DOLBY

Dolby Encoding Engine

User's Guide

2.3
4 February 2019

Copyright

Dolby Laboratories, Inc.
1275 Market Street
San Francisco, CA 94103-1410 USA
Telephone 415-558-0200
Fax 415-645-4000
http://www.dolby.com

Trademarks

Confidential information

Contents

1 Introduction

This documentation describes the use of the Dolby Encoding Engine, an encoder of file-based input into Dolby audio and video formats.

- Dolby Encoding Engine
- Using this information
- Channel abbreviations
- Resources
- Patent information
- Contacting Dolby

1.1 Dolby Encoding Engine

The Dolby Encoding Engine is a stand-alone application for encoding audio and video Dolby formats from file-based input.

Dolby Encoding Engine is a modular and scalable solution that enables Dolby next-generation imaging and audio experiences through its encoding capabilities.

Dolby Encoding Engine operates with the use of a command-line interface and XML configuration files.

1.2 Using this information

This documentation contains the information required to operate the Dolby Encoding Engine.

- *Product overview* covers the supported codecs and technologies, the content format, and system requirements.
- *Installation* covers the installation and license activation of Dolby Encoding Engine.
- *Operation* covers procedures and parameters for encoding media files, batch processing, and distributed encoding.
- *Encoding parameters* defines the parameters for the supported encoding jobs.
- *Distributed encoding of Dolby Vision* covers the simultaneous Dolby Vision encoding process.
- *Optimized workflows* describes ways to optimize encoding processes.
- *Custom YUV sink for Dolby Vision profile 5 preprocessor* covers the preprocessing output in formats other than YUV and RPU.
- *Batch processing* covers multiple batch operation scenarios.
- *Multiplexing scenarios* covers the multiplexing of audio and video input.
- *Audio editing jobs* covers trimming and concatenating of audio files.
- *Reference* covers the structure of the schema and XML files, input requirements for standard dynamic range (SDR) to Dolby Vision encoding, and error codes.
- *Appendix A* includes the open-source software and third-party software declarations.

Related information

Product overview on page 9
Dolby Encoding Engine installation on page 14
Operation on page 16
Encoding parameters on page 20
Distributed encoding of Dolby Vision on page 104
Optimized workflows on page 110
Custom YUV sink for Dolby Vision profile 5 preprocessor on page 112
Batch processing on page 114
Multiplexing scenarios on page 120
Audio editing job on page 121
Reference on page 128
Appendix A: Open-source and third-party software declarations on page 134

1.3 Channel abbreviations

This table lists the channel notations used in this documentation.

Channel number	Abbreviation	Channel
1	L	Left
2	R	Right
3	C	Center
4	LFE	Low-Frequency Effects
5	Ls	Left Surround
6	Rs	Right Surround
7	Lrs	Left Rear Surround
8	Rrs	Right Rear Surround

1.4 Resources

Standards and Dolby documentation provide additional information to assist you in designing your product.

Standards

- SMPTE ST 2084:2014, *High Dynamic Range Electro-Optical Transfer Function of Mastering Reference Displays*, available from http://www.smpte.org.
- SMPTE ST 2086:2014, *Mastering Display Color Volume Metadata Supporting High-Luminance and Wide Color Gamut Images*, available from http://www.smpte.org.
- ISO/IEC 14496-10:2014, *Information Technology—Coding of Audio-Visual Objects, Part 10: Advanced Video Coding*, available from http://www.iso.org.
- ISO/IEC 23008-5:2015, *Information Technology-High-Efficiency Coding and Media Delivery in Heterogeneous Environments, Part 5: Reference Software for High-Efficiency Video Coding*, available from http://www.iso.org.

- std-iso-iec-14496-15 :2015, *Information Technology Coding of Audio-Visual Objects, Part 15: Carriage of Network Abstraction Layer (NAL) Unit Structured Video in ISO Base Media File Format*, available from http://www.iso.org.
- ITU-R BT.709, *Parameter Values for the HDTV Standards for Production and International Program Exchange*, available from http://www.itu.int.
- ITU-R BT.2020:2015, *Parameter Values for Ultra-High Definition Television Systems for Production and International Program Exchange*, available from http://www.itu.int.
- ITU-R BT.2100:2016, *Image Parameter Values for HDR Television for Use in Production and International Program Exchange*, available from http://www.itu.int/.
- CEA-861.3:2015, *HDR Static Metadata Extensions*, available at https://www.cta.tech.
- EG 432-1:2010, *SMPTE Engineering Guideline—Digital Source Processing—Color Processing for D-Cinema*
- *H.265: High-Efficiency Video Coding*: 2016, available at https://www.itu.int/rec/T-REC-H.265-201612-I/en.
- ATSC A/52:2018, *Digital Audio Compression Standard (AC-3, E-AC-3)*, available from http://www.atsc.org.
- ETSI TS 103 420, *Object-Based Audio Coding for Enhanced AC-3 (E-AC-3)*.
- EBU Tech 3364, *Audio Definition Model Metadata Specification* http://tech.ebu.ch.
- EBU Tech 3285 suppl. 6, *Specification of the Broadcast Wave Format—A Format for Audio Data Files, Supplement 6: Dolby Metadata*, available from http://tech.ebu.ch.

Dolby documentation

- *Dolby Digital Plus with Dolby Atmos Content Professional Encoder Software Integration Guide*
- *Dolby Vision Professional Encoder Software Integration Development Manual*

1.5 Patent information

This product is protected by one or more patents in the United States and elsewhere.

For more information, including a specific list of patents protecting this product, please visit http://www.dolby.com/patents.

1.6 Contacting Dolby

Support services are available to address any questions about this product.

For any questions regarding the described technology, contact dee-support@dolby.com.

If you have comments or feedback about this information set, send us an email at documentation@dolby.com.

2 Product overview

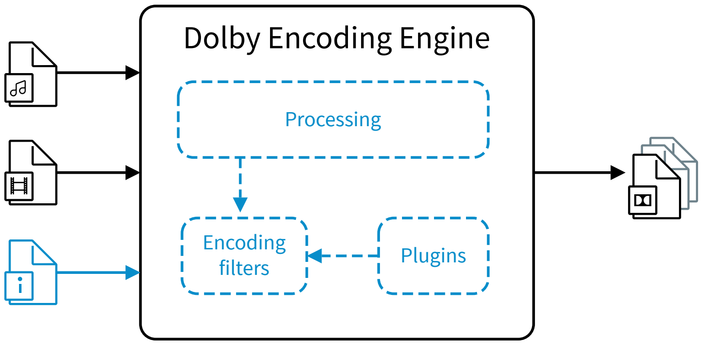
Dolby Encoding Engine is an application with integrated and extensible infrastructure designed to enable Dolby next-generation imaging and audio experiences through audio and video encoding.

- Dolby Encoding Engine functions
- Supported audio and video technologies
- Supported input and output formats
- Other processing types
- Dolby Vision plug-ins
- System requirements

2.1 Dolby Encoding Engine functions

Dolby Encoding Engine encodes file-based input into Dolby Digital Plus, Dolby Digital Plus with Dolby Atmos content, and Dolby Vision output.

Figure 1: Dolby Encoding Engine overview



The application can also multiplex Dolby Digital Plus, Dolby Digital Plus with Dolby Atmos content, and Dolby Vision content into MP4 and transport stream containers.

The application operates through CLI commands and XML configuration files that contain the encoding parameters, describe the encoding process configuration, the filters, inputs, and outputs.

Dolby Encoding Engine supports either encoding an individual asset or subjecting a batch of files to the same series of processing steps (batch processing).

You can use the Dolby Encoding Engine to encode audio, video, or both, depending on your product license. A limited time evaluation license is also available.

2.2 Supported audio and video technologies

Dolby Encoding Engine supports the following subset of audio and video technologies: Dolby Digital Plus, Dolby Digital Plus with Dolby Atmos content, and High-Efficiency Video Coding (HEVC) with Dolby Vision.

Dolby Digital Plus

Dolby Digital Plus is an advanced audio coding system based on the Dolby Digital coding system (originally developed to encode and play back multichannel digital audio). Dolby

Digital Plus introduces a number of enhancements to Dolby Digital, including support for a wider range of data rates, or increased channel count (in the case of Dolby Encoding Engine, up to 7.1 channels). Dolby Digital Plus is backward compatible with Dolby Digital.

Dolby Digital Plus enables the use of Dolby professional loudness correction and dynamic range control, both of which can be applied by Dolby Encoding Engine.

Dolby Digital Plus with Dolby Atmos content

Dolby Digital Plus with Dolby Atmos content relies on the joint object coding technology, which provides a set of algorithms to process and deliver object-based audio content for consumer entertainment applications. The technology applies spatial coding techniques to Dolby Atmos content and creates a representation of this content with a reduced bit rate.

HEVC and Dolby Vision

HEVC is an MPEG standard for video compression that improves on the H.264 (AVC) standard and allows for emission of Dolby Vision video content.

Dolby Vision is an end-to-end technology suite that enables the creation and distribution of content mastered with high dynamic range and wide color gamut.

Note: The Dolby Encoding Engine does not include an HEVC encoder plugin but requires such an encoder to be built and installed. For more information, refer to the *Dolby Encoding Engine HEVC Encoder Installation Guide*.

Related information

Other processing types on page 11

Dolby Vision plug-ins on page 12

2.3 Supported input and output formats

Dolby Encoding Engine supports the following input and output file formats.

File formats supported for Dolby Digital Plus encoding

The accepted input formats are:

- A .wav file
- A list of mono .wav files, each one containing a single channel for a combined multichannel output

The accepted output formats are:

- E-AC-3
- The E-AC-3 output can also be multiplexed into MP4 or transport stream containers directly, during the same encoding job.

File formats supported for Dolby Digital Plus with Dolby Atmos content encoding

The accepted input formats are:

- Audio Definition Model Broadcast Wave Format (ADM BWF): An industry standard mezzanine file format than can carry object-based audio and associated ADM metadata.
- Dolby Atmos master file set: A Dolby Atmos mezzanine format consisting of three files, one containing audio, one containing metadata, and one providing path names and other descriptive information.

The accepted output formats are:

- E-AC-3

- The E-AC-3 output can also be multiplexed into MP4 or transport stream containers directly, during the same encoding job.

File formats supported for Dolby Vision encoding

The accepted input formats are:

- Dolby Vision Material Exchange Format (MXF) (with Dolby Vision interleaved metadata) that adhere to:
  - P3 MXF: DCI-P3 color space as defined by EG 432-1:2010, *SMPTE Engineering Guideline—Digital Source Processing—Color Processing for D-Cinema*
  - BT.2020 MXF: Color space as defined by ITU-R BT.2020:2015, *Parameter Values for Ultra-High Definition Television Systems for Production and International Program Exchange*
- P3/BT.2020 MXF with sidecar Dolby Vision XML metadata
- JPEG 2000 frames with sidecar Dolby Vision XML metadata
- MOV with sidecar Dolby Vision XML metadata (with Apple ProRes video essence)
- Apple ProRes frames with sidecar Dolby Vision XML metadata
-

The accepted output formats are:

- HEVC
- The HEVC output can also be multiplexed into MP4 or transport stream containers directly, during the same encoding job.

Multiplexing scenarios

MP4 or transport stream containers can be the output of a single encoding job or a multiplexing job.

For a single encoding job, the input is one of the formats supported for the particular job, and the output container contains an audio-only or a video-only stream.

For a multiplexing job, the input is an audio and a video file, and therefore the output container (MP4 or transport stream) also contains both audio and video streams. In this case, the input is an HEVC video file and an E-AC-3 audio file.

Related information

Multiplexing scenarios on page 120

2.4 Other processing types

As part of the encoding jobs, Dolby Encoding Engine can provide additional signal processing.

Dolby professional loudness correction

Ensures balanced audio loudness levels in a dual-pass loudness correction. The first pass measures the audio level, and the second pass corrects it.
The first stage is called Dolby professional loudness metering, and it combines the use of standards-based loudness estimation algorithms with the Dialogue Intelligence technology.
The second stage is correction based on the measured loudness levels. Content in a Dolby metadata-aware format, such as Dolby Digital Plus, has the loudness metadata adjusted so that it correctly matches the associated audio content.
Depending on the encoding job, it may be possible to choose whether to measure and correct, or simply to measure and author metadata for downstream correction.

For Dolby Digital Plus (7.1) and Dolby Digital Plus with Dolby Atmos content encoding, correction is not supported and metering is the only option.

Dynamic range control

Compresses the dynamic range of a program by reducing (cutting) the level of the loudest sounds and increasing (boosting) the level of the quietest sounds to adjust the sound to the listening environment (for example, home theater or headphones).

Related information

Supported audio and video technologies on page 9

2.5 Dolby Vision plug-ins

A standard installation of Dolby Encoding Engine includes default plug-in implementations for J2K decoding, scaling, and noise reduction. The Dolby Encoding Engine allows you to replace these components with others that perform the same processing functions.

Note: The Dolby Encoding Engine does not include an HEVC encoder plugin but requires such an encoder to be built and installed. For more information, refer to the *Dolby Encoding Engine HEVC Encoder Installation Guide*.

The parameters of the plug-ins are not described in this documentation, because each plug-in can support a different set of parameters, with different parameter values and parameter names.

For all current information about Dolby Vision plug-ins, refer to https://github.com/DolbyLaboratories/dolby-encoding-engine. The repository contains example plug-ins as well as information on how to build new plug-ins. You must place the plug-in in the Dolby Encoding Engine structure. The location for the plug-ins is the Dolby Encoding Engine application folder or another location defined by the DEE_PLUGIN_PATH environment variable.

Related information

Supported audio and video technologies on page 9

2.6 System requirements

There are recommended and minimum hardware specifications, as well as supported operation systems, to ensure the operation of the Dolby Encoding Engine.

Operation systems

- Ubuntu 16.04 (64 bit)
- CentOS 7 (64 bit)
- Windows 10 (64 bit)

Additional software requirements

Windows computers require:

- Visual Studio 2010 or Visual C++ Redistributable Packages for Visual Studio 2010 (website)
- Visual Studio 2013 or Visual C++ Redistributable Packages for Visual Studio 2013 (website)
- Visual Studio 2015 or Visual C++ Redistributable Packages for Visual Studio 2015 (website)

Recommended hardware specifications

- CPU: Intel Xeon processor E5-1620 v4, 3.50 GHz
- RAM: 32 GB

Minimum hardware specifications

- CPU: any x86_64 processor supporting AVX2 and FMA3 instruction set
- RAM: 16 GB

3 Dolby Encoding Engine installation

Install the Dolby Encoding Engine using the provided installer package and the license file.

- Installing Dolby Encoding Engine on Linux
- Installing Dolby Encoding Engine on Windows
- Adding product license
- Installing the xmllint component

3.1 Installing Dolby Encoding Engine on Linux

Unpack the Dolby Encoding Engine and launch the installer from the CLI with the specified installation parameters.

Procedure

1. Optional: If you are upgrading to a newer version of the Dolby Encoding Engine, uninstall and remove all components of the previous version (including the plug-ins and other libraries).
2. Unpack the .zip file.
3. Add the permission to execute the file.
4. Optional: To display the available installer options, enter: `dolby_encoding_engine_install.bin -h`
5. To launch the installer, enter: `dolby_encoding_engine_install.bin -d` *`<folder_path>`*

   In this entry:

   `<folder_path>` is the installation folder.
6. Follow the instructions on the screen.

Results

The Dolby Encoding Engine installs on your computer.

3.2 Installing Dolby Encoding Engine on Windows

Unpack the Dolby Encoding Engine, and launch the installer file.

Procedure

1. Optional: If you are upgrading to a newer version of the Dolby Encoding Engine, uninstall and remove all components of the previous version (including the plug-ins and other libraries).
2. Unpack the .zip file.
3. Launch the `dolby_encoding_engine_install`.exe file.
4. Follow the instructions on the screen.

Results

The Dolby Encoding Engine installs on your computer.

3.3 Adding product license

To use the Dolby Encoding Engine, add your product license to the folder specified in this procedure.

About this task

We recommend placing the Dolby Encoding Engine product license in the application folder or the plug-ins folder (specified by the DEE_PLUGIN_PATH environment variable).

Placing the license file in a different location requires you to specify the license file path in the CLI for each operation, using the `--license-file` option.

Procedure

Place the license.lic file in the application folder or the plug-ins folder.

3.4 Installing the xmllint component

The xmllint Linux component is necessary for XML schema validation.

Prerequisites

Install the xmllint component, if it is not already installed on your Linux system. Schema validation is supported only on Linux systems.

Procedure

To install the xmllint component:

- For Ubuntu, in the command line, enter: `sudo apt-get install libxml2-utils`
- For CentOS 7, in the command line, enter: `sudo yum install libxml2`

Results

The xmllint component installs on your system.

4 Operation

The Dolby Encoding Engine operates with the use of a CLI and XML configuration files.

- Executing an encoding job
- Supported CLI options
- Command examples

For each encoding job, you need to execute a CLI command in which you specify the location of an XML configuration file. In the XML configuration file, you need to include all the encoding job parameters (unless you want to keep the default values).

To create the XML file you can use the following information:

- The encoding parameters chapter
- The XML configuration file reference section
- The XML template files provided with your Dolby Encoding Engine (in the xml_templates subfolder)
- The XSD file that you can generate from the Dolby Encoding Engine

4.1 Executing an encoding job

To launch an encoding job, enter a CLI command pointing at your XML configuration file.

Prerequisites

You need an XML configuration file that you can point at when you launch an encoding job in the CLI. The XML configuration file contains the parameters describing the encoding process.

You can use one of the XML templates provided with Dolby Encoding Engine in the xml_templates subfolder.

Procedure

In the CLI, enter: `dee -x` *`path/file_name option`*

In this entry:

path/file_name is the path and name of the XML configuration file.

option can be an optional command parameter.

Related information

Encoding parameters on page 20

XML configuration file on page 128

Command examples on page 19

4.2 Supported CLI options

You can modify the encoding job command, for example to monitor job progress, or to overwrite an XML parameter through CLI.

Some options have a long and short version, where indicated.

The option to execute an encoding job

--xml, -x *file_name*

Use this option to execute a Dolby Encoding Engine encoding job and specify the location of the XML configuration file according to the path and file name defined by *file_name*.

Additional options

--help, -h

Displays the list of supported CLI options.

--schema, -s *file_name*

Creates an XML schema (XSD) file with the path and file name defined by *file_name*.

--input-audio, -a *file_name*

Specifies the location of the input audio file with the path and file name defined by *file_name*; overwrites the parameter in the XML configuration file.

--input-video, -v *file_name*

Specifies the location of the input video file with the path and file name defined by *file_name*; overwrites the parameter in the XML configuration file.

--output, -o *file_name*

Specifies where to write the output file with path and file name defined by *file_name*; overwrites the parameter in the XML configuration file.

--log-file *file_name*

Specifies where to write the log file: path and file name defined by *file_name*. If not specified, the log displays as standard output.

--verbose *level*

Specifies verbosity level, where the *level* values are:

- `quiet` - does not output any messages.
- `normal` - outputs errors and warnings.
- `info` - outputs errors, warnings, and info messages.
- `debug` (default) - outputs all messages.

--license-file, -l *file_name*

Specifies the location of the license file with the path and file name defined by *file_name*; use only if the license file is not in the Dolby Encoding Engine application folder or the folder defined by the DEE_PLUGIN_PATH environment variable.

--trace-file *file_name*

Enables trace and specifies the trace record file with the path and file name defined by *file_name*. Refer to the “The trace feature” section.

--trace-length *number*

Specifies the number of trace frames (records) to keep in a buffer (default = 5,000). Refer to the “The trace feature” section.

--progress

Enables progress reporting.

--progress-interval *milliseconds*

Sets the progress reporting interval, in milliseconds (default = 1,000).

--diagnostics-interval *milliseconds*

Sets the diagnostics (for example MEM/CPU usage) reporting interval, in milliseconds (default = 1,000).

--disable-xml-validation

Disables XML configuration file validation.

--add-elem

Adds elements (parameters) to the configuration defined by the XML file; If a parameter already exists in the XML file, this function overwrites it. Refer to the “The --add-elem CLI option” section.

--no-links

Disables the support for non-ASCII characters in paths and filenames on Windows. Refer to the “ASCII characters and the --no-links CLI option” section.

Related information

The --add-elem CLI option on page 18

The trace feature on page 19

ASCII characters and the --no-links CLI option on page 19

4.2.1 The --add-elem CLI option

The option adds elements (parameters) to the configuration defined by the XML file; If a parameter already exists in the XML file, this function overwrites it.

The syntax is:

```
--add-elem param=value
```

where:

param is the configuration parameter (XML element name).

value is the parameter value.

The `--add-elem` option does not create a new version of the XML file, but it is based on the actual structure of the XML configuration file used in a particular encoding job. You must define the nesting of the added element if you want to add a new parameter or overwrite an existing parameter that appears more than once in the XML file:

```
--add-elem parent1:parent2:parent3:param=value
```

Omit the `<job_config>` root element in the nesting.

You can specify more than one value per parameter. To do that, use a space to separate the values and wrap the list in quotation marks:

```
--add-elem param="value1 value2 value3"
```

You can use the `--add-elem` option more than once for a single encoding job.

Related information

Supported CLI options on page 16

4.2.2 The trace feature

The trace feature allows you to record details of critical issues (for example, a crash or hang), which is especially useful in cases when you cannot provide Dolby with the means to reproduce the issues (for example, because of copyrights or a specific environment).

By default, trace is disabled.

Trace keeps only the most recent records, specified by the `--trace-length` option. The longer the trace record is, the better debugging information it provides. The recommended length of 5,000 frames is default. You can set the value to zero to generate unlimited trace records.

Beside the main trace file specified by the `--trace-file` option, there are two temporary files with the same name and the following extensions: .tmpa and .tmpb. Normally, when the Dolby Encoding Engine exits, it moves the contents of the temporary files to the main trace file and then deletes all the temporary files. However, there are critical errors that might prevent the Dolby Encoding Engine from writing the main trace file. In such a case, you can deliver the temporary files to Dolby for analysis.

Related information

Supported CLI options on page 16

4.2.3 ASCII characters and the --no-links CLI option

The `--no-links` option disables the support for non-ASCII characters in paths and filenames on Windows.

The Windows version of the Dolby Encoding Engine does not support non-ASCII characters internally in filenames or paths. However, the Dolby Encoding Engine can use a mechanism that allows for non-ASCII characters in paths and filenames. This mechanism creates a link between the non-ASCII path or filename and an internal equivalent with ASCII characters only. To disable this mechanism, use the `--no-links` option.

The link is symbollic when Dolby Encoding Engine runs with administrator rights. Otherwise, it is a hard link. The hard link has to refer to the same NTFS volume as the linked input/output file.

Related information

Supported CLI options on page 16

4.3 Command examples

There are several ways to use the additional command-line interface options.

- To create a file with the XML schema in the home folder, enter a command similar to this one:

  ```
  dee -s ~/schema.xsd
  ```

- To launch an encoding job with progress reporting, enter a command similar to this one:

  ```
  dee --progress -x ../mxf_dv_mezz_to_dv_profile_5_hevc.xml
  ```

- To launch an encoding job with input specified in the command-line interface and overwrite the input specified in the XML configuration file (if it is specified there), enter a command similar to this one:

  ```
  dee --xml ../mxf_dv_mezz_to_dv_profile_5_hevc.xml --input-video ../Video/Movie/movie.mxf
  ```

5 Encoding parameters

The Dolby Encoding Engine supports several types of encoding jobs.

- Dolby Digital Plus (1.0/2.0/5.1) encoding job
- Dolby Digital Plus (7.1) encoding job
- Dolby Atmos encoding job
- Dolby Atmos encoding job (from channel-based immersive input)
- Dolby Vision profile 5 encoding job
- Dolby Vision profile 8.1 encoding job
- HDR10 video encoding job
- SDR to Dolby Vision profile 4 encoding job
- SDR to Dolby Vision profile 8 encoding job
- SDR encoding job
- HEVC transcoding job

In each case, you must define parameters for three processing stages: input, filter, and output.

5.1 Dolby Digital Plus (1.0/2.0/5.1) encoding job

The Dolby Digital Plus (1.0/2.0/5.1) encoder produces Dolby Digital Plus audio streams, according to the parameters you specify in the job configuration XML file.

The input for this encoding job is in .wav format, and the output is in E-AC-3 format. The output can also be multiplexed into MP4 or transport stream containers.

The process for encoding Dolby Digital Plus with 1.0, 2.0, and 5.1 channel configurations is different than for the 7.1 configuration and therefore separate filters are defined for them. The difference between the two processes is loudness correction, which is not supported for 7.1.

Related information

XML example files for Dolby Digital Plus (1.0/2.0/5.1) on page 20
Input parameters for Dolby Digital Plus (1.0/2.0/5.1) on page 21
Filter parameters for Dolby Digital Plus (1.0/2.0/5.1) on page 22
Output parameters for Dolby Digital Plus (1.0/2.0/5.1) on page 25
Encoding parameters on page 20

5.1.1 XML example files for Dolby Digital Plus (1.0/2.0/5.1)

The XML template files are examples of a job configuration.

You can use them to create job configurations. You can find them in the xml_templates folder. For the Dolby Digital Plus (1.0/2.0/5.1) encoding job, refer to the following files:

- wav_pcm_to_ddp_ec3.xml
- wav_pcm_to_ddp_mp4.xml
- wav_list_pcm_to_ddp_ec3.xml
- wav_list_pcm_to_ddp_mp4.xml

1. wav_pcm_to_ddp_ec3.xml
2. wav_pcm_to_ddp_mp4.xml
3. wav_list_pcm_to_ddp_ec3.xml
4. wav_list_pcm_to_ddp_mp4.xml

Related information

Dolby Digital Plus (1.0/2.0/5.1) encoding job on page 20

5.1.2 Input parameters for Dolby Digital Plus (1.0/2.0/5.1)

This section contains a table with all the supported Dolby Digital Plus (1.0/2.0/5.1) input parameters.

The supported channel configurations of the input files are:

- One channel.
- Two channels (which can also be downmixed to mono).
- Six channels (which can also be downmixed to stereo or mono).
- Eight channels. (Downmix to mono, stereo, or 5.1 is required in this case.)

Some of the channel configurations can impact the parameters of the filter processing stage (for example, the `<downmix_config>` parameter).

The input can be either a single file or a list of mono files, in which case each file from the list contains a single channel.

Input parameters for a single file

Table 1: Input parameters for a single file

Parameter	Type	Value	Description
`<wav>`	XML element		Parent element for a single file with mono, stereo, or multichannel PCM.
`<file_name>`	String	*filename*	
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored input.
`<path>`	String	*folder_path*	Path to the directory that contains the input file (for example, `<path>path/to/media</path>`).

Input parameters for a file list

Table 2: Input parameters for a file list

Parameter	Type	Value	Description
`<wav_list>`	XML element		Parent element for a list of mono .wav files, each containing a single channel.
`<file_name_L>`	String	*filename*	Left channel input file name.
`<file_name_R>`	String	*filename*	Right channel input file name.
`<file_name_C>`	String	*filename*	Center channel input file name.
`<file_name_LFE>`	String	*filename*	Low-frequency effects channel input file name.

Table 2: Input parameters for a file list (continued)

Parameter	Type	Value	Description
<file_name_LS>	String	*filename*	Left Surround channel input file name.
<file_name_RS>	String	*filename*	Right Surround channel input file name.
<file_name_LRS>	String	*filename*	Left Rear Surround channel input file name.
<file_name_RRS>	String	*filename*	Right Rear Surround channel input file name.
<channel_configuration>	String	mono, stereo, 5.1, 7.1 Default: 5.1	Defines channel configuration that the input file set represents.
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored input.
<path>	String	*folder_path*	Path to the directory that contains the input files (for example, <path>path/to/media</path>).

Related information

Dolby Digital Plus (1.0/2.0/5.1) encoding job on page 20

5.1.3 Filter parameters for Dolby Digital Plus (1.0/2.0/5.1)

When creating an XML file, you can use the example of the filter configuration within an XML file and a table that includes all of the supported Dolby Digital Plus (1.0/2.0/5.1) filter parameters with the available and default values.

Table 3: Dolby Digital Plus (1.0/2.0/5.1) filter parameters

Parameter	Type	Value	Decription
<pcm_to_ddp>	XML element		Parent element for the Dolby Digital Plus (1.0/2.0/5.1) filter parameters.
<loudness> *	XML element		Parent element for loudness measurement parameters.
<measure_correct_preset>	XML element		Parent element for loudness measurement and correction parameters.
<preset>	String	atsc_a85_agile, ebu_r128, freetv_op59, arib_b32 Default: atsc_a85_agile	Loudness measurement and correction according to one of the preset standards.

Table 3: Dolby Digital Plus (1.0/2.0/5.1) filter parameters (continued)

Parameter	Type	Value	Decription
`<measure_only>`	XML element		Parent element for loudness measurement parameters (without loudness correction).
`<metering_mode>`	String	1770-1, 1770-2, 1770-3, LeqA Default: 1770-3	Loudness measuring mode according to one of the broadcast standards.
`<dialogue_intelligence>`	Boolean	Default: true	Identifies and analyzes dialogue segments as a basis for speech gating.
`<speech_threshold>`	Integer	0–100 Default: 20	If the percentage of speech is higher than the threshold, the encoder uses speech gating to set the dialogue normalization value (otherwise, level gating).
`<downmix_config>`	String	off, mono, stereo, 5.1 Default: off	Enables downmix.
`<data_rate>`	Integer	32, 40, 48, 56, 64, 80, 96, 104, 112, 120, 128, 144, 160, 176, 192, 200, 208, 216, 224, 232, 240, 248, 256, 272, 288, 304, 320, 336, 352, 368, 384, 400, 448, 512, 576, 640, 704, 768, 832, 896, 960, 1008, 1024, For mono: 32 (minimum), 1024 Default: 64 For stereo: 96 (minimum), 1024 Default: 128 For 5.1: 192 (minimum), 1024 Default: 192	Specifies the bit rate.
`<timecode_frame_rate>`	String	not_indicated, 23.976, 24, 25, 29.97, 30, 48, 50, 59.94, 60 Default: not_indicated	Frame rate associated with the specified timecode.
`<start>`	String	*HH:MM:SS:FF* or *HH:MM:SS.xx* (timecode) or first_frame_of_action Default: first_frame_of_action	Timecode (*HH:MM:SS:FF* or *HH:MM:SS.xx*) indicating starting point. Add df for drop frame (for example, *HH:MM:SS:FF*df).

