DirectX Video Acceleration (DXVA) Specification for H.264/MPEG-4 Scalable Video Coding (SVC) Off-Host VLD Mode Decoding

Gary J. Sullivan, Amy Lu, Yongjun Wu, and Ming-Chieh Lee
Microsoft Corporation
June 2012

Applies to:

DirectX Video Acceleration

Summary: This document is the specification for support of Scalable Video Coding (SVC) according to the H.264/MPEG-4 video coding standard within the Microsoft Windows DirectX Video Acceleration (DXVA) API/DDI context. The document describes high-level design concepts and specific SVC extensions to DXVA interfaces and data structures of H.264/MPEG-4 AVC decoding. This document specifies only off-host VLD profiles for H.264/MPEG4 SVC decoding.

Contents

1. Introduction

This document is the specification for decoding support of the Scalable Video Coding (SVC) extensions to the H.264/MPEG-4 Advanced Video Coding (AVC) standard in the context of the Microsoft Windows DirectX Video Acceleration (DXVA) API/DDI.

In the terminology of this specification, the term "ordinary H.264/AVC bitstream" refers to an H.264/AVC bitstream that does not use the SVC and MVC extensions that are specified in Annexes G and H of the H.264/AVC specification.

This specification assumes that the reader is familiar with the H.264/AVC standard and its SVC extensions in Annex G, and the Microsoft DXVA specification for ordinary H.264/AVC bitstream decoding.

Accelerators that conform to this specification must also support the decoding of ordinary H.264/AVC bitstreams that conform to the profiles that can be used as the base layer in corresponding SVC profiles as follows:

- Accelerators supporting the Scalable Baseline profile must also support the decoding of all bitstreams compatible with decoders conforming to the Constrained Baseline profile.
- Accelerators supporting the Scalable High profile must also support the decoding of all bitstreams compatible with decoders conforming to the Constrained Baseline, Main and High profiles.

References in this specification to "subclauses" refer to subclauses of the H.264/AVC standard specification.

References in this specification to "sections" refer to sections of this document or to sections of the Microsoft DXVA specification for ordinary H.264/AVC bitstream decoding.

2. Normative references

2.1 The H.264/AVC standard

The Scalable Video Coding (SVC) extensions to the H.264/AVC standard (referred to hereafter simply as SVC) are specified in Annex G of the following document:

ITU-T Rec. H.264 | ISO/IEC 14496-10 *Advanced video coding for generic audiovisual services*

The standard is available at http://www.itu.int/rec/T-REC-H.264. Unless otherwise specified, this document refers to the edition approved by ITU-T in January 2012 (posted at the ITU-T web site link above).

2.2 The DXVA Specification for Ordinary H.264/AVC Bitstream Decoding

The DXVA specification for ordinary H.264/AVC bitstream decoding is found in the Microsoft publication entitled "DirectX Video Acceleration Specification for H.264/AVC Decoding".

The specification is available at http://www.microsoft.com/downloads/details.aspx?FamilyID=3d1c290b-310b-4ea2-bf76-714063a6d7a6&DisplayLang=en. Unless otherwise specified, this document refers to the edition that includes updates as of 6 Jan. 2011.

3. Definitions

The definitions of the following terms are specified in the H.264/AVC standard and are used throughout this specification:

G.3.5 base quality layer representation

G.3.45 reference base picture

G.3.49 reference layer representation

G.3.66 target layer representation

4. Overview of the SVC extensions of H.264/AVC

4.1 Basic Design Structure

The SVC design, primarily specified in Annex G of the H.264/MPEG-4 Advanced Video Coding (AVC) standard, closely follows that found in the ordinary non-SVC form of the H.264/AVC standard. The data for a coded video sequence is structured into "access units" that are conveyed in the bitstream in decoding order. Each access unit contains one "primary coded picture" for a particular instant in output (display) time. Within some of the access units, there is a coded "base layer" picture that is formatted as an ordinary H.264/AVC coded picture and can be decoded by an ordinary non-SVC decoder that supports the H.264/AVC Constrained Baseline Profile or H.264/AVC High Profile. Within the same access unit or in other access units, there are one or more additional scalable enhancement layer representations that each represents an additional "enhancement layer" of a SVC encoded bitstream for the same instant in time. The primary coded picture may be either an enhancement layer picture (when enhancement layer data is present in the access unit) or a base layer picture (when no enhancement layer data is present in the access unit). The decoding of an access unit produces a decoded picture corresponding to the primary coded picture, and may additionally produce an associated stored "reference base picture" for use as a reference for the decoding process of other pictures. SVC supports three types of classes of scalability: temporal, quality, and spatial scalability as described below.

4.1.1 Temporal Scalability

A bitstream provides temporal scalability when the set of corresponding access units can be partitioned into a temporal base layer and one or more temporal enhancement layers so that the enhanced video has a higher frame rate than the base layer video. To support temporal scalability with a reasonable number of temporal layers, basically no changes to the design of H.264/AVC were required. The only significant change in SVC to support temporal scalability is for the high-level header signaling of the temporal layers, which is primarily handled by the host software decoder. Hence, accelerators capable of DXVA operation for ordinary H.264/AVC decoding require only minimal changes to support this feature (primarily only the recognition of NAL unit type 20 as another way to send coded slice data).

4.1.2 Spatial Scalability

A bitstream provides spatial scalability when the set of corresponding access units can be partitioned into a spatial base layer with a lower resolution, and one or more spatial enhancement layers that increase the spatial resolution of the decoded video sequence. In order to improve coding efficiency in comparison to simulcasting different spatial resolutions, additional so-called inter-layer prediction mechanisms are incorporated. Inter-layer prediction tools specified in SVC are described in Section 3.2.

4.1.3 Quality Scalability

A bitstream provides quality scalability when the set of corresponding access units can be partitioned into a quality base layer with a lower fidelity, and one or more quality enhancement layers that increase the visual quality of the decoded sequence without changing the spatial resolution. Because there is no change of spatial resolution in this case, the inter-layer prediction mechanisms are simplified in this case (as there is no need to apply upsampling and inter-layer deblocking to the reference layer macroblocks used for prediction of the enhancement layer).

4.2 Inter-layer Prediction for Spatial and Quality Scalability

The use of inter-layer prediction is indicated by a syntax element called no_inter_layer_pred_flag in the NAL unit header SVC extension. When no_inter_layer_pred_flag is 0, inter-layer prediction may be employed to improve coding efficiency. The slice header syntax element ref_layer_dq_id in the scalable extension specifies the layer representation of the current coded picture that is used for inter-layer prediction of the enhancement-layer coded picture. When ref_layer_dq_id is not present, it is inferred by rules specified in subclause G.7.4.3.4. The value of ref_layer_dq_id is the same across all slices of a layer representation.

