module hscaler (rst\_n\_scl,clk\_scl,en\_ff0,en\_ff1,en\_ff2,en\_ff3,scl\_i\_data\_r,scl\_i\_data\_g,scl\_i\_data\_b,scl\_cfg\_mode,

scl\_cfg\_flt,scl\_o\_data\_r,scl\_o\_data\_g,scl\_o\_data\_b);

input en\_ff0,en\_ff1,en\_ff2,en\_ff3;

input rst\_n\_scl,clk\_scl;

input scl\_cfg\_mode;

input [1:0] scl\_cfg\_flt;

input[7:0]scl\_i\_data\_r;

input[7:0]scl\_i\_data\_g;

input[7:0]scl\_i\_data\_b;

output[7:0]scl\_o\_data\_r;

output[7:0]scl\_o\_data\_g;

output[7:0]scl\_o\_data\_b;

reg[7:0]scl\_o\_data\_r;

reg[7:0]scl\_o\_data\_g;

reg[7:0]scl\_o\_data\_b;

//input r g b

reg [7:0] data\_in\_r\_0,data\_in\_r\_1,data\_in\_r\_2,data\_in\_r\_3,data\_in\_r\_4,data\_in\_r\_5;

reg [7:0] data\_in\_g\_0,data\_in\_g\_1,data\_in\_g\_2,data\_in\_g\_3,data\_in\_g\_4,data\_in\_g\_5;

reg [7:0] data\_in\_b\_0,data\_in\_b\_1,data\_in\_b\_2,data\_in\_b\_3,data\_in\_b\_4,data\_in\_b\_5;

always @(posedge clk\_scl or negedge rst\_n\_scl)

begin

if(!rst\_n\_scl)

begin

data\_in\_r\_0 <= 8'b0;

data\_in\_r\_1 <= 8'b0;

data\_in\_r\_2 <= 8'b0;

data\_in\_r\_3 <= 8'b0;

data\_in\_r\_4 <= 8'b0;

data\_in\_r\_5 <= 8'b0;

end

else

begin

data\_in\_r\_0 <= scl\_i\_data\_r;

data\_in\_r\_1 <= data\_in\_r\_0;

data\_in\_r\_2 <= data\_in\_r\_1;

data\_in\_r\_3 <= data\_in\_r\_2;

data\_in\_r\_4 <= data\_in\_r\_3;

data\_in\_r\_5 <= data\_in\_r\_4;

end

end

always @(posedge clk\_scl or negedge rst\_n\_scl)

begin

if(!rst\_n\_scl)

begin

data\_in\_g\_0 <= 8'b0;

data\_in\_g\_1 <= 8'b0;

data\_in\_g\_2 <= 8'b0;

data\_in\_g\_3 <= 8'b0;

data\_in\_g\_4 <= 8'b0;

data\_in\_g\_5 <= 8'b0;

end

else

begin

data\_in\_g\_0 <= scl\_i\_data\_g;

data\_in\_g\_1 <= data\_in\_g\_0;

data\_in\_g\_2 <= data\_in\_g\_1;

data\_in\_g\_3 <= data\_in\_g\_2;

data\_in\_g\_4 <= data\_in\_g\_3;

data\_in\_g\_5 <= data\_in\_g\_4;

end

end

always @(posedge clk\_scl or negedge rst\_n\_scl)

begin

if(!rst\_n\_scl)

begin

data\_in\_b\_0 <= 8'b0;

data\_in\_b\_1 <= 8'b0;

data\_in\_b\_2 <= 8'b0;

data\_in\_b\_3 <= 8'b0;

data\_in\_b\_4 <= 8'b0;

data\_in\_b\_5 <= 8'b0;

end

else

begin

data\_in\_b\_0 <= scl\_i\_data\_b;

data\_in\_b\_1 <= data\_in\_b\_0;

data\_in\_b\_2 <= data\_in\_b\_1;

data\_in\_b\_3 <= data\_in\_b\_2;

data\_in\_b\_4 <= data\_in\_b\_3;

data\_in\_b\_5 <= data\_in\_b\_4;

end

end

//???????

reg signed [10:0] coef\_0,coef\_1,coef\_2,coef\_3;

always @(posedge clk\_scl or negedge rst\_n\_scl)

begin

if (!rst\_n\_scl)

begin

coef\_0 <= 0;

coef\_1 <= 0;

coef\_2 <= 0;

coef\_3 <= 0;

end

else

begin

case(scl\_cfg\_flt)

