Color

cli

2021.09.08

Table of Contents

[ASpec Draft 1](#_Toc91173380)

[CSC - Color Space Conversion 1](#_Toc91173381)

[x.x.1 Algorithm Background 1](#_Toc91173382)

[x.x.2 Hardware Implementation 2](#_Toc91173383)

[HDR - Color Management on Dynamic Range 3](#_Toc91173384)

[x.x.1 General Information 3](#_Toc91173385)

[x.x.2 Algorithm Background 4](#_Toc91173386)

[x.x.3 Hardware Implementation 6](#_Toc91173387)

# ASpec Draft

## CSC - Color Space Conversion

The Color Space Conversion module is used for conversion between YUV and RGB color space. As the input and output are both digital and electronic signal, they are specifically called as Y’CbCr or R’G’B’ data.

### x.x.1 Algorithm Background

The conversion between R’G’B’ and Y’CbCr is carried out in this submodule (the CSC0 and CSC1 modules in the DPE architecture diagram). The signal range can be either full or limited.

The conversion is carried out by a linear mapping determined by a pre-configured 3x4 matrix .

For R’G’B’-to-Y’CbCr conversion,

For Y’CbCr-to-R’G’B’ conversion,

After the conversion, the color data needs to be clamped to [0, 1023] for UQ10.0 precision.

### x.x.2 Hardware Implementation

Data precision:

* Input: UQ10.0 R’G’B’ or Y’CbCr data
* Output: UQ10.0 R’G’B’ or Y’CbCr data
* Intermediate data: Q14.12

Let d\_in[3] and d\_out[3] denote the 3 channels of the input and output data. The CSC matrix is denoted as and is configured by the group of registers **dpe\_csc\_m\_ij**. Intermediate data is denoted as d\_inter[3] and d\_inter2[3].

The CSC operation can be implemented as follows:

# input: UQ10.0  
for r in range(3):  
 d\_inter2[r] = 0  
 for c in range(3):  
 d\_inter2[r] += M[r][c] \* d\_inter[c] # d\_inter2: Q14.12  
 d\_inter2[r] += m[r][3] # d\_inter2: Q14.12  
# 3) clamp result to [0, 1023] and get UQ10.0 output  
for k in range(3):  
 d\_inter2[k] = max(0, min(1023, d\_inter2[k])) # d\_inter2: UQ10.0  
 # get output data d\_out[3] with precision UQ10.0 from d\_inter2[3]

The precision for the CSC0 matrix coefficients are shown below, and the same is applied for CSC1 matrix coefficients (dpe\_csc1\_m\_00 ~ dpe\_csc1\_m\_23):

|  |  |  |  |
| --- | --- | --- | --- |
| Registers | Precision | Default Value | Notes |
| dpe\_csc0\_m\_00 | Q02.12 | 1 |  |
| dpe\_csc0\_m\_01 | Q02.12 | 0 |  |
| dpe\_csc0\_m\_02 | Q02.12 | 0 |  |
| dpe\_csc0\_m\_03 | Q11.12 | 0 |  |
| dpe\_csc0\_m\_10 | Q02.12 | 0 |  |
| dpe\_csc0\_m\_11 | Q02.12 | 1 |  |
| dpe\_csc0\_m\_12 | Q02.12 | 0 |  |
| dpe\_csc0\_m\_13 | Q11.12 | 0 |  |
| dpe\_csc0\_m\_20 | Q02.12 | 0 |  |
| dpe\_csc0\_m\_21 | Q02.12 | 0 |  |
| dpe\_csc0\_m\_22 | Q02.12 | 1 |  |
| dpe\_csc0\_m\_23 | Q11.12 | 0 |  |

There are also registers to control the enable signal of the CSC modules:

|  |  |  |  |
| --- | --- | --- | --- |
| Registers | Precision | Default Value | Notes |
| dpe\_csc0\_en | UQ01.00 | 0 | Enable signal for CSC0 |
| dpe\_csc1\_en | UQ01.00 | 0 | Enable signal for CSC1 |

## HDR - Color Management on Dynamic Range

The HDR module is responsible for the conversion between different color spaces and gamuts. The module supports both SDR and HDR signals.

