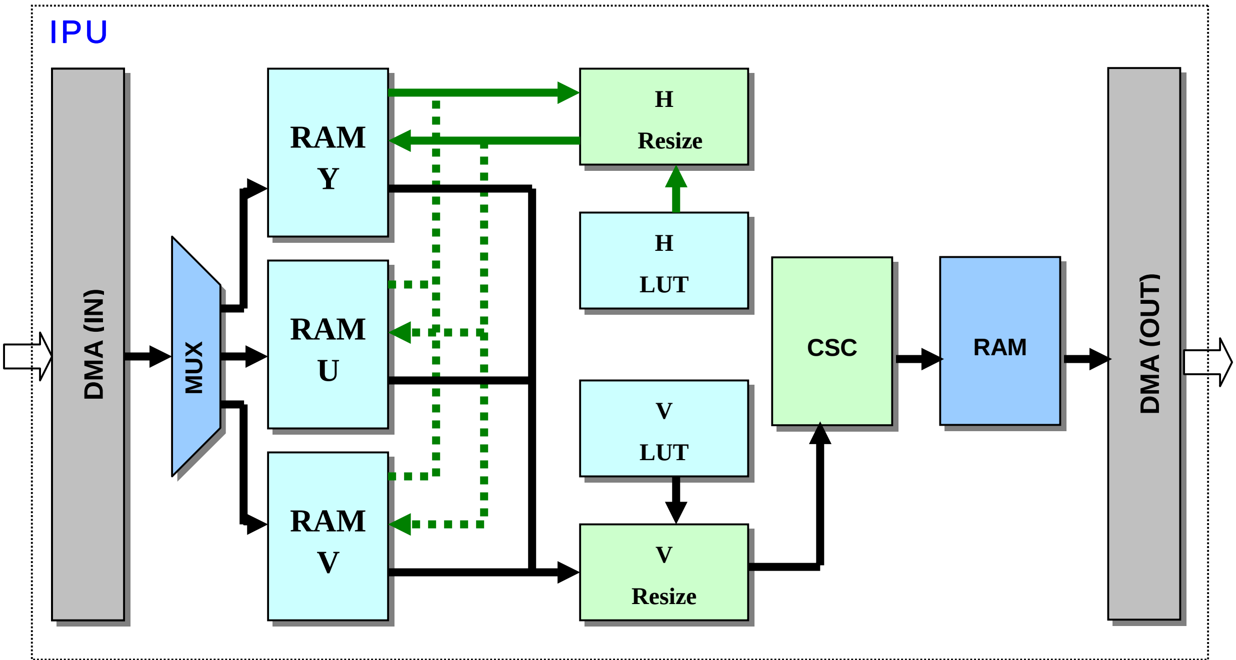
1 Overview

IPU (Image process unit) contains Resize and CSC (color space conversion), which is used for image post processing.

Features

- Input data: from external memory
- Input format: YUV /YCbCr (4:2:0, 4:2:2, 4:4:4, 4:1:1)
- Output format: RGB (565, 555, 888)
- Minimum input image size: 33x33
- Maximum input image size: 2047x2047
- Image resizing:
  -- Up scaling ratios up to 1:2 in fractional steps
  -- Down scaling ratios up to 20:1 in fractional steps

Block Diagram



2 Data flow

Input Data

Y, U, V (or Y, Cb, Cr; the following use YUV for convenience) data would be fetched from external memory by DMA burst read operation respectively.

Output Data

The data format after CSC could be RGB (565, 555, 888), and the data would be stored to the external memory by DMA burst write operation.

Resize Coefficients LUT

The resize coefficients look up table is preset by software according to specific format (see later chapter for detail). There are 2 tables support independent horizontal and vertical scaling. Each table has 20 entries that can accommodate up to 20 coefficients.

