ISO/IEC JTC 1/SC 29/WG 03 N1044

**ISO/IEC JTC 1/SC 29/WG 03  
MPEG Systems   
Convenorship: KATS (Korea, Republic of)**

**Document type:** Output Document

**Title:** Technologies under consideration on ISO/IEC 23090-13 VDI

**Status:** Approved

**Date of document:** 2023-10-25

**Source:** ISO/IEC JTC 1/SC 29/WG 03

**No. of pages:** 14 (with cover page)

**Email of Convenor:** young.L @ samsung . com

**Committee URL:** <https://isotc.iso.org/livelink/livelink/open/jtc1sc29wg3>

**INTERNATIONAL ORGANIZATION FOR STANDARDIZATION**

**ORGANISATION INTERNATIONALE DE NORMALISATION**

**ISO/IEC JTC 1/SC 29/WG 03 MPEG SYSTEMS**

**ISO/IEC JTC 1/SC 29/WG 03 N1044**

**October 2023, Hanover, DE**

|  |  |
| --- | --- |
| **Title** | **Technologies under consideration on ISO/IEC 23090-13 VDI** |
| **Source** | **WG 03, MPEG Systems** |
| **Status** | **Approved** |
| **Serial Number** | **23197** |

# Introduction

This document collects technologies being under study for consideration in the development of the standard ISO/IEC 23090-13 Video Decoder Interface

# Table of content

[Introduction 1](#_Toc149144890)

[Table of content 1](#_Toc149144891)

[1. Improvements and fixes on the API extension definition (m62994) 1](#_Toc149144892)

[2. About a hypothetical reference systems decoder model 1](#_Toc149144893)

[3. Starting point for a HRD-like model for VDI 3](#_Toc149144894)

[4. Levels for the VDI system decoder model 10](#_Toc149144895)

[5. Fixes For Inserting Function 13](#_Toc149144896)

# Improvements and fixes on the API extension definition (m62994)

ISO/IEC 23090-13 defines extensions to video decoding engine (VDE) APIs in a generic manner to address the needs of immersive applications.

It has been identified some possible improvement and fixes on:

* additional structure definitions and their semantics
* editorial improvements
* improve definition of the API extensions

For references, the proposed text changes were provided in m62994 and are also attached with the present document.

# About a hypothetical reference systems decoder model

## A concrete use case

A concrete use case for using VDI is the usage of in an AR split rendering context. As an example, 4 bitstreams may be available

* A separate video stream for left and right eye
* An alpha channel for blending
* Depth information

The information then can directly processed in an XR Runtime that conforms to an OpenXR API. However, such an interface typically requires RGBA data, i.e. YUV data may be converted to RGB directly.

In order to be referenced in an external specification, VDI needs to be combined as follows:

* A *video decoding interface* shall be supported with the following parameters
  + “Profile”: sub-sampling, bit depth (example: 4:2:0, 10 bit)
  + Aggregate level capabilities: MB/s, bitrate, output constraints (level 6.1)
  + Number of instances (for example 16)
  + Codecs included: (AVC and HEVC), i.e. bitstreams may have different codecs
* Functionalities of the *video decoding interface*:
  + Decoding of conforming video bitstreams
  + Common coded picture buffer for a group of decoders
  + Associated access units across bitstreams provided at the same time to the coded picture buffer are provided synchronously at the output for a group of decoders
  + SEI and other metadata messages associated the access unit are provided synchronously at the decoder output
  + Conversion to colour spaces
* Signaling in content:
  + Max capabilities of each bitstream (regular profile/level signaling)
  + Group of coded bitstreams are associated to a group with max capabilities

## Summary of possible additions

On a high-level, the proposed additions are

* The VDI is only document from an API perspective.
* It addresses aggregate capabilities, but those are not related to a video decoding pipeline that included:
  + Scheduling of input data provided to the decoder
  + Memory and bandwidth of the decoder on the output
* The definition of a conforming set/group of bitstreams treated by one video decoding engine
* The definition of a conforming video decoding platform with multiple concurrent decoders
* The provisioning of signaling in order to annotate one bitstream or a set of bitstream for conformance
* The definition of a Hypothetical Reference Decoder for the Video Decoding interface
* The potential ability to add well-defined post-processing such as colour space transform
* The provisioning of associated data which each decoded picture, such as timing information or rendering poses

