University of Tehran
College of Engineering
School of Electrical and Computer Engineering

Digital Systems 1

Dr.Navabi

Computer Assignment 6

Soroush Mesforush Mashhad

SN:810198472

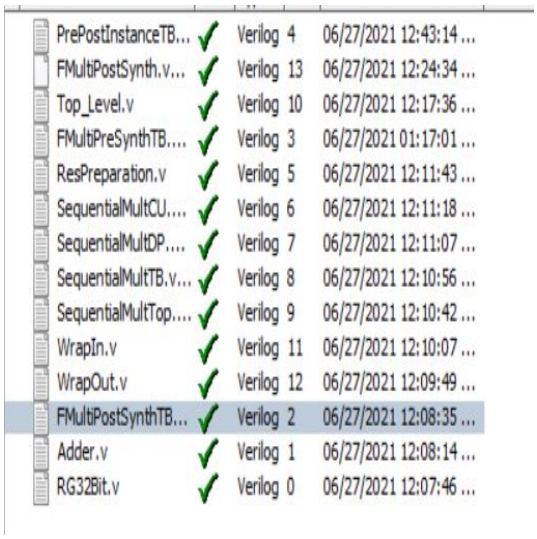
Tir 00

Abstract

In this computer assignment, we shall do our best to create a 32-bit floating point multiplier, I have done this using wrappers, a sequential multiplier and a combinational adder, due to Quartus II malfunctions I was forced to complete the project using Verilog files instead of SystemVerilog.

Project Generated Files(n)

A full view of the generated and used SystemVerilog files in my project can be seen below:



PrePostInstanceTB...	✓	Verilog 4	06/27/2021 12:43:14 ...
FMultiPostSynth.v...	✓	Verilog 13	06/27/2021 12:24:34 ...
Top_Level.v	✓	Verilog 10	06/27/2021 12:17:36 ...
FMultiPreSynthTB....	✓	Verilog 3	06/27/2021 01:17:01 ...
ResPreparation.v	✓	Verilog 5	06/27/2021 12:11:43 ...
SequentialMultCU....	✓	Verilog 6	06/27/2021 12:11:18 ...
SequentialMultDP....	✓	Verilog 7	06/27/2021 12:11:07 ...
SequentialMultTB.v...	✓	Verilog 8	06/27/2021 12:10:56 ...
SequentialMultTop....	✓	Verilog 9	06/27/2021 12:10:42 ...
WrapIn.v	✓	Verilog 11	06/27/2021 12:10:07 ...
WrapOut.v	✓	Verilog 12	06/27/2021 12:09:49 ...
FMultiPostSynthTB...	✓	Verilog 2	06/27/2021 12:08:35 ...
Adder.v	✓	Verilog 1	06/27/2021 12:08:14 ...
RG32Bit.v	✓	Verilog 0	06/27/2021 12:07:46 ...