### x.x.1 General Information

#### x.x.1.1 CM - Color Matrix

The HDR module supports the following features of a color matrix:

* Color spaces:
  + SDR:
    - ITU-R Rec BT.601
    - ITU-R Rec BT.709 / sRGB
  + HDR:
    - ITU-R Rec BT.2020

#### x.x.1.2 Transfer Functions (Gamma Correction)

The HDR module supports the following transfer functions to convert the input color signal between electronic and optical forms:

* Opto-Electronic Transfer Function (EOTF)
  + SDR:
    - BT.1886
    - sRGB
  + HDR:
    - BT.2100 PQ (PQ: Perceptual Quantizer)
* Electro-Optical Transfer Function (OETF)
  + SDR:
    - BT.601/BT.709/BT.2020
  + HDR:
    - BT.2100 HLG (HLG: Hybrid Log-Gamma)
* ~~Opto-Optical Transfer Function (OOTF)~~
  + ~~HDR:~~
    - ~~BT.2100 HLG~~

Transfer functions for HDR contents are specified in ITU-R Rec BT.2100, while other transfer functions are for SDR contents.

#### x.x.1.3 HDR Standard Support

The following standards are supported by the HDR module:

* PQ10
* HDR10
* HLG10

Specifically, the HDR module supports both PQ and HLG signals standardized in ITU-R Rec BT.2100. The bitwidth of these signals is limited to 10-bit. Static HDR metadata stardardized in SMPTE ST 2086 and the MaxCLL (Maximum Content Light Level) and the MaxFALL (Maximum Frame Average Light Level) metadata are supported.

The supported conversions are specified as follows:

* SDR signal <-> PQ/HLG signal (need tone mapping support)
* ~~PQ signal <-> HLG signal~~

For the conversion between SDR and PQ/HLG signal, the SDR signal should be a combination of one of the SDR color spaces and one of the appropriate SDR transfer functions; the PQ/HLG signal can be in either SDR or HDR color spaces.

### x.x.2 Algorithm Background

The goal for HDR module is to convert input R’G’B’ signal from one color space to another one, with the transfer function being possibly changed according to the destination color space.

#### x.x.2.1 Degamma and Tone Mapping

Degamma is the operation of removing the gamma from the input electronic R’G’B’ signal. The nonlinear/electronic R’G’B’ signal will be converted to linear/optical RGB signal after this operation.

Tone mapping is the operation to change the dynamic range of the input linear signal. It can be one of the following 3 cases: SDR to HDR, HDR to SDR, or HDR to HDR. To perform tone mapping, we need additional HDR static metadata to compute a tone mapping curve, and then perform tone mapping to map input signal to smaller/bigger dynamic range. Finally, we do some post-processing on the result signal to enhance its saturation.

According to the destination color space and transfer function, this procedure consists of the following steps:

* Step 1: apply EOTF / inverse OETF to the input nonlinear signal
* Step 2 (Optional): apply tone mapping curve
* Step 3 (Optional): perform saturation enhancement
* Step 4: perform a linear stretching operation

The step 1 above is mandatory to convert the input signal into its linear form. Step 2 and 3 are optional procedures for tone mapping. Step 4 is also a post-processing step on the result signal.

In step 3, let , and be the R, G, and B channel of the optical signal ( for one of the channels). The parameters for the RGB-to-Y conversion are , and respectively. Let be the converted luminance. Let be one of the color channels with the saturation enhanced.

The RGB to Y conversion is as follows:

The nonlinear operation of saturation enhancement is performed as follows:

where is the saturation enhancement parameter.