3 Register definition

IPU Control Register

IPU_CONTROL **0x0**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																							V_SCALE	H_SCALE					IPU_RST	FM_IRQ_EN	RSZ_EN	IPU_EN
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0

Bits	Name	Description	R/W
31:10	Reserved	Writing has no effect, read as zero.	R
9	V_SCALE	Vertical direction scale flag. 0: down scaling; 1: up scaling	RW
8	H_SCALE	Horizontal direction scale flag. 0: down scaling; 1: up scaling	RW
7:4	Reserved	Writing has no effect, read as zero.	R
3	IPU_RST	Reset IPU. Writing 1: reset IPU; 0: no effect. Read as zero	W
2	FM_IRQ_EN	Frame process finish interrupt enable. 1: enable; 0: disable	RW
1	RSZ_EN	Resize enable. 1: enable; 0: disable	RW
0	IPU_EN	IPU enable. 1: enable; 0: disable Once IPU enabled, IPU works until flag OUT_END in the IPU_STATUS is set value 1.	RW

NOTES:

Setting value 1 to IPU_RST will reset all software visible IPU registers immediately. It is not recommended that stopping IPU abruptly by clearing IPU_EN when IPU is running (IPU_EN=1, OUT_END=0), or writing value 1 to IPU_RST when DMA (in or out) is working (IPU_EN=1, OUT_END=0), otherwise, the result is unpredictable.

IPU Status Register

IPU_STATUS **0x4**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																																OUT_END
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0

Bits	Name	Description	R/W
31:1	Reserved	Writing has no effect, read as zero.	R
0	OUT_END	Output DMA termination flag. 1: finished; 0: not finished HW can only set value 1, SW can only clear it to value 0	R/W

NOTES:

If IPU_CONTROL.FM_IRQ_EN has been set 1, once OUT_END is set value 1 which denotes a frame's post process done, an low level active interrupt request will be issued until corresponding software handler clear OUT_END to value 0.

Data Format Register

D_FMT					0x8
Bit	31 … 18	17 16	15 … 3	2 1 0	
	(reserved)	OUT_FMT	(reserved)	IN_FMT	
RST	? ? ? ? ? ? ? ? ? ? ? ? ? ?	? ?	? ? ? ? ? ? ? ? ? ? ? ? ?	? ? ?	

Bits	Name	Description	R/W
31:18	Reserved	Writing has no effect, read as zero.	R
17:16	OUT_FMT	Indicates the destination data format: 00: RGB555 01: RGB565 10: RGB888 11: reserved	RW
15:3	Reserved	Writing has no effect, read as zero.	R
2:0	IN_FMT	Indicates the source data format: 000: YUV 4:2:0 001: YUV 4:2:2 010: YUV 4:4:4 011: YUV 4:1:1 100: YCbCr 4:2:0 101: YCbCr 4:2:2 110: YCbCr 4:4:4 111: YCbCr 4:1:1	RW

Input Y Data Address Register

Y_ADDR		0xc
Bit	31 … 0	
	Y_ADDR	
RST	? ?	

Bits	Name	Description	R/W
31:0	Y_ADDR	The source Y data buffer's start address	RW

Input U Data Address Register

U_ADDR **0x10**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	U_ADDR																															
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
31:0	U_ADDR	The source U data buffer's start address	RW

Input V Data Address Register

V_ADDR **0x14**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	V_ADDR																															
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
31:0	V_ADDR	The source V data buffer's start address	RW

Input Geometric Size Register

IN_FM_GS **0x18**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						IN_FM_W																IN_FM_H										
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
31:27	Reserved	Writing has no effect, read as zero.	R
26:16	IN_FM_W	The width of the input frame (unit: byte). Y data width is same as this value while U/V or Cb/Cr data width should do relatively zoom in according to the source data format.	RW
15:11	Reserved	Writing has no effect, read as zero.	R
10:0	IN_FM_H	The height of the input frame (unit: byte). Y data width is same as this value while U/V or Cb/Cr data width should do relatively zoom in according to the source data format.	RW

Input Y Data Line Stride Register

Y_STRIDE **0x1c**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																					Y_S											
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
31:12	Reserved	Writing has no effect, read as zero.	R
11:0	Y_S	The line stride of the source Y data in the external memory. (unit: byte)	RW

Input UV Data Line Stride Register

UV_STRIDE **0x20**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					U_S																V_S											
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
31:28	Reserved	Writing has no effect, read as zero.	R
27:16	U_S	The line stride of the source U data in the external memory. (unit: byte)	RW
15:12	Reserved	Writing has no effect, read as zero.	R
11:0	V_S	The line stride of the source V data in the external memory. (unit: byte)	RW

Output Frame Start Address Register

OUT_ ADDR **0x24**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	OUT_ ADDR																															
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
31:0	OUT_ ADDR	The output buffer's start address.	RW

Output Geometric Size Register

OUT_GS **0x28**

Bit	31 30	29 28 27 26 25 24 23 22 21 20 19 18 17 16	15 14	13 12 11 10 9 8 7 6 5 4 3 2 1 0
		OUT_FM_W		OUT_FM_H
RST	? ?	? ? ? ? ? ? ? ? ? ? ? ? ? ?	? ?	? ? ? ? ? ? ? ? ? ? ? ? ? ?