Figure 1: Generated and Compiled .v files

Questions

a

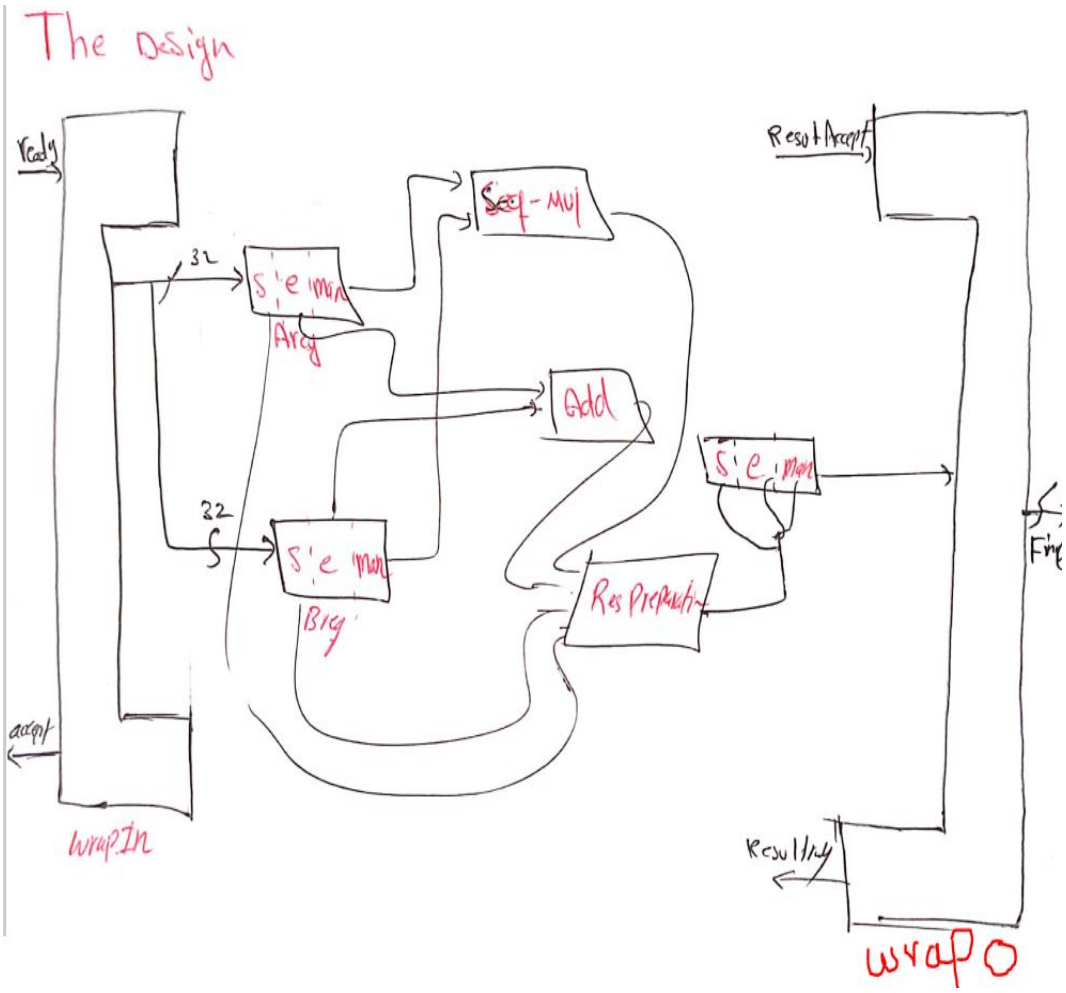