Table 3: Dolby Digital Plus (1.0/2.0/5.1) filter parameters (continued)

Parameter	Type	Value	Decription
<end>	String	*HH:MM:SS:FF* or *HH:MM:SS.xx* or end_of_file Default: end_of_file	Timecode (*HH:MM:SS:FF* or *HH:MM:SS.xx*) indicating ending point. Add df for drop frame, e.g. *HH:MM:SS:FF*df.
<prepend_silence_duration>	String	*seconds.miliseconds* Default: 0	Decimal number representing the duration of silence prepended to output.
<lfe_on>	Boolean	Default: true	Turns on Low-Frequency Effects (LFE). Affects 5.1 only.
<dolby_surround_mode>	String	not_indicated, yes, no Default: not_indicated	Specifies the method a channel-based decoder should use for playback. Affects stereo only.
<dolby_surround_ex_mode>	String	not_indicated, yes, no Default: no	Affects 5.1 only.
<drc>	XML element		Parent element for dynamic range control parameters
<line_mode_drc_profile>	String	film_standard, film_light, music_standard, music_light, speech Default: film_standard	Dynamic range control profile
<rf_mode_drc_profile>	String	film_standard, film_light, music_standard, music_light, speech Default: film_standard	Dynamic range control profile.
<lfe_lowpass_filter>	Boolean	true, false Default: true	
<surround_90_degree_phase_shift>	Boolean	true, false Default: true	Affects 5.1 only.
<surround_3db_attenuation>	Boolean	true, false Default: true	Affects 5.1 only.
<downmix>	XML element		Parent element for downmix parameters
<loro_center_mix_level>	String	+3, +1.5, 0, -1.5, -3, -4.5, -6, -inf Default: -3	Affects 5.1 only.
<loro_surround_mix_level>	String	-1.5, -3, -4.5, -6, -inf Default: -3	Affects 5.1 only.
<ltrt_center_mix_level>	String	+3, +1.5, 0, -1.5, -3, -4.5, -6, -inf Default: -3	Affects 5.1 only.

Table 3: Dolby Digital Plus (1.0/2.0/5.1) filter parameters (continued)

Parameter	Type	Value	Decription
`<ltrt_surround_mix_level>`	String	`-1.5`, `-3`, `-4.5`, `-6`, `-inf` Default: `-3`	Affects 5.1 only.
`<preferred_downmix_mode>`	String	not_indicated, loro, `ltrt`, `ltrt-pl2` Default: not_indicated	Affects 5.1 only.

[*] For loudness management in the Dolby Digital Plus (1.0/2.0/5.1) encoding process, choose either loudness metering and correction `<measure_correct_preset>`, or metering only `<metering_mode>`.

Related information

Dolby Digital Plus (1.0/2.0/5.1) encoding job on page 20

5.1.4 Output parameters for Dolby Digital Plus (1.0/2.0/5.1)

When creating an XML file, you can use the examples of the output configuration within an XML file and tables that include all the supported Dolby Digital Plus (1.0/2.0/5.1) output parameters with the available and default values.

Output parameters for E-AC-3

Table 4: E-AC-3 output parameters

Parameter	Type	Value	Description
`<ec3>`	XML element		Parent element for an E-AC-3 output file.
`<file_name>`	String	*`filename`*	Output file name.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored output.
`<path>`	String	*`Storage folder path`*	Path to the directory to save the output file (for example, `<path>path/to/media</path>`).

Output parameters for MP4

Table 5: MP4 output parameters

Parameter	Type	Value	Description
`<mp4>`	XML element		Parent element for an MP4 output file.
`<output_format>`	String	mp4, dash	Defines whether the output is an MP4 or DASH file.
`<min_frag_duration>`	Integer		Minimum fragment duration, in milliseconds.
`<max_frag_duration>`	Integer		Maximum fragment duration, in milliseconds.
`<mux_flag>`	String	encrypt_style_piff, write_iods, write_pdin, write_bloc, write_ainf, write_ctts_v1, write_subs_v1	Sets specified multiplexer flag.

Table 5: MP4 output parameters (continued)

Parameter	Type	Value	Description
<frag_flag>	String	frag_style_default, frag_style_cff, write_tfdt, write_sdtp, write_trik, write_sidx, write_mfra, force_tfra, no_base_data_offset, empty_trex, empty_tfhd, one_tfra_per_traf, default_base_is_moof	Sets specified fragmentation flag.
<track_flag>	String	track_enabled, track_in_movie, track_in_preview, track_in_poster	Sets specified track flag.
<file_name>	String	*filename*	Output file name.
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored output.
<path>	String	*Storage folder path*	Path to the directory to save the output file (for example, <path>path/to/media</path>).

Output parameters for transport stream

Table 6: Transport stream output parameters

Parameter	Type	Value	Description
<ts>	XML element		Parent element for a transport stream output file.
<packetize_standard>	String	mpeg, atsc, dvb Default: mpeg	Indicates the packetize standard used for substreams.
<packetize_mode>	String	cbr, vbr Default: vbr	Indicates constant bit rate (CBR) or variable bit rate (VBR) of the substreams.
<transport_stream_id>	Integer	Default: 0	Transport stream ID.
<audio_pid>	Integer	32–8186 Default: 40	Packet ID of the audio stream.
<video_pid>	Integer	32–8186 Default: 50	Packet ID of the video stream.
<file_name>	String	*filename*	Output file name.
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored output.

Table 6: Transport stream output parameters (continued)

Parameter	Type	Value	Description
`<path>`	String	*Storage folder path*	Path to the directory to save the output file (for example, `<path>path/to/media</path>`).
`<plugin>`	XML element	Depends on the selected plug-in. Example: `<base>`	Allows you to choose the transport stream multiplexing plug-in. The base implementation is delivered with Dolby Encoding Engine.

Related information

Dolby Digital Plus (1.0/2.0/5.1) encoding job on page 20

5.2 Dolby Digital Plus (7.1) encoding job

The Dolby Digital Plus (7.1) encoder produces Dolby Digital Plus audio streams, according to the parameters you specify in the job configuration XML file.

The input for this encoding job is in .wav format, and the output is in E-AC-3 format. The output can also be multiplexed into MP4 or transport stream containers.

The process for encoding Dolby Digital Plus with 7.1 channel configuration is different than for the 1.0, 2.0, and 5.1 configurations; therefore, separate filters are defined for them. The difference between the two processes is loudness correction, which is not supported for 7.1.

Related information

XML example files for Dolby Digital Plus (7.1) on page 27
Input parameters for Dolby Digital Plus (7.1) on page 27
Filter parameters for Dolby Digital Plus (7.1) on page 29
Output parameters for Dolby Digital Plus (7.1) on page 31
Encoding parameters on page 20

5.2.1 XML example files for Dolby Digital Plus (7.1)

The XML template files are examples of a job configuration.

You can use them to create job configurations. You can find them in the xml_templates folder. For the Dolby Digital Plus (7.1) encoding job, refer to the following files:

- wav_pcm_to_ddp_7_1_ec3.xml
- wav_pcm_to_ddp_7_1_mp4.xml
- wav_list_pcm_to_ddp_7_1_ec3.xml
- wav_list_pcm_to_ddp_7_1_mp4.xml

Related information

Dolby Digital Plus (7.1) encoding job on page 27

5.2.2 Input parameters for Dolby Digital Plus (7.1)

This section contains a table with all the supported Dolby Digital Plus (7.1) input parameters.

The supported channel configurations of the input files are:

- Eight channels
- Six channels (which are upmixed to eight channels and encoded as 7.1)

Some of the channel configurations can impact the parameters of the filter processing stage, for example the `<downmix_config>` parameter.

The input can be either a single file or a list of mono files, in which case each file from the list contains a single channel.

Input parameters for a single file

Table 7: Input parameters for a single file

Parameter	Type	Value	Description
`<wav>`	XML element		Parent element for a single file with mono, stereo, or multichannel PCM.
`<file_name>`	String	*filename*	
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored input.
`<path>`	String	*folder_path*	Path to the directory that contains the input file (for example, `<path>path/to/media</path>`).

Input parameters for a file list

Table 8: Input parameters for a file list

Parameter	Type	Value	Description
`<wav_list>`	XML element		Parent element for a list of mono .wav files, each containing a single channel.
`<file_name_L>`	String	*filename*	Left channel input file name.
`<file_name_R>`	String	*filename*	Right channel input file name.
`<file_name_C>`	String	*filename*	Center channel input file name.
`<file_name_LFE>`	String	*filename*	Low-frequency effects channel input file name.
`<file_name_LS>`	String	*filename*	Left Surround channel input file name.
`<file_name_RS>`	String	*filename*	Right Surround channel input file name.
`<file_name_LRS>`	String	*filename*	Left Rear Surround channel input file name.
`<file_name_RRS>`	String	*filename*	Right Rear Surround channel input file name.
`<channel_configuration>`	String	`5.1`, `7.1` Default: `5.1`	Defines channel configuration that the input file set represents. See the supported channel configurations information above.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored input.
`<path>`	String	*folder_path*	Path to the directory that contains the input files (for example:, `<path>path/to/media</path>`).

Related information

Dolby Digital Plus (7.1) encoding job on page 27

5.2.3 Filter parameters for Dolby Digital Plus (7.1)

When creating an XML file, you can use the example of the filter configuration within an XML file and a table that includes all the supported Dolby Digital Plus (7.1) filter parameters with the available and default values.

Table 9: Dolby Digital Plus (7.1) filter parameters

Parameter	Type	Value	Description
`<pcm_to_ddp_7_1>`	XML element		Parent element for the Dolby Digital Plus (7.1) filter parameters.
`<loudness>`*	XML element		Parent element for loudness measurement parameters.
`<metering_mode>`	String	`1770-1`, `1770-2`, `1770-3`, `LeqA` Default: `1770-3`	Loudness measurement mode according to one of the broadcast standards.
`<dialogue_intelligence>`	Boolean	`true`, `false` Default: `true`	Identifies and analyzes dialogue segments as a basis for speech gating.
`<speech_threshold>`	Integer	0–100 Default: 20	If the percentage of speech is higher than the threshold, encoder uses speech gating to set the dialogue normalization value. Otherwise, it uses level gating).
`<data_rate>`	Integer	`384`, `448`, `504`, `576`, `640`, `704`, `768`, `832`, `896`, `960`, `1008`, `1024` Default: `384`	Specifies the target bit rate.
`<timecode_frame_rate>`	String	`not_indicated`, `23.976`, `24`, `25`, `29.97`, `30`, `48`, `50`, `59.94`, `60` Default: `not_indicated`	Frame rate associated with the specified timecode.
`<start>`	String	*`HH:MM:SS:FF`* or *`HH:MM:SS.xx`* (timecode) or `first_frame_of_action` Default: `first_frame_of_action`	Timecode (*`HH:MM:SS:FF`* or *`HH:MM:SS.xx`*) indicating starting point. Add df for drop-frame, e.g. *`HH:MM:SS:FF`*df.

Table 9: Dolby Digital Plus (7.1) filter parameters (continued)

Parameter	Type	Value	Description
<end>	String	*HH:MM:SS:FF* or *HH:MM:SS.xx* or end_of_file Default: end_of_file	Timecode (*HH:MM:SS:FF* or *HH:MM:SS.xx*) indicating ending point. Add df for drop-frame, e.g. *HH:MM:SS:FF*df.
<prepend_silence_duration>	String	*seconds.miliseconds* Default: 0	Decimal number representing the duration of silence prepended to output.
<lfe_on>	Boolean	true, false Default: true	Turns on LFE.
<drc>	XML element		Parent element for dynamic range control parameters
<line_mode_drc_profile>	String	film_standard, film_light, music_standard, music_light, speech Default: film_standard	Dynamic range control profile for line mode.
<rf_mode_drc_profile>	String	film_standard, film_light, music_standard, music_light, speech Default: film_standard	Dynamic range control profile for RF mode.
<lfe_lowpass_filter>	Boolean	true, false Default: true	Enables LFE lowpass filter.
<surround_90_degree_phase_shift>	Boolean	true, false Default: true	Enables 90-degree phase shift to surround channels.
<surround_3db_attenuation>	Boolean	true, false Default: true	Enables 3 dB attenuation to surround channels.
<downmix>	XML element		Parent element for downmix parameters.
<loro_center_mix_level>	String	+3, +1.5, 0, -1.5, -3, -4.5, -6, -inf Default: -3	Lo/Ro center downmix level.
<loro_surround_mix_level>	String	-1.5, -3, -4.5, -6, -inf Default: -3	Lo/Ro surround downmix level.
<ltrt_center_mix_level>	String	+3, +1.5, 0, -1.5, -3, -4.5, -6, -inf Default: -3	Lt/Rt center downmix level.
<ltrt_surround_mix_level>	String	-1.5, -3, -4.5, -6, -inf Default: -3	Lt/Rt surround downmix level.
<preferred_downmix_mode>	String	not_indicated, loro, ltrt, ltrt-pl2 Default: not_indicated	Stereo downmix preference.

[*] Dolby Digital Plus (7.1) encoding supports loudness measurement but not loudness correction.

Related information

Dolby Digital Plus (7.1) encoding job on page 27

5.2.4 Output parameters for Dolby Digital Plus (7.1)

When creating an XML file, you can use the examples of the output configuration within an XML file and tables that include all the supported Dolby Digital Plus (7.1) output parameters with the available and default values.

Output parameters for E-AC-3

Table 10: Output parameters for E-AC-3

Parameter	Type	Value	Description
<ec3>	XML element		Parent element for an E-AC-3 output file.
<file_name>	String	*filename*	Output file name.
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored output.
<path>	String	*Storage folder path*	Path to the directory to save the output file (for example, <path>path/to/media</path>).

Output parameters for MP4

Table 11: MP4 output parameters

Parameter	Type	Value	Description
<mp4>	XML element		Parent element for an MP4 output file.
<output_format>	String	mp4, dash	Defines whether the output is an MP4 or DASH file.
<min_frag_duration>	Integer		Minimum fragment duration, in milliseconds.
<max_frag_duration>	Integer		Maximum fragment duration, in milliseconds.
<mux_flag>	String	encrypt_style_piff, write_iods, write_pdin, write_bloc, write_ainf, write_ctts_v1, write_subs_v1	Sets specified multiplexer flag.
<frag_flag>	String	frag_style_default, frag_style_cff, write_tfdt, write_sdtp, write_trik, write_sidx, write_mfra, force_tfra, no_base_data_offset, empty_trex, empty_tfhd, one_tfra_per_traf, default_base_is_moof	Sets specified fragmentation flag.
<track_flag>	String	track_enabled, track_in_movie, track_in_preview, track_in_poster	Sets specified track flag.

Table 11: MP4 output parameters (continued)

Parameter	Type	Value	Description
<file_name>	String	*filename*	Output file name.
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored output.
<path>	String	*Storage folder path*	Path to the directory to save the output file (for example, <path>path/to/media</path>).

Output parameters for transport stream

Table 12: Transport stream output parameters

Parameter	Type	Value	Description
<ts>	XML element		Parent element for a transport stream output file.
<packetize_standard>	String	mpeg, atsc, dvb Default: mpeg	Indicates the packetize standard used for substreams.
<packetize_mode>	String	cbr, vbr Default: vbr	Indicates constant bit rate (CBR) or variable bit rate (VBR) of the substreams.
<transport_stream_id>	Integer	Default: 0	Transport stream ID.
<audio_pid>	Integer	32–8186 Default: 40	Packet ID of the audio stream.
<video_pid>	Integer	32–8186 Default: 50	Packet ID of the video stream.
<file_name>	String	*filename*	Output file name.
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored output.
<path>	String	*Storage folder path*	Path to the directory to save the output file (for example, <path>path/to/media</path>).
<plugin>	XML element	Depends on the selected plug-in. Example: <base>	Allows you to choose the transport stream multiplexing plug-in. The base implementation is delivered with Dolby Encoding Engine.

Related information

Dolby Digital Plus (7.1) encoding job on page 27

5.3 Dolby Atmos encoding job

The Dolby Atmos encoder produces Dolby Digital Plus with Dolby Atmos content audio streams, according to the parameters you specify in the job configuration XML file.

The input for this encoding job is a Dolby Atmos master file set or ADM BWF.

The output is in E-AC-3 format. The output can also be multiplexed into an MP4 or transport stream containers.

Related information

XML example files for Dolby Atmos on page 33
Input parameters for Dolby Atmos on page 33
Filter parameters for Dolby Atmos on page 34
Output parameters for Dolby Atmos on page 35
Encoding parameters on page 20

5.3.1 XML example files for Dolby Atmos

The XML template files are examples of a job configuration.

You can use them to create job configurations. You can find them in the xml_templates folder. For the Dolby Atmos encoding job, refer to the following files:

- adm_atmos_mezz_to_atmos_ddp_ec3.xml
- adm_atmos_mezz_to_atmos_ddp_mp4.xml
- damf_atmos_mezz_to_atmos_ddp_ec3.xml
- damf_atmos_mezz_to_atmos_ddp_mp4.xml

Related information

Dolby Atmos encoding job on page 32

5.3.2 Input parameters for Dolby Atmos

When creating an XML file, you can use the examples of the input configuration within an XML file and tables that include all the supported Dolby Digital Plus with Dolby Atmos content input parameters with the available and default values.

The input file format can be either Dolby Atmos master file set or ADM BWF.

Input parameters for Dolby Atmos master file set

Table 13: Input parameters for Dolby Atmos master file set

Parameter	Type	Value	Description
`<damf>`	XML element		Parent element for a Dolby Atmos master file set.
`<file_name>`	String	*filename*	Specifies the .damf or .atmos file.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored input.
`<path>`	String	*folder_path*	Path to the directory that contains the input file set (for example, `<path>path/to/media</path>`).

Input parameters for ADM BWF

Table 14: Input parameters for ADM BWF files

Parameter	Type	Value	Description
<adm>	XML element		Parent element for ADM BWF files.
<file_name>	String	*filename*	
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored input.
<path>	String	*folder_path*	Path to the directory that contains the input file (for example, <path>path/to/media</path>).

Related information

Dolby Atmos encoding job on page 32

5.3.3 Filter parameters for Dolby Atmos

When creating an XML file, you can use the following Dolby Digital Plus with Dolby Atmos content filter parameters.

Table 15: Dolby Digital Plus with Dolby Atmos content filter parameters

Parameter	Type	Values	Notes
<atmos_mezz_to_atmos_ddp>	XML element		Parent element for the Dolby Digital Plus with Dolby Atmos content filter parameters.
<loudness> *	XML element		Parent element for loudness metering.
<metering_mode>	String	1770-4, 1770-3, 1770-2, 1770-1, LeqA Default: 1770-4	Loudness measuring mode according to one of the broadcast standards.
<dialogue_intelligence>	Boolean	true, false Default: true	Dialogue Intelligence enabled. Option ignored for 1770-1 or LeqA metering mode.
<speech_threshold>	Integer	0-100 Default: 20	Speech threshold percentage.
<data_rate>	Integer	384, 448, 640, 768 Default: 384	Target data rate.
<timecode_frame_rate>	String	not_indicated, 23.976, 24, 25, 29.97, 30, 48, 50, 59.94, 60 Default: not_indicated	Frame rate associated with the specified timecode.
<start>	String	*HH:MM:SS:FF* or *HH:MM:SS.xx* (timecode) or first_frame_of_action Default: first_frame_of_action	Timecode (*HH:MM:SS:FF* or *HH:MM:SS.xx*) indicating starting point. Add df for drop-frame, e.g. *HH:MM:SS:FF*df.

Table 15: Dolby Digital Plus with Dolby Atmos content filter parameters (continued)

Parameter	Type	Values	Notes
`<end>`	String	*`HH:MM:SS:FF`* or *`HH:MM:SS.xx`* or end_of_file Default: end_of_file	Timecode (*`HH:MM:SS:FF`* or *`HH:MM:SS.xx`*) indicating ending point. Add df for drop-frame, e.g. *`HH:MM:SS:FF`*df.
`<prepend_silence_duration>`	String	*`seconds.miliseconds`* Default: 0	Decimal number representing the duration of silence prepended to output.
`<drc>`	XML element		Parent element for dynamic range control parameters.
`<line_mode_drc_profile>`	String	film_standard, film_light, music_standard, music_light, speech, none Default: film_light	Profile for dynamic range control line mode.
`<rf_mode_drc_profile>`	String	film_standard, film_light, music_standard, music_light, speech, none Default: film_light	Profile for dynamic range control RF mode.
`<downmix>`			Parent element for downmix parameters.
`<loro_center_mix_level>`	String	+3, +1.5, 0, -1.5, -3, -4.5, -6, -inf Default: -3	Lo/Ro center downmix level.
`<loro_surround_mix_level>`	String	-1.5, -3, -4.5, -6, -inf Default: -3	Lo/Ro surround downmix level.
`<ltrt_center_mix_level>`	String	+3, +1.5, 0, -1.5, -3, -4.5, -6, -inf Default: -3	Lt/Rt center downmix level.
`<ltrt_surround_mix_level>`	String	-1.5, -3, -4.5, -6, -inf Default: -3	Lt/Rt surround downmix level.
`<preferred_downmix_mode>`	String	not_indicated, loro, ltrt, ltrt-pl2 Default: ltrt-pl2	Stereo downmix preference.

[*] Dolby Digital Plus with Dolby Atmos content encoding supports loudness measurement but not loudness correction.

Related information

Dolby Atmos encoding job on page 32

5.3.4 Output parameters for Dolby Atmos

When creating an XML file, you can use the examples of the output configuration. The tables include all the supported Dolby Digital Plus with Dolby Atmos content output parameters with the available and default values.

Output parameters for E-AC-3

Table 16: E-AC-3 output parameters

Parameter	Type	Value	Description
<ec3>	XML element		Parent element for an E-AC-3 output file.
<file_name>	String	*filename*	Output file name.
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored output.
<path>	String	*Storage folder path*	Path to the directory to save the output file (for example, <path>path/to/media</path>).

Output parameters for MP4

Table 17: MP4 output parameters

Parameter	Type	Value	Description
<mp4>	XML element		Parent element for an MP4 output file.
<output_format>	String	mp4, dash	Defines whether the output is an MP4 or DASH file.
<min_frag_duration>	Integer		Minimum fragment duration, in milliseconds.
<max_frag_duration>	Integer		Maximum fragment duration, in milliseconds.
<mux_flag>	String	encrypt_style_piff, write_iods, write_pdin, write_bloc, write_ainf, write_ctts_v1, write_subs_v1	Sets specified multiplexer flag.
<frag_flag>	String	frag_style_default, frag_style_cff, write_tfdt, write_sdtp, write_trik, write_sidx, write_mfra, force_tfra, no_base_data_offset, empty_trex, empty_tfhd, one_tfra_per_traf, default_base_is_moof	Sets specified fragmentation flag.
<track_flag>	String	track_enabled, track_in_movie, track_in_preview, track_in_poster	Sets specified track flag.
<file_name>	String	*filename*	Output file name.
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.

Table 17: MP4 output parameters (continued)

Parameter	Type	Value	Description
`<local>`	XML element		Parent element for parameters of locally stored output.
`<path>`	String	*Storage folder path*	Path to the directory to save the output file (for example, `<path>path/to/media</path>`).

Output parameters for transport stream

Table 18: Transport stream output parameters

Parameter	Type	Value	Description
`<ts>`	XML element		Parent element for a transport stream output file.
`<packetize_standard>`	String	`mpeg`, `atsc`, `dvb` Default: `mpeg`	Indicates the packetize standard used for substreams.
`<packetize_mode>`	String	`cbr`, `vbr` Default: `vbr`	Indicates constant bit rate (CBR) or variable bit rate (VBR) of the substreams.
`<transport_stream_id>`	Integer	Default: `0`	Transport stream ID.
`<audio_pid>`	Integer	32–8186 Default: `40`	Packet ID of the audio stream.
`<video_pid>`	Integer	32–8186 Default: `50`	Packet ID of the video stream.
`<file_name>`	String	*filename*	Output file name.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored output.
`<path>`	String	*Storage folder path*	Path to the directory to save the output file (for example, `<path>path/to/media</path>`).
`<plugin>`	XML element	Depends on the selected plug-in. Example: `<base>`	Allows you to choose the transport stream multiplexing plug-in. The base implementation is delivered with Dolby Encoding Engine.

Related information

Dolby Atmos encoding job on page 32

5.4 Dolby Atmos encoding job (from channel-based immersive input)

The Dolby Atmos encoder produces Dolby Digital Plus with Dolby Atmos content audio streams based on channel-based immersive input, according to the parameters you specify in the job configuration XML file.

The input for this encoding job is an interleaved .wav file with 16 tracks because the supported configuration is 9.1.6.

The output is in E-AC-3 format. The output can also be multiplexed into an MP4 or transport stream containers.

The process for encoding Dolby Atmos is different than the process for encoding Dolby Atmos with channel-based immersive input; therefore, separate filters are defined for them.

5.4.1 XML example files for Dolby Atmos (channel-based immersive)

The XML template files are examples of a job configuration.

You can use them to create job configurations. You can find them in the xml_templates folder. For the Dolby Atmos encoding job, refer to the wav_pcm_to_atmos_ddp_ec3.xml file.

5.4.2 Input parameters for Dolby Atmos (channel-based immersive)

When creating an XML file, you can use the tables that include all supported channel-based immersive input parameters.

The input for this encoding job is an interleaved .wav file with 16 tracks (the supported configuration is 9.1.6).

Input parameters for a PCM (WAV) file

Table 19: Input parameters for a PCM (WAV) file

Parameter	Type	Value	Description
`<wav>`	XML element		Parent element for a single file with mono, stereo, or multichannel PCM.
`<file_name>`	String	*`filename`*	
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored input.
`<path>`	String	*`folder_path`*	Path to the directory that contains the input file (for example, `<path>path/to/media</path>`).

5.4.3 Filter parameters for Dolby Atmos (channel-based immersive)

When creating an XML file, you can use the following Dolby Digital Plus with Dolby Atmos content filter parameters.

Table 20: Dolby Digital Plus with Dolby Atmos content filter parameters

Parameter	Type	Values	Notes
`<pcm_to_atmos_ddp>`	XML element		Parent element for the Dolby Digital Plus with Dolby Atmos content filter parameters.
`<loudness>` *	XML element		Parent element for loudness metering.
`<metering_mode>`	String	1770-4, 1770-3, 1770-2, 1770-1, LeqA Default: 1770-4	Loudness measuring mode according to one of the broadcast standards.

Table 20: Dolby Digital Plus with Dolby Atmos content filter parameters (continued)

Parameter	Type	Values	Notes
`<dialogue_intelligence>`	Boolean	true, false Default: true	Dialogue Intelligence enabled. Option ignored for 1770-1 or LeqA metering mode.
`<speech_threshold>`	Integer	0-100 Default: 20	Speech threshold percentage.
`<data_rate>`	Integer	384, 448, 640, 768 Default: 384	Target data rate.
`<timecode_frame_rate>`	String	not_indicated, 23.976, 24, 25, 29.97, 30, 48, 50, 59.94, 60 Default: not_indicated	Frame rate associated with the specified timecode.
`<start>`	String	*HH:MM:SS:FF* or *HH:MM:SS.xx* (timecode) or first_frame_of_action Default: first_frame_of_action	Timecode (*HH:MM:SS:FF* or *HH:MM:SS.xx*) indicating starting point. first_frame_of_action means sample position 0. Add df for drop-frame (for example, *HH:MM:SS:FF*df).
`<end>`	String	*HH:MM:SS:FF* or *HH:MM:SS.xx* or end_of_file Default: end_of_file	Timecode (*HH:MM:SS:FF* or *HH:MM:SS.xx*) indicating ending point. Add df for drop-frame (for example, *HH:MM:SS:FF*df).
`<prepend_silence_duration>`	String	*seconds.miliseconds* Default: 0	Decimal number representing the duration of silence prepended to output.
`<drc>`	XML element		Parent element for dynamic range control parameters.
`<line_mode_drc_profile>`	String	film_standard, film_light, music_standard, music_light, speech, none Default: film_light	Profile for dynamic range control line mode.
`<rf_mode_drc_profile>`	String	film_standard, film_light, music_standard, music_light, speech, none Default: film_light	Profile for dynamic range control RF mode.
`<downmix>`			Parent element for downmix parameters.
`<loro_center_mix_level>`	String	+3, +1.5, 0, -1.5, -3, -4.5, -6, -inf Default: -3	Lo/Ro Center downmix level.
`<loro_surround_mix_level>`	String	-1.5, -3, -4.5, -6, -inf Default: -3	Lo/Ro surround downmix level.
`<ltrt_center_mix_level>`	String	+3, +1.5, 0, -1.5, -3, -4.5, -6, -inf Default: -3	Lt/Rt Center downmix level.

Table 20: Dolby Digital Plus with Dolby Atmos content filter parameters (continued)

Parameter	Type	Values	Notes
`<ltrt_surround_mix_level>`	String	-1.5, -3, -4.5, -6, -inf Default: -3	Lt/Rt surround downmix level.
`<preferred_downmix_mode>`	String	not_indicated, loro, ltrt, ltrt-pl2 Default: ltrt-pl2	Stereo downmix preference.

* Dolby Digital Plus with Dolby Atmos content encoding supports loudness measurement but not loudness correction.

5.4.4 Output parameters for Dolby Atmos (channel-based immersive)

When creating an XML file, you can use the following Dolby Digital Plus with Dolby Atmos content output parameters.