SVC supports three inter-layer prediction tools: prediction of macroblock modes and associated motion parameters, prediction of motion parameters, and prediction of residual difference.

4.2.1 Inter-layer Mode and Motion Prediction

Inter-layer mode and motion prediction is signaled by a syntax element called base_mode_flag at the macroblock level. When base_mode_flag is 1 and a corresponding 8x8 block in the reference layer lies inside an intra coded macroblock, the macroblock in the target layer is predicted by *inter-layer intra prediction*. When base_mode_flag is 1 and the reference layer 8x8 block is inter coded, the macroblock in the enhancement layer is also inter coded. In the latter case, the partitioning data of the macroblock in the enhancement layer, together with the associated reference indexes and motion vectors, are derived from the corresponding data of the co-located 8x8 block in the reference layer (with resampling as necessary in the case of spatial scalability).

4.2.2 Inter-layer Motion Prediction

Inter-layer motion prediction is signaled by a syntax element called motion_prediction_flag at the macroblock partition or sub-macroblock level (i.e., for each 16x16, 16x8, 8x16, or 8x8 block region). When motion_prediction_flag is 1, the corresponding reference indexes for the macroblock partition are not coded in the enhancement layer (instead the reference indexes of the co-located reference layer macroblock partition or sub-macroblock are used), and the corresponding motion vector predictors for all blocks of the enhancement layer macroblock partition are formed by the motion vectors of the co-located blocks in the reference layer (scaled as necessary in the case of spatial scalability). When motion_prediction_flag is 0, the reference indexes and motion vectors for the corresponding macroblock partition in the target layer are coded using conventional H.264/AVC macroblock types in the corresponding spatial layer.

4.2.3 Inter-layer Residual Prediction

Inter-layer residual prediction is signaled by a syntax element called residual_prediction_flag at the macroblock level. When residual_prediction_flag is 1, the residual difference of the corresponding 8x8 sub-macroblock in the reference layer is used as a prediction for the residual signal of the macroblock in the enhancement layer (with block-wise upsampling using a bilinear filter as necessary in the case of spatial scalability). The residual prediction is done on the residual block basis, and can be used for all macroblocks in the enhancement layer (regardless of whether they are coded using inter-layer mode and motion prediction or just inter-layer motion prediction or only the conventional H.264/AVC macroblock types).

4.3 Reference Pictures

In SVC, the ordinary H.264/AVC reference picture list construction process is modified using two additional syntax elements: store_ref_base_pic_flag and use_ref_base_pic_flag. When store_ref_base_pic_flag is 1, the base quality layer representation may need to be stored in the decoded picture buffer (DPB) for inter prediction of subsequent pictures in decoding order in addition to storing the decoded target layer representation for output. Note that "reference base pictures" are not considered "decoded pictures".

For each coded slice, use_ref_base_pic_flag signals whether the base quality layer representation or the target layer representation of the reference picture is employed for motion-compensated prediction. When use_ref_base_pic_flag is 1, reference base pictures (when present) and decoded pictures (when reference base pictures are not present) are used as reference pictures for inter prediction, as specified in subclause G.8.2.3. When use_ref_base_pic_flag is 0, the reference base pictures are not used as reference pictures for inter prediction (i.e., only the decoded pictures are used for inter prediction). Note that the reference base pictures and decoded pictures have the same spatial resolution (because the value of dependency_id of the reference base pictures is equal to the maximum value of dependency_id for the current coded picture).

In H.264/AVC, reference pictures are placed into one or two ordered lists prior to decoding a slice. P slices use a single reference picture list, RefPicList0, and B slices use two reference picture lists, RefPicList0 and RefPicList1. The reference picture(s) used for the decoding of a coded block are identified by a reference index into RefPicList0 and/or RefPicList1 (if present). So-called "Key pictures" have both store_ref_base_pic_flag and use_ref_base_pic_flag being set to 1. SVC follows essentially the same procedure to construct reference picture list(s) for each coded slice as in ordinary H.264/AVC. The details are specified in subclause G.8.2.3.

This specification is intended to be sufficient for decoding the Scalable Baseline and Scalable High profiles defined in subclause G.10.1.1 and G.10.1.2, respectively. To support other SVC profiles would require incorporating some additional features into the design.

5. Basic DXVA design approach

5.1 General

This section provides an overview of the design for DXVA decoding of SVC bitstreams. It is intended as background overview information, and may be helpful for understanding the specific aspects defined in the subsequent sections. In the case of conflicts, later sections of this document override this section.

SVC DXVA support is designed for maximal consistency with the DXVA design for ordinary H.264/AVC bitstream decoding. To a maximum extent, the same basic data flow and similar data structures are used.

5.2 Primary Differences Relative to Ordinary H.264/AVC Decoding

5.2.1 Reference Picture Handling Scheme

When store_ref_base_pic_flag is 1, the host software decoder determines whether the decoding and storage of the reference base picture is necessary, and if so, instructs the accelerator to decode the reference base picture associated with the current coded picture (for use as a reference for inter prediction of subsequent pictures in decoding order) as well as the current primary coded picture. In such a case, the decoding process affects two output DXVA surfaces. As reference base pictures are also stored in DPB and the maximum size of the DPB for SVC specified in subclause G.10.2.1 is the same as for ordinary H.264/AVC, the memory capacity requirement for reference pictures in DXVA SVC is approximately the same as compared to DXVA for ordinary H.264/AVC bitstreams.

When use_ref_base_pic_flag is 1, the host software decoder uses the reference base pictures (when present) and decoded pictures (if reference base pictures are not present) as reference pictures for inter prediction as specified in subclause G.8.2.3. When use_ref_base_pic_flag is 0, the host software decoder uses the decoded pictures (regardless of whether the associated reference base pictures are present) as reference pictures for inter prediction. Note that the reference base pictures and decoded pictures have the same spatial resolution, so the manipulation of reference picture lists by the host software decoder is transparent to the accelerator.

5.2.2 Inter-Layer Prediction Support

When no_inter_layer_pred_flag is 0, inter-layer prediction may be used for decoding a coded slice in the target layer representation. In this case, the bitstream data corresponding to the coded slices in the reference layer representation must be conveyed to the accelerator. The corresponding reference layer representation of the current coded picture is signaled through the slice header syntax element ref_layer_dq_id.

When the long slice control data structure is used, the host software decoder parses the slice header to retrieve ref_layer_dq_id (or derives ref_layer_dq_id when it is not present), and sends the picture parameters, slice control parameters, and bitstream data associated with the coded slices in the reference layer representation to the accelerator in addition to sending the coded slices in the target layer representation. The host software decoder is recommended to not send coded slices that are neither in the reference layer representation nor in the target layer representation to the accelerator. The variable MinNoInterLayerPredFlag is set equal to the minimum value of no_inter_layer_pred_flag for the slices of the layer representation. The host software decoder is recommended to not send the reference layer data to the accelerator if the variable MinNoInterLayerPredFlag is 1. However, if the accelerator receives any such unnecessary coded slices, the accelerator shall discard the unnecessary data and perform the decoding process correctly without incurring any error.