Figure 2: The Design

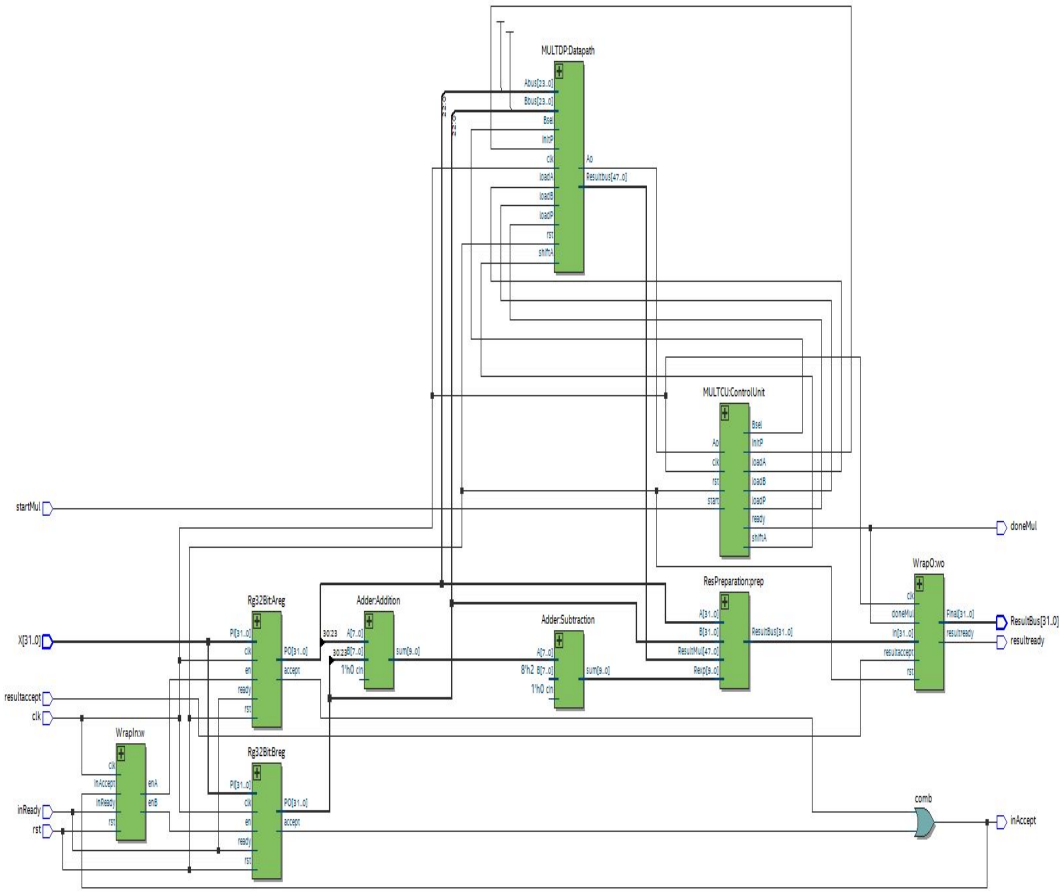
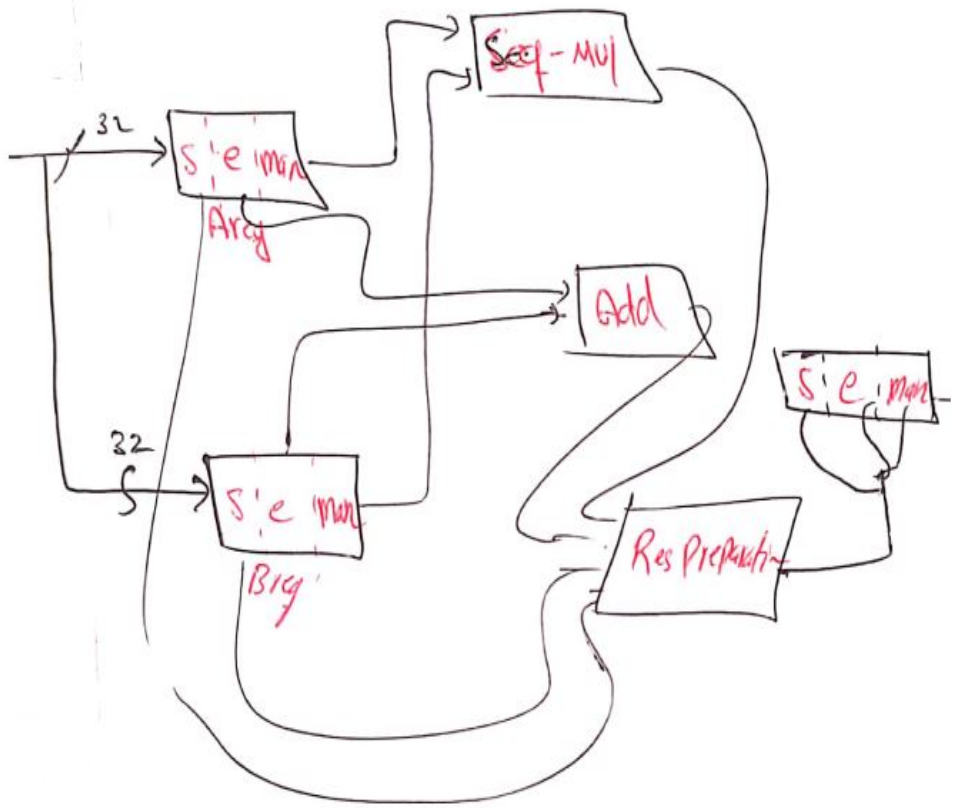


