module fpu\_topmodule( clk, rst,enable,select\_root,rmode, fpu\_op,ring\_enable,opa, opb, out, ready, underflow,overflow, inexact, exception, invalid);

input clk;

input rst;

input enable;

input select\_root;

input [1:0] rmode;

input [2:0] fpu\_op;

input [63:0] opa, opb;

output [4:0]ring\_enable;

output [63:0] out;

output ready;

output underflow;

output overflow;

output inexact;

output exception;

output invalid;

reg [4:0]ring\_enable;

reg [4:0] ring\_enable\_reg;

reg [63:0] opa\_reg;

reg [63:0] opb\_reg;

reg [2:0] fpu\_op\_reg;

reg [1:0] rmode\_reg;

reg enable\_reg;

reg enable\_reg\_1;

reg enable\_reg\_2;

reg enable\_reg\_3;

reg op\_enable;

reg [63:0] out;

reg [6:0] count\_cycles;

reg [6:0] count\_ready;

wire count\_busy = (count\_ready <= count\_cycles);

reg ready;

reg ready\_0;

reg ready\_1;

reg underflow;

reg overflow;

reg inexact;

reg exception;

reg invalid;

wire underflow\_0;

wire overflow\_0;

wire inexact\_0;

wire exception\_0;

wire invalid\_0;

wire add\_enable\_0 = (fpu\_op\_reg == 3'b000) & !(opa\_reg[63] ^ opb\_reg[63]);

wire add\_enable\_1 = (fpu\_op\_reg == 3'b001) & (opa\_reg[63] ^ opb\_reg[63]);

reg add\_enable;

wire sub\_enable\_0 = (fpu\_op\_reg == 3'b000) & (opa\_reg[63] ^ opb\_reg[63]);

wire sub\_enable\_1 = (fpu\_op\_reg == 3'b001) & !(opa\_reg[63] ^ opb\_reg[63]);

reg sub\_enable;

reg mul\_enable;

reg div\_enable;

reg sqroot\_enable;

wire [55:0] sum\_out;

wire [55:0] diff\_out;

reg [55:0] addsub\_out;

wire [55:0] mul\_out;

wire [55:0] div\_out;

wire [55:0] sqroot\_out;

reg [55:0] mantissa\_round;

wire [10:0] exp\_add\_out;

wire [10:0] exp\_sub\_out;

wire [11:0] exp\_mul\_out;

wire [11:0] exp\_div\_out;

wire [11:0] exp\_sqroot\_out;

reg [11:0] exponent\_round;

reg [11:0] exp\_addsub;

wire [11:0] exponent\_post\_round;

wire add\_sign;

wire sub\_sign;

wire mul\_sign;

wire div\_sign;

wire sqroot\_sign;

reg addsub\_sign;

reg sign\_round;

wire [63:0] out\_round;

wire [63:0] out\_except;

fpu\_add u1(

.clk(clk),.rst(rst),.enable(add\_enable),.opa(opa\_reg),.opb(opb\_reg),

.sign(add\_sign),.sum\_2(sum\_out),.exponent\_2(exp\_add\_out));

fpu\_sub u2(

.clk(clk),.rst(rst),.enable(sub\_enable),.opa(opa\_reg),.opb(opb\_reg),

.fpu\_op(fpu\_op\_reg),.sign(sub\_sign),.diff\_2(diff\_out),

.exponent\_2(exp\_sub\_out));

fpu\_mul u3(

.clk(clk),.rst(rst),.enable(mul\_enable),.opa(opa\_reg),.opb(opb\_reg),

.sign(mul\_sign),.product\_7(mul\_out),.exponent\_5(exp\_mul\_out));

fpu\_div u4(

.clk(clk),.rst(rst),.enable(div\_enable),.opa(opa\_reg),.opb(opb\_reg),

.sign(div\_sign),.mantissa\_7(div\_out),.exponent\_out(exp\_div\_out));

fpu\_sqroot u5(.clk(clk),.rst(rst),.enable(sqroot\_enable),.select(select\_root),.opa(opa\_reg),.opb(opb\_reg),.sign(sqroot\_sign),.exponent\_root(exp\_sqroot\_out),.mantissa\_root(sqroot\_out));

fpu\_round u6(.clk(clk),.rst(rst),.enable(op\_enable),.round\_mode(rmode\_reg),

.sign\_term(sign\_round),.mantissa\_term(mantissa\_round), .exponent\_term(exponent\_round),

.round\_out(out\_round),.exponent\_final(exponent\_post\_round));

fpu\_exceptions u7(.clk(clk),.rst(rst),.enable(op\_enable),.rmode(rmode\_reg),

.opa(opa\_reg),.opb(opb\_reg),

.in\_except(out\_round), .exponent\_in(exponent\_post\_round),

.mantissa\_in(mantissa\_round[1:0]),.fpu\_op(fpu\_op\_reg),.out(out\_except),

.ex\_enable(except\_enable),.underflow(underflow\_0),.overflow(overflow\_0),

.inexact(inexact\_0),.exception(exception\_0),.invalid(invalid\_0));

always @(posedge clk)

begin

case (fpu\_op\_reg)

3'b000: mantissa\_round <= addsub\_out;

3'b001: mantissa\_round <= addsub\_out;

3'b010: mantissa\_round <= mul\_out;

3'b011: mantissa\_round <= div\_out;

3'b100: mantissa\_round <= sqroot\_out;

default: mantissa\_round <= 0;

endcase

end

always @(posedge clk)

begin

case (fpu\_op\_reg)

3'b000: exponent\_round <= exp\_addsub;

3'b001: exponent\_round <= exp\_addsub;

3'b010: exponent\_round <= exp\_mul\_out;

3'b011: exponent\_round <= exp\_div\_out;

3'b100: exponent\_round <= exp\_sqroot\_out;

default: exponent\_round <= 0;

endcase

end

always @(posedge clk)

begin

case (fpu\_op\_reg)

3'b000: sign\_round <= addsub\_sign;

3'b001: sign\_round <= addsub\_sign;

3'b010: sign\_round <= mul\_sign;

3'b011: sign\_round <= div\_sign;

3'b100: sign\_round <= sqroot\_sign;

default: sign\_round <= 0;

endcase

end

always @(posedge clk)

begin

case (fpu\_op\_reg)

3'b000: count\_cycles <= 20;

3'b001: count\_cycles <= 21;

3'b010: count\_cycles <= 24;

3'b011: count\_cycles <= 71;

3'b100: count\_cycles <= 71;

default: count\_cycles <= 0;

endcase

end

always @(posedge clk)

begin

if (rst) begin

add\_enable <= 0;

sub\_enable <= 0;

mul\_enable <= 0;

div\_enable <= 0;

sqroot\_enable <=0;

addsub\_out <= 0;

addsub\_sign <= 0;

exp\_addsub <= 0;

end

else begin

add\_enable <= (add\_enable\_0 | add\_enable\_1) & op\_enable & ring\_enable\_reg[0];

sub\_enable <= (sub\_enable\_0 | sub\_enable\_1) & op\_enable & ring\_enable\_reg[1];

mul\_enable <= (fpu\_op\_reg == 3'b010) & op\_enable & ring\_enable\_reg[2];

div\_enable <= (fpu\_op\_reg == 3'b011) & op\_enable & enable\_reg\_3 & ring\_enable\_reg[3];

sqroot\_enable <=(fpu\_op\_reg ==3'b100) & op\_enable & enable\_reg\_3 &ring\_enable\_reg[4];

// div\_enable needs to be high for two clock cycles

addsub\_out <= add\_enable ? sum\_out : diff\_out;

addsub\_sign <= add\_enable ? add\_sign : sub\_sign;

exp\_addsub <= add\_enable ? { 1'b0, exp\_add\_out} : { 1'b0, exp\_sub\_out};

end

end

always @ (posedge clk)

begin

if (rst)

count\_ready <= 0;

else if (enable\_reg\_1)

count\_ready <= 0;

else if (count\_busy)

count\_ready <= count\_ready + 1;

end

always @(posedge clk)

begin

if (rst) begin

enable\_reg <= 0;

enable\_reg\_1 <= 0;

enable\_reg\_2 <= 0;

enable\_reg\_3 <= 0;

end

else begin

enable\_reg <= enable;

enable\_reg\_1 <= enable & !enable\_reg;

enable\_reg\_2 <= enable\_reg\_1;

enable\_reg\_3 <= enable\_reg\_1 | enable\_reg\_2;

end

end

always @(posedge clk)

begin

if (rst) begin

opa\_reg <= 0;

opb\_reg <= 0;

fpu\_op\_reg <= 0;

rmode\_reg <= 0;

op\_enable <= 0;

end

else if (enable\_reg\_1) begin

opa\_reg <= opa;

opb\_reg <= opb;

fpu\_op\_reg <= fpu\_op;

rmode\_reg <= rmode;

op\_enable <= 1;

end

end

always @(posedge clk)

begin

case(fpu\_op\_reg)

