

# 2024 Autumn ICLAB Midterm Project:

## Image Signal Processor (ISP) for Camera Auto Focus & Auto Exposure Algorithm

Jui-Huang Tsai

ericttsai.ee12@nycu.edu.tw

106902

### Abstract

The objective of this assignment is to design camera 3A algorithms using hardware description language, focusing primarily on autofocus—to calculate the optimal focal length within a given range—and auto-exposure algorithms, which adjust the image to a specified brightness and compute the average brightness of the adjusted image. Concurrently, the design utilizes the AXI4 transmission protocol to read and write image data from and to external DRAM.

To accommodate large volumes of image data within the design, the use of SRAM is encouraged in this assignment to reduce area overhead. This project is expected to enhance students' understanding and proficiency in hardware description language writing through the implementation of fundamental camera 3A algorithms. Furthermore, it aims to provide practical insight into the read and write operations of the AXI4 communication protocol and to strengthen skills and techniques related to SRAM operations.

Through this hands-on experience, students will gain a deeper comprehension of hardware design principles and their application in real-world imaging systems, while also developing expertise in efficient memory management and protocol implementation.

### Index Terms

Auto focus, Auto exposure, AXI4

1 0 0 0 0 0 0 0

### I. INTRODUCTION

In this assignment, we will primarily focus on designing Verilog implementations of camera 3A algorithms, specifically the auto focus and auto exposure algorithms. The following sections will briefly introduce the processes of autofocus and auto-exposure algorithms. Please use `tar xvf ~iclabTA01/Midterm_Project.tar` to extract the exercise file.

#### A. Auto Focus

The auto focus algorithm primarily works by simulating different focal lengths to identify the area with the maximum focus. The image size is fixed to  $3 \times 32 \times 32$ . The specific procedure is as follows:

I. Given 3 contrasts, from 0 to 2, cut the center of image to process. The coordinate of center of image for 4 contrasts are shown bellow (count from 0):

- 0:  $15 \leq x \leq 16, 15 \leq y \leq 16, 2 \times 2$ , total 4 points ✓
- 1:  $14 \leq x \leq 17, 14 \leq y \leq 17, 4 \times 4$ , total 16 points ✓
- 2:  $13 \leq x \leq 18, 13 \leq y \leq 18, 6 \times 6$ , total 36 points ✓
- 3:  $12 \leq x \leq 19, 12 \leq y \leq 19, 8 \times 8$ , total 64 points ✓

(14,13) (13,12)

Sum

Sum - next

$$\frac{192}{3} = b_4$$

$$b_4 + \frac{102}{38}$$

$$64 + 38 + 38 = 140$$

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA	AB	AC	AD	AE	AF	AG	AH	AI	AJ	AK	AL	AM	AN
1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31								
2	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63								
3	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96							
4	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160							
5	256	257	258	259	260	261	262	263	264	265	266	267	268	269	270	271	272	273	274	275	276	277	278	279	280	281	282	283	284	285	286	287								
6	512	513	514	515	516	517	518	519	520	521	522	523	524	525	526	527	528	529	530	531	532	533	534	535	536	537	538	539	540	541	542	543	544							
7	1024	1025	1026	1027	1028	1029	1030	1031	1032	1033	1034	1035	1036	1037	1038	1039	1040	1041	1042	1043	1044	1045	1046	1047	1048	1049	1050	1051	1052	1053	1054	1055	1056							
8	2048	2049	2050	2051	2052	2053	2054	2055	2056	2057	2058	2059	2060	2061	2062	2063	2064	2065	2066	2067	2068	2069	2070	2071	2072	2073	2074	2075	2076	2077	2078	2079	2080	2081	2082					
9	4096	4097	4098	4099	4100	4101	4102	4103	4104	4105	4106	4107	4108	4109	4110	4111	4112	4113	4114	4115	4116	4117	4118	4119	4120	4121	4122	4123	4124	4125	4126	4127	4128	4129						
10	8192	8193	8194	8195	8196	8197	8198	8199	8200	8201	8202	8203	8204	8205	8206	8207	8208	8209	8210	8211	8212	8213	8214	8215	8216	8217	8218	8219	8220	8221	8222	8223	8224							
11	16384	16385	16386	16387	16388	16389	16390	16391	16392	16393	16394	16395	16396	16397	16398	16399	16400	16401	16402	16403	16404	16405	16406	16407	16408	16409	16410	16411	16412	16413	16414	16415	16416							
12	32768	32769	32770	32771	32772	32773	32774	32775	32776	32777	32778	32779	32780	32781	32782	32783	32784	32785	32786	32787	32788	32789	32790	32791	32792	32793	32794	32795	32796	32797	32798	32799								
13	65536	65537	65538	65539	65540	65541	65542	65543	65544	65545	65546	65547	65548	65549	65550	65551	65552	65553	65554	65555	65556	65557	65558	65559	65560	65561	65562	65563	65564	65565	65566	65567								
14	131072	131073	131074	131075	131076	131077	131078	131079	131080	131081	131082	131083	131084	131085	131086	131087	131088	131089	131090	131091	131092	131093	131094	131095	131096	131097	131098	131099	131100	131101	131102	131103	131104							
15	262144	262145	262146	262147	262148	262149	262150	262151	262152	262153	262154	262155	262156	262157	262158	262159	262160	262161	262162	262163	262164	262165	262166	262167	262168	262169	262170	262171	262172	262173	262174	262175	262176							
16	524288	524289	524290	524291	524292	524293	524294	524295	524296	524297	524298	524299	524300	524301	524302	524303	524304	524305	524306	524307	524308	524309	524310	524311	524312	524313	524314	524315	524316	524317	524318	524319								
17	1048576	1048577	1048578	1048579	1048580	1048581	1048582	1048583	1048584	1048585	1048586	1048587	1048588	1048589	1048590	1048591	1048592	1048593	1048594	1048595	1048596	1048597	1048598	1048599	1048600	1048601	1048602	1048603	1048604	1048605	1048606									
18	2097152	2097153	2097154	2097155	2097156	2097157	2097158	2097159	2097160	2097161	2097162	2097163	2097164	2097165	2097166	2097167	2097168	2097169	2097170	2097171	2097172	2097173	2097174	2097175	2097176	2097177	2097178	2097179	2097180	2097181	2097182	2097183								
19	4194304	4194305	4194306	4194307	4194308	4194309	4194310	4194311	4194312	4194313	4194314	4194315	4194316	4194317	4194318	4194319	4194320	4194321	4194322	4194323	4194324	4194325	4194326	4194327	4194328	4194329	4194330	4194331	4194332	4194333	4194334	4194335								
20	8388608	8388609	8388610	8388611	8388612	8388613	8388614	8388615	8388616	8388617	8388618	8388619	8388620	8388621	8388622	8388623	8388624	8388625	8388626	8388627	8388628	8388629	8388630	8388631	8388632	8388633	8388634	8388635	8388636	8388637	8388638	8388639								
21	1677728	1677729	1677730	1677731	1677732	1677733	1677734	1677735	1677736	1677737	1677738	1677739	1677740	1677741	1677742	1677743	1677744	1677745	1677746	1677747	1677748	1677749	1677750	1677751	1677752	1677753	1677754	1677755	1677756	1677757	1677758	1677759								
22	3355456	3355457	3355458	3355459	3355460	3355461	3355462	3355463	3355464	3355465	3355466	3355467	3355468	3355469	3355470	3355471	3355472	3355473	3355474	3355475	3355476	3355477	3355478	3355479	3355480	3355481	3355482	3355483	3355484	3355485	3355486	3355487	3355488	3355489						
23	6710912	6710913	6710914	6710915	6710916	6710917	6710918	6710919	6710920	6710921	6710922	6710923	6710924	6710925	6710926	6710927	6710928	6710929	6710930	6710931	6710932	6710933	6710934	6710935	6710936	6710937	6710938	6710939	6710940	6710941	6710942	6710943	6710944							
24	1342112	1342113	1342114	1342115	1342116	1342117	1342118	1342119	1342120	1342121	1342122	1342123	1342124	1342125	1342126	1342127	1342128	1342129	1342130	1342131	1342132	1342133	1342134	1342135	1342136	1342137	1342138	1342139	1342140	1342141	1342142	1342143	1342144							
25	2684224	2684225	2684226	2684227	2684228	2684229	2684230	2684231	2684232	2684233	2684234	2684235	2684236	2684237	2684238	2684239	2684240	2684241	2684242	2684243	2684244	2684245	2684246	2684247	2684248	2684249	2684250	2684251	2684252	2684253	2684254	2684255	2684256	2684257	2684258	2684259				
26	5168448	5168449	5168450	5168451	5168452	5168453	5168454	5168455	5168456	5168457	5168458	5168459	5168460	5168461	5168462	5168463	5168464	5168465	5168466	5168467	5168468	5168469	5168470	5168471	5168472	5168473	5168474	5168475	5168476	5168477	5168478	5168479	5168480	5168481	5168482	5168483	5168484			
27	1024096	1024097	1024098	1024099	1024100	1024101	1024102	1024103	1024104	1024105	1024106	1024																												

