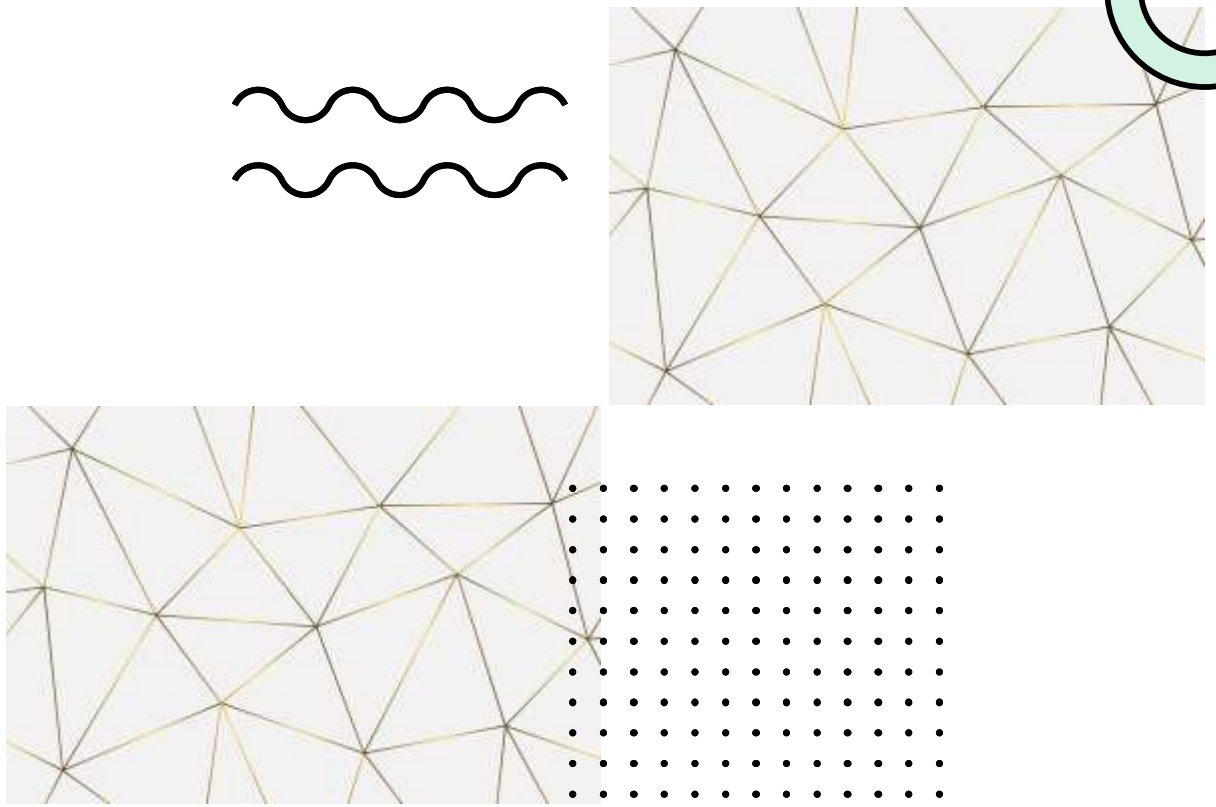
VGA LAB ROM NOTE



將圖片轉為12bit RGB圖檔

這邊我透過PYTHON進行圖片處理，分為以下步驟

1.加載圖像

2.圖像尺寸

3.提取紅綠藍分量

4.將每個通道轉為4bit格式

5.將各通道向左移進行拼接

6.初始化COE文件

7.RGB寫入COE文件中

8.最後一個逗號轉成分號

9.寫入COE中

```python
import numpy as np
from PIL import Image

# 1
image = Image.open("C:\\Users\\88697\\Desktop\\Xilinx\\0.Lab\\VGA_Lab\\fish.jpg")
image_array = np.array(image)

# 2
height, width, _ = image_array.shape

# 3
red = image_array[:,:,0]
green = image_array[:,:,1]
blue = image_array[:,:,2]

# 4
red_4bit = (red >> 4).astype(np.uint32)
green_4bit = (green >> 4).astype(np.uint32)
blue_4bit = (blue >> 4).astype(np.uint32)

# 5
rgb = (red_4bit << 8) + (green_4bit << 4) + blue_4bit

# 6
coe_content = "memory_initialization_radix=16;\nmemory_initialization_vector=\n"

#7
for i in range(height * width - 1):
    coe_content += "{:03X},\n".format(rgb.flatten()[i])

#8
coe_content += "{:03X};".format(rgb.flatten()[-1])

#9
with open("fish.coe", "w") as f:
    f.write(coe_content)

print("Sucessful")
```

