoding, Encoding and VPP.

Intel(R) Media SDK Utilisation in Video Applications

In previous labs, we have focuced on DL inference on the video applications but a lot more going on a video analytics applications. Intel provides a lot more tool to enhace the process.

At this section, we will run 6 application developed with OpenCV, Inference Engine and Media SDK C++ APIs each named as `tutorial_0` to `tutorial_5`.

All the source code of these examples placed under `/home/intel/Tutorials/interop_tutorials` folder.

App Name	Decoding	Pre-Process	Inference	Post-Process	Encoding
Tutorial 0	OpenCV	OpenCV	OpenCV	OpenCV	OpenCV
Tutorial 1	OpenCV	OpenCV	OpenVINO	OpenCV	OpenCV
Tutorial 2	Media SDK	OpenCV	OpenVINO	OpenCV	OpenCV
Tutorial 3	Media SDK	Media SDK	OpenVINO	OpenCV	OpenCV
Tutorial 4	Media SDK	Media SDK	OpenVINO	Media SDK	OpenCV
Tutorial 5	Media SDK	Media SDK	OpenVINO	Media SDK	Media SDK

In this set of tutorails, we will investigate how Intel(R) Media SDK into an End to End Video Application.

All applications uses `SSD GoogleNet v2` for object recognition this time.