Output parameters for E-AC-3

Table 21: E-AC-3 output parameters

Parameter	Type	Value	Description
`<ec3>`	XML element		Parent element for an E-AC-3 output file.
`<file_name>`	String	*filename*	Output file name.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored output.
`<path>`	String	*Storage folder path*	Path to the directory to save the output file (for example, `<path>path/to/media</path>`).

Output parameters for MP4

Table 22: MP4 output parameters

Parameter	Type	Value	Description
`<mp4>`	XML element		Parent element for an MP4 output file.
`<output_format>`	String	`mp4`, dash	Defines whether the output is an MP4 or DASH file.
`<min_frag_duration>`	Integer		Minimum fragment duration, in milliseconds.
`<max_frag_duration>`	Integer		Maximum fragment duration, in milliseconds.
`<mux_flag>`	String	`encrypt_style_piff`, `write_iods`, `write_pdin`, `write_bloc`, `write_ainf`, `write_ctts_v1`, `write_subs_v1`	Sets specified multiplexer flag.

Table 22: MP4 output parameters (continued)

Parameter	Type	Value	Description
<frag_flag>	String	frag_style_default, frag_style_cff, write_tfdt, write_sdtp, write_trik, write_sidx, write_mfra, force_tfra, no_base_data_offset, empty_trex, empty_tfhd, one_tfra_per_traf, default_base_is_moof	Sets specified fragmentation flag.
<track_flag>	String	track_enabled, track_in_movie, track_in_preview, track_in_poster	Sets specified track flag.
<file_name>	String	*filename*	Output file name.
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored output.
<path>	String	*Storage folder path*	Path to the directory to save the output file (for example, <path>path/to/media</path>).

Output parameters for transport stream

Table 23: Transport stream output parameters

Parameter	Type	Value	Description
<ts>	XML element		Parent element for a transport stream output file.
<packetize_standard>	String	mpeg, atsc, dvb Default: mpeg	Indicates the packetize standard used for substreams.
<packetize_mode>	String	cbr, vbr Default: vbr	Indicates constant bit rate (CBR) or variable bit rate (VBR) of the substreams.
<transport_stream_id>	Integer	Default: 0	Transport stream ID.
<audio_pid>	Integer	32–8186 Default: 40	Packet ID of the audio stream.
<video_pid>	Integer	32–8186 Default: 50	Packet ID of the video stream.
<file_name>	String	*filename*	Output file name.
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored output.

Table 23: Transport stream output parameters (continued)

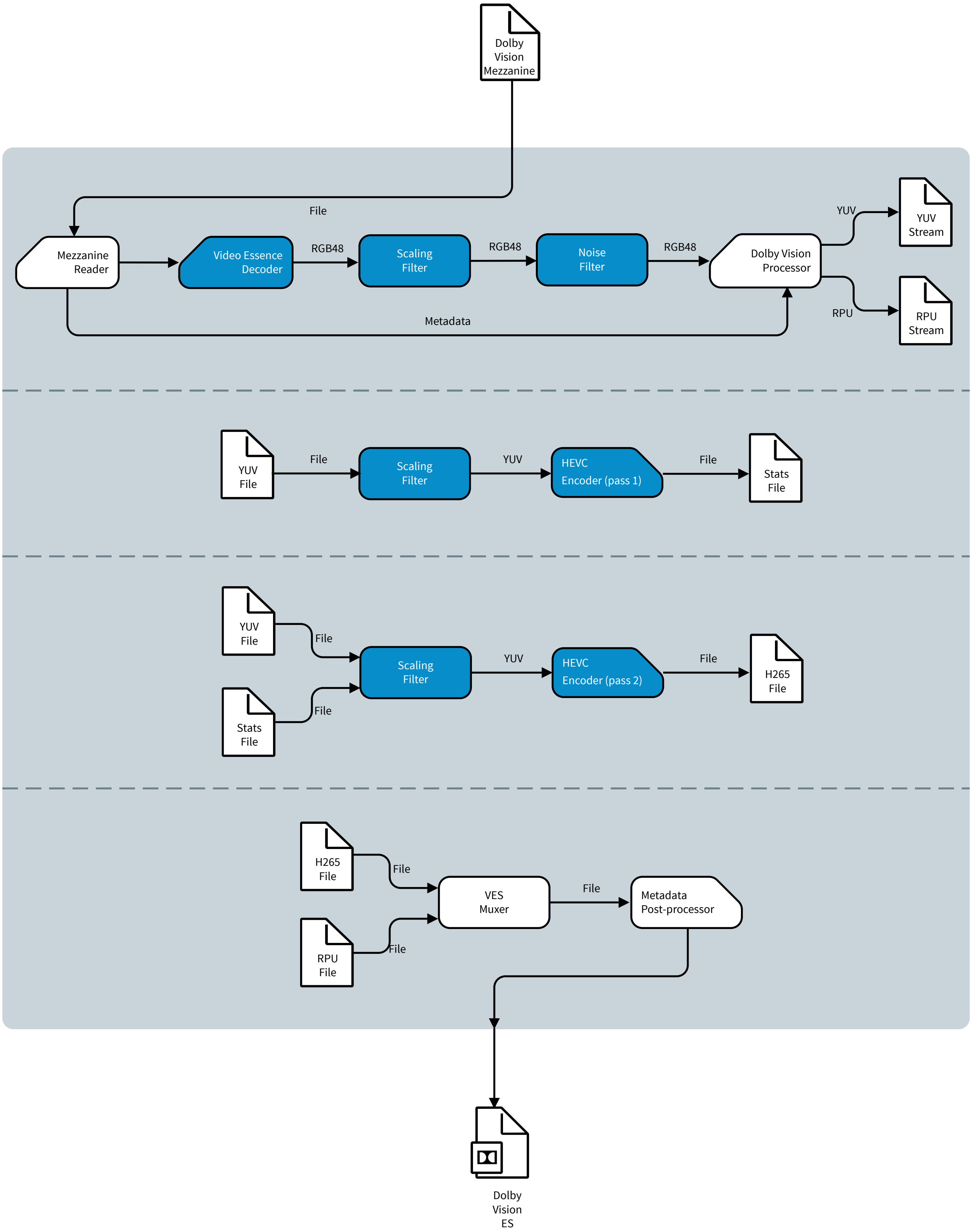
Parameter	Type	Value	Description
`<path>`	String	*`Storage folder path`*	Path to the directory to save the output file (for example, `<path>path/to/media</path>`).
`<plugin>`	XML element	Depends on the selected plug-in. Example: `<base>`	Allows you to choose the transport stream multiplexing plug-in. The base implementation is delivered with Dolby Encoding Engine.

5.5 Dolby Vision profile 5 encoding job

The Dolby Vision encoder can produce Dolby Vision video streams, according to the parameters you specify in the job configuration XML file.

A Dolby Vision profile 5 encoding job produces a Dolby Vision video stream suitable for playback on a device supporting Dolby Vision.

Figure 2: Dolby Vision profile 5 encode process



The input source can be MXF, MXF with sidecar Dolby Vision XML metadata, JPEG 2000 with sidecar Dolby Vision XML metadata, Apple ProRes with sidecar Dolby Vision XML metadata, and MOV with sidecar Dolby Vision XML metadata (with Apple ProRes essence).

The output is an HEVC video stream. The output can also be multiplexed into MP4 or transport stream containers.

As a part of the encoding job, the metadata postprocessor updates Dolby Vision metadata within the RPU after video elementary stream multiplexing. The postprocessor smooths out related metadata for scenes and recalculates maximum values from all frames within the content.

The components marked in blue in the figure above are replaceable plugins. A standard installation of Dolby Encoding Engine includes default plugin implementations for J2K decoding, scaling, and noise reduction. The Dolby Encoding Engine allows you to replace these components with others that perform the same processing functions.

Note: The Dolby Encoding Engine does not include an HEVC encoder plugin but requires such an encoder to be built and installed. For more information, refer to the *Dolby Encoding Engine HEVC Encoder Installation Guide*.

Related information

Multiple Dolby Vision data rates and resolutions on page 117

XML example files for Dolby Vision profile 5 on page 44

Input parameters for Dolby Vision profile 5 on page 44

Filter parameters for Dolby Vision profile 5 on page 47

Output parameters for Dolby Vision profile 5 on page 51

Encoding parameters on page 20

5.5.1 XML example files for Dolby Vision profile 5

The XML template files are examples of a job configuration.

You can use them to create job configurations. You can find them in the xml_templates folder. For the Dolby Vision profile 5 encoding job, refer to the following files:

- mxf_dv_mezz_to_dv_profile_5_hevc.xml
- mxf_dv_mezz_to_dv_profile_5_mp4.xml
- mxf_sidecar_dv_mezz_to_dv_profile_5_hevc.xml
- mxf_sidecar_dv_mezz_to_dv_profile_5_mp4.xml
- j2k_sidecar_dv_mezz_to_dv_profile_5_hevc.xml
- j2k_sidecar_dv_mezz_to_dv_profile_5_mp4.xml
- prores_sidecar_dv_mezz_to_dv_profile_5_hevc.xml
- mov_sidecar_dv_mezz_to_dv_profile_5_hevc.xml

Related information

Dolby Vision profile 5 encoding job on page 42

5.5.2 Input parameters for Dolby Vision profile 5

When creating an XML file, you can use Dolby Vision profile 5 input parameters.

The supported input file formats are:

- MXF (with JPEG 2000 essence)
- MXF with sidecar Dolby Vision XML metadata (with JPEG 2000 essence)

- JPEG 2000 with sidecar Dolby Vision XML metadata
- Apple ProRes with sidecar Dolby Vision XML metadata
- MOV with sidecar Dolby Vision XML metadata (with Apple ProRes essence)

Input parameters for MXF

Table 24: MXF input parameters

Parameter	Type	Value	Description
`<mxf>`	XML element		Parent element.
`<override_frame_rate>`	String	`23.976`, `24`, `25`, `29.97`, `30`, `48`, `50`, `59.94`, `60`	If present, the frame rate stored in the XML metadata within the container will be ignored and replaced with this value.
`<file_name>`	String	*filename*	
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored input.
`<path>`	String	*folder_path*	Path to the directory that contains the input file (for example, `<path>path/to/media</path>`).

Input parameters for MXF with sidecar metadata

Table 25: MXF with sidecar metadata input parameters

Parameter	Type	Value	Description
`<mxf_sidecar>`	XML element		Parent element.
`<override_frame_rate>`	String	`23.976`, `24`, `25`, `29.97`, `30`, `48`, `50`, `59.94`, `60`	If present, the frame rate stored in the XML metadata within the container will be ignored and replaced with this value.
`<mxf_file_name>`	String	*filename*	Name of the MXF file containing video essence (JPEG 2000).
`<md_file_name>`	String	*filename*	Name of the file containing Dolby Vision metadata.
`<index_file_name>`	String	*filename*	Optionally, the name of an index file (if it was generated in an MXF parsing job).
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored input..
`<path>`	String	*folder_path*	Path to the directory that contains the input file (for example, `<path>path/to/media</path>`).

Input parameters for JPEG 2000 with sidecar metadata

Table 26: JPEG 2000 with sidecar metadata input parameters

Parameter	Type	Value	Description
`<j2k_sidecar>`	XML element		Parent element.
`<width>`	Integer		Input picture width.
`<height>`	Integer		Input picture height.
`<frame_rate>`	String	*23.976*, *24*, *25*, *29.97*, *30*, *59.94*, *60*	Input frame rate.
`<j2k_prefix>`	String		Allows you to use JPEG 2000 files with a specific prefix. If left empty (default), all JPEG 2000 files found in storage are used.
`<md_file_name>`	String	*filename*	Name of the file containing Dolby Vision metadata.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored input.
`<path>`	String	*folder_path*	Path to the directory that contains the input file (for example, `<path>path/to/media</path>`).

Input parameters for Apple ProRes with sidecar metadata

Table 27: Apple ProRes with sidecar metadata input parameters

Parameter	Type	Value	Description
`<prores_sidecar>`	XML element		Parent element.
`<codec>`	String	apch, apcn, apcs, apco, ap4h, ap4x	The type of the Apple ProRes codec: • apch: Apple ProRes 422 HQ • apcn: Apple ProRes 422 • apcs: Apple ProRes 422 LT • apco: Apple ProRes 422 Proxy • ap4h: Apple ProRes 4444 • ap4x: Apple ProRes 4444 XQ The particular type of the Apple ProRes codec can be automatically obtained from the manifest file. (If you use such a file, see the information in this section).
`<width>`			Input picture width.
`<height>`			Input picture height.
`<frame_rate>`	String	23.976, 24, 25, 29.97, 30, 48, 50, 59.94, 60	Input picture frame rate.
`<prores_prefix>`	String		Allows you to use Apple ProRes files with a specific prefix. If left empty (default), all Apple ProRes files found in storage are used.

Table 27: Apple ProRes with sidecar metadata input parameters (continued)

Parameter	Type	Value	Description
<md_file_name>	String	*filename*	Name of the file containing Dolby Vision metadata.
<prores_manifest>	String	*filename*	Name of the optional manifest file containing details of the Apple ProRes frames (generated by the extract_prores filter). If the manifest is present, you do not have to provide the <codec>, <width>, <height>, and <frame_rate> parameters.
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored input.
<path>	String	*folder_path*	Path to the directory that contains the input file (for example, <path>path/to/media</path>).

Input parameters for MOV with sidecar metadata

Table 28: MOV with sidecar metadata input parameters

Parameter	Type	Value	Description
<mov_sidecar>	XML element		Parent element.
<override_frame_rate>	String	23.976, 24, 25, 29.97, 30, 48, 50, 59.94, 60	If present, the frame rate stored in the XML metadata within the container will be ignored and replaced with this value.
<track_id>	Integer	Default: -1	ID of track to demultiplex. Negative value selects the first found video track.
<mov_file_name>	String	*filename*	Name of the MOV file containing video essence (Apple ProRes).
<md_file_name>	String	*filename*	Name of the file containing Dolby Vision metadata.
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored input.
<path>	String	*folder_path*	Path to the directory that contains the input file (for example, <path>path/to/media</path>).

Related information

Dolby Vision profile 5 encoding job on page 42

5.5.3 Filter parameters for Dolby Vision profile 5

When creating an XML file, you can use the following Dolby Vision profile 5 filter parameters.

The Dolby Vision profile 5 filter parameters are:

The table below describes the available parameters. The table does not include the parameters required by plugins, since those parameters depend on the plugins you choose to implement in Dolby Encoding Engine.

Table 29: Dolby Vision profile 5 filter parameters

Parameter	Type	Value	Description
`<dv_mezz_to_dv_profile_5>`	XML element		Parent element for the Dolby Vision profile 5 filter parameters.
`<start>`	Integer	Default: 0	Start frame or start timecode.
`<end>`	Integer	Default: 0	End frame or end timecode. (0=end of file)
`<metadata_offset>`	Integer	Default: 0	Number of frames to skip while reading metadata. Use only when video frames and metadata are not aligned.
`<active_area_offset_top>`	Integer	Default: 0	Top offset of the active area in the source content. Active area determines where in the encoded picture the actual video content is located. If the video is encoded with solid color bars around the content, setting the appropriate active area will exclude them from processing.
`<active_area_offset_bottom>`	Integer	Default: 0	Bottom offset of the active area in the source content. Active area determines where in the encoded picture the actual video content is located. If the video is encoded with solid color bars around the content, setting the appropriate active area will exclude them from processing.
`<active_area_offset_left>`	Integer	Default: 0	Left offset of the active area in the source content. Active area determines where in the encoded picture the actual video content is located. If the video is encoded with solid color bars around the content, setting the appropriate active area will exclude them from processing.
`<active_area_offset_right>`	Integer	Default: 0	Right offset of the active area in the source content. Active area determines where in the encoded picture the actual video content is located. If the video is encoded with solid color bars around the content, setting the appropriate active area will exclude them from processing.

Table 29: Dolby Vision profile 5 filter parameters (continued)

Parameter	Type	Value	Description
`<decoder_parallel_frames>`	Integer	Values between 0 and 16. Default: 0	Number of parallel video essence pictures to process. Higher values can reduce processing time at the cost of increased CPU and memory usage. The default 0 value sets the parameter to auto. Depending on which decoder you select, the auto value means: • For the JPEG 2000 Base decoder, four decoder instatnces run in parallel. • For the Kakadu decoder, two multi-threaded instances run in parallel. • For the Apple ProRes decoder, the process uses all available cores.
`<prepend_frames>`	Integer	Default: 0	Number of black frames prepending the target content.
`<append_frames>`	Integer	Default: 0	Number of black frames appending the target content.
`<target_width>`	Integer list	Accepts multiple integer values, separated by a space. Default: 0	Width of target picture (0 = use source width).
`<target_height>`	Integer list	Accepts multiple integer values, separated by a space. Default: 0	Height of target picture (0 = use source height).
`<max_scene_frames>`	Integer	Default: 256	The maximum number of frames in a buffer within the Dolby Vision processor. It is recommended to use a number that is a multiple of the GOP size (e.g. for GOP Size=48, max_scene_frames=240) and not smaller than GOP size. This property has significant impact on RAM consumption during the Dolby Vision processor execution.
`<preprocessing_mode>`	String	optimized, standard Default: optimized	Allows you to choose between the faster optimized mode and the standard mode.
`<data_rate>`	Integer list	Accepts multiple integer values, separated by a space. Default: 15000	Average data rate in kbps.

Table 29: Dolby Vision profile 5 filter parameters (continued)

Parameter	Type	Value	Description
`<max_vbv_data_rate>`	Integer list	Accepts multiple integer values, separated by a space. Default: 15000	Maximum video buffering verifier data rate in kbps.
`<vbv_buffer_size>`	Integer list	Accepts multiple integer values, separated by a space. Default: 30000	Video buffering verifier buffer size in kb.
`<encode_pass_num>`	Integer	Default: 1	Number of encoder passes. *
`<disable_md_postproc>`	Boolean	Default: false	Disables Dolby Vision metadata post-processing.
`<max_cll>`	Integer	Values between -1 to 65535. Default: -1	Maximum content light level in units of 1 nit. The default negative value is ignored and forces filter to calculate actual value.
`<max_fall>`	Integer	Values between -1 to 65535. Default: -1	Maximum frame average light level in units of 1 nit. The default negative value is ignored and forces filter to calculate actual value.
`<prepend_frames_method>`	Integer	0, 1 Default: 1	RPU override for the prepending black frames method: • 0 - RPU pregenerated from file • 1 - RPU copied from first frame Note: The RPU pregenerated from file guarantees black frames but the feature is experimental and can cause undefined behaviour of the Dolby Encoding Engine.
`<append_frames_method>`	Integer	0, 1 Default: 1	RPU override for the appending black frames method: • 0 - RPU pregenerated from file • 1 - RPU copied from last frame Note: The RPU pregenerated from file guarantees black frames but the feature is experimental and can cause undefined behaviour of the Dolby Encoding Engine.
`<decoder>`	XML element	`<j2k_dec>` or `<prores_dec>`	Specifies the input decoder.
`<scaling>`	XML element	Example: `<base>`	Specifies the scaling plugin. The base implementation is delivered with Dolby Encoding Engine.
`<source_crop_top>`	Integer	Default: 0	A positive value indicates how many pixels should be removed from the top of the source picture before scaling; a negative value will pad the top with black pixels.

Table 29: Dolby Vision profile 5 filter parameters (continued)

Parameter	Type	Value	Description
`<source_crop_bottom>`	Integer	Default: 0	A positive value indicates how many pixels should be removed from the bottom of the source picture before scaling; a negative value will pad the bottom with black pixels.
`<source_crop_left>`	Integer	Default: 0	A positive value indicates how many pixels should be removed from the left side of the source picture before scaling; a negative value will pad the left side with black pixels.
`<source_crop_right>`	Integer	Default: 0	A positive value indicates how many pixels should be removed from the right side of the source picture before scaling; a negative value will pad the right side with black pixels.
`<noise>`	XML element	Example: `<base>`	Specifies the noise reduction filter plugin. The base implementation is delivered with Dolby Encoding Engine.
`<strength>`	Integer	Default: 0	In the case of the `<base>` noise filter, the only currently supported value is 0 (noise reduction turned off).
`<hevc_enc>`	XML element	Example: `<impact>`	Specifies the HEVC encoder plugin.

[*] An encoder pass is a single instance of processing a mezzanine YUV file.

Related information

Dolby Vision profile 5 encoding job on page 42

5.5.4 Output parameters for Dolby Vision profile 5

When creating an XML file, you can use the examples of the output configuration. Tables include all of the supported Dolby Vision profile 5 output parameters with the available and default values.

Output parameters for HEVC

Table 30: HEVC output parameters

Parameter	Type	Value	Description
`<hevc>`	XML element		Parent element for HEVC elementary stream output.
`<file_name>`	String	*`filename`*	Output file name.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored output.
`<path>`	String	*`Storage folder path`*	Path to the directory to save the output file (for example, `<path>path/to/media</path>`).

Output parameters for MP4

Table 31: MP4 output parameters

Parameter	Type	Value	Description
<mp4>	XML element		Parent element for an MP4 output file.
<output_format>	String	mp4, dash	Defines whether the output is an MP4 or DASH file.
<min_frag_duration>	Integer		Minimum fragment duration, in milliseconds.
<max_frag_duration>	Integer		Maximum fragment duration, in milliseconds.
<mux_flag>	String	encrypt_style_piff, write_iods, write_pdin, write_bloc, write_ainf, write_ctts_v1, write_subs_v1	Sets specified multiplexer flag.
<frag_flag>	String	frag_style_default, frag_style_cff, write_tfdt, write_sdtp, write_trik, write_sidx, write_mfra, force_tfra, no_base_data_offset, empty_trex, empty_tfhd, one_tfra_per_traf, default_base_is_moof	Sets specified fragmentation flag.
<track_flag>	String	track_enabled, track_in_movie, track_in_preview, track_in_poster	Sets specified track flag.
<file_name>	String	*filename*	Output file name.
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored output.
<path>	String	*Storage folder path*	Path to the directory to save the output file (for example, <path>path/to/media</path>).

Output parameters for transport stream

Table 32: Transport stream output parameters

Parameter	Type	Value	Description
<ts>	XML element		Parent element for a transport stream output file.
<packetize_standard>	String	mpeg, atsc, dvb Default: mpeg	Indicates the packetize standard used for substreams.
<packetize_mode>	String	cbr, vbr Default: vbr	Indicates constant bit rate (CBR) or variable bit rate (VBR) of the substreams.

Table 32: Transport stream output parameters (continued)

Parameter	Type	Value	Description
`<transport_stream_id>`	Integer	Default: `0`	Transport stream ID.
`<audio_pid>`	Integer	32–8186 Default: `40`	Packet ID of the audio stream.
`<video_pid>`	Integer	32–8186 Default: `50`	Packet ID of the video stream.
`<file_name>`	String	*`filename`*	Output file name.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored output.
`<path>`	String	*`Storage folder path`*	Path to the directory to save the output file (for example, `<path>path/to/media</path>`).
`<plugin>`	XML element	Depends on the selected plug-in. Example: `<base>`	Allows you to choose the transport stream multiplexing plug-in. The base implementation is delivered with Dolby Encoding Engine.

Related information

Dolby Vision profile 5 encoding job on page 42

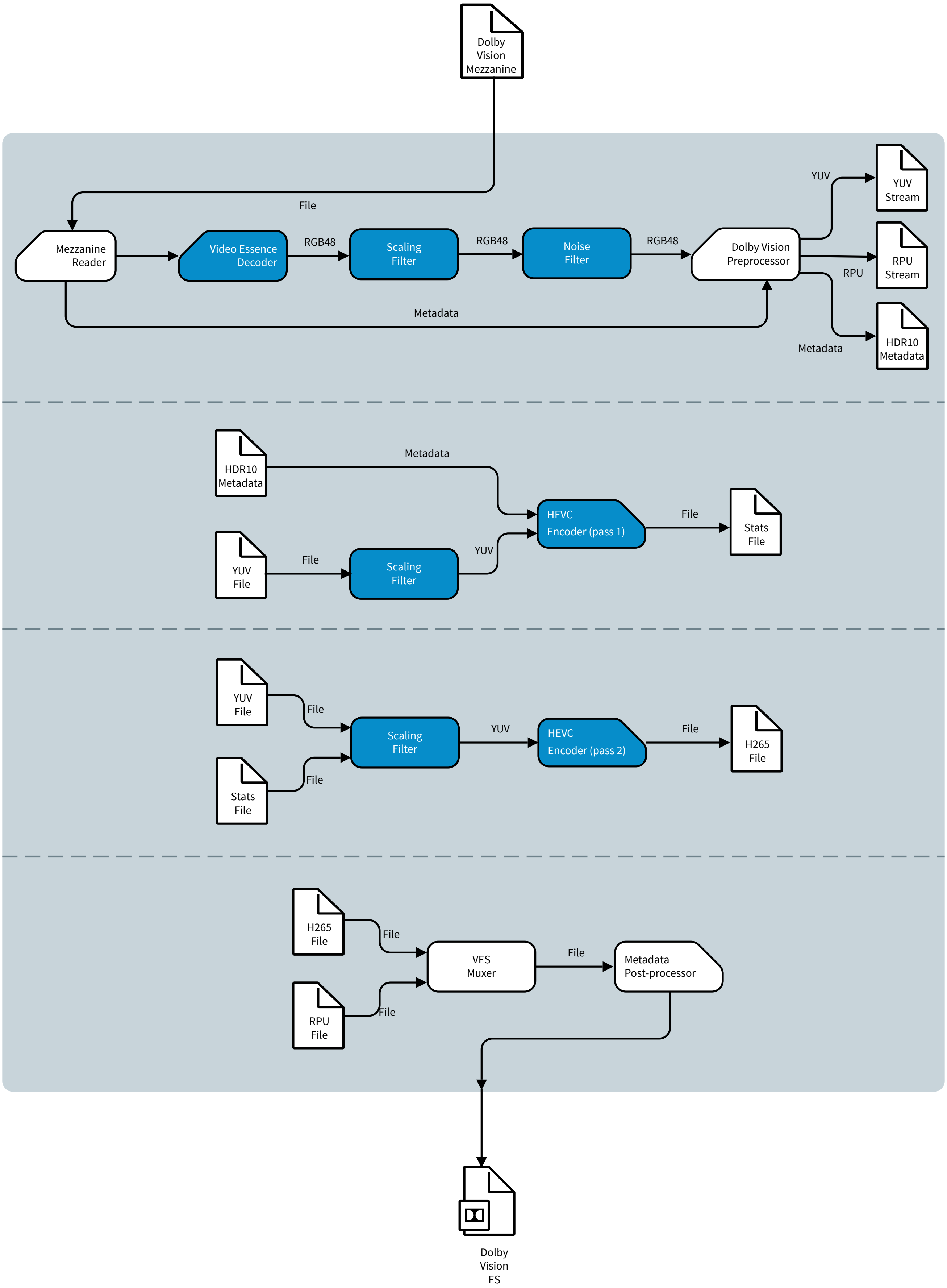
5.6 Dolby Vision profile 8.1 encoding job

The Dolby Vision encoder can produce Dolby Vision video streams, according to the parameters you specify in the job configuration XML file.

A Dolby Vision profile 8.1 allows the final Dolby Vision bitstream to service HDR10 and Dolby Vision enabled devices.

The Dolby Encoding Engine provides only the no-mapping variation of the Dolby Vision profile 8.1. That means that content mapping is not applied to the source to create the HDR10 base layer, which allows the HDR10 base layer to retain similar nit levels as the source.

Figure 3: Dolby Vision profile 8.1 encode process



The input source can be MXF, MXF with sidecar Dolby Vision XML metadata, JPEG 2000 with sidecar Dolby Vision XML metadata, Apple ProRes with sidecar Dolby Vision XML metadata, and MOV with sidecar Dolby Vision XML metadata (with Apple ProRes essence).

The output is an HEVC video stream. The output can also be multiplexed into MP4 or transport stream containers.

As a part of the encoding job, the metadata postprocessor updates the Dolby Vision metadata in the RPU within the video elementary stream. It also updates HDR10 supplemental enhancement information (SEI) messages, such as mastering display color volume and content light level information, within the base-layer video elementary stream.

The components marked in blue in the figure above are replaceable plug-ins. A standard installation of Dolby Encoding Engine includes default plug-in implementations for J2K decoding, scaling, and noise reduction. The Dolby Encoding Engine allows you to replace these components with others that perform the same processing functions.

Note: The Dolby Encoding Engine does not include an HEVC encoder plugin but requires such an encoder to be built and installed. For more information, refer to the *Dolby Encoding Engine HEVC Encoder Installation Guide*.

5.6.1 XML example files for Dolby Vision profile 8.1

There is example content of an XML job configuration file.

You can use them to create job configurations. You can find them in the xml_templates folder. For the Dolby Vision profile 8.1 encoding job, refer to the following files:

- mxf_dv_mezz_to_dv_profile_8_1_hevc.xml
- mxf_dv_mezz_to_dv_profile_8_1_mp4.xml
- mxf_sidecar_dv_mezz_to_dv_profile_8_1_hevc.xml
- mxf_sidecar_dv_mezz_to_dv_profile_8_1_mp4.xml
- j2k_sidecar_dv_mezz_to_dv_profile_8_1_hevc.xml
- j2k_sidecar_dv_mezz_to_dv_profile_8_1_mp4.xml
- prores_sidecar_dv_mezz_to_dv_profile_8_1_hevc.xml
- mov_sidecar_dv_mezz_to_dv_profile_8_1_hevc.xml

5.6.2 Input parameters for Dolby Vision profile 8.1

When creating an XML file, you can use the following Dolby Vision profile 8.1 input parameters.

The supported input file formats are:

- MXF (with JPEG 2000 essence)
- MXF with sidecar Dolby Vision XML metadata (with JPEG 2000 essence)
- JPEG 2000 with sidecar Dolby Vision XML metadata
- Apple ProRes with sidecar Dolby Vision XML metadata
- MOV with sidecar Dolby Vision XML metadata (with Apple ProRes essence)

Input parameters for MXF

Table 33: MXF input parameters

Parameter	Type	Value	Description
`<mxf>`	XML element		Parent element.
`<override_frame_rate>`	String	`23.976`, `24`, `25`, `29.97`, `30`, `48`, `50`, `59.94`, `60`	If present, the frame rate stored in the XML metadata within the container will be ignored and replaced with this value.
`<file_name>`	String	*filename*	
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored input.
`<path>`	String	*folder_path*	Path to the directory that contains the input file (for example, `<path>path/to/media</path>`).