Bits	Name	Description	R/W
31:30	Reserved	Writing has no effect, read as zero.	R
29:16	OUT_FM_W	The width of the output destination frame (unit: byte).	RW
15:14	Reserved	Writing has no effect, read as zero.	R
13:0	OUT_FM_H	The height of the output destination frame (unit: byte).	RW

Output Data Line Stride Register

OUT_STRIDE **0x2c**

Bit	31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
		OUT_S
RST	? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ?	? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ?

Bits	Name	Description	R/W
31:16	Reserved	Writing has no effect, read as zero.	R
15:0	OUT_S	The line stride of the destination data buffer in the external memory. (unit: byte)	RW

Resize Coefficients Table Index Register

RSZ_COEF_INDEX **0x30**

Bit	31 30 29 28 27 26 25 24 23 22 21	20 19 18 17 16	15 14 13 12 11 10 9 8 7 6 5	4 3 2 1 0
		HE_IDX		VE_IDX
RST	? ? ? ? ? ? ? ? ? ? ?	? ? ? ? ?	? ? ? ? ? ? ? ? ? ? ?	? ? ? ? ?

Bits	Name	Description	R/W
31:21	Reserved	Writing has no effect, read as zero.	R
20:16	HE_IDX	Indicates the end address of the horizontal resize look up table.	RW
15:5	Reserved	Writing has no effect, read as zero.	R
4:0	VE_IDX	Indicates the end address of the vertical resize look up table.	RW

CSC C0 Coefficient Register

CSC_C0_COEF **0x34**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																					C0_COEF											
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
31:12	Reserved	Writing has no effect, read as zero.	R
11:0	C0_COEF	The C0 coefficient of the YUV/YCbCr to RGB conversion. C0_COEF = [C0 * 1024 + 0.5]	RW

Note:

R = C0*(Y – X0) + C1*(Cr-128)

G = C0*(Y – X0) – C2*(Cb-128) – C3*(Cr-128)

B = C0*(Y – X0) + C4*(Cb-128)

CSC C1 Coefficient Register

CSC_C1_COEF **0x38**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																					C1_COEF											
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
31:12	Reserved	Writing has no effect, read as zero.	R
11:0	C1_COEF	The C1 coefficient of the YUV/YCbCr to RGB conversion. C1_COEF = [C1 * 1024 + 0.5]	RW

Note:

R = C0*(Y – X0) + C1*(Cr-128)

G = C0*(Y – X0) – C2*(Cb-128) – C3*(Cr-128)

B = C0*(Y – X0) + C4*(Cb-128)

CSC C2 Coefficient Register

CSC_C2_COEF **0x3c**

Bit	31:12	11:0
	(reserved)	C2_COEF
RST	? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ?	? ? ? ? ? ? ? ? ? ? ? ?

Bits	Name	Description	R/W
31:12	Reserved	Writing has no effect, read as zero.	R
11:0	C2_COEF	The C2 coefficient of the YUV/YCbCr to RGB conversion. C2_COEF = [C2 * 1024 + 0.5]	RW

Note:

R = C0*(Y – X0) + C1*(Cr-128)

G = C0*(Y – X0) – C2*(Cb-128) – C3*(Cr-128)

B = C0*(Y – X0) + C4*(Cb-128)

CSC C3 Coefficient Register

CSC_C3_COEF **0x40**

Bit	31:12	11:0
	(reserved)	C3_COEF
RST	? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ?	? ? ? ? ? ? ? ? ? ? ? ?

