Getting Started with Intel® Stress Bitstreams and Encoder (Intel® SBE) 2017 – AVS2

Version 2.3
Updated August 2, 2016

Contents

Stress Bitstreams

Package Description

This stream set is intended to validate a decoder for AVS2* format compliance. It covers both AVS2 Main and Main10 profiles. Visually clean streams with limited coverage are also available for both profiles. You can choose from the following packages:

Package	Contents
Main profile (82 streams)	• 40 files at 432×240 resolution, 50-500 frames • 37 files at 1920×1080, 10-100 frames • 4 files of 50 frames at 3840×2160 • 1 file at 4320x2400, 5 frames
Main 10 profile (82 streams)	• 40 files at 432×240 resolution, 50-500 frames • 37 files at 1920×1080, 10-100 frames • 4 files of 50 frames at 3840×2160 • 1 file at 4320x2400, 5 frames
Main visual (42 streams)	• 37 files at 1920×1080, 10-100 frames • 4 files of 50 frames at 3840×2160 • 1 file at 4320x2400, 5 frames
Main10 visual (42 streams)	• 37 files at 1920×1080, 10-100 frames • 4 files of 50 frames at 3840×2160 • 1 file at 4320x2400, 5 frames

Each package contains *all_bitstreams.md5* file with checksums for all encoded files. Each bitstream is complemented with MD5 sum for decoding result by RD14.0 reference model implementation.

The decoding result is assumed correct if it binary-matches the result of the reference decoder. This can be verified with the MD5 files included within the package. All the streams are AVS2-compliant to RD14 reference. The stream set does not contain any invalid streams for error-resilience testing. The stream set is not intended for decoder performance testing either, since a lot of bitstream features have distributions not typical for "real-world" video: many long motion vectors pointing out of frame, highly variable QP values, etc.

There are three basic buckets—*INTRA*, *INTER* and *EXTRA*. *INTRA* and *INTER* buckets validate the features related to intra and inter prediction, respectively, while *EXTRA* bucket contains streams covering other AVS2 format features not directly related to intra or inter prediction.

There are two types of streams: *Syntax* and *Stress*. *Syntax* streams are designed to test a certain subset of features, for example, all the intra-prediction modes or all loop-filter related parameters. *Stress* streams include all the features covered by the bucket, so they are useful for smoke testing: if a decoder passes the *Stress* stream, it is likely to pass all the *Syntax* streams from the bucket.

Filename Pattern

All files are named according to the following pattern:
(Purpose)_AVS2_(profile)_(resolution)_(stream_index)_(bucket)_(short_name)_(encoder_version).avs2

- (Purpose) is "Syntax" or "Stress".
- (profile) contains information about feature set, profile, and may indicate bit depth of subsampling option; for example, "Main", "Main10".
- (Resolution) is a pair of frame width and height joint with "x".
- (stream_index) is a 3-digit number. The first digit denotes a bucket (0 – intra, 1 – inter, 2 – extra, 3 – stress) and the other two digits indicate the number of the stream in the bucket. Such

indexing is useful for file sorting and test ordering: in most cases a stream with a smaller number is also less complex in terms of coding tools.

- (bucket): “intra”, “inter”, “extra”.
- (short_name) describes which features this stream tests.
- (_version): version of encoder used for stream generation. Package may contain streams of different encoder versions.

For example, Syntax_AVS2_Main10_1920x1080_ 104_inter_bframes_2.2.1.avs is the fourth stream from *INTER* bucket and it tests support of B-frames. Since the complexity of streams increases incrementally, the preceding streams in the bucket do not contain B-frames. Starting from this stream, some of the following streams may contain B-frames.

(stream_index) can be considered as a short ID of the stream. Pair of (bucket) and (short_name) is a short description of the content of the stream.

Video Content

Each bitstream is encoded from a synthetic video sequence with a subtitle describing *stream_index*, *bucket* and *short_name*. Encoded sequences may have strong visual artifacts.

Visually Clean Streams

Since version 2.2.3 Intel® SBE for AVS2 includes a pack of visually clean streams. In order to get rid of the visual artifacts randomization of some syntax elements was constrained. Coding unit skip mode was disabled to hide randomized prediction. QP and delta QP ranges were constrained to maintain visual consistency of encoded image. Secondary transform and weighted quantization were disabled as these technologies affect visual quality even on low QP values. Visually clean streams were generated from cinematic video sequence.

Stream Description

Test Case (Stream Name)	Description	Resolution, Number of Frames
Intra Bucket (10 streams of 240p, 10 streams of 1080p, 1 stream of 2160p)		
001_intra_basic	I-frames only, luma intra mode fixed to 0, fixed cu_size	432×240, 500 frames 1920×1080, 100 frames
002_intra_partitions	Random partitioning from 64x64 to 8x8 with nonsquare itraprediction and transform enabled	
003_intra_modes	All 33 intra modes, all intra coding unit (cu) types	
004_intra_loopfilter	Loopfilter enabled	
005_intra_qp	Qp randomized on frame level, written on frame or slice level	
006_intra_delta_qp	Qp randomized on block level, written as delta	
007_intra_sao	Sample adaptive offset enabled	
008_intra_alf	Adaptive loopfilter enabled	
009_intra_slices	Multiple slices per picture	
050_intra_stress	Full intra-parameters randomization	432×240, 500 frames 1920×1080, 100 frames 3840×2160, 50 frames

