XAPP616 (v1.0) April 22, 2003

Huffman Coding

Author: Latha Pillai

Summary

Huffman coding is used to code values statistically according to their probability of occurence. Short code words are assigned to highly probable values and long code words to less probable values. Huffman coding is used in MPEG-2 to further compress the bitstream. This application note describes how Huffman coding is done in MPEG-2 and its implementation.

Introduction

The output symbols from RLE are assigned binary code words depending on the statistics of the symbol. Frequently occurring symbols are assigned short code words whereas rarely occurring symbols are assigned long code words. The resulting code string can be uniquely decoded to get the original output of the run length encoder. The code assignment procedure developed by Huffman is used to get the optimum code word assignment for a set of input symbols.

The procedure for Huffman coding involves the pairing of symbols. The input symbols are written out in the order of decreasing probability. The symbol with the highest probability is written at the top, the least probability is written down last. The least two probabilities are then paired and added. A new probability list is then formed with one entry as the previously added pair. The least symbols in the new list are then paired. This process is continued till the list consists of only one probability value. The values "0" and "1" are arbitrarily assigned to each element in each of the lists.

Figure 1 shows the following symbols listed with a probability of occurrence where: A is 30%, B is 25%, C is 20%, D is 15%, and E = 10%.

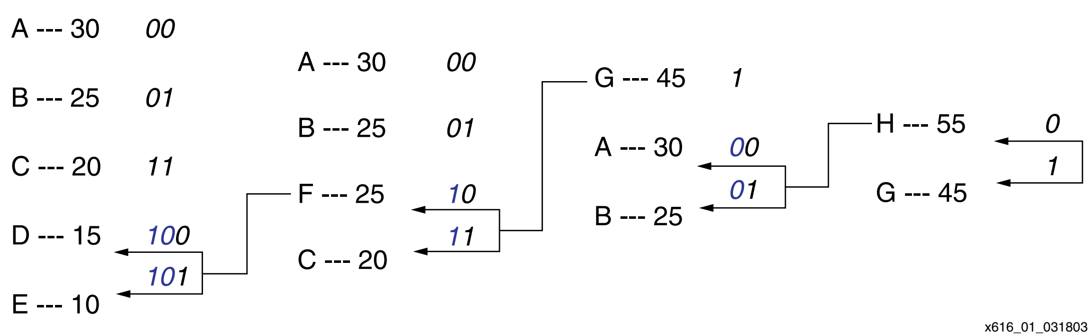


Figure 1: **Huffman Coding Procedure**

Step

1. Adding the two least probable symbols gives 25%. The new symbol is F
2. Adding the two least probable symbols gives 45%. The new symbol is G
3. Adding the two least probable symbols gives 55%. The new symbol is H
4. Write "0" and "1" on each branch of the summation arrows. These binary values are called branch binaries.

5. For each letter in each column, copy the binary numbers from the column on the right, starting from the right most column (i.e., in column three, G gets the value "1" from the G in column four.) For summation branches, append the binary from the right-hand side column to the left of each branch binary. For A and C in column three append "0" from H in column four to the left of the branch binaries. This makes A "00" and B "01".

Completing step 5 gives the binary values for each letter: A is "00", B is "01", C is "11", D is "100", and E is "101". The input with the highest probability is represented by a code word of length two, whereas the lowest probability is represented by a code word of length three.

Huffman Coding in MPEG-2

MPEG is a non-adaptive coding system, the code table does not change with the input video sequence. Many image sequences were coded and the statistics used to define the entries in the MPEG-2 Huffman table. JPEG still image compression, using adaptive entropy coding, where the table entries can be modified to better suit a particular picture. JPEG also uses adaptive arithmetic coding where the code table is changeable depending on the change in the statistics of the JPEG image.

MPEG defines a set of variable-length code (VLC) for each of the probable run/level combinations. The run/level combinations not found in the table are represented by an escape code followed by a six-bit code for the run and an eight or 16-bit code for the level. The end-of-block (EOB) code is used when all the remaining coefficients in the 8 x 8 block are zeroes. Coding of the 8 x 8 block starts from the DC coefficient in the zigzag order. When there are no more nonzero coefficients remaining in the zigzag order, the EOB code is used to terminate coding. Since the probability of occurrence of the EOB symbol is high, it is assigned a two-bit code "10".

