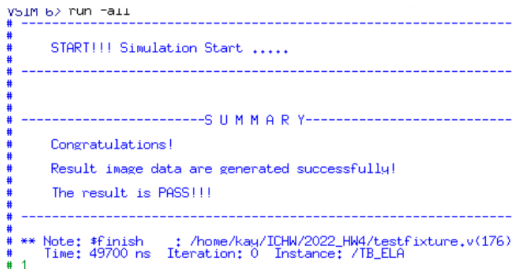
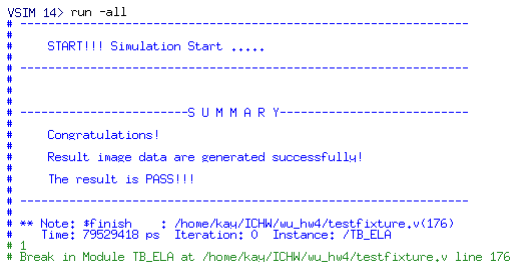
2022 Digital IC Design
Homework 4: Edge-Based Line Average interpolation

NAME	柳譯筑
Student ID	NE6101034

Simulation Result

Functional simulation	Pass	Gate-level simulation	Pass	Clock width	40 (ns)	Gate-level simulation time	79529 (ns)





Synthesis Result

Total logic elements	12,588/68,416(18%)
Total memory bit	8,820/68,416(13%)
Embedded multiplier 9-bit element	0/300(0%)

Flow Summary

Flow Status	Successful - Sat May 14 22:32:29 2022
Quartus II 32-bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Web Edition
Revision Name	ELA
Top-level Entity Name	ELA
Family	Cyclone II
Device	EP2C70F896C8
Timing Models	Final
Total logic elements	12,588 / 68,416 (18 %)
Total combinational functions	8,820 / 68,416 (13 %)
Dedicated logic registers	7,952 / 68,416 (12 %)
Total registers	7952
Total pins	39 / 622 (6 %)
Total virtual pins	0
Total memory bits	0 / 1,152,000 (0 %)
Embedded Multiplier 9-bit elements	0 / 300 (0 %)
Total PLLs	0 / 4 (0 %)

Description of your design

State:

Idle, Input, calculate, output,

在 input 的時候讀進奇數行的 img data,

```
for (i = 10'd0; i<10'd992; i=i+10'd1)begin
    if (i == cnt)begin
        n_img[i] = in_data;
    end else begin
        n_img[i] = {1'b0,img[i]};
    end
end
```

接著在 calculate 的階段把偶數行的 interpolate data 填滿。

我設定 D1, D2 , D3 分別為三個不同 pair 的 pixel 差值,

```
if(img[cnt-10'd33]>img[cnt+10'd33])begin
    D1 = img[cnt-10'd33]-img[cnt+10'd33];
end else begin
    D1 = img[cnt+10'd33]-img[cnt-10'd33];
end
if(img[cnt-10'd32]>img[cnt+10'd32])begin
    D2 = img[cnt-10'd32]-img[cnt+10'd32];
end else begin
    D2 = img[cnt+10'd32]-img[cnt-10'd32];
end
if(img[cnt-10'd31]>img[cnt+10'd31])begin
    D3 = img[cnt-10'd31]-img[cnt+10'd31];
end else begin
    D3 = img[cnt+10'd31]-img[cnt-10'd31];
end
```

要注意需要判斷在邊界的時候:

```
if (cnt[4:0]==5'd0)begin

    n_img[cnt] = ({1'b0,img[cnt-10'd32]}+{1'b0,img[cnt+10'd32]})>>1;
    D1 = 8'hff;
    D2 = 8'hff;
    D3 = 8'hff;
end else if (cnt[5:0]==6'd63)begin
```

```
    n_img[cnt] = ({1'b0,img[cnt-10'd32]}+{1'b0,img[cnt+10'd32]})>>1;
    D1 = 8'hff;
    D2 = 8'hff;
    D3 = 8'hff;
end
```

我直接把三個 D 的大小排列組合全部排出來:

```
// D1 min
if (D1<D2 && D1<D3)begin
    n_img[cnt] = ({1'b0,img[cnt-10'd33]}+{1'b0,img[cnt+10'd33]})>>1;
// D2 min
end else if(D2<D1 && D2<D3)begin
    n_img[cnt] = ({1'b0,img[cnt-10'd32]}+{1'b0,img[cnt+10'd32]})>>1;
// D3 min
end else if (D3<D1 && D3<D2)begin
    n_img[cnt] = ({1'b0,img[cnt-10'd31]}+{1'b0,img[cnt+10'd31]})>>1;
// D1=D2 min
end else if (D1==D2 && D1 < D3) begin
    n_img[cnt] = ({1'b0,img[cnt-10'd32]}+{1'b0,img[cnt+10'd32]})>>1;
// D2=D3 min
end else if (D3==D2 && D3 < D1)begin
    n_img[cnt] = ({1'b0,img[cnt-10'd32]}+{1'b0,img[cnt+10'd32]})>>1;
// D1 = D3 min
end else if(D1 == D3 && D1<D2)begin
    n_img[cnt] = ({1'b0,img[cnt-10'd33]}+{1'b0,img[cnt+10'd33]})>>1;
// D1=D2=D3 min
end else begin
    n_img[cnt] = ({1'b0,img[cnt-10'd32]}+{1'b0,img[cnt+10'd32]})>>1;
end
```

在 output 階段把 wen 拉高，所有資料 output 出去。

```
case(State)
    OUT:begin
        if (cnt == 10'd992)begin
            wen <= 1'b0;
```

```
        data_wr <= 8'hff;

end else begin
    wen <= 1'b1;
    data_wr <= img[addr];
end
end
default:begin
    wen = 1'b0;
    data_wr <= 8'hff;
end
endcase
```

這次作業 cycle 要改大才不會有 error，找了很久找不到 latch，最後是因為時間問題，總之成功合成真的感動～：D

*Scoring = (Total logic elements + total memory bit + 9*embedded multiplier 9-bit element) × (longest gate-level simulation time in ns)*