3'b000:

begin

ring\_enable\_reg <=5'b00001;

ring\_enable <=ring\_enable\_reg;

end

3'b001:

begin

ring\_enable\_reg <=5'b00010;

ring\_enable <=ring\_enable\_reg;

end

3'b010:

begin

ring\_enable\_reg <=5'b00100;

ring\_enable <=ring\_enable\_reg;

end

3'b011:

begin

ring\_enable\_reg <=5'b01000;

ring\_enable <=ring\_enable\_reg;

end

3'b100:

begin

ring\_enable\_reg <=5'b10000;

ring\_enable <=ring\_enable\_reg;

end

default

ring\_enable\_reg <=5'b00001;

endcase

end

always @(posedge clk)

begin

if (rst) begin

ready\_0 <= 0;

ready\_1 <= 0;

ready <= 0;

end

else if (enable\_reg\_1) begin

ready\_0 <= 0;

ready\_1 <= 0;

ready <= 0;

end

else begin

ready\_0 <= !count\_busy;

ready\_1 <= ready\_0;

ready <= ready\_1;

end

end

always @(posedge clk)

begin

if (rst) begin

underflow <= 0;

overflow <= 0;

inexact <= 0;

exception <= 0;

invalid <= 0;

out <= 0;

end

else if (ready\_1) begin

underflow <= underflow\_0;

overflow <= overflow\_0;

inexact <= inexact\_0;

exception <= exception\_0;

invalid <= invalid\_0;

out <= except\_enable ? out\_except : out\_round;

end

end

endmodule

module fpu\_add ( clk, rst, enable, opa, opb, sign, sum\_2, exponent\_2);

input clk;

input rst;

input enable;

input [63:0] opa, opb;

output sign;

output [55:0] sum\_2;

output [10:0]exponent\_2;

reg sign;

reg [10:0] exponent\_a;

reg [10:0] exponent\_b;

reg [51:0] mantissa\_a;

reg [51:0] mantissa\_b;

reg expa\_gt\_expb;

reg [10:0] exponent\_small;

reg [10:0] exponent\_large;

reg [51:0] mantissa\_small;

reg [51:0] mantissa\_large;

reg small\_is\_denorm;

reg large\_is\_denorm;

reg large\_norm\_small\_denorm;

reg [10:0] exponent\_diff;

reg [55:0] large\_add;

reg [55:0] small\_add;

reg [55:0] small\_shift;

wire small\_shift\_nonzero = |small\_shift[55:0];

wire small\_is\_nonzero = (exponent\_small > 0) | (mantissa\_small[51:0]);

wire small\_fraction\_enable = small\_is\_nonzero & !small\_shift\_nonzero;

wire [55:0] small\_shift\_2 = { 55'b0, 1'b1 };

reg [55:0] small\_shift\_3;

reg [55:0] sum;

wire sum\_overflow = sum[55];

reg [55:0] sum\_2;

reg [10:0] exponent;

wire sum\_leading\_one = sum\_2[54];

reg denorm\_to\_norm;

reg [10:0] exponent\_2;

always @(posedge clk)

begin

if (rst) begin

sign <= 0;

exponent\_a <= 0;

exponent\_b <= 0;

mantissa\_a <= 0;

mantissa\_b <= 0;

expa\_gt\_expb <= 0;

exponent\_small <= 0;

exponent\_large <= 0;

mantissa\_small <= 0;

mantissa\_large <= 0;

small\_is\_denorm <= 0;

large\_is\_denorm <= 0;

large\_norm\_small\_denorm <= 0;

exponent\_diff <= 0;

large\_add <= 0;

small\_add <= 0;

small\_shift <= 0;

small\_shift\_3 <= 0;

sum <= 0;

sum\_2 <= 0;

exponent <= 0;

denorm\_to\_norm <= 0;

exponent\_2 <= 0;

end

else if (enable) begin

sign <= opa[63];

exponent\_a <= opa[62:52];

exponent\_b <= opb[62:52];

mantissa\_a <= opa[51:0];

mantissa\_b <= opb[51:0];

expa\_gt\_expb <= exponent\_a > exponent\_b;

exponent\_small <= expa\_gt\_expb ? exponent\_b : exponent\_a;

exponent\_large <= expa\_gt\_expb ? exponent\_a : exponent\_b;

mantissa\_small <= expa\_gt\_expb ? mantissa\_b : mantissa\_a;

mantissa\_large <= expa\_gt\_expb ? mantissa\_a : mantissa\_b;

small\_is\_denorm <= !(exponent\_small > 0);

large\_is\_denorm <= !(exponent\_large > 0);

large\_norm\_small\_denorm <= (small\_is\_denorm && !large\_is\_denorm);

exponent\_diff <= exponent\_large - exponent\_small -large\_norm\_small\_denorm;

large\_add <= { 1'b0, !large\_is\_denorm, mantissa\_large, 2'b0 };

small\_add <= { 1'b0, !small\_is\_denorm, mantissa\_small, 2'b0 };

small\_shift <= small\_add >> exponent\_diff;

small\_shift\_3 <= small\_fraction\_enable ? small\_shift\_2 : small\_shift;

sum <= large\_add + small\_shift\_3;

sum\_2 <= sum\_overflow ? sum >> 1 : sum;

exponent <= sum\_overflow ? exponent\_large + 1: exponent\_large;

denorm\_to\_norm <= sum\_leading\_one & large\_is\_denorm;

exponent\_2 <= denorm\_to\_norm ? exponent + 1 : exponent;

end

end

endmodule

module fpu\_sub( clk, rst, enable, opa, opb, fpu\_op, sign, diff\_2, exponent\_2);

input clk;

input rst;

input enable;

input [63:0] opa, opb;

input [2:0] fpu\_op;

output sign;

output [55:0] diff\_2;

output [10:0] exponent\_2;

reg [6:0] diff\_shift;

reg [6:0] diff\_shift\_2;

reg [10:0] exponent\_a;

reg [10:0] exponent\_b;

reg [51:0] mantissa\_a;

reg [51:0] mantissa\_b;

reg expa\_gt\_expb;

reg expa\_et\_expb;

reg mana\_gtet\_manb;

reg a\_gtet\_b;

reg sign;

reg [10:0] exponent\_small;

reg [10:0] exponent\_large;

reg [51:0] mantissa\_small;

reg [51:0] mantissa\_large;

reg small\_is\_denorm;

reg large\_is\_denorm;

reg large\_norm\_small\_denorm;

reg small\_is\_nonzero;

reg [10:0] exponent\_diff;

reg [54:0] minuend;

reg [54:0] subtrahend;

reg [54:0] subtra\_shift;

wire subtra\_shift\_nonzero = |subtra\_shift[54:0];

wire subtra\_fraction\_enable = small\_is\_nonzero & !subtra\_shift\_nonzero;

wire [54:0] subtra\_shift\_2 = { 54'b0, 1'b1 };

reg [54:0] subtra\_shift\_3;

reg [54:0] diff;

reg diffshift\_gt\_exponent;

reg diffshift\_et\_55;

reg [54:0] diff\_1;

reg [10:0] exponent;

reg [10:0] exponent\_2;

wire in\_norm\_out\_denorm = (exponent\_large > 0) & (exponent== 0);

reg [55:0] diff\_2;

always @(posedge clk)

begin

if (rst) begin

exponent\_a <= 0;

exponent\_b <= 0;

mantissa\_a <= 0;

mantissa\_b <= 0;

expa\_gt\_expb <= 0;

expa\_et\_expb <= 0;

mana\_gtet\_manb <= 0;

a\_gtet\_b <= 0;

sign <= 0;

exponent\_small <= 0;

exponent\_large <= 0;

mantissa\_small <= 0;

mantissa\_large <= 0;

small\_is\_denorm <= 0;

large\_is\_denorm <= 0;

large\_norm\_small\_denorm <= 0;

small\_is\_nonzero <= 0;

exponent\_diff <= 0;

minuend <= 0;

subtrahend <= 0;

subtra\_shift <= 0;

subtra\_shift\_3 <= 0;

diff\_shift\_2 <= 0;

diff <= 0;

diffshift\_gt\_exponent <= 0;

diffshift\_et\_55 <= 0;

diff\_1 <= 0;

exponent <= 0;

exponent\_2 <= 0;

diff\_2 <= 0;

end

else if (enable) begin

exponent\_a <= opa[62:52];

exponent\_b <= opb[62:52];

mantissa\_a <= opa[51:0];

mantissa\_b <= opb[51:0];

expa\_gt\_expb <= exponent\_a > exponent\_b;

expa\_et\_expb <= exponent\_a == exponent\_b;

mana\_gtet\_manb <= mantissa\_a >= mantissa\_b;

a\_gtet\_b <= expa\_gt\_expb | (expa\_et\_expb & mana\_gtet\_manb);

sign <= a\_gtet\_b ? opa[63] :!opb[63] ^ (fpu\_op == 3'b000);

exponent\_small <= a\_gtet\_b ? exponent\_b : exponent\_a;

exponent\_large <= a\_gtet\_b ? exponent\_a : exponent\_b;

mantissa\_small <= a\_gtet\_b ? mantissa\_b : mantissa\_a;

mantissa\_large <= a\_gtet\_b ? mantissa\_a : mantissa\_b;

small\_is\_denorm <= !(exponent\_small > 0);

large\_is\_denorm <= !(exponent\_large > 0);

large\_norm\_small\_denorm <= (small\_is\_denorm == 1 && large\_is\_denorm == 0);

small\_is\_nonzero <= (exponent\_small > 0) | (mantissa\_small[51:0]);

exponent\_diff <= exponent\_large - exponent\_small - large\_norm\_small\_denorm;

minuend <= { !large\_is\_denorm, mantissa\_large, 2'b00 };

subtrahend <= { !small\_is\_denorm, mantissa\_small, 2'b00 };

subtra\_shift <= subtrahend >> exponent\_diff;

subtra\_shift\_3 <= subtra\_fraction\_enable ? subtra\_shift\_2 : subtra\_shift;

diff\_shift\_2 <= diff\_shift;

diff <= minuend - subtra\_shift\_3;

diffshift\_gt\_exponent <= diff\_shift\_2 > exponent\_large;

diffshift\_et\_55 <= diff\_shift\_2 == 55;

diff\_1 <= diffshift\_gt\_exponent ? diff << exponent\_large : diff << diff\_shift\_2;

exponent <= diffshift\_gt\_exponent ? 0 : (exponent\_large - diff\_shift\_2);

exponent\_2 <= diffshift\_et\_55 ? 0 : exponent;

diff\_2 <= in\_norm\_out\_denorm ? { 1'b0, diff\_1 >> 1} : {1'b0, diff\_1};

end

end

always @(diff)

casex(diff)

