## NAME

ffmpeg-bitstream-filters - FFmpeg bitstream filters

## DESCRIPTION

This document describes the bitstream filters provided by the libavcodec library.

A bitstream filter operates on the encoded stream data, and performs bitstream level modifications without performing decoding.

## BITSTREAM FILTERS

When you configure your FFmpeg build, all the supported bitstream filters are enabled by default. You can list all available ones using the configure option "--list-bsfs".

You can disable all the bitstream filters using the configure option "--disable-bsfs", and selectively enable any bitstream filter using the option "--enable-bsf=BSF", or you can disable a particular bitstream filter using the option "--disable-bsf=BSF".

The option "-bsfs" of the ff* tools will display the list of all the supported bitstream filters included in your build.

The ff* tools have a -bsf option applied per stream, taking a comma-separated list of filters, whose parameters follow the filter name after a ’=’.

```
ffmpeg -i INPUT -c:v copy -bsf:v filter1[=opt1=str1:opt2=str2][,filter2] OUTPUT
```

Below is a description of the currently available bitstream filters, with their parameters, if any.

### aac_adtstoasc

Convert MPEG-2/4 AAC ADTS to an MPEG-4 Audio Specific Configuration bitstream.

This filter creates an MPEG-4 AudioSpecificConfig from an MPEG-2/4 ADTS header and removes the ADTS header.

This filter is required for example when copying an AAC stream from a raw ADTS AAC or an MPEG-TS container to MP4A-LATM, to an FLV file, or to MOV/MP4 files and related formats such as 3GP or M4A. Please note that it is auto-inserted for MP4A-LATM and MOV/MP4 and related formats.

### av1_metadata

Modify metadata embedded in an AV1 stream.

**td** Insert or remove temporal delimiter OBUs in all temporal units of the stream.

**insert**
Insert a TD at the beginning of every TU which does not already have one.

**remove**
Remove the TD from the beginning of every TU which has one.

**color_primaries**
**transfer_characteristics**
**matrix_coefficients**
Set the color description fields in the stream (see AV1 section 6.4.2).

**color_range**
Set the color range in the stream (see AV1 section 6.4.2; note that this cannot be set for streams using BT.709 primaries, sRGB transfer characteristic and identity (RGB) matrix coefficients).

**tv** Limited range.

**pc** Full range.

**chroma_sample_position**
Set the chroma sample location in the stream (see AV1 section 6.4.2). This can only be set for 4:2:0 streams.

**vertical**
Left position (matching the default in MPEG-2 and H.264).

**colocated**
Top-left position.

**tick_rate**
Set the tick rate (*time_scale / num_units_in_display_tick*) in the timing info in the sequence header.

**num_ticks_per_picture**
Set the number of ticks in each picture, to indicate that the stream has a fixed framerate. Ignored if **tick_rate** is not also set.

**delete_padding**

Deletes Padding OBUs.

**chomp**

Remove zero padding at the end of a packet.

**dca_core**

Extract the core from a DCA/DTS stream, dropping extensions such as DTS-HD.

**dump_extra**

Add extradata to the beginning of the filtered packets except when said packets already exactly begin with the extradata that is intended to be added.

**freq** The additional argument specifies which packets should be filtered. It accepts the values:

**k**
**keyframe**
add extradata to all key packets

**e**
**all** add extradata to all packets

If not specified it is assumed **k**.

For example the following **ffmpeg** command forces a global header (thus disabling individual packet headers) in the H.264 packets generated by the "libx264" encoder, but corrects them by adding the header stored in extradata to the key packets:

```
ffmpeg -i INPUT -map 0 -flags:v +global_header -c:v libx264 -bsf:v dump_extra out.ts
```

**dv_error_marker**

Blocks in DV which are marked as damaged are replaced by blocks of the specified color.

**color**
The color to replace damaged blocks by

**sta** A 16 bit mask which specifies which of the 16 possible error status values are to be replaced by colored blocks. 0xFFFE is the default which replaces all non 0 error status values.