Concretely, in the VDI HRD:

* Inspired by the Hypothetical Reference Decoder of H.265/HEVC, the following are:
  + Elementary stream conformance
    - Single elementary stream conformance based on existing HRD (e.g. C.4 in HEVC)
    - Multiple elementary stream conformance, possibly to be extended based on the existing one defined in VDI in each codec instantiation (e.g. 7.3.2 Media and elementary stream constraints in VDI)
  + Decoder group
    - Single decoder conformance based on existing HRD (e.g. C.5 in HEVC)
    - Multiple decoder conformance, in the same group in VDI term, to be defined
* Profile/level restrictions apply to a set of concurrent decoders running in the same group
* Additional efforts
  + Creation of test and conformance test vectors for VDI
  + Creation of a test regime for a multi-decoder functionality

# Starting point for a HRD-like model for VDI

## Reminder of HEVC HRD

No tutorial of the complex HEVC HRD is provided, but the basic functionalities of the HRD are provided in Figure 1.

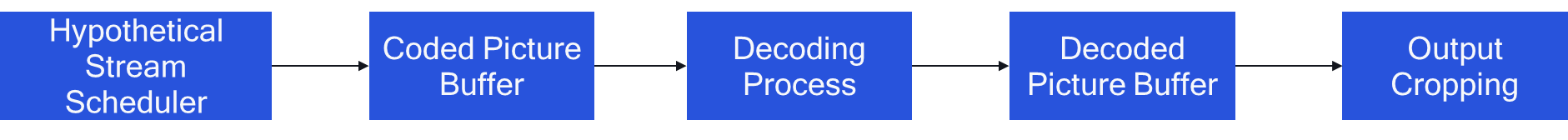


Figure HEVC HRD

The HRD operates as follows:

* The HRD is initialized at decoding unit 0, with both the CPB and the DPB being set to be empty (the DPB fullness is set equal to 0).
* Data associated with decoding units that flow into the CPB according to a specified arrival schedule are delivered by the hypothetical stream scheduler (HSS).
* The data associated with each decoding unit are removed and decoded instantaneously by the instantaneous decoding process at the CPB removal time of the decoding unit.
* Each decoded picture is placed in the DPB.
* A decoded picture is removed from the DPB when it becomes no longer needed for inter prediction reference and no longer needed for output.

The bitstream is tested by the HRD for conformance as specified in clause C.1

All of the following conditions shall be fulfilled

* The initial removal delay needs to be such that each AU is available in the buffer when it is removed.
* The CPB shall never overflow, i.e. the total number of bits in the CPB shall be smaller than or equal to CBP size
* In case of no low delay, the CPB shall never underflow, i.e.
  + AuNominalRemovalTime[ n ] >= AuFinalArrivalTime[ n ]
  + DuNominalRemovalTime[ m ] >= DuFinalArrivalTime[ m ]
* In case of low delay, if DuNominalRemovalTime[ m ] < DuFinalArrivalTime[ m ], then AuNominalRemovalTime[ n ] < AuFinalArrivalTime[ n ]
* Profile/level specific requirements for cpb removal times ávoiding to process too many samples at the output.
* The number of decoded pictures in the DPB shall be less than or equal to sps\_max\_dec\_pic\_buffering\_minus1[ HighestTid ]
* All reference pictures shall be present in the DPB when needed for prediction.
* Difference of maxPicOrderCnt − minPicOrderCnt is restricted
* Profile/level specific requirements for decoded picture output interval
* The decoding units need to match the access unit delays, but this seems to be buggy in the spec
* If DpbOutputTime[ m ] is greater than DpbOutputTime[ n ], then PicOrderCntVal[ m ] > PicOrderCntVal[ n ]

To check conformance of a decoder, test bitstreams conforming to the claimed profile, tier and level, as specified in clause C.4 are delivered by a hypothetical stream scheduler (HSS) both to the HRD and to the decoder under test (DUT).