55'b1??????????????????????????????????????????????????????: diff\_shift <= 0; 55'b01?????????????????????????????????????????????????????: diff\_shift <= 1;

55'b001????????????????????????????????????????????????????: diff\_shift <= 2;

55'b0001???????????????????????????????????????????????????: diff\_shift <= 3;

55'b00001??????????????????????????????????????????????????: diff\_shift <= 4;

55'b000001?????????????????????????????????????????????????: diff\_shift <= 5;

55'b0000001????????????????????????????????????????????????: diff\_shift <= 6;

55'b00000001???????????????????????????????????????????????: diff\_shift <= 7;

55'b000000001??????????????????????????????????????????????: diff\_shift <= 8;

55'b0000000001?????????????????????????????????????????????: diff\_shift <= 9;

55'b00000000001????????????????????????????????????????????: diff\_shift <= 10;

55'b000000000001???????????????????????????????????????????: diff\_shift <= 11;

55'b0000000000001??????????????????????????????????????????: diff\_shift <= 12;

55'b00000000000001?????????????????????????????????????????: diff\_shift <= 13;

55'b000000000000001????????????????????????????????????????: diff\_shift <= 14;

55'b0000000000000001???????????????????????????????????????: diff\_shift <= 15;

55'b00000000000000001??????????????????????????????????????: diff\_shift <= 16;

55'b000000000000000001?????????????????????????????????????: diff\_shift <= 17;

55'b0000000000000000001????????????????????????????????????: diff\_shift <= 18;

55'b00000000000000000001???????????????????????????????????: diff\_shift <= 19;

55'b000000000000000000001??????????????????????????????????: diff\_shift <= 20;

55'b0000000000000000000001?????????????????????????????????: diff\_shift <= 21;

55'b00000000000000000000001????????????????????????????????: diff\_shift <= 22;

55'b000000000000000000000001???????????????????????????????: diff\_shift <= 23;

55'b0000000000000000000000001??????????????????????????????: diff\_shift <= 24;

55'b00000000000000000000000001?????????????????????????????: diff\_shift <= 25;

55'b000000000000000000000000001????????????????????????????: diff\_shift <= 26;

55'b0000000000000000000000000001???????????????????????????: diff\_shift <= 27;

55'b00000000000000000000000000001??????????????????????????: diff\_shift <= 28;

55'b000000000000000000000000000001?????????????????????????: diff\_shift <= 29;

55'b0000000000000000000000000000001????????????????????????: diff\_shift <= 30;

55'b00000000000000000000000000000001???????????????????????: diff\_shift <= 31;

55'b000000000000000000000000000000001??????????????????????: diff\_shift <= 32;

55'b0000000000000000000000000000000001?????????????????????: diff\_shift <= 33;

55'b00000000000000000000000000000000001????????????????????: diff\_shift <= 34;

55'b000000000000000000000000000000000001???????????????????: diff\_shift <= 35;

55'b0000000000000000000000000000000000001??????????????????: diff\_shift <= 36;

55'b00000000000000000000000000000000000001?????????????????: diff\_shift <= 37;

55'b000000000000000000000000000000000000001????????????????: diff\_shift <= 38;

55'b0000000000000000000000000000000000000001???????????????: diff\_shift <= 39;

55'b00000000000000000000000000000000000000001??????????????: diff\_shift <= 40;

55'b000000000000000000000000000000000000000001?????????????: diff\_shift <= 41;

55'b0000000000000000000000000000000000000000001????????????: diff\_shift <= 42;

55'b00000000000000000000000000000000000000000001???????????: diff\_shift <=43;

55'b000000000000000000000000000000000000000000001??????????: diff\_shift <=44;

55'b0000000000000000000000000000000000000000000001?????????: diff\_shift <=45;

55'b00000000000000000000000000000000000000000000001????????: diff\_shift <=46;

55'b000000000000000000000000000000000000000000000001???????: diff\_shift <=47;

55'b0000000000000000000000000000000000000000000000001??????: diff\_shift<=48;

55'b00000000000000000000000000000000000000000000000001?????: diff\_shift<=49;

55'b000000000000000000000000000000000000000000000000001????: diff\_shift<=50;

55'b0000000000000000000000000000000000000000000000000001???: diff\_shift<=51;

55'b00000000000000000000000000000000000000000000000000001??: diff\_shift<=52;

55'b000000000000000000000000000000000000000000000000000001?: diff\_shift<=53;

55'b0000000000000000000000000000000000000000000000000000001: diff\_shift <= 54; 55'b0000000000000000000000000000000000000000000000000000000: diff\_shift <= 55;

endcase

endmodule

module fpu\_mul( clk, rst, enable, opa, opb, sign, product\_7, exponent\_5);

input clk;

input rst;

input enable;

input [63:0] opa, opb;

output sign;

output [55:0] product\_7;

output [11:0] exponent\_5;

reg [5:0] product\_shift;

reg [5:0] product\_shift\_2;

reg sign;

reg [51:0] mantissa\_a;

reg [51:0] mantissa\_b;

reg [10:0] exponent\_a;

reg [10:0] exponent\_b;

reg a\_is\_norm;

reg b\_is\_norm;

reg a\_is\_zero;

reg b\_is\_zero;

reg in\_zero;

reg [11:0] exponent\_terms;

reg exponent\_gt\_expoffset;

reg [11:0] exponent\_under;

reg [11:0] exponent\_1;

wire [11:0] exponent = 0;

reg [11:0] exponent\_2;

reg exponent\_gt\_prodshift;

reg [11:0] exponent\_3;

reg [11:0] exponent\_4;

reg exponent\_et\_zero;

reg [52:0] mul\_a;

reg [52:0] mul\_b;

reg [40:0] product\_a;

reg [40:0] product\_b;

reg [40:0] product\_c;

reg [25:0] product\_d;

reg [33:0] product\_e;

reg [33:0] product\_f;

reg [35:0] product\_g;

reg [28:0] product\_h;

reg [28:0] product\_i;

reg [30:0] product\_j;

reg [41:0] sum\_0;

reg [35:0] sum\_1;

reg [41:0] sum\_2;

reg [35:0] sum\_3;

reg [36:0] sum\_4;

reg [27:0] sum\_5;

reg [29:0] sum\_6;

reg [36:0] sum\_7;

reg [30:0] sum\_8;

reg [105:0] product;

reg [105:0] product\_1;

reg [105:0] product\_2;

reg [105:0] product\_3;

reg [105:0] product\_4;

reg [105:0] product\_5;

reg [105:0] product\_6;

reg product\_lsb;

wire [55:0] product\_7 = { 1'b0, product\_6[105:52], product\_lsb };

reg [11:0] exponent\_5;

always @(posedge clk)

begin

if (rst) begin

sign <= 0;

mantissa\_a <= 0;

mantissa\_b <= 0;

exponent\_a <= 0;

exponent\_b <= 0;

a\_is\_norm <= 0;

b\_is\_norm <= 0;

a\_is\_zero <= 0;

b\_is\_zero <= 0;

in\_zero <= 0;

exponent\_terms <= 0;

exponent\_gt\_expoffset <= 0;

exponent\_under <= 0;

exponent\_1 <= 0;

exponent\_2 <= 0;

exponent\_gt\_prodshift <= 0;

exponent\_3 <= 0;

exponent\_4 <= 0;

exponent\_et\_zero <= 0;

mul\_a <= 0;

mul\_b <= 0;

product\_a <= 0;

product\_b <= 0;

product\_c <= 0;

product\_d <= 0;

product\_e <= 0;

product\_f <= 0;

product\_g <= 0;

product\_h <= 0;

product\_i <= 0;

product\_j <= 0;

sum\_0 <= 0;

sum\_1 <= 0;

sum\_2 <= 0;

sum\_3 <= 0;

sum\_4 <= 0;

sum\_5 <= 0;

sum\_6 <= 0;

sum\_7 <= 0;

sum\_8 <= 0;

product <= 0;

product\_1 <= 0;

product\_2 <= 0;

product\_3 <= 0;

product\_4 <= 0;

product\_5 <= 0;

product\_6 <= 0;

product\_lsb <= 0;

exponent\_5 <= 0;

product\_shift\_2 <= 0;

end

else if (enable) begin

sign <= opa[63] ^ opb[63];

mantissa\_a <= opa[51:0];

mantissa\_b <= opb[51:0];

exponent\_a <= opa[62:52];

exponent\_b <= opb[62:52];

a\_is\_norm <= |exponent\_a;

b\_is\_norm <= |exponent\_b;

a\_is\_zero <= !(|opa[62:0]);

b\_is\_zero <= !(|opb[62:0]);

in\_zero <= a\_is\_zero | b\_is\_zero;

exponent\_terms <= exponent\_a + exponent\_b + !a\_is\_norm + !b\_is\_norm;

exponent\_gt\_expoffset <= exponent\_terms > 1021;

exponent\_under <= 1022 - exponent\_terms;

exponent\_1 <= exponent\_terms - 1022;

exponent\_2 <= exponent\_gt\_expoffset ? exponent\_1 : exponent;

exponent\_gt\_prodshift <= exponent\_2 > product\_shift\_2;

exponent\_3 <= exponent\_2 - product\_shift;

exponent\_4 <= exponent\_gt\_prodshift ? exponent\_3 : exponent;

exponent\_et\_zero <= exponent\_4 == 0;

mul\_a <= { a\_is\_norm, mantissa\_a };

mul\_b <= { b\_is\_norm, mantissa\_b };

product\_a <= mul\_a[23:0] \* mul\_b[16:0];

product\_b <= mul\_a[23:0] \* mul\_b[33:17];

product\_c <= mul\_a[23:0] \* mul\_b[50:34];

product\_d <= mul\_a[23:0] \* mul\_b[52:51];

product\_e <= mul\_a[40:24] \* mul\_b[16:0];

product\_f <= mul\_a[40:24] \* mul\_b[33:17];

product\_g <= mul\_a[40:24] \* mul\_b[52:34];