**ok** No error, no concealment

**err** Error, No concealment

**res** Reserved

**notok**
Error or concealment

**notres**
Not reserved

**Aa, Ba, Ca, Ab, Bb, Cb, A, B, C, a, b, erri, erru**
The specific error status code

see page 44-46 or section 5.5 of
<**http://web.archive.org/web/20060927044735/http://www.smpte.org/smpte_store/standards/pdf/s314m.pdf**>

**eac3_core**
Extract the core from a E-AC-3 stream, dropping extra channels.

**extract_extradata**
Extract the in-band extradata.

Certain codecs allow the long-term headers (e.g. MPEG-2 sequence headers, or H.264/HEVC (VPS/)SPS/PPS) to be transmitted either "in-band" (i.e. as a part of the bitstream containing the coded frames) or "out of band" (e.g. on the container level). This latter form is called "extradata" in FFmpeg terminology.

This bitstream filter detects the in-band headers and makes them available as extradata.

**remove**
When this option is enabled, the long-term headers are removed from the bitstream after extraction.

**filter_units**
Remove units with types in or not in a given set from the stream.

**pass_types**
List of unit types or ranges of unit types to pass through while removing all others. This is specified as a '|'-separated list of unit type values or ranges of values with '-'.

**remove_types**

Identical to **pass_types**, except the units in the given set removed and all others passed through.

Extradata is unchanged by this transformation, but note that if the stream contains inline parameter sets then the output may be unusable if they are removed.

For example, to remove all non-VCL NAL units from an H.264 stream:

```
ffmpeg -i INPUT -c:v copy -bsf:v 'filter_units=pass_types=1-5' OUTPUT
```

To remove all AUDs, SEI and filler from an H.265 stream:

```
ffmpeg -i INPUT -c:v copy -bsf:v 'filter_units=remove_types=35|38-40' OUTPUT
```

**hapqa_extract**

Extract Rgb or Alpha part of an HAPQA file, without recompression, in order to create an HAPQ or an HAPAlphaOnly file.

**texture**

Specifies the texture to keep.

**color**
**alpha**

Convert HAPQA to HAPQ

```
ffmpeg -i hapqa_inputfile.mov -c copy -bsf:v hapqa_extract=texture=color -tag:v HapY -metadata:s:v:0 encoder='
```

Convert HAPQA to HAPAlphaOnly

```
ffmpeg -i hapqa_inputfile.mov -c copy -bsf:v hapqa_extract=texture=alpha -tag:v HapA -metadata:s:v:0 encoder=
```

**h264_metadata**

Modify metadata embedded in an H.264 stream.

**aud** Insert or remove AUD NAL units in all access units of the stream.

**pass**
**insert**
**remove**

Default is pass.

**sample_aspect_ratio**
Set the sample aspect ratio of the stream in the VUI parameters. See H.264 table E-1.

**overscan_appropriate_flag**
Set whether the stream is suitable for display using overscan or not (see H.264 section E.2.1).

**video_format**
**video_full_range_flag**
Set the video format in the stream (see H.264 section E.2.1 and table E-2).

**colour_primaries**
**transfer_characteristics**
**matrix_coefficients**
Set the colour description in the stream (see H.264 section E.2.1 and tables E-3, E-4 and E-5).

**chroma_sample_loc_type**
Set the chroma sample location in the stream (see H.264 section E.2.1 and figure E-1).

**tick_rate**
Set the tick rate (time_scale / num_units_in_tick) in the VUI parameters. This is the smallest time unit representable in the stream, and in many cases represents the field rate of the stream (double the frame rate).

**fixed_frame_rate_flag**
Set whether the stream has fixed framerate - typically this indicates that the framerate is exactly half the tick rate, but the exact meaning is dependent on interlacing and the picture structure (see H.264 section E.2.1 and table E-6).

**zero_new_constraint_set_flags**
Zero constraint_set4_flag and constraint_set5_flag in the SPS. These bits were reserved in a previous version of the H.264 spec, and thus some hardware decoders require these to be zero. The result of zeroing this is still a valid bitstream.

**crop_left**
**crop_right**
**crop_top**
**crop_bottom**
Set the frame cropping offsets in the SPS. These values will replace the current ones if the stream

is already cropped.

These fields are set in pixels. Note that some sizes may not be representable if the chroma is subsampled or the stream is interlaced (see H.264 section 7.4.2.1.1).

**sei_user_data**

Insert a string as SEI unregistered user data. The argument must be of the form *UUID+string*, where the UUID is as hex digits possibly separated by hyphens, and the string can be anything.