All cropped decoded pictures output by the HRD shall also be output by the DUT, each cropped decoded picture output by the DUT shall be a picture with PicOutputFlag equal to 1, and, for each such cropped decoded picture output by the DUT, the values of all samples that are output shall be equal to the values of the samples produced by the specified decoding process.

For output timing decoder conformance, the HSS operates as described above, with delivery schedules selected only from the subset of values of SchedSelIdx for which the bit rate and CPB size are restricted as specified in Annex A for the specified profile, tier and level or with "interpolated" delivery schedules as specified below for which the bit rate and CPB size are restricted as specified in Annex A.

The same delivery schedule is used for both the HRD and the DUT.

In summary, the HRD specification is as follows

* For a bitstream conforming to a profile/level/tier and for given HRD parameters under HSS operation, the following holds
  + CBP does never overflow
  + CBP does never underflow
  + Maximum sample processing rate at the decoder output is not exceeded
  + Number of decoded pictures that need to be stored in DPB is not exceeded
  + The decoded picture output interval is not below a specified limit
* For a decoder conforming to a profile/level/tier and for given HRD parameters under HSS operation, any bitstream conforming to the HRD constraints shall be decodable

## Reminder of MPEG-4 and VDI System Decoder Model

The MPEG-4 System Decoder Model is defined as illustrated in Figure 2 below from ISO/IEC 144960-1.

A diagram of a decodeder

Description automatically generated

Figure 2 — Systems Decoder Model

The buffering model is using the conventions based on one branch of the SDM above a shown on Figure 4 below from ISO/IEC 144960-1.

A black and white logo

Description automatically generated

Figure 3 — Flow diagram for the systems decoder model

Legend:

|  |  |
| --- | --- |
| DB | Decoding buffer for the elementary stream. |
| CM | Composition memory for the elementary stream. |
| AU | The current access unit input to the decoder. |
| CU | The currentcomposition unit input to the composition memory. CU results from decoding AU. There may be several composition units resulting from decoding one access unit. |

## VDI HRD

### Key extensions

Assume there are N video streams that are concurrently decoded on the same device for the same application. Each of the N video streams follows some profile/level/tier requirements and HRD requirements and parameters specified along with the bitstream.