The VLC tables used in MPEG-2 are not true Huffman codes. They are optimized for a range of bit rates to sample rate ratios. Most of the code words in the MPEG-2 tables were carried over from the H.261 standard. The DCT coefficient tables in MPEG-2 assume equal probability for both positive and negative coefficients. MPEG-2 VLC uses a new table, Table 8. It is better suited for the statistics of intra-coded blocks. The EOB code for Table 7 has two bits but for Table 8, the EOB has four bits. This implies that an intra-coded block can have on average of 2^4 or 16 non-zero AC coefficients. For non intra-coded blocks, the statistics point to an average of 2^2 or four non-zero AC coefficients. Both Table 7 and Table 8 have 113 entries.

The VLC table consists of Huffman codes for different run/level combinations. The last bit "s" of the code denotes the sign of the level with s = 0 for positive and s = 1 for negative. The VLC table also includes the EOB code to indicate the status of the rest of the coefficients as zero. The EOB cannot occur as the only code in a block since no coding was done in the block. There are run/level combinations not defined in the VLC tables. When the variable length coder sees an undefined run/level combination, it codes the run into a 6-bit binary value and the level into a 12-bit signed level value. Before coding an undefined run/level pair, a 6-bit "escape code" is used to denote that the next six to12 bits are not from the VLC table.

Encoding

DC Coefficients

Due to high redundancy between adjacent quantized DC coefficients of 8 x 8 blocks, the difference in DC values is encoded using VLC. The difference signals range from –255 to 255 in MPEG-1 and from –2047 to 2047 in MPEG-2. The size of the differential DC value (dct_dc_size) is found in Table 1. The size denotes the number of bits used to represent a particular value (e.g., if the differential DC value is –78, the table shows a size seven. The seven bits will be used to represent the value –78. The dct_dc_size value is variable length coded using table Table 5 or Table 6. After coding the size bits, –78 is represented as `111110,0110010` where the first six bits define the dct_dc_size and the next seven bits are used to code the value –78. Two different VLC codes are used for coding the dct_dc_size for luma and chroma.

The difference path for 8 x 8 blocks within a macroblock are calculated in the order shown in Figure 2.

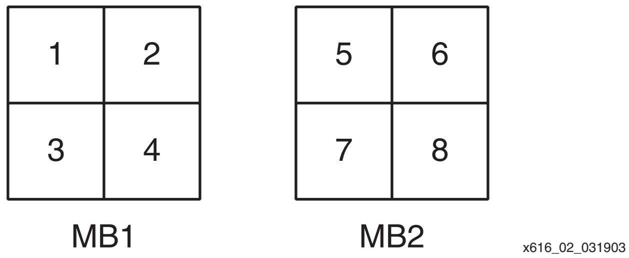


Figure 2: **Calculation Order in Adjacent Macroblocks**

Table 1: **Differential DC Additional Codes**

Differential DC	Size	Additional Code
–2047 to –1024	11	00000000000 to 01111111111
–1023 to –512	10	0000000000 to 0111111111
–511 to –256	9	000000000 to 011111111
–255 to –128	8	00000000 to 01111111
–127 to –64	7	0000000 to 0111111
–63 to –32	6	000000 to 011111
–31 to –16	5	00000 to 01111
–15 to –8	4	0000 to 0111
–7 to –4	3	000 to 011
–3 to –2	2	00 to 01
–1	1	0
0	0	
1	1	1
2 to 3	2	10 to 11
4 to 7	3	100 to 111
8 to 15	4	1000 to 1111
16 to 31	5	10000 to 11111
32 to 63	6	100000 to 111111
64 to 127	7	1000000 to 1111111
128 to 255	8	10000000 to 11111111
256 to 511	9	100000000 to 111111111
512 to 1023	10	1000000000 to 1111111111
1024 to 2047	11	10000000000 to 11111111111

AC Coefficients

The Huffman tables used for coding AC coefficients are selected based on the macroblock type and intra_vlc_format value according to Table 2. For MPEG-1 coding, only Table 7 is used. For MPEG-2, Table 8 is used for intra-coded blocks and Table 7 for non-intra coded blocks. In Table 7, there are two Huffman codes for the run/level combination of 0/1. The code "1s" is used if the 0/1 pair represents the first coefficient or the DC coefficient in the block. For subsequent 0/1 run/level pairs, "11s" is used as the Huffman code. The "s" in the code denotes the sign of the coefficient, "0" for positive and "1" for negative. The run/level pair for DC coefficient of "+1" and the EOB code has the same Huffman code. The differentiation is apparent since the EOB code will not be the first code in the block. In intra coding, since the DC value is coded separately, the first coded symbol is the first AC value. In this case, this first coded value can have a run/level of 0/1. A Huffman code of "11s" will not conflict with the EOB symbol. For a run/level pair that is not defined in the VLC Table 7 and Table 8, an escape code is used according to Table 9, followed by a 6-bit run symbol and 12-bit level symbol.

