 ISO/IEC JTC 1/SC 29/WG 3 N0218

**ISO/IEC JTC 1/SC 29/WG 3**

**MPEG Systems   
Convenorship: KATS (Korea, Republic of)**

**Document type:** Output Document

**Title: Systems Functionalities for Video Conformance**

**Status:** Approved

**Date of document:** 2021-04-30

**Source:** ISO/IEC JTC 1/SC 29/WG 3

**Expected action: ACT**

**Action due date:** 2021-04-30

**No. of pages:** 36 (with cover page)

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

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

**INTERNATIONAL ORGANISATION FOR STANDARDISATION**

**ORGANISATION INTERNATIONALE DE NORMALISATION**

**ISO/IEC JTC 1/SC 29/WG 3**

**CODING OF MOVING PICTURES AND AUDIO**

**ISO/IEC JTC 1/SC 29/WG 3 N** **00131**

**Online – October 2020**

# Introduction

This document collects the agreements of the BoG on Systems Functionalities for Video Conformance.

Contents

[1 Introduction 2](#_Toc70372447)

[2 Definitions 4](#_Toc70372448)

[2.1 Switching Workflow 4](#_Toc70372449)

[2.2 DASH Definitions: Media stream and Representation properties 5](#_Toc70372450)

[2.2.1 Switching and Random Access Support 5](#_Toc70372451)

[2.2.2 Media stream access points 6](#_Toc70372452)

[2.2.3 Non-overlapping Segments and Subsegments 7](#_Toc70372453)

[2.2.4 Conforming Segment track 7](#_Toc70372454)

[2.3 Conformance on bitstream level for HRD 7](#_Toc70372455)

[2.4 Splicing 8](#_Toc70372456)

[2.4.1 Bitstream Splicing 8](#_Toc70372457)

[2.4.2 Header Splicing 8](#_Toc70372458)

[3 Collection of Issues 8](#_Toc70372459)

[3.1 Template <Title> 8](#_Toc70372460)

[3.1.1 Background and Motivation 8](#_Toc70372461)

[3.1.2 Specification (according to ISO/IEC 23003-15) 8](#_Toc70372462)

[3.1.3 Recommendations for Systems Standards 8](#_Toc70372463)

[3.2 Clean Switch 8](#_Toc70372464)

[3.2.1 Background and Motivation 8](#_Toc70372465)

[3.2.2 General description 9](#_Toc70372466)

[3.2.3 What needs to be done 10](#_Toc70372467)

[3.2.4 Nice to have 10](#_Toc70372468)

[3.2.5 Specification (according to ISO/IEC 23090-15) 10](#_Toc70372469)

[3.2.6 Recommendations/Requirements for Systems Standards 10](#_Toc70372470)

[3.3 Fast Random Access 10](#_Toc70372471)

[3.3.1 General description 10](#_Toc70372472)

[3.3.2 General constraints 12](#_Toc70372473)

[3.3.3 What needs to be done 12](#_Toc70372474)

[3.3.4 References 12](#_Toc70372475)

[3.3.5 Nice to have 12](#_Toc70372476)

[3.3.6 Specification (according to ISO/IEC 23090-15) 12](#_Toc70372477)

[3.3.7 Recommendations for Systems Standards 12](#_Toc70372478)

[3.4 Unclean Switch 12](#_Toc70372479)

[3.4.1 General description 12](#_Toc70372480)

[3.4.2 General constraints 13](#_Toc70372481)

[3.4.3 Discussions 13](#_Toc70372482)

[3.4.4 Specification (according to ISO/IEC 23090-15) 14](#_Toc70372483)

[3.4.5 Recommendations/Requirements for Systems Standards 14](#_Toc70372484)

[3.5 Splicing 14](#_Toc70372485)

[3.5.1 General description 14](#_Toc70372486)

[3.5.2 General constraints 15](#_Toc70372487)

[3.5.3 Next steps 15](#_Toc70372488)

[3.5.4 Specification (according to ISO/IEC 23090-15) 15](#_Toc70372489)

[3.5.5 Recommendations/Requirements for Systems Standards 15](#_Toc70372490)

[3.6 Efficient Splicing 15](#_Toc70372491)

[3.6.1 General description 15](#_Toc70372492)

[3.6.2 General constraints 16](#_Toc70372493)

[3.6.3 Specification (according to ISO/IEC 23090-15) 16](#_Toc70372494)

[3.6.4 Recommendations/Requirements for Systems Standards 16](#_Toc70372495)

[4 Common Envelope 16](#_Toc70372496)

[4.1 Introduction 16](#_Toc70372497)

[4.2 Option 1: Signalling of decoder initialization information in file formats, DASH, etc 16](#_Toc70372498)

[4.2.1 Introduction 16](#_Toc70372499)

[4.2.2 DII signalling in the ISOBMFF 17](#_Toc70372500)

[4.2.3 DII signalling in DASH MPD 18](#_Toc70372501)

[5.13.X Video Decoder Initialization Information 20](#_Toc70372502)

[4.3 Option 2: Signalling of decoder initialization information in file formats, DASH, etc 21](#_Toc70372503)

[4.3.1 Abstract 21](#_Toc70372504)

[4.3.2 Introduction 21](#_Toc70372505)

[4.3.3 Proposal 24](#_Toc70372506)

[**11.9.4 VVC switchable tracks entity group** 25](#_Toc70372507)

[**11.9.4.1 General** 25](#_Toc70372508)

[**11.9.4.2 Syntax** 25](#_Toc70372509)

[**11.9.4.3 Semantics** 25](#_Toc70372510)

[4.3.4 **11.8.X Parameter set sample group** 25](#_Toc70372511)

[**4.3.5** **11.8.X.1 Definition** 25](#_Toc70372512)

[4.3.6 **11.8.X.2 Syntax** 26](#_Toc70372513)

[4.3.7 **11.8.X.3 Semantics** 26](#_Toc70372514)

[**1.1.** **Adding DPB information into the decoding capability information sample group** 26](#_Toc70372515)

[**11.8.10** **Decoder capability information sample group** 26](#_Toc70372516)

[5 Stream Switching Background 27](#_Toc70372517)

[5.1 Summary 27](#_Toc70372518)

[5.2 Problems that motivated the development of stream switching 27](#_Toc70372519)

[5.3 Multi-rate streaming with stream switching 28](#_Toc70372520)

[5.4 Alternative and Complementary Techniques 29](#_Toc70372521)

[5.4.1 Pre-roll buffer 29](#_Toc70372522)

[5.4.2 Real-time encoding/transcoding with bitrate adaptation 30](#_Toc70372523)

[5.4.3 Stream “thinning” 30](#_Toc70372524)

[5.4.4 The use of scalable coding 31](#_Toc70372525)

[5.4.5 Reference picture resampling 31](#_Toc70372526)

[5.4.6 “Dirty” switches 32](#_Toc70372527)

[5.4.7 S frames 32](#_Toc70372528)

[5.4.8 By-directional S frames (1998) 33](#_Toc70372529)

[5.4.9 SI and SP frames 33](#_Toc70372530)

[5.4.10 HESP 34](#_Toc70372531)

[5.5 Conclusions 34](#_Toc70372532)

[5.6 References 35](#_Toc70372533)

[5.7 Meeting Discussion 36](#_Toc70372534)

# Definitions

## Switching Workflow

In a system, multiple encodings are generated such that they can be switched in way that it is observed seamless. By doing so, multiple elementary streams of the same source content are encoded under certain constraints and form multiple tracks.

There exists an envelope signal for all of the tracks that permits initialization of the decoder such that switching can be done seamlessly.

Switching can be done by different means

* *Bitstream Switching*:
  + after initializing with the Initialization Segment/CMAF Header of the switch-from Representation/CMAF Track A, switching is possible at the switch point *t* by decoding and presenting switch-from Representation/CMAF Track A up to switch point *t*, and continue decoding and presenting the switch-to Representation/CMAF Track B from time *t* onwards.
  + conformance on bitstream level after switch on bitstream and timing/HRD
* *Header Switching*:
  + after initializing with the Initialization Segment/CMAF Header of the switch-from Representation A/CMAF Track A, switching is possible at the switch point *t* by decoding and presenting switch-from Representation/CMAF Track A up to switch point *t*, initializing the switch-to Representation/CMAF Track B with the associated Initialization Segment/CMAF Header from Representation/CMAF Track B and continue decoding and presenting the switch-to Representation/CMAF Track B from time *t* onwards.
  + Unclear what conformance is required here

## DASH Definitions: Media stream and Representation properties

### Switching and Random Access Support

A key functionality is the ability of the DASH Client to seamlessly switch across different Representations of the same media component without severely impacting the user experience.

Assume two Representations A and B. A switch from Representation A to Representation B at media time t is considered seamless, if the result of the presentation after this switch is applied is the same as if Representation A was decoded from the beginning and presented up to time t and Representation B is decoded from the beginning and presented from time t onwards.

Media Presentations may provide different Representations in one Adaptation Set representing the same media component. If such Representations are properly time-aligned (as expected by the Media Presentation), then DASH Clients may apply seamless switching across different Representations provided in one Adaptation Set at any time t to obtain a perceptually continuous experience.

However, in practical implementations, the operation of seamless switching can be complex, as switching at time t can require parallel download and decoding of two Representations. Therefore, providing suitable switching opportunities in regular time intervals simplifies client implementations. ISO/IEC 23009-1 provides means for providing suitable switching opportunities and in addition provides abilities to signal the position and media time of the switching opportunities.

For this purpose, three relevant concepts are defined to support seamless switching:

* Media stream access points in subclause 4.5.2 to signal positions where to easily switch to a Representation, and in addition where to suitable access a Representation at start-up or seek.
* Non-overlapping Segments and Subsegments in subclause 4.5.3 to signal that, at the signalled stream access points, no overlap decoding of Representations is necessary in order to provide a continuous switch.
* Segment concatenation in subclause 4.5.4 to signal that the concatenation of two Representations at a switch point results in a conforming bitstream.

These three properties are neither sufficient nor necessary for seamless switching, but certain implementation or profiles may use these properties in order to simplify practical implementations.

### Media stream access points

To be able to access a Representation, each of the media streams that are contained in the Representation requires media stream access points (SAPs). SAPs in the context of this document refer to the SAP definition in ISO/IEC 14496-12, Annex I. ISO/IEC 14496-12, Annex I.3 defines different types of SAPs that provide a relationship between the position where a stream can be accessed, relative to the start of a Segment or Subsegment, its presentation time and the presentation times and position of other access unit in the stream. The same SAP type definitions shall apply for this document.

A SAP is a position in a Representation that enables playback of a media stream to be started using only the information contained in Representation data starting from that position onwards (preceded by initializing data in the Initialization Segment, if any).

For each SAP, the properties, ISAP, TSAP, ISAU, TDEC, TEPT, and TPTF, are identified and defined in ISO/IEC 14496-12, Annex I.2.

In particular, TSAP is defined to be the earliest presentation time of any access unit of the media stream such that all access units of the media stream with presentation time greater than or equal to TSAP can be correctly decoded using data in the Representation starting at byte position ISAP and no data before ISAP.

NOTE The type of SAP is dependent only on which access units are correctly decodable and their arrangement in presentation order. The types informally correspond with some common terms:

— Type 1 corresponds to what is known in some coding schemes as a “Closed GoP random access point” (in which all access units, in decoding order, starting from ISAP can be correctly decoded, resulting in a continuous time sequence of correctly decoded access units with no gaps) and in addition the access unit in decoding order is also the first access unit in presentation order.