Figure 3: The RTL version of the design

b

Due to the combination of the sequential multiplier and the adder forming the FP part, therefore the shape shall be something like this:



c

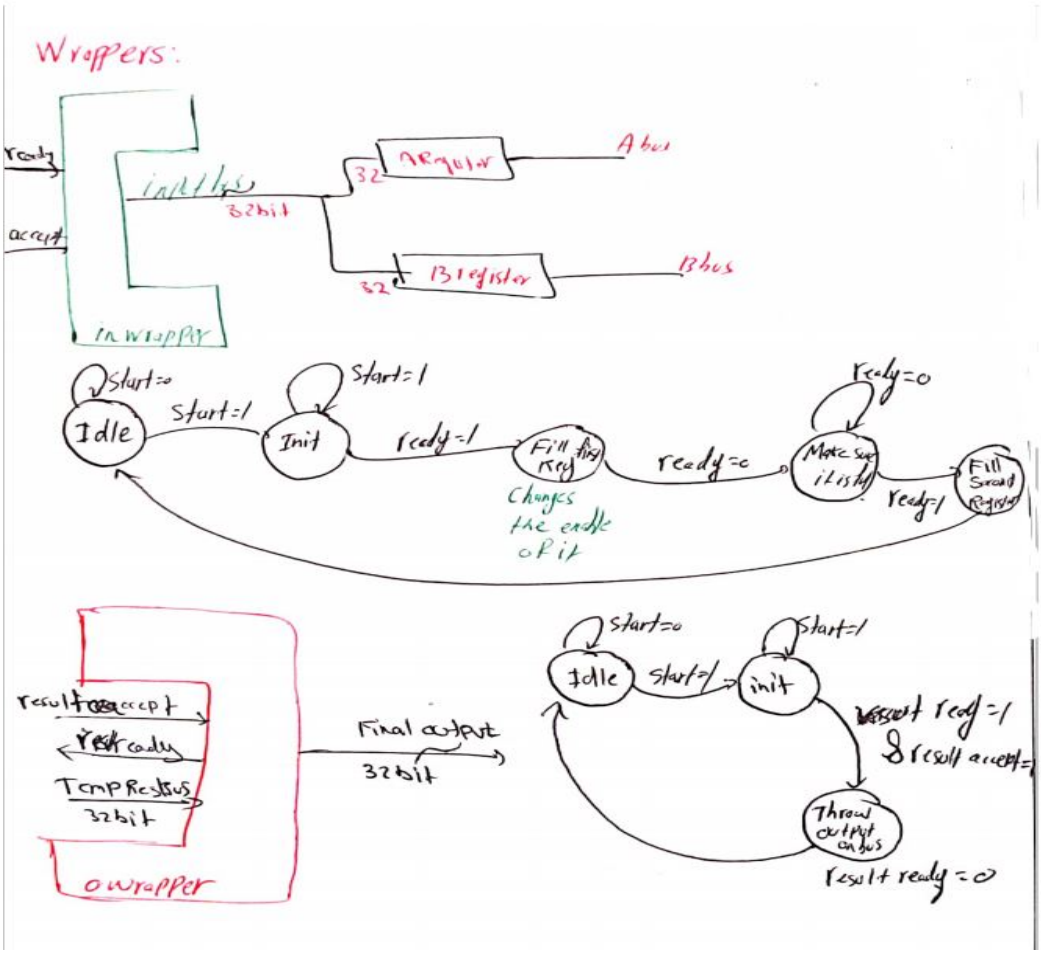


Figure 4: Wrapper Design

d

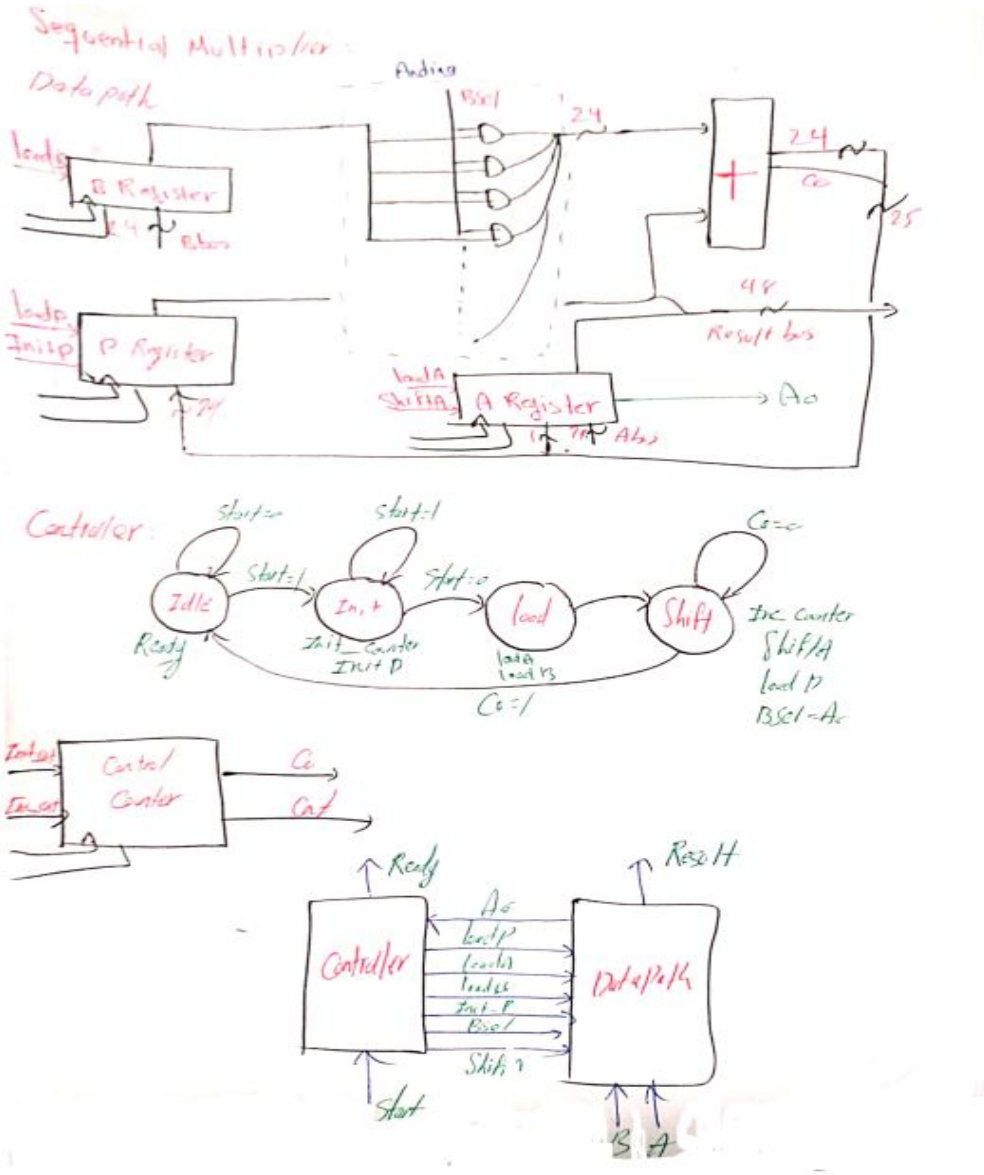


Figure 5: Sequential Multiplier

e

The simulation can be seen as follows.

