Intel® Stress Random Encoder for HEVC

Version 2.1 *Updated Nov 25, 2015*

1.1 General overview

Testing a decoder is a complex multi-criteria task. Code coverage of 100% lines of a decoder code does not guarantee the decoder is 100% compliant. On another hand creation of millions of streams to test all possible feature combinations is time and storage consuming. Random encoder partially solves these two issues. It is light weight encoder with no mode decision so it is as fast as decoder. Effective motion estimation and mode decision are not parts of video codec standard and not required to be tested in decoder side. So this most time consuming part is mostly omitted in Random Encoder in favor of speed and flexibility.

Codec developer will not want to keep all streams generated by random encoder, it is enough to keep only basic vectors. If a decoder fails to correctly decode a randomly generated stream then it makes sense to extend the test pool with the stream for future regression validation. Random Encoder is a great extension of codec validation in addition to Stress Bitstreams' basic vectors.

1.2 Compliance mode

Random encoder is highly configurable and flexible syntax (HEVC) encoder tool. In opposite to regular encoders it is not intended to achieve compression but only designed to create a valid specification-compliant stream. Compliant streams contain only allowed combinations of syntax elements and their values to test decoder for unusual cases or boundary stress cases where developers usually relax requirements to code development for higher decoding speed. Decoder must be compatible with any stream so its sloppy optimizations have to be carefully tracked for boundary cases where residuals overflow may break visual representation of the picture.

You can find the recommended Decode validation process with Random Encoder below. It is up to user to change the flow and to react on pass and fail events and even decide on the criteria of test passing. We recommend comparing the latest development branch of reference HM decoder for HEVC.

Random Encoder accepts as an input *an optional YUV file* and a *parfile* describing testing settings: features to utilize, fixed values, random values. As an output Random Encoder produces *encoded bitstream* and optionally writes *YUV file* with internal reconstruction data. This file is used to validate that Encoder generated proper compressed file and that resulted bitstream is valid.

If there is a mismatch between encoder reconstruct and reference-decoder result, you are welcome to report to Software Publisher (your Intel contact) with the case configuration to request for the fix if it fits to your license agreement with Intel. We will always appreciate your feedback.

Random Encoder has *seed* parameter (-s) defining initial random-engine state. Changing it allows to use the same parfile to produce totally different streams with the same scope of randomization defined by parfile. The main purpose of this seed feature is extensive testing with all possible syntax-element combinations. In addition, this feature can be used 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.

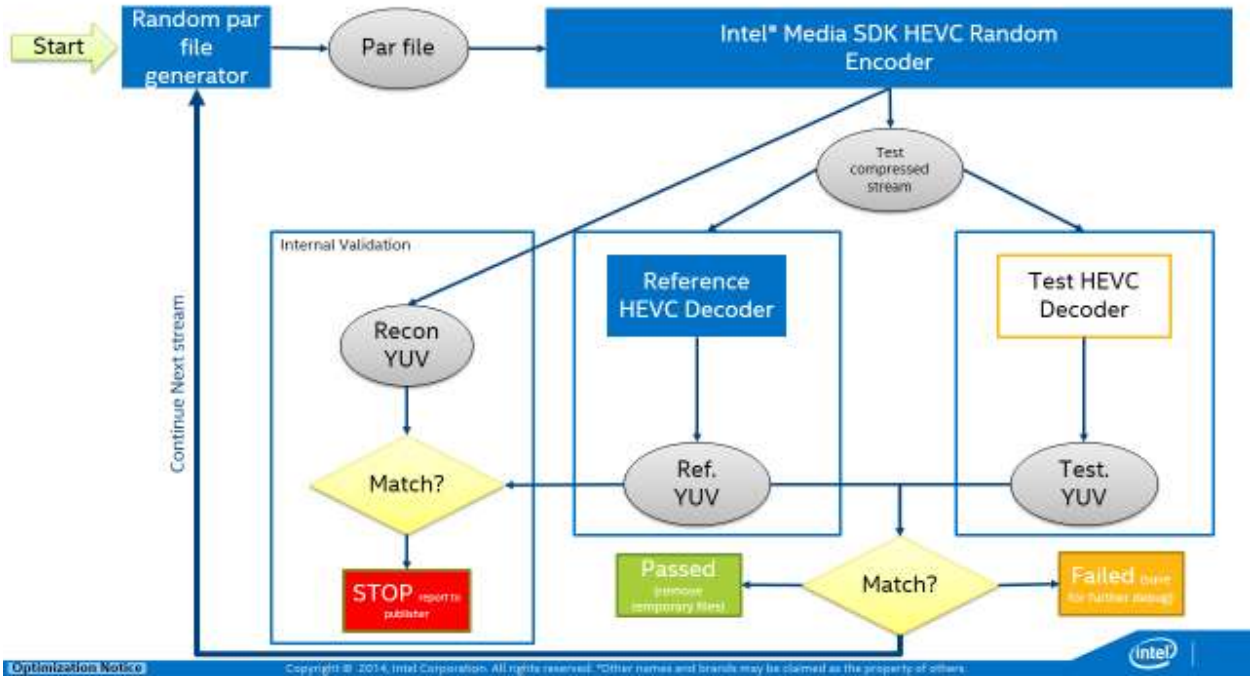
To summarize, typical workflow of compliance testing for HEVC consists of the following steps:

1. Produce test stream by feeding Random Encoder with a parfile and an optional input YUV file,
2. Decode the stream with reference decoder,
3. Verify that Encoder's reconstruct matches with reference-decoder result,
4. Decode the stream with your decoder and verify that its result matches with reference-decoder result,
5. Increment *seed* parameter and go to the step 1.

1.3 HEVC specific

To start validation cycle with HEVC Random Encoder make a decision regarding testing agenda and setup certain features and value range in random-encode parfile. Parfile is a plain text "name = value" file, it contains flags and settings for all syntax elements. Parfile can be used to limit features to a set which is currently implemented by the decoder under testing and focus testing on these features.

Random Encoder accepts an optional input YUV file and parfile and produces one bitstream per call. Therefore for extensive testing the best option is to modify input configuration file outside of random encoder. Illustration below shows an approach of using a parfile generating script (not included) for random encoder.



Level restrictions

HEVC streams have to abide restrictions imposed by tier and level specified in VPS and SPS. By default random encoder checks whether input parameters satisfy tier and level requirements specified in parfile. When no tier and level are specified, or keyword “auto” is used, random encoder tries to find the minimum possible tier and level combination necessary for stream encoded with input parameters. With `--verbose` option it is possible to see in details how tier and level checks are done. To skip tier and level checks it is possible to use command line switch `--ignore_level`.

Random encoder has no bitrate control, so if it is necessary to satisfy tier and level bitrate and compression rate requirements, user is expected to tune corresponding parameters. Settings that affect bitrate most significantly are `Stream.random_tu_probability`, CTB.ipcm_probability, `SPS.ipcm_bit_depth_luma_range`, `SPS.ipcm_bit_depth_chroma_range`, `CTB.transquant_bypass_probability`, `Slice.qp_range`, and `CTB.qp_range`. Generally random encoder produces very large streams because it is not a real encoder, and it is not trying to achieve any compression over source picture, or when pure random data is encoded and `Stream.random_tu_probability` is set to 100. But in most cases to test a decoder it is not necessary to produce streams with bitrate limited by tier and level requirements. In such cases it is possible to use `--ignore_level_bitrate` option and skip bitrate checks after bitstream has been generated.

Some decisions on parameter values are left on HEVC Random Encoder’s discretion for the sake of validity.