Input parameters for MXF with sidecar metadata

Table 34: MXF with sidecar metadata input parameters

Parameter	Type	Value	Description
`<mxf_sidecar>`	XML element		Parent element.
`<override_frame_rate>`	String	`23.976`, `24`, `25`, `29.97`, `30`, `48`, `50`, `59.94`, `60`	If present, the frame rate stored in the XML metadata within the container will be ignored and replaced with this value.
`<mxf_file_name>`	String	*filename*	Name of the MXF file containing video essence (JPEG 2000).
`<md_file_name>`	String	*filename*	Name of the file containing Dolby Vision metadata.
`<index_file_name>`	String	*filename*	Optionally, the name of an index file (if it was generated in an MXF parsing job).
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored input..
`<path>`	String	*folder_path*	Path to the directory that contains the input file (for example, `<path>path/to/media</path>`).

Input parameters for JPEG 2000 with sidecar metadata

Table 35: JPEG 2000 with sidecar metadata input parameters

Parameter	Type	Value	Description
`<j2k_sidecar>`	XML element		Parent element.
`<width>`	Integer		Input picture width.
`<height>`	Integer		Input picture height.

Table 35: JPEG 2000 with sidecar metadata input parameters (continued)

Parameter	Type	Value	Description
<frame_rate>	String	*23.976*, *24*, *25*, *29.97*, *30*, *59.94*, *60*	Input frame rate.
<j2k_prefix>	String		Allows you to use JPEG 2000 files with a specific prefix. If left empty (default), all JPEG 2000 files found in storage are used.
<md_file_name>	String	*filename*	Name of the file containing Dolby Vision metadata.
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored input.
<path>	String	*folder_path*	Path to the directory that contains the input file (for example, <path>path/to/media</path>).

Input parameters for Apple ProRes with sidecar metadata

Table 36: Apple ProRes with sidecar metadata input parameters

Parameter	Type	Value	Description
<prores_sidecar>	XML element		Parent element.
<codec>	String	apch, apcn, apcs, apco, ap4h, ap4x	The type of the Apple ProRes codec: • apch: Apple ProRes 422 HQ • apcn: Apple ProRes 422 • apcs: Apple ProRes 422 LT • apco: Apple ProRes 422 Proxy • ap4h: Apple ProRes 4444 • ap4x: Apple ProRes 4444 XQ The particular type of the Apple ProRes codec can be automatically obtained from the manifest file. (If you use such a file, see the information in this section).
<width>			Input picture width.
<height>			Input picture height.
<frame_rate>	String	23.976, 24, 25, 29.97, 30, 48, 50, 59.94, 60	Input picture frame rate.
<prores_prefix>	String		Allows you to use Apple ProRes files with a specific prefix. If left empty (default), all Apple ProRes files found in storage are used.
<md_file_name>	String	*filename*	Name of the file containing Dolby Vision metadata.

Table 36: Apple ProRes with sidecar metadata input parameters (continued)

Parameter	Type	Value	Description
`<prores_manifest>`	String	*filename*	Name of the optional manifest file containing details of the Apple ProRes frames (generated by the `extract_prores` filter). If the manifest is present, you do not have to provide the `<codec>`, `<width>`, `<height>`, and `<frame_rate>` parameters.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored input.
`<path>`	String	*folder_path*	Path to the directory that contains the input file (for example, `<path>path/to/media</path>`).

Input parameters for MOV with sidecar metadata

Table 37: MOV with sidecar metadata input parameters

Parameter	Type	Value	Description
`<mov_sidecar>`	XML element		Parent element.
`<override_frame_rate>`	String	`23.976`, `24`, `25`, `29.97`, `30`, `48`, `50`, `59.94`, `60`	If present, the frame rate stored in the XML metadata within the container will be ignored and replaced with this value.
`<track_id>`	Integer	Default: `-1`	ID of track to demultiplex. Negative value selects the first found video track.
`<mov_file_name>`	String	*filename*	Name of the MOV file containing video essence (Apple ProRes).
`<md_file_name>`	String	*filename*	Name of the file containing Dolby Vision metadata.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored input.
`<path>`	String	*folder_path*	Path to the directory that contains the input file (for example, `<path>path/to/media</path>`).

5.6.3 Filter parameters for Dolby Vision profile 8.1

When creating an XML file, you can use the following Dolby Vision profile 8.1 filter parameters.

The Dolby Vision profile 8.1 filter parameters are:

The table below describes the available parameters. The table does not include the parameters required by plugins, since those parameters depend on the plugins you choose to implement in Dolby Encoding Engine.

Table 38: Dolby Vision profile 8.1 filter parameters

Parameter	Type	Value	Description
<dv_mezz_to_dv_profile_8_1>	XML element		Parent element for the Dolby Vision profile 8.1 filter parameters.
<start>	Integer	Default: 0	Start frame or start timecode.
<end>	Integer	Default: 0	End frame or end timecode. (0=end of file)
<metadata_offset>	Integer	Default: 0	Number of frames to skip while reading metadata. Use only when video frames and metadata are not aligned.
<active_area_offset_top>	Integer	Default: 0	Top offset of the active area in the source content. Active area determines where in the encoded picture the actual video content is located. If the video is encoded with solid color bars around the content, setting the appropriate active area will exclude them from processing.
<active_area_offset_bottom>	Integer	Default: 0	Bottom offset of the active area in the source content. Active area determines where in the encoded picture the actual video content is located. If the video is encoded with solid color bars around the content, setting the appropriate active area will exclude them from processing.
<active_area_offset_left>	Integer	Default: 0	Left offset of the active area in the source content. Active area determines where in the encoded picture the actual video content is located. If the video is encoded with solid color bars around the content, setting the appropriate active area will exclude them from processing.
<active_area_offset_right>	Integer	Default: 0	Right offset of the active area in the source content. Active area determines where in the encoded picture the actual video content is located. If the video is encoded with solid color bars around the content, setting the appropriate active area will exclude them from processing.
<gx>	Integer	Values between -1 to 50000. Default: -1	Mastering display green x coordinate divided by 0.00002. The default negative value is ignored and forces filter to calculate actual value.
<gy>	Integer	Values between -1 to 50000. Default: -1	Mastering display green y coordinate divided by 0.00002. The default negative value is ignored and forces filter to calculate actual value.
<bx>	Integer	Values between -1 to 50000. Default: -1	Mastering display blue x coordinate divided by 0.00002. The default negative value is ignored and forces filter to calculate actual value.

Table 38: Dolby Vision profile 8.1 filter parameters (continued)

Parameter	Type	Value	Description
`<by>`	Integer	Values between -1 to 50000. Default: -1	Mastering display blue y coordinate divided by 0.00002. The default negative value is ignored and forces filter to calculate actual value.
`<rx>`	Integer	Values between -1 to 50000. Default: -1	Mastering display red x coordinate divided by 0.00002. The default negative value is ignored and forces filter to calculate actual value.
`<ry>`	Integer	Values between -1 to 50000. Default: -1	Mastering display red y coordinate divided by 0.00002. The default negative value is ignored and forces filter to calculate actual value.
`<wpx>`	Integer	Values between -1 to 50000. Default: -1	Mastering display white point x coordinate divided by 0.00002. The default negative value is ignored and forces filter to calculate actual value.
`<wpy>`	Integer	Values between -1 to 50000. Default: -1	Mastering display white point y coordinate divided by 0.00002. The default negative value is ignored and forces filter to calculate actual value.
`<max_cll>`	Integer	Values between -1 to 65535. Default: -1	Maximum content light level in units of 1 nit. The default negative value is ignored and forces filter to calculate actual value.
`<max_fall>`	Integer	Values between -1 to 65535. Default: -1	Maximum frame average light level in units of 1 nit. The default negative value is ignored and forces filter to calculate actual value.
`<decoder_parallel_frames>`	Integer	Values between 0 and 16. Default: 0	Number of parallel video essence pictures to process. Higher values can reduce processing time at the cost of increased CPU and memory usage. The default 0 value sets the parameter to auto. Depending on which decoder you select, the auto value means: • For the JPEG 2000 Base decoder, four decoder instatnces run in parallel. • For the Kakadu decoder, two multi-threaded instances run in parallel. • For the Apple ProRes decoder, the process uses all available cores.
`<prepend_frames>`	Integer	Default: 0	Number of black frames prepending the target content.
`<target_width>`	Integer list	Accepts multiple integer values, separated by a space. Default: 0	Width of target picture (0 = use source width).

Table 38: Dolby Vision profile 8.1 filter parameters (continued)

Parameter	Type	Value	Description
<target_height>	Integer list	Accepts multiple integer values, separated by a space. Default: 0	Height of target picture (0 = use source height).
<max_scene_frames>	Integer	Default: 256	The maximum number of frames in a buffer within the Dolby Vision processor. It is recommended to use a number that is a multiple of the GOP size (e.g. for GOP Size=48, max_scene_frames=240) and not smaller than GOP size. This property has significant impact on RAM consumption during the Dolby Vision processor execution.
<data_rate>	Integer list	Accepts multiple integer values, separated by a space. Default: 15000	Average data rate in kbps.
<max_vbv_data_rate>	Integer list	Accepts multiple integer values, separated by a space. Default: 15000	Maximum video buffering verifier data rate in kbps.
<vbv_buffer_size>	Integer list	Accepts multiple integer values, separated by a space. Default: 30000	Video buffering verifier buffer size in kb.
<encode_pass_num>	Integer	Default: 1	Number of encoder passes. *
<disable_md_postproc>	Boolean	Default: false	Disables Dolby Vision metadata post-processing.
<decoder>	XML element	<j2k_dec> or <prores_dec>	Specifies the input decoder.
<scaling>	XML element	Example: <base>	Specifies the scaling filter plugin. The base implementation is delivered with Dolby Encoding Engine.
<source_crop_top>	Integer	Default: 0	A positive value indicates how many pixels should be removed from the top of the source picture before scaling; a negative value will pad the top with black pixels.
<source_crop_bottom>	Integer	Default: 0	A positive value indicates how many pixels should be removed from the bottom of the source picture before scaling; a negative value will pad the bottom with black pixels.

Table 38: Dolby Vision profile 8.1 filter parameters (continued)

Parameter	Type	Value	Description
<source_crop_left>	Integer	Default: 0	A positive value indicates how many pixels should be removed from the left side of the source picture before scaling; a negative value will pad the left side with black pixels.
<source_crop_right>	Integer	Default: 0	A positive value indicates how many pixels should be removed from the right side of the source picture before scaling; a negative value will pad the right side with black pixels.
<noise>	XML element	Example: <base>	Specifies the noise reduction filter plugin. The base implementation is delivered with Dolby Encoding Engine.
<strength>	Integer	Default: 0	In the case of the <base> noise filter, the only currently supported value is 0 (noise reduction turned off).
<hevc_enc>	XML element	Example: <impact>	Specifies the HEVC encoder plugin.

* An encoder pass is a single instance of processing a mezzanine YUV file.

5.6.4 Output parameters for Dolby Vision profile 8.1

When creating an XML file, you can use Dolby Vision profile 8.1 output parameters.

Output parameters for HEVC

Table 39: HEVC output parameters

Parameter	Type	Value	Description
<hevc>	XML element		Parent element for HEVC elementary stream output.
<file_name>	String	*filename*	Output file name.
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored output.
<path>	String	*Storage folder path*	Path to the directory to save the output file (for example, <path>path/to/media</path>).

Output parameters for MP4

Table 40: MP4 output parameters

Parameter	Type	Value	Description
<mp4>	XML element		Parent element for an MP4 output file.
<output_format>	String	mp4, dash	Defines whether the output is an MP4 or DASH file.
<min_frag_duration>	Integer		Minimum fragment duration, in milliseconds.

Table 40: MP4 output parameters (continued)

Parameter	Type	Value	Description
<max_frag_duration>	Integer		Maximum fragment duration, in milliseconds.
<mux_flag>	String	encrypt_style_piff, write_iods, write_pdin, write_bloc, write_ainf, write_ctts_v1, write_subs_v1	Sets specified multiplexer flag.
<frag_flag>	String	frag_style_default, frag_style_cff, write_tfdt, write_sdtp, write_trik, write_sidx, write_mfra, force_tfra, no_base_data_offset, empty_trex, empty_tfhd, one_tfra_per_traf, default_base_is_moof	Sets specified fragmentation flag.
<track_flag>	String	track_enabled, track_in_movie, track_in_preview, track_in_poster	Sets specified track flag.
<file_name>	String	*filename*	Output file name.
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored output.
<path>	String	*Storage folder path*	Path to the directory to save the output file (for example, <path>path/to/media</path>).

Output parameters for transport stream

Table 41: Transport stream output parameters

Parameter	Type	Value	Description
<ts>	XML element		Parent element for a transport stream output file.
<packetize_standard>	String	mpeg, atsc, dvb Default: mpeg	Indicates the packetize standard used for substreams.
<packetize_mode>	String	cbr, vbr Default: vbr	Indicates constant bit rate (CBR) or variable bit rate (VBR) of the substreams.
<transport_stream_id>	Integer	Default: 0	Transport stream ID.
<audio_pid>	Integer	32–8186 Default: 40	Packet ID of the audio stream.
<video_pid>	Integer	32–8186 Default: 50	Packet ID of the video stream.
<file_name>	String	*filename*	Output file name.

Table 41: Transport stream output parameters (continued)

Parameter	Type	Value	Description
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored output.
`<path>`	String	*`Storage folder path`*	Path to the directory to save the output file (for example, `<path>path/to/media</path>`).
`<plugin>`	XML element	Depends on the selected plug-in. Example: `<base>`	Allows you to choose the transport stream multiplexing plug-in. The base implementation is delivered with Dolby Encoding Engine.

5.7 HDR10 video encoding job

The Dolby Vision encoder can produce HDR10 video streams according to the parameters you specify in the job configuration XML file.

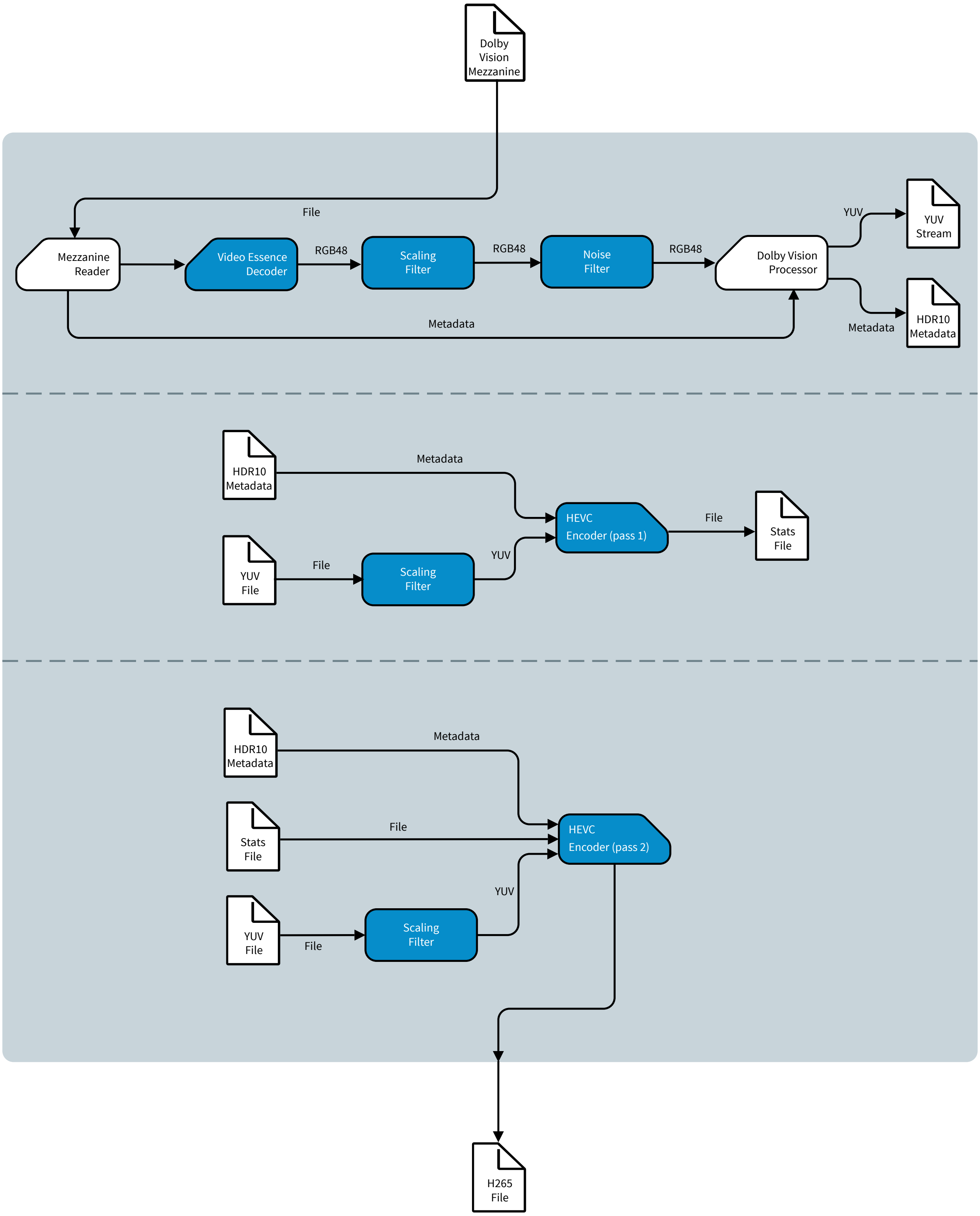
This encoding job allows you to create HDR10 video bitstreams from Dolby Vision mezzanine input.

The Dolby Encoding Engine provides two content mapping variations of the HDR10 video:

No mapping
: Content mapping is not applied to the source to create the HDR10 video, which allows the output to retain similar nit levels as the source.

Mapping to 1,000 or 600 nits
: Applies content mapping to map the source to 1,000 or 600 nits (peak).

Figure 4: HDR10 video encode process



The input source can be MXF, MXF with sidecar Dolby Vision XML metadata, JPEG 2000 with sidecar Dolby Vision XML metadata, Apple ProRes with sidecar Dolby Vision XML metadata, and MOV with sidecar Dolby Vision XML metadata (with Apple ProRes essence).

The output is an HEVC video stream. The output can also be multiplexed into MP4 or transport stream containers.

The components marked in blue in the preceding figure are replaceable plug-ins. A standard installation of Dolby Encoding Engine includes default plug-in implementations for JPEG 2000 decoding, scaling, and noise reduction. The Dolby Encoding Engine allows you to replace these components with others that perform the same processing functions.

Note: The Dolby Encoding Engine does not include an HEVC encoder plugin but requires such an encoder to be built and installed. For more information, refer to the *Dolby Encoding Engine HEVC Encoder Installation Guide*.

5.7.1 XML example files for HDR10 video

The XML template files are examples of a job configuration.

You can use them to create job configurations. You can find them in the xml_templates folder. For the HDR10 video encoding job, refer to the following files:

- mxf_dv_mezz_to_hdr10_hevc.xml
- mxf_dv_mezz_to_hdr10_mp4.xml
- mxf_sidecar_dv_mezz_to_hdr10_hevc.xml
- mxf_sidecar_dv_mezz_to_hdr10_mp4.xml
- j2k_sidecar_dv_mezz_to_hdr10_hevc.xml
- j2k_sidecar_dv_mezz_to_hdr10_mp4.xml

5.7.2 Input parameters for HDR10 video

When creating an XML file, you can use HDR10 video input parameters.

The supported input file formats are:

- MXF (with JPEG 2000 essence)
- MXF with sidecar Dolby Vision XML metadata (with JPEG 2000 essence)
- JPEG 2000 with sidecar Dolby Vision XML metadata
- Apple ProRes with sidecar Dolby Vision XML metadata
- MOV with sidecar Dolby Vision XML metadata (with Apple ProRes essence)

Input parameters for MXF

Table 42: MXF input parameters

Parameter	Type	Value	Description
<mxf>	XML element		Parent element.
<override_frame_rate>	String	23.976, 24, 25, 29.97, 30, 48, 50, 59.94, 60	If present, the frame rate stored in the XML metadata within the container will be ignored and replaced with this value.
<file_name>	String	*filename*	
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.

Table 42: MXF input parameters (continued)

Parameter	Type	Value	Description
`<local>`	XML element		Parent element for parameters of locally stored input.
`<path>`	String	*folder_path*	Path to the directory that contains the input file (for example, `<path>path/to/media</path>`).

Input parameters for MXF with sidecar metadata

Table 43: MXF with sidecar metadata input parameters

Parameter	Type	Value	Description
`<mxf_sidecar>`	XML element		Parent element.
`<override_frame_rate>`	String	`23.976`, `24`, `25`, `29.97`, `30`, `48`, `50`, `59.94`, `60`	If present, the frame rate stored in the XML metadata within the container will be ignored and replaced with this value.
`<mxf_file_name>`	String	*filename*	Name of the MXF file containing video essence (JPEG 2000).
`<md_file_name>`	String	*filename*	Name of the file containing Dolby Vision metadata.
`<index_file_name>`	String	*filename*	Optionally, the name of an index file (if it was generated in an MXF parsing job).
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored input..
`<path>`	String	*folder_path*	Path to the directory that contains the input file (for example, `<path>path/to/media</path>`).

Input parameters for JPEG 2000 with sidecar metadata

Table 44: JPEG 2000 with sidecar metadata input parameters

Parameter	Type	Value	Description
`<j2k_sidecar>`	XML element		Parent element.
`<width>`	Integer		Input picture width.
`<height>`	Integer		Input picture height.
`<frame_rate>`	String	*23.976*, *24*, *25*, *29.97*, *30*, *59.94*, *60*	Input frame rate.
`<j2k_prefix>`	String		Allows you to use JPEG 2000 files with a specific prefix. If left empty (default), all JPEG 2000 files found in storage are used.
`<md_file_name>`	String	*filename*	Name of the file containing Dolby Vision metadata.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.

Table 44: JPEG 2000 with sidecar metadata input parameters (continued)

Parameter	Type	Value	Description
`<local>`	XML element		Parent element for parameters of locally stored input.
`<path>`	String	*folder_path*	Path to the directory that contains the input file (for example, `<path>path/to/media</path>`).

Input parameters for Apple ProRes with sidecar metadata

Table 45: Apple ProRes with sidecar metadata input parameters

Parameter	Type	Value	Description
`<prores_sidecar>`	XML element		Parent element.
`<codec>`	String	apch, apcn, apcs, apco, ap4h, ap4x	The type of the Apple ProRes codec: • apch: Apple ProRes 422 HQ • apcn: Apple ProRes 422 • apcs: Apple ProRes 422 LT • apco: Apple ProRes 422 Proxy • ap4h: Apple ProRes 4444 • ap4x: Apple ProRes 4444 XQ The particular type of the Apple ProRes codec can be automatically obtained from the manifest file. (If you use such a file, see the information in this section).
`<width>`			Input picture width.
`<height>`			Input picture height.
`<frame_rate>`	String	`23.976`, `24`, `25`, `29.97`, `30`, `48`, `50`, `59.94`, `60`	Input picture frame rate.
`<prores_prefix>`	String		Allows you to use Apple ProRes files with a specific prefix. If left empty (default), all Apple ProRes files found in storage are used.
`<md_file_name>`	String	*filename*	Name of the file containing Dolby Vision metadata.
`<prores_manifest>`	String	*filename*	Name of the optional manifest file containing details of the Apple ProRes frames (generated by the `extract_prores` filter). If the manifest is present, you do not have to provide the `<codec>`, `<width>`, `<height>`, and `<frame_rate>` parameters.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored input.
`<path>`	String	*folder_path*	Path to the directory that contains the input file (for example, `<path>path/to/media</path>`).

Input parameters for MOV with sidecar metadata

Table 46: MOV with sidecar metadata input parameters

Parameter	Type	Value	Description
`<mov_sidecar>`	XML element		Parent element.
`<override_frame_rate>`	String	`23.976`, `24`, `25`, `29.97`, `30`, `48`, `50`, `59.94`, `60`	If present, the frame rate stored in the XML metadata within the container will be ignored and replaced with this value.
`<track_id>`	Integer	Default: -1	ID of track to demultiplex. Negative value selects the first found video track.
`<mov_file_name>`	String	*filename*	Name of the MOV file containing video essence (Apple ProRes).
`<md_file_name>`	String	*filename*	Name of the file containing Dolby Vision metadata.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored input.
`<path>`	String	*folder_path*	Path to the directory that contains the input file (for example, `<path>path/to/media</path>`).

5.7.3 Filter parameters for HDR10 video

When creating an XML file, you can use the following HDR10 video filter parameters.

The table below describes the available parameters. The table does not include the parameters required by plugins, since those parameters depend on the plugins you choose to implement in Dolby Encoding Engine.

Table 47: HDR10 filter parameters

Parameter	Type	Value	Description
`<dv_mezz_to_hdr10>`	XML element		Parent element for the HDR10 filter parameters.
`<start>`	Integer	Default: 0	Start frame or start timecode.
`<end>`	Integer	Default: 0	End frame or end timecode. (0=end of file)
`<metadata_offset>`	Integer	Default: 0	Number of frames to skip while reading metadata. Use only when video frames and metadata are not aligned.
`<active_area_offset_top>`	Integer	Default: 0	Top offset of the active area in the source content. Active area determines where in the encoded picture the actual video content is located. If the video is encoded with solid color bars around the content, setting the appropriate active area will exclude them from processing.

Table 47: HDR10 filter parameters (continued)

Parameter	Type	Value	Description
`<active_area_offset_bottom>`	Integer	Default: 0	Bottom offset of the active area in the source content. Active area determines where in the encoded picture the actual video content is located. If the video is encoded with solid color bars around the content, setting the appropriate active area will exclude them from processing.
`<active_area_offset_left>`	Integer	Default: 0	Left offset of the active area in the source content. Active area determines where in the encoded picture the actual video content is located. If the video is encoded with solid color bars around the content, setting the appropriate active area will exclude them from processing.
`<active_area_offset_right>`	Integer	Default: 0	Right offset of the active area in the source content. Active area determines where in the encoded picture the actual video content is located. If the video is encoded with solid color bars around the content, setting the appropriate active area will exclude them from processing.
`<content_mapping>`	String	600, 1000, true, false Default: false	Applies content mapping to map the peak luminance to 600 or 1000 nits. The `true` value is equal to the `1000` value and is supported for backward compatibility.
`<gx>`	Integer	Values between -1 to 50000. Default: -1	Mastering display green x coordinate divided by 0.00002. The default negative value is ignored and forces filter to calculate actual value.
`<gy>`	Integer	Values between -1 to 50000. Default: -1	Mastering display green y coordinate divided by 0.00002. The default negative value is ignored and forces filter to calculate actual value.
`<bx>`	Integer	Values between -1 to 50000. Default: -1	Mastering display blue x coordinate divided by 0.00002. The default negative value is ignored and forces filter to calculate actual value.
`<by>`	Integer	Values between -1 to 50000. Default: -1	Mastering display blue y coordinate divided by 0.00002. The default negative value is ignored and forces filter to calculate actual value.

Table 47: HDR10 filter parameters (continued)

Parameter	Type	Value	Description
`<rx>`	Integer	Values between -1 to 50000. Default: -1	Mastering display red x coordinate divided by 0.00002. The default negative value is ignored and forces filter to calculate actual value.
`<ry>`	Integer	Values between -1 to 50000. Default: -1	Mastering display red y coordinate divided by 0.00002. The default negative value is ignored and forces filter to calculate actual value.
`<wpx>`	Integer	Values between -1 to 50000. Default: -1	Mastering display white point x coordinate divided by 0.00002. The default negative value is ignored and forces filter to calculate actual value.
`<wpy>`	Integer	Values between -1 to 50000. Default: -1	Mastering display white point y coordinate divided by 0.00002. The default negative value is ignored and forces filter to calculate actual value.
`<max_peak_lum>`	Integer	Values between -1 to 2000000000. Default: -1	Maximum mastering display luminance in units of 0.0001 nit. The default negative value is ignored and forces filter to calculate actual value.
`<min_peak_lum>`	Integer	Values between -1 to 2000000000. Default: -1	Minimum mastering display luminance in units of 0.0001 nit. The default negative value is ignored and forces filter to calculate actual value.
`<max_cll>`	Integer	Values between -1 to 65535. Default: -1	Maximum content light level in units of 1 nit. The default negative value is ignored and forces filter to calculate actual value.
`<max_fall>`	Integer	Values between -1 to 65535. Default: -1	Maximum frame average light level in units of 1 nit. The default negative value is ignored and forces filter to calculate actual value.

Table 47: HDR10 filter parameters (continued)

Parameter	Type	Value	Description
`<decoder_parallel_frames>`	Integer	Values between 0 and 16. Default: 0	Number of parallel video essence pictures to process. Higher values can reduce processing time at the cost of increased CPU and memory usage. The default 0 value sets the parameter to auto. Depending on which decoder you select, the auto value means: • For the JPEG 2000 Base decoder, four decoder instatnces run in parallel. • For the Kakadu decoder, two multi-threaded instances run in parallel. • For the Apple ProRes decoder, the process uses all available cores.
`<prepend_frames>`	Integer	Default: 0	Number of black frames prepending the target content.
`<target_width>`	Integer list	Accepts multiple integer values, separated by a space. Default: 0	Width of target picture (0 = use source width).
`<target_height>`	Integer list	Accepts multiple integer values, separated by a space. Default: 0	Height of target picture (0 = use source height).
`<data_rate>`	Integer list	Accepts multiple integer values, separated by a space. Default: 15000	Average data rate in kbps.
`<max_vbv_data_rate>`	Integer list	Accepts multiple integer values, separated by a space. Default: 15000	Maximum video buffering verifier data rate in kbps.
`<vbv_buffer_size>`	Integer list	Accepts multiple integer values, separated by a space. Default: 30000	Video buffering verifier buffer size in kb.
`<encode_pass_num>`	Integer	Default: 1	Number of encoder passes. *
`<decoder>`	XML element	`<j2k_dec>` or `<prores_dec>`	Specifies the input decoder.
`<scaling>`	XML element	Example: `<base>`	Specifies the scaling filter plugin. The base implementation is delivered with Dolby Encoding Engine.