When the short slice control data structure is used, the host software decoder must still parse the slice header in the target layer representation, in the reference layer representation, and in any other layer representations that are necessary for inter prediction, and perform DPB management for decoded picture output, picture deletion, reference picture marking/unmarking, etc. The same optimizations apply in regard to MinNoInterLayerPredFlag and the handling of unnecessary reference layer data.

The accelerator performs off-host parsing to retrieve syntax elements of the macroblock level, and derives the variables of the reference layer representation for inter-layer motion compensation. When inter-layer intra prediction is indicated, the accelerator needs to reconstruct the co-located intra signal of the reference layer. When the reference and target layer representations have different resolutions, the accelerator must also upsample the co-located reference layer data.

In general, the SVC host software decoder requires modification to manage the DPB state for the coded pictures and base layer reference pictures, create the reference frame list for each coded picture, perform reference picture marking and base reference picture marking, prepare reference picture list(s), parse the sequence parameter set extensions, picture parameter set extensions, slice headers, prefix NAL units, reference picture list initialization and reordering, and (possibly) associated additional SEI messages. Accelerator-side modifications primarily relate to header format issues, inter-layer motion compensation, and memory requirements associated with inter-layer prediction.

As in the usual DXVA2 scheme for ordinary H.264/AVC bitstream decoding, the accelerator is responsible for tracking the sequential dependencies between the commanded decoding operations – ensuring that read operations used in prediction processes are not performed until the write operations to fill the frame buffer with the correct decoded data have been completed.

5.3 Support Only for Off-Host VLD Operation

Over time, the level of industry interest in supporting modes of DXVA operation other than off-host VLD operation (e.g., as in the DXVA_ModeH264_MoComp_NoFGT and DXVA_ModeH264_IDCT_NoFGT modes of DXVA operation for ordinary H.264/AVC bitstreams) appears to have waned. We therefore do not specify such modes of operation for SVC decoding.

6. DXVA Extensions for SVC

This section specifies the SVC-specific extensions to the data flow and data structures of the specification for ordinary H.264/AVC bitstream decoding. This section assumes that the reader is already familiar with the DXVA specification for ordinary (non-SVC) H.264/AVC bitstream decoding. DXVA specifications for SVC decoding follow the DXVA specifications for ordinary H.264/AVC decoding, except as specified in the following sections.

6.1 Extensions to DXVA Decoding Operations (Section 1.5)

Section 1.5 (DXVA decoding operations) of the DXVA specification for ordinary H.264/AVC bitstream decoding specifies the basic sequence of operations for H.264/AVC decoding. In order to minimize the API/DDI changes and make SVC DXVA decoding consistent with H.264/AVC DXVA picture decoding, **BeginFrame**, **Execute**, and **EndFrame** calls are used to perform the decoding of each picture in a scalable bitstream, as well as the decoding of its associated reference base picture as necessary.

Unlike ordinary H.264/AVC DXVA decoding, the decoding process of SVC DXVA may require the input of data buffers associated with multiple layer representations: target layer representation, a reference layer representation (when MinNoInterLayerPredFlag is 0), and other layer representations that are needed for inter-layer prediction. Each layer representation involves a picture parameters buffer, a quantization matrix buffer, one or more slice control buffers, and one or more bitstream data buffers (Note that there might be multiple slice control buffers and bitstream data buffers, possibly with wBadSliceChopping enabled, associated with one picture parameter buffer and one quantization matrix buffer, if one bitstream data buffer cannot hold the data for the entire access unit). To guarantee that the accelerator associates data buffers with a layer representation correctly, the host software decoder must send buffers to the accelerator between each BeginFrame/EndFrame pair based on the following rules:

- Each layer representation is associated with four types of buffers:
  - One picture parameters buffer of the type of *DXVA_PicParams_H264_SVC*,
  - One quantization matrix buffer,
  - One or more slice control buffers of the type of *DXVA_Slice_H264_SVC_Long* or *DXVA_Slice_H264_SVC_Short*, and
  - One or more bitstream data buffers.
- Groups of buffers must be sent to the accelerator in decoding order.

The picture parameters buffer uses the type definition *DXVA_PicParams_H264_SVC* specified in the later sections, rather than the type definition *DXVA_PicParams_H264* that is used in ordinary H.264/AVC DXVA decoding. The slice control buffer uses the type *DXVA_Slice_H264_SVC_Long* or the type *DXVA_Slice_H264_SVC_Short* definition specified in the later sections, rather than the *DXVA_Slice_H264_Long* or *DXVA_Slice_H264_Short* that is used in ordinary H.264/AVC DXVA decoding.

When the storage of a reference base picture is not needed, the host software decoder uses the following calls to decode the decoded picture, in a similar manner as for H.264/AVC DXVA picture decoding:

1. The decoder calls **BeginFrame** with the index of the decoded output surface associated with the decoded picture.
2. The decoder calls **Execute** one or more times with one or more data buffers (in obedience with the above buffer type sequence constraints and decoding order constraints).
3. The decoder calls **EndFrame** with the index of the decoded output surface associated with the decoded picture.

When the generation of reference base picture is signaled in the base quality layer representation with nal_ref_idc not equal to 0 and store_ref_base_pic_flag equal to 1, the host software decoder shall structure the function calls for decoding the reference base picture and the decoded picture as follows:

1. The decoder calls **BeginFrame** with the index of the decoded output surface associated with the reference base picture.
2. The decoder calls **BeginFrame** with the index of the decoded output surface associated with the decoded picture.
3. The decoder calls **Execute** one or more times with one or more data buffers (in obedience with the above buffer type sequence constraints and decoding order constraints).
4. The decoder calls **EndFrame** with the index of the decoded output surface associated with the reference base picture.
5. The decoder calls **EndFrame** with the index of the decoded output surface associated with the decoded picture.

6.2 Extensions to Configuration Parameters (Section 2.0)

This section describes the extensions to the configuration parameters for SVC decoding as previously specified in Section 2.0 of the DXVA specification for ordinary H.264/AVC bitstream decoding.

bConfigBitstreamRaw
Value equal to 5 or 6 for SVC long or short slice header formats in VLD DXVA profiles, respectively.

bConfigResidDiffHost
Equal to 0 for SVC VLD DXVA profiles (with bConfigBitstreamRaw equal to 5 or 6).

bConfigResidDiffAccelerator
Equal to 1 for SVC VLD DXVA profiles (with bConfigBitstreamRaw equal to 5 or 6).