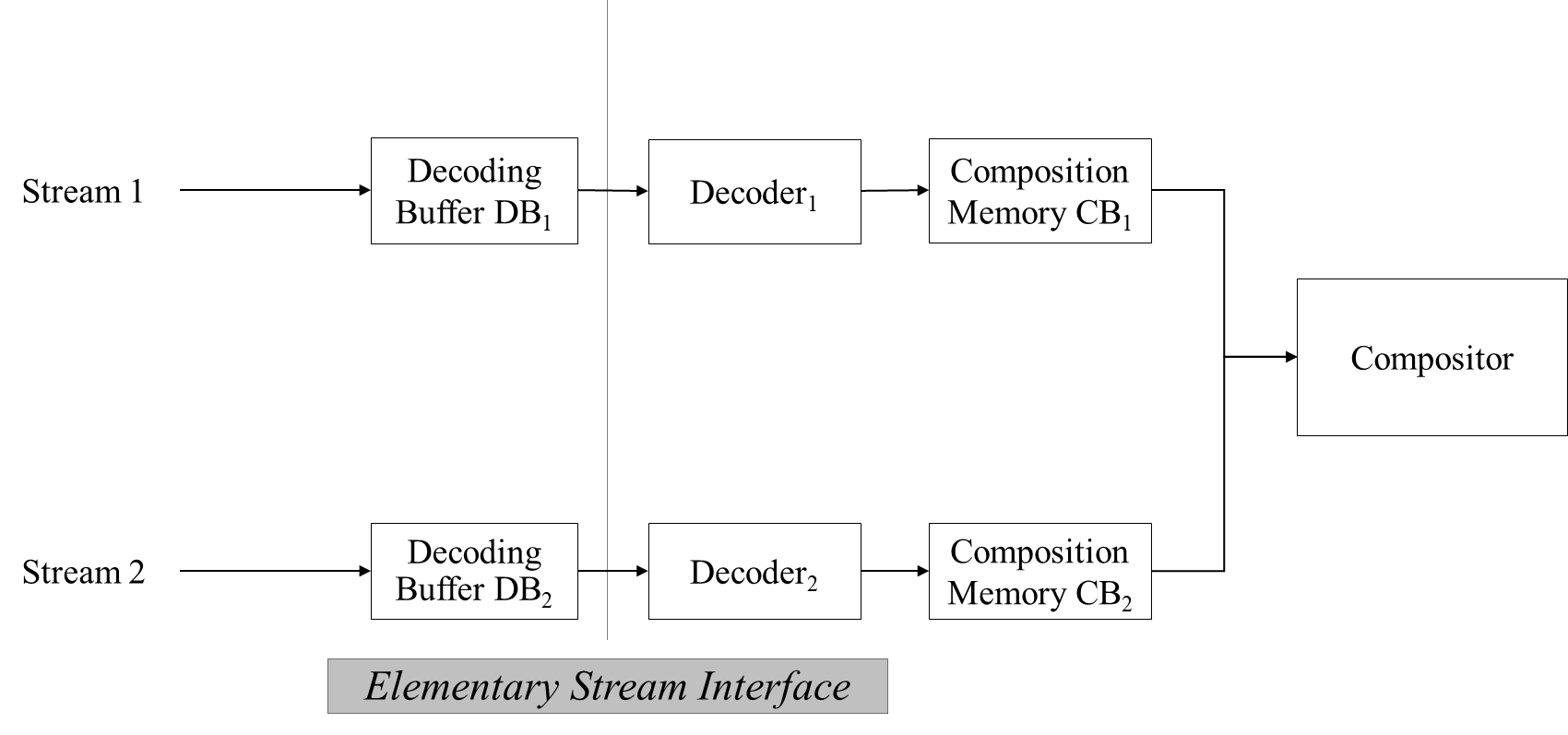
Three key models are to be considered

* Configuration 1: Decoding Buffer, Decoder and Composition Memory run independently for each of the N video streams.
  + Core Issue:
    - Aggregate bitrate, decoding complexity, Composition Memory size, and possibly other characteristics, must not exceed the limits of the VDE
* Configuration 2: Decoding Buffers are independent, but a common Composition Memory is used so that it ensures that the output of the N decoders can be provided with a proper timing in a synchronous manner to the next level:
  + Core Issues:
    - The core issue from configuration 1
    - the common Composition Memory not only follows the aggregate requirements, but also requires that all composition units (containing decoded pictures) need to be stored in the Composition Memory until every composition unit from all N decoders with the same composition time stamp has arrived
    - Each elementary stream any have different HRD parameters

NOTE The case where a common Decoding Buffer would be used for N video streams would violate the fundamental design of the SDM in MPEG-4 part 1 which is one Decoding Buffer is connected to a single Decoder and a Decoder can have one or more Decoding Buffers.

**Option 1: Individual Decoding Buffer and Composition Memory**

In Option 1, the Decoding Buffer and the Composition Memory for each elementary stream are treated individually. This is the case when the decoder inputs and outputs are operated independently which corresponds to the current situation without specific synchronization by the application.