Table 47: HDR10 filter parameters (continued)

Parameter	Type	Value	Description
`<source_crop_top>`	Integer	Default: 0	A positive value indicates how many pixels should be removed from the top of the source picture before scaling; a negative value will pad the top with black pixels.
`<source_crop_bottom>`	Integer	Default: 0	A positive value indicates how many pixels should be removed from the bottom of the source picture before scaling; a negative value will pad the bottom with black pixels.
`<source_crop_left>`	Integer	Default: 0	A positive value indicates how many pixels should be removed from the left side of the source picture before scaling; a negative value will pad the left side with black pixels.
`<source_crop_right>`	Integer	Default: 0	A positive value indicates how many pixels should be removed from the right side of the source picture before scaling; a negative value will pad the right side with black pixels.
`<noise>`	XML element	Example: <base>	Specifies the noise reduction filter plugin. The base implementation is delivered with Dolby Encoding Engine.
`<strength>`	Integer	Default: 0	In the case of the <base> noise filter, the only currently supported value is 0 (noise reduction turned off).
`<hevc_enc>`	XML element	Depends on the selected plugin. Example: `<impact>`	Specifies the HEVC encoder plugin.

[*] An encoder pass is a single instance of processing a mezzanine YUV file.

5.7.4 Output parameters for HDR10 video

When creating an XML file, you can use HDR10 video output parameters.

Output parameters for HEVC

Table 48: HEVC output parameters

Parameter	Type	Value	Description
`<hevc>`	XML element		Parent element for HEVC elementary stream output.
`<file_name>`	String	*filename*	Output file name.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.

Table 48: HEVC output parameters (continued)

Parameter	Type	Value	Description
<local>	XML element		Parent element for parameters of locally stored output.
<path>	String	*Storage folder path*	Path to the directory to save the output file (for example, <path>path/to/media</path>).

Output parameters for MP4

Table 49: MP4 output parameters

Parameter	Type	Value	Description
<mp4>	XML element		Parent element for an MP4 output file.
<output_format>	String	mp4, dash	Defines whether the output is an MP4 or DASH file.
<min_frag_duration>	Integer		Minimum fragment duration, in milliseconds.
<max_frag_duration>	Integer		Maximum fragment duration, in milliseconds.
<mux_flag>	String	encrypt_style_piff, write_iods, write_pdin, write_bloc, write_ainf, write_ctts_v1, write_subs_v1	Sets specified multiplexer flag.
<frag_flag>	String	frag_style_default, frag_style_cff, write_tfdt, write_sdtp, write_trik, write_sidx, write_mfra, force_tfra, no_base_data_offset, empty_trex, empty_tfhd, one_tfra_per_traf, default_base_is_moof	Sets specified fragmentation flag.
<track_flag>	String	track_enabled, track_in_movie, track_in_preview, track_in_poster	Sets specified track flag.
<file_name>	String	*filename*	Output file name.
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored output.
<path>	String	*Storage folder path*	Path to the directory to save the output file (for example, <path>path/to/media</path>).

Output parameters for transport stream

Table 50: Transport stream output parameters

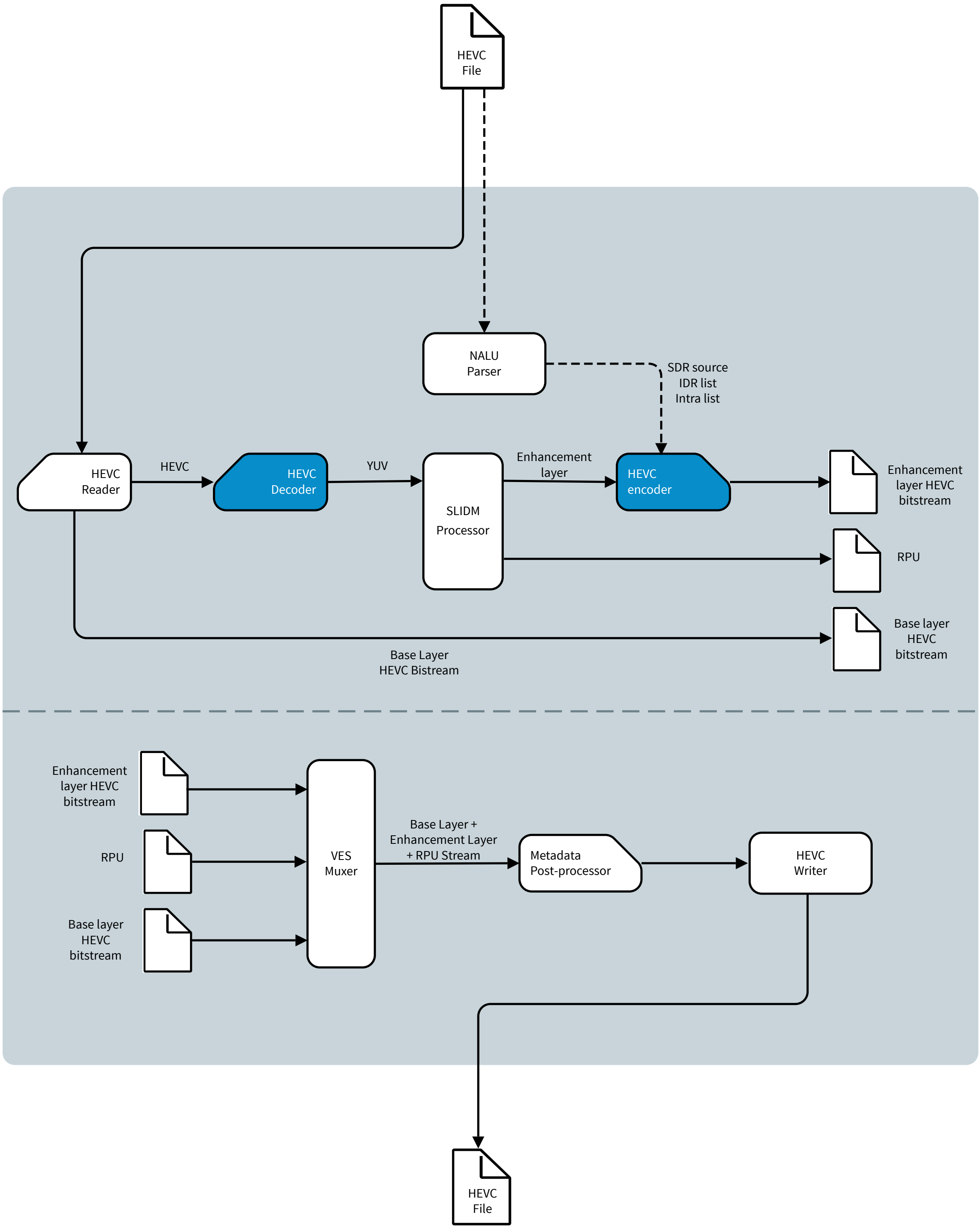
Parameter	Type	Value	Description
`<ts>`	XML element		Parent element for a transport stream output file.
`<packetize_standard>`	String	`mpeg`, `atsc`, `dvb` Default: `mpeg`	Indicates the packetize standard used for substreams.
`<packetize_mode>`	String	`cbr`, `vbr` Default: `vbr`	Indicates constant bit rate (CBR) or variable bit rate (VBR) of the substreams.
`<transport_stream_id>`	Integer	Default: `0`	Transport stream ID.
`<audio_pid>`	Integer	`32–8186` Default: `40`	Packet ID of the audio stream.
`<video_pid>`	Integer	`32–8186` Default: `50`	Packet ID of the video stream.
`<file_name>`	String	*`filename`*	Output file name.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored output.
`<path>`	String	*`Storage folder path`*	Path to the directory to save the output file (for example, `<path>path/to/media</path>`).
`<plugin>`	XML element	Depends on the selected plug-in. Example: `<base>`	Allows you to choose the transport stream multiplexing plug-in. The base implementation is delivered with Dolby Encoding Engine.

5.8 SDR to Dolby Vision profile 4 encoding job

The Dolby Vision encoder can produce Dolby Vision video streams, according to the parameters you specify in the job configuration XML file.

An standard dynamic range (SDR) to Dolby Vision profile 4 conversion enables the delivery of an image encoded from an SDR elementary stream and approximate a representation in Dolby Vision without a Dolby Vision source.

Figure 5: SDR to Dolby Vision profile 4 encode process



The input source is an HEVC that complies with the Dolby Vision requirements (also in the form of MP4 or transport stream containers). You can use the HEVC transcoding job to create HEVC streams that comply with the SDR source requirements for Dolby Vision.

The output is an HEVC video stream. The output can also be multiplexed into MP4 or transport stream containers.

As a part of the encoding job, the metadata postprocessor updates Dolby Vision metadata within the RPU after video elementary stream multiplexing. The postprocessor smooths out related metadata for scenes and recalculates maximum values from all frames within the content.

A standard installation of Dolby Encoding Engine includes default plug-in implementations for HEVC decoding and HEVC encoding. The Dolby Encoding Engine allows you to replace these components with others that perform the same processing functions.

Note: The Dolby Encoding Engine does not include an HEVC encoder plugin but requires such an encoder to be built and installed. For more information, refer to the *Dolby Encoding Engine HEVC Encoder Installation Guide*.

5.8.1 XML example files for SDR to Dolby Vision profile 4 conversion

The XML template files are examples of a job configuration.

You can use them to create job configurations. You can find them in the xml_templates folder. For the SDR to Dolby Vision profile 4 encoding job, refer to the following files:

- hevc_sdr_to_dv_profile_4_2_hevc.xml
- hevc_sdr_to_dv_profile_4_2_mp4.xml

5.8.2 Input parameters for SDR to Dolby Vision profile 4 conversion

When creating an XML file, you can use the following SDR to Dolby Vision profile 4 conversion input parameters.

The supported input file formats is an HEVC file that complies with the Dolby Vision requirements. The input can also be an MP4 or transport stream container with an HEVC video stream.

The input HEVC stream cannot contain more than six reference frames.

Input parameters for HEVC

Table 51: Input parameters for HEVC

Parameter	Type	Value	Description
<hevc>	XML element		Parent element.
<frame_rate>	String	23.976, 24, 25, 29.97, 30, 48, 50, 59.94, 60	Video frame rate. Required in use cases with HEVC decoding.
<dv_profile>	String	none (default)	Dolby Vision profile. For Dolby Vision profile 8.2 encoding, use the default value.
<file_name>	String	*filename*	
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored input.
<path>	String	*folder_path*	Path to the directory that contains the input file (for example, <path>path/to/media</path>).

Input parameters for MP4

Table 52: MP4 input parameters

Parameter	Type	Value	Description
`<mp4>`	XML element		Parent element for an MP4 input file.
`<track_id>`	Integer	Default: -1	ID of track to demultiplex. Negative value selects the first found video track.
`<override_frame_rate>`	String	`23.976`, `24`, `25`, `29.97`, `30`, `48`, `50`, `59.94`, `60`	If present, the frame rate stored in the XML metadata within the container will be ignored and replaced with this value.
`<file_name>`	String	*`filename`*	Output file name.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored output.
`<path>`	String	*`folder_path`*	Path to the directory that contains the input file (for example, `<path>path/to/media</path>`).

Input parameters for transport stream

Table 53: Transport stream input parameters

Parameter	Type	Value	Description
`<ts>`	XML element		Parent element for an transport stream input file.
`<pid>`	Integer	Default: -1	Packet ID of the stream to demultiplex. Negative value selects first found video stream.
`<override_frame_rate>`	String	`23.976`, `24`, `25`, `29.97`, `30`, `48`, `50`, `59.94`, `60`	If present, the frame rate stored in the XML metadata within the container will be ignored and replaced with this value.
`<file_name>`	String	*`filename`*	Output file name.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.

Table 53: Transport stream input parameters (continued)

Parameter	Type	Value	Description
`<local>`	XML element		Parent element for parameters of locally stored output.
`<path>`	String	*folder_path*	Path to the directory that contains the input file (for example, `<path>path/to/media</path>`).

5.8.3 Filter parameters for SDR to Dolby Vision profile 4 conversion

When creating an XML file, you can use SDR to Dolby Vision profile 4 conversion filter parameters.

The following table describes the available parameters. The table does not include the parameters required by plug-ins, because those parameters depend on the plug-ins you choose to implement in the Dolby Encoding Engine.

Table 54: SDR to Dolby Vision profile 4 conversion filter parameters

Parameter	Type	Value	Description
`<sdr_to_dv_profile_4_2>`	XML element		Parent element for the Dolby Vision profile 4.2 filter parameters.
`<el_data_rate>`	string	Default: 15000	Enhancement layer average data rate, in kbps.
`<el_max_vbv_data_rate>`	String	Default: 15000	Enhancement layer maximum video buffering verifier data rate, in kbps.
`<el_vbv_buffer_size>`	String	Default: 30000	Enhancement layer video buffering verifier buffer size, in kb.
`<encode_pass_num>`	Integer	Default: 1	Number of encoder passes. *
`<max_light_level>`	String	200, 4000 Default: 4000	Target nits maximum value (4,000 or 200).You can lower the target max nits to 200, if 4,000 amplifies noise levels to an undesirable degree in playback.
`<run_mode>`	String	linear_dynamic, singlenode Default: linear_dynamic	Allows you to choose the display mapping algorithm: single node (simple algorithm, faster) or linear dynamic (improved debanding algorithm, slower).
`<disable_md_postproc>`	Boolean	Default: false	Disables Dolby Vision metadata postprocessing.
`<disable_md_conform>`	Boolean	Default: false	Disables the Dolby Vision L1 metadata conformance check that normally runs at the end of processing.

Table 54: SDR to Dolby Vision profile 4 conversion filter parameters (continued)

Parameter	Type	Value	Description
<hevc_dec>	XML element	Example: <impact>	Specifies the HEVC decoder plug-in.
<hevc_enc>	XML element	Example: <impact>	Specifies the HEVC encoder plug-in.

* An encoder pass is a single instance of processing a mezzanine YUV file.

5.8.4 Output parameters for SDR to Dolby Vision profile 4 conversion

When creating an XML file, you can use several output parameters and values.

Output parameters for HEVC

Table 55: HEVC output parameters

Parameter	Type	Value	Description
<hevc>	XML element		Parent element for HEVC elementary stream output.
<file_name>	String	*filename*	Output file name.
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored output.
<path>	String	*Storage folder path*	Path to the directory to save the output file (for example, <path>path/to/media</path>).

Output parameters for MP4

Table 56: MP4 output parameters

Parameter	Type	Value	Description
<mp4>	XML element		Parent element for an MP4 output file.
<output_format>	String	mp4, dash	Defines whether the output is an MP4 or DASH file.
<min_frag_duration>	Integer		Minimum fragment duration, in milliseconds.
<max_frag_duration>	Integer		Maximum fragment duration, in milliseconds.
<mux_flag>	String	encrypt_style_piff, write_iods, write_pdin, write_bloc, write_ainf, write_ctts_v1, write_subs_v1	Sets specified multiplexer flag.
<frag_flag>	String	frag_style_default, frag_style_cff, write_tfdt, write_sdtp, write_trik, write_sidx, write_mfra, force_tfra, no_base_data_offset, empty_trex, empty_tfhd, one_tfra_per_traf, default_base_is_moof	Sets specified fragmentation flag.

Table 56: MP4 output parameters (continued)

Parameter	Type	Value	Description
<track_flag>	String	track_enabled, track_in_movie, track_in_preview, track_in_poster	Sets specified track flag.
<file_name>	String	*filename*	Output file name.
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored output.
<path>	String	*Storage folder path*	Path to the directory to save the output file (for example, <path>path/to/media</path>).

Output parameters for transport stream

Table 57: Transport stream output parameters

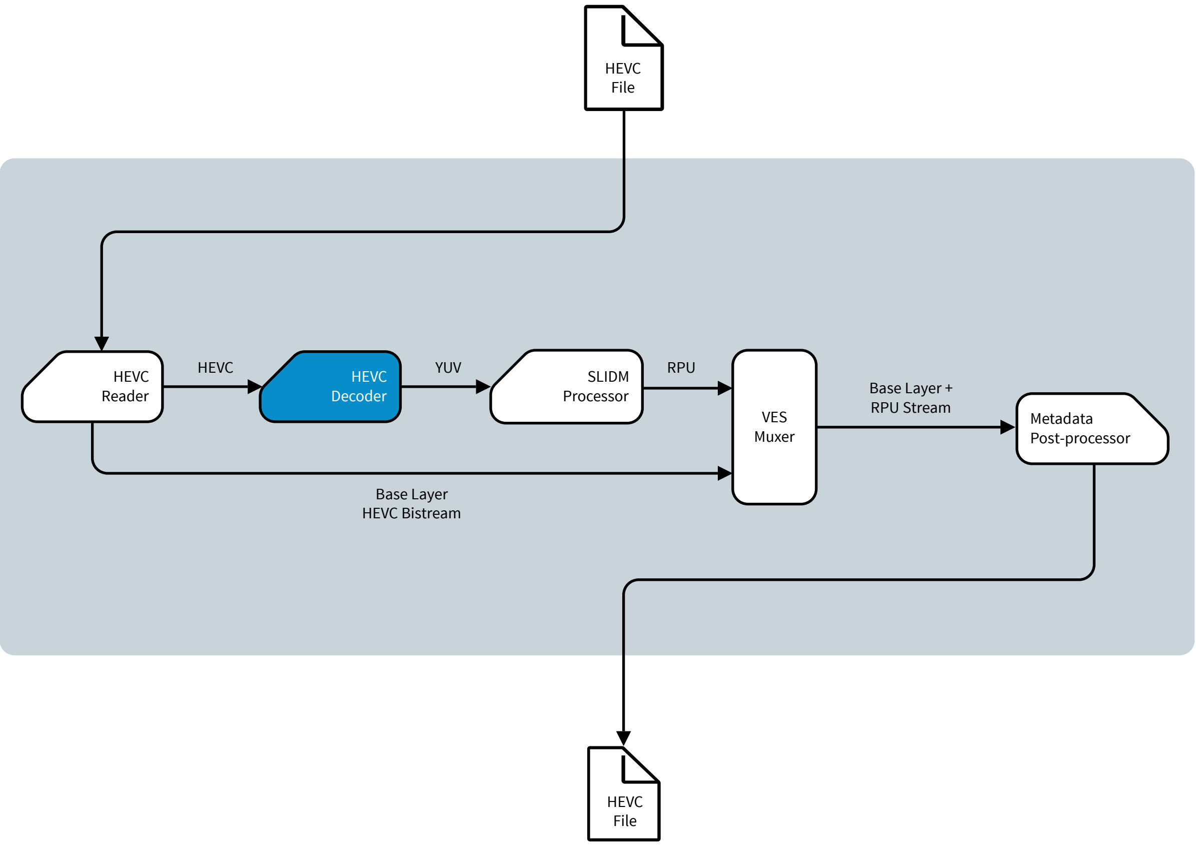
Parameter	Type	Value	Description
<ts>	XML element		Parent element for a transport stream output file.
<packetize_standard>	String	mpeg, atsc, dvb Default: mpeg	Indicates the packetize standard used for substreams.
<packetize_mode>	String	cbr, vbr Default: vbr	Indicates constant bit rate (CBR) or variable bit rate (VBR) of the substreams.
<transport_stream_id>	Integer	Default: 0	Transport stream ID.
<audio_pid>	Integer	32–8186 Default: 40	Packet ID of the audio stream.
<video_pid>	Integer	32–8186 Default: 50	Packet ID of the video stream.
<file_name>	String	*filename*	Output file name.
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored output.
<path>	String	*Storage folder path*	Path to the directory to save the output file (for example, <path>path/to/media</path>).
<plugin>	XML element	Depends on the selected plug-in. Example: <base>	Allows you to choose the transport stream multiplexing plug-in. The base implementation is delivered with Dolby Encoding Engine.

5.9 SDR to Dolby Vision profile 8 encoding job

The Dolby Vision encoder can produce Dolby Vision video streams, according to the parameters you specify in the job configuration XML file.

An SDR to Dolby Vision profile 8 conversion enables the delivery of an image encoded from an SDR elementary stream and approximate a representation in Dolby Vision without a Dolby Vision source.

Figure 6: SDR to Dolby Vision profile 8 encode process



The input source is an HEVC stream that complies with the SDR source requirements for Dolby Vision (also in the form of MP4 or transport stream containers). You can use the HEVC transcoding job to create HEVC streams that comply with the SDR source requirements for Dolby Vision.

The output is an HEVC stream. The output can also be multiplexed into MP4 or transport stream containers.

As a part of the encoding job, the metadata postprocessor updates Dolby Vision metadata within the RPU after video elementary stream multiplexing. The postprocessor smooths out related metadata for scenes and recalculates maximum values from all frames within the content.

A standard installation of Dolby Encoding Engine includes a default plug-in implementation for HEVC decoding. The Dolby Encoding Engine allows you to replace this component with others that perform the same processing functions.

Related information

HEVC transcoding job on page 97

Input requirements for SDR to Dolby Vision encoding on page 131

5.9.1 XML example files for SDR to Dolby Vision profile 8 conversion

The XML template files are examples of a job configuration.

You can use them to create job configurations. You can find them in the xml_templates folder. For the SDR to Dolby Vision profile 8 encoding job, refer to the following files:

- hevc_sdr_to_dv_profile_8_2_hevc.xml
- hevc_sdr_to_dv_profile_8_2_mp4.xml

5.9.2 Input parameters for SDR to Dolby Vision profile 8 conversion

When creating an XML file, you can use input parameters for SDR to Dolby Vision profile 8 conversion.

The supported input file formats is an HEVC file that complies with the Dolby Vision requirements. The input can also be an MP4 or transport stream container with an HEVC video stream.

The input HEVC stream cannot contain more than six reference frames.

Input parameters for HEVC

Table 58: HEVC input parameters

Parameter	Type	Value	Description
<hevc>	XML element		Parent element.
<frame_rate>	String	23.976, 24, 25, 29.97, 30, 48, 50, 59.94, 60	Video frame rate. Required in use cases with HEVC decoding.
<dv_profile>	String	none (default)	Dolby Vision profile. For Dolby Vision profile 8.2 encoding, use the default value.
<file_name>	String	*filename*	
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored input.
<path>	String	*folder_path*	Path to the directory that contains the input file (for example, <path>path/to/media</path>).

Input parameters for MP4

Table 59: MP4 input parameters

Parameter	Type	Value	Description
<mp4>	XML element		Parent element for an MP4 input file.
<track_id>	Integer	Default: -1	ID of track to demultiplex. Negative value selects the first found video track.

Table 59: MP4 input parameters (continued)

Parameter	Type	Value	Description
`<override_frame_rate>`	String	`23.976`, `24`, `25`, `29.97`, `30`, `48`, `50`, `59.94`, `60`	If present, the frame rate stored in the XML metadata within the container will be ignored and replaced with this value.
`<file_name>`	String	*filename*	Output file name.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored output.
`<path>`	String	*folder_path*	Path to the directory that contains the input file (for example, `<path>path/to/media</path>`).

Input parameters for transport stream

Table 60: Transport stream input parameters

Parameter	Type	Value	Description
`<ts>`	XML element		Parent element for an transport stream input file.
`<pid>`	Integer	Default: -1	Packet ID of the stream to demultiplex. Negative value selects first found video stream.
`<override_frame_rate>`	String	`23.976`, `24`, `25`, `29.97`, `30`, `48`, `50`, `59.94`, `60`	If present, the frame rate stored in the XML metadata within the container will be ignored and replaced with this value.
`<file_name>`	String	*filename*	Output file name.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored output.
`<path>`	String	*folder_path*	Path to the directory that contains the input file (for example, `<path>path/to/media</path>`).

5.9.3 Filter parameters for SDR to Dolby Vision profile 8 conversion

When creating an XML file, you can use filter parameters for SDR to Dolby Vision profile 8 conversion.

The following table describes the available parameters. The table does not include the parameters required by plug-ins, because those parameters depend on the plug-ins you choose to implement in the Dolby Encoding Engine.

Table 61: Filter parameters for SDR to Dolby Vision profile 8 conversion

Parameter	Type	Value	Description
`<sdr_to_dv_profile_8_2>`	XML element		Parent element for the Dolby Vision profile 8.2 filter parameters.
`<max_light_level>`	String	200, 4000 Default: 4000	Target nits maximum value (4,000 or 200).You can lower the target maximum nits to 200 if 4,000 amplifies noise levels to an undesirable degree in playback.
`<run_mode>`	String	linear_dynamic, singlenode Default: linear_dynamic	Allows you to choose the display mapping algorithm: single node (simple algorithm, faster) or linear dynamic (improved debanding algorithm, slower).
`<disable_md_postproc>`	Boolean	Default: false	Disables Dolby Vision metadata postprocessing.
`<disable_md_conform>`	Boolean	Default: false	Disables the Dolby Vision L1 metadata conformance check that normally runs at the end of processing.
`<hevc_dec>`	XML element	Example: `<impact>`	Specifies the HEVC decoder plug-in.

5.9.4 Output parameters for SDR to Dolby Vision profile 8 conversion

When creating an XML file, you can use several output parameters and values.

Output parameters for HEVC

Table 62: HEVC output parameters

Parameter	Type	Value	Description
`<hevc>`	XML element		Parent element for HEVC elementary stream output.
`<file_name>`	String	*filename*	Output file name.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored output.
`<path>`	String	*Storage folder path*	Path to the directory to save the output file (for example, `<path>path/to/media</path>`).

Output parameters for MP4

Table 63: MP4 output parameters

Parameter	Type	Value	Description
<mp4>	XML element		Parent element for an MP4 output file.
<output_format>	String	mp4, dash	Defines whether the output is an MP4 or DASH file.
<min_frag_duration>	Integer		Minimum fragment duration, in milliseconds.
<max_frag_duration>	Integer		Maximum fragment duration, in milliseconds.
<mux_flag>	String	encrypt_style_piff, write_iods, write_pdin, write_bloc, write_ainf, write_ctts_v1, write_subs_v1	Sets specified multiplexer flag.
<frag_flag>	String	frag_style_default, frag_style_cff, write_tfdt, write_sdtp, write_trik, write_sidx, write_mfra, force_tfra, no_base_data_offset, empty_trex, empty_tfhd, one_tfra_per_traf, default_base_is_moof	Sets specified fragmentation flag.
<track_flag>	String	track_enabled, track_in_movie, track_in_preview, track_in_poster	Sets specified track flag.
<file_name>	String	*filename*	Output file name.
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored output.
<path>	String	*Storage folder path*	Path to the directory to save the output file (for example, <path>path/to/media</path>).

Output parameters for transport stream

Table 64: Transport stream output parameters

Parameter	Type	Value	Description
<ts>	XML element		Parent element for a transport stream output file.
<packetize_standard>	String	mpeg, atsc, dvb Default: mpeg	Indicates the packetize standard used for substreams.
<packetize_mode>	String	cbr, vbr Default: vbr	Indicates constant bit rate (CBR) or variable bit rate (VBR) of the substreams.

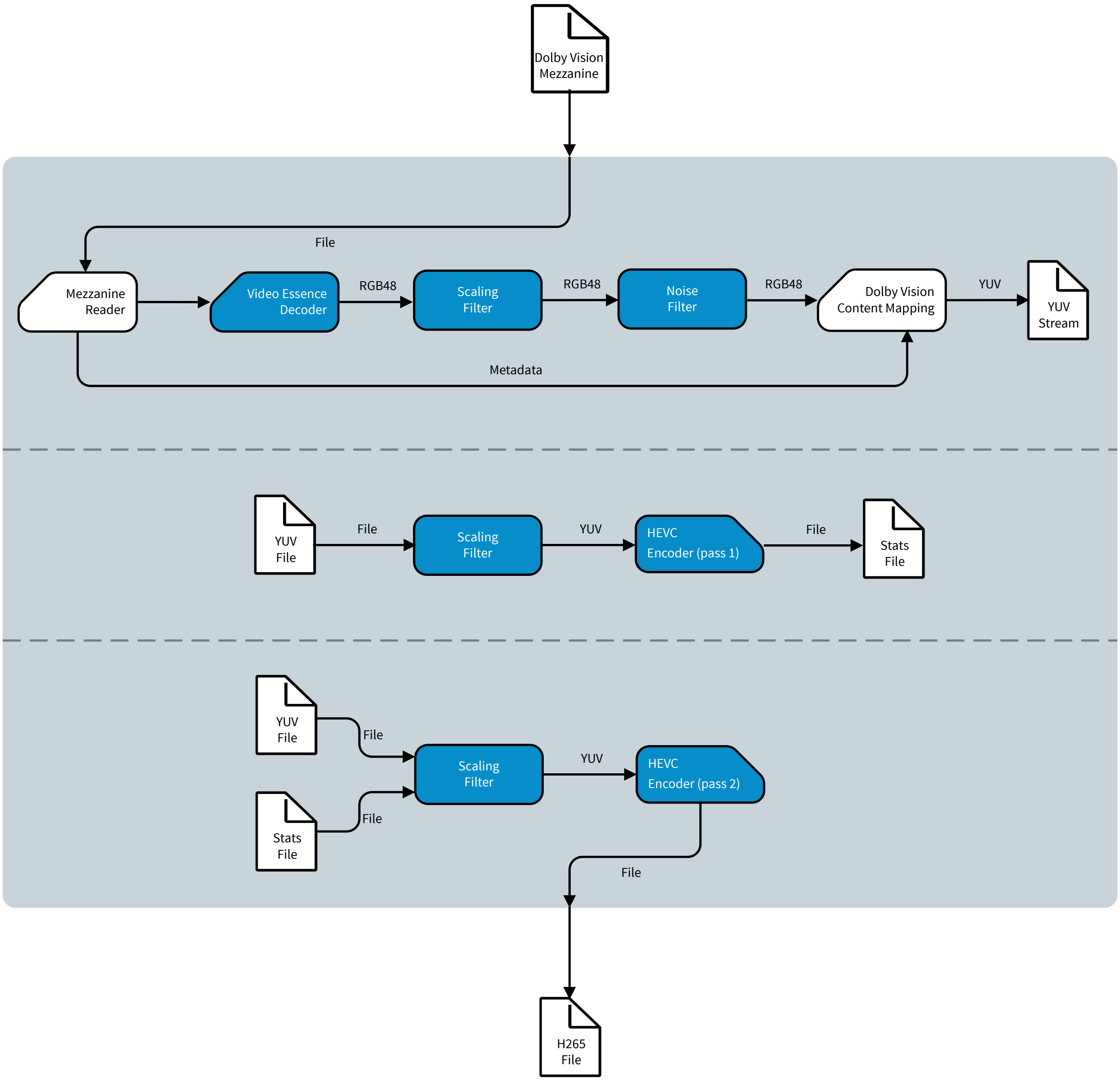
Table 64: Transport stream output parameters (continued)

Parameter	Type	Value	Description
`<transport_stream_id>`	Integer	Default: `0`	Transport stream ID.
`<audio_pid>`	Integer	`32–8186` Default: `40`	Packet ID of the audio stream.
`<video_pid>`	Integer	`32–8186` Default: `50`	Packet ID of the video stream.
`<file_name>`	String	*`filename`*	Output file name.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored output.
`<path>`	String	*`Storage folder path`*	Path to the directory to save the output file (for example, `<path>path/to/media</path>`).
`<plugin>`	XML element	Depends on the selected plug-in. Example: <base>	Allows you to choose the transport stream multiplexing plug-in. The base implementation is delivered with Dolby Encoding Engine.