— Type 2 corresponds to what is known in some coding schemes as a “Closed GoP random access point”, for which the first access unit in decoding order in the media stream starting from ISAU is not the first access unit in presentation order.

— Type 3 corresponds to what is known in some coding schemes as an “Open GoP random access point”, in which there are some access units in decoding order following ISAU that cannot be correctly decoded and have presentation times less than TSAP.

— Type 4 corresponds to what is known in some coding schemes as a "gradual decoder refresh (GDR) random access point”, in which there are some access units in decoding order starting from and following ISAU that cannot be correctly decoded and have presentation times less than TSAP.

— Type 5 corresponds to the case for which there is at least one access unit in decoding order starting from ISAP that cannot be correctly decoded and has presentation time greater than TDEC and where TDEC is the earliest presentation time of any access unit starting from ISAU.

— Type 6 corresponds to the case for which there is at least one access unit in decoding order starting from ISAP that cannot be correctly decoded and has presentation time greater than TDEC and where TDEC is not the earliest presentation time of any access unit starting from ISAU.

SAPs are mostly relevant for two purposes in this document:

1) For randomly accessing a Media Presentation, for example at the startup of the Media Presentation, after a seeking operation or after an error event especially in live cases.

2) To permit switching between two Representations whereby for seamless switching each media stream *i* in the switch-from Representation is presented up to TSAP(*i*) and each media stream *i* in the switch-to Representation is presented from the media stream access point starting from TSAP(*i*).

There are obvious benefits for the client to be able to identify SAPs and one or several of their properties, in particular ISAP and TSAP for each media stream without requiring to access data at positions following ISAP. DASH provides functionalities to explicitly signal such information by using signals in the MPD or the Segment Index or combinations of the two.

### Non-overlapping Segments and Subsegments

Segments and Subsegments represent units for which the client has an exact map on how to access and download the unit using HTTP GET or HTTP partial GET methods.

Segments (respectively Subsegments) are typically generated by segmenting encoded media streams into appropriate units. If the generation of Segments (respectively Subsegments) adheres to certain rules, then the sequential decoding and presentation of Media Segments (respectively Subsegments) results in a correct presentation of all contained media streams. To define such rules the notion of “non-overlapping” segments (respectively Subsegments) is defined as follows.

Let

— *TE*(*S,i*) be the earliest presentation time of any access unit in stream *i* of a Segment or Subsegment *S*, and

— *TL*(*S,i*) be the latest presentation time of any access unit in stream *i* of a Segment or Subsegment *S*.

Then two segments (respectively Subsegments), *A* and *B*, which may or may not be of different Representations, are *non-overlapping* if *TL*(*A,i*) < *TE*(*B,i*) for all media streams *i* in A and B or if *TL*(*B,i*) < *TE*(*A,i*) for all streams *i* in A and B where *i* refers to the same media component.

The property of “non-overlapping” segments (respectively Subsegments) is used to define the terms Segment alignment and Subsegment alignment.

### Conforming Segment track

A sequence of Segments (respectively Subsegments) is a “conforming Segment (respectively Subsegment) track” if the concatenation of all Segments (respectively Subsegments) in the sequence of Segments (respectively Subsegments) results in a bitstream that conforms to the media formats in use (including container and codecs).

NOTE This implies that a player conforming to the media format can play the resulting bitstream.

## Conformance on bitstream level for HRD

In order to avoid underflow/overflow there are three options

* Encoder preparation constrain streams and jointly encode.
* increase CPB, but also encoder option to constrain fluctuation
* client level and use buffering period SEI

can we send a conforming bitstream by sending buffer period SEI at every switch point, but decoder will not read it.

## Splicing

### Bitstream Splicing

The bitstream after splicing is such that a conforming bitstream is achieved by concatenation of the bitstream with the bitstream after the splice, possibly with a “timeline discontinuity”.

Note: splicing is not time-continuous

### Header Splicing

The bitstream after splicing is such that a conforming bitstream is achieved by concatenation of the bitstream by adding information from the switch header to the bitstream after the splice, possibly with a “timeline discontinuity”.

Note: splicing is not time-continuous

<Editor’s Note: rename to continuous and continuous, not use splicing>

# Collection of Issues

## Template <Title>

### Background and Motivation

### Specification (according to ISO/IEC 23003-15)

#### Functional Stage

#### Purpose

#### Type

#### Test and Observation

#### Additional Recommendations

### Recommendations for Systems Standards

## Clean Switch

### Background and Motivation

When receiving a live stream in adaptive streaming, transitions between representations shall be done without service interruption. From a video decoder viewpoint, the reinitialization of the decoding instance should be avoided. There is a need to build conformant bitstreams that integrate different transitions in parameters such as resolutions or frame rate.

As an illustration of the scenario, the figure 3.3-1 represents two different representations (1080p and 720p resolution) of a video content with an aligned GOP structure on RAP locations.

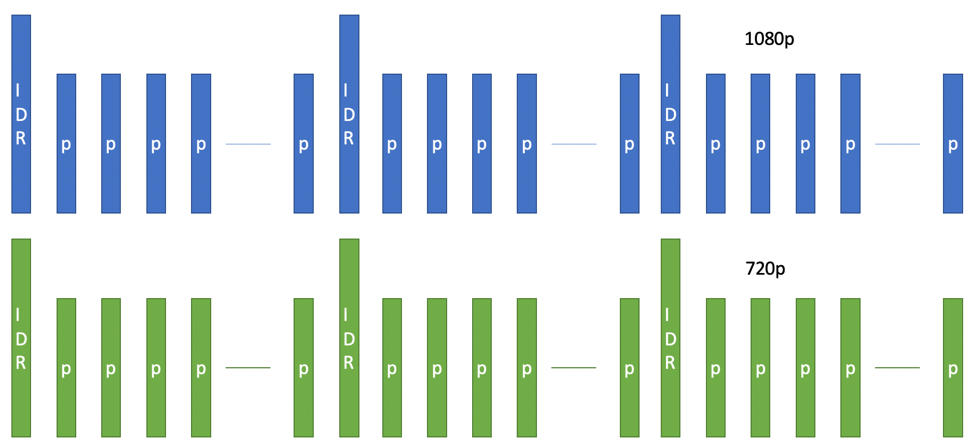


Figure 3.3-1: GOP structure of a 1080p and 720p representation

When switching from one representation to the other, the resulting bitstream from a video decoder viewpoint would become the junction of the 2 representations at the selected switching point as shown in figure 3.3-2.

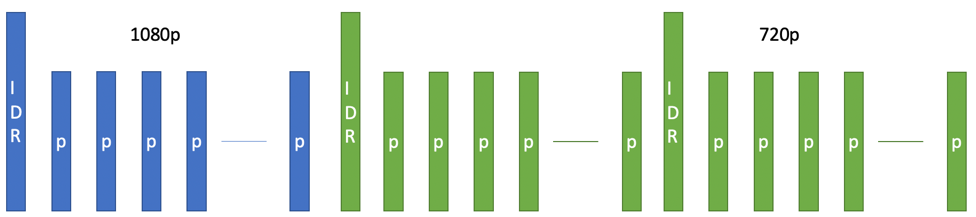


Figure 3.3-2: Resulting bitstream combining 2 representations with different properties

When switch between 2 representations with different properties, the reinitialization of the decoder instance can only avoided if all the necessary guarantees are provided in advance such as the dimension of the CPB and DPB. The parameter sets associated to the second stream have to be known in advance.

### General description

Bitstream Switching

* Create a bitstream 1 with regular IDR frames
* Create a bitstream 2 with regular IDR frames at the same position with the same content
* Decode bitstream 1 up to a frame and decode bitstream 2 from an IDR frame
* Check the concatenated bitstream for conformance
* Provide such a conformance bitstream

Header Switching

* Create a bitstream 1 with regular IDR frames
* Create a bitstream 2 with regular IDR frames at the same position with the same content
* Create a specific header from bitstream 2 that can inserted when switching to bitstream 1 from bitstream 1.
* Decode bitstream 1 up to a frame and decode bitstream 2 from an IDR frame
* Check the two bitstreams for conformance
* Provide such a conformance bitstream

### What needs to be done

Some open questions:

* Requirements/Constraints for blue/green streams
  + At initialization, you need to take into account
* We document how to do it right and we create bitstreams for conformance that even work.
* What are switches we want to test
  + Same resolutions
  + Different resolutions
  + Different framerates
  + Different bitrates (i.e. HRD Buffer)

### Nice to have

* Put guidelines for encoders
* We can create requirements for tracks in CMAF switching sets that follow the above guidelines.
* By doing CMAF we get also DASH and hopefully HLS.

### Specification (according to ISO/IEC 23090-15)

#### Functional Stage

#### Purpose

#### Type

#### Test and Observation

#### Additional Recommendations

### Recommendations/Requirements for Systems Standards

## Fast Random Access

### General description

When accessing a live stream, the receiver needs to wait for the next available Random-Access-Point (RAP) so as to be able to start decoding the stream. The RAP period has a direct impact of both the visual quality (has to be not too short) and the decoding start delay (has to be not too long).

With adaptive streaming, it is possible to significantly reduce the decoding starting delay by constructing a representation containing only RAPs (e.g. IDR frames) at a shorter RAP period than the main representation RAP period.

The figure 3.2-1 illustrates a generic GOP structure for the main representation to be accessed.

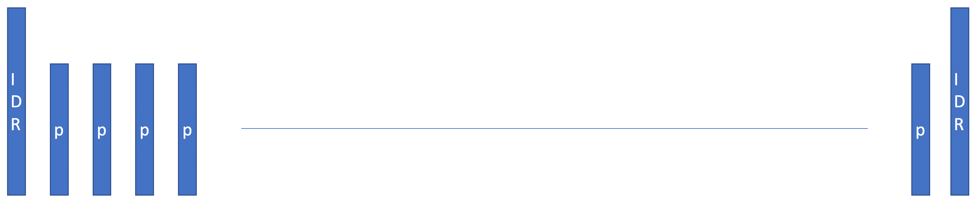


Figure 3.2-1: Generic GOP structure of the main representation

The figure 3.2-2 illustrates a fast access representation containing only RAPs.

Une image contenant texte, pelle, outil

Description générée automatiquement

Figure 3.2-2: Fast access representation with shorter RAP period

In order to be able to start with any IDR from the fast access representation and continue decoding the pictures from the main representation, the latter needs to be constructed so as to ensure conformance form a video decoder viewpoint.

For this reason, the concept of fast-RAP (fRAP) picture is introduced and illustrated in Figure 3.2-3 below.

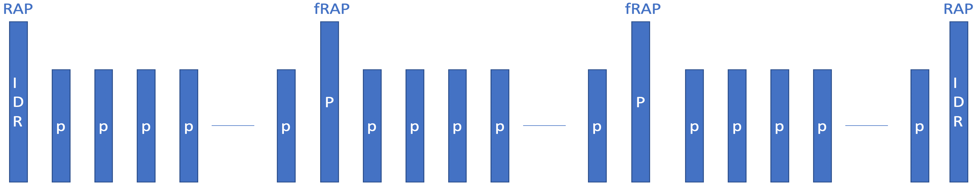


Figure 3.2-3: GOP structure of the main representation with regular fRAP pictures

With such a configuration, the accessing delay to the stream is significantly reduced while the bitrate consumption of the main representation is not impacted by too many Intra pictures as illustrated in the last figure 3.2-4 below.