Finally, in step 4, a linear stretch operation is performed, where and are the stretching parameters:

#### x.x.2.2 Color Space Conversion between Different Color Bases

During this procedure, the color representation of the input linear RGB signal is converted from one color basis to another by the following 2 steps:

* RGB-to-XYZ conversion for the input color space
* XYZ-to-RGB conversion for the output color space

The 2 conversions are carried out as linear mappings determined by the 3x3 RGB-to-XYZ and 3x3 XYZ-to-RGB matrices. Let and denote the input and output signal in this procedure. Let the RGB-to-XYZ matrix and XYZ-to-RGB matrix be and respectively. The 2 matrices can be merged into 1 matrix . The procedure can be described as the following:

#### x.x.2.3 Gamma

Gamma is the operation of applying appropriate gamma function for the linear RGB signal. The linear/optical RGB signal is converted to nonlinear/electronic R’G’B’ signal after this operation. Either an inverse EOTF (for the case where EOTF is applied in Degamma) or an OETF (for the case where inverse OETF is applied in Degamma) is applied.

The Gamma operation is performed by utilizing a pre-configured 1-D LUT for the nonlinear operation on each one of the 3 channels of the result signal from color space conversion.

After the gamma operation, a linear stretch operation is performed on the output signal , where and are the stretching parameters:

### x.x.3 Hardware Implementation

**Note:** all the bit clipping operations may need further check, especially the *NEED CHECK* portions.

Data format and precision:

* Input: UQ10.0 R’G’B’ data
* Output: UQ10.0 R’G’B’ data

Firstly, 2 basic procedures need to be defined, that is, the procedure to find the index for linear interpolation, and the procedure to carry out linear interpolation. They are defined by the pseudo code below:

def find\_interp\_idx(x, L):  
 """  
 Find the index for linear interpolation.  
 x: input data;   
 L: input LUT (array).  
 Return linear interpolation index @p idx in range [0, len(L)].  
 """  
 if x < L[0]:  
 idx = 0  
 elif x >= L[len(L)-1]: # 20210908: > to >=  
 idx = len(L)  
 else:  
 idx = -1 # invalid case  
 for i in range(1, len(L)):  
 if x >= L[i-1] and x < L[i]: # 20210908: <= to <  
 idx = i  
 break  
 assert idx >= 0 and idx <= len(L)  
 return idx  
   
def linterp(x, Lx, Ly, Ls):  
 """  
 Perform linear interpolation.  
 x: input data;   
 Lx: input LUT;   
 Ly: output LUT;  
 Ls: LUT for the slope terms (Ly[i]-Ly[i-1]) / (Lx[i]-Lx[i-1]);  
 size: len(Lx)-1  
 Return linear-interpolated value @p y.  
 """  
 # find interpolation index  
 i = find\_interp\_idx(x, Lx) # i: UQ6.0  
 if i == 0:  
 return Ly[0]  
 elif i == len(Lx): # 20210908 fix: Ly -> Lx  
 return Ly[i-1]  
 else:  
 # linear interpolation  
 # y = Ly[i-1] + (Ly[i]-Ly[i-1]) / (Lx[i]-Lx[i-1]) \* (x-Lx[i-1])  
 # = Ly[i-1] + Ls[i-1] \* (x-Lx[i-1])  
 sub\_term = x - Lx[i-1]  
 mul\_term = Ls[i-1] \* sub\_term  
 y = Ly[i-1] + mul\_term  
 return y

#### x.x.3.1 Degamma and Tone Mapping

Details of the above steps are as follows:

##### (1) Step 1 and 2

For hardware implementation, step 1 and 2 are performed together by merging the operations into a pre-configured 1-D lookup table (LUT) for the nonlinear operations on each one of the 3 channels of the input signal.