product\_h <= mul\_a[52:41] \* mul\_b[16:0];

product\_i <= mul\_a[52:41] \* mul\_b[33:17];

product\_j <= mul\_a[52:41] \* mul\_b[52:34];

sum\_0 <= product\_a[40:17] + product\_b;

sum\_1 <= sum\_0[41:7] + product\_e;

sum\_2 <= sum\_1[35:10] + product\_c;

sum\_3 <= sum\_2[41:7] + product\_h;

sum\_4 <= sum\_3 + product\_f;

sum\_5 <= sum\_4[36:10] + product\_d;

sum\_6 <= sum\_5[27:7] + product\_i;

sum\_7 <= sum\_6 + product\_g;

sum\_8 <= sum\_7[36:17] + product\_j;

product <= { sum\_8, sum\_7[16:0], sum\_5[6:0], sum\_4[9:0], sum\_2[6:0],

sum\_1[9:0], sum\_0[6:0], product\_a[16:0] };

product\_1 <= product >> exponent\_under;

product\_2 <= exponent\_gt\_expoffset ? product : product\_1;

product\_3 <= product\_2 << product\_shift\_2;

product\_4 <= product\_2 << exponent\_2;

product\_5 <= exponent\_gt\_prodshift ? product\_3 : product\_4;

product\_6 <= exponent\_et\_zero ? product\_5 >> 1 : product\_5;

product\_lsb <= |product\_6[51:0];

exponent\_5 <= in\_zero ? 12'b0 : exponent\_4;

product\_shift\_2 <= product\_shift;

end

end

always @(product)

casex(product)

106'b1?????????????????????????????????????????????????????????????????????????????????????????????????????????: product\_shift <= 0;

106'b01????????????????????????????????????????????????????????????????????????????????????????????????????????: product\_shift <= 1;

106'b001???????????????????????????????????????????????????????????????????????????????????????????????????????: product\_shift <= 2;

106'b0001??????????????????????????????????????????????????????????????????????????????????????????????????????: product\_shift <= 3;

106'b00001?????????????????????????????????????????????????????????????????????????????????????????????????????: product\_shift <= 4;

106'b000001????????????????????????????????????????????????????????????????????????????????????????????????????: product\_shift <= 5;

106'b0000001???????????????????????????????????????????????????????????????????????????????????????????????????: product\_shift <= 6;

106'b00000001??????????????????????????????????????????????????????????????????????????????????????????????????: product\_shift <= 7;

106'b000000001?????????????????????????????????????????????????????????????????????????????????????????????????: product\_shift <= 8;

106'b0000000001????????????????????????????????????????????????????????????????????????????????????????????????: product\_shift <= 9;

106'b00000000001???????????????????????????????????????????????????????????????????????????????????????????????: product\_shift <= 10;

106'b000000000001??????????????????????????????????????????????????????????????????????????????????????????????: product\_shift <= 11;

106'b0000000000001?????????????????????????????????????????????????????????????????????????????????????????????: product\_shift <= 12;

106'b00000000000001????????????????????????????????????????????????????????????????????????????????????????????: product\_shift <= 13;

106'b000000000000001???????????????????????????????????????????????????????????????????????????????????????????: product\_shift <= 14;

106'b0000000000000001??????????????????????????????????????????????????????????????????????????????????????????: product\_shift <= 15;

106'b00000000000000001?????????????????????????????????????????????????????????????????????????????????????????: product\_shift <= 16;

106'b000000000000000001????????????????????????????????????????????????????????????????????????????????????????: product\_shift <= 17;

106'b0000000000000000001???????????????????????????????????????????????????????????????????????????????????????: product\_shift <= 18;

106'b00000000000000000001??????????????????????????????????????????????????????????????????????????????????????: product\_shift <= 19;

106'b000000000000000000001?????????????????????????????????????????????????????????????????????????????????????: product\_shift <= 20;

106'b0000000000000000000001????????????????????????????????????????????????????????????????????????????????????: product\_shift <= 21;

106'b00000000000000000000001???????????????????????????????????????????????????????????????????????????????????: product\_shift <= 22;

106'b000000000000000000000001??????????????????????????????????????????????????????????????????????????????????: product\_shift <= 23;

106'b0000000000000000000000001?????????????????????????????????????????????????????????????????????????????????: product\_shift <= 24;

106'b00000000000000000000000001????????????????????????????????????????????????????????????????????????????????: product\_shift <= 25;

106'b000000000000000000000000001???????????????????????????????????????????????????????????????????????????????: product\_shift <= 26;

106'b0000000000000000000000000001??????????????????????????????????????????????????????????????????????????????: product\_shift <= 27;

106'b00000000000000000000000000001?????????????????????????????????????????????????????????????????????????????: product\_shift <= 28;

106'b000000000000000000000000000001????????????????????????????????????????????????????????????????????????????: product\_shift <= 29;

106'b0000000000000000000000000000001???????????????????????????????????????????????????????????????????????????: product\_shift <= 30;

106'b00000000000000000000000000000001??????????????????????????????????????????????????????????????????????????: product\_shift <= 31;

106'b000000000000000000000000000000001?????????????????????????????????????????????????????????????????????????: product\_shift <= 32;

106'b0000000000000000000000000000000001????????????????????????????????????????????????????????????????????????: product\_shift <= 33;

106'b00000000000000000000000000000000001???????????????????????????????????????????????????????????????????????: product\_shift <= 34;

106'b000000000000000000000000000000000001??????????????????????????????????????????????????????????????????????: product\_shift <= 35;

106'b0000000000000000000000000000000000001?????????????????????????????????????????????????????????????????????: product\_shift <= 36;

106'b00000000000000000000000000000000000001????????????????????????????????????????????????????????????????????: product\_shift <= 37;

106'b000000000000000000000000000000000000001???????????????????????????????????????????????????????????????????: product\_shift <= 38;

106'b0000000000000000000000000000000000000001??????????????????????????????????????????????????????????????????: product\_shift <= 39;

106'b00000000000000000000000000000000000000001?????????????????????????????????????????????????????????????????: product\_shift <= 40;

106'b000000000000000000000000000000000000000001????????????????????????????????????????????????????????????????: product\_shift <= 41;

106'b0000000000000000000000000000000000000000001???????????????????????????????????????????????????????????????: product\_shift <= 42;

106'b00000000000000000000000000000000000000000001??????????????????????????????????????????????????????????????: product\_shift <= 43;

106'b000000000000000000000000000000000000000000001?????????????????????????????????????????????????????????????: product\_shift <= 44;

106'b0000000000000000000000000000000000000000000001????????????????????????????????????????????????????????????: product\_shift <= 45;

106'b00000000000000000000000000000000000000000000001???????????????????????????????????????????????????????????: product\_shift <= 46;

106'b000000000000000000000000000000000000000000000001??????????????????????????????????????????????????????????: product\_shift <= 47;

106'b0000000000000000000000000000000000000000000000001?????????????????????????????????????????????????????????: product\_shift <= 48;

106'b00000000000000000000000000000000000000000000000001????????????????????????????????????????????????????????: product\_shift <= 49;

106'b000000000000000000000000000000000000000000000000001???????????????????????????????????????????????????????: product\_shift <= 50;

106'b0000000000000000000000000000000000000000000000000001??????????????????????????????????????????????????????: product\_shift <= 51;

106'b00000000000000000000000000000000000000000000000000001?????????????????????????????????????????????????????: product\_shift <= 52;

106'b000000000000000000000000000000000000000000000000000000????????????????????????????????????????????????????: product\_shift <= 53;

endcase

endmodule

module fpu\_div( clk, rst, enable, opa, opb, sign, mantissa\_7,

exponent\_out);

input clk;

input rst;

input enable;

input [63:0] opa;

input [63:0] opb;

output sign;

output [55:0] mantissa\_7;

output [11:0] exponent\_out;

parameter preset = 53;

reg [53:0] dividend\_reg;

reg [53:0] divisor\_reg;

reg enable\_reg;

reg enable\_reg\_2;

reg enable\_reg\_a;

reg enable\_reg\_b;

reg enable\_reg\_c;

reg enable\_reg\_d;

reg enable\_reg\_e;

reg [5:0] dividend\_shift;

reg [5:0] dividend\_shift\_2;

reg [5:0] divisor\_shift;

reg [5:0] divisor\_shift\_2;

reg [5:0] count\_out;

reg [11:0] exponent\_out;

wire sign = opa[63] ^ opb[63];

reg [51:0] mantissa\_a;

reg [51:0] mantissa\_b;

wire [10:0] expon\_a = opa[62:52];

wire [10:0] expon\_b = opb[62:52];

wire a\_is\_norm = |expon\_a;

wire b\_is\_norm = |expon\_b;

wire a\_is\_zero = !(|opa[62:0]);

wire [11:0] exponent\_a = { 1'b0, expon\_a};

wire [11:0] exponent\_b = { 1'b0, expon\_b};

reg [51:0] dividend\_a;

reg [51:0] dividend\_a\_shifted;

wire [52:0] dividend\_denorm = { dividend\_a\_shifted, 1'b0};

wire [53:0] dividend\_1 = a\_is\_norm ? { 2'b01, dividend\_a } : { 1'b0, dividend\_denorm};

reg [51:0] divisor\_b;

reg [51:0] divisor\_b\_shifted;

wire [52:0] divisor\_denorm = { divisor\_b\_shifted, 1'b0};

wire [53:0] divisor\_1 = b\_is\_norm ? { 2'b01, divisor\_b } : { 1'b0, divisor\_denorm};

wire [5:0] count\_index = count\_out;

wire count\_nonzero = !(count\_index == 0);

reg [53:0] quotient;

reg [53:0] quotient\_out;

reg [53:0] remainder;

reg [53:0] remainder\_out;

reg remainder\_msb;

reg count\_nonzero\_reg;

reg count\_nonzero\_reg\_2;