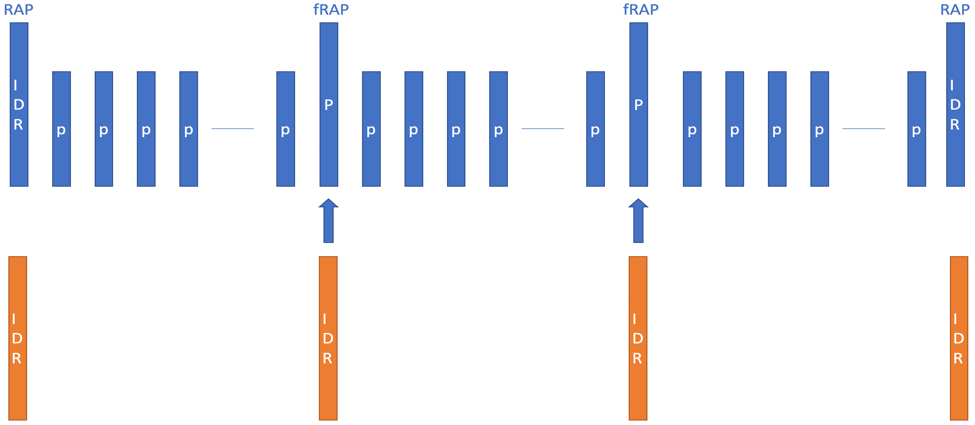


Figure 3.2-4: Possible switching points from the Fast access representation to fRAP pictures of the main representation.

Therefore, the resulting bitstream to be generated for the conformance verification will look like the figure 3.2-5.

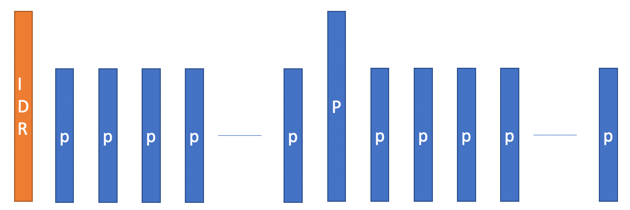


Figure 3.2-5: Resulting bitstream from the use of an independent IDR in place of a fRAP.

### General constraints

The switch between 2 representations without decoder reinitialization implies functional and interdependencies constraints. A first and non-exhaustive consideration of those constraints is listed below:

Functional

* SPS and PPS shall be the same
* SEI messages shall be coherent
* HRD needs to also conformant

Interdependencies:

* pictures following the fRAP shall not refer to reference frames preceding the fRAP.

### What needs to be done

Some open questions:

* Requirements/Constraints for blue streams
  + Prediction structure (no reference prior to fRAP)
  + Temporal motion vector prediction cannot be used across
  + HRD
* Requirements for orange streams
* Requirements for system combining streams
* What different cases do we want to test
  + Orange and blue have same resolution
  + Orange and blue have different resolution

### References

* this is another document that may be referenced: M54752
* B. Girod, et al, “Scalable codec architectures for Internet video-on-demand,” ACSSC, pp. 357 – 361, 1997

### Nice to have

* Can we check what bitrate savings we get

Interested additional people: BD, YR/KL

### Specification (according to ISO/IEC 23090-15)

#### Functional Stage

#### Purpose

#### Type

#### Test and Observation

#### Additional Recommendations

### Recommendations for Systems Standards

## Unclean Switch

### General description

In an adaptive streaming scenario, the switches between representations are by default constrained to locations corresponding to RAPs. More efficient switch can be performed with shorter GOP size (i.e. shorter RAP periods) but it comes at the price of more bitrate consumption by I frames.

This scenario proposes to rely on fast-RAP pictures (fRAP) that can be used as switching points between representations.

The example in figure 3.4-1 considers a scenario in which 2 representations of the same content are available at different qualities.

Une image contenant texte, clôture

Description générée automatiquement

Figure 3.4-1: Two representations implementing fRAP switching points

With such a configuration in place, switching points can be considered at the fRAP locations as shown in figure 3.4-2.

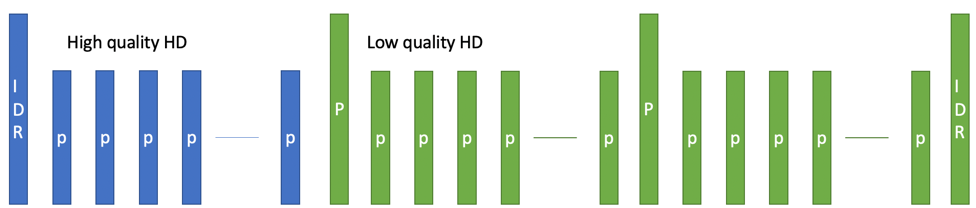


Figure 3.4-2: Combination of 2 representation at an fRAP switching point

It is accepted and understood that the decoded pictures from the switching point at the fRAP until the next IDR are likely to provide visual artefacts but without any decoding process interruption.

### General constraints

Without any RAP between the representations, SPS, PPS are required to be the same. CPB level (and HRD parameters) need to be investigated so as to avoid the video decoder to run over/underflow…

### Discussions

* Obviously creates a mismatch, but service providers would allow it, or not allows it.
* May not have the highest priority.
* Similar aspects are done above need to be taken care of

### Specification (according to ISO/IEC 23090-15)

#### Functional Stage

#### Purpose

#### Type

#### Test and Observation

#### Additional Recommendations

### Recommendations/Requirements for Systems Standards

## Splicing

### General description

In many service scenarios, the need for switching between contents is required, such as for ad-insertion, local content distribution.

In this case, for example a main live stream is distributed. The bitstream includes splice points for which a downstream network node or the client do replacements of content and splicing. The splice point is signalled. The splice point is supported by a clean IDR. The concatenated bitstream is now produced and in an ideal case, this bitstream is conforming. It is important to understand to test different configurations.

In this case, splicing points are identified so as to ensure a frame accurate and continuity of service as illustrated in figure 3.5-1.

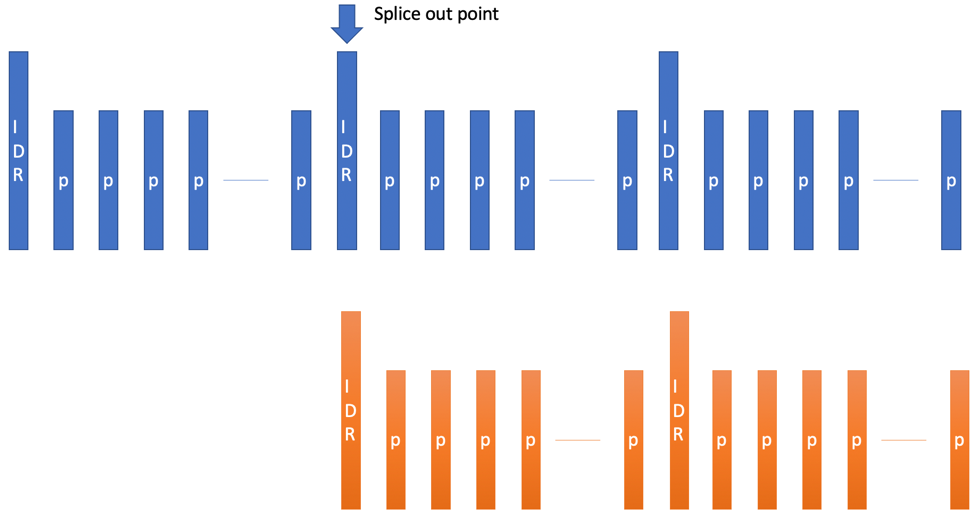


Figure 3.5-1: Splicing point at an IDR

From a video decoder perspective, the continuity of service is guaranteed by the continuity of decoding process of a single bitstream as shown in figure 3.5-2.

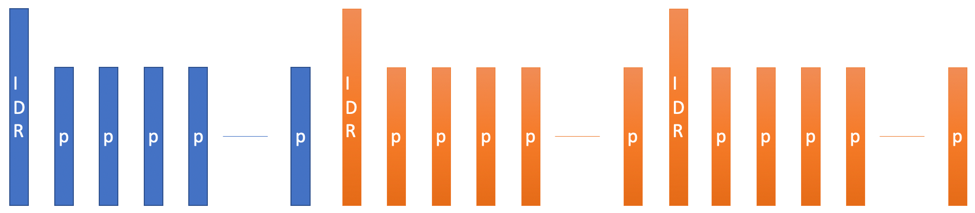


Figure 3.5-2: Resulting bitstream from a splicing

### General constraints

Consistency between the two combined streams is necessary such as the video formats (resolution, frame rate, colour space and dynamic range) and video codec parameters (profile, level) that impact the buffers size initially allocated.

Another point of consideration is the update of the buffer models defined in the HRD since the second bitstream has no knowledge of the previous decoder buffer states at the splicing point.

Some open questions:

* Requirements for blue streams
* Requirements for orange streams
* Requirements for system combining streams

### Next steps

* Not a lot of difference to clean switching
* Main differences:
  + different content
  + concept of independent encoding and splicing
  + This is for example documented in
    - CTA WAVEs Splice Conditioning spec
    - DASH-IF ad insertion document
    - DASH profile for CMAF content

### Specification (according to ISO/IEC 23090-15)

#### Functional Stage

#### Purpose

#### Type

#### Test and Observation

#### Additional Recommendations

### Recommendations/Requirements for Systems Standards

## Efficient Splicing

### General description

In another case, the splice-out point is not an IDR. This may change the blue stream