8 × 4 × 2<sup>2</sup>  
(2 × 3)  
→ 15

III. Convert the selection area to grayscale. This can be represented by the following formula:

$$G_{center}(i, j) = I_{center}(i, j, 0) \cdot 0.25 + I_{center}(i, j, 1) \cdot 0.5 + I_{center}(i, j, 2) \cdot 0.25$$

The definition of each symbol:

$$[I_{center} \in \mathbb{N}^{3 \times H \times W}, 0 \leq I \leq 255], \quad [G_{center} \in \mathbb{N}^{1 \times H \times W}, 0 \leq G \leq 255],$$

IV. Calculate the difference between each element on the x-axis and y-axis separately, and take the absolute value. Sum up all the calculated differences.

$$D_{contrast\_h*w} = \sum_{j=0}^{H-1} \sum_{i=0}^{W-2} |G_{center}(i+1, j) - G_{center}(i, j)| + \sum_{i=0}^{W-1} \sum_{j=0}^{H-2} |G_{center}(i, j+1) - G_{center}(i, j)|$$

$\bar{j} = 0 \dots 1 \quad \bar{i} = 0 \dots D_{contrast\_h*w} \in \mathbb{N}^3, h = w, w = 2, 4, 6 \quad \bar{j} = 0 \dots 1$

V. Divide the summed value by the processed image's area, and record this value.

$$D_{contrast\_h*w} = D_{contrast\_h*w} / (H^2)$$

VI. Repeat steps III. to V. for four contrast. Find the maximum contrast value, which is the optimal solution. If there are two or above value are same, output the smaller one's contrast value. E.g. if contrast value 0 and 2 both's solution are same but larger than 1's, please output 0.

$$\max\_contrast = \arg \max(D_{contrast\_h*w})$$

The python and numpy-like pseudo code for color auto focus function is shown below.