reg [11:0] expon\_term;

reg expon\_uf\_1;

reg [11:0] expon\_uf\_term\_1;

reg [11:0] expon\_final\_1;

reg [11:0] expon\_final\_2;

reg [11:0] expon\_shift\_a;

reg [11:0] expon\_shift\_b;

reg expon\_uf\_2;

reg [11:0] expon\_uf\_term\_2;

reg [11:0] expon\_uf\_term\_3;

reg expon\_uf\_gt\_maxshift;

reg [11:0] expon\_uf\_term\_4;

reg [11:0] expon\_final\_3;

reg [11:0] expon\_final\_4;

wire quotient\_msb = quotient\_out[53];

reg expon\_final\_4\_et0;

reg expon\_final\_4\_term;

reg [11:0] expon\_final\_5;

reg [51:0] mantissa\_1;

wire [51:0] mantissa\_2 = quotient\_out[52:1];

wire [51:0] mantissa\_3 = quotient\_out[51:0];

wire [51:0] mantissa\_4 = quotient\_msb ? mantissa\_2 : mantissa\_3;

wire [51:0] mantissa\_5 = (expon\_final\_4 == 1) ? mantissa\_2 : mantissa\_4;

wire [51:0] mantissa\_6 = expon\_final\_4\_et0 ? mantissa\_1 : mantissa\_5;

wire [107:0] remainder\_a = { quotient\_out[53:0] , remainder\_msb, remainder\_out[52:0]};

reg [6:0] remainder\_shift\_term;

reg [107:0] remainder\_b;

wire [55:0] remainder\_1 = remainder\_b[107:52];

wire [55:0] remainder\_2 = { quotient\_out[0] , remainder\_msb, remainder\_out[52:0], 1'b0 };

wire [55:0] remainder\_3 = { remainder\_msb , remainder\_out[52:0], 2'b0 };

wire [55:0] remainder\_4 = quotient\_msb ? remainder\_2 : remainder\_3;

wire [55:0] remainder\_5 = (expon\_final\_4 == 1) ? remainder\_2 : remainder\_4;

wire [55:0] remainder\_6 = expon\_final\_4\_et0 ? remainder\_1 : remainder\_5;

wire m\_norm = |expon\_final\_5;

wire rem\_lsb = |remainder\_6[54:0];

wire [55:0] mantissa\_7 = { 1'b0, m\_norm, mantissa\_6, remainder\_6[55], rem\_lsb };

always @ (posedge clk)

begin

if (rst)

exponent\_out <= 0;

else

exponent\_out <= a\_is\_zero ? 12'b0 : expon\_final\_5;

end

always @ (posedge clk)

begin

if (rst)

count\_out <= 0;

else if (enable\_reg)

count\_out <= preset;

else if (count\_nonzero)

count\_out <= count\_out - 1;

end

always @ (posedge clk)

begin

if (rst) begin

quotient\_out <= 0;

remainder\_out <= 0;

end

else begin

quotient\_out <= quotient;

remainder\_out <= remainder;

end

end

always @ (posedge clk)

begin

if (rst)

quotient <= 0;

else if (count\_nonzero\_reg)

quotient[count\_index] <= !(divisor\_reg > dividend\_reg);

end

always @ (posedge clk)

begin

if (rst) begin

remainder <= 0;

remainder\_msb <= 0;

end

else if (!count\_nonzero\_reg & count\_nonzero\_reg\_2) begin

remainder <= dividend\_reg;

remainder\_msb <= (divisor\_reg > dividend\_reg) ? 0 : 1;

end

end

always @ (posedge clk)

begin

if (rst) begin

dividend\_reg <= 0;

divisor\_reg <= 0;

end

else if (enable\_reg\_e) begin

dividend\_reg <= dividend\_1;

divisor\_reg <= divisor\_1;

end

else if (count\_nonzero\_reg)

dividend\_reg <= (divisor\_reg > dividend\_reg) ? dividend\_reg << 1 :

(dividend\_reg - divisor\_reg) << 1;

// divisor doesn't change for the divide

end

always @ (posedge clk)

begin

if (rst) begin

expon\_term <= 0;

expon\_uf\_1 <= 0;

expon\_uf\_term\_1 <= 0;

expon\_final\_1 <= 0;

expon\_final\_2 <= 0;

expon\_shift\_a <= 0;

expon\_shift\_b <= 0;

expon\_uf\_2 <= 0;

expon\_uf\_term\_2 <= 0;

expon\_uf\_term\_3 <= 0;

expon\_uf\_gt\_maxshift <= 0;

expon\_uf\_term\_4 <= 0;

expon\_final\_3 <= 0;

expon\_final\_4 <= 0;

expon\_final\_4\_et0 <= 0;

expon\_final\_4\_term <= 0;

expon\_final\_5 <= 0;

mantissa\_a <= 0;

mantissa\_b <= 0;

dividend\_a <= 0;

divisor\_b <= 0;

dividend\_shift\_2 <= 0;

divisor\_shift\_2 <= 0;

remainder\_shift\_term <= 0;

remainder\_b <= 0;

dividend\_a\_shifted <= 0;

divisor\_b\_shifted <= 0;

mantissa\_1 <= 0;

end

else if (enable\_reg\_2) begin

expon\_term <= exponent\_a + 1023;

expon\_uf\_1 <= exponent\_b > expon\_term;

expon\_uf\_term\_1 <= expon\_uf\_1 ? (exponent\_b - expon\_term) : 0;

expon\_final\_1 <= expon\_term - exponent\_b;

expon\_final\_2 <= expon\_uf\_1 ? 0 : expon\_final\_1;

expon\_shift\_a <= a\_is\_norm ? 0 : dividend\_shift\_2;

expon\_shift\_b <= b\_is\_norm ? 0 : divisor\_shift\_2;

expon\_uf\_2 <= expon\_shift\_a > expon\_final\_2;

expon\_uf\_term\_2 <= expon\_uf\_2 ? (expon\_shift\_a - expon\_final\_2) : 0;

expon\_uf\_term\_3 <= expon\_uf\_term\_2 + expon\_uf\_term\_1;

expon\_uf\_gt\_maxshift <= (expon\_uf\_term\_3 > 51);

expon\_uf\_term\_4 <= expon\_uf\_gt\_maxshift ? 52 : expon\_uf\_term\_3;

expon\_final\_3 <= expon\_uf\_2 ? 0 : (expon\_final\_2 - expon\_shift\_a);

expon\_final\_4 <= expon\_final\_3 + expon\_shift\_b;

expon\_final\_4\_et0 <= (expon\_final\_4 == 0);

expon\_final\_4\_term <= expon\_final\_4\_et0 ? 0 : 1;

expon\_final\_5 <= quotient\_msb ? expon\_final\_4 : expon\_final\_4 - expon\_final\_4\_term;

mantissa\_a <= opa[51:0];

mantissa\_b <= opb[51:0];

dividend\_a <= mantissa\_a;

divisor\_b <= mantissa\_b;

dividend\_shift\_2 <= dividend\_shift;

divisor\_shift\_2 <= divisor\_shift;

remainder\_shift\_term <= 52 - expon\_uf\_term\_4;

remainder\_b <= remainder\_a << remainder\_shift\_term;

dividend\_a\_shifted <= dividend\_a << dividend\_shift\_2;

divisor\_b\_shifted <= divisor\_b << divisor\_shift\_2;

mantissa\_1 <= quotient\_out[53:2] >> expon\_uf\_term\_4;

end

end

always @ (posedge clk)

begin

if (rst) begin

count\_nonzero\_reg <= 0;

count\_nonzero\_reg\_2 <= 0;

enable\_reg <= 0;

enable\_reg\_a <= 0;

enable\_reg\_b <= 0;

enable\_reg\_c <= 0;

enable\_reg\_d <= 0;

enable\_reg\_e <= 0;

end

else begin

count\_nonzero\_reg <= count\_nonzero;

count\_nonzero\_reg\_2 <= count\_nonzero\_reg;

enable\_reg <= enable\_reg\_e;

enable\_reg\_a <= enable;

enable\_reg\_b <= enable\_reg\_a;

enable\_reg\_c <= enable\_reg\_b;

enable\_reg\_d <= enable\_reg\_c;

enable\_reg\_e <= enable\_reg\_d;

end

end

always @ (posedge clk)

begin

if (rst)

enable\_reg\_2 <= 0;

else if (enable)

enable\_reg\_2 <= 1;

end

always @(dividend\_a)

casex(dividend\_a)

52'b1???????????????????????????????????????????????????: dividend\_shift <= 0;

52'b01??????????????????????????????????????????????????: dividend\_shift <= 1;

52'b001?????????????????????????????????????????????????: dividend\_shift <= 2;

52'b0001????????????????????????????????????????????????: dividend\_shift <= 3;

52'b00001???????????????????????????????????????????????: dividend\_shift <= 4;

52'b000001??????????????????????????????????????????????: dividend\_shift <= 5;

52'b0000001?????????????????????????????????????????????: dividend\_shift <= 6;

52'b00000001????????????????????????????????????????????: dividend\_shift <= 7;

52'b000000001???????????????????????????????????????????: dividend\_shift <= 8;

52'b0000000001??????????????????????????????????????????: dividend\_shift <= 9;

52'b00000000001?????????????????????????????????????????: dividend\_shift <= 10;

52'b000000000001????????????????????????????????????????: dividend\_shift <= 11;

52'b0000000000001???????????????????????????????????????: dividend\_shift <= 12;

52'b00000000000001??????????????????????????????????????: dividend\_shift <= 13;

52'b000000000000001?????????????????????????????????????: dividend\_shift <= 14;

52'b0000000000000001????????????????????????????????????: dividend\_shift <= 15;

52'b00000000000000001???????????????????????????????????: dividend\_shift <= 16;

52'b000000000000000001??????????????????????????????????: dividend\_shift <= 17;