Table 2: **Selection of DCT Coefficient VLC Tables**

intra_vlc_format	0	1
intra coded blocks (macroblock_intra = 1)	Reference Table 7	Reference Table 8
non-intra coded blocks (macroblock_intra = 0)	Reference Table 7	Reference Table 7

Notes:
1. This table was taken from ISO/IEC 13818-2: 1995 (E), Table 7-3.

Table 7 and Table 8 are stored in ROMs. The run/level value is used to access the ROM and the corresponding variable length code is read out.

Decoding

DC Coefficient

Three predictor values are maintained for each color component. The predictor values are set at the start of the slice, or when a non-intra macroblock is decoded, or when a macroblock is skipped. The predictor values for different intra_dc_precisions are shown in Table 3.

Table 3: **Relationship Between intra_dc_precision and the Predictor Reset Value**

intra_dc_precision	Bits of Precision	Reset Value
0	8	128
1	9	256
2	10	512
3	11	1024

Notes:
1. This table was taken from ISO/IEC 13818-2: 1995 (E), Table 7-2.(Reference Item 1)

The DC coefficient is decoded by first getting the differential value from the coded stream. This differential_dc value is then added to the predictor to get the actual DC value. The new predictor value then becomes the actual DC value just decoded.

The decoding process can be described in the following manner (from ISO/IEC 13818-2: 1995 (E)(Reference Item 1))

QFS[0] shall be calculated from dc_dct_size and dc_dct_differential by any process equivalent to:

```
if ( dc_dct_size == 0 ) {
  dct_diff = 0;
} else {
  half_range = 2 ^ ( dc_dct_size - 1 ); Note ^ denotes power (not XOR)
                if ( dc_dct_differential >= half_range )
    dct_diff = dc_dct_differential;
  else
    dct_diff = (dc_dct_differential + 1) - (2 * half_range);
}
QFS[0] = dc_dct_pred[cc] + dct_diff;
dc_dct_pred[cc] = QFS[0]
```

Note: *dct_diff* and *half_range* are temporary variables that are not used elsewhere in this specification.

It is a requirement of the bitstream that QFS[0] shall lie in the range of zero to $(2^{8 + intra\ dc\ precision}) - 1$

AC Coefficient

The run/level pair is decoded from the Huffman code using a look-up table (LUT). There are three possible options for the Huffman code. If the code represents an EOB, all the remaining coefficients are set to "0". If the code represents an escape code, the next six bits represent the run and the following 12 bits represent the level.

If the VLC denotes a normal coefficient, then the run coefficients are set to zero and the following level coefficient is set to the level value depending on the value of s. When s == 0, the signed level is the same as level. When s == 1, the signed level is equal to (–level).

Huffman Implementation

The run/length pair is mapped to variable length codes from the code table. The code words are concatenated together and the output is partitioned into fixed length segments. The implementation shown in Figure 3 is similar to the one proposed by Lei and Sun(Reference Item 2).