6.3 Extensions to Picture Parameters Data Structure (Section 4.0)

This section describes the use of the DXVA picture parameters data structure for SVC decoding as previously specified in Section 4.0 of the DXVA specification for ordinary H.264/AVC bitstream decoding.

```
typedef struct _DXVA_PicParams_H264_SVC {

(all the parameters of the Section 4.0 Picture Parameters Data Structure,
followed by the following additional parameters)

  UCHAR   RefBasePicFlag[16];
  union {
    struct {
      USHORT  inter_layer_deblocking_filter_control_present_flag : 1;
      USHORT  chroma_phase_x_plus1_flag : 1;
      USHORT  seq_ref_layer_chroma_phase_x_plus1_flag : 1;
      USHORT  adaptive_tcoeff_level_prediction_flag : 1;
      USHORT  slice_header_restriction_flag : 1;
      USHORT  store_ref_base_pic_flag : 1;
      USHORT  ShiftXYis16Flag : 1;
      USHORT  constrained_intra_resampling_flag : 1;
      USHORT  ref_layer_chroma_phase_x_plus1_flag : 1;
      USHORT  tcoeff_level_prediction_flag : 1;
      USHORT  IdrPicFlag : 1;
      USHORT  NextLayerSpatialResolutionChangeFlag : 1;
      USHORT  NextLayerMaxTCoeffLevelPredFlag : 1;
    }
    USHORT  wBitFields;
  };

  UCHAR   extended_spatial_scalability_idc;
  UCHAR   chroma_phase_y_plus1;
  union {
    struct {
      SHORT   seq_scaled_ref_layer_left_offset;
      SHORT   seq_scaled_ref_layer_top_offset;
      SHORT   seq_scaled_ref_layer_right_offset;
      SHORT   seq_scaled_ref_layer_bottom_offset;
    };
    struct {
      SHORT   scaled_ref_layer_left_offset;
      SHORT   scaled_ref_layer_top_offset;
      SHORT   scaled_ref_layer_right_offset;
      SHORT   scaled_ref_layer_bottom_offset;
    };
  };
  UCHAR   seq_ref_layer_chroma_phase_y_plus1;
```



```
  UCHAR   LayerType;
  UCHAR   dependency_id;
  UCHAR   quality_id;
  USHORT  ref_layer_dq_id;
  UCHAR   disable_inter_layer_deblocking_filter_idc;
  CHAR    inter_layer_slice_alpha_c0_offset_div2;
  CHAR    inter_layer_slice_beta_offset_div2;
  UCHAR   ref_layer_chroma_phase_y_plus1;

  SHORT   NextLayerScaledRefLayerLeftOffset;
  SHORT   NextLayerScaledRefLayerRightOffset;
  SHORT   NextLayerScaledRefLayerTopOffset;
  SHORT   NextLayerScaledRefLayerBottomOffset;
  USHORT  NextLayerPicWidthInMbs;
  USHORT  NextLayerPicHeightInMbs;
  UCHAR   NextLayerDisableInterLayerDeblockingFilterIdc;
  UCHAR   NextLayerInterLayerSliceAlphaC0OffsetDiv2;
  UCHAR   NextLayerInterLayerSliceBetaOffsetDiv2;

  UCHAR   DeblockingFilterMode;

} DXVA_PicParams_H264_SVC, *LPDXVA_PicParams_H264_SVC;
```

To decode SVC bitstreams with a DXVA accelerator, in addition to all of the parameters and flags specified in Section 4.0 (Picture Parameters Data Structure) in the DXVA specification for ordinary H.264/AVC bitstream decoding, the additional picture parameters shown above shall be sent to the accelerator. Instead of creating and filling *DXVA_PicParams_H264*, the host software decoder needs to create and fill *DXVA_PicParams_H264_SVC* for each coded picture, which contains all of the parameters specified in the *DXVA_PicParams_H264* data structure and some additional picture parameters.

Clarifications on the use of individual parameters of *DXVA_PicParams_H264* are as follows:

wBitFields
Provides an alternate way to access the bit fields specified above.

CurrPic
Specifies the uncompressed destination as specified in the following table:

Value	**Description**
Index associated with the uncompressed destination surface that specifies the destination surface for the current decoded picture.	The current picture parameters buffer is referred to by one or more slice control buffers associated with the target layer representation.
Index associated with the uncompressed destination surface that specifies the destination surface for the reference base picture.	The current picture parameters buffer is referred to by one or more slice control buffers associated with the base quality layer representation with the nal_ref_idc syntax element not equal to 0 and the store_ref_base_pic_flag syntax element equal to 1.
0xFF	Otherwise

RefBasePicFlag
Contains a list of 16 values, with 1 indicating that the corresponding reference picture in RefFrameList is a reference base picture and 0 indicating that the corresponding reference picture in RefFrameList is a decoded picture.

RefFrameList
Contains a list of 16 uncompressed frame buffer surfaces, each contains either a decoded picture or a reference base picture. Entries that will not be used for decoding the current picture, or any subsequent pictures, are indicated by setting **bPicEntry** to 0xFF. If **bPicEntry** is not 0xFF, the entry may be used as a reference surface for decoding the current picture or a subsequent picture (in decoding order). Reference base pictures may have the same PicOrderCnt value as the associated decoded picture but the **RefBasePicFlag** value of the reference base picture is 1 while the **RefBasePicFlag** value of the decoded pictures is 0. All uncompressed surfaces that correspond to pictures currently marked as "used for reference" must appear in the RefFrameList array. Non-reference surfaces (those which only contain pictures for which the value of RefPicFlag was 0 when the picture was decoded) shall not appear in RefFrameList for a subsequent picture. In addition, surfaces that contain only pictures marked as "unused for reference" shall not appear in RefFrameList for a subsequent picture.

ShiftXYis16Flag
Must be 0 or 1, with 1 indicating the resampling precision is 16 and 0 indicating the resampling precision is calculated based on the resolution for the reference layer representation.

NextLayerSpatialResolutionChangeFlag
Indicates the derived variable SpatialResolutionChangeFlag of the layer representation with ref_layer_dq_id equal to DQId of the current layer representation when LayerType is equal to 1 or the layer representation with DQId less than and closest to the DQId of the current layer representation when LayerType is equal to 5. If LayerType of the current layer representation is not equal to 1 or 5, NextLayerSpatialResolutionChangeFlag has no meaning and shall be 0, and the accelerator shall ignore its value.

NextLayerMaxTCoeffLevelPredFlag
Indicates the derived variable MaxTCoeffLevelPredFlag of the layer representation with ref_layer_dq_id equal to DQId of the current layer representation when LayerType is equal to 1 or the layer representation with DQId less than and closest to the DQId of the current layer representation when LayerType is equal to 5. If LayerType of the current layer representation is not equal to 1 or 5, NextLayerMaxTCoeffLevelPredFlag has no meaning and shall be 0, and the accelerator shall ignore its value.