2'd0:begin

coef\_0 <= -3;

coef\_1 <= 498;

coef\_2 <= 18;

coef\_3 <= -1;

end

2'd1: begin

coef\_0 <= -38;

coef\_1 <= 376;

coef\_2 <= 202;

coef\_3 <= -28;

end

2'd2: begin

coef\_0 <= -28;

coef\_1 <= 202;

coef\_2 <= 376;

coef\_3 <= -38;

end

2'd3: begin

coef\_0 <= -1;

coef\_1 <= 18;

coef\_2 <= 498;

coef\_3 <= -3;

end

default:begin

coef\_0 <= 0;

coef\_1 <= 512;

coef\_2 <= 0;

coef\_3 <= 0;

end

endcase

end

end

//???&???

wire signed [8:0] itmp\_r\_0,itmp\_r\_1,itmp\_r\_2,itmp\_r\_3;

wire signed [8:0] itmp\_g\_0,itmp\_g\_1,itmp\_g\_2,itmp\_g\_3;

wire signed [8:0] itmp\_b\_0,itmp\_b\_1,itmp\_b\_2,itmp\_b\_3;

reg signed [20:0] ff\_r\_0,ff\_r\_1,ff\_r\_2,ff\_r\_3;

reg signed [20:0] ff\_g\_0,ff\_g\_1,ff\_g\_2,ff\_g\_3;

reg signed [20:0] ff\_b\_0,ff\_b\_1,ff\_b\_2,ff\_b\_3;

assign itmp\_r\_0 = {1'b0,data\_in\_r\_0};

assign itmp\_r\_1 = {1'b0,data\_in\_r\_1};

assign itmp\_r\_2 = {1'b0,data\_in\_r\_2};

assign itmp\_r\_3 = {1'b0,data\_in\_r\_3};

assign itmp\_g\_0 = {1'b0,data\_in\_g\_0};

assign itmp\_g\_1 = {1'b0,data\_in\_g\_0};

assign itmp\_g\_2 = {1'b0,data\_in\_g\_0};

assign itmp\_g\_3 = {1'b0,data\_in\_g\_0};

assign itmp\_b\_0 = {1'b0,data\_in\_b\_0};

assign itmp\_b\_1 = {1'b0,data\_in\_b\_0};

assign itmp\_b\_2 = {1'b0,data\_in\_b\_0};

assign itmp\_b\_3 = {1'b0,data\_in\_b\_0};

always @(posedge clk\_scl or negedge rst\_n\_scl)

begin

if(!rst\_n\_scl)

begin

ff\_r\_0 <= 0;

ff\_r\_1 <= 0;

ff\_r\_2 <= 0;

ff\_r\_3 <= 0;

ff\_g\_0 <= 0;

ff\_g\_1 <= 0;

ff\_g\_2 <= 0;

ff\_g\_3 <= 0;

ff\_b\_0 <= 0;

ff\_b\_1 <= 0;

ff\_b\_2 <= 0;

ff\_b\_3 <= 0;

end

else if(en\_ff2==1 && en\_ff3==0) //???????????

begin

ff\_r\_0 <= itmp\_r\_0\*coef\_3;

ff\_r\_1 <= itmp\_r\_1\*coef\_2;

ff\_r\_2 <= itmp\_r\_2\*coef\_1;

ff\_r\_3 <= itmp\_r\_2\*coef\_0;

ff\_g\_0 <= itmp\_g\_0\*coef\_3;

ff\_g\_1 <= itmp\_g\_1\*coef\_2;

ff\_g\_2 <= itmp\_g\_2\*coef\_1;

ff\_g\_3 <= itmp\_g\_2\*coef\_0;

ff\_b\_0 <= itmp\_b\_0\*coef\_3;

ff\_b\_1 <= itmp\_b\_1\*coef\_2;

ff\_b\_2 <= itmp\_b\_2\*coef\_1;

ff\_b\_3 <= itmp\_b\_2\*coef\_0;

end

else if(en\_ff0==0 && en\_ff1==1) //?????????????

begin

ff\_r\_0 <= itmp\_r\_1\*coef\_3;

ff\_r\_1 <= itmp\_r\_1\*coef\_2;

ff\_r\_2 <= itmp\_r\_2\*coef\_1;

ff\_r\_3 <= itmp\_r\_3\*coef\_0;

ff\_g\_0 <= itmp\_g\_1\*coef\_3;

ff\_g\_1 <= itmp\_g\_1\*coef\_2;

ff\_g\_2 <= itmp\_g\_2\*coef\_1;

ff\_g\_3 <= itmp\_g\_3\*coef\_0;

ff\_b\_0 <= itmp\_b\_1\*coef\_3;

ff\_b\_1 <= itmp\_b\_1\*coef\_2;

ff\_b\_2 <= itmp\_b\_2\*coef\_1;

ff\_b\_3 <= itmp\_b\_3\*coef\_0;

end

else if(en\_ff1==0 && en\_ff2==1) //?????????????