Verilog Code & Testbench

```
`timescale 1ns/1ns
module SequentialMultTB();
        reg clk=1'b0;
        reg rst=0;
        reg start=0;
        reg [23:0] A;
        reg [23:0] B;
        wire [47:0] Resultbus;
        wire ready;
        SequentialMultTop UUT (clk, rst, start, A,B, Resultbus, ready);
        always #5 clk <= ~clk;
        initial begin
                 #3 rst=1;
                 #3 rst=0;
                #13 A=24'd2;
                #13 B=24'd2;
                 #3 start=1;
                #13 start=0;
                #300 A=24'd5;
                #100 B=24'd12;
                #3 start=1;
                #13 start=0;
                #400 $stop;
        end
endmodule
```

```
`timescale 1ns/1ns
module SequentialMultTop(input clk,rst,start,input [23:0] A,B,output[47:0] Resultbus,output ready);
wire A0;
wire loadA,shiftA,loadB,loadP,InitP,Bsel;
  MULTDP DP(clk,rst,loadA,loadB,loadP,shiftA,InitP,Bsel,A,B,Resultbus,A0);
  MULTCU CU(clk,rst,start,A0,loadA,shiftA,loadB,loadP,InitP,Bsel,ready);
endmodule
```

Figure 6: Verilog Code & Testbench

Waveform

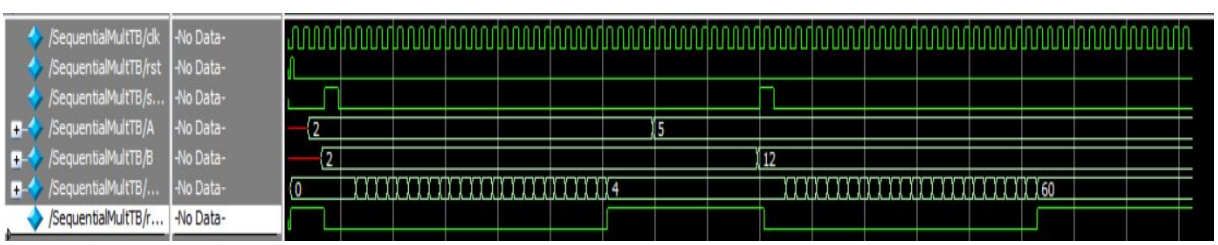


Figure 7: Waveform

f& g

Due to these parts being embedded in the design I couldn't test them separately, their functionality is included in the next parts.

h

The simulation can be seen as follows.