For example, **086f3693-b7b3-4f2c-9653-21492feee5b8+hello** will insert the string ‘‘hello’’ associated with the given UUID.

**delete_filler**

Deletes both filler NAL units and filler SEI messages.

**display_orientation**

Insert, extract or remove Display orientation SEI messages. See H.264 section D.1.27 and D.2.27 for syntax and semantics.

**pass**
**insert**
**remove**
**extract**

Default is pass.

Insert mode works in conjunction with "rotate" and "flip" options. Any pre-existing Display orientation messages will be removed in insert or remove mode. Extract mode attaches the display matrix to the packet as side data.

**rotate**

Set rotation in display orientation SEI (anticlockwise angle in degrees). Range is -360 to +360. Default is NaN.

**flip** Set flip in display orientation SEI.

**horizontal**
**vertical**

Default is unset.

**level**

Set the level in the SPS. Refer to H.264 section A.3 and tables A-1 to A-5.

The argument must be the name of a level (for example, **4.2**), a level_idc value (for example, **42**), or the special name **auto** indicating that the filter should attempt to guess the level from the input stream properties.

**h264_mp4toannexb**

Convert an H.264 bitstream from length prefixed mode to start code prefixed mode (as defined in the Annex B of the ITU-T H.264 specification).

This is required by some streaming formats, typically the MPEG-2 transport stream format (muxer "mpegts").

For example to remux an MP4 file containing an H.264 stream to mpegts format with **ffmpeg**, you can use the command:

```
ffmpeg -i INPUT.mp4 -codec copy -bsf:v h264_mp4toannexb OUTPUT.ts
```

Please note that this filter is auto-inserted for MPEG-TS (muxer "mpegts") and raw H.264 (muxer "h264") output formats.

**h264_redundant_pps**

This applies a specific fixup to some Blu-ray streams which contain redundant PPSs modifying irrelevant parameters of the stream which confuse other transformations which require correct extradata.

**hevc_metadata**

Modify metadata embedded in an HEVC stream.

**aud** Insert or remove AUD NAL units in all access units of the stream.

**insert**

**remove**

**sample_aspect_ratio**

Set the sample aspect ratio in the stream in the VUI parameters.

**video_format**

**video_full_range_flag**

Set the video format in the stream (see H.265 section E.3.1 and table E.2).

**colour_primaries**
**transfer_characteristics**
**matrix_coefficients**
Set the colour description in the stream (see H.265 section E.3.1 and tables E.3, E.4 and E.5).

**chroma_sample_loc_type**
Set the chroma sample location in the stream (see H.265 section E.3.1 and figure E.1).

**tick_rate**
Set the tick rate in the VPS and VUI parameters (time_scale / num_units_in_tick). Combined with **num_ticks_poc_diff_one**, this can set a constant framerate in the stream. Note that it is likely to be overridden by container parameters when the stream is in a container.

**num_ticks_poc_diff_one**
Set poc_proportional_to_timing_flag in VPS and VUI and use this value to set num_ticks_poc_diff_one_minus1 (see H.265 sections 7.4.3.1 and E.3.1). Ignored if **tick_rate** is not also set.

**crop_left**
**crop_right**
**crop_top**
**crop_bottom**
Set the conformance window cropping offsets in the SPS. These values will replace the current ones if the stream is already cropped.

These fields are set in pixels. Note that some sizes may not be representable if the chroma is subsampled (H.265 section 7.4.3.2.1).

**level**
Set the level in the VPS and SPS. See H.265 section A.4 and tables A.6 and A.7.

The argument must be the name of a level (for example, **5.1**), a *general_level_idc* value (for example, **153** for level 5.1), or the special name **auto** indicating that the filter should attempt to guess the level from the input stream properties.

**hevc_mp4toannexb**
Convert an HEVC/H.265 bitstream from length prefixed mode to start code prefixed mode (as defined in the Annex B of the ITU-T H.265 specification).

This is required by some streaming formats, typically the MPEG-2 transport stream format (muxer

"mpegts").