*Figure 4 Option 1: Individual Decoding Buffer and Composition Memory*

The operation is as follows:

* Data associated with each access units flow into the Decoder of each elementary stream according to a specific arrival time. For each access unit:
  + all data associated with an access unit is removed from its Decoding Buffer and decoded instantaneously by the instantaneous decoding process at the time of the Decoding Time Stamp (DTS) of this access unit.
  + Each composition unit (from the decoding of an access unit) is guaranteed to be available in the Composition Memory (CM) at the time of Composition Time Stamp (CTS).
  + A composition unit is available up to the time of the Composition Time Stamp (CTS) of the next composition unit. After this time, the composition unit may replaced by the new composition unit.
* At any point time,
  + each of the individual elementary streams conforms to the signaled profile/level/tier and HRD parameters of the individual elementary stream
  + Aggregate bitrate, decoding complexity, Composition Memory size, and possibly other characteristics, must not exceed the limits of the VDE

Common HRD parameters for initial delay may be specified

* + - 1. **Option 2: Individual Decoding Buffer – joint Composition Memory**

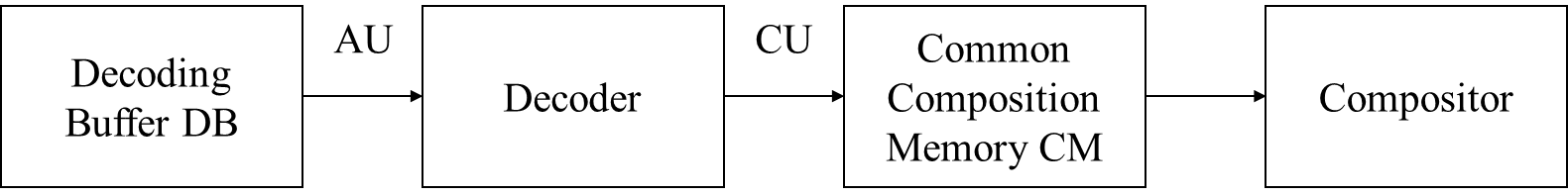
In Option 2, which is likely the most relevant one, a set of elementary streams has individual Decoding Buffers, but they share a common Composition Memory.

A black background with white text

Description automatically generated

*Figure 5 Individual Decoding Buffers – common Composition Memory*

Compared to the MPEG-4 part 1 flow diagram, the flow of data introduces a Common Composition Memory as opposed to the original Composition Memory:



NOTE that in this mode, each AU when decoded constitute a single CU as opposed to the MPEG-4 System Decoder

The operation is as follows

* Data associated with each access units flow into the Decoder of each elementary stream according to a specific arrival time. For each access unit:

all data associated with an access unit is removed from its Decoding Buffer and decoded instantaneously by the instantaneous decoding process at the time of the Decoding Time Stamp (DTS) of this access unit.