Let denote one of the 3 channels of the input electronic signal, is the result of step 2. Each channel of the input signal is degammaed and then tonemapped using lookup tables and for electronic and optical signals with linear interpolation. The LUTs and are configured as 2 groups of registers **dpe\_hdr\_dg\_lutx\_ij** and **dpe\_hdr\_dg\_luty\_ij**. In order to perform linear interpolation, an LUT of pre-computed linear interpolation slope term is also configured. It is determined by the group of registers **dpe\_hdr\_dg\_luts\_ij**.

Step 1 and 2 can described as follows:

# perform combinations of degamma and tonemapping  
# E\_inter: UQ10.0  
# linterp():  
# - sub\_term: UQ10.0  
# - mul\_term: UQ0.26 \* UQ10.0 -> UQ10.26, clip to UQ1.26  
# - y: UQ1.29 + UQ1.26 -> clip to UQ1.29  
# O: UQ1.29  
O = linterp(E\_inter, L\_E, L\_O, L\_s\_e2o)

Data and related precisions for the LUT :

// L\_E (370 bits)  
double dpe\_hdr\_dg\_lutx[37] = {  
 0, 2, 4, 6, 8, 10, 12, 14, 18, 22,   
 26, 34, 42, 50, 58, 74, 90, 106, 122, 154,   
 186, 218, 250, 314, 378, 442, 506, 570, 634, 698,   
 762, 826, 890, 922, 954, 986, 1023  
}; // UQ10.0

The related registers and their precision are summarized in the tables below:

1. Registers for the LUT

|  |  |  |  |
| --- | --- | --- | --- |
| Registers | Precision | Default Value | Notes |
| dpe\_hdr\_dg\_luty\_00 | UQ00.29 | 0 | L\_O[0] |
| dpe\_hdr\_dg\_luty\_01 ~ dpe\_hdr\_dg\_luty\_08 | UQ00.26 | 0 | L\_O[1] ~ L\_O[8] |
| dpe\_hdr\_dg\_luty\_09 ~ dpe\_hdr\_dg\_luty\_18 | UQ01.23 | 0 | L\_O[9] ~ L\_O[18] |
| dpe\_hdr\_dg\_luty\_19 ~ dpe\_hdr\_dg\_luty\_26 | UQ01.19 | 0 | L\_O[19] ~ L\_O[26] |
| dpe\_hdr\_dg\_luty\_27 ~ dpe\_hdr\_dg\_luty\_36 | UQ01.16 | 0 | L\_O[27] ~ L\_O[36] |

1. Registers for the LUT

|  |  |  |  |
| --- | --- | --- | --- |
| Registers | Precision | Default Value | Notes |
| dpe\_hdr\_dg\_luts\_00 ~ dpe\_hdr\_dg\_luts\_08 | UQ00.26 | 0 | L\_s\_e2o[0] ~ L\_s\_e2o[8] |
| dpe\_hdr\_dg\_luts\_09 ~ dpe\_hdr\_dg\_luts\_16 | UQ00.23 | 0 | L\_s\_e2o[9] ~ L\_s\_e2o[16] |
| dpe\_hdr\_dg\_luts\_17 ~ dpe\_hdr\_dg\_luts\_35 | UQ00.19 | 0 | L\_s\_e2o[17] ~ L\_s\_e2o[35] |

##### (2) Step 3

In step 3, let , and be the R, G, and B channel of the optical signal ( for one of the channels). The parameters for the RGB-to-Y conversion are , and respectively. Let be the converted luminance. Let be one of the color channels with the saturation enhanced. The nonlinear operation is done via lookup tables , and the additional slope term LUT .

