**HW8**

**-Daksh Thapar 2018137**

**-Tanmaya Gupta 2018200**

module top

#(

parameter integer C\_S\_AXIS\_X\_TDATA\_WIDTH = 32,

parameter integer C\_S\_AXIS\_T\_TDATA\_WIDTH = 32,

parameter integer C\_S\_AXIS\_N\_TDATA\_WIDTH = 32

)

(

input s\_aclk,

input s\_aresetn,

input [C\_S\_AXIS\_X\_TDATA\_WIDTH-1:0] s\_x\_tdata,

input s\_x\_tvalid,

input [C\_S\_AXIS\_X\_TDATA\_WIDTH-1:0] s\_alpha\_tdata,

input s\_alpha\_tvalid,

input [C\_S\_AXIS\_T\_TDATA\_WIDTH-1:0] s\_t\_tdata,

input s\_t\_tvalid,

input [C\_S\_AXIS\_N\_TDATA\_WIDTH-1:0] s\_n\_tdata,

input s\_n\_tvalid,

output s\_x\_tready,

output s\_t\_tready,

output s\_n\_tready,

output s\_alpha\_tready,

output final\_q\_valid,

output [31:0] final\_q

);

//1

wire float\_x\_valid, float\_x\_ready;

wire [31:0] float\_x;

floating\_point\_0 fx(

.aclk(s\_aclk),

.aresetn(s\_aresetn),

.s\_axis\_a\_tvalid(s\_x\_tvalid),

.s\_axis\_a\_tready(s\_x\_tready),

.s\_axis\_a\_tdata(s\_x\_tdata),

.m\_axis\_result\_tvalid(float\_x\_valid),

.m\_axis\_result\_tready(float\_x\_ready),

.m\_axis\_result\_tdata(float\_x)

);

//2

wire float\_t\_valid, float\_t\_ready;

wire [31:0] float\_t;

floating\_point\_0 ft(

.aclk(s\_aclk),

.aresetn(s\_aresetn),

.s\_axis\_a\_tvalid(s\_t\_tvalid),

.s\_axis\_a\_tready(s\_t\_tready),

.s\_axis\_a\_tdata(s\_t\_tdata),

.m\_axis\_result\_tvalid(float\_t\_valid),

.m\_axis\_result\_tready(float\_t\_ready),

.m\_axis\_result\_tdata(float\_t)

);

//3

wire float\_n\_valid, float\_n\_ready;

wire [31:0] float\_n;

floating\_point\_0 fn(

.aclk(s\_aclk),

.aresetn(s\_aresetn),

.s\_axis\_a\_tvalid(s\_n\_tvalid),

.s\_axis\_a\_tready(s\_n\_tready),

.s\_axis\_a\_tdata(s\_n\_tdata),

.m\_axis\_result\_tvalid(float\_n\_valid),

.m\_axis\_result\_tready(float\_n\_ready),

.m\_axis\_result\_tdata(float\_n)

);

//4

wire float\_alpha\_valid, float\_alpha\_ready;

wire [31:0] float\_alpha;

floating\_point\_0 falpha(

.aclk(s\_aclk),

.aresetn(s\_aresetn),

.s\_axis\_a\_tvalid(s\_alpha\_tvalid),

.s\_axis\_a\_tready(s\_alpha\_tready),

.s\_axis\_a\_tdata(s\_alpha\_tdata),

.m\_axis\_result\_tvalid(float\_alpha\_valid),

.m\_axis\_result\_tready(float\_alpha\_ready),

.m\_axis\_result\_tdata(float\_alpha)

);

//5

wire div1\_valid,div1\_ready;

wire [31:0] div1\_data;

divide x\_by\_t(

.aclk(s\_aclk),

.aresetn(s\_aresetn),

.s\_axis\_a\_tvalid(float\_x\_valid),

.s\_axis\_a\_tready(float\_x\_ready),

.s\_axis\_a\_tdata(float\_x),

.s\_axis\_b\_tvalid(float\_t\_valid),

.s\_axis\_b\_tready(float\_t\_ready),

.s\_axis\_b\_tdata(float\_t),

.m\_axis\_result\_tvalid(div1\_valid),

.m\_axis\_result\_tready(div1\_ready),

.m\_axis\_result\_tdata(div1\_data)

