

General Description

CAVLC(Content adaptive Variable Length Coding) IP block is capable of decoding 1 to 3 syntax elements per clock cycle with variable input and variable output rates. Any of the internal functional stage can be skipped for optimal performance based on the input stream currently decoding. For example, PHASE III (TotalZeros) and PHASE IV (Runs) decoding can be completely skipped if number of Total Coefficients is equal to Maximum number of coefficient allowed (i.e. 4 or 16).

Applications

- High Quality Video, Low Bit-Rate, Low Power Applications
- Wireless Video
- Video Streaming
- Video Conferencing
- Video Surveillance

Features

- Content Adaptive Variable Length Decoder compliant with H.264
- Decodes all Macro-Block Types
- Supports Raster ordering at the output

Functional Description

The CAVLC block performs four functions:

- Parses the CAVLC data stream and recreates the 4 x 4 sub-blocks.
- Interprets the CBP input to insert zeros into the output datastream for the appropriate luma and chroma sub-blocks.
- Takes the DC components from the IIT block and inserts into the output datastream.
- Indicates to BSIF when the end of the macroblock is reached.

For mb_type = 0, each sub-block is coded separately. Input order is zig-zag scan of 4 x 4 (and 2 x 2) blocks of encoded transform coefficients.

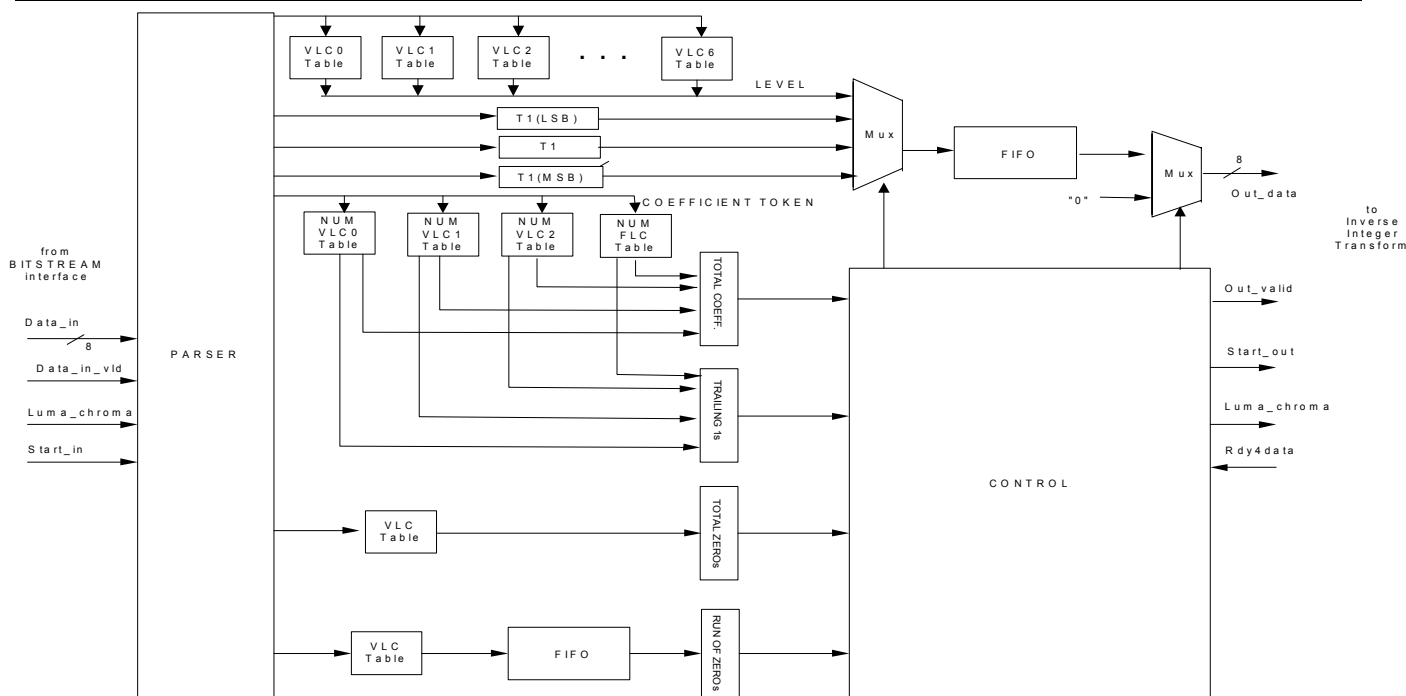


Fig. 1 CAVLC Decoder Block Diagram

Output order is highest in frequency first. The missing higher frequencies are zero. These zero's will be inserted by the CAVLC block into the sub-block before being passing to IIT.

N_U and N_L are the "total_coeff" of the neighboring blocks. $(N_U + N_L + 1) \gg 1$ will select the VLC table. N_U and N_L are kept within the CAVLC block in a set of registers.

The sequence goes from top to bottom as time increases. The CBP[5:0] input is 6 bits wide.