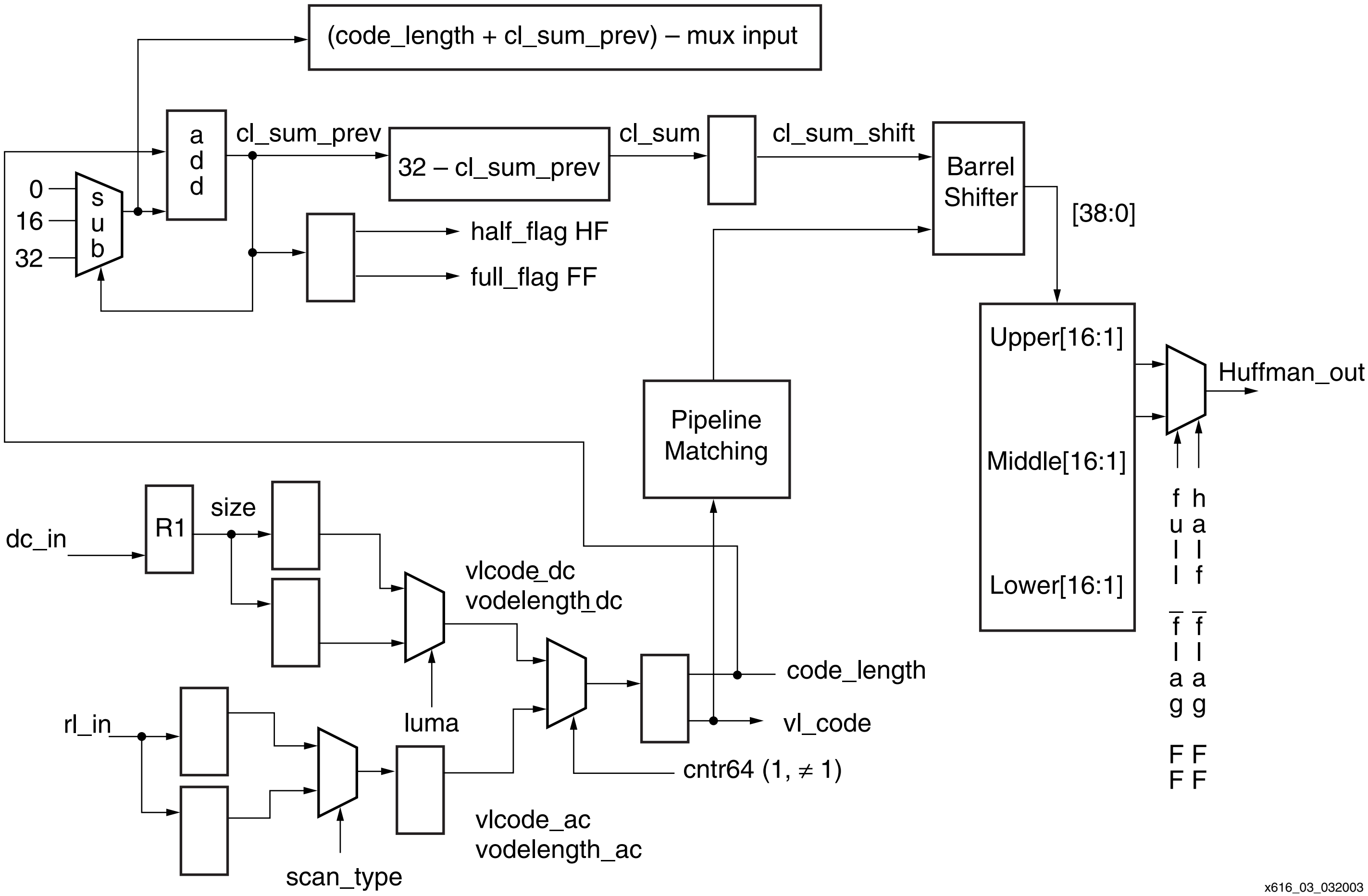
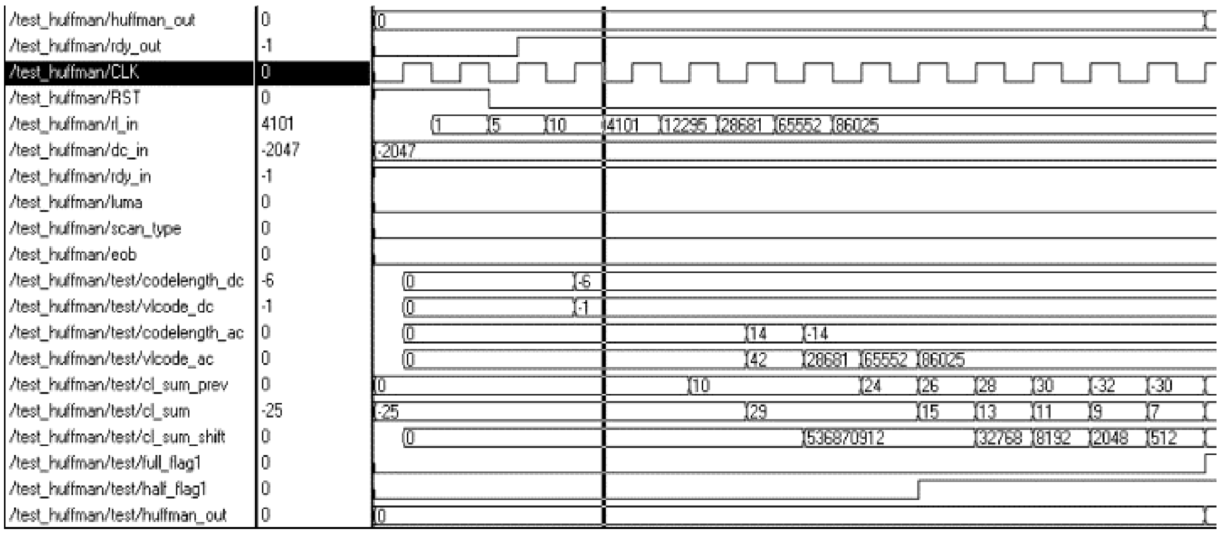