);

//8

wire log1\_valid, log1\_ready;

wire [31:0] log1\_data;

logarithm log\_n(

.aclk(s\_aclk),

.aresetn(s\_aresetn),

.s\_axis\_a\_tvalid(float\_n\_valid),

.s\_axis\_a\_tready(float\_n\_ready),

.s\_axis\_a\_tdata(float\_n),

.m\_axis\_result\_tvalid(log1\_valid),

.m\_axis\_result\_tready(log1\_ready),

.m\_axis\_result\_tdata(log1\_data)

);

//14

wire mul3\_valid, mul3\_ready;

wire [31:0] mul3\_data;

multiply alpha\_into\_lnn(

.aclk(s\_aclk),

.aresetn(s\_aresetn),

.s\_axis\_a\_tvalid(float\_alpha\_valid),

.s\_axis\_a\_tready(float\_alpha\_ready),

.s\_axis\_a\_tdata(float\_alpha),

.s\_axis\_b\_tvalid(log1\_valid),

.s\_axis\_b\_tready(log1\_ready),

.s\_axis\_b\_tdata(log1\_data),

.m\_axis\_result\_tvalid(mul3\_valid),

.m\_axis\_result\_tready(mul3\_ready),

.m\_axis\_result\_tdata(mul3\_data)

);

//7

wire div3\_valid,div3\_ready;

wire [31:0] div3\_data;

divide two\_lnn\_by\_t(

.aclk(s\_aclk),

.aresetn(s\_aresetn),

.s\_axis\_a\_tvalid(mul3\_valid),

.s\_axis\_a\_tready(mul3\_ready),

.s\_axis\_a\_tdata(mul3\_data),

.s\_axis\_b\_tvalid(float\_t\_valid),

// .s\_axis\_b\_tready(float\_t\_ready),

.s\_axis\_b\_tdata(float\_t),

.m\_axis\_result\_tvalid(div3\_valid),

.m\_axis\_result\_tready(div3\_ready),

.m\_axis\_result\_tdata(div3\_data)

);

//12

wire mul1\_valid, mul1\_ready;

wire [31:0] mul1\_data;

multiply m\_into\_m(

.aclk(s\_aclk),

.aresetn(s\_aresetn),

.s\_axis\_a\_tvalid(div1\_valid),

.s\_axis\_a\_tready(div1\_ready),

.s\_axis\_a\_tdata(div1\_data),

.s\_axis\_b\_tvalid(div1\_valid),

// .s\_axis\_b\_tready(div1\_ready),

.s\_axis\_b\_tdata(div1\_data),

.m\_axis\_result\_tvalid(mul1\_valid),

.m\_axis\_result\_tready(mul1\_ready),

.m\_axis\_result\_tdata(mul1\_data)

);

//9

wire sub1\_valid,sub1\_ready;

wire [31:0] sub1\_data;

subtract m\_minus\_msquare(

.aclk(s\_aclk),

.aresetn(s\_aresetn),

.s\_axis\_a\_tvalid(div1\_valid),

// .s\_axis\_a\_tready(div1\_ready),

.s\_axis\_a\_tdata(div1\_data),

.s\_axis\_b\_tvalid(mul1\_valid),

.s\_axis\_b\_tready(mul1\_ready),

.s\_axis\_b\_tdata(mul1\_data),

.m\_axis\_result\_tvalid(sub1\_valid),

.m\_axis\_result\_tready(sub1\_ready),

.m\_axis\_result\_tdata(sub1\_data)

);

//16

wire sr2\_valid, sr2\_ready;

wire [31:0] sr2\_data;

sqr\_root two\_ln\_n\_by\_t(

.aclk(s\_aclk),

.aresetn(s\_aresetn),

.s\_axis\_a\_tvalid(div3\_valid),

.s\_axis\_a\_tready(div3\_ready),

.s\_axis\_a\_tdata(div3\_data),

.m\_axis\_result\_tvalid(sr2\_valid),

.m\_axis\_result\_tready(sr2\_ready),

.m\_axis\_result\_tdata(sr2\_data)