Description of command-line application options

Random Encoder is a command-line application, which accepts the following parameters. Most parameters specified on command line may also be specified in parfile. These parameters are listed in the help message after command line switch.

```
--help                                  Show command line help and exit.
--version                               Show version and exit.
--verbose                               Show additional output.
Stream.verbose
-i INPUT                                Set input YUV source file name.
Stream.source_file
-o OUTPUT                               Set output HEVC stream file name.
Stream.stream_file
-r RECONSTRUCT                          Set reconstruct YUV file name.
Stream.reconstrut_file
--statistics                            Output statistics to a specified file.
Stream.statistics_file
-f FRAMES                               Set number of encoded frames.
Stream.num_frames
-s SEED                                 Set random generator initial seed.
Stream.seed
--profile                               Set profile. Valid values are 1 for "main"
VPS.profile_idc_range                   and 2 for "main10" profiles.
-l TIER_LEVEL                           Specify stream tier and level and check
VPS.tier_level                          whether their requirements are satisfied.
                                        A reserved word "auto" (case insensitive)
                                        may be used to make encoder calculate tier
                                        and level itself based on stream parameters.
                                        Valid tier string values are "main" and
                                        "high" (case insensitive). Valid level
                                        string values are "1", "2", "2.1", "3",
                                        "3.1", "4", "4.1", "5", "5.1", "5.2",
                                        "6", "6.1", "6.2".
                                        TierLevel examples: "Main_4", "high_6.1".
--ignore_level                          Don't exit with error if level requirements
Stream.ignore_level_flag                are not satisfied by the stream parameters.
--ignore_level_bitrate                  Don't exit with error if level requirements
Stream.ignore_level_bitrate_flag        are not satisfied by produced stream bitrate
                                        and compression ratio.
-w WIDTH                                Set frame width.
SPS.frame_width
-h HEIGHT                               Set frame height.
SPS.frame_height
--auto_fix_max_CU_size                  When automatically selecting level with
Stream.auto_fix_max_CU_size_flag        "-l auto" allow random encoder to change
                                        SPS.log2_max_cu_size_range to 5 if it is
                                        necessary for level 5 or higher.
-bdl BIT_DEPTH                          Set luma bit depth to a value or range, e.g.
SPS.bit_depth_luma_range                "8" or "8-10".
-bdc BIT_DEPTH                          Set chroma bit depth to a value or range,
SPS.bit_depth_chroma_range              e.g. "8" or "8-10".
--convert_to_10_bit                     Convert 8-bit input YUV to 10-bit to encode
Stream.convert_input_to_10_bit_flag     into main10 profile.
--convert_to_12_bit                     Convert 8-bit input YUV to 12-bit to encode
Stream.convert_input_to_10_bit_flag     12-bit profiles.
-b PROBABILITY                          Set random transform blocks probability.
CTB.random_tu_probability

-cfmt CHROMA_FORMAT                     Set chroma format, the same as SPS.chroma_format_idc
```

Extensive testing and parallel execution

Random Encoder as fast as ≈50 000 cases a week for a single process. For satisfactory validation, it will need a month execution with NO fails in decoder on a corner case. It is possible to execute several processes in parallel on a single computer to increase coverage for the shorter time period as many as hardware memory and CPU cores allows.

Main10 notes

Main10 profile allows different bit depths for luma and chroma. To generate streams with varying bit depths it is necessary to set SPS bit_depth_luma_range and bit_depth_chroma_range parameters to a range of values instead of a fixed value or use –bdl and –bdc command line switches. Bit depth changes can happen only on IDR frame so it is necessary to set Stream idr_interval to a positive value instead of default zero value. To enable changes for bit depth values in SPS header it is necessary to enable SPS sps_header_change_interval.

Reconstructed YUV file is output in maximum possible bit depth specified for stream rounded to bytes. E.g. if bit_depth_luma_range is set to 8-9 and bit_depth_chroma_range is set to 8 then both luma and chroma are output in 2 byte unsigned values even though bit_depth_chroma_range gets 2 byte bit depths.

Format Range Extensions notes

To generate streams compliant with HEVC Main 4:2:2 10 Format range extension profile, it is enough to set profile_idc_range input parameter to [4, 4] or pass an option "–profile 4" in command line with simultaneous set of chroma_format_idc_range parameter to [2, 2] or pass an option "-cfmt 2" in command line.

Specific for Main 4:2:2 10 profile technologies may be enabled by setting needed limits for syntax values with corresponding input parameters listed below in SPS and/or PPS sections.