5.10 SDR encoding job

The Dolby Encoding Engine can produce SDR video streams from Dolby Vision mezzanine input according to parameters you specify in the job configuration XML file.

Figure 7: SDR encode process



The input source can be MXF, MXF with sidecar Dolby Vision XML metadata, JPEG 2000 with sidecar Dolby Vision XML metadata, Apple ProRes with sidecar Dolby Vision XML metadata, and MOV with sidecar Dolby Vision XML metadata (with Apple ProRes essence).

The output is an HEVC video stream. The output can also be multiplexed into MP4 or transport stream containers.

The components marked in blue in the preceding figure are replaceable plug-ins. A standard installation of Dolby Encoding Engine includes default plug-in implementations for JPEG 2000 decoding, scaling, and noise reduction. The Dolby Encoding Engine allows you to replace these components with others that perform the same processing functions.

Note: The Dolby Encoding Engine does not include an HEVC encoder plugin but requires such an encoder to be built and installed. For more information, refer to the *Dolby Encoding Engine HEVC Encoder Installation Guide*.

5.10.1 XML example files for SDR

The XML template files are examples of a job configuration.

You can use them to create job configurations. You can find them in the xml_templates folder. For the SDR encoding job, refer to the following files:

- mxf_dv_mezz_to_sdr_hevc.xml
- mxf_dv_mezz_to_sdr_mp4.xml
- mxf_sidecar_dv_mezz_to_sdr_hevc.xml
- mxf_sidecar_dv_mezz_to_sdr_mp4.xml
- j2k_sidecar_dv_mezz_to_sdr_hevc.xml
- j2k_sidecar_dv_mezz_to_sdr_mp4.xml

5.10.2 Input parameters for SDR

When creating an XML file, you can use several SDR input parameters.

The supported input file formats are:

- MXF (with JPEG 2000 essence)
- MXF with sidecar Dolby Vision XML metadata (with JPEG 2000 essence)
- JPEG 2000 with sidecar Dolby Vision XML metadata
- Apple ProRes with sidecar Dolby Vision XML metadata
- MOV with sidecar Dolby Vision XML metadata (with Apple ProRes essence)

Input parameters for MXF

Table 65: MXF input parameters

Parameter	Type	Value	Description
`<mxf>`	XML element		Parent element.
`<override_frame_rate>`	String	`23.976`, `24`, `25`, `29.97`, `30`, `48`, `50`, `59.94`, `60`	If present, the frame rate stored in the XML metadata within the container will be ignored and replaced with this value.
`<file_name>`	String	*filename*	
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored input.
`<path>`	String	*folder_path*	Path to the directory that contains the input file (for example, `<path>path/to/media</path>`).

Input parameters for MXF with sidecar metadata

Table 66: MXF with sidecar metadata input parameters

Parameter	Type	Value	Description
`<mxf_sidecar>`	XML element		Parent element.
`<override_frame_rate>`	String	`23.976`, `24`, `25`, `29.97`, `30`, `48`, `50`, `59.94`, `60`	If present, the frame rate stored in the XML metadata within the container will be ignored and replaced with this value.
`<mxf_file_name>`	String	*filename*	Name of the MXF file containing video essence (JPEG 2000).
`<md_file_name>`	String	*filename*	Name of the file containing Dolby Vision metadata.
`<index_file_name>`	String	*filename*	Optionally, the name of an index file (if it was generated in an MXF parsing job).
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored input..
`<path>`	String	*folder_path*	Path to the directory that contains the input file (for example, `<path>path/to/media</path>`).

Input parameters for JPEG 2000 with sidecar metadata

Table 67: JPEG 2000 with sidecar metadata input parameters

Parameter	Type	Value	Description
`<j2k_sidecar>`	XML element		Parent element.
`<width>`	Integer		Input picture width.
`<height>`	Integer		Input picture height.
`<frame_rate>`	String	*23.976*, *24*, *25*, *29.97*, *30*, *59.94*, *60*	Input frame rate.
`<j2k_prefix>`	String		Allows you to use JPEG 2000 files with a specific prefix. If left empty (default), all JPEG 2000 files found in storage are used.
`<md_file_name>`	String	*filename*	Name of the file containing Dolby Vision metadata.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored input.
`<path>`	String	*folder_path*	Path to the directory that contains the input file (for example, `<path>path/to/media</path>`).

Input parameters for Apple ProRes with sidecar metadata

Table 68: Apple ProRes with sidecar metadata input parameters

Parameter	Type	Value	Description
<prores_sidecar>	XML element		Parent element.
<codec>	String	apch, apcn, apcs, apco, ap4h, ap4x	The type of the Apple ProRes codec: • apch: Apple ProRes 422 HQ • apcn: Apple ProRes 422 • apcs: Apple ProRes 422 LT • apco: Apple ProRes 422 Proxy • ap4h: Apple ProRes 4444 • ap4x: Apple ProRes 4444 XQ The particular type of the Apple ProRes codec can be automatically obtained from the manifest file. (If you use such a file, see the information in this section).
<width>			Input picture width.
<height>			Input picture height.
<frame_rate>	String	23.976, 24, 25, 29.97, 30, 48, 50, 59.94, 60	Input picture frame rate.
<prores_prefix>	String		Allows you to use Apple ProRes files with a specific prefix. If left empty (default), all Apple ProRes files found in storage are used.
<md_file_name>	String	*filename*	Name of the file containing Dolby Vision metadata.
<prores_manifest>	String	*filename*	Name of the optional manifest file containing details of the Apple ProRes frames (generated by the extract_prores filter). If the manifest is present, you do not have to provide the <codec>, <width>, <height>, and <frame_rate> parameters.
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored input.
<path>	String	*folder_path*	Path to the directory that contains the input file (for example, <path>path/to/media</path>).

Input parameters for MOV with sidecar metadata

Table 69: MOV with sidecar metadata input parameters

Parameter	Type	Value	Description
`<mov_sidecar>`	XML element		Parent element.
`<override_frame_rate>`	String	`23.976`, `24`, `25`, `29.97`, `30`, `48`, `50`, `59.94`, `60`	If present, the frame rate stored in the XML metadata within the container will be ignored and replaced with this value.
`<track_id>`	Integer	Default: -1	ID of track to demultiplex. Negative value selects the first found video track.
`<mov_file_name>`	String	*filename*	Name of the MOV file containing video essence (Apple ProRes).
`<md_file_name>`	String	*filename*	Name of the file containing Dolby Vision metadata.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored input.
`<path>`	String	*folder_path*	Path to the directory that contains the input file (for example, `<path>path/to/media</path>`).

5.10.3 Filter parameters for SDR

When creating an XML file, you can use the following SDR filter parameters.

The table below describes the available parameters. The table does not include the parameters required by plugins, since those parameters depend on the plugins you choose to implement in the Dolby Encoding Engine.

Table 70: SDR filter parameters

Parameter	Type	Value	Description
`<dv_mezz_to_sdr>`	XML element		Parent element for the SDR filter parameters.
`<start>`	Integer	Default: 0	Start frame or start timecode.
`<end>`	Integer	Default: 0	End frame or end timecode. (0=end of file)
`<metadata_offset>`	Integer	Default: 0	Number of frames to skip while reading metadata. Use only when video frames and metadata are not aligned.
`<active_area_offset_top>`	Integer	Default: 0	Top offset of the active area in the source content. Active area determines where in the encoded picture the actual video content is located. If the video is encoded with solid color bars around the content, setting the appropriate active area will exclude them from processing.

Table 70: SDR filter parameters (continued)

Parameter	Type	Value	Description
`<active_area_offset_bottom>`	Integer	Default: 0	Bottom offset of the active area in the source content. Active area determines where in the encoded picture the actual video content is located. If the video is encoded with solid color bars around the content, setting the appropriate active area will exclude them from processing.
`<active_area_offset_left>`	Integer	Default: 0	Left offset of the active area in the source content. Active area determines where in the encoded picture the actual video content is located. If the video is encoded with solid color bars around the content, setting the appropriate active area will exclude them from processing.
`<active_area_offset_right>`	Integer	Default: 0	Right offset of the active area in the source content. Active area determines where in the encoded picture the actual video content is located. If the video is encoded with solid color bars around the content, setting the appropriate active area will exclude them from processing.
`<decoder_parallel_frames>`	Integer	Values between 0 and 16. Default: 0	Number of parallel video essence pictures to process. Higher values can reduce processing time at the cost of increased CPU and memory usage. The default 0 value sets the parameter to auto. Depending on which decoder you select, the auto value means: • For the JPEG 2000 Base decoder, four decoder instatnces run in parallel. • For the Kakadu decoder, two multi-threaded instances run in parallel. • For the Apple ProRes decoder, the process uses all available cores.
`<prepend_frames>`	Integer	Default: 0	Number of black frames prepending the target content.
`<target_width>`	Integer list	Accepts multiple integer values, separated by a space. Default: 0	Width of target picture (0 = use source width).
`<target_area>`	Integer list	Accepts multiple integer values, separated by a space. Default: 0	Height of target picture (0 = use source height).

Table 70: SDR filter parameters (continued)

Parameter	Type	Value	Description
`<max_scene_frames>`	Integer	Default: 256	The maximum number of frames in a buffer within the Dolby Vision processor. It is recommended to use a number that is a multiple of the GOP size (e.g. for GOP Size=48, max_scene_frames=240) and not smaller than GOP size. This property has significant impact on RAM consumption during the Dolby Vision processor execution.
`<data_rate>`	Integer list	Accepts multiple integer values, separated by a space. Default: 15000	Average data rate in kbps.
`<max_vbv_data_rate>`	Integer list	Accepts multiple integer values, separated by a space. Default: 15000	Maximum video buffering verifier data rate in kbps.
`<vbv_buffer_size>`	Integer list	Accepts multiple integer values, separated by a space. Default: 30000	Video buffering verifier buffer size in kb.
`<encode_pass_num>`	Integer	Default: 1	Number of encoder passes. *
`<disable_md_postproc>`	Boolean	Default: false	Disables Dolby Vision metadata post-processing.
`<decoder>`	XML element	`<j2k_dec>` or `<prores_dec>`	Specifies the input decoder.
`<scaling>`	XML element	Example: `<base>`	Specifies the scaling filter plugin. The base implementation is delivered with Dolby Encoding Engine.
`<source_crop_top>`	Integer	Default: 0	A positive value indicates how many pixels should be removed from the top of the source picture before scaling; a negative value will pad the top with black pixels.
`<source_crop_bottom>`	Integer	Default: 0	A positive value indicates how many pixels should be removed from the bottom of the source picture before scaling; a negative value will pad the bottom with black pixels.
`<source_crop_left>`	Integer	Default: 0	A positive value indicates how many pixels should be removed from the left side of the source picture before scaling; a negative value will pad the left side with black pixels.

Table 70: SDR filter parameters (continued)

Parameter	Type	Value	Description
<source_crop_right>	Integer	Default: 0	A positive value indicates how many pixels should be removed from the right side of the source picture before scaling; a negative value will pad the right side with black pixels.
<noise>	XML element	Example: <base>	Specifies the noise reduction filter plugin. The base implementation is delivered with Dolby Encoding Engine.
<strength>	Integer	Default: 0	In the case of the <base> noise filter, the only currently supported value is 0 (noise reduction turned off).
<hevc_enc>	XML element	Depends on the selected plugin. Example: <impact>	Specifies the HEVC encoder plugin.

[*] An encoder pass is a single instance of processing a mezzanine YUV file.

5.10.4 Output parameters for SDR

When creating an XML file, you can use several output parameters and values.

Output parameters for HEVC

Table 71: HEVC output parameters

Parameter	Type	Value	Description
<hevc>	XML element		Parent element for HEVC elementary stream output.
<file_name>	String	*filename*	Output file name.
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored output.
<path>	String	*Storage folder path*	Path to the directory to save the output file (for example, <path>path/to/media</path>).

Output parameters for MP4

Table 72: MP4 output parameters

Parameter	Type	Value	Description
<mp4>	XML element		Parent element for an MP4 output file.
<output_format>	String	mp4, dash	Defines whether the output is an MP4 or DASH file.
<min_frag_duration>	Integer		Minimum fragment duration, in milliseconds.
<max_frag_duration>	Integer		Maximum fragment duration, in milliseconds.

Table 72: MP4 output parameters (continued)

Parameter	Type	Value	Description
`<mux_flag>`	String	encrypt_style_piff, write_iods, write_pdin, write_bloc, write_ainf, write_ctts_v1, write_subs_v1	Sets specified multiplexer flag.
`<frag_flag>`	String	frag_style_default, frag_style_cff, write_tfdt, write_sdtp, write_trik, write_sidx, write_mfra, force_tfra, no_base_data_offset, empty_trex, empty_tfhd, one_tfra_per_traf, default_base_is_moof	Sets specified fragmentation flag.
`<track_flag>`	String	track_enabled, track_in_movie, track_in_preview, track_in_poster	Sets specified track flag.
`<file_name>`	String	*filename*	Output file name.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored output.
`<path>`	String	*Storage folder path*	Path to the directory to save the output file (for example, `<path>path/to/media</path>`).

Output parameters for transport stream

Table 73: Transport stream output parameters

Parameter	Type	Value	Description
`<ts>`	XML element		Parent element for a transport stream output file.
`<packetize_standard>`	String	mpeg, atsc, dvb Default: mpeg	Indicates the packetize standard used for substreams.
`<packetize_mode>`	String	cbr, vbr Default: vbr	Indicates constant bit rate (CBR) or variable bit rate (VBR) of the substreams.
`<transport_stream_id>`	Integer	Default: 0	Transport stream ID.
`<audio_pid>`	Integer	32–8186 Default: 40	Packet ID of the audio stream.
`<video_pid>`	Integer	32–8186 Default: 50	Packet ID of the video stream.
`<file_name>`	String	*filename*	Output file name.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.

Table 73: Transport stream output parameters (continued)

Parameter	Type	Value	Description
`<local>`	XML element		Parent element for parameters of locally stored output.
`<path>`	String	*Storage folder path*	Path to the directory to save the output file (for example, `<path>path/to/media</path>`).
`<plugin>`	XML element	Depends on the selected plug-in. Example: `<base>`	Allows you to choose the transport stream multiplexing plug-in. The base implementation is delivered with Dolby Encoding Engine.

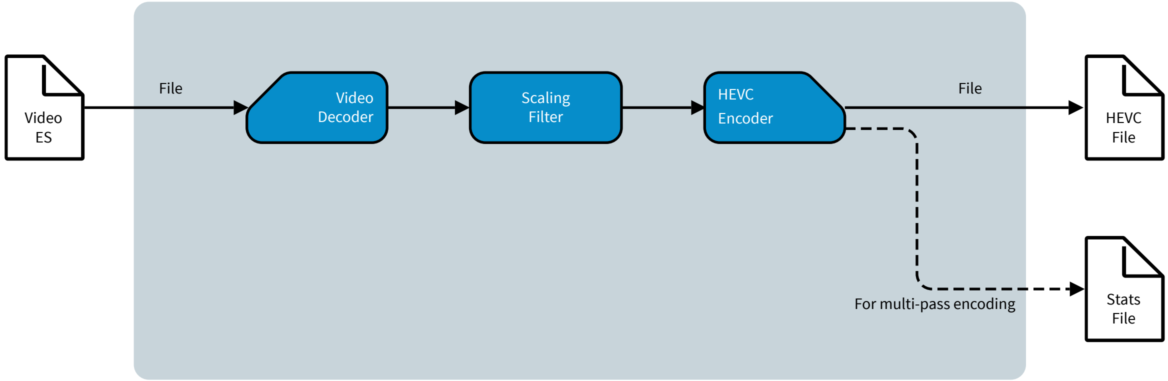
5.11 HEVC transcoding job

The HEVC transcoder can produce HEVC from various video formats according to the parameters you specify in the job configuration XML file.

The HEVC transcoder is used to transcode various video formats into HEVC.

Currently, it upports HEVC-to-HEVC transcoding only. This transcoding job can be used to create HEVC streams that comply with the SDR source requirements for Dolby Vision. You can later use such a HEVC SDR stream to create Dolby Vision content (for example, in a Dolby Vision profile 8.2 encoding job).

Figure 8: HEVC transcode process



The input source is an HEVC file (also in the form of MP4 or transport stream containers).

The output is an HEVC file. The output can also be multiplexed into MP4 or transport stream containers.

A standard installation of Dolby Encoding Engine allows you to replace the scaling, HEVC decoder, and HEVC encoder components with others that perform the same processing functions. A standard installation of Dolby Encoding Engine includes default plug-in implementation for scaling and HEVC decoder.

Note: The Dolby Encoding Engine does not include an HEVC encoder plugin but requires such an encoder to be built and installed. For more information, refer to the *Dolby Encoding Engine HEVC Encoder Installation Guide*.

Related information

SDR to Dolby Vision profile 8 encoding job on page 82

Input requirements for SDR to Dolby Vision encoding on page 131

5.11.1 XML example files for High-Efficiency Video Coding transcode

The XML template files are examples of a job configuration.

You can use them to create job configurations. You can find them in the xml_templates folder. For the High-Efficiency Video Coding encoding job, refer to the following files:

- hevc_transcode_to_hevc_hevc.xml
- hevc_transcode_to_hevc_mp4.xml
- ts_transcode_to_hevc_ts.xml

5.11.2 Input parameters for High-Efficiency Video Coding transcode

When creating an XML file, you can use several HEVC transcode input parameters.

The supported input file format is an HEVC. The input can also be an MP4 container with an HEVC video stream.

Input parameters for HEVC

Table 74: HEVC input parameters

Parameter	Type	Value	Description
<hevc>	XML element		Parent element.
<frame_rate>	String	23.976, 24, 25, 29.97, 30, 48, 50, 59.94, 60	Video frame rate. Required in use-cases with HEVC decoding.
<dv_profile>	String	none (default)	Dolby Vision profile. For HEVC transcoding, use the default value.
<file_name>	String	*filename*	
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored input.
<path>	String	*folder_path*	Path to the directory that contains the input file (for example, <path>path/to/media</path>).

Input parameters for MP4

Table 75: MP4 input parameters

Parameter	Type	Value	Description
<mp4>	XML element		Parent element for an MP4 input file.
<track_id>	Integer	Default: -1	ID of track to demultiplex. Negative value selects the first found video track.
<override_frame_rate>	String	23.976, 24, 25, 29.97, 30, 48, 50, 59.94, 60	If present, the frame rate stored in the XML metadata within the container will be ignored and replaced with this value.

Table 75: MP4 input parameters (continued)

Parameter	Type	Value	Description
`<file_name>`	String	*filename*	Output file name.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored output.
`<path>`	String	*folder_path*	Path to the directory that contains the input file (for example, `<path>path/to/media</path>`).

Input parameters for transport stream

Table 76: Transport stream input parameters

Parameter	Type	Value	Description
`<ts>`	XML element		Parent element for an transport stream input file.
`<pid>`	Integer	Default: -1	Packet ID of the stream to demultiplex. Negative value selects first found video stream.
`<override_frame_rate>`	String	23.976, 24, 25, 29.97, 30, 48, 50, 59.94, 60	If present, the frame rate stored in the XML metadata within the container will be ignored and replaced with this value.
`<file_name>`	String	*filename*	Output file name.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored output.
`<path>`	String	*folder_path*	Path to the directory that contains the input file (for example, `<path>path/to/media</path>`).

5.11.3 Filter parameters for High-Efficiency Video Coding transcode

When creating an XML file, you can use the following HEVC transcode filter parameters.

The table below describes the available parameters. The table does not include the parameters required by plugins, since those parameters depend on the plugins you choose to implement in Dolby Encoding Engine.

Table 77: HEVC transcode filter parameters

Parameter	Type	Value	Description
<transcode_to_hevc>	XML element		Parent element for the HEVC transcode filter parameters.
<start>	Integer	Default: 0	Start frame or start timecode.
<end>	Integer	Default: 0	End frame or end timecode. (0=end of file)
<target_width>	Integer list	Accepts multiple integer values, separated by a space. Default: 0	Width of target picture (0 = use source width).
<target_area>	Integer list	Accepts multiple integer values, separated by a space. Default: 0	Height of target picture (0 = use source height).
<data_rate>	Integer list	Accepts multiple integer values, separated by a space. Default: 15000	Average data rate in kbps.
<max_vbv_data_rate>	Integer list	Accepts multiple integer values, separated by a space. Default: 15000	Maximum video buffering verifier data rate in kbps.
<vbv_buffer_size>	Integer list	Accepts multiple integer values, separated by a space. Default: 30000	Video buffering verifier buffer size in kb.
<encode_pass_num>	Integer	Default: 1	Number of encoder passes. *
<decoder>	XML element	<hevc_dec>	Specifies the input decoder. Currently, this filter only supports <hevc_dec>.
<scaling>	XML element	Example: <base>	Specifies the scaling filter plugin. The base implementation is delivered with Dolby Encoding Engine.
<source_crop_top>	Integer	Default: 0	A positive value indicates how many pixels should be removed from the top of the source picture before scaling; a negative value will pad the top with black pixels.
<source_crop_bottom>	Integer	Default: 0	A positive value indicates how many pixels should be removed from the bottom of the source picture before scaling; a negative value will pad the bottom with black pixels.

Table 77: HEVC transcode filter parameters (continued)

Parameter	Type	Value	Description
`<source_crop_left>`	Integer	Default: 0	A positive value indicates how many pixels should be removed from the left side of the source picture before scaling; a negative value will pad the left side with black pixels.
`<source_crop_right>`	Integer	Default: 0	A positive value indicates how many pixels should be removed from the right side of the source picture before scaling; a negative value will pad the right side with black pixels.
`<hevc_enc>`	XML element	Example: `<impact>`	Specifies the HEVC encoder plugin.

* An encoder pass is a single instance of processing a mezzanine YUV file.

5.11.4 Output parameters for High-Efficiency Video Coding transcode

When creating an XML file, you can use several output parameters and values.

Output parameters for HEVC

Table 78: HEVC output parameters

Parameter	Type	Value	Description
`<hevc>`	XML element		Parent element for HEVC elementary stream output.
`<file_name>`	String	*filename*	Output file name.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored output.
`<path>`	String	*Storage folder path*	Path to the directory to save the output file (for example, `<path>path/to/media</path>`).

Output parameters for MP4

Table 79: MP4 output parameters

Parameter	Type	Value	Description
`<mp4>`	XML element		Parent element for an MP4 output file.
`<output_format>`	String	mp4, dash	Defines whether the output is an MP4 or DASH file.
`<min_frag_duration>`	Integer		Minimum fragment duration, in milliseconds.
`<max_frag_duration>`	Integer		Maximum fragment duration, in milliseconds.
`<mux_flag>`	String	`encrypt_style_piff`, `write_iods`, `write_pdin`, `write_bloc`, `write_ainf`, `write_ctts_v1`, `write_subs_v1`	Sets specified multiplexer flag.

Table 79: MP4 output parameters (continued)

Parameter	Type	Value	Description
`<frag_flag>`	String	`frag_style_default`, `frag_style_cff`, `write_tfdt`, `write_sdtp`, `write_trik`, `write_sidx`, `write_mfra`, `force_tfra`, `no_base_data_offset`, `empty_trex`, `empty_tfhd`, `one_tfra_per_traf`, `default_base_is_moof`	Sets specified fragmentation flag.
`<track_flag>`	String	`track_enabled`, `track_in_movie`, `track_in_preview`, `track_in_poster`	Sets specified track flag.
`<file_name>`	String	*`filename`*	Output file name.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored output.
`<path>`	String	*`Storage folder path`*	Path to the directory to save the output file (for example, `<path>path/to/media</path>`).

Output parameters for transport stream

Table 80: Transport stream output parameters

Parameter	Type	Value	Description
`<ts>`	XML element		Parent element for a transport stream output file.
`<packetize_standard>`	String	`mpeg`, `atsc`, `dvb` Default: `mpeg`	Indicates the packetize standard used for substreams.
`<packetize_mode>`	String	`cbr`, `vbr` Default: `vbr`	Indicates constant bit rate (CBR) or variable bit rate (VBR) of the substreams.
`<transport_stream_id>`	Integer	Default: `0`	Transport stream ID.
`<audio_pid>`	Integer	`32`–`8186` Default: `40`	Packet ID of the audio stream.
`<video_pid>`	Integer	`32`–`8186` Default: `50`	Packet ID of the video stream.
`<file_name>`	String	*`filename`*	Output file name.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored output.

Table 80: Transport stream output parameters (continued)

Parameter	Type	Value	Description
`<path>`	String	*Storage folder path*	Path to the directory to save the output file (for example, `<path>path/to/media</path>`).
`<plugin>`	XML element	Depends on the selected plug-in. Example: `<base>`	Allows you to choose the transport stream multiplexing plug-in. The base implementation is delivered with Dolby Encoding Engine.

6 Distributed encoding of Dolby Vision

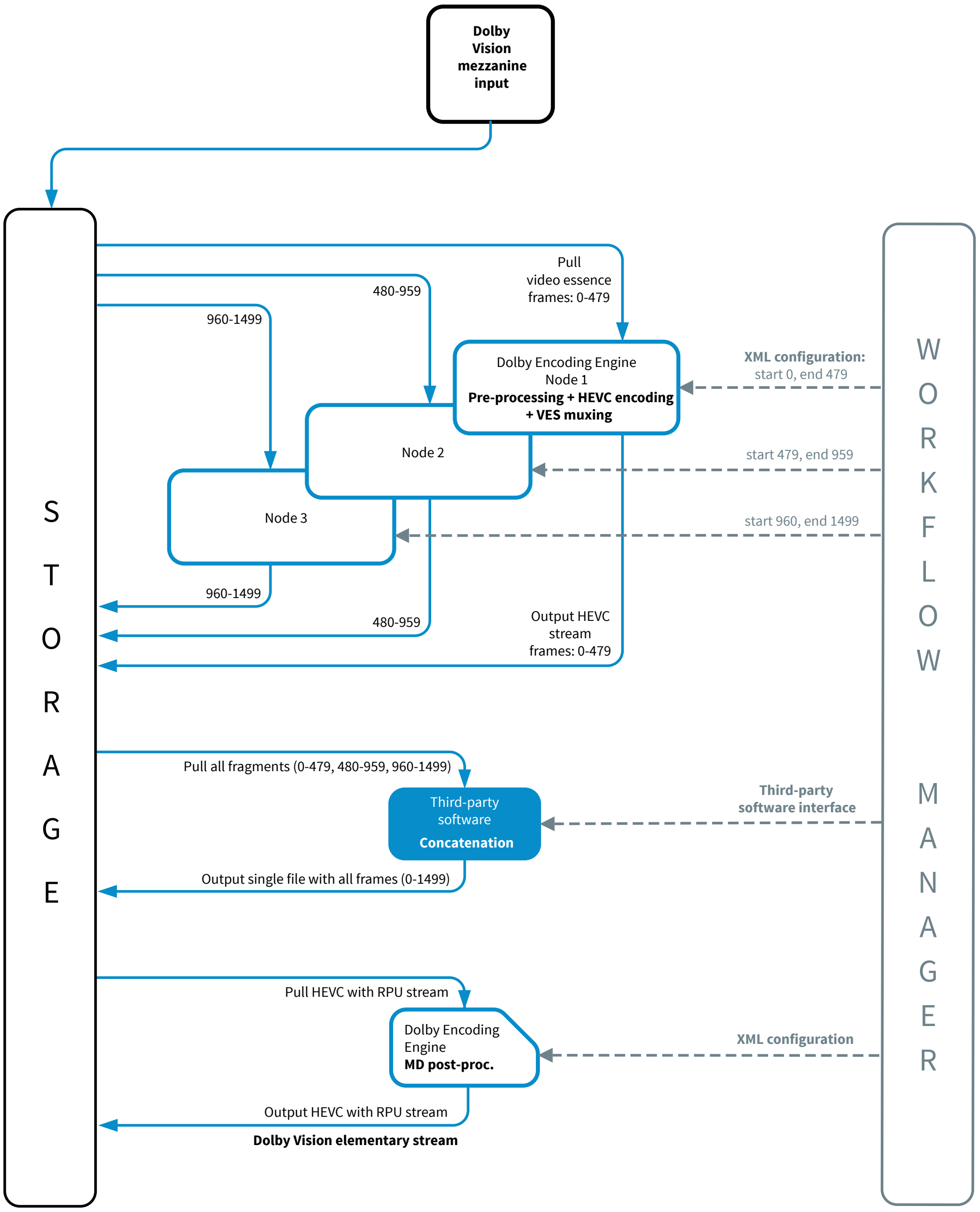
The Dolby Vision profile 5 encoding process can be performed simultaneously by multiple Dolby Encoding Engine nodes to reduce the time necessary for a particular encoding job. There are two scenarios of the distributed encoding process.