# 1) RGB-to-Y conversion  
Y = kr\*O[0] + kg\*O[1] + kb\*O[2] # Y: UQ1.29  
# 2) perform saturation enhancement  
for k in range(3):  
 X = 0 if Y == 0 else O[k] / Y # X: UQ5.6, O[k]: UQ1.29  
 # O\_new[k] = O[k] \* (O[k]/Y)^p   
 # where p is the saturation enhancement parameter  
 # linterp():  
 # - sub\_term: UQ5.6 - UQ5.6 -> clip to UQ5.6  
 # - mul\_term: UQ5.12 \* UQ5.6 -> UQ10.18 -> clip to UQ3.6  
 # - y: UQ3.6 + UQ3.6 -> clip to UQ3.6  
 # O\_new: UQ1.29 \* UQ3.6 -> UQ4.35 -> clamp to UQ1.29 ([0, 1])  
 O\_new[k] = max(0, min(1, O[k] \* linterp(X, L\_Sx, L\_Sy, L\_Ss)))

Data and related precisions for the LUT :

// L\_Sx (102 bits)  
double dpe\_hdr\_tm\_lutsx[13] = {  
 0.000000 /\* UQ0.6 \*/, 0.015625 /\* UQ0.6 \*/, 0.031250 /\* UQ0.6 \*/,  
 0.062500 /\* UQ0.6 \*/, 0.125000 /\* UQ0.6 \*/, 0.234375 /\* UQ0.6 \*/,  
 0.468750 /\* UQ0.6 \*/, 0.93750 /\* UQ5.5 \*/, 1.87500 /\* UQ5.5 \*/,  
 3.75000 /\* UQ5.5 \*/, 7.50000 /\* UQ5.5 \*/, 15.00000 /\* UQ5.5 \*/,  
 30.00000 /\* UQ5.5 \*/  
};

The related registers are shown in the tables below:

1. The RGB-to-Y parameters , and

|  |  |  |  |
| --- | --- | --- | --- |
| Registers | Precision | Default Value | Notes |
| dpe\_hdr\_tm\_rgb2y\_kr | UQ00.12 | 0 |  |
| dpe\_hdr\_tm\_rgb2y\_kg | UQ00.12 | 0 |  |
| dpe\_hdr\_tm\_rgb2y\_kb | UQ00.12 | 0 |  |

1. The output data LUT

|  |  |  |  |
| --- | --- | --- | --- |
| Registers | Precision | Default Value | Notes |
| dpe\_hdr\_tm\_lutsy\_00 ~ dpe\_hdr\_tm\_lutsy\_08 | UQ01.06 | 0 | L\_Sy[0] ~ L\_Sy[8] |
| dpe\_hdr\_tm\_lutsy\_09 ~ dpe\_hdr\_tm\_lutsy\_12 | UQ03.06 | 0 | L\_Sy[9] ~ L\_Sy[12] |

1. The fraction term LUT for linear interpolation

|  |  |  |  |
| --- | --- | --- | --- |
| Registers | Precision | Default Value | Notes |
| dpe\_hdr\_tm\_lutss\_00 ~ dpe\_hdr\_tm\_lutss\_06 | UQ05.07 | 0 | L\_Ss[0] ~ L\_Ss[6] |
| dpe\_hdr\_tm\_lutss\_07 ~ dpe\_hdr\_tm\_lutss\_11 | UQ00.12 | 0 | L\_Ss[7] ~ L\_Ss[11] |

##### (3) Step 4

Finally, in step 4, a linear stretch operation is performed, where and are the stretching parameters:

# 1) perform linear stretch operation  
for k in range(3):  
 O\_out[k] = a1\*O\_out[k] + b1 # O\_out: clip to Q1.29  
 # clamp result to range [0, 1]  
 O\_out[k] = max(0, min(1, O\_out[k])) # O\_out: UQ1.29

Related registers are shown in the table below:

|  |  |  |  |
| --- | --- | --- | --- |
| Registers | Precision | Default Value | Notes |
| dpe\_hdr\_dg\_stretch\_a | UQ14.12 | 1 |  |
| dpe\_hdr\_dg\_stretch\_b | Q14.12 | 0 |  |

#### x.x.3.2 Color Space Conversion

Let and denote the 3 channels of the input and output optical signal. The CSC matrix is denoted as and is configured by the group of registers **dpe\_hdr\_csc\_m\_ij**. After the CSC operation, the output must be clamped to range [0, ], where is the peak luminance (maximum display light) of the output/destination signal (unit: ).