Example of a command line with parameters for enabling Main 4:2:2 10 profile is below

-p 501_main422_10bit.json -i someYUV_422p.yuv -w 1920 -h 1080 -o 1.hevc -f 100 -bdl 10 -bdc 10 -cfmt 2 --profile 4 --convert_to_10_bit

where someYUV_422p.yuv is a raw video sequence sampled in 4:2:2 format, possibly 8-bit.

Main 12, Main 4:4:4 12, Main 4:2:2 12, Main 4:4:4 10 HEVC profiles matching the same HEVC profile index equal to 4 may be enabled by –cfmt and –bdl, -bdc options as well.

Parfiles of series 60x are provided to validation of HEVC REext Main 4:4:4 specific technologies.

Parfile fields and values

Parfile is a JSON file which has the following sections: "Stream", "VPS", "SPS", "PPS", "Slice", "CTB" and "SEI". Each section has an array of values, each of which may have the following types: string, number, flag, range or probability. String types are used for file names and tier.

Numbers specify values which don't change for the during all stream generation process. Flags are also unchangeable values which can be only "true" or "false". Range values are used for values which vary randomly in the specified range including its beginning and the end. Some values and their ranges are corrected automatically, e.g. log2_max_cu_size_range is generated after log2_min_cu_size_range, and its minimum can't be less than generated value for log2_min_cu_size_range. Probability parameters specify probability in percent (0-100%) for Boolean values which. 0% mean that value is always false, 100% mean that value is always true.

REext specific technologies are controlled by parameters at the bottom of the SPS and PPS sections.

Parameter name	Type and max/min values if they exist	Default value	Description
Stream section			
seed	number	1234556789	Random generator initial seed value.
num_frames	number	0	Number of frames in output stream.
start_frame_num	number	0	Ignored.
frame_rate	number	30	Stream frame rate.
source_file	string	“”	Name of input YUV file.
stream_file	string	“”	Name of output file.
reconstrut_file	string	“”	Name of output reconstruct file.
statistics_file	string	“”	Name of output statistics file.
verbose	flag	false	Whether to show additional output in console.
convert_input_to_10_bit_flag	flag	false	Convert input 8 but YUV to bits by shifting it left by two bits.
ignore_level_flag	flag	false	Ignore level unsatisfied requirements.
ignore_level_bitrate_flag	flag	false	Ignore only stream bitrate level requirement.
auto_fix_max_CU_size_flag	flag	false	Automatically correct maximum CTB size in “auto” level mode.
disable_sei_flag	flag	false	Disable all SEI.
vps_header_change_interval	number	1	Let generator create a new VPS every Nth IDR.
sps_header_change_interval	number	1	Let generator create a new SPS every Nth IDR.
pps_header_change_interval	number	1	Let generator create a new PPS every Nth IDR.
idr_interval	number	0	Insert IDR every idr_interval CVSs. When zero, no intra frame becomes IDR. IDRs are necessary for bit depth

			variations because they happen only on IDR frames.
num_p_range	range	0 – 0	Number of P frames between IDR frames.
num_b_range	range	0 – 0	Number of B frames between P frames.
max_ref_idx_l0_range	range 1 – 15	1 – 15	Maximum number of L0 reference frames for P slices. Limited by value generated for num_p_range.
max_ref_idx_l1_range	number 1 – 15	1 – 15	Maximum number of L1 reference frames. Limited by value generated for num_b_range.
max_b_ref_idx_l0_range	number 1 – 15	1 – 15	Maximum number of L0 reference frames for B slices. Limited by value generated for num_p_range.
low_delay_probability	probability	50%	Probability of choosing low delay or random access mode for B frames.
slice_number_range	range 1 – 64	1	Maximum number of slices that a frame may have.
VPS section			
number_of_temporal_layers_range	range 1 – 1	1	Number of temporal layers.
profile_idc_range	range 1 – 2	1 – 1	Sets bitstream profile. 1 for main or 2 for main10.
tier_level	string	“auto”	String describing output stream tier and level information in the same format as for “-l” command line option.
SPS section			
frame_width	number	1920	Stream width resolution.
frame_height	number	1080	Stream height resolution.
chroma_format_idc_range	range 1 – 1	1	Chroma format idc.