);

//11

wire add2\_valid, add2\_ready;

wire [31:0] add2\_data;

add minus\_msquare\_plus\_blah(

.aclk(s\_aclk),

.aresetn(s\_aresetn),

.s\_axis\_a\_tvalid(sub1\_valid),

.s\_axis\_a\_tready(sub1\_ready),

.s\_axis\_a\_tdata(sub1\_data),

.s\_axis\_b\_tvalid(sr2\_valid),

.s\_axis\_b\_tready(sr2\_ready),

.s\_axis\_b\_tdata(sr2\_data),

.m\_axis\_result\_tvalid(add2\_valid),

.m\_axis\_result\_tready(add2\_ready),

.m\_axis\_result\_tdata(add2\_data)

);

//13

wire mul2\_valid, mul2\_ready;

wire [31:0] mul2\_data;

multiply blah\_into\_t(

.aclk(s\_aclk),

.aresetn(s\_aresetn),

.s\_axis\_a\_tvalid(add2\_valid),

.s\_axis\_a\_tready(add2\_ready),

.s\_axis\_a\_tdata(add2\_data),

.s\_axis\_b\_tvalid(float\_t\_valid),

// .s\_axis\_b\_tready(float\_t\_ready),

.s\_axis\_b\_tdata(float\_t),

.m\_axis\_result\_tvalid(mul2\_valid),

.m\_axis\_result\_tready(mul2\_ready),

.m\_axis\_result\_tdata(mul2\_data)

);

//6

wire div2\_valid,div2\_ready;

wire [31:0] div2\_data;

divide lnn\_by\_blah(

.aclk(s\_aclk),

.aresetn(s\_aresetn),

.s\_axis\_a\_tvalid(log1\_valid),

// .s\_axis\_a\_tready(log1\_ready),

.s\_axis\_a\_tdata(log1\_data),

.s\_axis\_b\_tvalid(mul2\_valid),

.s\_axis\_b\_tready(mul2\_ready),

.s\_axis\_b\_tdata(mul2\_data),

.m\_axis\_result\_tvalid(div2\_valid),

.m\_axis\_result\_tready(div2\_ready),

.m\_axis\_result\_tdata(div2\_data)

);

//15

wire sr1\_valid, sr1\_ready;

wire [31:0] sr1\_data;

sqr\_root blah(

.aclk(s\_aclk),

.aresetn(s\_aresetn),

.s\_axis\_a\_tvalid(div2\_valid),

.s\_axis\_a\_tready(div2\_ready),

.s\_axis\_a\_tdata(div2\_data),

.m\_axis\_result\_tvalid(sr1\_valid),

.m\_axis\_result\_tready(sr1\_ready),

.m\_axis\_result\_tdata(sr1\_data)

);

//10

add m\_plus\_blah(

.aclk(s\_aclk),

.aresetn(s\_aresetn),

.s\_axis\_a\_tvalid(div1\_valid),

// .s\_axis\_a\_tready(div1\_ready),

.s\_axis\_a\_tdata(div1\_data),

.s\_axis\_b\_tvalid(sr1\_valid),

.s\_axis\_b\_tready(sr1\_ready),

.s\_axis\_b\_tdata(sr1\_data),

.m\_axis\_result\_tvalid(final\_q\_valid),

.m\_axis\_result\_tready(1),

.m\_axis\_result\_tdata(final\_q)

);

endmodule

module testbench(

);

reg s\_aclk = 0;

always #5 s\_aclk = ~ s\_aclk;

reg s\_aresetn = 0;

reg [31:0] x\_tdata;

reg x\_tvalid;

reg [31:0] t\_tdata;

reg t\_tvalid;

reg [31:0] n\_tdata;

reg n\_tvalid;

reg [31:0] alpha\_tdata;

reg alpha\_tvalid;

wire final\_q\_valid;

wire [31:0] final\_q;

top

#(

.C\_S\_AXIS\_X\_TDATA\_WIDTH(32),

.C\_S\_AXIS\_T\_TDATA\_WIDTH(32),

.C\_S\_AXIS\_N\_TDATA\_WIDTH(32)