COE

fish.coe

C: > Users > 88697 > fish.coe

```
1    memory_initialization_radix=16;
2    memory_initialization_vector=
3    FFF,
4    FFF,
5    FFF,
```

```
63597    FFF,
63598    FFF,
63599    FFF,
63600    FFF,
63601    FFF,
63602    AAA;
```

1.16進位,
2.比對圖片的基本資訊。
3.共有63600筆資料。(300*212)
4.FFF 因為RGB各暫4bit 總共12bit

Verilog

```
//IMG
parameter       C_IMG_WIDTH         =       'd300    ,
                C_IMG_HEIGH         =       'd212    ,
                C_IMG_PIX_NUM       =       'd63600  ; //300*212
```

1.圖片基本資訊

```
//ROM
reg     [15:0]          R_ROM_ADDR; //input addr
wire    [11:0]          ROM_DATA;   //output
```

1.ROM 的PORT

```
blk_mem_gen_0 u1 (
    .clka(R_clk_25M),
    .addra(R_ROM_ADDR),
    .douta(ROM_DATA)
  );
```

Verilog

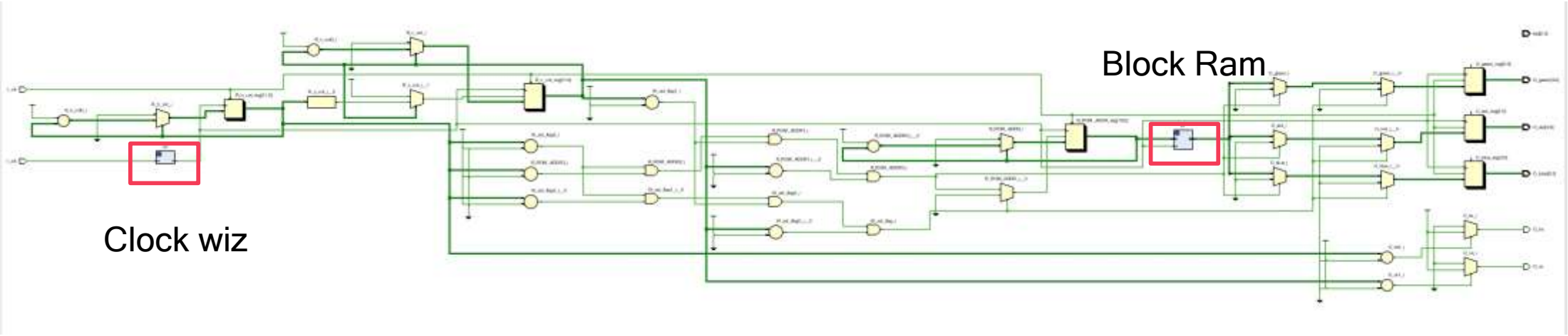
```
else if(W_act_flag) begin
    if(                 //IMG WIDTH HEIGH
        R_h_cnt >= (C_H_SYNC_PULSE + C_H_BACK_PORCH)                    &&
        R_h_cnt <= (C_H_SYNC_PULSE + C_H_BACK_PORCH + C_IMG_WIDTH - 'd1)  &&
        R_v_cnt >= (C_V_SYNC_PULSE + C_V_BACK_PORCH)                    &&
        R_v_cnt <= (C_V_SYNC_PULSE + C_V_BACK_PORCH + C_IMG_HEIGH - 'd1)
       )
        begin
            O_red       <= ROM_DATA[11:8]    ; //red
            O_green     <= ROM_DATA[7:4]     ; //green
            O_blue      <= ROM_DATA[3:0]      ; //blue
      O_red     = 'b1111;
      O_green   = 'b0000;
      O_blue    = 'b0000;

            if(R_ROM_ADDR == C_IMG_PIX_NUM  - 'd1)
            begin
                R_ROM_ADDR <= 'd0;
            end
            else
                R_ROM_ADDR <= R_ROM_ADDR +'d1;
        end
    else
        begin
            O_red       <=  'd0        ;
            O_green     <=  'd0        ;
            O_blue      <=  'd0        ;
            R_ROM_ADDR  <=  R_ROM_ADDR  ;
        end
end
else
    begin
        O_red       <=  'd0        ;
        O_green     <=  'd0        ;
        O_blue      <=  'd0        ;
        R_ROM_ADDR  <=  R_ROM_ADDR  ;
```

與W_act_flag相同的邏輯，針對圖片長寬進行判定

與python所排列的方式，進行定義

總地址，與圖片資料總數相同。

RTL



BUG

1.模擬時序觸發判定造成問題以及COE內部資料處理問題。
2.當圖片較大，會不斷的移動，如下圖，初步認為ROM 需設置ENABLE腳位。

成果

1.正常輸出結果

ISSUE

- 需要再去計算為何，正緣觸發以及敏感觸發的相差。