---

**Algorithm 1** Python and Numpy-like pseudo code for function Color Auto Focus.

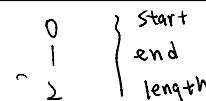
---

```

1 def color_auto_focus(image):
2     """
3         Process auto-focus algorithm for color images
4         :param image: 32x32x3 color image, numpy array
5         :return: Index of the best "focus"
6     """
7     def calculate_contrast(img):
8         # Convert to grayscale using dot product
9         gray = np.dot(img[:, :, 3], [0.25, 0.5, 0.25])
10        gray = np.array(gray)
11
12        # Calculate differences in 'gray'
13        dx = np.diff(gray, axis=1) # horizontal diff
14        dy = np.diff(gray, axis=0) # vertical diff
15
16        return int(np.sum(np.abs(dx)) + np.sum(np.abs(dy))) / (img.size[0]*img.size[1])
17
18    contrasts = []
19    center_point = image.shape[0] // 2
20
21    for i in range(0, 3): # Simulate different "focus", iterate over center area
22        # Extract different sized centers
23        center = image[center_point-1-i:center_point+1+i,
24                      center_point-1-i:center_point+1+i, :]
25        contrasts.append(calculate_contrast(center))
26
27    # Find the maximum value's index in the contrasts list
28    return np.argmax(contrasts)

```

---



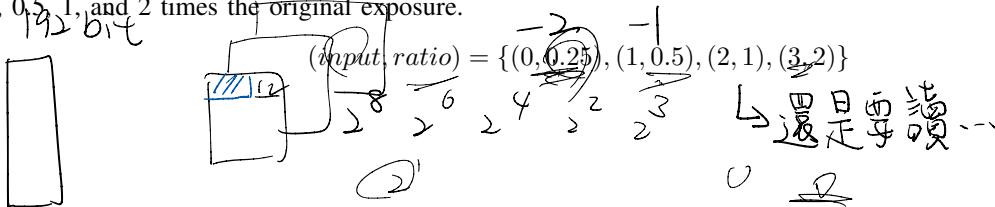
### B. Auto Exposure

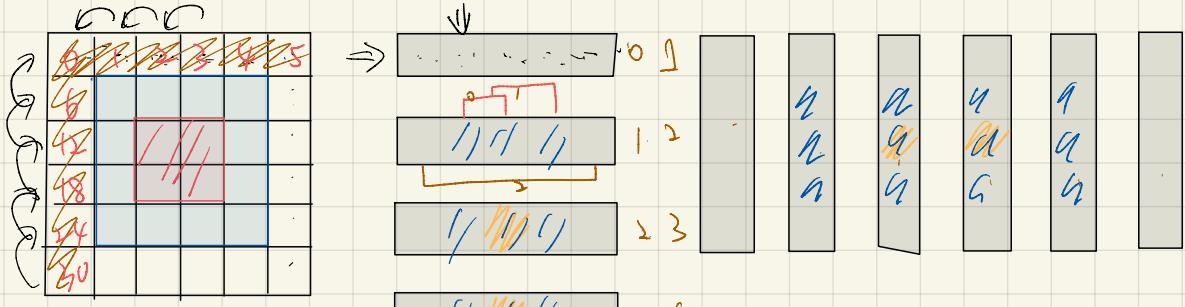
The purpose of this algorithm is to adjust the exposure ratio of a given image, which size is fixed to  $3 \times 32 \times 32$ . RGB 000

$$0 \leq i, j \leq 31$$

$-8 \sim 7$

I. First, input the entire image and the desired exposure adjustment ratio. There are 4 options, from 0 to 3, corresponding to 0.25, 0.5, 1, and 2 times the original exposure.





↑  
右下來後用轉  
的

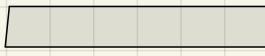
8 9 10 11 12

gray sum

↑ L<sub>18</sub>-組

diff-result

diff<sub>0</sub>



L<sub>18</sub>-組

focus 2 focus 3 focus 4

Cnt 168

168: 174
169
170
171
172
173
174
175
176
177
178
179

AF: { not computed → make in ratio mode = 1 and write  
computed → output

AE: { not computed → compute  
computed { ratio ≠ 1 → compute  
ratio = 1 → output

shift 0 ] shift input from DRAM  
shift 15 ]

0 0  
0 0  
1 1  
1 1

compute when: ① first time

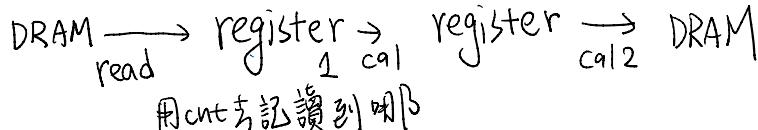
② num\_of\_div greater than 8

第一次算: 不要更新 num\_of\_div, 直接當 8 就好

接下來遇到: 如果是 8 而且是乘法 ⇒ 重算, 要 clamp

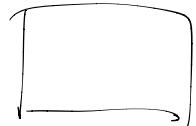
除法照常更新, 除到零直接立 flag

AF: 如果差值 <= 1; 遇到該圖片重算



- II. Adjust the original image according to the chosen exposure ratio and save the adjusted picture. If any adjusted value exceeds the maximum (255), set that value to 255.

$$I_{i,j} = \begin{cases} I_{i,j} * \text{ratio} & \text{for } 0 \leq (I_{i,j} * \text{ratio}) < 256 \\ 255 & \text{otherwise (clamp)} \end{cases}$$



- III. Finally, calculate the average value of the adjusted image and return this average. The detailed method is as follows:

- Multiply each element in the R and B channels by 0.25
- Multiply each element in the G channel by 0.5 因為channel為0→直接不拿該channel的資料
- Sum all these values and divide by the original image size (1024)
- This result is the average value.

$$\text{Avg} = \frac{\sum_{i=0}^{32} \sum_{j=0}^{32} [I(i,j,0) \cdot 0.25 + I(i,j,1) \cdot 0.5 + I(i,j,2) \cdot 0.25]}{32 \cdot 32}$$

$$32 \times 32 = 1024$$

- IV. Due to possible precision errors in the calculations, the answer allows a margin of error of  $\pm 1$ . ( $|\text{golden} - \text{ans}| < 2$ )

The python and numpy-like pseudo code for color auto exposure function is shown below.