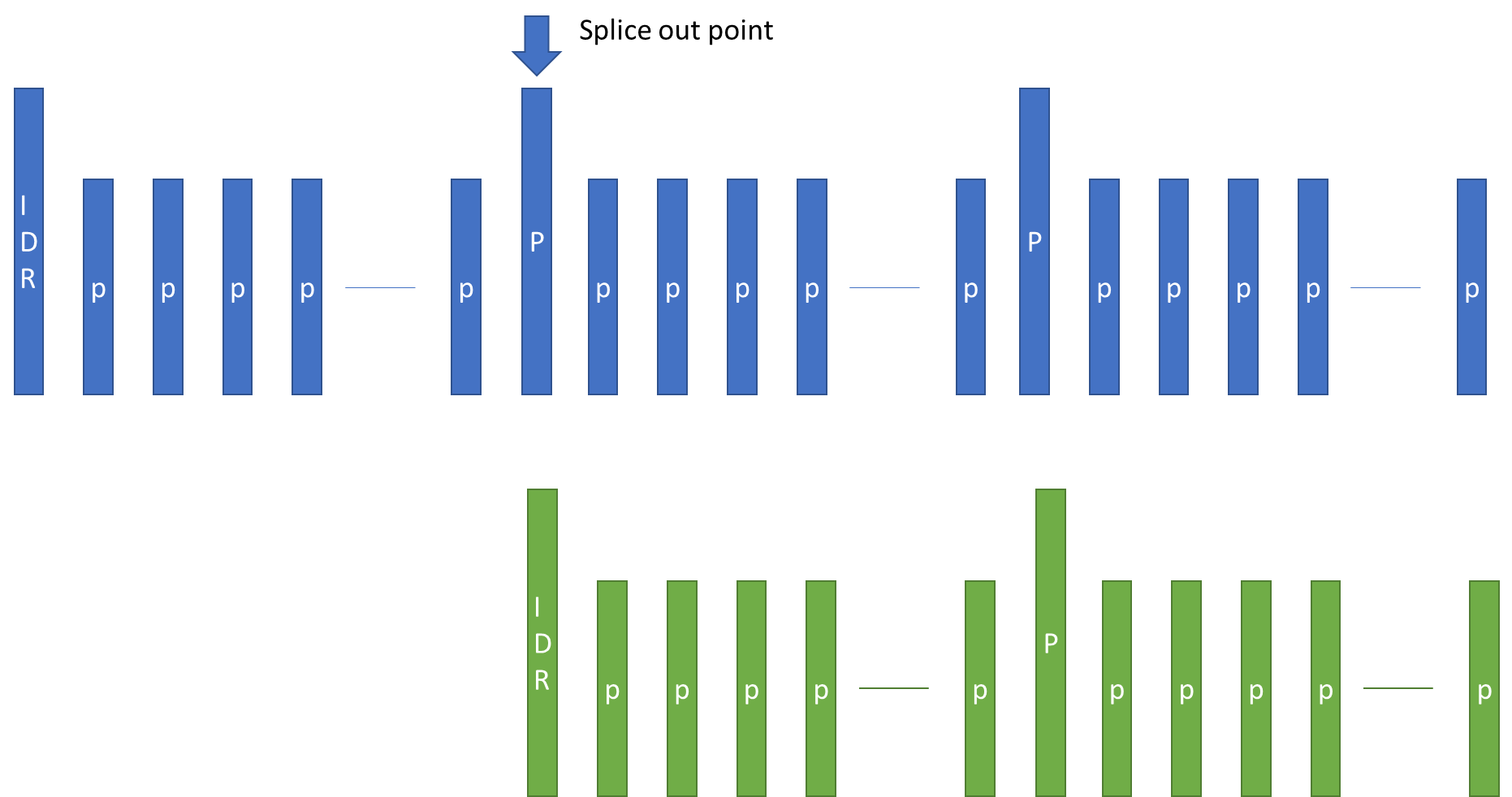


Figure 3.6-1: Splicing point at a P frame

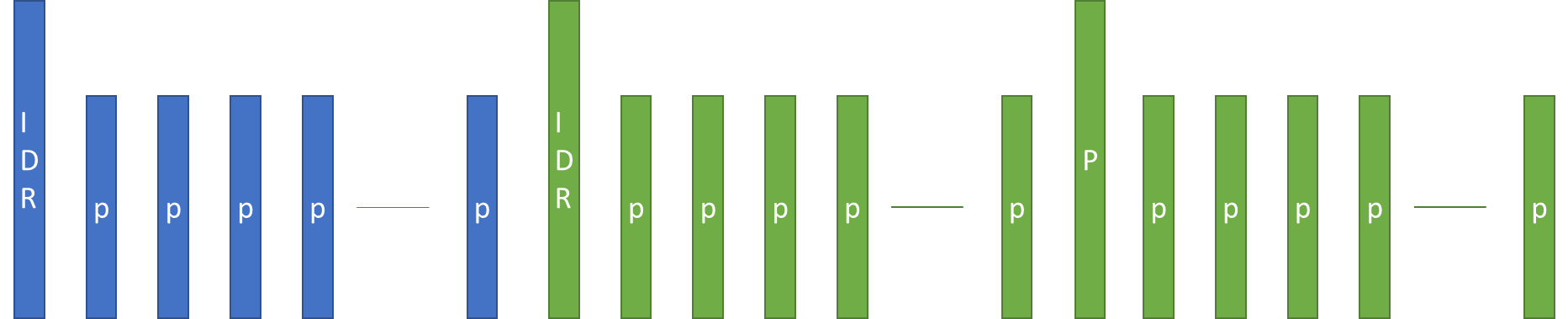


Figure 3.6-2: Resulting bitstream from a splicing

### General constraints

Similar as the above, but now the blue stream needs to be even more constrained to also.

### Specification (according to ISO/IEC 23090-15)

#### Functional Stage

#### Purpose

#### Type

#### Test and Observation

#### Additional Recommendations

### Recommendations/Requirements for Systems Standards

# Common Envelope

## Introduction

## Option 1: Signalling of decoder initialization information in file formats, DASH, etc

### Introduction

In adaptive streaming, the streaming client often requests to switch to a different bitstream with a different spatial resolution, for adapting to the changing network conditions, e.g., bandwidth. Different bitstreams are typically stored in different file format (FF) tracks and encapsulated in different DASH Representations within the same Adaptation Set, thus such bitstream switching typically involves switching of FF tracks and DASH Representations. It is assumed that the switch-to bitstream is of the same codec and the same profile as the switch-from bitstream. Depending on the availability of other information such as the highest spatial resolution and how the decoder is implemented, when the spatial resolution changes, particularly when to be higher, a decoder may be re-initialized. According to the MPEG input documents m54516 and m54522, decoder initialization or reinitialization typically takes 100-200 ms of time and the best case can be just 16 ms, depending on various factors including memory allocation time, secure/nonsecure, mapping time, memory fragmentation of double data rate (DDR), internal fragmentation of system memory management unit (SMMU), etc. Consequently, decoder re-initialization can disrupt a streaming session, affect playback continuity, etc.

In order to address such problems, the DASH-IF initiated the New Codecs Test Cases work item ([https://docs.google.com/document/d/1jV1vs2TsietU34EMPIPHsgILAWiKhEr7dk39hmCc18o/edit#heading=h.z5pebcnfqpbj](https://docs.google.com/document/d/1jV1vs2TsietU34EMPIPHsgILAWiKhEr7dk39hmCc18o/edit%23heading=h.z5pebcnfqpbj)), and later on in October 2020 the MPEG Systems WG started the Systems for Video AHG.

We believe that, to solve the decoder initialization problem, a key aspect is to avoid or reduce decoder re-initializations as much as possible in an application session. One way towards that goal is to signal information needed for decoder initialization for a scope as wide as possible, and at the same time provide guidelines in relevant specifications (e.g., in NOTEs of video coding standards, and in guidelines of systems and application specifications) to encourage or even require decoders implementations to use the signalled DII when initializing the decoder.

This contribution presents a proposal of DII signalling in the ISOBMFF and in DASH MPD. Similar information can be signalled for CMAF.

### DII signalling in the ISOBMFF

For DII signalling in the ISOBMFF, we propose to define a new track group, named video decoder initialization information track group, to signal the decoder initialization information, as in below.

#### 8.3.4.X Video decoder initialization information track group

##### 8.3.4.X.1 Definition

TrackGroupTypeBox with track\_group\_type equal to 'vdii' indicates that this track belongs to a group of video tracks that use the same profile of the same video codec and share the same video decoder initialization information.

NOTE 1 When the video decoder is initialized according to the video decoder initialization information, it is expected that no video decoder reinitialization is needed when bitstream switching occurs between any two of the bitstreams carried in this group of tracks.

NOTE 2 The video codec used for a chunk of a video track is indicated by the sample entry type.

##### 8.3.4.X.2 Syntax

aligned(8) class VdiiGroupBox extends TrackGroupTypeBox('vdii')  
{  
 unsigned int(8) dii\_profile\_max\_tier\_idc;  
 unsigned int(8) dii\_max\_level\_idc;  
 unsigned int(2) dii\_max\_chroma\_format\_idc;  
 unsigned int(4) dii\_max\_bitdepth\_minus8;  
 bit(2) reserved;  
 unsigned int(8) dii\_max\_num\_dec\_pics\_in\_dpb\_minus1;  
 unsigned int(16) dii\_max\_pic\_width;  
 unsigned int(16) dii\_max\_pic\_height;  
}

##### 8.3.4.X.1 Semantics

dii\_profile\_max\_tier\_idc indicates the profile and, when applicable (e.g., HEVC and VVC), the maximum tier to which the video bitstreams carried in the group of tracks conform. When the video codec is AVC, dii\_profile\_max\_tier\_idc shall be equal to the profile\_idc. When the video codec is HEVC or VVC, the first 7 bits of dii\_profile\_max\_tier\_idc shall be equal to general\_profile\_idc, and the last bit of dii\_profile\_max\_tier\_idc shall be equal to the greatest value of general\_tier\_flag for all the video bitstreams carried in the group of tracks.

dii\_max\_level\_idc indicates the maximum level to which the video bitstreams carried in the group of tracks conform.

dii\_max\_chroma\_format\_idc indicates the maximum value of sps\_chroma\_format\_idc (when the video codec is VVC) or chroma\_format\_idc (when the video codec is AVC or HEVC) of the SPS referenced by a picture when decoding the bitstreams carried in the group of tracks.

dii\_max\_bitdepth\_minus8 plus 8 indicates the maximum bit depth, in units of bits, of each colour component of a decoded picture when decoding the bitstreams carried in the group of tracks. The value of dii\_max\_bitdepth\_minus8 shall be in the range of 0 to 8, inclusive.

dci\_max\_num\_dec\_pics\_in\_dpb\_minus1 plus 1 indicates the maximum number of decoded pictures stored in the decoded picture buffer at any moment when decoding the bitstreams carried in the group of tracks.

dci\_max\_pic\_width indicates the maximum width, in units of luma samples, of a decoded picture when decoding the bitstreams carried in the group of tracks.

dci\_max\_pic\_height indicates the maximum height, in units of luma samples, of a decoded picture when decoding the bitstreams carried in the group of tracks.

### DII signalling in DASH MPD

For DII signalling in DASH MPD, we propose to optionally signal the information in the Adaptation Set, as in below:

**Table 5 — Semantics of AdaptationSet element**

| **Element or Attribute Name** | | | | **Use** | **Description** |
| --- | --- | --- | --- | --- | --- |
|  |  | **AdaptationSet** | |  | Adaptation Set description. |
|  |  |  | @xlink:href | O | specifies a reference to a remote element entity that shall contain exactly one element of type **AdaptationSet**. |
|  |  |  | @xlink:actuate | OD  default: 'onRequest' | specifies the processing instructions, which can be either "onLoad" or "onRequest". |
|  |  |  | @id | O | specifies a unique identifier for this Adaptation Set in the scope of the Period. The attribute shall be a unique unsigned integer value in the scope of the containing Period.  The attribute shall not be present in a remote element entity.  If not present, no identifier for the Adaptation Set is specified. |
|  |  |  | **...** | ... | ... |
|  |  |  | **SegmentTemplate** | 0...1 | specifies default Segment Template information.  Information in this element is overridden by information in the **Representation.SegmentTemplate**, if present.  For more details, see subclause 5.3.9. |
|  |  |  | **VideoDecoderInitInfo** | 0...1 | specifies the video decoder initialization information for all Representations in this Adaptation Set.  For more details, see subclause 5.3.X. |
|  |  |  | **Representation** | 0 … N | specifies a Representation.  At least one Representation element shall be present in each Adaptation Set. The actual element may however be part of a remote element entity if xlink is used on the containing **AdaptationSet** element.  For more details, refer to subclause 5.3.5. |
| **Key**  For attributes: M=mandatory, O=optional, OD=optional with default value, CM=conditionally mandatory, F=fixed  For elements: <minOccurs>...<maxOccurs> (N=unbounded)  The conditions only hold without using xlink:href. If linking is used, then all attributes are "optional" and <minOccurs=0>.  Elements are **bold**; attributes are non-bold and preceded with an @; list of elements and attributes is in ***italics bold*** referring to those taken from the Base type that has been extended by this type. | | | | | |