52'b0000000000000000001?????????????????????????????????: dividend\_shift <= 18;

52'b00000000000000000001????????????????????????????????: dividend\_shift <= 19;

52'b000000000000000000001???????????????????????????????: dividend\_shift <= 20;

52'b0000000000000000000001??????????????????????????????: dividend\_shift <= 21;

52'b00000000000000000000001?????????????????????????????: dividend\_shift <= 22;

52'b000000000000000000000001????????????????????????????: dividend\_shift <= 23;

52'b0000000000000000000000001???????????????????????????: dividend\_shift <= 24;

52'b00000000000000000000000001??????????????????????????: dividend\_shift <= 25;

52'b000000000000000000000000001?????????????????????????: dividend\_shift <= 26;

52'b0000000000000000000000000001????????????????????????: dividend\_shift <= 27;

52'b00000000000000000000000000001???????????????????????: dividend\_shift <= 28;

52'b000000000000000000000000000001??????????????????????: dividend\_shift <= 29;

52'b0000000000000000000000000000001?????????????????????: dividend\_shift <= 30;

52'b00000000000000000000000000000001????????????????????: dividend\_shift <= 31;

52'b000000000000000000000000000000001???????????????????: dividend\_shift <= 32;

52'b0000000000000000000000000000000001??????????????????: dividend\_shift <= 33;

52'b00000000000000000000000000000000001?????????????????: dividend\_shift <= 34;

52'b000000000000000000000000000000000001????????????????: dividend\_shift <= 35;

52'b0000000000000000000000000000000000001???????????????: dividend\_shift <= 36;

52'b00000000000000000000000000000000000001??????????????: dividend\_shift <= 37;

52'b000000000000000000000000000000000000001?????????????: dividend\_shift <= 38;

52'b0000000000000000000000000000000000000001????????????: dividend\_shift <= 39;

52'b00000000000000000000000000000000000000001???????????: dividend\_shift <= 40;

52'b000000000000000000000000000000000000000001??????????: dividend\_shift <= 41;

52'b0000000000000000000000000000000000000000001?????????: dividend\_shift <= 42;

52'b00000000000000000000000000000000000000000001????????: dividend\_shift <= 43;

52'b000000000000000000000000000000000000000000001???????: dividend\_shift <= 44;

52'b0000000000000000000000000000000000000000000001??????: dividend\_shift <= 45;

52'b00000000000000000000000000000000000000000000001?????: dividend\_shift <= 46;

52'b000000000000000000000000000000000000000000000001????: dividend\_shift <= 47;

52'b0000000000000000000000000000000000000000000000001???: dividend\_shift <= 48;

52'b00000000000000000000000000000000000000000000000001??: dividend\_shift <= 49;

52'b000000000000000000000000000000000000000000000000001?: dividend\_shift <= 50;

52'b0000000000000000000000000000000000000000000000000001: dividend\_shift <= 51;

52'b0000000000000000000000000000000000000000000000000000: dividend\_shift <= 52;

endcase

always @(divisor\_b)

casex(divisor\_b)

52'b1???????????????????????????????????????????????????: divisor\_shift <= 0;

52'b01??????????????????????????????????????????????????: divisor\_shift <= 1;

52'b001?????????????????????????????????????????????????: divisor\_shift <= 2;

52'b0001????????????????????????????????????????????????: divisor\_shift <= 3;

52'b00001???????????????????????????????????????????????: divisor\_shift <= 4;

52'b000001??????????????????????????????????????????????: divisor\_shift <= 5;

52'b0000001?????????????????????????????????????????????: divisor\_shift <= 6;

52'b00000001????????????????????????????????????????????: divisor\_shift <= 7;

52'b000000001???????????????????????????????????????????: divisor\_shift <= 8;

52'b0000000001??????????????????????????????????????????: divisor\_shift <= 9;

52'b00000000001?????????????????????????????????????????: divisor\_shift <= 10;

52'b000000000001????????????????????????????????????????: divisor\_shift <= 11;

52'b0000000000001???????????????????????????????????????: divisor\_shift <= 12;

52'b00000000000001??????????????????????????????????????: divisor\_shift <= 13;

52'b000000000000001?????????????????????????????????????: divisor\_shift <= 14;

52'b0000000000000001????????????????????????????????????: divisor\_shift <= 15;

52'b00000000000000001???????????????????????????????????: divisor\_shift <= 16;

52'b000000000000000001??????????????????????????????????: divisor\_shift <= 17;

52'b0000000000000000001?????????????????????????????????: divisor\_shift <= 18;

52'b00000000000000000001????????????????????????????????: divisor\_shift <= 19;

52'b000000000000000000001???????????????????????????????: divisor\_shift <= 20;

52'b0000000000000000000001??????????????????????????????: divisor\_shift <= 21;

52'b00000000000000000000001?????????????????????????????: divisor\_shift <= 22;

52'b000000000000000000000001????????????????????????????: divisor\_shift <= 23;

52'b0000000000000000000000001???????????????????????????: divisor\_shift <= 24;

52'b00000000000000000000000001??????????????????????????: divisor\_shift <= 25;

52'b000000000000000000000000001?????????????????????????: divisor\_shift <= 26;

52'b0000000000000000000000000001????????????????????????: divisor\_shift <= 27;

52'b00000000000000000000000000001???????????????????????: divisor\_shift <= 28;

52'b000000000000000000000000000001??????????????????????: divisor\_shift <= 29;

52'b0000000000000000000000000000001?????????????????????: divisor\_shift <= 30;

52'b00000000000000000000000000000001????????????????????: divisor\_shift <= 31;

52'b000000000000000000000000000000001???????????????????: divisor\_shift <= 32;

52'b0000000000000000000000000000000001??????????????????: divisor\_shift <= 33;

52'b00000000000000000000000000000000001?????????????????: divisor\_shift <= 34;

52'b000000000000000000000000000000000001????????????????: divisor\_shift <= 35;

52'b0000000000000000000000000000000000001???????????????: divisor\_shift <= 36;

52'b00000000000000000000000000000000000001??????????????: divisor\_shift <= 37;

52'b000000000000000000000000000000000000001?????????????: divisor\_shift <= 38;

52'b0000000000000000000000000000000000000001????????????: divisor\_shift <= 39;

52'b00000000000000000000000000000000000000001???????????: divisor\_shift <= 40;

52'b000000000000000000000000000000000000000001??????????: divisor\_shift <= 41;

52'b0000000000000000000000000000000000000000001?????????: divisor\_shift <= 42;

52'b00000000000000000000000000000000000000000001????????: divisor\_shift <= 43;

52'b000000000000000000000000000000000000000000001???????: divisor\_shift <= 44;

52'b0000000000000000000000000000000000000000000001??????: divisor\_shift <= 45;

52'b00000000000000000000000000000000000000000000001?????: divisor\_shift <= 46;

52'b000000000000000000000000000000000000000000000001????: divisor\_shift <= 47;

52'b0000000000000000000000000000000000000000000000001???: divisor\_shift <= 48;

52'b00000000000000000000000000000000000000000000000001??: divisor\_shift <= 49;

52'b000000000000000000000000000000000000000000000000001?: divisor\_shift <= 50;

52'b0000000000000000000000000000000000000000000000000001: divisor\_shift <= 51;

52'b0000000000000000000000000000000000000000000000000000: divisor\_shift <= 52;

endcase

endmodule

module fpu\_sqroot(clk,rst,enable,select,opa,opb,sign,exponent\_root,mantissa\_root);

input clk;

input rst;

input enable;

input select;

input [63:0]opa,opb;

output sign;

output [11:0]exponent\_root;

output [55:0]mantissa\_root;

reg sign;

reg[11:0]exponent\_root;

reg[55:0]mantissa\_root;

integer i;

reg fs,eo;

reg[10:0]ea;

reg[11:0]ec;

reg[11:0]even,odd,shift;

reg[51:0]mantissa;

reg[52:0]ma;

reg[54:0]mr;

reg[56:0]x,t,m;

reg[55:0]r;

always@(enable or select or opa or opb)

begin

if(enable==1'b1)

begin

if (select==1'b1)

begin

sign=opa[63];

ea=opa[62:52];

mantissa=opa[51:0];

end

else

begin

sign=opb[63];

ea=opb[62:52];

mantissa=opb[51:0];

end

end

else

begin

sign=1'b0;

ea=11'b0;

mantissa=52'b0;

end

end

always @(eo or ma or ea)

begin

eo=ea%(11'b00000000010);

if(eo==1'b0)

begin

even=ea+11'b01111111110;

shift= even>>1;

ec={1'b0,shift[10:0]};

exponent\_root=ec;

fs=1'b1;

end

else

begin

odd=ea+11'b01111111111;

shift= odd>>1;

ec={1'b0,shift[10:0]};

exponent\_root=ec;

fs=1'b0;

end

end

always@(posedge clk)

begin

if(enable==1'b1)

begin

if (rst==1'b1)

begin

t=57'b010000000000000000000000000000000000000000000000000000000;

mr=55'b0;

i=0;

end

else

begin

if (i==0)

begin

if(fs==1'b1)

begin

ma={1'b1,mantissa};

ma=ma<<1;

x=ma<<1;

i=i+1;

end

else

begin

ma={1'b1,mantissa};

ma=ma<<1;

x=ma;

i=i+1;

end

end

else if(i==1)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[55:0]=t[55:0]<<1;

r[55]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[55:0]=t[55:0]>>1;

r[55]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==2)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[54:0]=t[54:0]<<1;

r[54]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[54:0]=t[54:0]>>1;

r[54]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==3)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[53:0]=t[53:0]<<1;

r[53]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[53:0]=t[53:0]>>1;

r[53]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==4)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[52:0]=t[52:0]<<1;

r[52]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[52:0]=t[52:0]>>1;

r[52]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==5)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[51:0]=t[51:0]<<1;

r[51]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[51:0]=t[51:0]>>1;