The procedure can be described in the pseudo code below:

# 1) perform color space conversion: RGB-to-RGB  
for r in range(3):  
 O\_out[r] = 0  
 for c in range(3):  
 O\_out[r] += M[r][c] \* O\_in[c] # O\_out: Q4.41  
 # clip O\_out[r] from Q4.41 to Q4.29 after inner loop  
 # clamp result to range [0, 1]  
 O\_out[r] = max(0, min(1, O\_out[r])) # O\_out: UQ1.29

The related registers are shown in the tables below:

1. The CSC matrix

|  |  |  |  |
| --- | --- | --- | --- |
| Registers | Precision | Default Value | Notes |
| dpe\_hdr\_csc\_m\_00 | Q02.12 | 1 |  |
| dpe\_hdr\_csc\_m\_01 | Q02.12 | 0 |  |
| dpe\_hdr\_csc\_m\_02 | Q02.12 | 0 |  |
| dpe\_hdr\_csc\_m\_10 | Q02.12 | 0 |  |
| dpe\_hdr\_csc\_m\_11 | Q02.12 | 1 |  |
| dpe\_hdr\_csc\_m\_12 | Q02.12 | 0 |  |
| dpe\_hdr\_csc\_m\_20 | Q02.12 | 0 |  |
| dpe\_hdr\_csc\_m\_21 | Q02.12 | 0 |  |
| dpe\_hdr\_csc\_m\_22 | Q02.12 | 1 |  |

#### x.x.3.3 Gamma

Let denote one of the 3 channels of the input optical signal, is the result of the gamma operation via lookup tables and for optical and electronic signals with linear interpolation. After the gamma operation, we perform a linear stretch as in degamma with another set of parameters and , and then we can get the HDR output R’G’B’ data d\_out[3] with precision UQ10.0.

The procedure is shown as the pseudo code below:

# 1) perform gamma operation  
for k in range(3):  
 # perform gamma operation  
 # O: UQ1.29  
 # linterp():  
 # - sub\_term: UQ1.29 - UQ1.29 -> clip to UQ1.29  
 # - mul\_term: UQ27.1 \* UQ1.29 -> UQ28.30, clip to UQ10.4  
 # - y: UQ10.4 + UQ10.4 -> clip to UQ10.4  
 E[k] = linterp(O[k], L\_O, L\_E, L\_s\_o2e) # E: UQ10.4  
 # linear stretch  
 E[k] = a2\*E[k] + b2 # E: clip to Q10.4  
 # clamp result to range [0, 1023]  
 E[k] = max(0, min(1023, E[k])) # E: Q10.4  
 # perform rounding on clamped Q10.4 result to UQ10.0  
 E[k] = int(E[k] + 0.5) # E: UQ10.0

Data and related precisions for the LUT :