begin

ff\_r\_0 <= itmp\_r\_2\*coef\_3;

ff\_r\_1 <= itmp\_r\_2\*coef\_2;

ff\_r\_2 <= itmp\_r\_2\*coef\_1;

ff\_r\_3 <= itmp\_r\_3\*coef\_0;

ff\_g\_0 <= itmp\_g\_2\*coef\_3;

ff\_g\_1 <= itmp\_g\_2\*coef\_2;

ff\_g\_2 <= itmp\_g\_2\*coef\_1;

ff\_g\_3 <= itmp\_g\_3\*coef\_0;

ff\_b\_0 <= itmp\_b\_2\*coef\_3;

ff\_b\_1 <= itmp\_b\_2\*coef\_2;

ff\_b\_2 <= itmp\_b\_2\*coef\_1;

ff\_b\_3 <= itmp\_b\_3\*coef\_0;

end

else

begin

ff\_r\_0 <= itmp\_r\_0\*coef\_3;

ff\_r\_1 <= itmp\_r\_1\*coef\_2;

ff\_r\_2 <= itmp\_r\_2\*coef\_1;

ff\_r\_3 <= itmp\_r\_3\*coef\_0;

ff\_g\_0 <= itmp\_g\_0\*coef\_3;

ff\_g\_1 <= itmp\_g\_1\*coef\_2;

ff\_g\_2 <= itmp\_g\_2\*coef\_1;

ff\_g\_3 <= itmp\_g\_3\*coef\_0;

ff\_b\_0 <= itmp\_b\_0\*coef\_3;

ff\_b\_1 <= itmp\_b\_1\*coef\_2;

ff\_b\_2 <= itmp\_b\_2\*coef\_1;

ff\_b\_3 <= itmp\_b\_3\*coef\_0;

end

end

//??????

reg signed [21:0] ff1\_r\_0,ff1\_r\_1;

reg signed [21:0] ff1\_g\_0,ff1\_g\_1;

reg signed [21:0] ff1\_b\_0,ff1\_b\_1;

always @(posedge clk\_scl or negedge rst\_n\_scl)

begin

if(!rst\_n\_scl)

begin

ff1\_r\_0 <= 0;

ff1\_r\_1 <= 0;

ff1\_g\_0 <= 0;

ff1\_g\_1 <= 0;

ff1\_b\_0 <= 0;

ff1\_b\_1 <= 0;

end

else

begin

ff1\_r\_0 <= ff\_r\_0+ff\_r\_1;

ff1\_r\_1 <= ff\_r\_2+ff\_r\_3;

ff1\_g\_0 <= ff\_g\_0+ff\_g\_1;

ff1\_g\_1 <= ff\_g\_2+ff\_g\_3;

ff1\_b\_0 <= ff\_b\_0+ff\_b\_1;

ff1\_b\_1 <= ff\_b\_2+ff\_b\_3;

end

end

//??????

reg signed [22:0] ff2\_r;

reg signed [22:0] ff2\_g;

reg signed [22:0] ff2\_b;

always @(posedge clk\_scl or negedge rst\_n\_scl)

begin

if(!rst\_n\_scl)

begin

ff2\_r <= 0;

ff2\_g <= 0;

ff2\_b <= 0;

end

else

begin

ff2\_r <= ff1\_r\_0+ff1\_r\_1;

ff2\_g <= ff1\_g\_0+ff1\_g\_1;

ff2\_b <= ff1\_b\_0+ff1\_b\_1;

end

end

//output r g b

always @(posedge clk\_scl or negedge rst\_n\_scl)

begin

if(!rst\_n\_scl)

begin

scl\_o\_data\_r <= 8'b0;

scl\_o\_data\_g <= 8'b0;

scl\_o\_data\_b <= 8'b0;

end

else if(scl\_cfg\_mode==0)

begin

scl\_o\_data\_r <= data\_in\_r\_5;

scl\_o\_data\_g <= data\_in\_g\_5;

scl\_o\_data\_b <= data\_in\_b\_5;

end

else

begin

if(ff2\_r<0)

begin

scl\_o\_data\_r <= 0;

end

else if(ff2\_r>130560)

begin

scl\_o\_data\_r <= 255;

end

else begin

scl\_o\_data\_r <= ff2\_r[16:9];

end

if(ff2\_g<0) begin

scl\_o\_data\_g <= 0;

end

else if(ff2\_g>130560)

begin

scl\_o\_data\_g <= 255;

end

else begin

scl\_o\_data\_g <= ff2\_g[16:9];

end

if(ff2\_b<0) begin

scl\_o\_data\_b <= 0;

end

else if(ff2\_b>130560)

begin

scl\_o\_data\_b <= 255;

end

else begin

scl\_o\_data\_b <= ff2\_b[16:9];

end

end

end

endmodule