r[51]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==6)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[50:0]=t[50:0]<<1;

r[50]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[50:0]=t[50:0]>>1;

r[50]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==7)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[49:0]=t[49:0]<<1;

r[49]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[49:0]=t[49:0]>>1;

r[49]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==8)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[48:0]=t[48:0]<<1;

r[48]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[48:0]=t[48:0]>>1;

r[48]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==9)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[47:0]=t[47:0]<<1;

r[47]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[47:0]=t[47:0]>>1;

r[47]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==10)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[46:0]=t[46:0]<<1;

r[46]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[46:0]=t[46:0]>>1;

r[46]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==11)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[45:0]=t[45:0]<<1;

r[45]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[45:0]=t[45:0]>>1;

r[45]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==12)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[44:0]=t[44:0]<<1;

r[44]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[44:0]=t[44:0]>>1;

r[44]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==13)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[43:0]=t[43:0]<<1;

r[43]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[43:0]=t[43:0]>>1;

r[43]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==14)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[42:0]=t[42:0]<<1;

r[42]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[42:0]=t[42:0]>>1;

r[42]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==15)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[41:0]=t[41:0]<<1;

r[41]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[41:0]=t[41:0]>>1;

r[41]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==16)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[40:0]=t[40:0]<<1;

r[40]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[40:0]=t[40:0]>>1;

r[40]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==17)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[39:0]=t[39:0]<<1;

r[39]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[39:0]=t[39:0]>>1;

r[39]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==18)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[38:0]=t[38:0]<<1;

r[38]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[38:0]=t[38:0]>>1;

r[38]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==19)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[37:0]=t[37:0]<<1;

r[37]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[37:0]=t[37:0]>>1;

r[37]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==20)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[36:0]=t[36:0]<<1;

r[36]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[36:0]=t[36:0]>>1;

r[36]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==21)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[35:0]=t[35:0]<<1;

r[35]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[35:0]=t[35:0]>>1;

r[35]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==22)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[34:0]=t[34:0]<<1;

r[34]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[34:0]=t[34:0]>>1;

r[34]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==23)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[33:0]=t[33:0]<<1;

r[33]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[33:0]=t[33:0]>>1;

r[33]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==24)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[32:0]=t[32:0]<<1;

r[32]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[32:0]=t[32:0]>>1;

r[32]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==25)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[31:0]=t[31:0]<<1;

r[31]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[31:0]=t[31:0]>>1;

r[31]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==26)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[30:0]=t[30:0]<<1;

r[30]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[30:0]=t[30:0]>>1;

r[30]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==27)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[29:0]=t[29:0]<<1;

r[29]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[29:0]=t[29:0]>>1;

r[29]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==28)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[28:0]=t[28:0]<<1;

r[28]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[28:0]=t[28:0]>>1;

r[28]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==29)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[27:0]=t[27:0]<<1;

r[27]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[27:0]=t[27:0]>>1;

r[27]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==30)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[26:0]=t[26:0]<<1;

r[26]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[26:0]=t[26:0]>>1;

r[26]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==31)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[25:0]=t[25:0]<<1;

r[25]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[25:0]=t[25:0]>>1;

r[25]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==32)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[24:0]=t[24:0]<<1;

r[24]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[24:0]=t[24:0]>>1;

r[24]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==33)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[23:0]=t[23:0]<<1;

r[23]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[23:0]=t[23:0]>>1;

r[23]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==34)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[22:0]=t[22:0]<<1;

r[22]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[22:0]=t[22:0]>>1;

r[22]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==35)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[21:0]=t[21:0]<<1;

r[21]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[21:0]=t[21:0]>>1;

r[21]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==36)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[20:0]=t[20:0]<<1;

r[20]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[20:0]=t[20:0]>>1;

r[20]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==37)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[19:0]=t[19:0]<<1;

r[19]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[19:0]=t[19:0]>>1;

r[19]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==38)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[18:0]=t[18:0]<<1;

r[18]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[18:0]=t[18:0]>>1;

r[18]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==39)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[17:0]=t[17:0]<<1;

r[17]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[17:0]=t[17:0]>>1;

r[17]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==40)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[16:0]=t[16:0]<<1;

r[16]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[16:0]=t[16:0]>>1;

r[16]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==41)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[15:0]=t[15:0]<<1;

r[15]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[15:0]=t[15:0]>>1;

r[15]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==42)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[14:0]=t[14:0]<<1;

r[14]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[14:0]=t[14:0]>>1;

r[14]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==43)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[13:0]=t[13:0]<<1;

r[13]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[13:0]=t[13:0]>>1;

r[13]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==44)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[12:0]=t[12:0]<<1;

r[12]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[12:0]=t[12:0]>>1;

r[12]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==45)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[11:0]=t[11:0]<<1;

r[11]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[11:0]=t[11:0]>>1;

r[11]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==46)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[10:0]=t[10:0]<<1;

r[10]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[10:0]=t[10:0]>>1;

r[10]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==47)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[9:0]=t[9:0]<<1;

r[9]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[9:0]=t[9:0]>>1;

r[9]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==48)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[8:0]=t[8:0]<<1;

r[8]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[8:0]=t[8:0]>>1;

r[8]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==49)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[7:0]=t[7:0]<<1;

r[7]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[7:0]=t[7:0]>>1;

r[7]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==50)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[6:0]=t[6:0]<<1;

r[6]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[6:0]=t[6:0]>>1;

r[6]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==51)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[5:0]=t[5:0]<<1;

r[5]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[5:0]=t[5:0]>>1;

r[5]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==52)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[4:0]=t[4:0]<<1;

r[4]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[4:0]=t[4:0]>>1;

r[4]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==53)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[3:0]=t[3:0]<<1;

r[3]=1'b1;

t={t[56],r};

i=i+1;

end

else

begin

x=x<<1;

r[3:0]=t[3:0]>>1;

r[3]=1'b0;

t={t[56],r};

i=i+1;

end

end

else if(i==54)

begin

if(x>=t)

begin

m=x-t;

x=m<<1;

r[2:0]=t[2:0]<<1;

r[2]=1'b1;

t={t[56],r};

mr=t[55:0];

mantissa\_root=mr;

i=i+1;

end

else

begin

x=x<<1;

r[2:0]=t[2:0]>>1;

r[2]=1'b0;

t={t[56],r};

mr=t[55:0];

mantissa\_root=mr;

i=i+1;

end

end

end

end

else

begin

mantissa\_root=56'b0;

end

end

endmodule

module fpu\_round( clk, rst, enable, round\_mode, sign\_term,

mantissa\_term, exponent\_term, round\_out, exponent\_final);

input clk;

input rst;

input enable;

input [1:0] round\_mode;

input sign\_term;

input [55:0] mantissa\_term;

input [11:0] exponent\_term;

output [63:0] round\_out;

output [11:0] exponent\_final;

wire [55:0] rounding\_amount = { 53'b0, 1'b1, 2'b0};

wire round\_nearest = (round\_mode == 2'b00);

wire round\_to\_zero = (round\_mode == 2'b01);

wire round\_to\_pos\_inf = (round\_mode == 2'b10);

wire round\_to\_neg\_inf = (round\_mode == 2'b11);

wire round\_nearest\_trigger = round\_nearest & mantissa\_term[1];

wire round\_to\_pos\_inf\_trigger = !sign\_term & |mantissa\_term[1:0];

wire round\_to\_neg\_inf\_trigger = sign\_term & |mantissa\_term[1:0];

wire round\_trigger = ( round\_nearest & round\_nearest\_trigger)

| (round\_to\_pos\_inf & round\_to\_pos\_inf\_trigger)

| (round\_to\_neg\_inf & round\_to\_neg\_inf\_trigger);

reg [55:0] sum\_round;

wire sum\_round\_overflow = sum\_round[55];

reg [55:0] sum\_round\_2;

reg [11:0] exponent\_round;

reg [55:0] sum\_final;

reg [11:0] exponent\_final;

reg [63:0] round\_out;

always @(posedge clk)

begin

if (rst) begin

sum\_round <= 0;

sum\_round\_2 <= 0;

exponent\_round <= 0;

sum\_final <= 0;

exponent\_final <= 0;

round\_out <= 0;

end

else begin

sum\_round <= rounding\_amount + mantissa\_term;

sum\_round\_2 <= sum\_round\_overflow ? sum\_round >> 1 : sum\_round;

exponent\_round <= sum\_round\_overflow ? (exponent\_term + 1) : exponent\_term;

sum\_final <= round\_trigger ? sum\_round\_2 : mantissa\_term;

exponent\_final <= round\_trigger ? exponent\_round : exponent\_term;

round\_out <= { sign\_term, exponent\_final[10:0], sum\_final[53:2] };

end

end

endmodule

module fpu\_exceptions( clk, rst, enable, rmode, opa, opb, in\_except,

exponent\_in, mantissa\_in, fpu\_op, out, ex\_enable, underflow, overflow,

inexact, exception, invalid);

input clk;

input rst;

input enable;

input [1:0] rmode;

input [63:0] opa;

input [63:0] opb;

input [63:0] in\_except;

input [11:0] exponent\_in;

input [1:0] mantissa\_in;

input [2:0] fpu\_op;

output [63:0] out;

output ex\_enable;

output underflow;

output overflow;

output inexact;

output exception;

output invalid;

reg [63:0] out;

reg ex\_enable;

reg underflow;

reg overflow;

reg inexact;

reg exception;

reg invalid;

reg in\_et\_zero;

reg opa\_et\_zero;

reg opb\_et\_zero;

reg input\_et\_zero;

reg add;

reg subtract;

reg multiply;

reg divide;

reg sqroot;

reg opa\_QNaN;

reg opb\_QNaN;

reg opa\_SNaN;

reg opb\_SNaN;

reg opa\_pos\_inf;

reg opb\_pos\_inf;

reg opa\_neg\_inf;