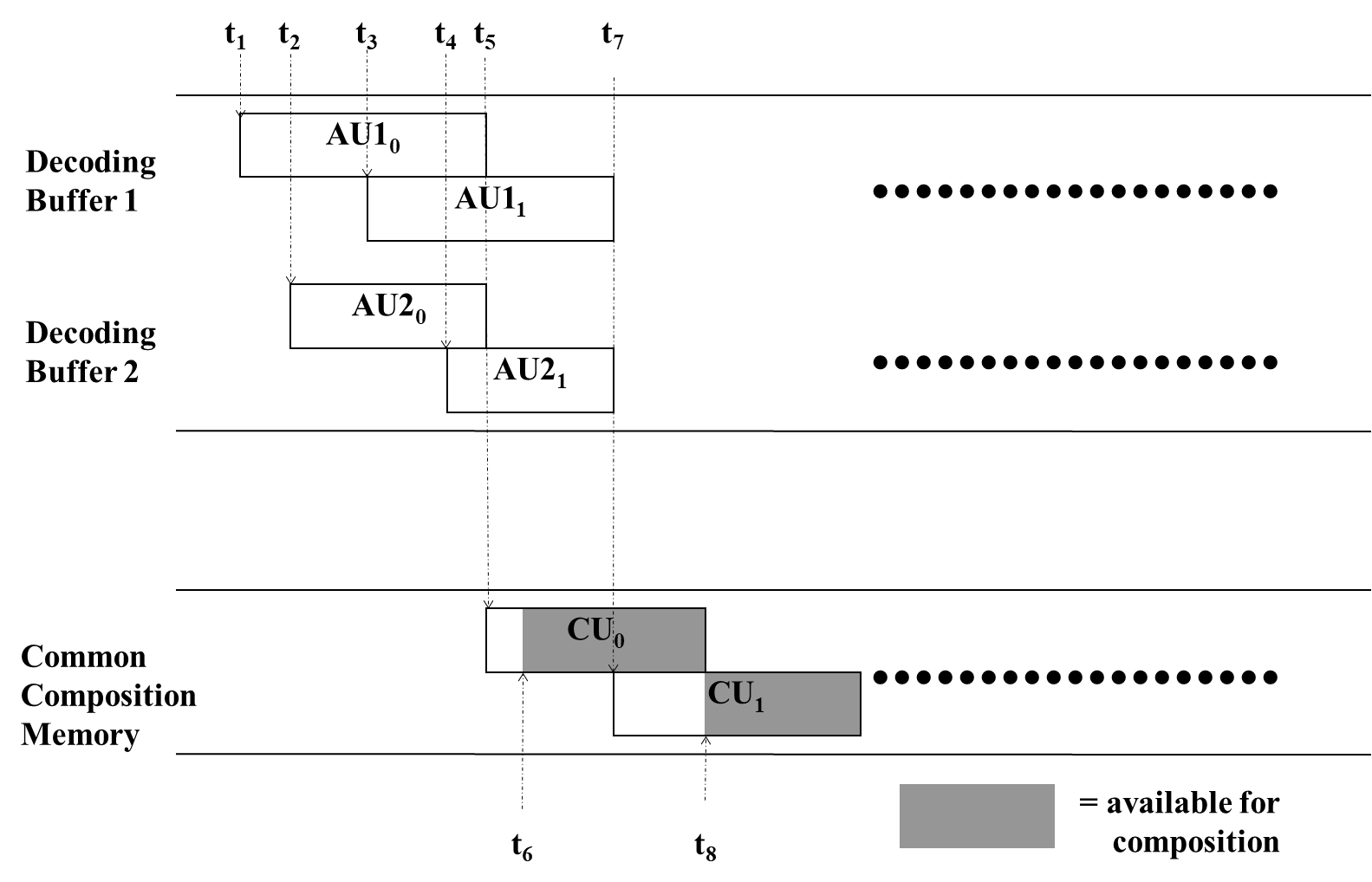
Each composition unit (from the decoding of an access unit) is guaranteed to be available in the Composition Memory (CM) at the time of Composition Time Stamp (CTS).

A composition unit is available up to the time of the Composition Time Stamp (CTS) of the next composition unit. After this time, the composition unit may replaced by the new composition unit. In addition, DTSs across all the access units as well as CTSs across all the composition units share the same clock reference (i.e. timeline).

* At any point time,
  + The same requirements as for configuration 1
  + The common Composition Memory size conforms to common aggregate requirements.
* Again common HRD parameters for initial delay may be specified

**Timing model**

The key aspect of the timing model is that the decoding time of the AU from the different elementary streams is identical. However, the Aus may have different arrival time in the respective Decoding Buffers.



Legend:

|  |  |
| --- | --- |
| t1 | Arrival time of AU10 in Decoding Buffer1 |
| t2 | Arrival time of AU20 in Decoding Buffer 2 |
| t3 | Arrival time of AU11 in Decoding Buffer 1 |
| t4 | Arrival time of AU21 in Decoding Buffer 2 |
| t5 | Decoding time of AU10  Decoding time of AU20 |
| t6 | Composition time of CU0 |
| t7 | Decoding time of AU11  Decoding time of AU21 |
| t8 | Composition time of CU1 |

* + - 1. **Elementary stream conformance**

The elementary stream conformance is defined as follows

* A set of elementary streams of coded data conforming to the specification shall fulfil all requirements specified in this clause
* The set of elementary stream is tested by the VDI HRD for conformance
* The set of elementary stream may be annotated with
  + Common profile/level/tier information
  + Common HRD parameters, in particular for initial delay