DeblockingFilterMode

Value	Description
0	The boundary filter strength is decided as if profile_idc equal to 83.
1	The boundary filter strength is decided as if profile_idc equal to 86.

Indicates the algorithm for deciding boundary filter strength when DQId is greater than 0. When DQId is zero, DeblockingFilterMode has no meaning and shall be 0, and the accelerator shall ignore its value. Values other than 0 and 1 are prohibited.

LayerType

Value	Description
0	The current layer representation is the target layer representation with the same DQId as the target layer representation of the previous picture.
1	The current layer representation is the reference layer representation or any layer representation that is needed for deriving syntax elements, derived variables, or decoded samples of the reference layer representation.
2	The current layer representation is neither the target layer representation nor needed for inter-layer prediction.
3	The current layer representation is the target layer representation that has dependency_id equal to the dependency_id of the target layer representation of the previous picture and quality_id not equal to the quality_id of the target layer representation of the previous picture.
4	The current layer representation is the target layer representation that has dependency_id not equal to the dependency_id of the target layer representation of the previous layer.
5	The current layer representation is inferred as the reference layer representation or as needed for deriving syntax elements, derived variables, or decoded samples of the reference layer representation.
6	The current layer representation is inferred as neither the target layer representation nor needed for inter-layer prediction.
7	The host decoder cannot readily determine this value because all the slices associated with the reference layer representation or all the slices associated with a layer representation that is needed for deriving syntax elements, derived variables, or decoded samples of the reference layer representation is missing.

The value 0 indicates that the current layer representation is the target layer representation.

The value 1 indicates that the current layer representation is needed for deriving syntax elements, derived variables, or decoded samples of the reference layer representation. The accelerator may need to parse the slice bitstream for inter-layer motion compensation or reconstruct intra blocks if inter-layer intra prediction is employed.

The value 2 indicates that the current layer representation is neither of the target layer representation nor needed for inter-layer prediction. The accelerator does not have to do any particular processing in response to this value, but the host software decoder may send such coded slices to the accelerator for e.g. potential error concealment purposes.

The value 3 indicates that the current layer representation is the target layer representation that has dependency_id equal to the dependency_id of the target layer representation of the previous picture and quality_id not equal to the quality_id of the target layer representation of the previous picture. The accelerator shall perform the complete decoding of the picture if only layer representations with LayerType 0, 1, 2, or /and 4 are present between a BeginFrame/EndFrame pair.

The value 4 indicates that the current layer representation is the target layer representation that has dependency_id not equal to the dependency_id of the target layer representation of the previous picture. The values 0, 3, and 4 shall not coexist between a pair of BeginFrame/EndFrame pair. There shall be at least one layer representation with LayerType 0, 3, or 4 sent to the accelerator between a BeginFrame/EndFrame pair.

The value 5 indicates that the current layer representation is inferred to be needed for deriving syntax elements, derived variables, or decoded samples of the reference layer representation. The value 6 indicates that the current layer representation is inferred to be neither the target layer representation nor needed for inter-layer prediction. The host decoder shall not specify value 5 or 6 when the value of 1 or 2 is determinable for a layer representation. The inference of value 5 or 6 is obtained from, for example, LayerType of the layer representations of the previous pictures with the same DQId.

The value 7 indicates that the host decoder cannot readily determine this value because all the slices associated with the reference layer representation are missing or all the slices associated with a layer representation that is needed for deriving syntax elements, derived variables, or decoded samples of the reference layer representation are missing. The host decoder shall not specify value 7 when the value of 0-4 is determinable for a layer representation. It is recommended for the host decoder to use the value 5 or 6 rather than the value 7 when LayerType 1 or 2 is indeterminate. Layer representations with LayerType equal to 7 shall not coexist with layer representations with LayerType equal to 5 or 6 between a BeginFrame/EndFrame pair.

Other values are not allowed by this specification.

NextLayerScaledRefLayerLeftOffset, NextLayerScaledRefLayerRightOffset, NextLayerScaledRefLayerTopOffset, NextLayerScaledRefLayerBottomOffset
Indicate the horizontal and vertical cropping offsets, in units of samples, of the layer representation with ref_layer_dq_id equal to DQId of the current layer representation when LayerType value is equal to 1 or the layer representation with DQId less than and closest to the DQId of the current layer representation when LayerType is equal to 5. In other words, NextLayerScaledRefLayerLeftOffset, NextLayerScaledRefLayerRightOffset, NextLayerScaledRefLayerTopOffset, and NextLayerScaledRefLayerBottomOffset are equal to the values of variables ScaledRefLayerLeftOffset, ScaledRefLayerRightOffset, ScaledRefLayerTopOffset, and ScaledRefLayerBottomOffset of the layer representation with ref_layer_dq_id equal to DQId of the current layer representation when LayerType is equal to 1 or the layer representation with DQId less than and closest to the DQId of the current layer representation when LayerType is equal to 5, respectively. If LayerType of the current layer representation is not equal to 1 or 5, NextLayerScaledRefLayerLeftOffset, NextLayerScaledRefLayerRightOffset, NextLayerScaledRefLayerTopOffset, and NextLayerScaledRefLayerBottomOffset have no meaning and shall be 0, and the accelerator shall ignore their values.

NextLayerPicWidthInMbs, NextLayerPicHeightInMbs
Indicate the width and height, in units of macroblocks, of the layer representation with ref_layer_dq_id equal to DQId of the current layer representation when LayerType equal to 1 or the layer representation with DQId less than and closest to the DQId of the current layer representation when LayerType is equal to 5. If LayerType of the current layer representation is not equal to 1 or 5, NextLayerPicWidthInMbs and NextLayerPicHeightInMbs have no meaning and shall be 0, and the accelerator shall ignore their values.

NextLayerDisableInterLayerDeblockingFilterIdc
Indicate the syntax element disable_inter_layer_deblocking_filter of the layer representation with ref_layer_dq_id equal to DQId of the current layer representation when LayerType is equal to 1 or the layer representation with DQId less than and closest to the DQId of the current layer representation when LayerType is equal to 5. If LayerType of the current layer representation is not equal to 1 or 5, NextLayerDisableInterLayerDeblockingFilterIdc has no meaning and shall be 0, and the accelerator shall ignore its value.

NextLayerInterLayerSliceAlphaC0OffsetDiv2
Indicate the syntax element inter_layer_slice_alpha_c0_offset_div2 of the layer representation with ref_layer_dq_id equal to DQId of the current layer representation when LayerType is equal to 1 or the layer representation with DQId less than and closest to the DQId of the current layer representation when LayerType is equal to 5. If LayerType of the current layer representation is not equal to 1 or 5, NextLayerInterLayerSliceAlphaC0OffsetDiv2 has no meaning and shall be 0, and the accelerator shall ignore its value.