Figure 3: **Huffman Implementation Block Diagram**

Huffman Implementation Results

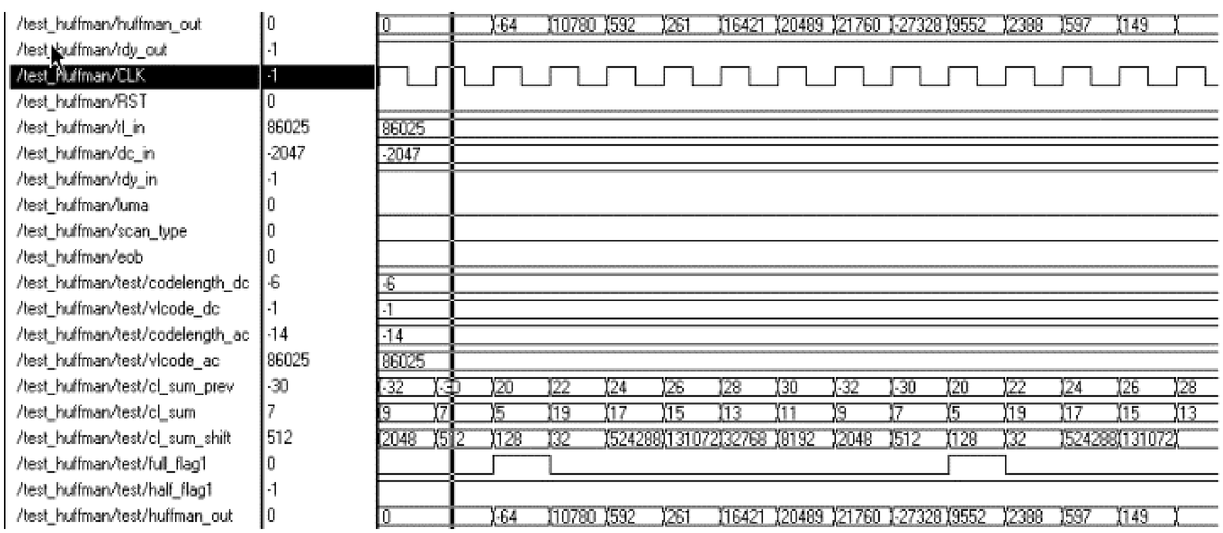
Table 4: **Huffman Implementation Results by Device Type**

Device	Post-Route (Synthesis Constraint)	LUTs and Flip Flops
XCV300E -8 BG352	100.3 (100 MHz)	785 LUTs 340 FFs
XCV300 -6 PQ256	100.3 (100 MHz)	777 LUTs 340 FFs
XC2S200 -6 FG256	100.3 (100 MHz)	777 LUTs 340 FFs



x616_04_031903

Figure 4: **Input "rl-in" to the Huffman Coder**



x616_05_031903

Figure 5: **Output "Huffman-out" from the Huffman Coder**

The performance can be improved by adding more pipeline stages in the designs. Keep the place and route effort level on "High". To get an update on performance and utilization, always re-run the designs using the latest Xilinx software.