bit_depth_luma_range	range 8 – 10	8 – 8	Bit depth of luma samples.
bit_depth_chroma_range	range 8 – 10	8 – 8	Bit depth of chroma samples.
log2_min_cu_size_range	range 3 – 6	3 – 6	CTB maximum size. Limited by value generated for log2_min_cu_size_range.
log2_max_cu_size_range	range 4 – 6	4 – 6	CTB maximum size.
log2_min_tu_size_range	range 2 – 5	2 – 5	Minimum size of TU. Limited by value generated for log2_min_cu_size_range – 1.
log2_max_tu_size_range	range 2 – 5	2 – 5	Maximum size of TU. Limited by value generated for log2_min_tu_size_range.
log2_max_tu_depth_intra_range	range 0 – 4	0 – 4	Maximum depth of splitting inter CTBs. Limited by values generated for log2_max_cu_size_range - log2_min_tu_size_range.
log2_max_tu_depth_inter_range	range 0 – 4	0 – 4	Maximum depth of splitting intra CTBs. Limited by values generated for log2_max_cu_size_range - log2_min_tu_size_range.
enable_ipcm_probability	probability	50	Whether to enable IPCM in SPS.
log2_min_ipcm_size_range	range 3 – 5	3 – 5	Minimum size of IPCM CU. Limited by value generated for log2_max_cu_size_range and log2_max_cu_size_range.
log2_max_ipcm_size_range	range 3 – 5	3 – 5	Maximum size of IPCM CU. Limited by value generated for log2_max_cu_size_range and log2_max_cu_size_range.

ipcm_bit_depth_luma_range	number 1-10	1 – 10	Bit depth of ICPM luma samples. Limited by value generated for bit_depth_luma_range.
ipcm_bit_depth_chroma_range	number 1-10	1 – 10	Bit depth of ICPM chroma samples. Limited by value generated for bit_depth_chroma_range.
ipcm_loop_filter_disable_probability	probability	50	Specifies pcm_loop_filter_disabled_flag value.
enable_temporal_mvp_probability	probability	75	Enable temporal motion vector prediction.
enable_sao_probability	probability	75	Enable SAO filter.
scaling_list_probability	probability	75	Enable scaling list.
scaling_list_data_probability	probability	50	Scaling list data in SPS.
amp_enabled_probability	probability	75	Enable asymmetric motion partitions.
strong_intra_smoothing_probability	probability	50	Specifies strong_intra_smoothing_enabled_flag value.
enable_vui_probability	probability	65	Enable VUI with random contents.
range_extension_probability	probability	0	Enable RExt specific syntax.
transform_skip_rotation_enabled_probability	probability	0	Enable rotation in transform skip
transform_skip_context_enabled_probability	probability	0	Enable modification of transform skip context
implicit_rdpcm_enabled_probability	probability	0	Intra RDPCM
explicit_rdpcm_enabled_probability	probability	0	Inter RDPCM
intra_smoothing_disabled_probability	probability	0	Disable intra smoothing
high_precision_offsets_enabled_probability	probability	0	Enable high precision offsets in weighted predicition