Bits	Name	Description	R/W
31:12	Reserved	Writing has no effect, read as zero.	R
11:0	C3_COEF	The C3 coefficient of the YUV/YCbCr to RGB conversion. C3_COEF = [C3 * 1024 + 0.5]	RW

Note:

R = C0*(Y – X0) + C1*(Cr-128)

G = C0*(Y – X0) – C2*(Cb-128) – C3*(Cr-128)

B = C0*(Y – X0) + C4*(Cb-128)

CSC C4 Coefficient Register

CSC_C4_COEF **0x44**

Bit	31:12	11:0
		C4_COEF
RST	? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ?	? ? ? ? ? ? ? ? ? ? ? ?

Bits	Name	Description	R/W
31:12	Reserved	Writing has no effect, read as zero.	R
11:0	C4_COEF	The C4 coefficient of the YUV/YCbCr to RGB conversion. C4_COEF = [C4 * 1024 + 0.5]	RW

Note:

R = C0*(Y – X0) + C1*(Cr-128)

G = C0*(Y – X0) – C2*(Cb-128) – C3*(Cr-128)

B = C0*(Y – X0) + C4*(Cb-128)

Horizontal Resize Coefficients Look Up Table Register group

HRSZ_COEF_LUT **0x48 ~ 0x94**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																							W_COEF								IN_EN	OUT_EN
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
31:13	Reserved	Writing has no effect, read as zero.	R
9:2	W_COEF	Weighting coefficients, 8 bits length, that is to say the precision is 1/128. For up-scaling, $W_k = 1 - (kn/m - [kn/m])$, k = 0, 1, … m-1. For down-scaling, for (t=0, k=0; k < n; k++) { If ([(tn+1)/m] – k >=1) { W_k = 0;} else if ((tn+1)/m – k == 0) { W_k = 1; t++;} else { W_k = 1 – ((tn+1)/m – [tn/m]); t++;} } $W\_COEF_k = [128 * W_k]$ Here n stands for original pixel points, m stands for pixel points after resize. For example down-scaling 5:3, n = 5, m = 3. Moreover, m and n are prime, that is, for example 8:2 should be converted to 4:1. When IPU_CONTROL.RSZ_EN set as 1 and m:n = 1:1, all coefficients should be calculated as up-scale case.	RW
1	IN_EN	Tick for whether new pixel would be used. IN_EN = 0, no new pixel IN_EN = 1, one new pixel In down scale case, IN_EN always equals 1. In up scale case, For (i=0, k=0; k < m; k++) { If(i<= kn/m) { IN_EN_k = 1; i++;} else { IN_EN_k =0;} }	RW
0	OUT_EN	Tick for whether current interpolation would be output. OUT_EN = 0, current interpolation would not be output OUT_EN = 1, current interpolation would be output In up scale case, OUT_EN always equals 1. In down scale case, For (k=0, t=0; t < n; t++) { If([(kn+1)/m] – t >=1)	RW

		OUT_EN $_t$ = 0; else {OUT_EN $_t$ =1; k++;} }	

NOTES*:*

The coefficient number equals to max (m, n). HLUT (horizontal look up table) and VLUT (vertical look up table) are independent, so the two tables may have different coefficient number. Therefore,

RSZ_COEF_INDEX.VIDX = The coefficient number of VLUT – 1

RSZ_COEF_INDEX.HIDX = The coefficient number of HLUT – 1

Moreover, when m=1 for down-scaling, discard above formula and use following rules:

1. $W\_COEF_0 = 64$ ($W_0 = 0.5$), and $W\_COEF_{1 \sim n-1} = 0$
2. IN_EN always equals 1
3. $OUT\_EN_0 = 1$, and $OUT\_EN_{1 \sim n-1} = 0$

Following are two examples of setting LUT

Resize coefficients for 7:3

W	W_COEF	IN_EN	OUT_EN	Pixel1	Pixel2	OUT
2/3	85	1	1	P [0]	P [1]	P [0] * 2/3 + P [1] * 1/3
0	0	1	0	P [1]	P [2]	No new pixel out
1/3	42	1	1	P [2]	P [3]	P [2] * 1/3 + P [3] * 2/3
0	0	1	0	P [3]	P [4]	No new pixel out
0	0	1	0	P [4]	P [5]	No new pixel out
1	128	1	1	P [5]	P [6]	P [5] * 1 + P [6] * 0
0	0	1	0	P [6]	P [7]	No new pixel out