For example to remux an MP4 file containing an HEVC stream to mpegts format with **ffmpeg**, you can use the command:

```
ffmpeg -i INPUT.mp4 -codec copy -bsf:v hevc_mp4toannexb OUTPUT.ts
```

Please note that this filter is auto-inserted for MPEG-TS (muxer "mpegts") and raw HEVC/H.265 (muxer "h265" or "hevc") output formats.

**imxdump**

Modifies the bitstream to fit in MOV and to be usable by the Final Cut Pro decoder. This filter only applies to the mpeg2video codec, and is likely not needed for Final Cut Pro 7 and newer with the appropriate **-tag:v**.

For example, to remux 30 MB/sec NTSC IMX to MOV:

```
ffmpeg -i input.mxf -c copy -bsf:v imxdump -tag:v mx3n output.mov
```

**mjpeg2jpeg**

Convert MJPEG/AVI1 packets to full JPEG/JFIF packets.

MJPEG is a video codec wherein each video frame is essentially a JPEG image. The individual frames can be extracted without loss, e.g. by

```
ffmpeg -i ../some_mjpeg.avi -c:v copy frames_%d.jpg
```

Unfortunately, these chunks are incomplete JPEG images, because they lack the DHT segment required for decoding. Quoting from <**http://www.digitalpreservation.gov/formats/fdd/fdd000063.shtml**>:

Avery Lee, writing in the rec.video.desktop newsgroup in 2001, commented that "MJPEG, or at least the MJPEG in AVIs having the MJPG fourcc, is restricted JPEG with a fixed -- and *omitted* -- Huffman table. The JPEG must be YCbCr colorspace, it must be 4:2:2, and it must use basic Huffman encoding, not arithmetic or progressive. . . . You can indeed extract the MJPEG frames and decode them with a regular JPEG decoder, but you have to prepend the DHT segment to them, or else the decoder won’t have any idea how to decompress the data. The exact table necessary is given in the OpenDML spec."

This bitstream filter patches the header of frames extracted from an MJPEG stream (carrying the AVI1 header ID and lacking a DHT segment) to produce fully qualified JPEG images.

```
ffmpeg -i mjpeg-movie.avi -c:v copy -bsf:v mjpeg2jpeg frame_%d.jpg
exiftran -i -9 frame*.jpg
ffmpeg -i frame_%d.jpg -c:v copy rotated.avi
```

**mjpegadump**

Add an MJPEG A header to the bitstream, to enable decoding by Quicktime.

**mov2textsub**

Extract a representable text file from MOV subtitles, stripping the metadata header from each subtitle packet.

See also the **text2movsub** filter.

**mp3decomp**

Decompress non-standard compressed MP3 audio headers.

**mpeg2_metadata**

Modify metadata embedded in an MPEG-2 stream.

**display_aspect_ratio**

Set the display aspect ratio in the stream.

The following fixed values are supported:

**4/3**
**16/9**
**221/100**

Any other value will result in square pixels being signalled instead (see H.262 section 6.3.3 and table 6-3).

**frame_rate**

Set the frame rate in the stream. This is constructed from a table of known values combined with a small multiplier and divisor - if the supplied value is not exactly representable, the nearest representable value will be used instead (see H.262 section 6.3.3 and table 6-4).

**video_format**

Set the video format in the stream (see H.262 section 6.3.6 and table 6-6).

**colour_primaries**

**transfer_characteristics**
**matrix_coefficients**
Set the colour description in the stream (see H.262 section 6.3.6 and tables 6-7, 6-8 and 6-9).

**mpeg4_unpack_bframes**
Unpack DivX-style packed B-frames.

DivX-style packed B-frames are not valid MPEG-4 and were only a workaround for the broken Video for Windows subsystem. They use more space, can cause minor AV sync issues, require more CPU power to decode (unless the player has some decoded picture queue to compensate the 2,0,2,0 frame per packet style) and cause trouble if copied into a standard container like mp4 or mpeg-ps/ts, because MPEG-4 decoders may not be able to decode them, since they are not valid MPEG-4.

For example to fix an AVI file containing an MPEG-4 stream with DivX-style packed B-frames using **ffmpeg**, you can use the command:

```
ffmpeg -i INPUT.avi -codec copy -bsf:v mpeg4_unpack_bframes OUTPUT.avi
```

**noise**
Damages the contents of packets or simply drops them without damaging the container. Can be used for fuzzing or testing error resilience/concealment.