Verilog Code & Testbench

```verilog
`timescale 1ns/1ns
module FMultiPreSynthTB();
        reg clk=1'b0;
        reg rst=0;
        reg start=0;
        reg [31:0] A = 32'b01000000101000100000000000000000;
        reg [31:0] B = 32'b11000000011100000000000000000000;
        reg [31:0] Tempbus;
        reg resultaccept=0;
        reg ready = 0;
        wire accept,doneMul,resultready;
        wire [31:0] ResultBus;
        Top_Level UUT (clk, rst, start,resultaccept,Tempbus, ResultBus, ready , accept,doneMul,resultready);
        always #10 clk <= ~clk;
        initial begin
                #40 rst=1;
                #40 rst=0;
                #40 Tempbus = A;
                #40 ready = 1;
                #40 if(accept) begin ready = 0; end
                #40 Tempbus = B;
                #40 ready = 1;
                #40 if(accept) ready = 0;
                #40;
                #40 start=1;
                #40 start=0;
                #800 resultaccept=1;
                #200 resultaccept=0;
                #40 rst=1;
                #40 rst=0;
                #40 Tempbus =32'b01000000001000000000000000000000;
                #40 ready = 1;
                #40 if(accept) begin ready = 0; end
                #40 Tempbus = 32'b01000001100011000000000000000000;
                #40 ready = 1;
                #40 if(accept) ready = 0;
                #40;
                #40 start=1;
                #40 start=0;
                #800 resultaccept=1;
                #60 $stop;
        end
endmodule
```

```verilog
`timescale 1ns/1ns
module Top_Level(input clk, rst, startMul,resultaccept, input [31:0] X ,output [31:0] ResultBus, input inReady, output inaccept,output doneMul,resultready);
        wire A0, loadA, shiftA, loadB, loadP, Init0, Reel,co,cosub,NMlised;
        wire [47:0] ResultMul, PMlised;
        wire [22:0] PMantissa;
        wire [31:0] A, B,TempbusBus;
        wire enA, enB, inacceptA, inacceptB;
        wire [9:0] Radder, Rempl,Remp;
        or(inaccept,inacceptA,inacceptB);
        Reg32Bit Areg(.DI(X), .clk(clk), .rst(rst), .en(enA), .ready(inReady),.DO(A), .accept(inacceptA));
        Reg32Bit Breg(.DI(X), .clk(clk), .rst(rst), .en(enB), .ready(inReady), .DO(B), .accept(inacceptB));
        WrapIn w1(.clk(clk), .rst(rst), .inReady(inReady), .inaccept(inaccept), .enA(enA), .enB(enB));
        Wrap0 w0(.clk(clk),.rst(rst),.doneMul(doneMul),.resultaccept(resultaccept),.in(TempbusBus),.resultready(resultready),.Final(ResultBus));
        MULTIP dp(.clk(clk), .rst(rst), .loadA(loadA), .loadB(loadB), .loadP(loadP), .shiftA(shiftA), .Init0(Init0), .Reel(Reel), .Abus({1'b1,A[22:0]}), .Bbus({1'b1,B[22:0]}), .ResultBus(Resu
        MULTCU cu(.clk(clk), .rst(rst), .start(startMul), .A0(A0), .loadA(loadA), .shiftA(shiftA), .loadB(loadB), .loadP(loadP), .Init0(Init0), .Reel(Reel), .ready(doneMul));
        Adder Addition(.A(A[30:23]), .B(B[30:23]), .cin(1'b0), .sum(Radder), .co(co));
        Adder Subtraction(.A(Radder), .B({10'b1000000010}), .cin(1'b0), .sum(Remp), .co(cosub));
        ResSeparation prep(.NMlised(NMlised),.PMlised(PMlised),.PMantissa(PMantissa),.Rempl(Rempl),.ResultMul(ResultMul),.ResultBus(TempbusBus),.A(A),.B(B),.Remp(Remp));
```

Figure 8: Verilog Code & Testbench

Waveform

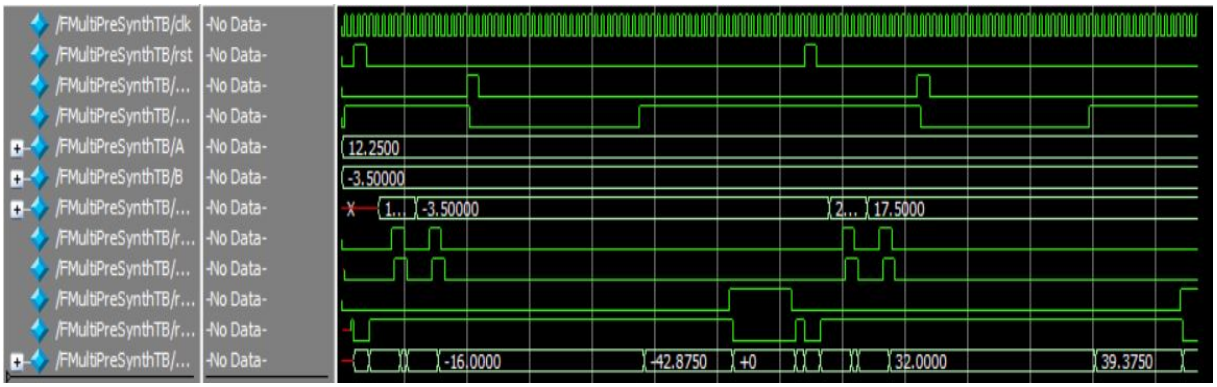
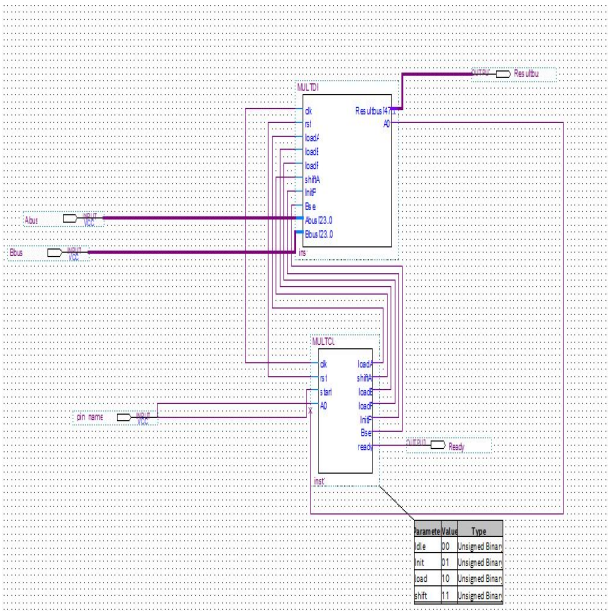
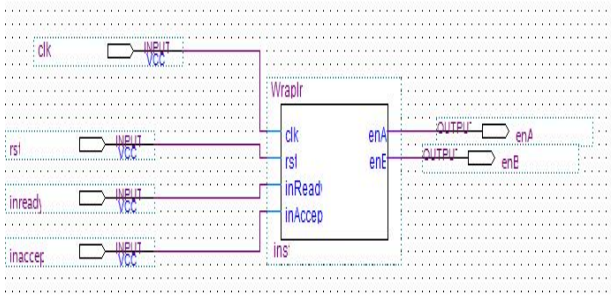