#### Algorithm 2 Python and Numpy-like pseudo code for function Color Auto Exposure.

```

1 def color_auto_exposure(image, ratio):
2     """
3         Process auto-exposure algorithm for color images
4         :param image: 32x32x3 color image, numpy array
5         :param ratio: Adjustment ratio
6         :return: Mean value of adjusted image
7     """
8
9     # Adjust each color channel
10    # Use np.clip to limit values between 0-255
11    adjusted_image = np.clip(image * ratio, 0, 255).astype(np.uint8)
12
13    # return mean value of adjusted image
14    return np.mean(np.dot(adjusted_image[...,:3], [0.25, 0.5, 0.25]))

```

$$\begin{array}{r} 64 \\ 3 \\ \hline 192 \end{array}$$

## II. DESIGN

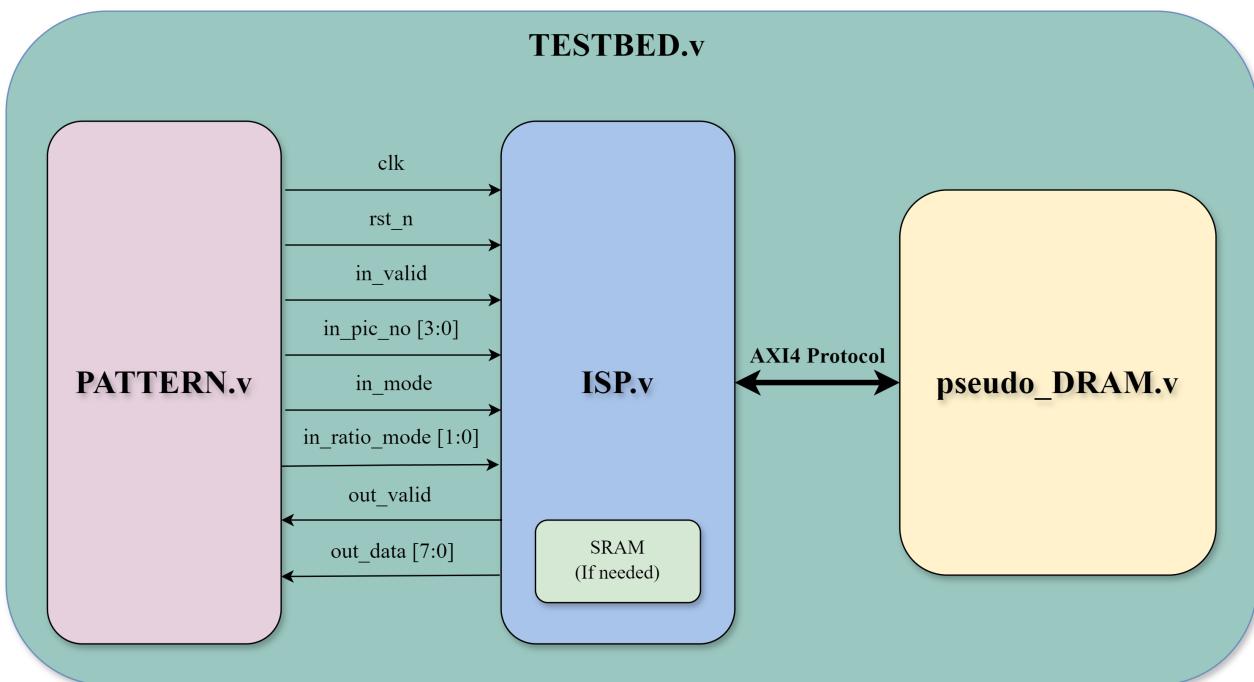


Fig. 2: ISP System for midterm project.

TABLE I: Signal Description between PATTERN.v and ISP.v

Signal	Width	From	Description
clk	1	PATTERN.v	System clock.
rst_n	1	PATTERN.v	Reset all system.
in_valid	1	PATTERN.v	When in_valid=1, means the signal in_mode, in_pic_no and in_ratio_mode signals are valid. Will be high for only 1 cycle.
in_pic_no	4	PATTERN.v	Select which picture from the DRAM. Only valid when in_valid=1
in_mode	1	PATTERN.v	Select algorithm type. 0 for Auto Focus, 1 for Auto Exposure. Only valid when in_valid=1
in_ratio_mode	1	PATTERN.v	Select the ratio for auto exposure. 0 means 0.25, 1 means 0.5, 2 means 1, 3 means 2. Only valid when in_valid=1 and in_mode=1.
out_valid	1	ISP.v	When out_valid=1, means the our_data is valid. Note that cannot be overlapped with in_valid.
out_data	8	ISP.v	(Return the best contrasts when algorithm mode is auto focus, means value for the picture adjusted after auto exposure.)

TABLE II: AXI Signal for Write Address Channel

Signal	Width	From	Description
awid_s_inf	4	ISP.v	Write address ID. In this project, we only use this to recognize master, reordering method is not supported.
awaddr_s_inf	32	ISP.v	Write address. Address start from 32'h10000, and each pic location is 32'h10000 + 3072*pic_no.
awsizes_s_inf	3	ISP.v	Burst size. Only support 3'b100 in this exercise.
awburst_s_inf	2	ISP.v	Burst type. Only INCR(2'b01) support in this Project.
awlen_s_inf	8	ISP.v	Burst length. The burst length gives the exact number of transfers in a burst.
awvalid_s_inf	1	ISP.v	Write address valid. 1 = address and control information available; 0 = address and control information not available.
awready_s_inf	1	pseudo_DRAM.v	Write address ready. 1 = slave ready; 0 = slave not ready.

The system is composed of three parts: PATTERN.v, ISP.v, and pseudo\_DRAM.v. PATTERN.v is responsible for sending instructions and verifying the correctness of the answers. ISP.v implements the algorithm for auto focus and auto exposure, and AXI4 for reading and writing data with pseudo\_DRAM.v. If large amounts of data need to be stored, SRAM can be added within ISP.v to facilitate temporary storage of large data volumes. pseudo\_DRAM.v stores all image data, which ISP.v needs to read and write using AXI4, the AXI4 function inside pseudo\_DRAM has been implement by TA. The connectivity of the ISP system can refer to the fig. 2. The channel length and description between ISP.v and PATTERN.v is shown in table 1.