Resize coefficients for 3:5

W	W_COEF	IN_EN	OUT_EN	Pixel1	Pixel2	OUT
1	128	1	1	P [0]	P [1]	P [0] * 1 + P [1] * 0
2/5	51	0	1	P [0]	P [1]	P [0] * 2/5 + P [1] * 3/5
4/5	102	1	1	P [1]	P [2]	P [1] * 4/5 + P [2] * 1/5
1/5	25	0	1	P [1]	P [2]	P [1] * 1/5 + P [2] * 4/5
3/5	76	1	1	P [2]	P [3]	P [2] * 3/5 + P [3] * 2/5

Vertical Resize Coefficients Look Up Table Register group

VRSZ_COEF_LUT **0x98 ~ 0xe4**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																							W_COEF								IN_EN	OUT_EN
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Function descriptions are same as horizontal LUT.

Resized width and height calculation

For software, to preset correct value for register OUT_GS, please refer to following formula.
Set IW stand for original input frame width, IH stand for original input frame height, OW stand for new frame width after resize, OH stand for new frame height after resize.

In Up-scale case (n < m):

If [(IW - 1) * (m/n)] * (n/m) ==(IW-1) then
    OW = [(IW - 1) * (m/n)] + 1;
Else OW = [(IW - 1) * (m/n)] + 2;
If [(IH - 1) * (m/n)] * (n/m) == (IH-1) then
    OH = [(IH - 1) * (m/n)] + 1;
Else OH = [(IH - 1) * (m/n)] + 2;

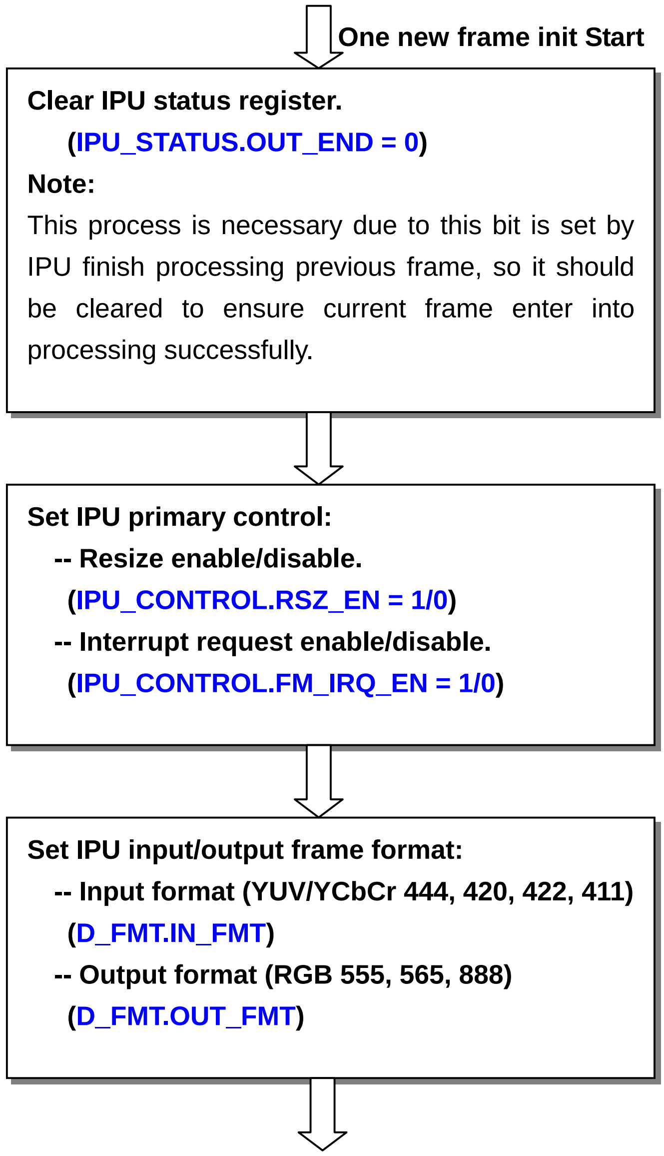
In Down-scale case (n>m):

If [(IW - 1) * (m/n)] * (n/m) ==(IW-1) then
    OW = [(IW - 1) * (m/n)];
Else OW = [(IW - 1) * (m/n)] + 1;
If [(IH - 1) * (m/n)] * (n/m) ==(IH-1) then
    OH = [(IH - 1) * (m/n)];
Else OH = [(IH - 1) * (m/n)] + 1;

For example:
A 36x46 frame with the horizontal resize ratio of 4:5 (up-scale) and vertical resize ratio of 7:6 (down-scale), by the expressions above we get its new size after resize from the following process.