persistent_rice_adaptation_enabled_probability	probability	0	Enable persistent Rice adaptation
cabac_bypass_alignment_enabled_probability	probability	0	Enable CABAC bypass alignment
PPS section			
dependent_slice_probability	probability	50	Enable dependent slices.
sign_data_hiding_probability	probability	50	Specifies sign_data_hiding_enabled_flag value.
cabac_init_present_probability	probability	50	Specifies cabac_init_present_flag value.
Init_qp_minus26	range	–50, 25	PPS QP control.
constrained_intra_probability	probability	50	Specifies constrained_intra_pred_flag value.
enable_transform_skip_probability	probability	50	Enable transform skip in PPS.
enable_cu_qp_delta_probability	probability	50	Specifies cu_qp_delta_enabled_flag value.
diff_cu_qp_delta_depth_range	range 0 – 3	0 – 3	Specifies diff_cu_qp_delta_depth value. Limited by value generated for log2_max_cu_size_range - log2_min_cu_size_range.
enable_transquant_bypass_probability	probability	50	Enable transquant bypass in PPS.
weighted_pred_probability	probability	50	Specifies weighted_pred_flag value.
weighted_bipred_probability	probability	50	Specifies weighted_bipred_flag value.
log2_luma_weight_denom_range	range 0 – 7	0 – 7	Specifies luma_log2_weight_denom value.
log2_chroma_weight_denom_range	range 0 – 7	0 – 7	Specifies chroma_log2_weight_denom value.
wavefront_or_tiles_probabilities	probabilities array	34, 34	First probability is for wavefront, second for tiles. The rest of 100% is for neither wavefront nor tiles.

uniform_spacing_probability	probability	50	Specifies uniform_spacing_flag value.
loop_filter_across_tiles_probability	probability	50	Enable loop filter across tiles.
LoopFilterAcrossSlicesEnabledFlag	probability	50	Enable loop filter across some slices.
deblocking_control_present_probability	probability	75	Specifies deblocking_filter_control_present_flag value.
deblocking_filter_override_probability	probability	50	Specifies deblocking_filter_override_enabled_flag value.
enable_deblocking_probability	probability	75	Enables deblocking in PPS.
scaling_list_data_probability	probability	50	Enables scaling list in PPS.
log2_parallel_merge_level_range	range 2 – 6	2 – 6	Value of Log2ParMrgLevel in HEVC specification. Limited by value generated for log2_max_cu_size_range.
lists_modification_present_probability	probability	50	Specifies lists_modification_present_flag value.
range_extension_probability	probability	50	Enables RExt specific syntax.
log2_max_transform_skip_block_size_minus2_range	range	0 – 3	Specifies limit depth for transform skip
chroma_qp_adjustment_enabled_probability	probability	50	Enables chroma qp offsets lists
cu_chroma_qp_adjustment_depth_range	range	0 – 3	Specifies depth for chroma qp offsets
cross_component_prediction_enabled_probability	probability	0	Enables CCP
Slice section			
enable_temporal_mvp_probability	probability	75	Specifies slice_temporal_mvp_enabled_flag value.

enable_luma_sao_probability	probability	50	Specifies slice_sao_luma_flag value.
enable_chroma_sao_probability	probability	50	Specifies slice_sao_chroma_flag value.
mvd_l1_zero_probability	probability	50	Specifies mvd_l1_zero_flag value.
cabac_init_probability	probability	50	Specifies cabac_init_flag value.
collocated_from_l0_probability	probability	50	Specifies collocated_from_l0_flag value
collocated_ref_idx_range	range 0 – 15	0 – 14	Specifies collocated_ref_idx value.
max_num_merge_cand_range	range 1 – 5	1 – 5	Specifies number of merge candidates used for merge prediction.
slice_qp_delta	range	-75, 75	Slice QP control.
deblocking_filter_override_probability	probability	50	Specifies deblocking_filter_override_flag value.
enable_deblocking_probability	probability	75	Enable deblocking in slice.
loop_filter_across_slices_probability	probability	50	Specifies slice_loop_filter_across_slices_enabled_flag value.
CTB section			
random_tu_probability	probability	100	Probability in percent of randomly generating TU residual coefficients.
ipcm_probability	probability	10	Probability of IPCM intra block.
transquant_bypass_probability	probability	10	Probability of transquant bypass CUs.
transform_skip_probability	probability	10	Probability of transform skip CUs.
skip_probability	probability	20	Probability of skip blocks for inter PUs.
merge_probability	probability	50	Probability in percent of merge prediction for inter PUs.