Parameters:

**amount**
Accepts an expression whose evaluation per-packet determines how often bytes in that packet will be modified. A value below 0 will result in a variable frequency. Default is 0 which results in no modification. However, if neither amount nor drop is specified, amount will be set to *-1*. See below for accepted variables.

**drop**
Accepts an expression evaluated per-packet whose value determines whether that packet is dropped. Evaluation to a positive value results in the packet being dropped. Evaluation to a negative value results in a variable chance of it being dropped, roughly inverse in proportion to the magnitude of the value. Default is 0 which results in no drops. See below for accepted variables.

**dropamount**
Accepts a non-negative integer, which assigns a variable chance of it being dropped, roughly inverse in proportion to the value. Default is 0 which results in no drops. This option is kept for backwards compatibility and is equivalent to setting drop to a negative value with the same

magnitude i.e. "dropamount=4" is the same as "drop=-4". Ignored if drop is also specified.

Both "amount" and "drop" accept expressions containing the following variables:

**n** The index of the packet, starting from zero.

**tb** The timebase for packet timestamps.

**pts** Packet presentation timestamp.

**dts** Packet decoding timestamp.

**nopts**
Constant representing AV_NOPTS_VALUE.

**startpts**
First non-AV_NOPTS_VALUE PTS seen in the stream.

**startdts**
First non-AV_NOPTS_VALUE DTS seen in the stream.

**duration**
**d** Packet duration, in timebase units.

**pos** Packet position in input; may be -1 when unknown or not set.

**size** Packet size, in bytes.

**key** Whether packet is marked as a keyframe.

**state**
A pseudo random integer, primarily derived from the content of packet payload.

*Examples*

Apply modification to every byte but don’t drop any packets.

```
ffmpeg -i INPUT -c copy -bsf noise=1 output.mkv
```

Drop every video packet not marked as a keyframe after timestamp 30s but do not modify any of the

remaining packets.

```
ffmpeg -i INPUT -c copy -bsf:v noise=drop='gt(t\,30)*not(key)' output.mkv
```

Drop one second of audio every 10 seconds and add some random noise to the rest.

```
ffmpeg -i INPUT -c copy -bsf:a noise=amount=-1:drop='between(mod(t\,10)\,9\,10)' output.mkv
```

**null**

This bitstream filter passes the packets through unchanged.

**pcm_rechunk**

Repacketize PCM audio to a fixed number of samples per packet or a fixed packet rate per second. This is similar to the **asetnsamples audio filter** but works on audio packets instead of audio frames.

**nb_out_samples, n**

Set the number of samples per each output audio packet. The number is intended as the number of samples *per each channel*. Default value is 1024.

**pad, p**

If set to 1, the filter will pad the last audio packet with silence, so that it will contain the same number of samples (or roughly the same number of samples, see **frame_rate**) as the previous ones. Default value is 1.

**frame_rate, r**

This option makes the filter output a fixed number of packets per second instead of a fixed number of samples per packet. If the audio sample rate is not divisible by the frame rate then the number of samples will not be constant but will vary slightly so that each packet will start as close to the frame boundary as possible. Using this option has precedence over **nb_out_samples**.

You can generate the well known 1602-1601-1602-1601-1602 pattern of 48kHz audio for NTSC frame rate using the **frame_rate** option.

```
ffmpeg -f lavfi -i sine=r=48000:d=1 -c pcm_s16le -bsf pcm_rechunk=r=30000/1001 -f framecrc -
```

**pgs_frame_merge**

Merge a sequence of PGS Subtitle segments ending with an "end of display set" segment into a single packet.

This is required by some containers that support PGS subtitles (muxer "matroska").

**prores_metadata**

Modify color property metadata embedded in prores stream.