Reference Design

A Huffman implementation reference design in both VHDL and Verilog is available on the Xilinx FTP site at:

ftp://ftp.xilinx.com/pub/applications/xapp/xapp616.zip

Conclusion

This Huffman coding application note describes the Huffman coding algorithms used in an MPEG-2 encoder. The design uses the Huffman table described in the MPEG-2 ISO/IEC 13818-2:1995 (E) document. The reference design files show the efficient implementation of the algorithms on Xilinx devices. The code can be used to target any Xilinx device. The code can be optimized by instantiating the adder/subtractor and multiplier units when targeting Virtex devices. Adding more pipeline stages can increase the performance of both blocks. The code takes in variable length data and sends out a fixed length (16-bit) Huffman coded data.

References

1. MPEG-2 Video IS document, ISO/IEC 13818-2: 1995(E)
2. Lei and Sun. An entropy coding system for digital HDTV applications. IEEE Transactions on Circuits and Systems for Video Technology, 1(1):147-155, March 1991
3. Image and Video Compression Standards, Second Edition, by Vasudev Bhaskaran and Konstantinos Konstantinides, ISBN 0-7923-9952-8
4. MPEG Video Compression Standard, by Mitchell, Pennebaker, Fogg, and LeGall, ISBN 0-412-08771-5

Appendix A

The following tables are referenced in this application note.

Table 5: **Variable Length Codes for dct_dc_size_luminance**

Variable Length Code	dct_dc_size_luminance
100	0
00	1
01	2
101	3
110	4
1110	5
1111 0	6
1111 10	7
1111 110	8
1111 1110	9
1111 1111 0	10
1111 1111 1	11

Notes:
1. This table is taken from Table B-12, ISO/IEC 13818-2: 1995(E)

Table 6: **Variable Length Codes for dct_dc_size_chrominance**

Variable Length Code	dct_dc_size_chrominance
`00`	0
`01`	1
`10`	2
`110`	3
`1110`	4
`1111 0`	5
`1111 10`	6
`1111 110`	7
`1111 1110`	8
`1111 1111 0`	9
`1111 1111 10`	10
`1111 1111 11`	11

Notes:

1. This table is taken from Table B-13, ISO/IEC 13818-2: 1995(E)

Table 7: **DCT Coefficients Table Zero**

Variable Length Code (Note 1)	Run	Level
`10` (Note 2)	End of Block	
`1 s` (Note 3)	0	1
`11 s` (Note 4)	0	1
`011 s`	1	1
`0100 s`	0	2
`0101 s`	2	1
`0010 1 s`	0	3
`0011 1 s`	3	1
`0011 0 s`	4	1
`0001 10 s`	1	2
`0001 11 s`	5	1
`0001 01 s`	6	1
`0001 00 s`	7	1
`0000 110 s`	0	4
`0000 100 s`	2	2
`0000 111 s`	8	1
`0000 101 s`	9	1
`0000 01 s`	Escape	

Table 7: **DCT Coefficients Table Zero *(Continued)***

Variable Length Code (Note 1)	Run	Level
0010 0110 s	0	5
0010 0001 s	0	6
0010 0101 s	1	3
0010 0100 s	3	2
0010 0111 s	10	1
0010 0011 s	11	1
0010 0010 s	12	1
0010 0000 s	13	1
0000 0010 10 s	0	7
0000 0011 00 s	1	4
0000 0010 11 s	2	3
0000 0011 11 s	4	2
0000 0010 01 s	5	2
0000 0011 10 s	14	1
0000 0011 01 s	15	1
0000 0010 00 s	16	1
0000 0001 1101 s	0	8
0000 0001 1000 s	0	9
0000 0001 0011 s	0	10
0000 0001 0000 s	0	11
0000 0001 1011 s	1	5
0000 0001 0100 s	2	4
0000 0001 1100 s	3	3
0000 0001 0010 s	4	3
0000 0001 1110 s	6	2
0000 0001 0101 s	7	2
0000 0001 0001 s	8	2
0000 0001 1111 s	17	1
0000 0001 1010 s	18	1
0000 0001 1001 s	19	1
0000 0001 0111 s	20	1
0000 0001 0110 s	21	1
0000 0000 1101 0 s	0	12
0000 0000 1100 1 s	0	13