- Basic distributed encoding
- Expanded distributed encoding

6.1 Basic distributed encoding

In this scenario, the Dolby Encoding Engine processes the input in chunks, and divides the encoding process into parts while a separate node performs the metadata post-processing.

Figure 9: Basic distributed encoding process


Dolby Vision mezzanine input
STORAGE
WORKFLOW MANAGER
Pull video essence frames: 0-479
480-959
960-1499
Dolby Encoding Engine Node 1 Pre-processing + HEVC encoding + VES muxing
Node 2
Node 3
XML configuration: start 0, end 479
start 479, end 959
start 960, end 1499
960-1499
480-959
Output HEVC stream frames: 0-479
Pull all fragments (0-479, 480-959, 960-1499)
Third-party software Concatenation
Third-party software interface
Output single file with all frames (0-1499)
Pull HEVC with RPU stream
Dolby Encoding Engine MD post-proc.
XML configuration
Output HEVC with RPU stream
Dolby Vision elementary stream

The process consists of three stages:

1. Encoding, where each node executes Dolby Vision pre-processing, HEVC encoding, and VES multiplexing:

   a. Use the dv_mezz_to_dv_profile_5 filter.

   b. Set disable_md_postproc to true.

   c. Make sure the HEVC encoder is set to use fixed group-of-pictures (GOP) size.

   d. max_scene_frames=M*GOP_Size.

      Note: M is integer >=1. GOP_Size is the number of frames set by the user and reflects further High-Efficiency Video Coding encoder settings. Maximum value is 256. If M*GOP_Size exceeds 256, reduce M or use max_scene_frames=256.

   e. To divide the input file, use the start and end parameters, where start=(N-1)*max_scene_frames and end=(N*max_scene_frames)-1.

Note: N is chunk index starting from 1.

2. Concatenation:

   a. The concatenation of the output files produced by these parallel processes must be performed by a third-party application. Dolby Encoding Engine does not concatenate output files.

   b. You can perform the concatenation by executing the following commands:

      - On Linux: `cat file_name_1.hevc file_name_2.hevc file_name_3.hevc > /storage/concatenated_file_name.hevc`
      - On Windows: `copy file_name_1.hevc + file_name_2.hevc + file_name_3.hevc \storage\concatenated_file_name.hevc`

3. Metadata postprocessing

   a. See the example XML file hevc_dv_md_postproc_hevc.xml.

Any kind of a workflow manager or script governing the distributed encoding process must be provided by the customer. Dolby does not provide these for the Dolby Encoding Engine.

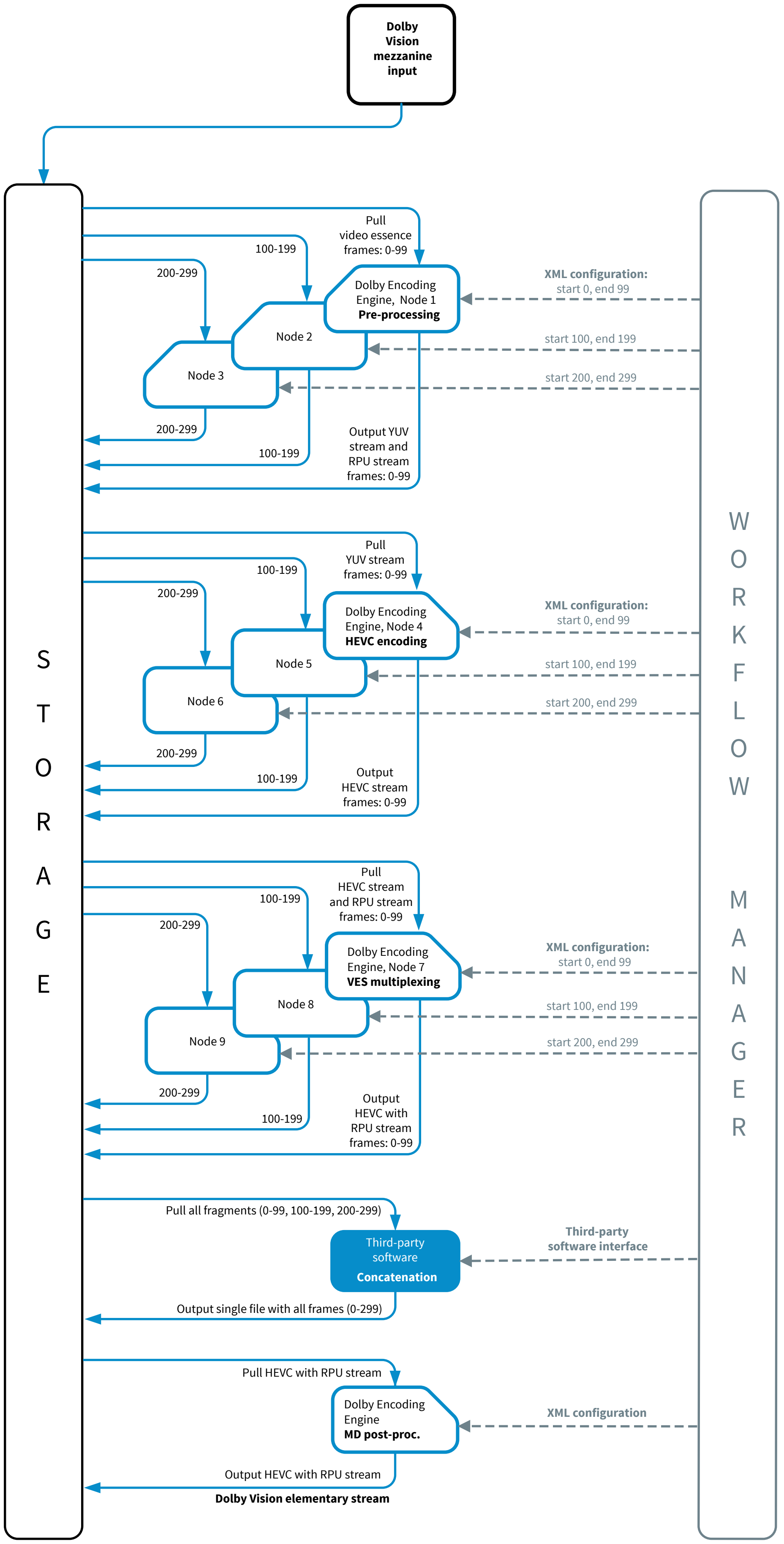
Related information

MXF parsing job on page 110

6.2 Expanded distributed encoding

In this scenario, the Dolby Encoding Engine processes the input in chunks, and divides the process into separate parts for preprocessing, encoding, video elementary stream multiplexing, and metadata postprocessing.

Figure 10: Expanded distributed encoding process


Dolby Vision mezzanine input
S T O R A G E
W O R K F L O W M A N A G E R
Pull video essence frames: 0-99
100-199
200-299
Dolby Encoding Engine, Node 1
Pre-processing
Node 2
Node 3
XML configuration:
start 0, end 99
start 100, end 199
start 200, end 299
200-299
100-199
Output YUV stream and RPU stream frames: 0-99
Pull YUV stream frames: 0-99
100-199
200-299
Dolby Encoding Engine, Node 4
HEVC encoding
Node 5
Node 6
XML configuration:
start 0, end 99
start 100, end 199
start 200, end 299
200-299
100-199
Output HEVC stream frames: 0-99
Pull HEVC stream and RPU stream frames: 0-99
100-199
200-299
Dolby Encoding Engine, Node 7
VES multiplexing
Node 8
Node 9
XML configuration:
start 0, end 99
start 100, end 199
start 200, end 299
200-299
100-199
Output HEVC with RPU stream frames: 0-99
Pull all fragments (0-99, 100-199, 200-299)
Third-party software
Concatenation
Third-party software interface
Output single file with all frames (0-299)
Pull HEVC with RPU stream
Dolby Encoding Engine
MD post-proc.
XML configuration
Output HEVC with RPU stream
Dolby Vision elementary stream

The process consists of five stages:

1. Dolby Vision preprocessing, which produces a YUV segment and a corresponding reference picture unit (RPU) stream:
    a. Use the `dv_mezz_preproc_profile_5` filter.

        Note: Optionally, you can use another filter as the first step of this process:

        - In the case of an MXF input file, you can use the `extract_j2k` filter, which extracts JPEG 2000 with sidecar output from the MXF. The JPEG 2000 with sidecar content is later used as input for the `dv_mezz_preproc_profile_5` filter.
        - In the case of a MOV input file, you can use the `extract_prores` filter, which extracts Apple ProRes output from the MOV file. The Apple ProRes content is later used as input for the `dv_mezz_preproc_profile_5` filter.

        This way, the process can be distributed among more Dolby Encoding Engine nodes.
    b. To divide the input, use the `start` and `end` parameters.
    c. For further reference, see these example XML files in the xml_templates folder:
        - mxf_dv_mezz_preproc_profile_5_yuv420_rpu.xml
        - mxf_sidecar_dv_mezz_preproc_profile_5_yuv420_rpu.xml
        - mov_sidecar_dv_mezz_preproc_profile_5_yuv420_rpu.xml
    d. For the optional `extract_j2k` and `extract_prores` filters, see mxf_extract_j2k_j2k_sidecar.xml and mov_extract_prores_prores.xml.
2. HEVC encoding of the YUV content:
    a. Use the `encode_to_hevc` filter.
    b. To divide the input, use the `start` and `end` parameters.
    c. See the example XML file yuv420_encode_to_hevc_hevc.xml.
3. Video elementary stream (VES) multiplexing that combines a reference picture unit (RPU) stream and an HEVC stream and produces a Dolby Vision elementary stream:
    a. Use the `dv_ves_mux` filter.
    b. To divide the input, use the `start` and `end` parameters.
    c. See the example XML file bl_el_rpu_dv_ves_mux_hevc.xml.
    d. In the example XML file, remove the `<el_file_name>` element (because the enhanced layer is not supported in profile 5 encoding).
4. Concatenation:
    a. The concatenation of the output files produced by these parallel processes must be performed by a third-party application. Dolby Encoding Engine does not concatenate output files.
    b. You can perform the concatenation by executing the following commands:
        - On Linux: `cat file_name_1.hevc file_name_2.hevc file_name_3.hevc > /`*`storage`*`/concatenated_file_name.hevc`
        - On Windows: `copy file_name_1.hevc + file_name_2.hevc + file_name_3.hevc \storage\concatenated_file_name.hevc`
5. Metadata postprocessing:
    a. Use the `dv_md_postproc` filter.

b. See the example XML file hevc_dv_md_postproc_hevc.xml.

Any kind of a workflow manager or script governing the distributed encoding process must be provided by the customer. Dolby does not provide these for the Dolby Encoding Engine.

Related information

MXF parsing job on page 110

7 Optimized workflows

The encoding process can be optimized in order to save time and resources (for example, by performing the MXF parsing job).

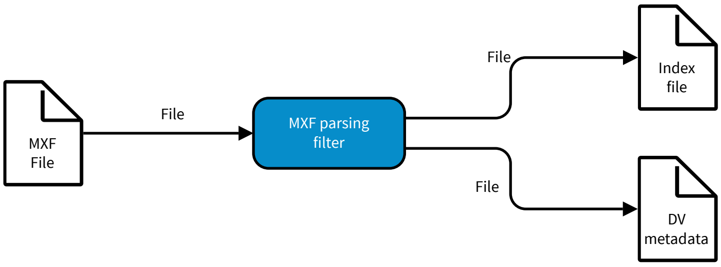
7.1 MXF parsing job

In this job, you can optimize video encoding by parsing the MXF input only once, to skip the parsing stage during subsequent multiple video encoding jobs with that same input.

In the MXF parsing job, the `parse_mxf` filter generates an MXF index file and extracts Dolby Vision metadata from the MXF input.

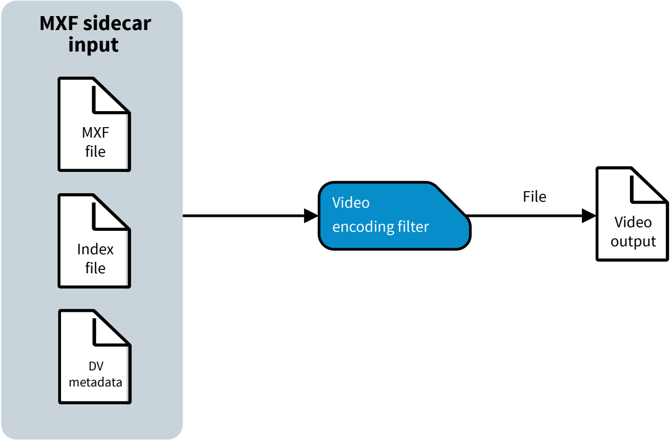
This process can be particularly beneficial when combined with one of the distributed encoding scenarios. Refer to the “Distributed Encoding of Dolby Vision” chapter for details.

Figure 11: MXF parsing job (performed one time)



To encode video, use the MXF index file and the Dolby Vision metadata file generated in the MXF parsing job, and combine them with the MXF file as MXF sidecar input.

Figure 12: Video encoding workflow: Optimized using the MXF parsing job (can be performed multiple times)



XML example file for MXF parsing job

The XML template files are examples of a job configuration. You can use them to create job configurations. You can find them in the xml_templates folder. For the MXF parsing job, refer to mxf_parse_mxf_index_dvmd.xml.

Input parameters of the MXF parsing job

The supported input format is MXF.

Filter parameters of the MXF parsing job

Using the filter parameters, you can to specify if you want to generate both, the MXF index file and the Dolby Vision metadata, or only one of them.

Table 81: MXF parsing filter parameters

Parameter	Type	Value	Description
`<parse_mxf>`	XML element		Parent element for the MXF parsing filter parameters.
`<generate_index_file>`	Boolean	Default: `true`	Generates the index file.
`<generate_checksums>`	Boolean	Default: `true`	Generates checksums for the index file. Disable to reduce processing time.
`<extract_dv_md>`	Boolean	Default: `false`	Generates the Dolby Vision metadata. Set to `false` if your MXF does not contain Dolby Vision metadata or you do not need it.
`<debug_level>`	Integer	`0`–`3` Default: 1	Parser log verbosity: `0` = quiet, 1 = errors, 2 = warnings, 3 = debug.

Output parameters of the MXF parsing job

The output format must be defined as `<generic>`, because the MXF index file and the Dolby Vision metadata are separate plain text files.

If you generate both files, the `<file_name>` element must contain two files: the first one is the MXF index file, and the second one is the Dolby Vision metadata file. If you generate only one file (no matter which one), the `<file_name>` element must also contain only one file.

Related information

Dolby Vision profile 5 encoding job on page 42

Dolby Vision profile 8.1 encoding job on page 53

HDR10 video encoding job on page 64

SDR encoding job on page 87

Basic distributed encoding on page 104

Expanded distributed encoding on page 106

8 Custom YUV sink for Dolby Vision profile 5 preprocessor

By default, the Dolby Vision Profile 5 preprocessor outputs YUV and reference picture unit (RPU) files. The custom YUV sink allows you to output pictures in other formats, as well.

Example applications include:

- Using an HEVC encoder outside of a Dolby Encoding Engine pipeline where the encoder does not accept YUV as an input
- The need to apply compression due to disk space limitations or data transfer efficiency
- Diskless operation for preprocessing and one-pass encoding where the sink is configured to encode instead of to convert

Internal process description

1. A script and an interpreter generate an execution command. (The script and the interpreter are specified in the enabling custom YUV sink procedure by the `<cmd_gen>` and `<interpreter>` tags.)
2. The Dolby Encoding Engine executes the command.
3. The Dolby Encoding Engine communicates with the executed third-party software via named pipes so the program that works as an YUV converter has to support pipe input.
    - Windows: The pipe is created using a `CreateNamedPipe()` WinAPI method.
    - Linux: The pipe is created using a `mkfifo()` method.

XML example file for custom YUV sink

The XML template files are examples of a job configuration. You can use them to create job configurations. You can find them in the xml_templates folder. For the custom YUV sink, refer to the mxf_dv_mezz_preproc_profile_5_custom_yuv_sink.xml template.

8.1 Enabling custom YUV sink

Procedure

1. Open the mxf_dv_mezz_preproc_profile_5_custom_yuv_sink.xml template (located in xml_templates in the Dolby Encoding Engine installation directory).
2. Configure the input `<file_name>` and `<path>`.

   You can use `mxf`, `mxf_sidecar`, or `j2k_sidecar` inputs in the same way as for any Dolby Vision profile 5 preprocessing.
3. Configure the filter settings.
4. Configure the `<pipe_cmd>` section inside the `<custom_yuv_sink>` tag by specifying the `<cmd_gen>` and `<interpreter>` fields.

   In the Dolby Encoding Engine installation directory, you can find two examples:

    - cmd_gen_custom_yuv_sink_example1.py output: MXF with losslessly compressed JPEG 2000 frames

- cmd_gen_custom_yuv_sink_example2.py output: Losslessly compressed HEVC bitstream using x265 encoder

5. Configure the `<generic>` output settings. You must specify two file names. The first entry is the converted video file, for example an HEVC or an MXF file. The second entry is the RPU file.

   The `<generic>` output is necessary for correct operation.

6. In the `<misc>` section, specify if you want to keep the temp file and where to store it.

9 Batch processing

Dolby Encoding Engine supports multiple batch operation scenarios to speed up and simplify the encoding process.

- Using multiple input files
- Specifying multiple output locations
- Automated naming of output files
- Multiple Dolby Vision data rates and resolutions

9.1 Using multiple input files

You can use wildcards to specify multiple input files to undergo the same encoding process sequentially (one after another).

About this task

In this process, the number of produced output files is the same as the number of input files.

Procedure

1. In the XML configuration file, use the asterisk (*) wildcard to specify multiple input files, as in the following example.

```
<input>
    <audio>
        <wav>
            <file_name>*.wav</file_name>
            <storage>
                <local>
                    <path>PATH</path>
                </local>
            </storage>
        </wav>
    </audio>
</input>
```

   The preceding example defines all files with a .wav extension in the specified folder as input files.

2. Specify the filter type and other desired parameters in the XML configuration file.
3. As the output file name, use the `$base($var(input:`*`file_name`*`))`*`.extension`* syntax, where:

   `file_name` is the name of a particular input file that is the basis for the output file name.

   `.extension` is the desired extension of the output files.

Because the number of input files is the same as the number of output files, each output file must have a unique name. To achieve that, you must use the preceding syntax so that the name of each input file works as a basis for each output file name.

4. Save the XML file.
5. Execute the encode command in the CLI including the new XML file.

Related information

Automated naming of output files on page 116

9.2 Specifying multiple output locations

You can specify multiple output locations for the same encoding process.

About this task

To specify multiple output locations, define multiple output elements in the XML configuration file. The output file type must be the same in each case (for example, .ec3). You can use the same output file name for each of the output locations.

Procedure

1. In the XML configuration file, specify the input and filter parameters.
2. Define a separate output element for each desired output location. Use the same output file type in each case. For example:

```
<output>
    <ec3>
        <file_name>FILE_NAME_1.ec3</file_name>
        <storage>
          <local>
            <path>PATH_1</path>
          </local>
        </storage>
    </ec3>
</output>
<output>
    <ec3>
        <file_name>FILE_NAME_2.ec3</file_name>
        <storage>
          <local>
            <path>PATH_2</path>
          </local>
        </storage>
    </ec3>
</output>
<output>
    <ec3>
        <file_name>FILE_NAME_1.ec3</file_name>
        <storage>
          <local>
            <path>PATH_3</path>
          </local>
        </storage>
    </ec3>
</output>
```

In the XML configuration file:

FILE_NAME_1 and *FILE_NAME_2* are different output file names for the same output file saved in multiple locations; you can use the same file name more than once.

PATH_1, *PATH_2*, and *PATH_3* are the multiple output location folder paths to save the output file.

3. Save the XML file.
4. Execute the encode command in the CLI, including the new XML file.

9.3 Automated naming of output files

The Dolby Encoding Engine can create output file names automatically based on the encoding parameters from the XML configuration file.

To automatically create output file names based on parameters from the XML configuration file, use this syntax:

$*function*(*variable*)

In this syntax:

function is one of the naming functions listed in this section.

variable is the name of the XML element whose value is the basis for the output file name; the variable can also be another function.

The available naming functions are:

var(*variable*)
Translates a colon-separated list of XML element names into the value of the last element in the list, for example:
$var(filter:audio:data_rate) inserts the value of the <data_rate> element.

base(var)
Strips the file extension, if present.

plain(var)
Makes a string more suitable for a file name by replacing all forbidden characters (\ / : * ? " < > |) with an underscore.

sequence()
Inserts a number that is incremented with each call. For example, the first call returns a 0, the second a 1, and so on.
Use the sequence() function only for batch processing, with more than one output file.

Within a single XML element, the functions are called from right to left.

The naming function syntax can be used as only a part of the output file name (for example, to add a prefix or suffix).

Example

```
<job_config>
    <input>
        <damf>
            <file_name>FILE_NAME</file_name>
            <storage>
```

```
              <local>
                <path>INPUT_PATH</path>
              </local>
            </storage>
        </damf>
    </input>
    <filter>
        <audio>
            <atmos_mezz_to_atmos_ddp>
                <data_rate>448</data_rate>
            </atmos_mezz_to_atmos_ddp>
        </audio>
    </filter>
    <output>
        <ec3>
            <file_name>$base($var(input:file_name))_encoded_at_
$var(filter:audio:data_rate)_kbps.ec3</file_name>
            <storage>
              <local>
                <path>OUTPUT_PATH</path>
              </local>
            </storage>
        </ec3>
    </output>
</job_config>
```

The XML configuration file above returns the following output file:
FILE_NAME_encoded_at_448_kbps.ec3

Related information

Multiple Dolby Vision data rates and resolutions on page 117

Using multiple input files on page 114

9.4 Multiple Dolby Vision data rates and resolutions

You can define multiple values for parameters such as data rate or resolution in a single Dolby Vision encoding job.

Multiple stream High-Efficiency Video Coding encoding

This process performs multiple HEVC encoding jobs using a single file produced by the Dolby Vision processor. This way, only the High-Efficiency Video Coding encoding step needs to be repeated, whereas Dolby Vision preprocessing pass runs only once.

The following Dolby Vision profile 5 encoding parameters can have multiple space-separated values:

- target_width
- target_height
- data_rate
- max_vbv_data_rate
- vbv_buffer_size

For example:

```
<target_width>3840 3840 1920 1920 1280 1280</target_width>
<target_height>2160 2160 1080 1080 720 720</target_height>
<data_rate>16000 10000 7500 5000 3500 2250</data_rate>
<max_vbv_data_rate>16000 10000 7500 5000 3500 2250</max_vbv_data_rate>
<vbv_buffer_size>32000 22500 12000 11000 10000 9000</vbv_buffer_size>
```

When specifying multiple target resolutions, be aware of these constraints:

- The list must start with the highest resolution.
- All resolutions must have the same aspect ratio.

If you decide to use multiple space-separated values for the data-rate and resolution parameters, use the same number of values in each case (as in the preceding example, where there are six values for each parameter). Note that you can repeat the values.

The number of generated streams is equal to the number of values. Based on the preceding example, the six generated streams are as listed in the following table.

Table 82: Example of multiple data-rate streams

Parameter name	Stream 1	Stream 2	Stream 3	Stream 4	Stream 5	Stream 6
target_width	3840	3840	1920	1920	1280	1280
target_height	2160	2160	1080	1080	720	720
data_rate	16000	10000	7500	5000	3500	2250
max_vbv_data_rate	16000	10000	7500	5000	3500	2250
vbv_buffer_size	32000	22500	12000	11000	10000	9000

File naming

Unless you produce a single multiplexed container output in this process, you must define an output file name for each of the generated streams. These output files can be named in one of two ways:

1. Manual naming: You can specify a name for each of the files, respective of their order in the stream.

   ```
   <file_name>FILE-1 FILE-2 FILE-3 ... FILE-N</file_name>
   ```

2. Automated naming: You can automatically generate file output names according to the parameters of each stream.

   ```
   <file_name>output_$var(target_width)x$var(target_height)_$var(data_rate)_
   $var(max_vbv_data_rate)_$var(vbv_buffer_size).h265</file_name>
   ```

   Note: The output file name must include all parameters with more than one value, using the var function, as in the preceding example.

   As a result, Dolby Encoding Engine generates these files:

   - output_3840x2160_16000_16000_32000.h265
   - output_3840x2160_10000_10000_22500.h265
   - output_1920x1080_7500_7500_12000.h265
   - output_1920x1080_5000_5000_11000.h265

- output_1280x720_3500_3500_10000.h265
- output_1280x720_2250_2250_9000.h265

You can find a complete XML configuration file example, multiple_resolutions_datarate_encoding.xml, in the xml_templates folder.

Related information

Dolby Vision profile 5 encoding job on page 42
Automated naming of output files on page 116

10 Multiplexing scenarios

You can multiplex audio and video input using Dolby Encoding Engine. Dolby Encoding Engine can perform multiplexing and encoding during a single job.

Dolby Encoding Engine also supports the following multiplexing scenarios:

- Dolby Vision encoding and audio pass-through to MP4 or transport stream
- Dolby Vision pass-through and audio encoding to MP4 or transport stream
- Dolby Vision pass-through and audio pass-through to MP4 or transport stream
- Dolby Vision encoding and audio encoding to MP4 or transport stream

Pass-through is used when the particular content is already encoded and is ready to be multiplexed into an MP4 or transport stream container. To use encoded content for pass-through, define it in the input section of the XML configuration file. In the following XML example, the video input is an already encoded HEVC file. Therefore, the HEVC input is not referenced in the filter section. In the filter section, define only the filters for the content to encode during the process. As output, define MP4 or transport stream.

For more information regarding the supported multiplexing scenarios, see the following XML examples in the xml_templates folder:

- adm_hevc_atmos_mezz_to_atmos_ddp_video_passthrough_mp4.xml
- damf_mxf_atmos_mezz_to_atmos_ddp_dv_mezz_to_dv_profile_5_mp4.xml
- damf_mxf_sidecar_atmos_mezz_to_atmos_ddp_dv_mezz_to_dv_profile_5_mp4.xml
- ec3_hevc_audio_passthrough_video_passthrough_mp4.xml
- ec3_j2k_sidecar_audio_passthrough_dv_mezz_to_dv_profile_5_mp4.xml
- ec3_mxf_audio_passthrough_dv_mezz_to_dv_profile_5_mp4.xml
- wav_list_j2k_sidecar_pcm_to_ddp_7_1_dv_mezz_to_dv_profile_5_mp4.xml
- wav_mxf_pcm_to_ddp_dv_mezz_to_dv_profile_5_mp4.xml

11 Audio editing job

With the audio editing filter (edit_ddp), you can trim and concatenate (join) Dolby Digital Plus and Dolby Digital Plus with Dolby Atmos content files.

- XML example file for audio editing
- Input parameters for audio editing
- Filter parameters for audio editing
- Output parameters for audio editing
- Editing audio files

The input can be trimmed, concatenated, or trimmed and concatenated in a single job. In each case, in the final stage of the job, the Dolby Encoding Engine checks whether the output stream is a valid Dolby Digital Plus stream.

Concatenation restrictions

When concatenating two files, follow these guidelines:

- Join files on a silent section.
- The core acmod of the files must match (for example, 5.1 or 7.1).
- You cannot join a file with Dolby Atmos with a file without Dolby Atmos.
- The independent substream (for example I0) and dependent substream (for example D0) layout must match.
- When joining two Dolby Atmos files, the Dolby Atmos cluster configuration must be the same, for example:
    - You can join a stream with LFE and 15 objects only with another stream with LFE and 15 objects.
    - You can join a 9.1.6 Dolby Atmos stream only with another 9.1.6 Dolby Atmos stream.

11.1 XML example file for audio editing

The XML template files are examples of a job configuration.

You can use them to create job configurations. You can find them in the xml_templates folder. For the audio editing job, refer to the following file: ec3_edit_ddp_ec3.xml.

11.2 Input parameters for audio editing

When creating an XML file, you can use several audio edit input parameters.

The supported input formats are Dolby Digital Plus and Dolby Digital Plus with Dolby Atmos content files.

The input element in the XML file can be either <ec3> or <generic>. You can use the <ec3> element with both Dolby Digital Plus and Dolby Digital Plus with Dolby Atmos content type of input. You can use the <generic> element with Dolby Digital Plus input files.

In the concatenation process, to define two or more input files, use the <generic> element.

Input parameters for the <ec3> element

Table 83: <ec3> element input parameters

Parameter	Type	Value	Description
<ec3>	XML element		Parent element.
<stream_type>	String	atmos, non_atmos Default: non_atmos	Type of Dolby Digital Plus stream.
<file_name>	String	*filename*	
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored input.
<path>	String	*folder_path*	Path to the directory that contains the input file (for example, <path>path/to/media</path>).

Parameters for the <generic> element

Table 84: The <generic> element input parameters

Parameter	Type	Value	Description
<generic>	XML element		Parent element
<file_name>	String	*filename*	Input file name.
<type>	String		Optional, minimal signaling of the input type.
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored input.
<path>	String	*folder_path*	Path to the directory that contains the input file (for example, <path>path/to/media</path>).

11.3 Filter parameters for audio editing

When creating an XML file, you can use several audio editing filter parameters.