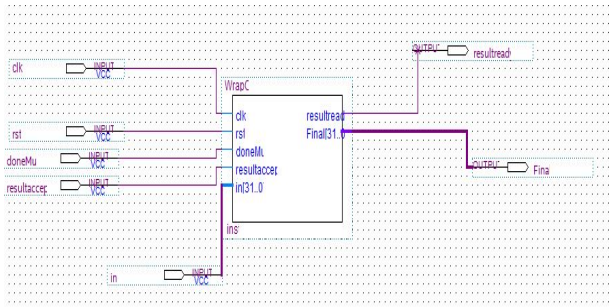
Figure 9: Waveform

i&j&k

The symbols can be seen as below, also the synthesized files are included in the project.







l&m

In this part we instantiate both of the pre and post synth files.

Verilog Code & Testbench

```
`timescale 1ns/1ns
module Top_Level(input clk, rst, startMul,resultaccept, input [31:0] X ,output [31:0] ResultBus, input inReady, output inAccept,output doneMul,resultready);
        wire A0, loadA, shiftA, loadB, loadP, InitP, Bsel,co,cosub,Nmlized;
        wire [47:0] ResultMul, PNmlized;
        wire [22:0] PMantissa;
        wire [31:0] A, B,TempResBus;
        wire enA, enB, inAcceptA, inAcceptB;
        wire [9:0] Radder, Rexp1,Rexp;
        or(inAccept,inAcceptA,inAcceptB);
        Rg32Bit Areg(.PI(X), .clk(clk), .rst(rst), .en(enA), .ready(inReady),.PO(A), .accept(inAcceptA));
        Rg32Bit Breg(.PI(X), .clk(clk), .rst(rst), .en(enB), .ready(inReady), .PO(B), .accept(inAcceptB));
        WrapIn w(.clk(clk), .rst(rst), .inReady(inReady), .inAccept(inAccept), .enA(enA), .enB(enB));
        WrapO wo(.clk(clk),.rst(rst),.doneMul(doneMul),.resultaccept(resultaccept),.in(TempResBus),.resultready(resultready),.Final(ResultBus));
        MULTDP dp(.clk(clk), .rst(rst), .loadA(loadA), .loadB(loadB), .loadP(loadP), .shiftA(shiftA), .InitP(InitP), .Bsel(Bsel), .Abus({1'b1,A[22:0]}), .Bbus({1'b1,B[22:0]}), .Resultbus(ResultMul), .A0(A0));
        MULTCU cu(.clk(clk), .rst(rst), .start(startMul), .A0(A0), .loadA(loadA), .shiftA(shiftA), .loadB(loadB), .loadP(loadP), .InitP(InitP), .Bsel(Bsel), .ready(doneMul));
        Adder Addition(.A(A[30:23]), .B(B[30:23]), .cin(1'b0), .sum(Radder), .co(co));
        Adder Subtraction(.A(Radder), .B({10'b1000000010}), .cin(1'b0), .sum(Rexp), .co(cosub));
        ResPreparation prep(.Nmlized(Nmlized),.PNmlized(PNmlized),.PMantissa(PMantissa),.Rexp1(Rexp1),.ResultMul(ResultMul),.ResultBus(TempResBus),.A(A),.B(B),.Rexp(Rexp));
endmodule
```