Table 7: **DCT Coefficients Table Zero *(Continued)***

Variable Length Code (Note 1)	Run	Level
0000 0000 1100 0 s	0	14
0000 0000 1011 1 s	0	15
0000 0000 1011 0 s	1	6
0000 0000 1010 1 s	1	7
0000 0000 1010 0 s	2	5
0000 0000 1001 1 s	3	4
0000 0000 1001 0 s	5	3
0000 0000 1000 1 s	9	2
0000 0000 1000 0 s	10	2
0000 0000 1111 1 s	22	1
0000 0000 1111 0 s	23	1
0000 0000 1110 1 s	24	1
0000 0000 1110 0 s	25	1
0000 0000 1101 1 s	26	1
0000 0000 0111 11 s	0	16
0000 0000 0111 10 s	0	17
0000 0000 0111 01 s	0	18
0000 0000 0111 00 s	0	19
0000 0000 0110 11 s	0	20
0000 0000 0110 10 s	0	21
0000 0000 0110 01 s	0	22
0000 0000 0110 00 s	0	23
0000 0000 0101 11 s	0	24
0000 0000 0101 10 s	0	25
0000 0000 0101 01 s	0	26
0000 0000 0101 00 s	0	27
0000 0000 0100 11 s	0	28
0000 0000 0100 10 s	0	29
0000 0000 0100 01 s	0	30
0000 0000 0100 00 s	0	31
0000 0000 0011 000 s	0	32
0000 0000 0010 111 s	0	33
0000 0000 0010 110 s	0	34
0000 0000 0010 101 s	0	35

Table 7: **DCT Coefficients Table Zero *(Continued)***

Variable Length Code (Note 1)	Run	Level
0000 0000 0010 100 s	0	36
0000 0000 0010 011 s	0	37
0000 0000 0010 010 s	0	38
0000 0000 0010 001 s	0	39
0000 0000 0010 000 s	0	40
0000 0000 0011 111 s	1	8
0000 0000 0011 110 s	1	9
0000 0000 0011 101 s	1	10
0000 0000 0011 100 s	1	11
0000 0000 0011 011 s	1	12
0000 0000 0011 010 s	1	13
0000 0000 0011 001 s	1	14
0000 0000 0001 0011 s	1	15
0000 0000 0001 0010 s	1	16
0000 0000 0001 0001 s	1	17
0000 0000 0001 0000 s	1	18
0000 0000 0001 0100 s	6	3
0000 0000 0001 1010 s	11	2
0000 0000 0001 1001 s	12	2
0000 0000 0001 1000 s	13	2
0000 0000 0001 0111 s	14	2
0000 0000 0001 0110 s	15	2
0000 0000 0001 0101 s	16	2
0000 0000 0001 1111 s	27	1
0000 0000 0001 1110 s	28	1
0000 0000 0001 1101 s	29	1
0000 0000 0001 1100 s	30	1
0000 0000 0001 1011 s	31	1

Notes:

1. The last bit "s" denotes the sign of the level, "0" for positive "1" for negative.
2. "End of Block" shall not be the only code of the block.
3. This code shall be used for the first (DC) coefficient in the block
4. This code shall be used for all other coefficients
5. This table is taken from Table B-14, ISO/IEC 13818-2: 1995(E)

Table 8: **DCT Coefficients Table One**

Variable Length Code (Note 1)	Run	Level
`0110` (Note 2)	End of Block	
`10 s`	0	1
`010 s`	1	1
`110 s`	0	2
`0010 1 s`	2	1
`0111 s`	0	3
`0011 1 s`	3	1
`0001 10 s`	4	1
`0011 0 s`	1	2
`0001 11 s`	5	1
`0000 110 s`	6	1
`0000 100 s`	7	1
`1110 0 s`	0	4
`0000 111 s`	2	2
`0000 101 s`	8	1
`1111 000 s`	9	1
`0000 01`	Escape	
`1110 1 s`	0	5
`0001 01 s`	0	6
`1111 001 s`	1	3
`0010 0110 s`	3	2
`1111 010 s`	10	1
`0010 0001 s`	11	1
`0010 0101 s`	12	1
`0010 0100 s`	13	1
`0001 00 s`	0	7
`0010 0111 s`	1	4
`1111 1100 s`	2	3
`1111 1101 s`	4	2
`0000 0010 0 s`	5	2
`0000 0010 1 s`	14	1
`0000 0011 1 s`	15	1
`0000 0011 01 s`	16	1