Table 1: CAVLC input/output sequences for mb_type =0

	Input	Output
Luma_dc_ac : sub_block0	Missing if CBP[0] is 0	Insert 0's for this sub_block if input field is missing.
Luma_dc_ac : sub_block1	Missing if CBP[0] is 0	Insert 0's for this sub_block if input field is missing.
Luma_dc_ac : sub_block2	Missing if CBP[0] is 0	Insert 0's for this sub_block if input field is missing.
Luma_dc_ac : sub_block3	Missing if CBP[0] is 0	Insert 0's for this sub_block if input field is missing.
Luma_dc_ac : sub_block4	Missing if CBP[1] is 0	Insert 0's for this sub_block if input field is missing.
Luma_dc_ac : sub_block5	Missing if CBP[1] is 0	Insert 0's for this sub_block if input field is missing.
Luma_dc_ac : sub_block6	Missing if CBP[1] is 0	Insert 0's for this sub_block if input field is missing.
Luma_dc_ac : sub_block7	Missing if CBP[1] is 0	Insert 0's for this sub_block if input field is missing.
Luma_dc_ac : sub_block8	Missing if CBP[2] is 0	Insert 0's for this sub_block if input field is missing.
Luma_dc_ac : sub_block9	Missing if CBP[2] is 0	Insert 0's for this sub_block if input field is missing.
Luma_dc_ac : sub_block10	Missing if CBP[2] is 0	Insert 0's for this sub_block if input field is missing.

Luma_dc_ac : sub_block11	Missing if CBP[2] is 0	Insert 0's for this sub_block if input field is missing.
Luma_dc_ac : sub_block12	Missing if CBP[3] is 0	Insert 0's for this sub_block if input field is missing.
Luma_dc_ac : sub_block13	Missing if CBP[3] is 0	Insert 0's for this sub_block if input field is missing.
Luma_dc_ac : sub_block14	Missing if CBP[3] is 0	Insert 0's for this sub_block if input field is missing.
Luma_dc_ac : sub_block15	Missing if CBP[3] is 0	Insert 0's for this sub_block if input field is missing.
Chroma.dc	Missing if CBP[5:4] is 00	Insert 0's for this sub_block if input field is missing.
Chroma.dc	Missing if CBP[5:4] is 00	Insert 0's for this sub_block if input field is missing.
Chroma.ac : sub_block0	Missing if CBP[5] is 0	Insert 0's for this sub_block if input field is missing.
Chroma.ac : sub_block1	Missing if CBP[5] is 0	Insert 0's for this sub_block if input field is missing.
Chroma.ac : sub_block2	Missing if CBP[5] is 0	Insert 0's for this sub_block if input field is missing.
Chroma.ac : sub_block3	Missing if CBP[5] is 0	Insert 0's for this sub_block if input field is missing.
Chroma.ac : sub_block0	Missing if CBP[5] is 0	Insert 0's for this sub_block if input field is missing.
Chroma.ac : sub_block1	Missing if CBP[5] is 0	Insert 0's for this sub_block if input field is missing.
Chroma.ac : sub_block2	Missing if CBP[5] is 0	Insert 0's for this sub_block if input field is missing.
Chroma.ac : sub_block3	Missing if CBP[5] is 0	Insert 0's for this sub_block if input field is missing.

To each Intra_16x16 prediction macroblock, an Intra16x16PredMode is assigned, which specifies the Intra_16x16 prediction mode. CodedBlockPatternChroma contains the coded block pattern value for chroma as specified in Table 2. CodedBlockPatternLuma specifies whether for the luma component non-zero AC transform coefficient levels are present. CodedBlockPatternLuma equal to 0 specifies that there are no AC transform coefficient levels in the luma component of the macroblock. CodedBlockPatternLuma equal to 15 specifies that at least one AC transform coefficient level is in the luma component of the macroblock, requiring scanning of AC transform coefficient levels for all 16 of the 4x4 blocks in the 16x16 block.

Table 2: CAVLC input/output sequence for mb_type not equal to 0, i.e. Intra_16x16

CodedBlockPatternChroma	Description
0	All chroma transform coefficient levels are equal to 0.
1	One or more chroma DC transform coefficient levels are non-zero. All chroma AC transform coefficient levels are equal to 0.
2	Zero or more chroma DC transform coefficient levels are non-zero valued. One or more chroma AC transform coefficient levels are non-zero valued.

Note: This is the SAME cbp for luma and chroma as shown for Intra4x4, except it must be decoded from the mb_type field. Also, the cbp for LUMA only has options of 0 or 15, i.e. no AC components or all AC components.

Table 3: Input/Output Description

SIGNAL	Dir.	Wid.	DESCRIPTION
bsif_start_slice	Input	1	Signal to indicate start of slice
bsif_start_frame	Input	1	Signal to indicate start of frame
bsif_cavlcdata_valid	Input	1	Input data is valid
bsif_start_cavlc	Input	1	Start cavlc data for macroblock
bsif_intra16	Input	1	Data lines contain cdp info for intra16x16
bsif_data	Input	8	Data Input
bsif_cbp	Input	1	Data lines contain cdp info for intra4x4
cavlc_hold	Output	1	Hold signal
cavlc_eomb	Output	1	End cavlc data for macroblock
cavlc_eomb_bit	Output	3	Indicates last bit of the cavlc data
cavlc_data_s	Output	8	Data Output
cavlc_data_valid	Output	1	Output data is valid
cavlc_start	Output	3	Start info from CAVLC
cavlc_luma_chrom	Output	1	Indicates whether data is Luma or Chroma
iit_data_s	Input	8	Data from IIT
iit_data_valid	Input	1	Data from IIT is valid
iit_start	Input	3	Start Info from IIT
iit_hold	Input	1	Hold signal

For more information, please contact us at:

PixSil Technology Corporation
 4533 MacArthur Blvd., Newport Beach, CA 92660
 Email: info@pixsiltech.com Internet: www.pixsiltech.com