Whether they have a common or separate Composition Memory

* The set of elementary streams may even have different profiles or they may be different codecs even. In this case some common terminology across codecs needs to be defined.
  + - 1. **Elementary stream generation scenarios**

There are a couple of scenarios that may be considered for elementary stream generation

* Scenario 1: joint generation
  + Elementary streams are encoded with VDI decoding in mind
  + Overall HRD parameters need to be defined
  + Overall a set of encoders is controlled ensure that the common HRD parameters are maintained
* Scenario 2: individual generation
  + Elementary stream are encoded independently
  + Each elementary stream is annotated with individual profile/level/tier and HRD parameters
  + Additional information may be provided for each elementary stream to support joint decoding (for example decoded pictures)
  + From individual annotation, a common HRD operation may be derived by the decoder

Scenario 3: Extension to scenario 2 for which each elementary stream includes encoding metadata such that HRD parameters can be derived on the fly at the decoding instance.

# Levels for the VDI system decoder model

At the MPEG meeting #142, the Technologies under consideration on ISO/IEC 23090-13 VDI (N0903) **Error! Reference source not found.** was published with a clause on the hypothetical reference systems decoder model.

This hypothetical reference systems decoder model is a new concept on top of the VDI Video Decoding Engine such that performances of this engine can be defined similarly as the performances of a standardised video decoder.

This contribution proposes the following:

* Define the parameters of a VDE level aligned with the relevant parameters definition in AVC, HEVC, VVC, etc.
* Define three VDE levels as starting point based on levels corresponding to 1080p, 4K and 8K resolutions.

**Background**

* 1. ***Levels in HEVC***

**Table A.8 — General tier and level limits**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Level** | **Max luma picture size MaxLumaPs (samples)** | **Max CPB size MaxCPB (CpbVclFactor or CpbNalFactor bits)** | | **Max slice segments per picture MaxSlice Segments PerPicture** | **Max # of tile rows MaxTileRows** | **Max # of tile columns MaxTileCols** |
| **Main tier** | **High tier** |
| 1 | 36 864 | 350 | — | 16 | 1 | 1 |
| 2 | 122 880 | 1 500 | — | 16 | 1 | 1 |
| 2.1 | 245 760 | 3 000 | — | 20 | 1 | 1 |
| 3 | 552 960 | 6 000 | — | 30 | 2 | 2 |
| 3.1 | 983 040 | 10 000 | — | 40 | 3 | 3 |
| 4 | 2 228 224 | 12 000 | 30 000 | 75 | 5 | 5 |
| 4.1 | 2 228 224 | 20 000 | 50 000 | 75 | 5 | 5 |
| 5 | 8 912 896 | 25 000 | 100 000 | 200 | 11 | 10 |
| 5.1 | 8 912 896 | 40 000 | 160 000 | 200 | 11 | 10 |
| 5.2 | 8 912 896 | 60 000 | 240 000 | 200 | 11 | 10 |
| 6 | 35 651 584 | 60 000 | 240 000 | 600 | 22 | 20 |
| 6.1 | 35 651 584 | 120 000 | 480 000 | 600 | 22 | 20 |
| 6.2 | 35 651 584 | 240 000 | 800 000 | 600 | 22 | 20 |

**Table A.9 — Tier and level limits for the video profiles**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Level** | **Max luma sample rate MaxLumaSr (samples/sec)** | **Max bit rate MaxBR (BrVclFactor or BrNalFactor bits/s)** | | **Min compression ratio MinCrBase** | |
| **Main tier** | **High tier** | **Main tier** | **High tier** |
| 1 | 552 960 | 128 | - | 2 | 2 |
| 2 | 3 686 400 | 1 500 | - | 2 | 2 |
| 2.1 | 7 372 800 | 3 000 | - | 2 | 2 |
| 3 | 16 588 800 | 6 000 | - | 2 | 2 |
| 3.1 | 33 177 600 | 10 000 | - | 2 | 2 |
| 4 | 66 846 720 | 12 000 | 30 000 | 4 | 4 |
| 4.1 | 133 693 440 | 20 000 | 50 000 | 4 | 4 |
| 5 | 267 386 880 | 25 000 | 100 000 | 6 | 4 |
| 5.1 | 534 773 760 | 40 000 | 160 000 | 8 | 4 |
| 5.2 | 1 069 547 520 | 60 000 | 240 000 | 8 | 4 |
| 6 | 1 069 547 520 | 60 000 | 240 000 | 8 | 4 |
| 6.1 | 2 139 095 040 | 120 000 | 480 000 | 8 | 4 |
| 6.2 | 4 278 190 080 | 240 000 | 800 000 | 6 | 4 |