NextLayerInterLayerSliceBetaOffsetDiv2
Indicate the syntax element inter_layer_slice_beta_offset_div2 of the layer representation with ref_layer_dq_id equal to DQId of the current layer representation when LayerType is equal to 1 or the layer representation with DQId less than and closest to the DQId of the current layer representation when LayerType is equal to 5. If LayerType of the current layer representation is not equal to 1 or 5, NextLayerInterLayerSliceBetaOffsetDiv2 has no meaning and shall be 0, and the accelerator shall ignore its value.

The rest of the structure members correspond to the H.264/AVC syntax elements of the same name and affect the decoding process accordingly. Syntax elements or derived variables **ref_layer_dq_id. disable_inter_layer_deblocking_filter_idc, inter_layer_slice_alpha_c0_offset_div2, inter_layer_slice_beta_offset_div2, constrained_intra_resampling_flag, ref_layer_chroma_phase_x_plus1_flag, ref_layer_chroma_phase_y_plus1, scaled_ref_layer_left_offset, scaled_ref_layer_top_offset, scaled_ref_layer_right_offset, scaled_ref_layer_bottom_offset, tcoeff_level_prediction_flag, dependency_id, quality_id**, and **IdrPicFlag** are defined in picture parameters buffer as the values of slice header syntax elements which must be same across all slices of a layer representation. If the syntax element is not present in the bitstream and has no inferred value according to the H.264/AVC specification, the host software decoder shall set the value to 0. Accelerators can rely on this constraint being fulfilled.

6.4 Extensions to Slice Control Data Structure (Section 6.0)

This section describes the use of the slice control data structure for SVC decoding as previously specified in Section 6.0 of the DXVA specification for ordinary H.264/AVC bitstream decoding.

bConfigBitstreamRaw	Slice Data Control Data Structure
5	DXVA_Slice_H264_SVC_Long
6	DXVA_Slice_H264_SVC_Short

When bConfigBitstreamRaw is 5 or 6, the slice control buffer is accompanied by a bitstream data buffer. When bConfigBitstreamRaw is 5, the decoder uses the slice control data structure of *DXVA_Slice_H264_SVC_Long*. When bConfigBitstreamRaw is 6, the decoder uses the slice control data structure of *DXVA_Slice_H264_SVC_Short* as specified below.

```
typedef struct _DXVA_Slice_H264_SVC_Long {

(all the parameters of the Section 6.1 Long Slice Control Data Structure,
followed by the following additional parameters)

  union {
    struct {
      USHORT  no_inter_layer_pred_flag : 1;
      USHORT  base_pred_weight_table_flag : 1;
      USHORT  slice_skip_flag : 1;
      USHORT  adaptive_base_mode_flag : 1;
      USHORT  default_base_mode_flag : 1;
      USHORT  adaptive_motion_prediction_flag : 1;
      USHORT  default_motion_prediction_flag : 1;
      USHORT  adaptive_residual_prediction_flag : 1;
      USHORT  default_residual_prediction_flag : 1;
    }
    USHORT wBitFields;
  };
  USHORT  num_mbs_in_slice_minus1;
  UCHAR   scan_idx_start;
  UCHAR   scan_idx_end;

} DXVA_Slice_H264_SVC_Long, *LPDXVA_Slice_H264_SVC_Long;


typedef struct _DXVA_Slice_H264_SVC_Short {

(all the parameters of the Section 6.1 Short Slice Control Data Structure,
followed by the following additional parameters)

  union {
    struct {
      USHORT  no_inter_layer_pred_flag : 1;
    }
    USHORT wBitFields;
  };

} DXVA_Slice_H264_SVC_Short, *LPDXVA_Slice_H264_SVC_Short;
```

To decode SVC bitstreams with a DXVA accelerator, in addition to all of the parameters and flags specified in Section 6.1 (Long and Short Slice Control Data Structures) in the DXVA specification for ordinary H.264/AVC bitstream decoding, the additional slice control parameters shown above shall be sent to the accelerator for each picture. Instead of creating and filling *DXVA_Slice_H264_Long* or *DXVA_Slice_H264_Short* the host software decoder needs to create and fill *DXVA_Slice_H264_SVC_Long* or *DXVA_Slice_H264_SVC_Short* for each coded picture, which contains all of the parameters specified in the *DXVA_Slice_H264_Long* or *DXVA_Slice_H264_Short* data structure and some additional slice control parameters.

Clarifications on the use of individual parameters of *DXVA_PicParams_H264* are as follows:

wBitFields
Provides an alternate way to access the bit fields specified above.

BSNALunitDataLocation
If wBadSliceChopping is 0 or 1, this member locates the NAL unit with nal_unit_type equal to 1, 5, or 20 for the current slice. The value is the byte offset, from the start of the bitstream data buffer, of the first byte of the start code prefix in the byte stream NAL unit that contains the NAL unit with nal_unit_type equal to 1, 5 or 20. (The start code prefix is the start_code_prefix_one_3bytes syntax element. The byte stream NAL unit syntax is defined in Annex B of the H.264/AVC specification. The current slice is the slice associated with this slice control data structure.)

The host software decoder shall not include any NAL units with values of nal_unit_type other than 1, 5, and 20 in the bitstream data buffers. However, the accelerator shall allow any such NAL units to be present and should ignore their content if present.

SliceGroupMap
The original definition specified in the DXVA AVC specification applies with the following additional constraints: When num_slice_groups_minus1 is 0, the host software decoder cannot truncate the picture parameters data buffer before this array. In this case, the host software decoder can set the corresponding array members to zero, and the accelerator shall ignore the contents of the array.

The rest of the members correspond to the SVC syntax elements or SVC variables of the same name and affect the decoding process accordingly. If the syntax element is not present in the bitstream and has no inferred value according to SVC, the host software decoder shall set the value to the default as defined in SVC. Accelerators can rely on this constraint being fulfilled.

6.5 Extensions to Restricted Mode Profiles (Section 13.0)

This section describes restricted mode profiles for SVC decoding as previously specified in Section 13.0 of the DXVA specification for ordinary H.264/AVC bitstream decoding.

The GUIDs that identify these profiles will be defined in the header file dxva.h and are included here for reference purposes.

6.5.1 DXVA_ModeH264_VLD_SVC_Scalable_Constrained_Baseline_Profile

This profile supports the features necessary for a decoder that conforms to the Scalable Constrained Baseline Profile. In this profile, the accelerator performs bitstream parsing, inverse quantization scaling, inverse transform processing, motion compensation, deblocking, inter-layer motion compensation, upsampling, and inter-layer deblocking (without film-grain synthesis).

