CANON RAW V2 LOSSLESS JPEG DECOMPRESSION + UNSLICING

v1.02
Laurent Clévy
@lorenzo2472
http://lclevy.free.fr/cr2

```
$ craw2tool.exe -i ../pics/IMG_0596_sraw.CR2
2592x1728, 15bits, 3 comp, YUV422/sraw/sraw2, "Canon EOS 60D", [5, 864, 864]
```

from TIFF #IFD3:
StripOffset = 0x31558c (offset to RAW image)
StripByteCounts = 0x504f25 (RAW image size)
CR2 slices layout = [5, 864, 864]

FILE CONTENT

TIFF header, CR2 header, IFDs, other images...

```
0x31558c:                                       ffd8

0x31558e:                                           ffc4
0x315590:  00440000 01050101 01010101 01010101
0x3155a0:  00000000 01020304 05060708 090a0b0c
0x3155b0:  0d0e0f01 00030101 01010101 01010101
0x3155c0:  01010100 00010203 04050607 08090a0b
0x3155d0:  0c0d0e0f

0x3155d4:                ffc30011 0f06c00a 20030121
0x3155e0:  00021100 031100

0x3155e7:                     ff da000c03 01000210
0x3155f0:  03100100 00

0x3155f5:                ff00e0 0ba28968 c700b088
0x315600:  507b5114 c396a84a c14c31d1 047214f6
...

...
0x81a4a0:  750e4b2c 5b556279 da5b81d1 75d33fff
0x81a4b0:  d9
```

SOI MARKER

Marker	0xffd8 (Start Of Image)

DHT HEADER

Marker	0xffc4 (Define Huffman Table)
Length	0x44
Table class/Huffman table index	0 (DC)/0
Number of huffman codes of length i	[00,01,05,01,01,01,01,01,01,01,01,01,01,00,00,00]
Values associated with each huffman code	[00,[01,02,03,04,05],06,07,08,09,0a,0b,0c,0d,0e,0f]
Table class/Huffman table index	0 (DC)/1
Number of huffman codes of length i	[00,03,01,01,01,01,01,01,01,01,01,01,01,01,01,00]
Values associated with each huffman code ...	[[00,01,02],03,04,05,06,07,08,09,0a,0b,0c,0d,0e,0f]

SOF3 HEADER

Marker	0xffc3 (Start Of Frame 3: Lossless sequential)
Length	0x11
Sample precision	15 (YCbCr). Could be 12 or 14 for RGGB
Number of lines	0x6c0 (1728)
Number of samples per line	0xa20 (2592)
Number of image components per frame	3 (for YCbCr). Could be 2 or 4 for RGGB

Component identifier	Horizontal sampling Factor	Vertical Sampling Factor	Quantization table
1	2	1	0
2	1	1	0
3	1	1	0

SOS HEADER

Marker	0xffda (Start Of Scan)
Length	0x0c
Number of components	3 (for YCrCb). Could be 2 or 4 for RGGB

Scan component selector	DC table	AC table
1	0	0
2	1	0
3	1	0

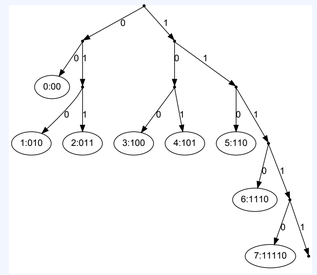
Start of spectral or predictor selection	1 (always 1)
End of spectral selection	0 (always 0)
Successive approximation bit positions	0 (always 0)

SCAN DATA

Stream of:
Length of following difference value — Huffman encoded
Difference value — Difference between previous column value of same component
...

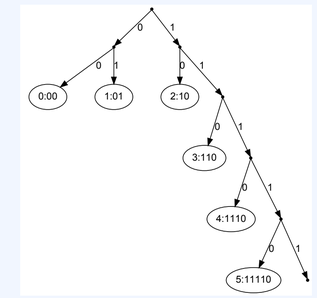
HUFFMAN TABLE & TREE #0

```
DC, Destination= 0
 maxCodeLen=13
 nbCodePerSizes: 00010501010101010101010000000
  number of codes of length  1 bits: 0 ( )
  number of codes of length  2 bits: 1 ( 00:00, )
  number of codes of length  3 bits: 5 ( 01:010, 02:011, 03:100, 04:101, 05:110, )
  number of codes of length  4 bits: 1 ( 06:1110, )
  number of codes of length  5 bits: 1 ( 07:11110, )
  number of codes of length  6 bits: 1 ( 08:111110, )
  number of codes of length  7 bits: 1 ( 09:1111110, )
  number of codes of length  8 bits: 1 ( 0a:11111110, )
  number of codes of length  9 bits: 1 ( 0b:111111110, )
  number of codes of length 10 bits: 1 ( 0c:1111111110, )
  number of codes of length 11 bits: 1 ( 0d:11111111110, )
  number of codes of length 12 bits: 1 ( 0e:111111111110, )
  number of codes of length 13 bits: 1 ( 0f:1111111111110, )
  number of codes of length 14 bits: 0 ( )
  number of codes of length 15 bits: 0 ( )
  number of codes of length 16 bits: 0 ( )
 totalNumberOfCodes= 16
```