reg opb\_neg\_inf;

reg opa\_inf;

reg opb\_inf;

reg NaN\_input;

reg SNaN\_input;

reg a\_NaN;

reg div\_by\_0;

reg div\_0\_by\_0;

reg div\_inf\_by\_inf;

reg div\_by\_inf;

reg mul\_0\_by\_inf;

reg mul\_inf;

reg div\_inf;

reg add\_inf;

reg sub\_inf;

reg sqrt\_inf;

reg addsub\_inf\_invalid;

reg addsub\_inf;

reg out\_inf\_trigger;

reg out\_pos\_inf;

reg out\_neg\_inf;

reg round\_nearest;

reg round\_to\_zero;

reg round\_to\_pos\_inf;

reg round\_to\_neg\_inf;

reg inf\_round\_down\_trigger;

reg mul\_uf;

reg div\_uf;

reg sqrt\_uf;

reg underflow\_trigger;

reg invalid\_trigger;

reg overflow\_trigger;

reg inexact\_trigger;

reg except\_trigger;

reg enable\_trigger;

reg NaN\_out\_trigger;

reg SNaN\_trigger;

wire [10:0] exp\_2047 = 11'b11111111111;

wire [10:0] exp\_2046 = 11'b11111111110;

reg [62:0] NaN\_output\_0;

reg [62:0] NaN\_output;

wire [51:0] mantissa\_max = 52'b1111111111111111111111111111111111111111111111111111;

reg [62:0] inf\_round\_down;

reg [62:0] out\_inf;

reg [63:0] out\_0;

reg [63:0] out\_1;

reg [63:0] out\_2;

always @(posedge clk)

begin

if (rst) begin

in\_et\_zero <= 0;

opa\_et\_zero <= 0;

opb\_et\_zero <= 0;

input\_et\_zero <= 0;

add <= 0;

subtract <= 0;

multiply <= 0;

divide <= 0;

sqroot <=0;

opa\_QNaN <= 0;

opb\_QNaN <= 0;

opa\_SNaN <= 0;

opb\_SNaN <= 0;

opa\_pos\_inf <= 0;

opb\_pos\_inf <= 0;

opa\_neg\_inf <= 0;

opb\_neg\_inf <= 0;

opa\_inf <= 0;

opb\_inf <= 0;

NaN\_input <= 0;

SNaN\_input <= 0;

a\_NaN <= 0;

div\_by\_0 <= 0;

div\_0\_by\_0 <= 0;

div\_inf\_by\_inf <= 0;

div\_by\_inf <= 0;

mul\_0\_by\_inf <= 0;

mul\_inf <= 0;

div\_inf <= 0;

add\_inf <= 0;

sub\_inf <= 0;

sqrt\_inf <= 0;

addsub\_inf\_invalid <= 0;

addsub\_inf <= 0;

out\_inf\_trigger <= 0;

out\_pos\_inf <= 0;

out\_neg\_inf <= 0;

round\_nearest <= 0;

round\_to\_zero <= 0;

round\_to\_pos\_inf <= 0;

round\_to\_neg\_inf <= 0;

inf\_round\_down\_trigger <= 0;

mul\_uf <= 0;

div\_uf <= 0;

sqrt\_uf <= 0;

underflow\_trigger <= 0;

invalid\_trigger <= 0;

overflow\_trigger <= 0;

inexact\_trigger <= 0;

except\_trigger <= 0;

enable\_trigger <= 0;

NaN\_out\_trigger <= 0;

SNaN\_trigger <= 0;

NaN\_output\_0 <= 0;

NaN\_output <= 0;

inf\_round\_down <= 0;

out\_inf <= 0;

out\_0 <= 0;

out\_1 <= 0;

out\_2 <= 0;

end

else if (enable) begin

in\_et\_zero <= !(|in\_except[62:0]);

opa\_et\_zero <= !(|opa[62:0]);

opb\_et\_zero <= !(|opb[62:0]);

input\_et\_zero <= !(|in\_except[62:0]);

add <= fpu\_op == 3'b000;

subtract <= fpu\_op == 3'b001;

multiply <= fpu\_op == 3'b010;

divide <= fpu\_op == 3'b011;

sqroot <= fpu\_op == 3'b100;

opa\_QNaN <= (opa[62:52] == 2047) & |opa[51:0] & opa[51];

opb\_QNaN <= (opb[62:52] == 2047) & |opb[51:0] & opb[51];

opa\_SNaN <= (opa[62:52] == 2047) & |opa[51:0] & !opa[51];

opb\_SNaN <= (opb[62:52] == 2047) & |opb[51:0] & !opb[51];

opa\_pos\_inf <= !opa[63] & (opa[62:52] == 2047) & !(|opa[51:0]);

opb\_pos\_inf <= !opb[63] & (opb[62:52] == 2047) & !(|opb[51:0]);

opa\_neg\_inf <= opa[63] & (opa[62:52] == 2047) & !(|opa[51:0]);

opb\_neg\_inf <= opb[63] & (opb[62:52] == 2047) & !(|opb[51:0]);

opa\_inf <= (opa[62:52] == 2047) & !(|opa[51:0]);

opb\_inf <= (opb[62:52] == 2047) & !(|opb[51:0]);

NaN\_input <= opa\_QNaN | opb\_QNaN | opa\_SNaN | opb\_SNaN;

SNaN\_input <= opa\_SNaN | opb\_SNaN;

a\_NaN <= opa\_QNaN | opa\_SNaN;

div\_by\_0 <= divide & opb\_et\_zero & !opa\_et\_zero;

div\_0\_by\_0 <= divide & opb\_et\_zero & opa\_et\_zero;

div\_inf\_by\_inf <= divide & opa\_inf & opb\_inf;

div\_by\_inf <= divide & !opa\_inf & opb\_inf;

mul\_0\_by\_inf <= multiply & ((opa\_inf & opb\_et\_zero) | (opa\_et\_zero & opb\_inf));

mul\_inf <= multiply & (opa\_inf | opb\_inf) & !mul\_0\_by\_inf;

div\_inf <= divide & opa\_inf & !opb\_inf;

add\_inf <= (add & (opa\_inf | opb\_inf));

sub\_inf <= (subtract & (opa\_inf | opb\_inf));

sqrt\_inf<= (sqroot &(opa\_inf | opb\_inf));

addsub\_inf\_invalid <= (add & opa\_pos\_inf & opb\_neg\_inf) | (add & opa\_neg\_inf & opb\_pos\_inf) |

(subtract & opa\_pos\_inf & opb\_pos\_inf) | (subtract & opa\_neg\_inf & opb\_neg\_inf);

addsub\_inf <= (add\_inf | sub\_inf) & !addsub\_inf\_invalid;

out\_inf\_trigger <= addsub\_inf | mul\_inf | div\_inf | div\_by\_0 | sqrt\_inf | (exponent\_in > 2046);

out\_pos\_inf <= out\_inf\_trigger & !in\_except[63];

out\_neg\_inf <= out\_inf\_trigger & in\_except[63];

round\_nearest <= (rmode == 2'b00);

round\_to\_zero <= (rmode == 2'b01);

round\_to\_pos\_inf <= (rmode == 2'b10);

round\_to\_neg\_inf <= (rmode == 2'b11);

inf\_round\_down\_trigger <= (out\_pos\_inf & round\_to\_neg\_inf) |

(out\_neg\_inf & round\_to\_pos\_inf) |

(out\_inf\_trigger & round\_to\_zero);

mul\_uf <= multiply & !opa\_et\_zero & !opb\_et\_zero & in\_et\_zero;

div\_uf <= divide & !opa\_et\_zero & in\_et\_zero;

sqrt\_uf <= sqroot & ((!opa\_et\_zero) | (!opb\_et\_zero)) & in\_et\_zero;

underflow\_trigger <= div\_by\_inf | mul\_uf | div\_uf |sqrt\_uf;

invalid\_trigger <= SNaN\_input | addsub\_inf\_invalid | mul\_0\_by\_inf |

div\_0\_by\_0 | div\_inf\_by\_inf | sqrt\_inf;

overflow\_trigger <= out\_inf\_trigger & !NaN\_input;

inexact\_trigger <= (|mantissa\_in[1:0] | out\_inf\_trigger | underflow\_trigger) &

!NaN\_input;

except\_trigger <= invalid\_trigger | overflow\_trigger | underflow\_trigger |

inexact\_trigger;

enable\_trigger <= except\_trigger | out\_inf\_trigger | NaN\_input;

NaN\_out\_trigger <= NaN\_input | invalid\_trigger;

SNaN\_trigger <= invalid\_trigger & !SNaN\_input;

NaN\_output\_0 <= a\_NaN ? { exp\_2047, 1'b1, opa[50:0]} : { exp\_2047, 1'b1, opb[50:0]};

NaN\_output <= SNaN\_trigger ? { exp\_2047, 2'b01, opa[49:0]} : NaN\_output\_0;

inf\_round\_down <= { exp\_2046, mantissa\_max };

out\_inf <= inf\_round\_down\_trigger ? inf\_round\_down : { exp\_2047, 52'b0 };

out\_0 <= underflow\_trigger ? { in\_except[63], 63'b0 } : in\_except;

out\_1 <= out\_inf\_trigger ? { in\_except[63], out\_inf } : out\_0;

out\_2 <= NaN\_out\_trigger ? { in\_except[63], NaN\_output} : out\_1;

end

end

always @(posedge clk)

begin

if (rst) begin

ex\_enable <= 0;

underflow <= 0;

overflow <= 0;

inexact <= 0;

exception <= 0;

invalid <= 0;

out <= 0;

end

else if (enable) begin

ex\_enable <= enable\_trigger;

underflow <= underflow\_trigger;

overflow <= overflow\_trigger;

inexact <= inexact\_trigger;

exception <= except\_trigger;

invalid <= invalid\_trigger;

out <= out\_2;

end

end

endmodule