**XML syntax**

|  |
| --- |
| <!-- Adaptation Set --> |
| <xs:complexType name="AdaptationSetType"> |
| <xs:complexContent> |
| <xs:extension base="RepresentationBaseType"> |
| <xs:sequence> |
| <xs:element name="Accessibility" type="DescriptorType" minOccurs="0" maxOccurs="unbounded"/> |
| <xs:element name="Role" type="DescriptorType" minOccurs="0" maxOccurs="unbounded"/> |
| <xs:element name="Rating" type="DescriptorType" minOccurs="0" maxOccurs="unbounded"/> |
| <xs:element name="Viewpoint" type="DescriptorType" minOccurs="0" maxOccurs="unbounded"/> |
| <xs:element name="ContentComponent" type="ContentComponentType" minOccurs="0" maxOccurs="unbounded"/> |
| <xs:element name="BaseURL" type="BaseURLType" minOccurs="0" maxOccurs="unbounded"/> |
| <xs:element name="SegmentBase" type="SegmentBaseType" minOccurs="0"/> |
| <xs:element name="SegmentList" type="SegmentListType" minOccurs="0"/> |
| <xs:element name="SegmentTemplate" type="SegmentTemplateType" minOccurs="0"/> |
| <xs:element name=" VideoDecoderInitInfo" type=" VideoDecoderInitInfoType" minOccurs="0"/> |
| <xs:element name="Representation" type="RepresentationType" minOccurs="0" maxOccurs="unbounded"/> |
| **...** |

### 5.13.X Video Decoder Initialization Information

#### 5.13.X.1 Definition

The element **VideoDecoderInitInfo** may be present in an Adaptation Set to provides the video decoder initialization information for video bitstreams carried in all the Representations in the Adaptation Set. The presence of this element indicates that all Representations of the Adaptation Set use the same profile of the same video codec and share the same decoder initialization information.

When the video decoder is initialized according to the video decoder initialization information, when present, it is expected that no video decoder reinitialization is needed when bitstream switching occurs between any two of the bitstreams carried in all the Representations in this Adaptation Set.

NOTE The video codec and profile bitstreams carried in the Representations in the Adaptation Set are signalled by the @codecs attribute.

#### 5.13.X.2 Semantics

**Table Y — Semantics of VideoDecoderInitInfo element**

| **Element or Attribute Name** | | | | **Use** | **Description** |
| --- | --- | --- | --- | --- | --- |
|  |  | **VideoDecoderInitInfo** | |  | Decoder Initialization Information description |
|  |  |  | @maxTier | O | specifies the maximum tier for all the video bitstreams carried in the Representations in the Adaptation Set. When the codec is AVC or another video codec that does not have tier specified, this attribute shall not be present. When the codec is HEVC or VVC or another video codec that has tier specified, this attribute shall be present. |
|  |  |  | @maxLevel | M | specifies the maximum level for all the video bitstreams carried in the Representations in the Adaptation Set. |
|  |  |  | @maxChromaFormat | M | specifies the maximum value of sps\_chroma\_format\_idc (when the video codec is VVC) or chroma\_format\_idc (when the video codec is AVC or HEVC) of the SPS referenced by a picture when decoding the video bitstreams carried in the Representations in the Adaptation Set. |
|  |  |  | @maxBitDepthMinus8 | M | specifies the maximum bit depth, in units of bits, of each colour component of a decoded picture when decoding the bitstreams carried in the in the Representations in the Adaptation Set. The value shall be in the range of 0 to 8, inclusive. |
|  |  |  | @maxNumDecPicsMinus1 | M | the value plus 1 specifies the maximum number of decoded pictures stored in the decoded picture buffer at any moment when decoding the bitstreams carried in the Representations in the Adaptation Set. |
|  |  |  | @maxDecPicWidth | M | specifies the maximum width, in units of luma samples, of a decoded picture when decoding the bitstreams carried in the Representations in the Adaptation Set. |
|  |  |  | @maxDecPicHeight | M | specifies the maximum height, in units of luma samples, of a decoded picture when decoding the bitstreams carried in the Representations in the Adaptation Set. |
| **Legend:**  For attributes: M=mandatory, O=optional | | | | | |

#### XML syntax

|  |
| --- |
| <!-- VideoDecoderInitInfo --> |
| <xs:complexType name="VideoDecoderInitInfoType"> |
| <xs:attribute name=" maxTier" type="xs:unsignedInt"/> |
| <xs:attribute name=" maxLevel" type="xs:unsignedInt" use="required"/> |
| <xs:attribute name=" maxChromaFormat" type="xs:unsignedInt" use="required"/> |
| <xs:attribute name=" maxBitDepthMinus8" type="xs:unsignedInt" use="required"/> |
| <xs:attribute name=" maxNumDecPicsMinus1" type="xs:unsignedInt" use="required"/> |
| <xs:attribute name=" maxDecPicWidth" type="xs:unsignedInt" use="required"/> |
| <xs:attribute name=" maxDecPicHeight" type="xs:unsignedInt" use="required"/> |
| </xs:complexType> |

## Option 2: Signalling of decoder initialization information in file formats, DASH, etc

### Abstract

The current specification already includes some text regarding avoidance of decoder re-initialization when sample entries change within a track for particular cases. In such cases, all samples within the track are marked as belonging to the same decoding capability information sample group. However, with the ongoing discussion in the AHG on “Systems Functionalities for Video Conformance”, it is clear that further information is required if decoder re-initialization needs to be avoided in streaming scenarios using DASH or when a player switches from one track to another. A solution is proposed in this contribution.

### Introduction

Decoder re-initialization due to track changes or switching in DASH environment can lead to several problems. On the one hand, with single-layer open GOP resolution change being supported by VVC, there might be instances of open GOP resolution switching through track switching or representation switching in a DASH environment. Re-initializing the decoder as a result thereof would start a new decoding process splitting a CVS into two dropping potential RASL AUs as discussed below. This would lead to a non-continuous playout experience with gaps of dropped frames. Also, even when open GOP structures are not used, it has been discussed in m54516 and m54522 that decoder initialization or re-initialization can take around 16-200 ms, while in typical cases it is above 100 ms. Thus, it may also lead to a non-continuous playout experience even when no AUs are dopped.

In particular, when playback of more than one track or a DASH environment is considered, where picture formats change, there might be problems when a decoder re-initialization happens or when a CVS is split into two due to the presence of “incorrect” parameter sets as explained below.

One example for cases where a file with more than one alternative tracks may be present, and switching of one track to another might be desired is RoI applications. The content covering different spatial regions is offered into two or more different variants. Thereby, it is possible to zoom in or out based on the user input. In the example shown in the figure, one track corresponds to the whole video and another track to a subset thereof (RoI).



Figure : open GOP structure for two bitstreams for ROI.

In particular with open GOP structures as the ones depicted in the figure, a player could switch from one track to another at every CRA and zoom in or out.

In order for such an approach to work, two things are required:

1. The SPSs of the largest video (i.e. corresponding to the whole video, not the RoI) need to be used.
2. The parser needs to be aware that these alternative tracks do not require re-initialization i.e. that they are switchable and that CRAs result of a track switch do not start a new CVS.

Thus, a parser could extract the following bitstream and decode it if the content has been encoded properly.



Figure : Bitstream result of switching tracks at CRA for ROI.

Note that the aspect related to using proper parameter sets also applies to “regular” adaptive streaming services that do resolution switch with open GOP structures. In particular when there are in-band parameter sets. The following figure shows an example of the issue resulting from switching at CRAs when there are in-band parameter sets and there is a resolution switch.



Figure : non-conforming bitstream after switching with in-band SPS storage.

The correct result that would lead to a conforming bitstream for the depicted switch is shown in the following.



Figure : Conforming bitstream after switching with in-band SPS storage.

One could think that in an alternative solution, the SPSs stored in the 1080p bitstream should be the one for 4K but this would lead to the problem that devices only capable of decoding a lower level could not decode the 1080p bitstream. Therefore, a solution is required that depending on the maximum operation point that a decoder is targeting to decode proper parameter sets are included, e.g. corresponding to 1080p or 4K.

The examples above relate to open GOP switching of bitstreams either in different tracks within a file or in a same track as a result of an adaptive streaming system like DASH. The discussion relates to signalling tracks that are switchable and selection of correct parameter set to avoid start of a new CVS at CRA AUs.

As aforementioned, even for use-cases not considering open GOP switching points, there are further concerns for re-initialization of a decoder in general in DASH scenarios due to the large time required for decoder re-initialization.

The current decoding capability information sample group allows marking samples within a track as belonging to the sample group which contains a DCI NAL unit. The intent of the DCI NAL unit in VVC is to indicate conformance of the whole bitstream even if it contains multiple CVSs, describing the profile-tier-level(s) that a bitstream conforms to. Thereby, a decoder would know before parsing all SPSs in the bitstream that it can decode the whole bitstream. This has been used in the ISOBMFF integration of VVC to be allow indicating that when the sample entry changes within a track the decoder should not be re-initialized through a dedicated sample group ‘dcfi’.

Considering adaptive streaming, if the player took the decoding capability information sample group of the initialization segment of the representation that requires highest decoding capabilities, re-initialization of the decoder when switching could be avoided.

Unfortunately, the DCI NAL unit does not contain information regarding the DPB management, such as number of picture slots required for decoding the bitstream, biggest picture sizes, chroma formats, etc. It is asserted that, depending on the implementation, changes of these values within a bitstream could require decoder re-initialization, e.g. when the decoder was initialized with a DPB for small picture sizes at the beginning of the bitstream and picture sizes increase at the start of a following CVS of the bitstream.

The proposal in the following section provides solutions for:

* Signaling switchable tracks that do not require decoder re-initialization
* Reconstructing the bitstream with proper parameter sets according to target operating point and thus e.g. avoid starting a CVS at CRA AUs preventing thus output of RASL pictures when switching happens.
* Providing DPB information that applies to the whole bitstream to avoid decoder re-initialization when sample entries change or when switching happens (e.g. track or segments switching in a DASH scenario).

### Proposal

#### Switchable tracks

A new entity group is proposed that indicates that tracks are switchable at samples with SAP type up to 3 without requiring decoder re-initialization and without starting a new CVS. The entity group describes a track hierarchy, which makes sure that when the sample description box of a track with hierarchy value a is used for initializing the decoder, switches at the described SAP positions are possible to and from any track with hierarchy value b <= a.

**11.9.4 VVC switchable tracks entity group**

**11.9.4.1 General**

The VVC switchable tracks entity group defines a set of tracks among which it is possible to switch using the sample description box of a single track pertaining to the entity group and without re-initializing the decoder. When the decoder is initialized with the sample description box of a track i that belong to the entity group with a particular value of track\_switch\_hierarchy\_id[i], any track j of the entity group with value track\_switch\_hierarchy\_id[j] smaller than or equal to track\_switch\_hierarchy\_id[i] can be used with the same sample description box. Switching from one track to another only leads to a valid VVC bitstream if the switch is performed at a SAP sample with a SAP type equal to or smaller than 3.

When an EntityToGroupBox with grouping\_type equal to 'swtk' is present, the EntityToGroupBox shall be contained in the GroupsListBox in the file-level MetaBox and shall not be contained in MetaBoxes of other levels.