1. Configuration parameters:
    - **bConfigBitstreamRaw** = 5 for long slice header mode or 6 for short slice header mode
    - **bConfigMBcontrolRasterOrder** = 0
    - **bConfigResidDiffHost** = 0
    - **bConfigSpatialResid8** = 0
    - **bConfigResid8Subtraction** = 0
    - **bConfigSpatialHost8or9Clipping** = 0
    - **bConfigSpatialResidInterleaved** = 0
    - **bConfigIntraResidUnsigned** = 0

- **bConfigResidDiffAccelerator** = 1
- **bConfigHostInverseScan** = 1
- **bConfigSpecificIDCT** = 2
- **bConfig4GroupedCoefs** = 0

2. Picture parameters associated with base layer coded slices (i.e., DQId = 0):
   - **frame_mbs_only_flag** = 1
   - **chroma_format_idc** = 1
   - **bit_depth_luma_minus8** = 0
   - **bit_depth_chroma_minus8** = 0
   - **weighted_pred_flag** = 0
   - **weighted_bipred_idc** = 0
   - **entropy_coding_mode_flag** = 0
   - **transform_8x8_mode_flag** = 0
   - **second_chroma_qp_index_offset** = 0
   - **redundant_pic_cnt_present_flag** = 0
   - **num_slice_groups_minus1** = 0
   - **slice_type** = 0, 2, 5 or 7
3. Picture parameters with non-base layer coded slices (DQId > 0):
   - **chroma_format_idc** = 1
   - **bit_depth_luma_minus8** = 0
   - **bit_depth_chroma_minus8** = 0
   - **frame_mbs_only_flag** = 1
   - **num_slice_groups_minus1** = 0
   - **slice_type** = 0, 2, 5 or 7
4. All data buffers shall contain only data that is consistent with the constraints specified for the Scalable Constrained Baseline Profile of the H.264/AVC specification.
5. Film-grain synthesis is not supported in this profile.

The associated GUID definition for the corresponding entry in the dxva.h header file is as follows:

```
// {9B8175D4-D670-4CF2-A9F0-FA56DF71A1AE}
DEFINE_GUID(<<name>>,
0x9b8175d4, 0xd670, 0x4cf2, 0xa9, 0xf0, 0xfa, 0x56, 0xdf, 0x71, 0xa1, 0xae);
```

6.5.2 DXVA_ModeH264_VLD_SVC_Scalable_Baseline Profile

This profile supports the features necessary for a decoder that conforms to the Scalable Baseline Profile. In this profile, the accelerator performs bitstream parsing, inverse quantization scaling, inverse transform processing, motion compensation, deblocking, inter-layer motion compensation, upsampling, and inter-layer deblocking (without film-grain synthesis).

1. Configuration parameters:
   - **bConfigBitstreamRaw** = 5 for long slice header mode or 6 for short slice header mode
   - **bConfigMBcontrolRasterOrder** = 0
   - **bConfigResidDiffHost** = 0
   - **bConfigSpatialResid8** = 0
   - **bConfigResid8Subtraction** = 0
   - **bConfigSpatialHost8or9Clipping** = 0
   - **bConfigSpatialResidInterleaved** = 0
   - **bConfigIntraResidUnsigned** = 0
   - **bConfigResidDiffAccelerator** = 1
   - **bConfigHostInverseScan** = 1
   - **bConfigSpecificIDCT** = 2
   - **bConfig4GroupedCoefs** = 0
2. Picture parameters associated with base layer coded slices (i.e., DQId = 0):
   - **frame_mbs_only_flag** = 1
   - **chroma_format_idc** = 1
   - **bit_depth_luma_minus8** = 0
   - **bit_depth_chroma_minus8** = 0
   - **weighted_pred_flag** = 0
   - **weighted_bipred_idc** = 0
   - **entropy_coding_mode_flag** = 0
   - **transform_8x8_mode_flag** = 0
   - **second_chroma_qp_index_offset** = 0
   - **num_slice_groups_minus1** = 0
   - **redundant_pic_cnt_present_flag** = 0
   - **slice_type** = 0, 2, 5 or 7
3. Picture parameters with non-base layer coded slices (DQId > 0):
   - **chroma_format_idc** = 1
   - **bit_depth_luma_minus8** = 0
   - **bit_depth_chroma_minus8** = 0
   - **frame_mbs_only_flag** = 1

- **num_slice_groups_minus1** in the range of 0 to 7 (inclusive)
- **slice_group_map_type** = 0 or 2.
- **slice_type** = 0, 1, 2, 5, 6 or 7

4. All data buffers shall contain only data that is consistent with the constraints specified for the Scalable Baseline Profile of the H.264/AVC specification.
5. Film-grain synthesis is not supported in this profile.

The associated GUID definition for the corresponding entry in the dxva.h header file is as follows:

```
// {C30700C4-E384-43E0-B982-2D89EE7F77C4}
DEFINE_GUID(DXVA_ModeH264_VLD_SVC_Scalable_Baseline,
0xc30700c4, 0xe384, 0x43e0, 0xb9, 0x82, 0x2d, 0x89, 0xee, 0x7f, 0x77, 0xc4);
```

6.5.3 DXVA_ModeH264_VLD_SVC_Scalable_Constrained_High_Profile

This profile supports the features necessary for a decoder that conforms to the Scalable Constrained High Profile. In this profile, the accelerator performs bitstream parsing, inverse quantization scaling, inverse transform processing, motion compensation, deblocking, inter-layer motion compensation, upsampling, and inter-layer deblocking (without film-grain synthesis).

1. Configuration parameters:
   - **bConfigBitstreamRaw** = 5 for long slice header mode or 6 for short slice header mode
   - **bConfigMBcontrolRasterOrder** = 0
   - **bConfigResidDiffHost** = 0
   - **bConfigSpatialResid8** = 0
   - **bConfigResid8Subtraction** = 0
   - **bConfigSpatialHost8or9Clipping** = 0
   - **bConfigSpatialResidInterleaved** = 0
   - **bConfigIntraResidUnsigned** = 0
   - **bConfigResidDiffAccelerator** = 1
   - **bConfigHostInverseScan** = 1
   - **bConfigSpecificIDCT** = 2
   - **bConfig4GroupedCoefs** = 0
2. Picture parameters associated with base layer coded slices (i.e., DQId = 0):
   - **frame_mbs_only_flag** = 1
   - **num_slice_groups_minus1** = 0
   - **redundant_pic_cnt_present_flag** = 0
   - **chroma_format_idc** = 0 or 1
   - **bit_depth_luma_minus8** = 0

- **bit_depth_chroma_minus8** = 0
- **slice_type** = 0, 2, 5 or 7

3. Picture parameters associated with non-base layer coded slices (i.e., DQId > 0):
   - **chroma_format_idc** = 1
   - **bit_depth_luma_minus8** = 0
   - **bit_depth_chroma_minus8** = 0
   - **frame_mbs_only_flag** = 1
   - **redundant_pic_cnt_present_flag** = 0
   - **num_slice_groups_minus1** = 0
   - **slice_type** = 0, 2, 5 or 7
4. All data buffers shall contain only data that is consistent with the constraints specified for the Scalable Constrained High Profile of the H.264/AVC specification.
5. Film-grain synthesis is not supported in this profile.