For Width: [(36 - 1) * (5/4)] * (4/5) = 34.4 ≠ (36-1)
    So OW = [(36 - 1) * (5/4)] + 2 = 45
For Height: [(46 - 1) * (6/7)] * (7/6) = 44.33 ≠ (46 - 1)
    So OH = [(46 - 1) * (6/7)] + 1 = 39

IPU Initialization Flow



Set input frame size:

- **-- Input frame width (Eg: 288x188 frame) (IN_FM_GS.IN_FM_W = 288)**
- **-- Input frame height (Eg: 288x188 frame) (IN_FM_GS.IN_FM_H = 188)**
- **-- Y frame stride (Y_STRIDE.Y_S)**
- **-- U frame stride (UV_STRIDE.U_S)**
- **-- V frame stride (UV_STRIDE.V_S)**

Note: Frame width/height value should be restricted according to frame format (ensure it is a legal size). In 411 case the value should be multiple of 4. In 420/422 case the value should be multiple of 2. 444 case, the value can be any integer in the legal range (33 ~ 2047). Moreover, all the three stride values should be set to ensure frames' every line start address are word aliened, reference to attached Figure A-1.

Set input/output data start address:

- **-- Y frame start address (Y_ADDR.Y_ADDR)**
- **-- U frame start address (U_ADDR.U_ADDR)**
- **-- V frame start address (V_ADDR.V_ADDR)**
- **-- Output frame start address (OUT_ADDR.OUT_ADDR)**

Note: Y/U/V frame start address must be word aliened. Output frame start address can be half-word or word aliened except RGB888 format, which should be aliened by word.

Set output frame size:

- **-- Output frame width (OUT_GS.OUT_FM_W)**
- **-- Output frame height (OUT_GS.OUT_FM_H)**
- **-- Output frame stride (OUT_STRIDE.OUT_S)**

Note: All values above are united by byte, so the values filled in are the pixel numbers left shifted by 1 or 2 according to output format. For example: RGB888, each pixel takes 4 bytes, so the width value is pixel numbers * 4. Such as a RGB888 frame with size of 120x80 and stride of 124, their value should be filled as: OUT_GS.OUT_FM_W=120<<2; OUT_GS.OUT_FM_H=80<<2;OUT_STRIDE.OUT_S=124<<2

Set CSC coefficients:

CSC_C0_COEF = 0x400(YUV)/ 0x4A8(YCbCr)

CSC_C1_COEF = 0x59C(YUV)/0x662(YCbCr)

CSC_C2_COEF = 0x161(YUV)/0x341(YCbCr)

CSC_C3_COEF = 0x2DC(YUV)/0x190(YCbCr)

CSC_C4_COEF = 0x718(YUV)/0x812(YCbCr)

Note: More information reference to the following attached Table A-1.

Resize Enable?

No

Yes

Set Vertical direction scale flag:

(IPU_CONTROL.V_SCALE)

Set Horizontal direction scale flag:

(IPU_CONTROL.H_SCALE)

Note: For example, Vertical resize ratio is 4:5 and Horizontal resize ratio is 7:6, that is vertical up-scale and horizontal down-scale, so these two bits should be set as:

IPU_CONTROL.V_SCALE=1; IPU_CONTROL.H_SCALE=0

Set resize coefficients table index register

-- Vertical LUT (RSZ_COEF_INDEX.VE_IDX)

-- Horizontal LUT (RSZ_COEF_INDEX.HE_IDX)

Note: For example, Vertical resize ratio is 3:5 and Horizontal resize ratio is 7:2, according to the calculation expressions in LUT register group's description table, Vertical LUT has 5 coefficients and Horizontal LUT has 7 coefficients. So they should be set as:

RSZ_COEF_INDEX.VE_IDX=4 (VLUT index from 0~4)

RSZ_COEF_INDEX.HE_IDX=6 (HLUT index from 0~6)

Set Vertical direction Look-Up Table:

(VRSZ_COEF_LUT)

Set Horizontal direction Look-Up Table:

(HRSZ_COEF_LUT)

Note: Reference to calculation expressions in LUT register group's description table above.