**11.9.4.2 Syntax**

aligned(8) class SwitchableTracks extends EntityToGroupBox('swtk',0,0)  
{  
 for (i = 0; i < num\_entities\_in\_group; i++)  
 unsigned int(16) track\_switch\_hierarchy\_id[i];  
}

**11.9.4.3 Semantics**

track\_switch\_hierarchy\_id[i] specifies the hierarchy of a track within the tracks belonging to the entity group.

#### Parameter set sample groups

A new sample group is proposed that contains a sequence parameter set NAL unit instead of carrying the in-band parameter sets within the samples, so that depending which maximum operation point is targeted, the proper sequence parameter sets are placed into the reconstructed bitstream from the sample group information.

### **11.8.X Parameter set sample group**

### **11.8.X.1 Definition**

A sample group description entry of this sample group contains an SPS NAL unit.

When a sample is mapped to a parameter set sample group ('pase'), it indicates that the SPS NAL unit contained within the sample group needs to be inserted into the reconstructed AU if the target maximum picture format corresponds to the picture format indicated within the 'pase' sample group.

### **11.8.X.2 Syntax**

class ParameterSetSamplegroup () extends VisualSampleGroupEntry (’pase’)  
{  
 unsigned int(16) max\_width;  
 unsigned int(16) max\_height;  
 unsigned int(16) paramSetNalUnitLength;  
 bit(8\* paramSetNalUnitLength) paramSetNalUnit;  
}

### **11.8.X.3 Semantics**

max\_width and max\_height specify the maximum picture format of the target reconstructed bitstream for which a SPS NAL unit is to be inserted.

paramSetNalUnitLength indicates the length in bytes of the SPS NAL unit.

paramSetNalUnit contains an SPS NAL unit as specified in ISO/IEC 23090-3.

* 1. **Adding DPB information into the decoding capability information sample group**

It is proposed to add DPB information that corresponds to the maximum decoding capability required to decode a bitstream.

* + 1. **Decoder capability information sample group**

**11.8.10.1 Definition**

A sample group description entry of this sample group contains a DCI NAL unit.

When samples of two sample entries are mapped to the same decoder capability information sample group description entry, a player can switch sample entries without re-initialization of the decoder.

If any DCI NAL unit is present in any sample entry or in-band, it shall be exactly the same as the DCI NAL unit included in the corresponding decoder capability information sample group.

**11.8.10.2 Syntax**

class DecoderConfigurationInformation extends VisualSampleGroupEntry ('dcfi') {  
 unsigned int(16) max\_width;  
 unsigned int(16) max\_height;  
 unsigned int(3) max\_bit\_depth\_minus8;  
 unsigned int(2) max\_chroma\_format;  
 unsigned int(3) reserved = 0;  
 unsigned int(8) max\_dec\_pic\_buffering\_minus1;  
 unsigned int(16) dciNalUnitLength;  
 bit(8\*nalUnitLength) dciNalUnit;  
}

* + - 1. **Semantics**

max\_width indicates the maximum picture width of pictures of a sample belonging to the sample group.

max\_heigth indicates the maximum picture height of pictures of a sample belonging to the sample group.

max\_bit\_depth\_minus8 plus 8 indicates the maximum bit depth of pictures of a sample belonging to the sample group.

max\_chroma\_format indicates the maximum chroma format of pictures of a sample belonging to the sample group.

max\_dec\_pic\_buffering\_minus1 plus 1 indicates the maximum value of dpb\_max\_dec\_pic\_buffering\_minus1, as specified in ISO/IEC 23090-3, in the VPS and/or SPS NAL units referred to by any picture of a sample belonging to the sample group

dciNalUnitLength indicates the length in bytes of the DCI NAL unit.

dciNalUnit contains a DCI NAL unit as specified in ISO/IEC 23090-3.

# Stream Switching Background

## Summary

We review several efforts/ideas in the history of streaming media technologies that have been tried before eventually settling on using synchronized IDR pictures and IDR-level switching as mass-deployed and used today in protocols such as MPEG DASH or HLS. In all cases, cons and pros of each approach will be discussed and reasons why it was not mass adopted will be explained.

## Problems that motivated the development of stream switching

An essential problem that all early streaming systems faced, and which still is largely present with Internet today is – **time-varying behavior of networks** used for delivery of streaming media content [1-3]. In Figure 1, we show example of such network bandwidth fluctuations. This plot was captured on a modern PC connected over WiFi network and while downloading a large video file.

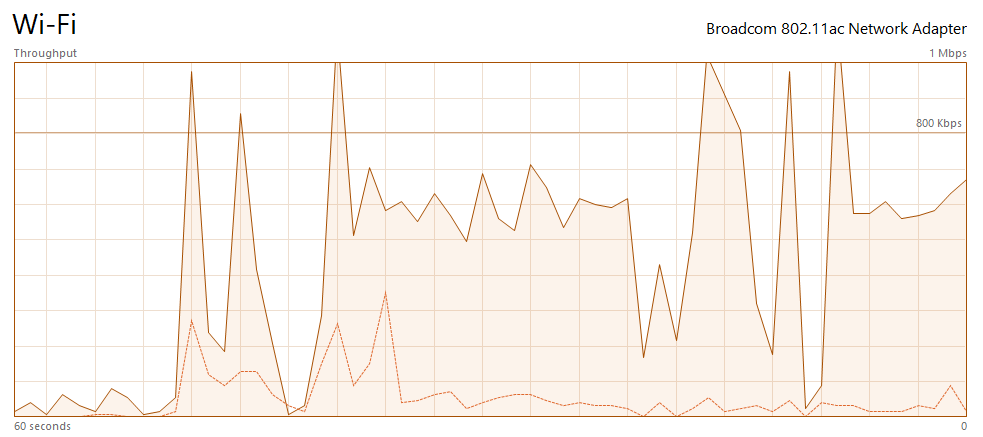


Figure 1. Example of network bandwidth fluctuations observed over time.

There are several underlying reasons for such bandwidth fluctuations. They include:

* packet losses and the effects of retransmissions at various links of the network, as well as end-to-end retransmissions and bandwidth control mechanisms of the TCP
* varying amounts of data concurrently sent in different segments of the internet and the resulting congestions at local or global levels
* the effects of routers pushing packets through different routes, resulting in different delays
* the effects of schedulers in wireless networks regulating transmissions to different receivers
* the effects of CDNs and other content caches, etc.

The presence of such bandwidth variations have motivated the development of techniques enabling continuous playback of media content despite bandwidth changes.

The second problem that early streaming systems have tried to address was a **diversity of internet access methods** and thus different maximum transmission speeds that could be used. Specific categories of such access methods back in 1990s included 28.8k modems, 56k modems, single ISDN, dual ISDN, and broadband type connections [1-3].

To serve each category of users efficiently, multiple streams become necessary. But presentation of such streams as multiple links embedded in web pages (as in example, shown in Figure 2) was clearly undesirable. It was consuming web page’s real estate, confusing the user, and required extra work.



Figure 2. Example of 1990s’ web page with links to multiple video streams provided.

This problem motivated the development of mechanisms for automatic detection of bandwidth and then selection of the right streams.

## Multi-rate streaming with stream switching

The eventual solution to both problems mentioned earlier – was **multi-rate streaming system with automatic stream switching.**

To the best of author’s knowledge, the first commercially successful example of such streaming system was RealSystem G2, developed by RealNetworks in 1998. The adaptive multi-rate functionality in this system was called “SureStream” technology [2,3].

In Figure 3 we show example of its operation.

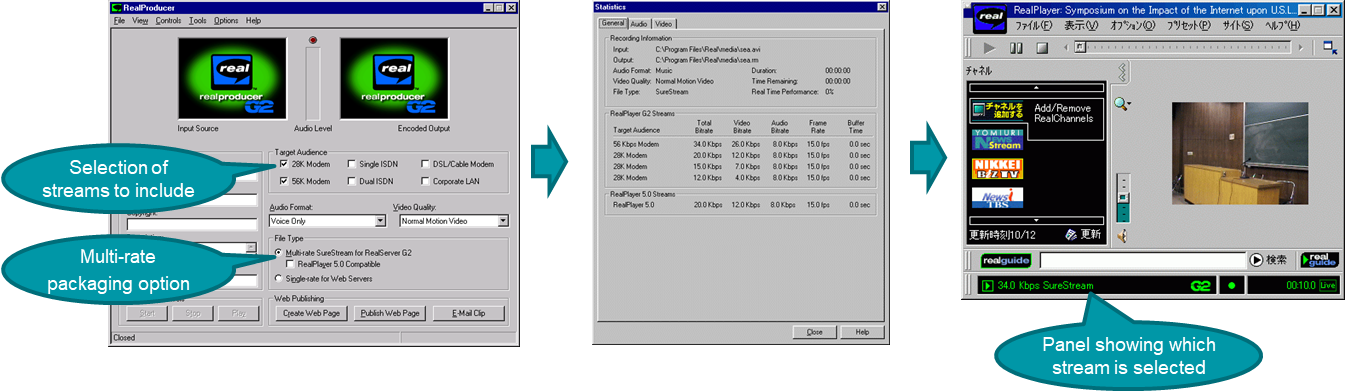


Figure 3. Screenshots of RealSystem G2 encoder, encoded streams, and a player.

Essentially, the input media was encoded at several different bitrates, specified in “audience” settings in the encoder, then all such streams were packaged in a single “surestream” container, and then sub-selection of streams for transmission was done by the server, driven by the rate detection and switching logic in the player [2,3].

By performing such adaptation, the stalls that would otherwise occur with bandwidth drops were prevented. The example logic of switching with varying bandwidth is illustrated in Figure 4.

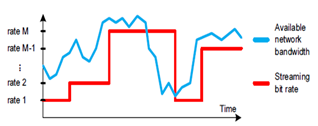


Figure 4. Example of time-varying network behavior and switching logic.

As we can see, this early streaming system was very similar to modern HTTP-based ABR systems. The main differences were in the methods used for transmission of the encoded streams. Instead of HTTP and TCP-based transmission of full encoded segments, such early streaming systems used UDP-based transports (RDT – Real Data Transport or RTP) and transmission was done on packet-level. Such systems were also designed to tolerate packet losses, with codecs implementing data concealment methods. I or IRD frames were used not only as means of achieving random access points but also as means for preventing of artifact propagations in cases of packet losses [1-3].

References [4-8] are examples of early patents filed on various aspects of such systems.

But in principle, we see that multi-rate streaming and switching mechanisms that powers today’s streaming standards such as MPEG DASH [9] and HLS [10] were indeed well known, implemented, and practically deployed much earlier – at least since 1998.

## Alternative and Complementary Techniques

In this section we will now described several additional or alternative technologies to that have been tried before settling on stream switching mechanism.

### Pre-roll buffer

The first measure that most early streaming systems (such as RealAudio, NetShow, ViVo, etc [1-3]) took to try to minimize the effects of varying network bandwidth was the introduction of the pre-roll buffer. Essentially, as shown in Figure 5, these systems were delaying beginning of the edia playback by few seconds, and using this time to accumulate bits and seconds of playback time, which could be used later to continue to play in the event when bandwidth drops.

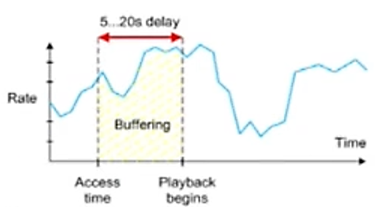
****

Figure 5. Example of time-varying network behavior and switching logic.

By itself, such buffering technique was mostly adequate to achieve satisfactory playback of audio [2]. But with the addition of video, the need in more sophisticated network adaptation mechanism was apparent.