Table 85: Audio editing filter parameters

Parameter	Type	Value	Description
<edit_ddp>	XML element		Parent element
<timecode_frame_rate>	String	not_indicated, 23.976, 24, 25, 29.97, 29.97df, 30 Default: not_indicated	Frame rate associated with a specified timecode.
<start>	String	*HH:MM:SS:FF* or *HH:MM:SS.xx*	Trim starting point (trims the frames before this point).
<end>	String	*HH:MM:SS:FF* or *HH:MM:SS.xx* or end_of_file Default: end_of_file	Trim ending point (trims the frames after this point).

Table 85: Audio editing filter parameters (continued)

Parameter	Type	Value	Description
`<order>`	String	`trim_join`, `join_trim` Default: `trim_join`	Order of operations: trim first or join first. This parameter impacts the interpretation of the `<start>` and `<end>` parameters: • `trim_join` first trims the two input files and then joins them • `join_trim` first joins the input files and then trims the newly created single file
`<invert_trim>`	Boolean	Default: `false`	When true, trimming removes audio frames within the specified time range instead of removing frames outside of (before and after) the specified time range.

11.4 Output parameters for audio editing

When creating an XML file, you can use several output parameters and values.

Output parameters for E-AC-3

Table 86: E-AC-3 output parameters

Parameter	Type	Value	Description
`<ec3>`	XML element		Parent element for an E-AC-3 output file.
`<file_name>`	String	*`filename`*	Output file name.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored output.
`<path>`	String	*`Storage folder path`*	Path to the directory to save the output file (for example, `<path>path/to/media</path>`).

11.5 Editing audio files

Depending on the parameters, the input can be trimmed, or concatenated, or trimmed and concatenated in a single job.

Procedure

The `edit_ddp` filter allows you to perform the following actions:

- To trim a file (without concatenation), specify only one input file and choose the desired time range using the `<start>` and `<end>` parameters. You can additionally use the `<invert_trim>` parameter - refer to the “Filter parameters for audio editing” section for details.
- To concatenate two files without trimming, set the trim `<start>` parameter to `0:0:0.0` and `<end>` to `end_of_file`.

- To trim and concatenate files in the same job, speicfy two input files and choose the desired trimming time range using the `<start>` and `<end>` parameters. The trimming time range applies to both of the input files. You can additionally use the `<order>` and `<invert_trim>` parameters - refer to the “Filter parameters for audio editing” section for details.

12 Atmos conversion job

With the Dolby Atmos conversion filter (`convert_atmos_mezz`), you can convert Dolby Atmos mezzanine formats, and optionally adjust their frame rate parameters.

Use this filter to convert between the Dolby Atmos mezzanine formats (for example, from a Dolby Atmos master file set to an ADM BWF). Optionally, use this filter for adjusting the mezzanine input to a different frame rate (for example, change the frame rate from 23.976 to 24).

The input as well as the output for this job is a Dolby Atmos master file set or an ADM BWF.

Table 87: Supported configurations of input and output

Input format	Output format
Dolby Atmos master file set	Dolby Atmos master file set
Dolby Atmos master file set	ADM BWF
ADM BWF	Dolby Atmos master file set
ADM BWF	ADM BWF

Note:

- If you use non-ASCII characters in the input paths and filenames on Windows, execute Dolby Encoding Engine with the `--no-links` option, for example:

  ```
  dee -x config.xml --no-links
  ```

- You must use an .atmos extension for Dolby Atmos master file set output and .wav for ADM BWF output.

12.1 XML example file for Dolby Atmos conversion job

The XML template files are examples of a job configuration.

You can use them to create job configurations. You can find them in the xml_templates folder. For the Dolby Atmos conversion job, refer to the following files:

- damf_convert_atmos_mezz_damf.xml
- damf_convert_atmos_mezz_adm.xml

12.2 Input parameters for Dolby Atmos conversion

When creating an XML file, you can use several Dolby Atmos conversion input parameters.

The input file format can be either Dolby Atmos master file set or ADM BWF.

Input parameters for Dolby Atmos master file set

Table 88: Input parameters for Dolby Atmos master file set

Parameter	Type	Value	Description
`<damf>`	XML element		Parent element for a Dolby Atmos master file set.
`<file_name>`	String	*filename*	Specifies the .damf or .atmos file.
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored input.
`<path>`	String	*folder_path*	Path to the directory that contains the input file set (for example, `<path>path/to/media</path>`).

Input parameters for ADM BWF

Table 89: Input parameters for ADM BWF files

Parameter	Type	Value	Description
`<adm>`	XML element		Parent element for ADM BWF files.
`<file_name>`	String	*filename*	
`<storage>`	XML element		Parent element for storage parameters. The only currently supported storage type is local.
`<local>`	XML element		Parent element for parameters of locally stored input.
`<path>`	String	*folder_path*	Path to the directory that contains the input file (for example, `<path>path/to/media</path>`).

12.3 Filter parameters for Dolby Atmos conversion

When creating an XML file, you can use several Dolby Atmos conversion filter parameters.

Table 90: Dolby Atmos conversion filter parameters

Parameter	Type	Value	Description
`<convert_atmos_mezz>`	XML element		Parent element for the Dolby Atmos conversion filter parameters.
`<source_frame_rate>`	String	auto, 23.976, 24, 25, 29.97, 29.97df, 30 Default: auto	Specifies the source frame rate. • If the input file contains an FPS indicator value, use the auto setting to retain it. • If the input file does not contain an FPS indicator value, type in manually the actual frame rate value of the source content. In this case auto is interpreted as 23.976. • You cannot overwrite an existing FPS indicator with a different value.

Table 90: Dolby Atmos conversion filter parameters (continued)

Parameter	Type	Value	Description
<target_frame_rate>	String	auto, 23.976, 24, 25, 29.97, 29.97df, 30 Default: auto	Specifies the desired target frame rate. Use auto to retain the source frame rate.
<ffoa>	String	*HH:MM:SS:FF* or *HH:MM:SS.xx* (timecode) or auto Default: auto	Specifies the first frame of action (FFOA). Add df for drop frame (for example, *HH:MM:SS:FF*df). Use auto to retain FFOA from the source.

12.4 Output parameters for Dolby Atmos conversion

When creating an XML file, you can use several output parameters and values.

The input file format can be either Dolby Atmos master file set or ADM BWF.

Output parameters for Dolby Atmos master file set

Table 91: Output parameters for Dolby Atmos master file set

Parameter	Type	Value	Description
<damf>	XML element		Parent element for a Dolby Atmos master file set.
<file_name>	String	*filename*	Specifies the .atmos file.
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored output.
<path>	String	*folder_path*	Path to the directory that contains the output file set (for example, <path>path/to/media</path>).

Output parameters for ADM BWF

Table 92: Output parameters for ADM BWF files

Parameter	Type	Value	Description
<adm>	XML element		Parent element for ADM BWF files.
<file_name>	String	*filename*	
<storage>	XML element		Parent element for storage parameters. The only currently supported storage type is local.
<local>	XML element		Parent element for parameters of locally stored output.
<path>	String	*folder_path*	Path to the directory that contains the output file set (for example, <path>path/to/media</path>).

13 Reference

Concept definition.

13.1 XML configuration file

A set of rules describes the XML configuration file, which is required to perform an encoding job with the Dolby Encoding Engine.

For each encoding process (encoding job), you need to execute a CLI command in which you specify the location of an XML configuration file. In the XML configuration file, you need to include all of the encoding job parameters (unless you want to keep the default values).

To create the XML file, you can use the following information:

- The parameters for encoding jobs described in the Operation chapter
- The XML reference chapter
- The XML template files provided with your Dolby Encoding Engine (in the xml_templates subfolder)
- The XSD file that you can generate using the Dolby Encoding Engine

Related information

Operation on page 16

Executing an encoding job on page 16

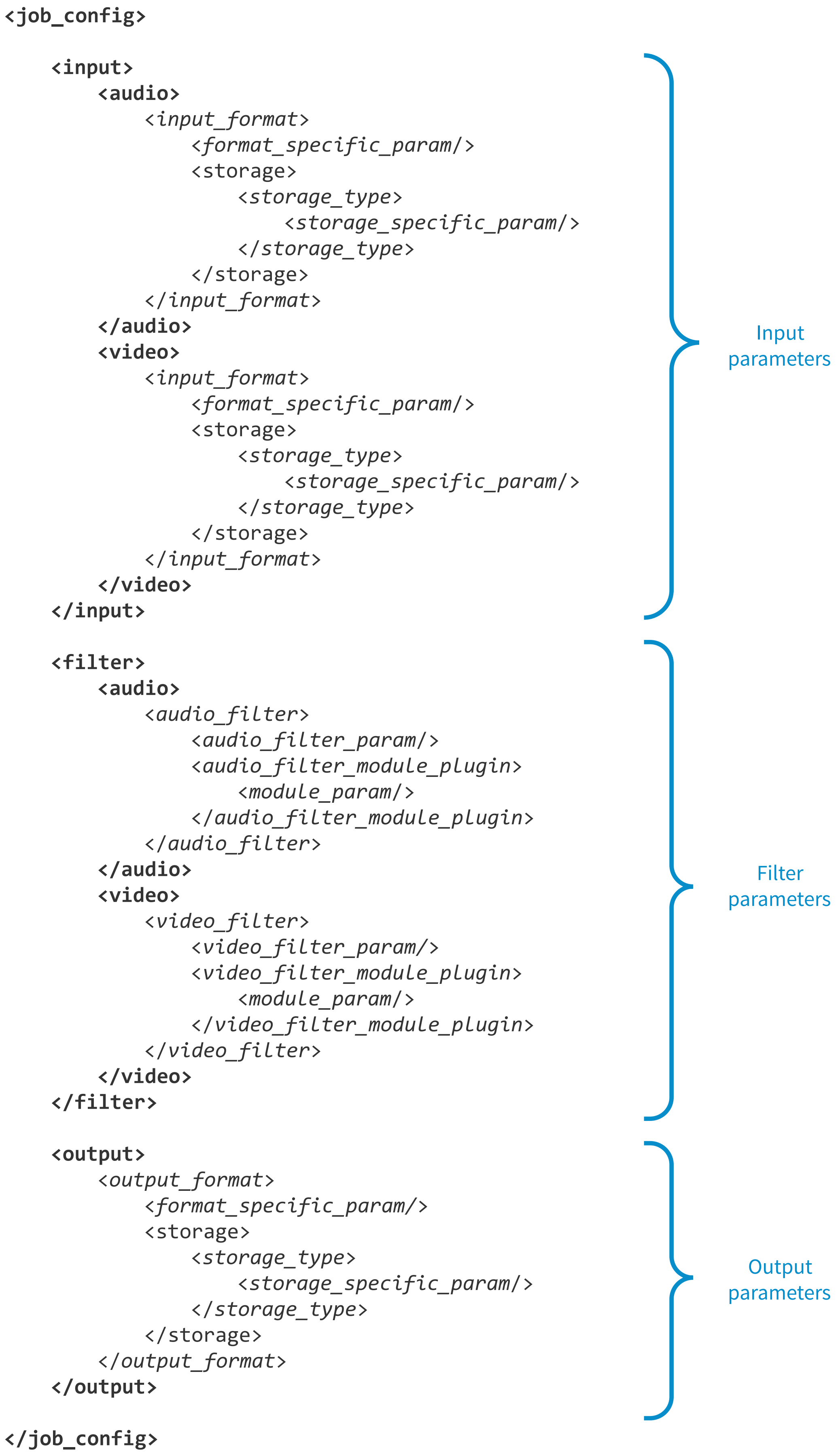
Encoding parameters on page 20

13.1.1 XML file structure

The XML configuration file defines the parameters for the Dolby Encoding Engine processing stages: input, filter, and output.

The XML configuration file consists of the elements `<input/>`, `<filter/>`, and `<output/>`, which include the parameters for the corresponding Dolby Encoding Engine processing stages.

Figure 13: XML configuration file structure



```
<job_config>

    <input>
        <audio>
            <input_format>
                <format_specific_param/>
                <storage>
                    <storage_type>
                        <storage_specific_param/>
                    </storage_type>
                </storage>
            </input_format>
        </audio>
        <video>
            <input_format>
                <format_specific_param/>
                <storage>
                    <storage_type>
                        <storage_specific_param/>
                    </storage_type>
                </storage>
            </input_format>
        </video>
    </input>

    <filter>
        <audio>
            <audio_filter>
                <audio_filter_param/>
                <audio_filter_module_plugin>
                    <module_param/>
                </audio_filter_module_plugin>
            </audio_filter>
        </audio>
        <video>
            <video_filter>
                <video_filter_param/>
                <video_filter_module_plugin>
                    <module_param/>
                </video_filter_module_plugin>
            </video_filter>
        </video>
    </filter>

    <output>
        <output_format>
            <format_specific_param/>
            <storage>
                <storage_type>
                    <storage_specific_param/>
                </storage_type>
            </storage>
        </output_format>
    </output>

</job_config>
```

13.1.2 Navigating to XML example files

The XML example files help you define Dolby Encoding Engine tasks by modifying the content of the files.

About this task

The Dolby Encoding Engine application folder includes a set of XML example files with predefined processing jobs. By modifying the content of those files, you can create new processing instructions for Dolby Encoding Engine. You can also use those files as a template to build new XML files.

Procedure

In the Dolby Encoding Engine installation folder, navigate to the xml_templates folder.

13.1.3 XSD file structure

You can generate an XSD file from Dolby Encoding Engine to see the available parameters and their structure in the XML file.

Dolby Encoding Engine operates based on instructions written in an XML file. The XSD (XML schema definition) file that you can generate with Dolby Encoding Engine specifies how to formally describe the elements in the XML file. In the XSD file, you can check the available plugins and parameters and the structure of the elements for the XML file.

To understand how XSD defines the elements and structure of the XML file, see the following example.

The following code shows a fragment of an XSD file.

```
<xs:element name="atmos_mezz_to_atmos_ddp">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="loudness" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="metering_mode" default="1770-4">
              <xs:simpleType>
                <xs:restriction base="xs:string">
                  <xs:enumeration value="1770-4"/>
                  <xs:enumeration value="1770-3"/>
                  <xs:enumeration value="1770-2"/>
                  <xs:enumeration value="1770-1"/>
                  <xs:enumeration value="LeqA"/>
                </xs:restriction>
              </xs:simpleType>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

This code defines the <atmos_mezz_to_atmos_ddp>, <loudness>, and <metering_mode> elements. It also defines their nesting. The <loudness> element is optional. The <metering_mode> element has a list of acceptable string values and a default value.

The corresponding XML structure is as follows:

```
<atmos_mezz_to_atmos_ddp>
    <loudness>
        <metering_mode>1770-3</metering_mode>
    </loudness>
</atmos_mezz_to_atmos_ddp>
```

Alternatively, using the XSD file, you can also generate human-readable documentation of an XML file structure. To do this, you need third-party software not referenced in this documentation.

13.1.4 Generating the XSD file

Generate the XSD file to check the available parameters and their structure in the XML file.

Procedure

To generate the XSD file, on the command line, enter: `dee -s` *path/file_name*`.xsd`
The XSD file appears in the specified folder.

13.2 Input requirements for SDR to Dolby Vision encoding

In the case of SDR to Dolby Vision encoding (for example, Dolby Vision profile 8.2), the HEVC input must conform to the certain requirements.

Related information

SDR to Dolby Vision profile 8 encoding job on page 82
HEVC transcoding job on page 97

13.2.1 HEVC requirements

HEVC bitstreams must conform to the base layer requirements for SDR to Dolby Vision encoding.

Table 93: HEVC requirements

General	• Bitstreams generated by an H.265-compatible encoder must conform to ITU-T H.265 and support up to Main 10 Profile Level 5.1. • Supports H.265 elementary stream in annex B format (with start codes). • The bitstream shall include delimiter Network Abstraction Layer (NAL) units.
Profile	H.265 Main 10

Table 93: HEVC requirements (continued)

Level	5.1 (max)
supplemental enhancement information	• The presence of a buffering period and a picture timing supplemental enhancement information message for each picture in the coded bitstream is required. • The presence of a buffering-period supplemental enhancement information message for each buffering period in the coded bitstream is required. • For the H.265 elementary stream, the concatenation_flag field of the buffering period supplemental enhancement information message shall be set to 1 for the first buffering period supplemental enhancement information message of all segments, except for the first segment when concatenating video elementary streams. • The presence of a picture-timing supplemental enhancement information message for each picture in the coded bitstream is required. • delimiters must be included in the bitstream.

13.2.2 Frame rate, bit rate, and resolution requirements

H.265 bitstreams must conform to the resolutions supported by target devices.

The frame rate and bit rate must conform to the constraints specified within the *Dolby Vision Profiles and Levels Specifications for Dolby Vision*.

Table 94: Resolution requirements

Target device resolutions	• 3840×2160 • 2560×1440 • 1920×1080 • 1280×720 • 720×480 • 640×480
Constraints for Dolby Vision profile 4, if target resolutions cannot be met	The ratio of picture resolution between the base layer and enhancement layer is 1:¼, which is half the size in both horizontal and vertical directions. In the case of 4:2:0 sampling in video compression, this implies the height and width of the base layer to be even numbers. Therefore, the height and width of the base layer must be multiples of four.

The following table is an extract from the *Dolby Vision Profiles and Levels Specifications for Dolby Vision*.

Table 95: Dolby Vision levels, frame rates, and bit rates

Level ID	Level name	Example maximum resolution x frame	Maximum bit rates (base layer and enhancement layer combined)	
			Main tier (Mbps)	High tier (Mbps)
1	hd24	1280×720 × 24 fps	20	50
2	hd30	1280×720 × 30 fps	20	50

Table 95: Dolby Vision levels, frame rates, and bit rates (continued)

Level ID	Level name	Example maximum resolution x frame	Maximum bit rates (base layer and enhancement layer combined)	
			Main tier (Mbps)	High tier (Mbps)
3	fhd24	1920×1080 × 24 fps	20	70
4	fhd30	1920×1080 × 30 fps *	20	70
5	fhd60	1920×1080 × 60 fps †	20	70
6	uhd24	3840×2160 × 24 fps	25	130
7	uhd30	3840×2160 × 30 fps *	25	130
8	uhd48	3840×2160 × 48 fps	40	130
9	uhd60	3840×2160 × 60 fps †	40	130

* This level also supports 25 fps frame rates at this resolution.

† This level also supports 50 fps frame rates at this resolution.

13.3 Error codes

Table 96: Dolby Encoding Engine error codes

Error code	Description	Description
1	License error	• Could not find license file • Could not open license file • License integrity check failed (license modified) • Error while reading license: a broken license; fall back from one of the preceding)
2	CLI error	• Incorrectly formatted CLI • Missing parameters
3	CLI option error	Unknown option
5	File open error	Cannot open file
10	Runtime error	For explanation, look at the detailed log messages
11	Critical error	Unhandled exception

14 Appendix A: Open-source and third-party software declarations

- Open-source software declarations
- Third party software declarations

14.1 Open-source software declarations

The Dolby Encoding Engine uses code from the following open-source software libraries:

ASDCPLib license

BBC Audio Toolbox license

Boost license

libpng license

There is no warranty against interference with your enjoyment of the library or against infringement. There is no warranty that our efforts or the library will fulfill any of your particular purposes or needs. This library is provided with all faults, and the entire risk of satisfactory quality, performance, accuracy, and effort is with the user.

Some files in the "contrib" directory and some configure-generated files that are distributed with libpng have other copyright owners and are released under other open source licenses.

ibpng versions 0.97, January 1998, through 1.0.6, March 20, 2000, are Copyright (c) 1998-2000 Glenn Randers-Pehrson, are derived from libpng-0.96, and are distributed according to the same disclaimer and license as libpng-0.96, with the following individuals added to the list of Contributing Authors:

Tom Lane
Glenn Randers-Pehrson
Willem van Schaik

libpng versions 0.89, June 1996, through 0.96, May 1997, are Copyright (c) 1996-1997 Andreas Dilger, are derived from libpng-0.88, and are distributed according to the same disclaimer and license as libpng-0.88, with the following individuals added to the list of Contributing Authors:

John Bowler
Kevin Bracey
Sam Bushell
Magnus Holmgren
Greg Roelofs
Tom Tanner

Some files in the "scripts" directory have other copyright owners but are released under this license.

libpng versions 0.5, May 1995, through 0.88, January 1996, are Copyright (c) 1995-1996 Guy Eric Schalnat, Group 42, Inc.

For the purposes of this copyright and license, "Contributing Authors" is defined as the following set of individuals:

Andreas Dilger
Dave Martindale
Guy Eric Schalnat
Paul Schmidt
Tim Wegner

The PNG Reference Library is supplied "AS IS". The Contributing Authors and Group 42, Inc. disclaim all warranties, expressed or implied, including, without limitation, the warranties of merchantability and of fitness for any purpose. The Contributing Authors and Group 42, Inc. assume no liability for direct, indirect, incidental, special, exemplary, or consequential damages, which may result from the use of the PNG Reference Library, even if advised of the possibility of such damage.

Permission is hereby granted to use, copy, modify, and distribute this source code, or portions hereof, for any purpose, without fee, subject to the following restrictions:

1. The origin of this source code must not be misrepresented.
2. Altered versions must be plainly marked as such and must not be misrepresented as being the original source.
3. This Copyright notice may not be removed or altered from any source or altered source distribution.

The Contributing Authors and Group 42, Inc. specifically permit, without fee, and encourage the use of this source code as a component to supporting the PNG file format in commercial products. If you use this source code in a product, acknowledgment is not required but would be appreciated.

LibTiff license

Copyright (c) 1988-1997 Sam Leffler

Copyright (c) 1991-1997 Silicon Graphics, Inc.

Permission to use, copy, modify, distribute, and sell this software and its documentation for any purpose is hereby granted without fee, provided that (i) the above copyright notices and this permission notice appear in all copies of the software and related documentation, and (ii) the names of Sam Leffler and Silicon Graphics may not be used in any advertising or publicity relating to the software without the specific, prior written permission of Sam Leffler and Silicon Graphics.

THE SOFTWARE IS PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EXPRESS, IMPLIED OR OTHERWISE, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

IN NO EVENT SHALL SAM LEFFLER OR SILICON GRAPHICS BE LIABLE FOR ANY SPECIAL, INCIDENTAL, INDIRECT OR CONSEQUENTIAL DAMAGES OF ANY KIND, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER OR NOT ADVISED OF THE POSSIBILITY OF DAMAGE, AND ON ANY THEORY OF LIABILITY, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

MXFLib license

Copyright (c) 2002-2007, Matt Beard, BBC R&D and Metaglue Corporation

This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.
2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.
3. This notice may not be removed or altered from any source distribution.

nanomsg license

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN

AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

"nanomsg" is a trademark of Martin Sustrik

OpenJPEG license

The copyright in this software is being made available under the 2-clauses BSD License, included below. This software may be subject to other third party and contributor rights, including patent rights, and no such rights are granted under this license.

Copyright (c) 2002-2014, Universite catholique de Louvain (UCL), Belgium

Copyright (c) 2002-2014, Professor Benoit Macq

Copyright (c) 2003-2014, Antonin Descampe

Copyright (c) 2003-2009, Francois-Olivier Devaux

Copyright (c) 2005, Herve Drolon, FreeImage Team

Copyright (c) 2002-2003, Yannick Verschueren

Copyright (c) 2001-2003, David Janssens

Copyright (c) 2011-2012, Centre National d'Etudes Spatiales (CNES), France

Copyright (c) 2012, CS Systemes d'Information, France

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS `AS IS' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

OpenSSL license

Copyright (c) 1998-2016 The OpenSSL Project. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. All advertising materials mentioning features or use of this software must display the following acknowledgment: "This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit. (http://www.openssl.org/)"
4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact openssl-core@openssl.org.
5. Products derived from this software may not be called "OpenSSL" nor may "OpenSSL" appear in their names without prior written permission of the OpenSSL Project.
6. Redistributions of any form whatsoever must retain the following acknowledgment:

   "This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/)"

THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT "AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This product includes cryptographic software written by Eric Young (eay@cryptsoft.com). This product includes software written by Tim Hudson (tjh@cryptsoft.com).

Original SSLeay license

Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com)

All rights reserved.

This package is an SSL implementation written by Eric Young (eay@cryptsoft.com). The implementation was written so as to conform with Netscapes SSL.

This library is free for commercial and noncommercial use as long as the following conditions are adhered to. The following conditions apply to all code found in this distribution, be it the RC4, RSA, lhash, DES, etc., code; not just the SSL code. The SSL documentation included with this distribution is covered by the same copyright terms except that the holder is Tim Hudson (tjh@cryptsoft.com).

Copyright remains Eric Young's, and as such any Copyright notices in the code are not to be removed. If this package is used in a product, Eric Young should be given attribution as the author of the parts of the library used. This can be in the form of a textual message at program startup or in documentation (online or textual) provided with the package.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must display the following acknowledgment:

"This product includes cryptographic software written by Eric Young (eay@cryptsoft.com)"

The word 'cryptographic' can be left out if the routines from the library being used are not cryptographic related.

4. If you include any Windows specific code (or a derivative thereof) from the apps directory (application code) you must include an acknowledgment:

   "This product includes software written by Tim Hudson (tjh@cryptsoft.com)"

THIS SOFTWARE IS PROVIDED BY ERIC YOUNG "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The license and distribution terms for any publicly available version or derivative of this code cannot be changed, i.e., this code cannot simply be copied and put under another distribution license (including the GNU Public License).

pyinstaller license

Copyright (c) 2010-2018, PyInstaller Development Team

Copyright (c) 2005-2009, Giovanni Bajo

Based on previous work under copyright (c) 2002 McMillan Enterprises, Inc.

PyInstaller is licensed under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

Bootloader Exception:

In addition to the permissions in the GNU General Public License, the authors give you unlimited permission to link or embed compiled bootloader and related files into combinations with other programs, and to distribute those combinations without any restriction coming from the use of those files. (The General Public License restrictions do apply in other respects; for example, they cover modification of the files, and distribution when not linked into a combine executable.)

Bootloader and Related Files:

Bootloader and related files are files which are embedded within the final executable. This includes files in directories:

```
./bootloader/
./PyInstaller/loader
```

About the PyInstaller Development Team:

The PyInstaller Development Team is the set of contributors to the PyInstaller project. A full list with details is kept in the documentation directory, in the file ``doc/CREDITS.rst``.

The core team that coordinates development on GitHub can be found here: https://github.com/pyinstaller/pyinstaller. As of 2015, it consists of:

- Hartmut Goebel
- Martin Zibricky

- David Vierra
- David Cortesi

Our Copyright Policy:

PyInstaller uses a shared copyright model. Each contributor maintains copyright over their contributions to PyInstaller. But, it is important to note that these contributions are typically only changes to the repositories. Thus, the PyInstaller source code, in its entirety is not the copyright of any single person or institution. Instead, it is the collective copyright of the entire PyInstaller Development Team. If individual contributors want to maintain a record of what changes/contributions they have specific copyright on, they should indicate their copyright in the commit message of the change, when they commit the change to the PyInstaller repository.

With this in mind, the following banner should be used in any source code file to indicate the copyright and license terms:

RapidJSON license

The Dolby Encoding Engine uses code from RapidJSON (https://github.com/miloyip/rapidjson) under the MIT license:

Xerces-C++ license

yaml-cpp license

14.2 Third party software declarations

The Dolby Encoding Engine uses code from the following third party software.

Ittiam tsmuxer

For the Ittiam tsmuxer license, go to the Ittiam Systems Incorporated web page: https://www.ittiam.com

14.2.1 Plugin pack third party software declarations

The plugin pack version of the Dolby Encoding Engine uses code from the following third party software.

Apple ProRes

For details about the Apple ProRes license, go to: https://support.apple.com/en-us/HT200321

Kakadu

For details about the Kakadu license, go to: http://kakadusoftware.com/licence/

Glossary

ADM

Audio Definition Model. A metadata specification for describing channel-based, object-based, or scene-based audio.

ADM BWF

Audio Definition Model Broadcast Wave Format.

CLI

Command-line interface.

dialnorm

Dialogue normalization value.

Dolby Atmos Metadata Format

A format used by Dolby Atmos to store the metadata needed for processing.

dynamic range control

An audio compression metadata parameter applied to audio to limit the dynamic range.

elementary stream

A bitstream that is the output of an audio or video encoder and contains only one type of data, such as audio or video.

H.265

Also known as High Efficiency Video Coding (HEVC), ISO/IEC 23008-2, and ITU-T H.265. An MPEG standard for video compression that improves on the H.264 (AVC) video compression standard and extends support to 10-bit ultra-high-definition video. The standard was developed jointly by the Moving Picture Experts Group (MPEG) and Video Coding Experts Group (VCEG).

HDR10

An open-source video format that is characterized by certain properties, such as bit depth, color primaries, metadata, and other factors.

HEVC

High-Efficiency Video Coding. See H.265 on page 143.

JP2

JPEG 2000. An image compression standard that uses the .jp2 extension.

LFE

Low-Frequency Effects. A band-limited channel specifically intended for deep, low-pitched sounds.

LUT

Look-up table. In the context of Dolby Vision, a look-up table is a series of precalcuated display management data stored in a static text file. The LUT saves processing time and cycles for display management.

MEL
: Minimal enhancement layer.

MP4
: A digital multimedia container format defined in MPEG-4 (ISO/IEC-14496, parts 12 and 14) most commonly used to store video and audio streams.

MPEG-4
: An MPEG standard (ISO/IEC 14496) for a group of audio and video coding formats and related technologies.

MXF
: Material Exchange Format. A file format used to transfer and store different types of content (for example, audio, video, data, or metadata). MXF currently supports various compression and encoding formats, and its specification can be extended to new essence formats, if needed.

NAL
: Network Abstraction Layer.

reference picture unit
: A container for Dolby Vision metadata that is encoded within a Dolby Vision bitstream. Each frame in a Dolby Vision bitstream contains one reference picture unit.

SDR
: Standard dynamic range. A Rec. 709 signal with peak luminance of 100 cd/m².

SEI
: Supplemental enhancement information. Data unit that carries supplemental video information about decoding or display, introduced in H.264.

transport stream
: As defined in ISO/IEC 13818-1, a packetized bitstream that is used to transmit audio and video information. A transport stream is made up of multiplexed program elementary streams.