Figure 10: Top Level

```
`timescale 1ns/1ns
module PrePostInstanceTB();
        reg clk=1'b0;
        reg rst=0;
        reg startMul=0;
        reg [31:0] A = 32'b01000001010001000000000000000000;
        reg [31:0] B = 32'b11000000011000000000000000000000;
        reg [31:0] Tempbus;
        reg ready1 = 0;
        reg ready2 = 0;
        reg resultaccept=0;
        wire accept1,accept2,doneMul1,doneMul2,resultready1,resultready2;
        wire [31:0] ResultBus1;
        wire [31:0] ResultBus2;
        FMultiPostSynth UUT2 (clk, rst, startMul,resultaccept ,Tempbus, ResultBus2, ready2 , accept2,doneMul2,resultready2);
        Top_Level UUT1 (clk, rst, startMul,resultaccept ,Tempbus, ResultBus1, ready1 , accept1,doneMul1,resultready1);
        always #10 clk <= ~clk;
        initial begin
                #40 rst=1;
                #40 rst=0;
                #40 Tempbus = A;
                #40 ready1 = 1;ready2=1;
                #40 if(accept1) begin ready1 = 0; end if(accept2) begin ready2 = 0; end
                #40 Tempbus = B;
                #40 ready1 = 1;ready2=1;
                #40 if(accept1) ready1 = 0; if(accept2) ready2 = 0;
                #40;
                #40 startMul=1;
                #40 startMul=0;
                #800 resultaccept=1;
                #400 resultaccept=0;
                #40 rst=1;
                #40 rst=0;
                #40 Tempbus = 32'b01000000000100000000000000000000;
                #40 ready1 = 1;ready2=1;
                #40 if(accept1) begin ready1 = 0; end if(accept2) begin ready2 = 0; end
                #40 Tempbus = 32'b01000001100011000000000000000000;
                #40 ready1 = 1;ready2=1;
                #40 if(accept1) ready1 = 0; if(accept2) ready2 = 0;
                #40;
                #40 startMul=1;
                #40 startMul=0;
                #800 resultaccept=1;
                #200 $stop;
        end
endmodule
```

Figure 11: Testbench

Waveform

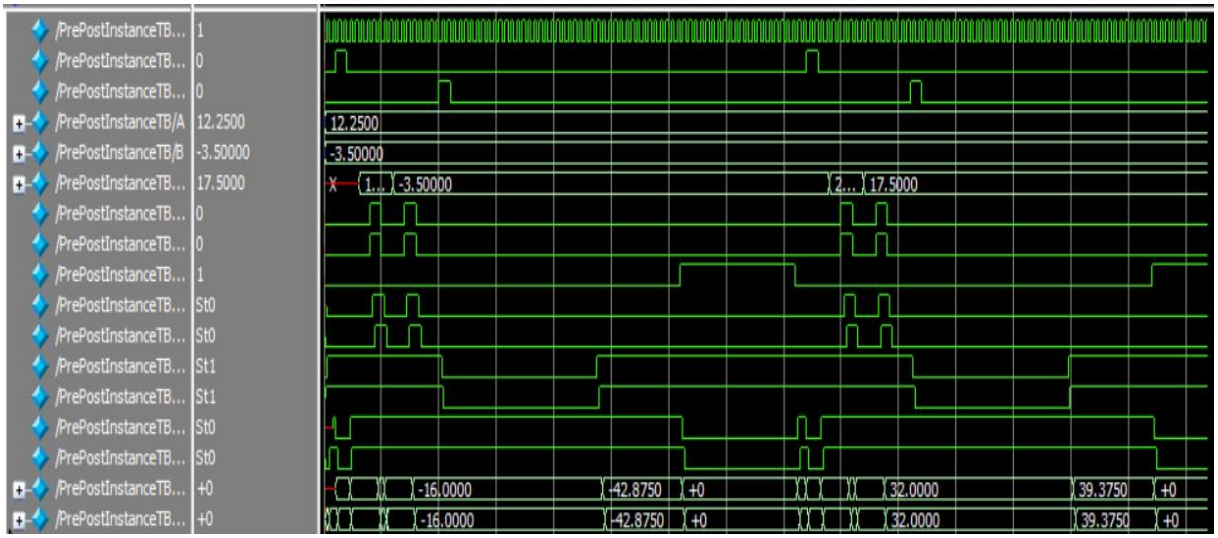


Figure 12: Waveform

o

Has been satisfied in previous parts.

p

The compilation report is as follows.

Flow Status	Successful - Sun Jun 27 16:53:38 2021
Quartus Prime Version	20.1.0 Build 711 06/05/2020 SJ Lite Edition
Revision Name	FMultiPostSynth
Top-level Entity Name	FMultiPostSynth
Family	Cyclone IV E
Device	EP4CE10F17C8L
Timing Models	Final
Total logic elements	250 / 10,320 (2 %)
Total registers	183
Total pins	72 / 180 (40 %)
Total virtual pins	0
Total memory bits	0 / 423,936 (0 %)
Embedded Multiplier 9-bit elements	0 / 46 (0 %)
Total PLLs	0 / 2 (0 %)