Still, even after multi-rate adaptive streaming mechanism was introduced, the pre-roll delay remained part of the design. Such delay was still needed to continue playback while staying at same rate, and also to have a balance of playback seconds needed to safely implement the switches.

Pre-roll remains to be essential complementary feature in modern HTTP-based systems as well.

### Real-time encoding/transcoding with bitrate adaptation

The need to operate with varying bandwidth has existed not only in streaming, but also in early video conferencing systems such CLI, PictureTel, as well as ITU-T Rec. H.230 [11] and ITU-T Rec. H.324 [12] – standards-based systems.

The solution in VC case was fairly straightforward – direct encoders at each terminal device to use bitrates that are matching the estimated bandwidth of connection to the receiver. Systems implementing such logic have existed in late 1980s [11].

Somewhat similar encoding rate adaption was also done in GSM wireless communication systems [13]. Such systems have adopted multi-rate speech codecs (from full-rate and half-rate GSM codecs to AMR and AMR-WB) and so-called “codec mode adaptation” algorithms to chose codec mode best matching available system throughput [13].

However, in both VC and wireless communication the delivery of each rate-adapted encoded stream was targeting single receiver. In streaming, there are many receivers! Same stream can be watched by thousands or millions of viewers! And if adaptive transcoding would need to be done for each receiver customarily – this approach quickly becomes non-practical. Video transcoding is generally a much more expensive operation compare to data caching/relay functions performed by CDNs.

But remarkably-enough, real-time transcoding for media delivery was actually tried, and on a mass scale, before it was abandoned. The companies that tried it were ByteMobile, Vantrix, Ortiva wireless, Mobixell, and several other so-called “mobile video optimization” companies operating around 2008-2014 time-frame [14]. What effectively they have been doing – is real-time transcoding of progressive download files published by YouTube and other popular websites on the internet. Such in-place transcoders were deployed at gateways to mobile core networks, and were used to reduce/manage network loads. Such solutions were also pitched as having benefits to end users – minimizing stalls during playback and wireless data bills. However, with transition of YouTube, Netflix and other sites to use of adaptive streaming protocols such as HLS or DASH – and the use of secure transmission modes and DRMs – such in-network transcoders have become much less effective, and all above-mentioned companies have either pivoted to different businesses or ceased to exist [14].

### Stream “thinning”

Stream “thinning” was a technique that early RealVideo servers (prior to release of G2 in 1998) have been using to try to dynamically reduce bitrates of single-rate-encoded video streams [2].

Example of thinning operations include progressive removal of B frames or P-frames preceding I/IDR-frames, after all B frames are removed. At the encoded stream level RealVideo was a variable frame-rate codec, so that removal of frames with no coding dependencies could have been accomplished without the need to repacketize of change any data in encoded streams [2]. At decoder end, RealVideo used frame-rate upsampler that was trying to synthesize frames that could have been dropped [2].

Naturally, with thinning, there were limits to variation of bitrates that could be supported. The drop of a significant number of frames was also noticeable by viewers. In other words, this method was clearly inferior to adaptive stream switching method. Its use was discontinued after the release of RealSystem G2.

### The use of scalable coding

Early ideas of scalable coding, also known as “hierarchical coding”, or “successive refinement of information” can be traced to 1980s and pioneering work of information theorists: V. N. Koshelev, T. Cover, B. Rimoldi, and others [15-17]. Most video compression standards, starting from H.263 [18] and MPEG-2 [19] to HEVC [21] incorporate them in form of so-called spatial, temporal and SNR scalability modes. For streaming applications, spatial and SNR scalability modes are of prime interest. Spatial scalability enables a stream to be composed of 2 sub-streams: a base encoded stream at resolution and rate , and an extended stream at resolution and bitrate .

Compared to 2 independently encoded streams:

Stream 1: resolution and rate and

Stream 2: resolution and rate ,

the storage of scalable streams should take less space:

as extended stream usually takes much fewer bits to encode.

However, in terms of quality, in almost all cases, direct encoding at final resolution and composite bitrate becomes considerably better than quality of 2-layer scalable encoding.

Conversely, if we equalize quality of direct encoding at resolution and 2-layer scalable encoding, then the number of bits required for direct encoding will be:

where is certain gain factor (typically around 5-15%) by which direct encoding is more efficient than scalable encoding.

Therefore, by considering that the average bandwidth that may be used by streaming system is

where p is the probability of load of the first rendition, we observe that a system with direct encoding will be by

more efficient (in terms of use of CDN bandwidth) than a system with scalable encoding.

Of course, one can also argue that since storage-wise scalable encoding should take less CDN edge cache space, and therefore have lover cache miss probability. But this argument falls apart when distribution of usage of renditions is highly asymmetric, as it is typically the case in many modern systems.

In practice, to the best of our knowledge, scalable coding has never been successfully deployed for mass-scale streaming applications. It does have certain appeal however, for other multimedia applications, such as broadcast or video conferencing [18].

### Reference picture resampling

This is a feature allowing a stream to continue decoding in cases of transitions from frames of different spatial resolutions. It’s been a feature in many standard and proprietary video codecs.

For example, in H.263+ it was introduced as Annex P [23]. Same feature is also present in a newly designed VVC standard [24].

While initially this feature was proposed for use in video conferencing applications [23], it was actually quite useful in early streaming applications for implementing features such as “dirty” switches as well as S frames.

### “Dirty” switches

Video players in RealSystem G2 used one particular optimization: they were switching up to a higher bitrate streams only at locations of I/IDR-frames in that bitrate stream. However, they also were able to a stream with lower bitrate at arbitrary I/IDR or P frames. If such a stream had different resolution, the decoder was applying reference picture resizing process as part of the switch.

Such more-fine grain switches have shown some promise in improving system’s response to bandwidth changes. However, in the later versions of RealSystem and introductions of more complex video codecs (RealVideo 8-10 [2]) this feature has been discontinued, and the switches down have been allowed only on IDR frames.

One problem with dirty switches was the need to make sure that all implementations of decoders have sufficiently advanced error resilience and error concealment tools. With the introduction of hardware/firmware implementations of RealVideo 8-10 in mobile devices, this was almost impossible to achieve.

### S frames

S frames have been introduced around 1996-97 in papers by B. Girod, N. Farber, and U. Horn [25-26]. More detailed description can be also found in patents [27-28].

There were several variants of S-frames proposed. This, in Figure 6, we show Intraframe- type switches from base layer. On the left, we show prediction structure, and on the right we show the structure of the decoder needed to support it. The SRC block in this decoding process is a reference picture resampler.

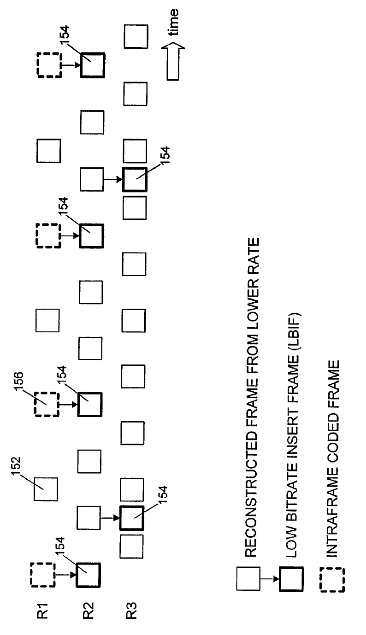
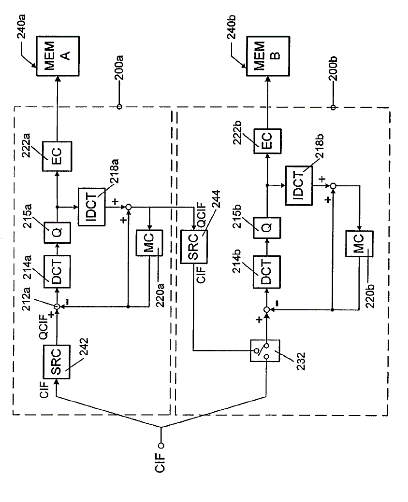
(a) (b)

Figure 6. Intra-frame-based switching structure (a), and decoder structure used to support this (b).

In Figure 7, we show prediction structures and switches with intra-coded S frames, as well as decoder structure needed to support such operation. Here again the SRC module denotes the reference picture resampling operation. There rest shows that S frame is effectively a P-frame connecting two frames in different layers.

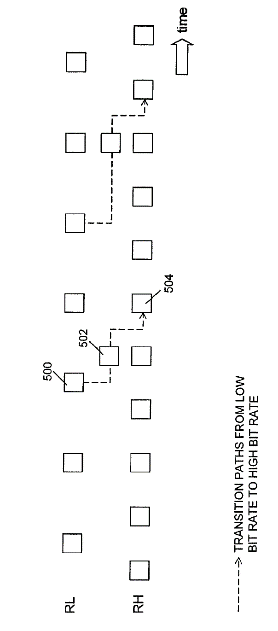
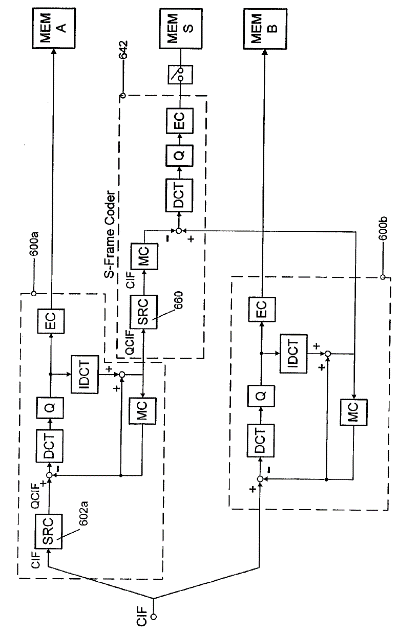
(a)(b) 

Figure 7. Inter-frame-based switching structure (a), and decoder structure used to support this (b).

What is important to note – is that in practice S-frames do not achieve perfect synchronization between layers. While they code the mismatch between 2 reference states, such coding is lossy, and even if same exact quantizer used for both receiving stream and the S frame, some mismatch will still remain in to be in the system.

### By-directional S frames (1998)

References [27-28] also describe a variant of design of S frames enabling symmetric by-directional switching. We explain their operation in Figure 8.

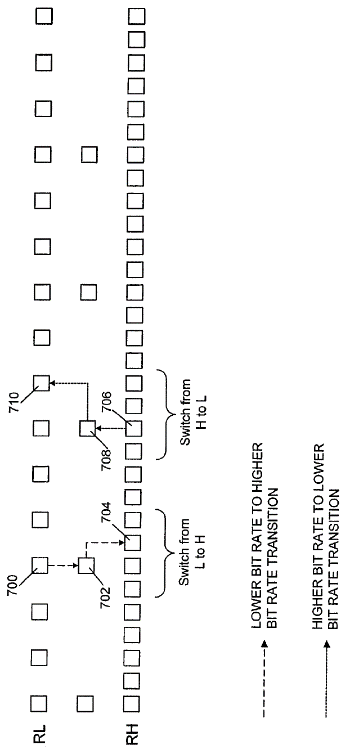
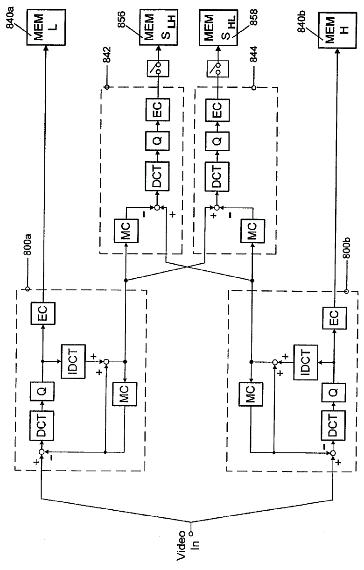
(a) (b)

Figure 8. By-directional switching structure (a), and decoder structure used to support this (b).