**Proposed VDE level definition**

* 1. ***Level definition template***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Level** | **Max aggregated luma picture size MaxAggLumaPs (samples)** | **Max aggregated bit rate MaxAggBR (1000 bits/s)** | **Max aggregated luma sample rate MaxAggLumaSr (samples/sec)** | **Max aggregrated DB size MaxAggBB (1000 bits)** | **Max number of decoder instances** |
| n | integer | integer | integer | integer | integer |

The levels are mainly used from an application to signal the requirements that the content has from an application. Therefore, by describing the content, a client application, using that description, will be able to identify whether the content is suitable for consumption, by means of comparing with the capabilities of the underlying device.

* 1. ***Proposed VDE levels***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Level** | **Max aggregated luma picture size MaxAggLumaPs (samples)** | **Max aggregated bit rate MaxAggBR (1000 bits/s)** | **Max aggregated luma sample rate MaxAggLumaSr (samples/sec)** | **Max aggregrated DB size MaxAggDB (1000 bits)** | **Max number of decoder instances** |
| 1.0 | 2 073 600 | 50 000 | 133 693 440 | 50 000 | 2 |
| 2.0 | 8 294 400 | 160 000 | 534 773 760 | 160 000 | 2 |
| 2.1 | 8 294 400 | 160 000 | 534 773 760 | 160 000 | 4 |
| 3.0 | 35 389 440 | 800 000 | 4 278 190 080 | 800 000 | 2 |
| 3.1 | 35 389 440 | 800 000 | 4 278 190 080 | 800 000 | 4 |
| 3.2 | 35 389 440 | 800 000 | 4 278 190 080 | 800 000 | 8 |

NOTE 1 The following HEVC levels were used to derive aggregated parameters:

* Level 1: Level 4.1 (high tier) @ 1,920×1,080
* Level 2: Level 5.1 (high tier) @ 3,840×2,160
* Level 3: Level 6.2 (high tier) @ 8,192×4,320

The Max aggregated luma picture size in HEVC is calculated to be dividable by 16, this is a constraint that is ignored as not relevant for the VDE levels.

NOTE 2 The number of instance has been defined in such a way that each instance may operate at minimum at 30fps - 1080p..

# Fixes For Inserting Function

[Editor’s Note: This Section demonstrates a problem in the definitions and invites for further study. Two possible lines of actions are documented, but not a solution per ce.]

There is an inconsistency between the definition of the inserting function and the constraints on the media streams resulting of this inserting function.

* 1. ***Problem***

**Problem**

In clause 7.2.4, the inserting function is defined as the merging of two input media streams. The end of the process is further defined as “ The inserting operation stops as soon as one of the two input media streams ends.”

However, the media streams constraints for the inserting function are defined as follows:

*(In clause 7.4.2.2.2 Constraints for the inserting function)*

A VVC media stream generated as output of the inserting function shall comply to these rules in addition to the rules in 7.2.2:

* The number of VCL NAL units in the output media stream is equal to the sum of the number of VCL NAL units in both input media streams.
* For each VCL NAL unit in the output media stream, there shall exist a VCL NAL unit in of one of the two input media streams that is bit exact identical.

The same goes for other codec bindings.

* 1. ***Proposal***

To solve the inconsistency, there exist thus two options:

* Option 1: The inserting function ends with the longest input media stream
* Option 2: The output media streams length constraint align on the shortest input media stream