module sigmoid(In,Out,Sign);

input [19:0]In;

input Sign;

output [19:0]Out;

wire [19:0]out\_temp;

wire [19:0] LA\_tmp,TA\_tmp;

Linear\_Approx LA1( In,LA\_tmp);

Taylor\_Approx TA1( In,TA\_tmp);

NR\_Approx NR1(LA\_tmp,TA\_tmp,out\_temp);

assign Out = Sign?20'd65536-out\_temp:out\_temp;

endmodule

module Linear\_Approx(LA\_In ,LA\_Out);

input [19:0] LA\_In;

output reg [19:0] LA\_Out;

always@(\*)

begin

if (LA\_In>=20'd327680) // if in > 5

LA\_Out=20'd65536; // out = 1

else if (LA\_In<20'd327680 && LA\_In>=20'd155648) // if in < 5 and in> 2.375

LA\_Out=(LA\_In>>5)+20'd55296; //out = 0.03125\*In + 0.84375

else if (LA\_In<20'd155648 && LA\_In>=20'd65536) // if in < 2.375 and in > 1

LA\_Out=(LA\_In>>3)+ 20'd40960; //out = 0.125 \*In + 0.625

else if (LA\_In<20'd65536 && LA\_In>=0) //if in < 1 and in > 0

LA\_Out=(LA\_In>>2)+20'd32768; // out = 0.25 \*In + 0.5

else

LA\_Out=20'd0;

end

endmodule

module Taylor\_Approx(TA\_In,TA\_Out);

input [19:0]TA\_In;

output [19:0]TA\_Out;

wire [39:0] Ln2,y,r\_final;

wire [3:0]M,J;

wire [11:0]r\_dummy;

wire [19:0]Acc,LUT\_Out,R;

assign Ln2 = 20'd94548\* TA\_In; // X/ln2 ~ X \* 1.442

assign M=Ln2[35:32]; // M = 4 bits left to the decimal

assign J=Ln2[31:28]; // J = L bits left to M (here L = 4)

LUT\_Def Lt(J,LUT\_Out); // 2^(-j/16)

assign r\_dummy=Ln2[27:16]; // R\_dummy = 12 - bits to the left of J

assign r\_final=20'd45408\*{8'd0,r\_dummy}; // R\_final = Ln2 \* 20 bit R\_dummy

assign R=r\_final [35:16]; // R = 20 bits from MSB side of R\_final

assign Acc=LUT\_Out>>M; // 2^-(M +(j/16)) -- shifting by M

assign y=Acc\*(20'd65536-R); // (2^-(M +(j/16))) \* (1-R)

assign TA\_Out=y[35:16]; // slice 20 bits from MSB

endmodule

module LUT\_Def(key,value);

input [3:0]key;

output reg [19:0]value;

always@(key)

begin

case(key)

4'd0: value=20'd65536; //1

4'd1: value=20'd62757; //0.957

4'd2: value=20'd60096; //0.917

4'd3 : value=20'd57548; //0.878

4'd4: value=20'd55103; //0.840

4'd5: value=20'd52772; //0.805

4'd6: value=20'd50535; //0.771

4'd7: value=20'd48392; //0.738

4'd8: value=20'd46340; //0.707

4'd9: value=20'd44376; //0.677

4'd10: value=20'd42494; //0.648

4'd11: value=20'd40693; //0.620

4'd12: value=20'd38967; //0.594

4'd13: value=20'd37315; //0.569

4'd14: value=20'd35733; //0.545

4'd15: value=20'd34218; //0.522

default: value=20'd0;

endcase

end

endmodule

module NR\_Approx( Y\_0,TA,Res);

input [19:0] Y\_0,TA;

output [19:0] Res;

wire [19:0] sigma;

wire [39:0] sigma\_square;

wire [59:0] res\_temp,temp;

assign sigma = Y\_0<<1; // 2\* y0

assign sigma\_square = Y\_0 \* Y\_0; // y0^2

assign temp = (20'd65536+TA)\*sigma\_square; // ((y0^2)\*(1+e^(-x)))

assign res\_temp = {8'd0,sigma,32'd0} - temp; // ((2\*Y0) - ((y0^2)\*(1+e^(-x))))

assign Res = res\_temp[51:32]; // slice 20bit MSB

endmodule

module sigm\_tb();

reg [19:0]x ;

reg sign;

wire [19:0]y ;

sigmoid s(x,y,sign) ;

initial

begin

sign = 1'b0;

x=20'd000000;

#50

x=20'd183500;

#50

x=20'd229376;

#50

x=20'd401198;

#50

sign = 1'b1;

x=20'd8192;

#50

x=20'd49152;

#50

x=20'd108134;

#100

$finish;

end

endmodule