The associated GUID definition for the corresponding entry in the dxva.h header file is as follows:

// {8EFA5926-BD9E-4B04-8B72-8F977DC44C36}
DEFINE_GUID (DXVA_ModeH264_VLD_SVC_Restricted_Scalable_High_Progressive,
0x8efa5926, 0xbd9e, 0x4b04, 0x8b, 0x72, 0x8f, 0x97, 0x7d, 0xc4, 0x4c, 0x36);

6.5.4 DXVA_ModeH264_VLD_SVC_Scalable_High Profile

This profile supports the features necessary for a decoder that conforms to the Scalable High Profile. In this profile, the accelerator performs bitstream parsing, inverse quantization scaling, inverse transform processing, motion compensation, deblocking, inter-layer motion compensation, upsampling, and inter-layer deblocking (without film-grain synthesis).

1. Configuration parameters:
   - **bConfigBitstreamRaw** = 5 for long slice header mode or 6 for short slice header mode
   - **bConfigMBcontrolRasterOrder** = 0
   - **bConfigResidDiffHost** = 0
   - **bConfigSpatialResid8** = 0
   - **bConfigResid8Subtraction** = 0
   - **bConfigSpatialHost8or9Clipping** = 0
   - **bConfigSpatialResidInterleaved** = 0
   - **bConfigIntraResidUnsigned** = 0
   - **bConfigResidDiffAccelerator** = 1
   - **bConfigHostInverseScan** = 1
   - **bConfigSpecificIDCT** = 2

- **bConfig4GroupedCoefs** = 0

2. Picture parameters associated with base layer coded slices (i.e., DQId = 0):
   - **num_slice_groups_minus1** = 0
   - **redundant_pic_cnt_present_flag** = 0
   - **chroma_format_idc** = 0 or 1
   - **bit_depth_luma_minus8** = 0
   - **bit_depth_chroma_minus8** = 0
   - **slice_type** = 0, 1, 2, 5, 6 or 7
3. Picture parameters associated with non-base layer coded slices (i.e., DQId > 0):
   - **chroma_format_idc** = 1
   - **bit_depth_luma_minus8** = 0
   - **bit_depth_chroma_minus8** = 0
   - **redundant_pic_cnt_present_flag** = 0
   - **num_slice_groups_minus1** = 0
   - **slice_type** = 0, 1, 2, 5, 6 or 7
4. All data buffers shall contain only data that is consistent with the constraints specified for the Scalable High Profile of the H.264/AVC specification.
5. Film-grain synthesis is not supported in this profile.

The associated GUID definition for the corresponding entry in the dxva.h header file is as follows:

```
// {728012C9-66A8-422F-97E9-B5E39B51C053}
DEFINE_GUID (DXVA_ModeH264_VLD_SVC_Scalable_High,
0x728012c9, 0x66a8, 0x422f, 0x97, 0xe9, 0xb5, 0xe3, 0x9b, 0x51, 0xc0, 0x53);
```

Annex A Compatibility with H.264/AVC Decoding

Over time, the industry has released products supporting modes of DXVA operations for H.264/AVC decoding. A software decoder may use available DXVA operations of H.264/AVC decoding for the decoding of SVC bitstreams in some instances by performing a certain conversion process.

A.1 Host Conversion from H.264/AVC-based SVC to H.264/AVC

When the accelerator is capable of DXVA operation for ordinary H.264/AVC decoding but not DXVA operation for SVC decoding, the host software decoder may take the compressed data of the SVC base layer of a scalable bitstream and send it through the DXVA interface, using H.264/AVC data structures with proper parameter settings, in order to utilize the available acceleration for single-layer decoding.

In this case, the host software decoder is responsible for any necessary conversion of the SVC bitstream to the form of an ordinary H.264/AVC bitstream, and the use of proper settings of the H.264/AVC DXVA data structures. Hardware accelerator drivers capable of DXVA operation for ordinary H.264/AVC decoding may not be aware of the host software decoder conversion process.

When applicable, the host software decoder may perform the SVC-to-AVC rewriting process to take advantage of additional information in NAL units with nal_unit_type equal to 20 before sending the bitstream through the DXVA interface.

A.2 Ordinary H.264/AVC Bitstream Decoding with SVC DXVA

When the hardware accelerator driver is capable of SVC DXVA decoding, the accelerator shall be able to decode ordinary H.264/AVC bitstream under SVC DXVA with corresponding constraints.

Accelerators supporting the DXVA_ModeH264_VLD_SVC_Scalable_Baseline profile shall be capable of decoding any ordinary H.264/AVC bitstream compatible with decoders conforming to the Constrained Baseline profile.

Accelerators supporting the DXVA_ModeH264_SVC_Scalable_High profile shall be capable of decoding any ordinary H.264/AVC bitstream compatible with decoders conforming to the Constrained Baseline, Main, and High profiles.

During SVC DXVA decoding, it shall be possible to switch between the supported SVC bitstream and a supported ordinary H.264/AVC bitstream without re-creating the software decoder connection to support the specific ordinary H.264/AVC bitstream decoding.

Annex B Loss Handling (Informative)

When communicating over an unreliable channel, some NAL units may be lost during the transport. This section informatively describes some error handling procedures that may be used to mitigate packet losses in the target layer representation and in the reference layer representation. The selection of error concealment schemes is out of the scope of this specification.

When the long slice data structure is used, the host software decoder has complete information of dependency layers, reference frame lists, and reference picture lists. When packet losses involve NAL units in the target layer representation or NAL units that will be used for reference for inter-layer prediction, the host software decoder may change the target layer representation to a layer representation associated with smaller DQId, and send coded slices associated with the new reference and target layers to the accelerator. The accelerator can detect this operation via the change of DQId corresponding to coded slices with LayerType equal to 0, but it shall proceed with the decoding process without the need of extra action by the software decoder in this case.

When the short slice data structure is used or when the host software decoder prefers to have the accelerator fully control the error handling process, the host software decoder may or may not keep the target layer unchanged and send all coded slices with smaller DQId to the accelerator. The accelerator should utilize the data sent from the host software decoder for error concealment purposes as necessary.

In all cases, the accelerator shall report any errors that may hinder the decoding process to the host software decoder using the Status Report Data Structure, in the manner specified in the DXVA specification for ordinary H.264/AVC bitstream decoding.