As can be easily seen, the S frames in this case have same structures as P-frames, but with the difference that residual image may be either added or subtracted from the reconstruction.

This way, same switch frame could be used to support transitions in both directions.

In passing we must note that S-frames (including their by-directional variants) have been fully pro prototyped and tested in the design phase of RealSystem G2 [2]. However, in the end, the decision was not to include them in the final release because of the complexity of supporting them on a system level.

### SI and SP frames

SI and SP pictures are variants of S frames that have been proposed and adopted in the H.264 extended profile [29-30].

In Figure 9 we show prediction structures corresponding to SI and SP frames, and in Figure 10, we show the respective decoding processes for each.

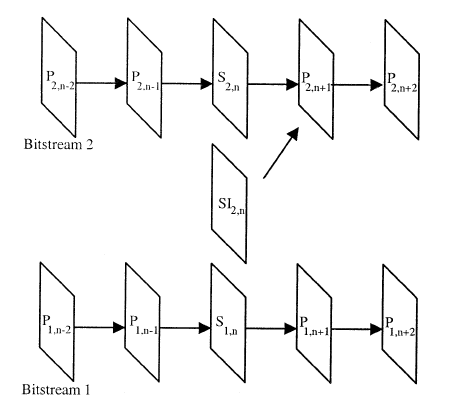
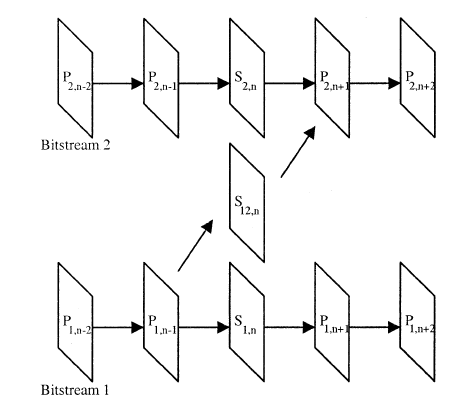
(a) (b)

Figure 9. Temporal placement of (a) SI and (b) SP frames.

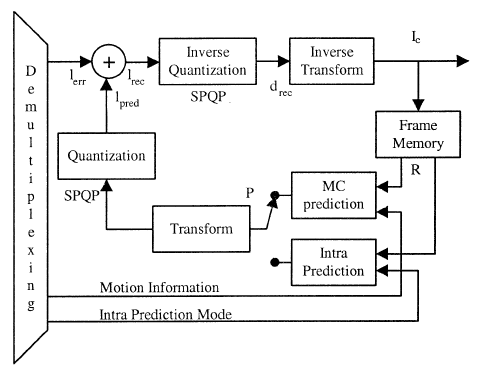
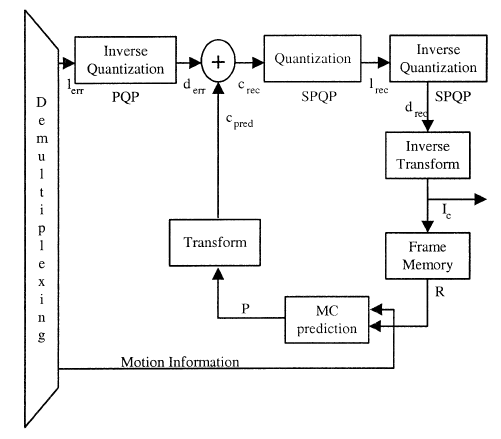
(a)(b) 

Figure 10. Decoding process for (a)SI -frame and (b) SP-frame.

Observations:

* Compared to S frames SI/SP frames add extra steps of transformation and re-quantization of the reconstructed reference frames in the decoder loop
* Such re-quantization is done using SPQP levels which are assumed to be same for SI/SP frame and QP level of stream that codec must join
* Such re-quantization, however, does not guarantee that the resulting reference frames in both transitional frame and in-stream continuation frame will be exactly the same. It does minimize possible difference between them, but it does not make them mathematically identical.

As such the use of SP frames have the potential of bringing of all same effects as traditional S frames. The authors are not aware of any deployed systems using SI/SP frames.

### HESP

HESP protocol [31] is a recent development, promoted by THEOplayer and SynaMedia [31]. In a nutshell, HESP looks like an attempt to create a variant of functionality which was known earlier as S frames and dirty switching. Same comments as we provided above for each apply in this case.

## Conclusions

We have reviewed several well-known efforts around methods of implementing stream switching other than switching on synchronously placed IDR frames.

We noted that none of these methods have achieve wide-spread deployments or were successful in practice. The main reasons in most cases were very minor gains in terms of reduced bandwidth relative to system complexity that such methods were introduced.

We further note that some methods, such as S-frames, dirty-switching, etc., have been introducing mismatch in the decoder state after switching, and that had significant problems in practice. Thus, only decoders that have been specifically designed to have error concealment modes, as well as handling issues such as HDR buffer underflows and overflows were suitable for receiving such streams. This dramatically reduced the reach of streaming solutions relying on such technology.

And considering all of the above, we recommend not to repeat past mistakes and don’t introduce new solutions/features that would cause decoders to operate in non-defined non-conformant regimes at ES levels. In our view, any solutions that we would recommend must be supplied with exact conformance process and means to ensure that encoder is operating in modes that are defined by standards and in which decodability and quality of the resulting streams can be guaranteed.

There are some lessons that history teaches us, and we hope we can learn from them.

## References

1. D. Wu, Y. T. Hou, W. Zhu, Y.-Q. Zhang, and J. M. Peha, “Streaming Video Over the Internet: Approaches and Directions,” IEEE Trans. Circuits Syst. Video Technol., 11(3):282–300, Mar. 2001.
2. G. Conklin, G. Greenbaum, K. Lillevold, A. Lippman, and Y. Reznik, "Video Coding for Streaming Media Delivery on the Internet," IEEE Trans. CSVT, 2001, 11 (3), pp. 20-34.
3. A. Lippman, "Video coding for multiple target audiences," Proc. SPIE 3653, VCIP'99, 28 Dec. 1998.
4. R. Agarwal, et al, “System and Method for Providing Random Access to a Multimedia Object Over a Network”, US Patent 6314466, priority: Oct 6, 1998, granted: Nov 6, 2001.
5. G. Greenbaum et al., “System and Method for Generating Multiple Synchronized Encoded Representations of Media Data”, US Patent 7,885,340, priority Apr 27, 1999, granted Feb 8, 2011.
6. D. Lanphear, “Stochastic adaptive streaming of content”, US Patent 6,968,387, filed Jan 10, 2003, granted Nov 22, 2005.
7. J. Lango, et al, “Pre-computing streaming media payload method and apparatus”, US patent 6,742,082, filed Oct 16, 2001, granted May 25, 2004.
8. J. Lango, et al, “Streaming media bitrate switching methods and apparatus”, US Patent 7,054,911, priority Jun 12, 2001, granted May 30, 2006.
9. ISO/IEC 23009-1, DASH Part 1, 3rd Edition DIS, 2019.
10. R. Pantos, W. May, “HTTP live streaming, RFC 8216,” IETF, <https://tools.ietf.org/html/rfc8216>
11. ITU-T Rec. H.320: Narrow-band visual telephone systems and terminal equipment, 1990.
12. ITU-T Rec. H.324: Terminal for low bit-rate multimedia communication, Mar 1996.
13. T. Halonen, J. Romero, and J. Melero, “GSM, GPRS and EDGE Performance: Evolution towards 3G/UMTS”, John Wiley & Sons Ltd., West Sussex, UK, 2003.
14. P. Lopez, “Mobile video monetization 2015”, CoreAnalysis, <https://www.coreanalysis.ca/>
15. V. N. Koshelev, "Hierarchical coding of discrete sources", Problemy peredachi informatsii vol. 16, no. 3, 1980, pp. 31–49.
16. W. H. R. Equitz and T. M. Cover, "Successive refinement of information," in IEEE Transactions on Information Theory, vol. 37, no. 2, pp. 269-275, March 1991.
17. B. Rimoldi, "Successive refinement of information: characterization of the achievable rates," in IEEE Transactions on Information Theory, vol. 40, no. 1, pp. 253-259, Jan. 1994.
18. ITU-T Rec. H.263, Annex O: Temporal, SNR, and Spatial Scalability mode, Feb 1998.
19. ISO/IEC 13818-2 | ITU-T Rec. H.262, MPEG-2 Video, Jul. 1995.
20. ISO/IEC 14496-10 | ITU-T Rec. H.264, Advanced Video Coding, Dec. 2003.
21. ISO/IEC 23008-2 | ITU-T Rec. H.265, High efficiency video coding, Dec. 2013.
22. B. Girod, “Scalable video for multimedia systems”, Computers & Graphics, vol. 17, no 3, 1993, pp 269—276.
23. ITU-T Rec. H.263, Annex P: Reference Picture Resampling, Feb 1998.
24. ISO/IEC 23090-3 | ITU-T Rec. H.266, Versatile Video Coding, DIS, Jul 2020.
25. B. Girod, N. Farber, and U. Horn, “Scalable Codec Architectures for Internet Video-on-Demand”, in Proc. Thirty-First Asilomar Conference on Signals, Systems and Computers, Pacific Grove, CA, USA, 1997, pp. 357-361.
26. N. Farber and B. Girod, "Robust H.263 compatible video transmission for mobile access to video servers," Proceedings of International Conference on Image Processing, Santa Barbara, CA, 1997, pp. 73-76.
27. S. Ericsson, B. Girod, N. Farber, Y. Reznik, “Method and apparatus for providing scalable pre-compressed digital video with reduced quantization based artifacts”, US Patent 6,480,541, filed October 23, 1998, granted November 12, 2002.
28. S. Ericsson, B. Girod, N. Farber, Y. Reznik, “Method and apparatus for providing scalable pre-compressed digital video with reduced quantization based artifacts”, US Patent 7,075,986, filed: November 12, 2002, granted July 11, 2006.
29. M. Karczewicz and R. Kurceren, "The SP- and SI-frames design for H.264/AVC," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 13, no. 7, pp. 637-644, July 2003.
30. M. Karczewicz, R. Kurceren, “Video decoder architecture and method for using same”, US Patent 7,477,689, Priority: 2001-01-03, Filed: Jun 16, 2004, Issued: Jan 13, 2009.
31. HESP documentation: <https://www.hespalliance.org/>
32. Y. A. Reznik, K.O. Lillevold, and B. Zhang, “On video codec HRD behavior and DASH MPD attributes,” Mpeg input document M56084, Jan 2021.

## Meeting Discussion

Summary:

* this reacts to the use case of unclean/dirty switch and random access - referring to referencing "wrong" reference frames.
* advise to not run into such a situation with undefined decoder behaviour

For issues that relate to "unclean" operations, we should not prohibit per se, but if permitted, we should make very clear guidance on how to ensure conformance at the receiver towards the video decoder, both on bitstream and HRD level. Also need to check output signal range, if this may cause problem.