For write data into pseudo\_DRAM.v, you need to comply with the AXI4 protocol, which can be refereed from the AXI slide. Note that the clk and rst\_n signal should be sent into pseudo\_DRAM.v from ISP.v. In table II, it is related to write address control. Note that the head of image's memory site is started from 32'h10000 + (3\*32\*32)\*pic\_no. In the table III is about the write data channel controlling, and the table IV is for write response channel controlling. For reading data from pseudo\_DRAM.v, please refer to tabel V, VI.

The DRAM content data is shown in fig. 3 note that each location in DRAM can store 8 bits data. The first picture was started at 32'h10000, if the design need to get other picture, the formula for location calculation is 32'h10000 + pic\_no \* 3 \* 32 \* 32. If you want to get whole picture in a round, you can set burst\_len to 192, this is calculated from

$$\frac{32 * 32 * 3 * 8bits}{128bits}$$

For in\_mode is set to 0 when in\_invalid is high, your design should execute auto focus algorithm. First, you should get data from the DRAM with corresponding picture through the AXI4 protocol and save into your design. After that, the design can start conducting auto focus algorithm as mentioned in the chapter I-A. After calculating, you should return the best contrast to the PATTERN.v through out\_data when out\_valid is high.

If the mode is auto exposure algorithm now, same as the previous one, you should get the data from the DRAM and refer to the chapter I-A to transform the image to the desired brightness. You need to write back the image after adjusting to DRAM, or you may fail if encounter the same picture in next coming testing. You also need to return the mean brightness of the adjusted picture through out\_data when out\_valid is high to PATTERN.v.

For the PATTERN.v, you can use the code bellowing to mimic the pseudo\_DRAM, which will lead easier for your pattern to check the result from your design.

```

1 parameter DRAM_p_r = ".../00_TESTBED/DRAM/dram.dat";
2 reg [7:0] DRAM_r [0:196607];
3
4 initial begin
5 $readmemh(DRAM_p_r, DRAM_r);
6 end

```

Listing 1: example code for PATTERN.v to mimic pseudo\_DRAM.

TABLE III: AXI Signal for Write Data Channel

Signal	Width	From	Description
wdata_s_inf	128	ISP.v	Write data.
wlast_s_inf	1	ISP.v	Write last. This signal indicates the last transfer in a write burst.
wvalid_s_inf	1	ISP.v	Write valid. 1 = write data and strobes available; 0 = write data and strobes not available.
wready_s_inf	1	pseudo_DRAM.v	Write ready. 1 = slave ready; 0 = slave not ready.

TABLE IV: AXI Signal for Write Response Channel

Signal	Width	From	Description
bid_s_inf	4	pseudo_DRAM.v	Response ID. The BID value must match the AWID value.
bresp_s_inf	2	pseudo_DRAM.v	Write response. In this project we only issue OKAY(2'b00)
bvalid_s_inf	1	pseudo_DRAM.v	Write response valid. 1 = write response available; 0 = write response not available.
bready_s_inf	1	ISP.v	Response ready. 1 = master ready; 0 = master not ready.

As known, there are some delay caused by other issue in the real DRAM. Our DRAM has simulated this situation too. The LAT\_MAX is the maximum latency that may happened in read data mode; hence, the LAT\_min is the minimum latency in read data mode. During the demo, TA will use the 'PERF' mode list in the bellowing verilog code. The parameter 'd\_DRAM\_p\_r' is the location for the pseudo\_DRAM to read your data. 'd\_DRAM\_R\_LAT' is the fixed latency for read data, available when 'd\_RANDOM\_R\_LAT' is 0. 'd\_DRAM\_W\_LAT' is the latency for writing data mode. 'd\_RANDOM\_R\_LAT' can decide the latency for read mode is fixed or random.

```

1 `ifdef SAMPLE
2 `define LAT_MAX 10
3 `define LAT_MIN 1
4 `endif
5 `ifdef FUNC
6 `define LAT_MAX 10
7 `define LAT_MIN 1
8 `endif
9 `ifdef PERF
10 `define LAT_MAX 500
11 `define LAT_MIN 300
12 `endif
13
14 // Modify your "d_DRAM_p_r" in this directory path to initialized DRAM Value
15 // Modify d_DRAM_R_LAT          for Initial Read Data Latency,
16 //           d_DRAM_W_LAT        for Initial Write Data Latency
17 //           d_RANDOM_R_LAT      for 1: Random Read Data Latency 0: DRAM_R_LAT Read Data
18   Latency
19 `define d_DRAM_p_r "../../00_TESTBED/DRAM/dram.dat"
20 `define d_DRAM_R_LAT 1
21 `define d_DRAM_W_LAT 1
22 `define d_RANDOM_R_LAT 1

```

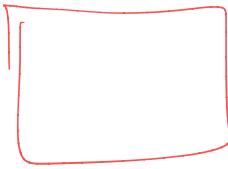
TABLE V: AXI Signal for Read Address Channel

Signal	Width	From	Description
arid_s_inf	4	ISP.v	Read address ID. In this project, we only use this to recognize master, reordering method is not supported
araddr_s_inf	32	ISP.v	Read address. Each image location is started from $32'h10000 + 3072 * \text{pic\_no}$ .
arlen_s_inf	8	ISP.v	Burst length.
arsize_s_inf	3	ISP.v	Burst size. Only support 3b'100.
arburst_s_inf	2	ISP.v	Burst type. Only INCR: 2b'01 support in this Project.
arvalid_s_inf	1	ISP.v	Read address valid. 1 = address and control information valid; 0 = address and control information not valid.
arready_s_inf	1	pseudo_DRAM.v	Read address ready. 1 = slave ready; 0 = slave not ready.