)

t1(

.final\_q\_valid(final\_q\_valid),

.final\_q(final\_q),

.s\_aclk(s\_aclk),

.s\_aresetn(s\_aresetn),

.s\_x\_tvalid(x\_tvalid),

.s\_x\_tdata(x\_tdata),

.s\_t\_tvalid(t\_tvalid),

.s\_t\_tdata(t\_tdata),

.s\_n\_tvalid(n\_tvalid),

.s\_n\_tdata(n\_tdata),

.s\_alpha\_tvalid(alpha\_tvalid),

.s\_alpha\_tdata(alpha\_tdata)

);

initial

begin

#0 s\_aresetn=0;

x\_tdata = 32'd0;

t\_tdata = 32'd0;

n\_tdata = 32'd0;

alpha\_tdata = 32'd0;

#100 s\_aresetn = 1;

#10 x\_tdata = 32'd4;

t\_tdata = 32'd2;

n\_tdata = 32'd100;

alpha\_tdata = 32'd2;

#4 x\_tvalid = 1;

t\_tvalid = 1;

n\_tvalid = 1;

alpha\_tvalid = 1;

#4 x\_tvalid = 0;

t\_tvalid = 0;

n\_tvalid = 0;

alpha\_tvalid = 0;

end

endmodule

**My TestBench-**

X=4

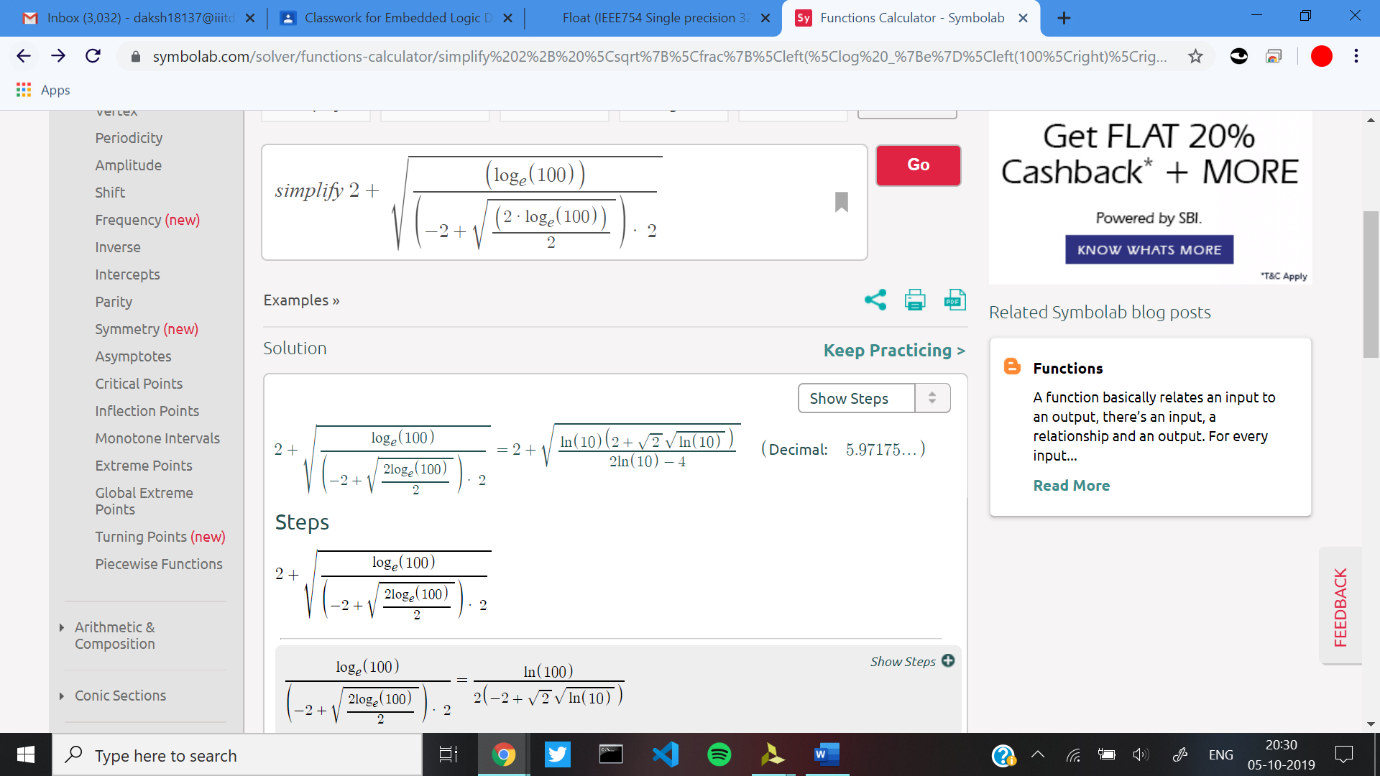
T=2

N=100

Alpha=2

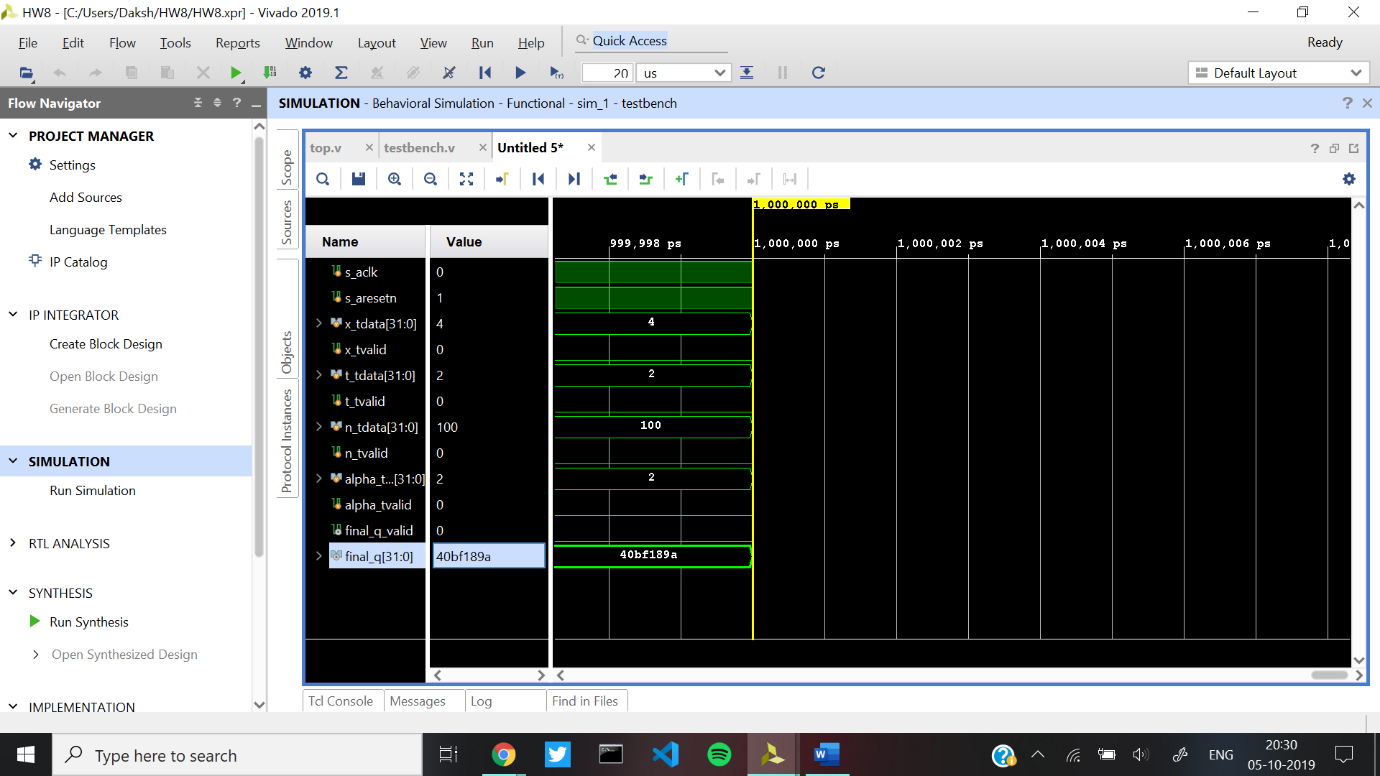
Calculated correct value-

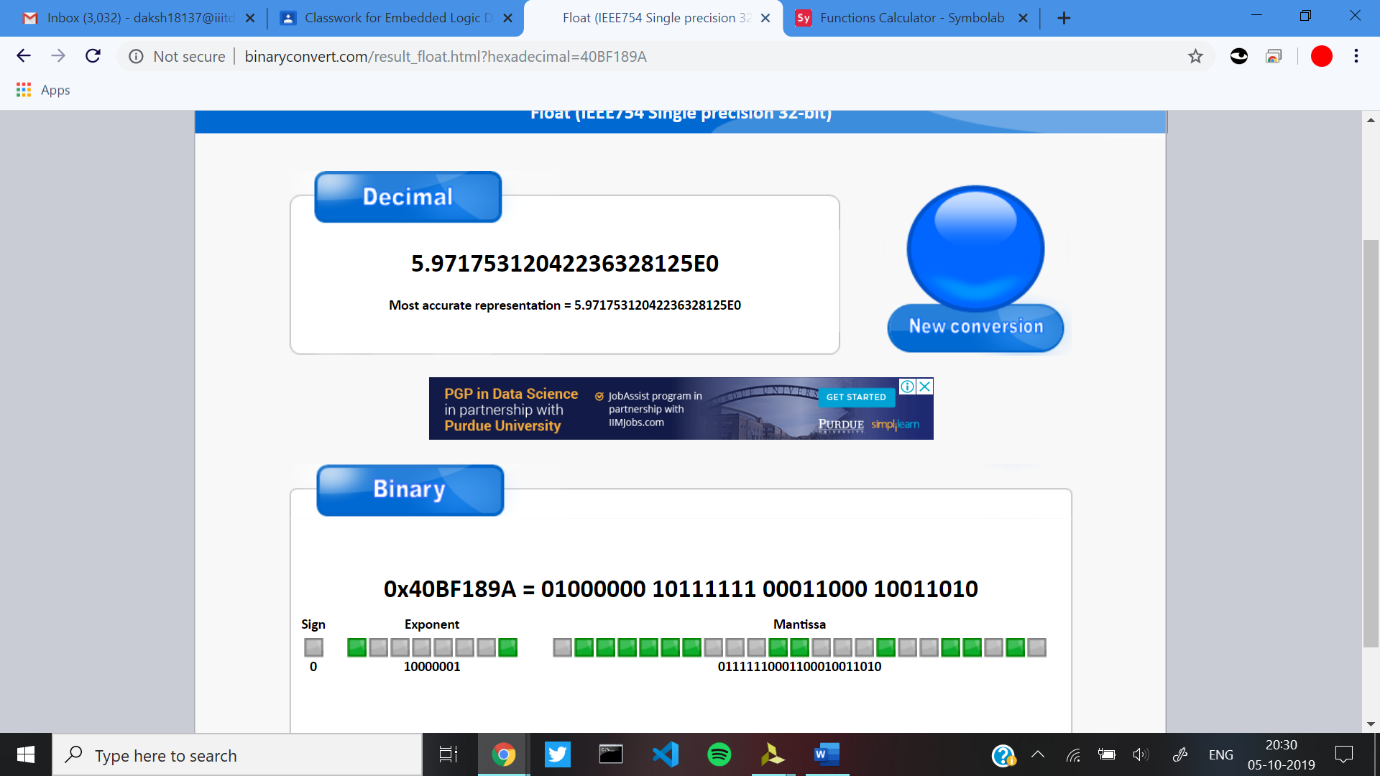
**=5.97175…**



Observed Value-

**=5.97175312042236328125**

****

****

Elaborated Design-

­­