Table 8: **DCT Coefficients Table One *(Continued)***

Variable Length Code (Note 1)	Run	Level
1111 011 s	0	8
1111 100 s	0	9
0010 0011 s	0	10
0010 0010 s	0	11
0000 0011 00 s	2	4
0000 0001 1100 s	3	3
0000 0001 0010 s	4	3
0000 0001 1110 s	6	2
0000 0001 0101 s	7	2
0000 0001 0001 s	8	2
0000 0001 1111 s	17	1
0000 0001 1010 s	18	1
0000 0001 1001 s	19	1
0000 0001 0111 s	20	1
0000 0001 0110 s	21	1
1111 1010 s	0	12
1111 1011 s	0	13
1111 1110 s	0	14
1111 1111 s	0	15
0000 0000 1011 0 s	1	6
0000 0000 1010 1 s	1	7
0000 0000 1010 0 s	2	5
0000 0000 1001 1 s	3	4
0000 0000 1001 0 s	5	3
0000 0000 1000 1 s	9	2
0000 0000 1000 0 s	10	2
0000 0000 1111 1 s	22	1
0000 0000 1111 0 s	23	1
0000 0000 1110 1 s	24	1
0000 0000 1110 0 s	25	1
0000 0000 1101 1 s	26	1
0000 0000 0111 11 s	0	16
0000 0000 0111 10 s	0	17
0000 0000 0111 01 s	0	18

Table 8: **DCT Coefficients Table One *(Continued)***

Variable Length Code (Note 1)	Run	Level
0000 0000 0111 00 s	0	19
0000 0000 0110 11 s	0	20
0000 0000 0110 10 s	0	21
0000 0000 0110 01 s	0	22
0000 0000 0110 00 s	0	23
0000 0000 0101 11 s	0	24
0000 0000 0101 10 s	0	25
0000 0000 0101 01 s	0	26
0000 0000 0101 00 s	0	27
0000 0000 0100 11 s	0	28
0000 0000 0100 10 s	0	29
0000 0000 0100 01 s	0	30
0000 0000 0100 00 s	0	31
0000 0000 0011 000 s	0	32
0000 0000 0010 111 s	0	33
0000 0000 0010 110 s	0	34
0000 0000 0010 101 s	0	35
0000 0000 0010 100 s	0	36
0000 0000 0010 011 s	0	37
0000 0000 0010 010 s	0	38
0000 0000 0010 001 s	0	39
0000 0000 0010 000 s	0	40
0000 0000 0011 111 s	1	8
0000 0000 0011 110 s	1	9
0000 0000 0011 101 s	1	10
0000 0000 0011 100 s	1	11
0000 0000 0011 011 s	1	12
0000 0000 0011 010 s	1	13
0000 0000 0011 001 s	1	14
0000 0000 0001 0011 s	1	15
0000 0000 0001 0010 s	1	16
0000 0000 0001 0001 s	1	17
0000 0000 0001 0000 s	1	18
0000 0000 0001 0100 s	6	3

Table 8: **DCT Coefficients Table One *(Continued)***

Variable Length Code (Note 1)	Run	Level
0000 0000 0001 1010 s	11	2
0000 0000 0001 1001 s	12	2
0000 0000 0001 1000 s	13	2
0000 0000 0001 0111 s	14	2
0000 0000 0001 0110 s	15	2
0000 0000 0001 0101 s	16	2
0000 0000 0001 1111 s	27	1
0000 0000 0001 1110 s	28	1
0000 0000 0001 1101 s	29	1
0000 0000 0001 1100 s	30	1
0000 0000 0001 1011 s	31	1

Notes:

1. The last bit "s" denotes the sign of the level, "0" for positive "1" for negative.
2. "End of Block" is not the only code of the block.
3. This code is used for the first (DC) coefficient in the block
4. This code is used for all other coefficients
5. This table is taken from Table B-15, ISO/IEC 13818-2: 1995(E)

Table 9: **Encoding of Run and Level Following an Escape Code**

Fixed Length Code	Run
0000 00	0
0000 01	1
0000 10	2
...	...
...	...
...	...
...	...
1111 11	63

Fixed Length Code	Signed Level
1000 0000 0001	–2047
1000 0000 0010	–2046
...	...
1111 1111 1111	–1
0000 0000 0000	forbidden
0000 0000 0001	+1
...	...
0111 1111 1111	+2047

Notes:

1. This table is taken from Table B-16, ISO/IEC 13818-2: 1995(E)

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
04/22/03	1.0	Initial Xilinx release.