HUFFMAN TABLE & TREE #1

```
DC, Destination= 1
 maxCodeLen=15
 nbCodePerSizes: 00030101010101010101010101010100
  number of codes of length  1 bits: 0 ( )
  number of codes of length  2 bits: 3 ( 00:00, 01:01, 02:10, )
  number of codes of length  3 bits: 1 ( 03:110, )
  number of codes of length  4 bits: 1 ( 04:1110, )
  number of codes of length  5 bits: 1 ( 05:11110, )
  number of codes of length  6 bits: 1 ( 06:111110, )
  number of codes of length  7 bits: 1 ( 07:1111110, )
  number of codes of length  8 bits: 1 ( 08:11111110, )
  number of codes of length  9 bits: 1 ( 09:111111110, )
  number of codes of length 10 bits: 1 ( 0a:1111111110, )
  number of codes of length 11 bits: 1 ( 0b:11111111110, )
  number of codes of length 12 bits: 1 ( 0c:111111111110, )
  number of codes of length 13 bits: 1 ( 0d:1111111111110, )
  number of codes of length 14 bits: 1 ( 0e:11111111111110, )
  number of codes of length 15 bits: 1 ( 0f:111111111111110, )
  number of codes of length 16 bits: 0 ( )
 totalNumberOfCodes= 16
```



CANON RAW JPEG PROPERTIES

Sample precision	15 bits	15 bits	14 or 12 bits
Number of components	3	3	4 or 2
Horizontal Sampling Factor	2	2	1
Vertical Sampling Factor	1	2	1
Image type	sraw/sraw2	mraw/sraw1	normal RAW (RGGB)
Decompressed data layout	Y1 Y2 Cb Cr...	Y1 Y2 Y3 Y4 Cb Cr	RGRGRGRGR...GBGBGBGB...
Image components layout (x means missing data)	Y1 Cb Cr Y2 x x Y1 Cb Cr Y2 x x	Y1 Cb Cr Y2 x x Y3 x x Y4 x x	R G R G R G R G G B G B G B G B

LOSSLESS JPEG DECOMPRESSION (ITU-T81)

Uncompression algorithm:
1. Find Huffman code in compressed stream for the following difference code length. SOS section tells which Huffman table to use (here #0 for Y, #1 for Cb and Cr). Length 0 means difference value is 0.
2. Move the stream pointer [Huffman_code_length] bits forward, read [Huffman_code_value] bits = difference_code (see *1).
3. We can compute the current component value by adding the previous value for this component and difference value.
4. goto 1 until end of scan data, to fill image slice per slice.

```
Resulting output after decompression (missing Cb and Cr values must be interpolated before conversion to RGB):
   Y    Cb    Cr |   Y    Cb    Cr |   Y    Cb    Cr |   Y    Cb    Cr |   Y    Cb    Cr |   Y    Cb    Cr
  47    -3    -2 |  42     .     . |  40    -2    -2 |  40     .     . |  42    -2    -2 |  42     .     .
```

SCAN DATA (COMPRESSED)

```
   f    f    0    0    e    0    0    b    a    2    8    9    6    8    c    7    0    0    b    0
1111 1111 0000 0000 1110 0000 0000 1011 1010 0010 1000 1001 0110 1000 1100 0111 0000 0000 1011 0000
```

Huffman code (Diff. code length)	Difference code = Diff. value	Previous value	Component value
111111111110(14)	00000000101110 = -16337	16384 (see below)	47 (Y, unsigned)
100(3)	010 = -5	47	42 (Y, unsigned)
10(2)	00 = -3	16384 = 0	16381 = -3 (Cb, signed)
10(2)	01 = -2	16384 = 0	16382 = -2 (Cr, signed)
011(2)	01 = -2	42	40 (Y)
00(0)	0	40	40 (Y)
01(1)	1 = 1	-3	-2 (Cb)
00(0)	0	-2	-2 (Cr)
011(2)	10 = 2	40	42 (Y)
00(0)	0	42	42 (Y)
00(0)	0	-2	-2 (Cb)
00(0)	0	-2	-2 (Cr)

Difference code length	Difference codes (*1)	Difference value
0		0
1	0, 1	-1, 1
2	00, 01, 10, 11	-3, -2, 2, 3
3	000 ... 111	-7, -6, -5, -4, 4, 5, 6, 7
4	0000 ... 1111	-15 ... 15
...	...	...
15	000000000000000 ... 111111111111111	-32768 ... 32768

Default previous value:
2^(jpeg->bits-1)
here 2^14 = 16384.

0xff00: to encode 0xff in a JPEG stream, to avoid ambiguity with markers, write 0xff followed by 0x00, and 0x00 will be ignored at decoding.

UNSLICING

CR2 slices layout is [5, 864, 864]. Means "first 5 slices of 864, and a last one of 864". Horizontal sampling factor is 2, thus here slice width is 432 and not 864.
Slices must be filled one by one from left to right and within slices from left to right, top / bottom.