Inter Bucket (14 streams of 240p, 14 streams of 1080p, 1 stream of 2160p)		
101_inter_basic	I and P-frames, fixed cu type, cu size	432×240, 500 frames 1920×1080, 100 frames
102_inter_partitions	Random cu type, partitions from 64x64 to 8x8, asymmetric motion partition enabled	
103_inter_fframes	F-frames enabled	
104_inter_bframes	B-frames enabled	
105_inter_loopfilter	Loopfilter enabled	
106_inter_qp	Qp randomized on frame level, written on frame or slice level	
107_intra_delta_qp	Qp randomized on block level, written as delta	
108_inter_sao	Sample adaptive offset enabled	
109_inter_alf	Adaptive loopfilter enabled	
110_inter_pmvr	PMVR enabled	
111_inter_skip	Sample adaptive offset enabled	
112_inter_tmpid	Adaptive loopfilter enabled	
113_inter_slices	Multiple slices per picture	
150_inter_stress	Full inter-parameters randomization	432×240, 500 frames 1920×1080, 100 frames 3840×2160, 50 frames
Extra Bucket (5 streams of 240p, 5 streams of 1080p, 1 stream of 2160p)		
201_extra_sectrans	Secondary transform enabled	432×240, 500 frames 1920×1080, 100 frames
202_extra_weight_quant	Weighted quantization enabled	
203_extra_chroma_dqp	Chroma delta qp enabled	
204_extra_interlace	Interlaced field-coded stream	
250_extra_stress	Full syntax randomization, except for background picture prediction	432×240, 500 frames 1920×1080, 100 frames 3840×2160, 50 frames
Stress Bucket (11 streams of 240p, 8 streams of 1080p, 1 stream of 2160p, 1 stream of 4320x2400)		
301_stress_longmv	Max possible delta_mv	1920×1080, 30 frames
302_stress_slicepos_ext	Resolution big enough to cover slice position extension	4320×2400, 5 frames
303_stress_background	Scene picture enabled	1920×1080, 50 frames
350_stress	Full syntax randomization	432×240, 500 frames 1920×1080, 100 frames 10 of 432×240, 50 frames 5 of 1920×1080, 10 frames 3840×2160, 50 frames

Methodology

It is easy to check if the decoder passes all the tests: MD5 checksum decoding results must be identical to those included in the package. If something goes wrong, you can use the stream sets to quickly identify which AVS2-format feature causes the problem. The complexity of streams rises with *stream_index*: a decoder is unlikely to pass *002_intra_* test if it has not passed *001_intra_*.

If a decoder fails the *106th* test but passes all the previous tests, the problem is most likely related to the features enabled in this stream. Investigation of mismatch root cause will require comparison of reference and test results, debugging the decoders or using special tools for bitstream analysis and mode-decision visualization.

Each *Syntax* test case is provided in two resolutions: 1920×1080 and 432×240. Both streams have the same *stream_index* and *short_name* and cover exactly the same syntax-elements scope. In most cases a stream of smaller resolution will be more convenient for the developer's testing and debugging purposes, but not all combinations of syntax elements are guaranteed to be available at 240p resolution, so 1080p streams are required for final validation. *Stress* test cases in addition to 240p and 1080p have additional streams of 3840×2160 resolution.

Details on Reference Decoder

MD5 files were generated for plain-YUV420 results of RD15.0 reference decoder. The following command line was used:

```
      ldecod <input>.avs <output>.yuv
```

Random Encoder

Introduction

Random Encoder is a command-line application that accepts the following parameters:

```
Usage: avs2_random_encoder -i <in.yuv> -p <config.json> [-o <out.avs>]
                -w <width> -h <height> [-s <seed> -f <# of frames>]
                [-r <reconstruct.yuv> -v <visual>]
Options:
  -i STRING, --input=STRING
                      input file (raw yuv)
  -o STRING, --output=STRING
                      output avs bitstream
  -w INT, --width=INT   width of input yuv
  -h INT, --height=INT  height of input yuv
  -f INT, --frames=INT  number of frames to process
  -r STRING, --recon=STRING
                      output file for encoder's reconstruct (optional)
  -p STRING, --parfile=STRING
                      json file with distribution parameters (optional)
  -s INT, --seed=INT    seed for random engine
  -v INT, --visual=INT  if set to 1 encoder generates visually clean streams
```

Input and Output

Random Encoder takes uncompressed YUV stream as input and encodes it with random mode decisions. Output is AVS2 bistream. Modes to randomize and range of randomization are defined by parfile (`-p` option). Parfile is a JSON file with comments.

Parfile Editing: Weight and Range Parameters

There are two kinds of parameters in a parfile: weights and range. In most cases logical or enumerated parameters are defined by relative weights. For example, setting "sao_flag" : [1, 5] means one chance of disabled SAO against 5 chances of SAO enabled. Numeric parameters with wide range are defined by specifying minimum and maximum values. For example, luma intra mode acceptable values are from 0 to 32, and in parfile it is defined as "luma_intra_mode " : [0, 32]. Actual values will be spread uniformly in this range.

To disable randomization for weights-defined parameters, weights for all except one values must be set to zero. To fix range-defined parameter you should set min and max values to the same value.

Encoder-output Validation

Random Encoder can output results of internal reconstruct to YUV file (`-r` parameter). It is useful to validate the random encoder by checking that it matches the reference decoder results. For example:

```
REM Execute random encoder.
avs2_random_encoder -i input.yuv -p config.json -o out.avs -r recon.yuv -w 352 -h 288

REM Decode generated bitstream with reference decoder.
ldecod out.avs dec.yuv

REM Verify that encoder's internal reconstruct is bit-exact to result of
REM reference decoder by ensuring that their MD5 sums are identical.
md5sum -b recon.yuv dec.yuv
```

Exhaustive Decoder Validation

Seed parameter `-s` defines initial random-engine state. Changing this seed allows to produce different streams with the same parfile. This feature is helpful to create small bug-reproducers (setting frame number parameter `-f` to some small value) for the parfiles which are known to generate the streams causing failure of examined decoder.

Legal Information

Optimization Notice