// L\_O (701 bits)  
double dpe\_hdr\_g\_lutx[33] = {  
 0.000000007450580596923828125 /\* UQ0.27 \*/,  
 0.000000014901161193847656250 /\* UQ0.27 \*/,  
 0.000000029802322387695312500 /\* UQ0.27 \*/,  
 0.000000059604644775390625000 /\* UQ0.27 \*/,  
 0.000000104308128356933593750 /\* UQ0.27 \*/,  
 0.000000186264514923095703125 /\* UQ0.27 \*/,  
 0.000000335276126861572265625 /\* UQ0.27 \*/,  
 0.00000059604644775390625 /\* UQ0.23 \*/,  
 0.00000107288360595703125 /\* UQ0.23 \*/,  
 0.00000190734863281250000 /\* UQ0.23 \*/,  
 0.00000345706939697265625 /\* UQ0.23 \*/,  
 0.00000619888305664062500 /\* UQ0.23 \*/,  
 0.00001120567321777343750 /\* UQ0.23 \*/,  
 0.00002002716064453125000 /\* UQ0.23 \*/,  
 0.00003588199615478515625 /\* UQ0.23 \*/,  
 0.00006449222564697265625 /\* UQ0.23 \*/,  
 0.0001163482666015625 /\* UQ0.19 \*/,  
 0.0002079010009765625 /\* UQ0.19 \*/,  
 0.0003719329833984375 /\* UQ0.19 \*/,  
 0.0006694793701171875 /\* UQ0.19 \*/,  
 0.0011997222900390625 /\* UQ0.19 \*/,  
 0.0021553039550781250 /\* UQ0.19 \*/,  
 0.0038661956787109375 /\* UQ0.19 \*/,  
 0.0069389343261718750 /\* UQ0.19 \*/,  
 0.0124511718750000 /\* UQ1.16 \*/,  
 0.0223388671875000 /\* UQ1.16 \*/,  
 0.0401000976562500 /\* UQ1.16 \*/,  
 0.0719757080078125 /\* UQ1.16 \*/,  
 0.1291503906250000 /\* UQ1.16 \*/,  
 0.2317810058593750 /\* UQ1.16 \*/,  
 0.4159545898437500 /\* UQ1.16 \*/,  
 0.7464752197265625 /\* UQ1.16 \*/,  
 1.0000000000000000 /\* UQ1.16 \*/  
};

The related registers are shown in the tables below:

1. Registers for the LUT of output electrical signal

|  |  |  |  |
| --- | --- | --- | --- |
| Registers | Precision | Default Value | Notes |
| dpe\_hdr\_g\_luty\_00 ~ dpe\_hdr\_g\_luty\_09 | UQ06.04 | 0 | L\_E[0] ~ L\_E[9] |
| dpe\_hdr\_g\_luty\_10 ~ dpe\_hdr\_g\_luty\_18 | UQ08.04 | 0 | L\_E[10] ~ L\_E[18] |
| dpe\_hdr\_g\_luty\_19 ~ dpe\_hdr\_g\_luty\_32 | UQ10.04 | 0 | L\_E[19] ~ L\_E[32] |

1. Registers for the LUT of the slope term for linear interpolation

|  |  |  |  |
| --- | --- | --- | --- |
| Registers | Precision | Default Value | Notes |
| dpe\_hdr\_g\_luts\_00 ~ dpe\_hdr\_g\_luts\_03 | UQ27.01 | 0 | L\_s\_o2e[0] ~ L\_s\_o2e[3] |
| dpe\_hdr\_g\_luts\_04 ~ dpe\_hdr\_g\_luts\_16 | UQ25.01 | 0 | L\_s\_o2e[4] ~ L\_s\_o2e[16] |
| dpe\_hdr\_g\_luts\_17 ~ dpe\_hdr\_g\_luts\_22 | UQ18.01 | 0 | L\_s\_o2e[17] ~ L\_s\_o2e[22] |
| dpe\_hdr\_g\_luts\_23 ~ dpe\_hdr\_g\_luts\_31 | UQ14.01 | 0 | L\_s\_o2e[23] ~ L\_s\_o2e[31] |

1. Other related registers

|  |  |  |  |
| --- | --- | --- | --- |
| Registers | Precision | Default Value | Notes |
| dpe\_hdr\_g\_stretch\_a | UQ14.12 | 1 |  |
| dpe\_hdr\_g\_stretch\_b | Q14.12 | 0 |  |

#### x.x.3.4 Other Registers

There are some additional registers such as the enable signals of various functionalities in the HDR module.

|  |  |  |  |
| --- | --- | --- | --- |
| Registers | Precision | Default Value | Notes |
| dpe\_hdr\_en | UQ01.00 | 0 | Enable signal for HDR module |
| dpe\_hdr\_tm\_sat\_en | UQ01.00 | 1 | Enable signal for saturation enhancement after tone mapping |