**color_primaries**

Set the color primaries. Available values are:

**auto** Keep the same color primaries property (default).

**unknown**
**bt709**
**bt470bg**
BT601 625

**smpte170m**
BT601 525

**bt2020**
**smpte431**
DCI P3

**smpte432**
P3 D65

**transfer_characteristics**

Set the color transfer. Available values are:

**auto** Keep the same transfer characteristics property (default).

**unknown**
**bt709**
BT 601, BT 709, BT 2020

**smpte2084**
SMPTE ST 2084

**arib-std-b67**
ARIB STD-B67

**matrix_coefficients**

Set the matrix coefficient. Available values are:

**auto** Keep the same colorspace property (default).

**unknown**
**bt709**
**smpte170m**
BT 601

**bt2020nc**

Set Rec709 colorspace for each frame of the file

```
ffmpeg -i INPUT -c copy -bsf:v prores_metadata=color_primaries=bt709:color_trc=bt709:colorspace=bt709 outpu
```

Set Hybrid Log-Gamma parameters for each frame of the file

```
ffmpeg -i INPUT -c copy -bsf:v prores_metadata=color_primaries=bt2020:color_trc=arib-std-b67:colorspace=bt2
```

**remove_extra**

Remove extradata from packets.

It accepts the following parameter:

**freq** Set which frame types to remove extradata from.

**k** Remove extradata from non-keyframes only.

**keyframe**
Remove extradata from keyframes only.

**e, all**
Remove extradata from all frames.

**setts**

Set PTS and DTS in packets.

It accepts the following parameters:

**ts**
**pts**
**dts** Set expressions for PTS, DTS or both.

**duration**
Set expression for duration.

**time_base**
Set output time base.

The expressions are evaluated through the eval API and can contain the following constants:

**N** The count of the input packet. Starting from 0.

**TS** The demux timestamp in input in case of "ts" or "dts" option or presentation timestamp in case of "pts" option.

**POS**
The original position in the file of the packet, or undefined if undefined for the current packet

**DTS**
The demux timestamp in input.

**PTS** The presentation timestamp in input.

**DURATION**
The duration in input.

**STARTDTS**
The DTS of the first packet.

**STARTPTS**
The PTS of the first packet.

**PREV_INDTS**
The previous input DTS.

**PREV_INPTS**
The previous input PTS.

**PREV_INDURATION**
The previous input duration.

**PREV_OUTDTS**

The previous output DTS.

**PREV_OUTPTS**
The previous output PTS.

**PREV_OUTDURATION**
The previous output duration.

**NEXT_DTS**
The next input DTS.

**NEXT_PTS**
The next input PTS.

**NEXT_DURATION**
The next input duration.

**TB** The timebase of stream packet belongs.

**TB_OUT**
The output timebase.

**SR** The sample rate of stream packet belongs.

**NOPTS**
The AV_NOPTS_VALUE constant.

**text2movsub**
Convert text subtitles to MOV subtitles (as used by the "mov_text" codec) with metadata headers.

See also the **mov2textsub** filter.

**trace_headers**
Log trace output containing all syntax elements in the coded stream headers (everything above the level of individual coded blocks). This can be useful for debugging low-level stream issues.

Supports AV1, H.264, H.265, (M)JPEG, MPEG-2 and VP9, but depending on the build only a subset of these may be available.

**truehd_core**

Extract the core from a TrueHD stream, dropping ATMOS data.

**vp9_metadata**

Modify metadata embedded in a VP9 stream.

**color_space**

Set the color space value in the frame header. Note that any frame set to RGB will be implicitly set to PC range and that RGB is incompatible with profiles 0 and 2.

**unknown**
**bt601**
**bt709**
**smpte170**
**smpte240**
**bt2020**
**rgb**

**color_range**

Set the color range value in the frame header. Note that any value imposed by the color space will take precedence over this value.

**tv**
**pc**

**vp9_superframe**

Merge VP9 invisible (alt-ref) frames back into VP9 superframes. This fixes merging of split/segmented VP9 streams where the alt-ref frame was split from its visible counterpart.

**vp9_superframe_split**

Split VP9 superframes into single frames.

**vp9_raw_reorder**

Given a VP9 stream with correct timestamps but possibly out of order, insert additional show-existing-frame packets to correct the ordering.

## SEE ALSO

**ffmpeg**(1), **ffplay**(1), **ffprobe**(1), **libavcodec**(3)

## AUTHORS

The FFmpeg developers.

For details about the authorship, see the Git history of the project (https://git.ffmpeg.org/ffmpeg), e.g. by typing the command **git log** in the FFmpeg source directory, or browsing the online repository at <**https://git.ffmpeg.org/ffmpeg**>.

Maintainers for the specific components are listed in the file *MAINTAINERS* in the source code tree.