TABLE VI: AXI Signal for Read Data Channel

Signal	Width	From	Description
rid_s_inf	4	pseudo_DRAM.v	Read ID tag. The RID value is generated by the slave and must match the ARID value.
rdata_s_inf	128	pseudo_DRAM.v	Read data.
rresp_s_inf	2	pseudo_DRAM.v	Read response. In this project we only issue OKAY (2'b00).
rlast_s_inf	1	pseudo_DRAM.v	Read last.
rvalid_s_inf	1	pseudo_DRAM.v	Read valid. 1 = read data available; 0 = read data not available.
rready_s_inf	1	ISP.v	Read ready. 1= master ready; 0 = master not ready.

32



Listing 2: The verilog code in pseudo\_DRAM about setting up latency parameter.

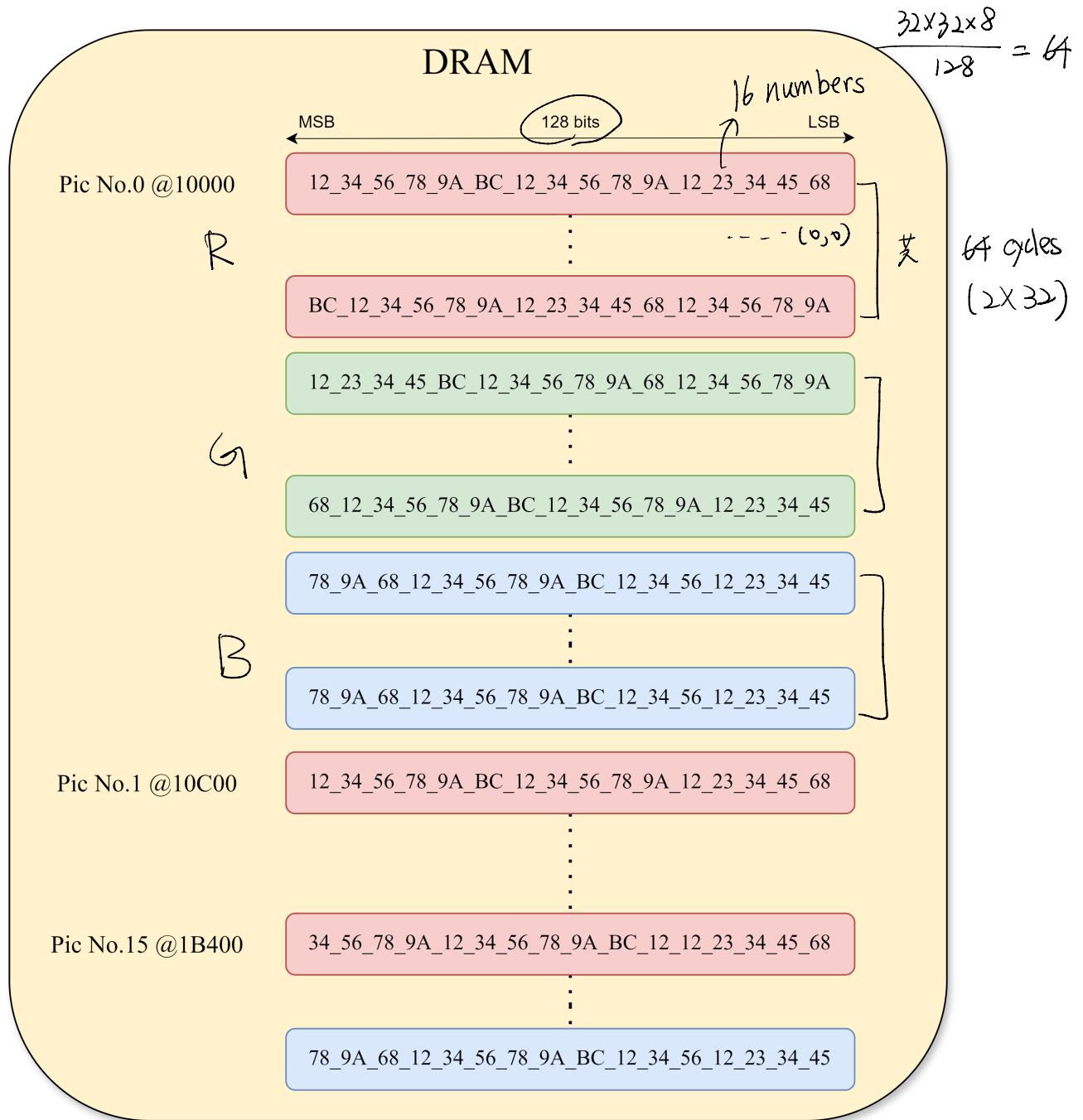
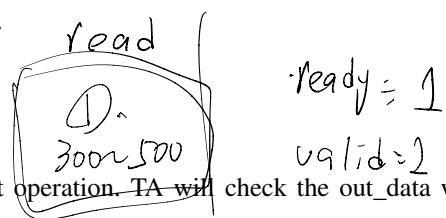


Fig. 3: DRAM content illustration.



## III. DESIGN RULES

- I. After `rst_n`, all output (including AXI) signals should be set to 0.
- II. You need to raise `out_valid` only when your design has done current input operation. TA will check the `out_data` when the `out_valid` is high.
- III. `out_valid` signal cannot overlap with `in_valid`.
- IV. The next `in_valid` signal will be high in next 2 cycle after `out_valid` is high.
- V. The system will NOT check the data stored in DRAM.

- VI. The AXI write mode should be completed when the writing action is done, which means the w\_last should be high when the last data is writing back to the DRAM. Otherwise, it will fail when next time attempt to write data back to the DRAM.
- VII. We encourage you use the SRAM, or the area may exceed the limitation. If you decide to use, remember to add the relative path of your DRAM to the filelist.f, and add the correspond .db to the 02\_SYN/.synopsys\_dc.setup.
- VIII. TA has provided some python file for this lab under 00\_TESTBED folder:
- pic\_generation.py: It can generate dram.dat and dram.json file. dram.dat can be used by PATTERN.v and pseudo\_DRAM.v. dram.json has the same data with dram.dat, but it is more readability for helping user to debugging.
  - cam3A.py: It provide the algorithm for auto focus and auto exposure in python manner, you can refer the algorithm from this file.
  - check\_pattern.py: It can read the dram.dat and calculate the data by python, you can use this program to help u to debug.
- IX. TA will use dram1.dat, dram2.dat and dram3.dat provide to you inside the 00\_TESTBED/DRAM folder to demo your design. However, TA will also use some hidden cases to verify your design too. Fortunately, TA will only use the .dat file randomly produce from the pic\_generation.py to test.