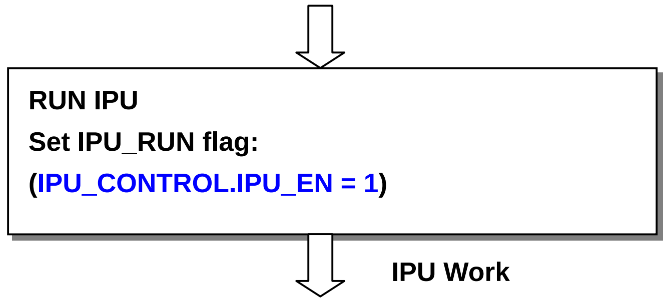


Table A-1. YUV/YCbCr to RGB CSC Equations

Input data	Matrix	CSC_COEF register values
YUV	R = C0*(Y – X0) + C1*(V-128) G = C0*(Y – X0) – C2*(U-128) – C3*(V-128) B = C0*(Y – X0) + C4*(U-128) X0: 0 C0: 1 C1: 1.4026 C2: 0.3444 C3: 0.7144 C4: 1.7730	CSC_C0_COEF = 0x400 CSC_C1_COEF= 0x59C CSC_C2_COEF = 0x161 CSC_C3_COEF = 0x2DC CSC_C4_COEF = 0x718
YCbCr	R = C0*(Y – X0) + C1*(Cr-128) G = C0*(Y – X0) – C2*(Cb-128) – C3*(Cr-128) B = C0*(Y – X0) + C4*(Cb-128) X0: 16 C0: 1.164 C1: 1.596 C2: 0.813 C3: 0.391 C4: 2.018	CSC_C0_COEF = 0x4A8 CSC_C1_COEF = 0x662 CSC_C2_COEF = 0x341 CSC_C3_COEF = 0x190 CSC_C4_COEF = 0x812

Table A-2. Output data package format

Format	Package
RGB888	Bit-31 24 23 16 15 8 7 0 EMPTY R7 R0 G7 G0 B7 B0
RGB555	15 14 10 9 5 4 0 Empty R7 R3 G7 G3 B7 B3
RGB565	15 11 10 5 4 0 R7 R3 G7 G2 B7 B3
Note: All R/G/B data are little-endian type; all the empty bits in the above figure are filled with 0.	

Figure A-1 Source Data storing format in external memory.

Example: YUV420 118x80 frame

	Bit-31			Bit-0	
	...				
Y Frame	Y (3,0)	Y (2,0)	Y (1,0)	Y (0,0)	← Y Frame start Address, Line 0 start
Y Frame	Y (7,0)	Y (6,0)	Y (5,0)	Y (4,0)	
Y Frame	...				← Y Frame Line 0, other entries of memory
Y Frame	Data out of frame		Y (117,0)	Y (116,0)	← Y Line 0 end -- Y (117,0), from here to Line 0 start stands for frame width
Y Frame	...				
Y Frame	Y (3,1)	Y (2,1)	Y (1,1)	Y (0,1)	← Y Frame Line 1 start, from Line 0 start to here stands for the Y stride value
Y Frame	...				
Y Frame	...		Y (117,79)	Y (116,79)	← Y Frame end -- Y (117,79)
	...				
U Frame	U (3,0)	U (2,0)	U (1,0)	U (0,0)	← U Frame start Address, Line 0 start
U Frame	...				
U Frame	Out data	U (58,0)	U (57,0)	U (56,0)	← U Frame Line 0 end -- U (58,0)
U Frame	...				
U Frame	Out data	U (58,39)	U (57,39)	U (56,39)	← U Frame end -- U (58,39)
	...				
V Frame	V (3,0)	V (2,0)	V (1,0)	V (0,0)	← V Frame start Address, Line 0 start
V Frame	...				
V Frame	Out data	V (58,39)	V (57,39)	V (56,39)	← V Frame end -- V (58,39)
	...				

Note:
1. Every line's start address should be word aliened.
2. All pixel data should be stored as little-endian type.
3. Destination data (RGB) storing format in external memory is similar with above figure, but RGB555 and RGB565 frame's every line start address can be half-word aliened (RGB888 frame still need word aliened).