qp_change_probability	probability	25	Probability of changing QP on the next CU.
qp_range	range 0 – 51	0 – 51	Range of QP values generated for CU.
intra_in_inter_probability	probability	10	Probability of generating intra prediction for P and B slices.
intra_NxN_probability	probability	50	Probability of NxN intra prediction.
SEI section			
activate_parameter_sets_probability	probability	20	Enable active parameter sets SEI.
buffering_period_probability	probability	20	Enable buffering period SEI.
picture_timing_probability	probability	20	Enable picture timing SEI.
decoded_picture_hash_probability	probability	100	Enable decoded picture hash SEI.
filler_probability	probability	80	Enable filler payload SEI.
filler_payload_length_range	range	1 – 50	Number of payload bytes for filler SEI.
user_data_registered_probability	probability	80	Enable user data registered SEI.
user_data_registered_length_range	range	1 – 50	Number of payload bytes for user data registered SEI.
user_data_unregistered_probability	probability	80	Enable user data unregistered SEI.
user_data_unregistered_length_range	range	1 – 50	Number of payload bytes for user data unregistered SEI.

1.4 Error Resilience Encoder

Description

Error Resilience Encoder for HEVC is a tool that allows you to generate broken video streams to test behavior of HEVC decoder on various types of errors. What's more important – it allows to control type and positioning of errors, which makes stream invalid. This tool can be used to pinpoint flaws in error handling and to more precisely define expected behavior on the wide range of possible errors.

Error Resilience Encoder is based on Intel® Random Encoder: at first, valid compliant stream is generated with Random Encoder, utilizing all of its flexibility, then destructive changes are applied to it, based on user-set parameters.

Error Resilience Encoder accepts as an input a *YUV file* and a *parfile* – a JSON-formatted file, describing testing settings: features to utilize, fixed values, random values. As an output, Encoder produces encoded broken bitstream.

Important restrictions

Although Error Resilience Encoder is an extension of Random Encoder, it's not intended to generate valid streams. Currently, only first 15 frames of the stream are fully customizable by user. For the rest of the frames, if user-set parameters don't meet the required level of degradation, additional errors are introduced into stream. Error Resilience Encoder comes as a separate product from Random Encoder.

Broken stream generation

Rules of error generation are described by "broken" section in parfile. his chapter explains its parameters in detail. There are 3 invalidation options: bitwise randomization, packet-level failures and corrupted syntax elements. Corresponding subsections in JSON are: "bit", "packet" and "syntax".

Bit randomization

The idea of bit randomization is to simply invert some bits in valid HEVC stream. Main parameter here is the probability with wich specific bit will be inverted – parameter "prob" in subsection "bit". Each frame of the generated HEVC stream consists of separately coded parts: headers, and compressed frame data. Error Resilience Encoders allows to specify separate probabilities for each of this parts. If some of this parameters are not specified explicitly, default "prob" is used instead. Bit randomization allows to simulate low-level network and storage errors.

slice_header_broken_probability
code_ptl_broken_probability
scaling_list_broken_probability
vps_broken_probability
sps_broken_probability
pps_broken_probability
seq_broken_probability
code_profile_tier_broken_probability
nal_broken_probability
frame_data_probability
xcode_scaling_list_probability
short_term_ref_pic_set_probability
hrd_parameters_probability

Packet Randomization

Main idea is to operate on packet level, i.e. with set of bytes. "prob" in "packet" section stands for probability for each byte wether corrupted packet starts with this byte. Parameter "size" :

[min, max] determines ranged distribution of the length of this packet. This packet could be then cut from the stream, turned to zeroes, or duplicated, depending on parameter "mode", which is weighted distribution. As special case, packet-level randomization allows you to cut sections or entire frames from stream.

slice_header_broken_probability
code_ptl_broken_probability
scaling_list_broken_probability
vps_broken_probability
sps_broken_probability
pps_broken_probability
seq_broken_probability
code_profile_tier_broken_probability
nal_broken_probability
frame_data_probability
xcode_scaling_list_probability
short_term_ref_pic_set_probability
hrd_parameters_probability
size
mode

Syntax randomization

This section allows you to corrupt syntax elements, where bitstream specification made it possible. Mostly it consists of the elements of the uncompressed header. Each parameter is a probability with wich this feature will be enabled. The following table lists all of the features.