#### IV. SPECIFICATIONS

##### A. Top module

- I. Top module name: ISP (filename: ISP.v)
- II. Your design should trigger at positive edge clock, and the pattern will trigger and receive input/output signal at negative edge clock.
- III. Remember to add the SRAM .v file to filelist.f if you have utilize SRAM in your design.

##### B. Reset

- I. It is **asynchronous reset** and **active-low** architecture. If you use synchronous reset (reset after clock starting) in your design, you may fail to reset signals.
- II. The reset signal (rst\_n) would be given only once at the beginning of simulation. All output signals (including AXI signals) should be reset after the reset signal is asserted.
- III. All signal should be reset to 0.

##### C. Design Constraints

- I. The maximum clock period is 20 ns.
- II. Your latency should be less than 20000 cycles for each operation.
- III. All outputs are synchronized at clock rising edge.
- IV. You CANNOT use any designware IP in this lab.

##### D. Synthesis

- I. The input delay and the output delay are  $\frac{\text{clock\_period}}{2}$ .
- II. The output load should be set to 0.05.
- III. The synthesis result of data type cannot include any **LATCH**.
- IV. Synthesis time should less than 1 hour.
- V. The total area (ISP.v and your SRAM) should be less than 750,000.
- VI. Remember to add .db file to the .synopsys\_dc.setup file if you using SRAM.

##### E. Gate Level Simulation

- I. The gate-level simulation cannot include any timing violations without the notimingcheck command.

##### F. Supplement

- I. Don't use any wire/reg/submodule/parameter name called \*error\*, \*congratulation\*, \*latch\* or \*fail\* otherwise you will fail the lab. Note: \* means any char in front of or behind the word. e.g: error\_note is forbidden.
- II. Don't write Chinese or other language comments in the file you sent.
- III. Verilog command for design compiler optimizations are forbidden.
- IV. Any error messages during synthesize and simulation, regardless of the result will lead to failure in this lab.
- V. Any form of display or printing information in Verilog design is forbidden. You may use this methodology during debugging, but the file you turn in should not contain any coding that is not synthesized.
- VI. Plagiarism is strictly prohibited!

## V. GRADING

- 70%: Functionality
- 30%: (ISP Area, including SRAM) \* (latency \* cycle\_time)<sup>2</sup>

## VI. NOTES

I. Submit your files through 09\_SUBMIT in the midterm project folder

- 1st\_demo deadline: **2024/11/18 (Mon) 12:00:00 (noon)**
- 2nd\_demo deadline: **2024/11/20 (Wed) 12:00:00 (noon)**

II. The file needs to be submitted in midterm project:

- ISP.v
- filelist.f
- .synopsys\_dc.setup
- 04\_MEM folder

The cycle time of your design will be submitted by '00\_tar xx.x'.

## VII. CONCLUSION

This midterm project presents a comprehensive exercise in designing and implementing crucial components of an Image Signal Processor (ISP) for camera systems. By focusing on the auto-focus and auto-exposure algorithms, students are challenged to apply their knowledge of hardware description languages in a practical context that mirrors real-world imaging applications. The integration of the AXI4 transmission protocol for DRAM interaction adds an extra layer of complexity, simulating the constraints and considerations of actual system designs.

This project not only reinforces fundamental concepts in digital design but also introduces students to advanced topics in image processing and system-on-chip architecture. The emphasis on efficient memory management through SRAM utilization encourages students to consider hardware resource optimization, a critical skill in VLSI design. Furthermore, the strict design rules and synthesis constraints provide valuable experience in meeting industry-standard requirements for timing, area, and power efficiency.

By engaging with this multifaceted assignment, students will gain a deeper understanding of the intricate relationship between algorithms, hardware design, and system-level considerations. This holistic approach to learning will undoubtedly prepare them for the challenges they may face in their future careers in the field of electronic engineering and computer architecture.

## VIII. REFERENCE WAVEFORM

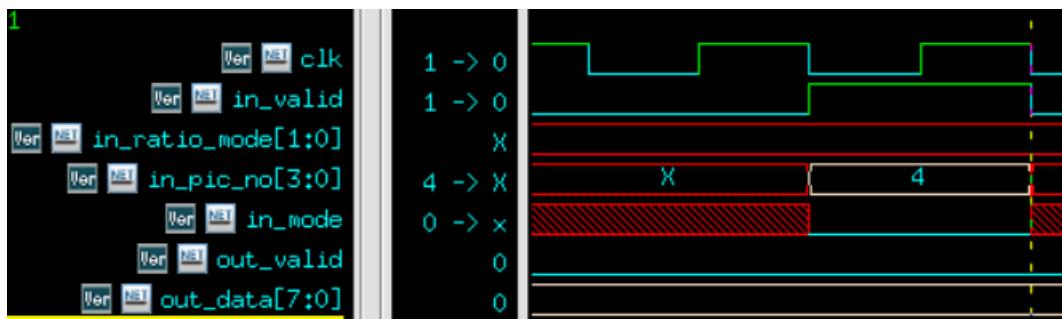


Fig. 4: Waveform for input mode 0: Auto Focus.

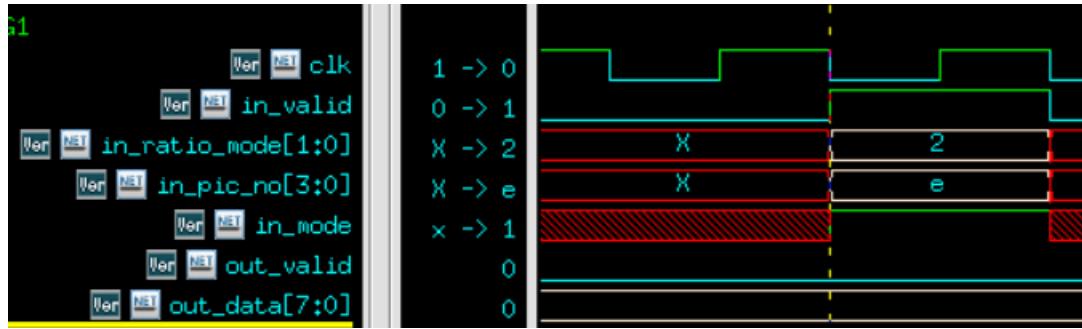


Fig. 5: Waveform for input mode 1: Auto Exposure.

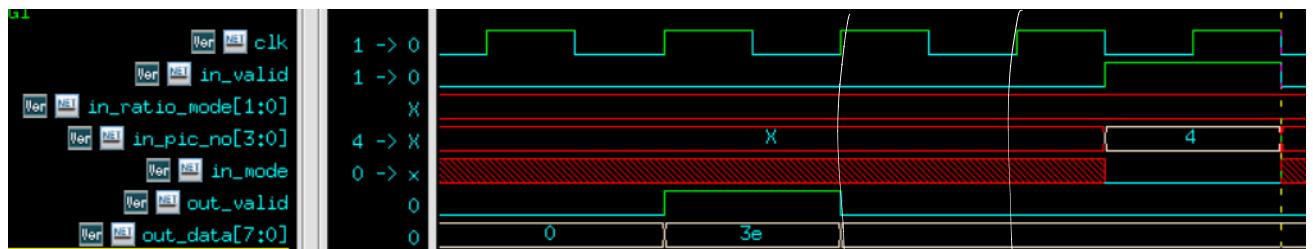


Fig. 6: Output waveform, the next input valid signal will come after 1 cycle.

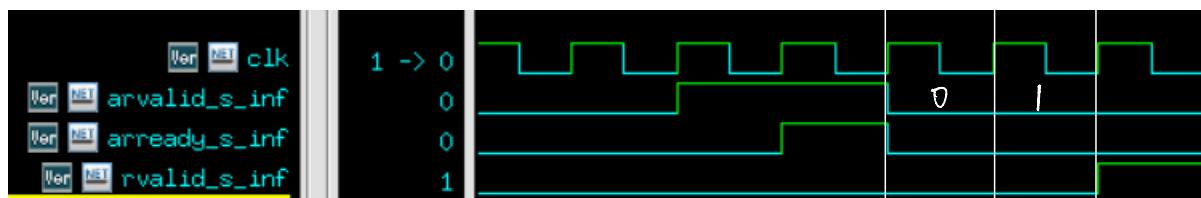
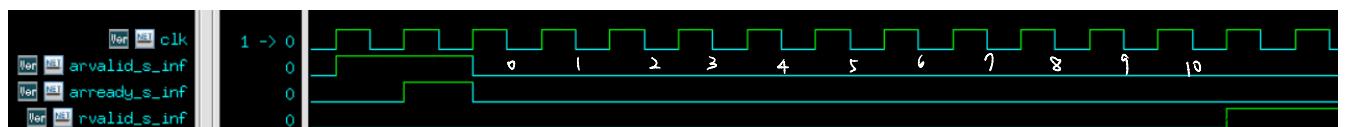
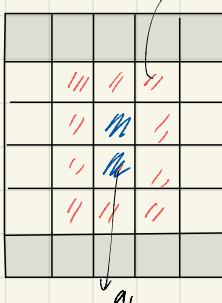
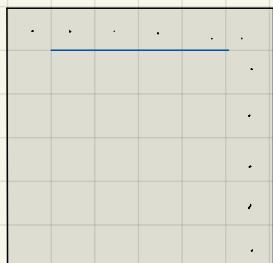
Fig. 7: Waveform for read latency set to 1.

Fig. 8: Waveform for read latency set to 10.



$$\frac{a_1+a_2}{4} \geq \frac{a_1+b_1+a_2+b_2+15}{16} \quad \frac{a+b}{16} = \frac{a+b+c+35}{36}$$

$\downarrow$

$$3ba + 3bb \geq 1ba + 1bb + 1bc$$

$$2a_1 + 2a_2 \geq 4b_1 + 4b_2 + 60$$

$$3a_1 + 3a_2 \geq b_1 + b_2 + 15$$

$$\begin{aligned}3ba + 3bb &\geq 1ba + 1bb + 1bc + 1b \times 35 \\20a + 20b &\geq 1bc + 1b \times 35 \\5a + 5b &\geq 4c + 140\end{aligned}$$

$$8+5=13$$

$$\log_{\sqrt{3}} 36 = 5$$

How to "detect" not enough resolution without reading data from DRAM?

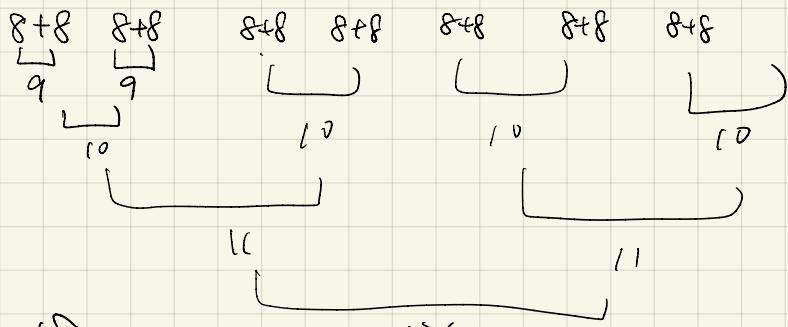
18 /

18 ≥

15,5

[5, 0]

$$8 + 10 = 18$$



$$9 + 9 = 18$$