slice_type_broken_probability
num_ref_idx_l0_active_minus1_broken_probability
num_ref_idx_l1_active_minus1_broken_probability
short_term_ref_pic_set_idx_broken_probability
num_long_term_pics_broken_probability
num_long_term_sps_broken_probability
num_poc_total_cur_broken_probability
slice_tc_offset_div2_broken_probability
collocated_ref_idx_probability
five_minus_max_num_merge_cand_probability
num_entry_point_offsets_probability
offset_len_probability
entry_point_offset_probability
slice_qp_delta_probability
slice_qp_delta_cb_probability
slice_qp_delta_cr_probability
vps_id_probability
unit_type_probability
temporal_id_probability
code_profile_tier_profile_space_probability

code_profile_tier_profile_idc_probability
code_profile_tier_bit_depth_constraint_probability
code_profile_tier_chroma_format_con_probability
ptl_level_idc_probability
ui_max_tlayers_probability
ui_max_layers_probability
num_hrd_parameters_probability
num_op_sets_probability
sps_vps_id_probability
sps_max_t_layers_probability
sps_sps_id_probability
sps_chroma_format_idc_probability
sps_pic_width_in_luma_samples_probability
sps_pic_height_in_luma_samples_probability
sps_bit_depth_probability
sps_bits_for_poc_probability
sps_log2_min_coding_block_size_probability
sps_log2_diff_max_min_coding_block_size_probability
sps_quadtree_tu_log2_min_size_probability
sps_quadtree_tu_log2_max_size_probability
sps_quadtree_tu_max_depth_inter_probability
sps_quadtree_tu_max_depth_intra_probability
sps_pcm_bit_depth_intra_probability
sps_pcm_log2_min_size_probability
sps_pcm_log2_max_size_probability
vui_aspect_ratio_idc_probability
vui_video_format_probability
vui_colour_primaries_probability
vui_transfer_characteristics_probability
vui_matrix_coefficients_probability
vui_chroma_sample_loc_type_top_field_probability
vui_chroma_sample_loc_type_bottom_field_probability
vui_min_spatial_segmentation_idc_probability
vui_max_bytes_per_pic_denom_probability
vui_max_bits_per_min_cu_denom_probability
vui_log2_max_mv_length_horizontal_probability
vui_log2_max_mv_length_vertical_probability
hrd_tick_divisor_minus2_probability
hrd_du_cpb_removal_delay_length_minus1_probability
hrd_dpb_output_delay_du_length_minus1_probability
hrd_bit_rate_scale_probability
hrd_cpb_size_scale_probability
hrd_du_cpb_size_scale_probability
hrd_initial_cpb_removal_delay_length_minus1_probability
hrd_cpb_removal_delay_length_minus1_probability
hrd_dpb_output_delay_length_minus1_probability

pps_pps_id_probability
pps_sps_id_probability
pps_num_extra_slice_header_bits_probability
pps_num_ref_idx_l0_default_active_probability
pps_num_ref_idx_l1_default_active_probability
pps_pic_init_qp_minus26_probability
pps_max_cu_dqp_depth_probability
pps_qp_offset_probability
pps_deblocking_filter_beta_offset_div2_probability
pps_deblocking_filter_tc_offset_div2_probability
pps_log2_parallel_merge_level_minus2_probability
pps_log2_max_transform_skip_block_size_probability
pps_diff_cu_chroma_qp_offset_depth_probability
pps_chroma_qp_offset_list_len_probability
pps_log2_sao_offset_scale_probability
vps_vps_id_probability
vps_max_dec_pic_buffering_probability
vps_num_reorder_pics_probability
vps_max_latency_increase_probability
scaling_list_list_id_probability
xscaling_list_data_probability
ref_delta_idx_probability
ref_delta_rps_sign_probability
ref_abs_delta_rps_probability
ref_num_negative_pics_probability
ref_num_positive_pics_probability
ref_delta_poc_probability

Legal Information

Optimization Notice

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel.

Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804