به نام خدا

UNIVERSITY OF TEHRAN
Electrical and Computer Engineering Department

Computer Assignment 6

Digital Systems I - ECE 894

Deadline : 6 Tir 1400

Erfan Panahi

810198369

Spring 1400

List of Problems:.

Computer Assignment 6

24bit-seqental multiplier. File: Multiplier24bit.v

A. Datapath

Block-Diagram:

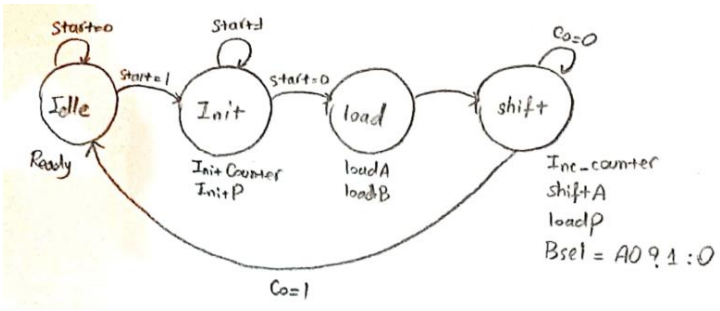
Lecture 33 (RTL methodology , Slide 48)

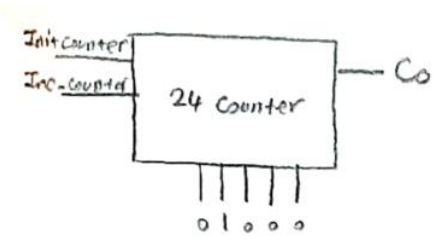
Verilog description:

```
module Mult24DP(input clk,rst,loadA,loadB,loadP,shiftA,initP,Bsel,input
[23:0]ABus,BBus,output [47:0]ResultBus,output A0);
    reg [23:0]Areg,Breg,Preg;
    wire [23:0]B_and;
    wire [24:0]add_Bus;
    //Bregister
    always @(posedge clk,posedge rst) begin
        if(rst) Breg <= 24'b0;
        else if(loadB) Breg <= BBus;
    end
    //Pregister
    always @(posedge clk,posedge rst) begin
        if(rst) Preg <= 24'b0;
        else begin
            if(initP) Preg <= 24'b0;
            else if(loadP) Preg <= add_Bus [24:1];
        end
    end
    //Ashift-register
    always @(posedge clk,posedge rst) begin
        if(rst) Areg <= 24'b0;
        else begin
            if(loadA) Areg <= ABus;
            else if(shiftA) Areg <= {add_Bus[0],Areg[23:1]};
        end
    end

    assign B_and = Bsel ? Breg : 24'b0;
    assign add_Bus = B_and + Preg;
    assign ResultBus = {Preg,Areg};
    assign A0 = Areg[0];

endmodule
```

B. Controller

State-Diagram:





Verilog description:

```
module Mult24CU(input clk,rst,start,A0,output reg
loadA,shiftA,loadB,loadP,initP,Bsel,ready);
    wire Co;
    reg init_counter,inc_counter;
    reg [1:0] pstate,nstate;
    reg [4:0] count;
    parameter Idle = 0,Init = 1, Load = 2, Shift = 3;
    //Counter
    always @(pstate,start,A0,Co) begin
        nstate = 0;
        {loadA,shiftA,loadB,loadP,initP,Bsel,ready} = 7'b0;
        {init_counter,inc_counter} = 2'b0;
        case(pstate)
            Idle: begin nstate = start ? Init : Idle; ready = 1'b1; end
            Init: begin nstate = start ? Init : Load; init_counter = 1'b1;initP = 1'b1;
end
            Load: begin nstate = Shift; loadA = 1'b1; loadB = 1'b1; end
            Shift: begin nstate = Co ? Idle : Shift; loadP = 1'b1; shiftA = 1'b1;
inc_counter = 1'b1; Bsel = A0; end
        endcase
    end

    always @(posedge clk,posedge rst) begin
        if(rst) pstate <= Idle;
        else pstate <= nstate;
    end

    always @(posedge clk,posedge rst) begin
        if(rst) count <= 5'd0;
        else begin
            if(init_counter) count <= 5'd8;
            else if(inc_counter) count <= count + 1'd1;
        end
    end

    assign Co = & count;

endmodule
```

C. Top-Level and Testbench:

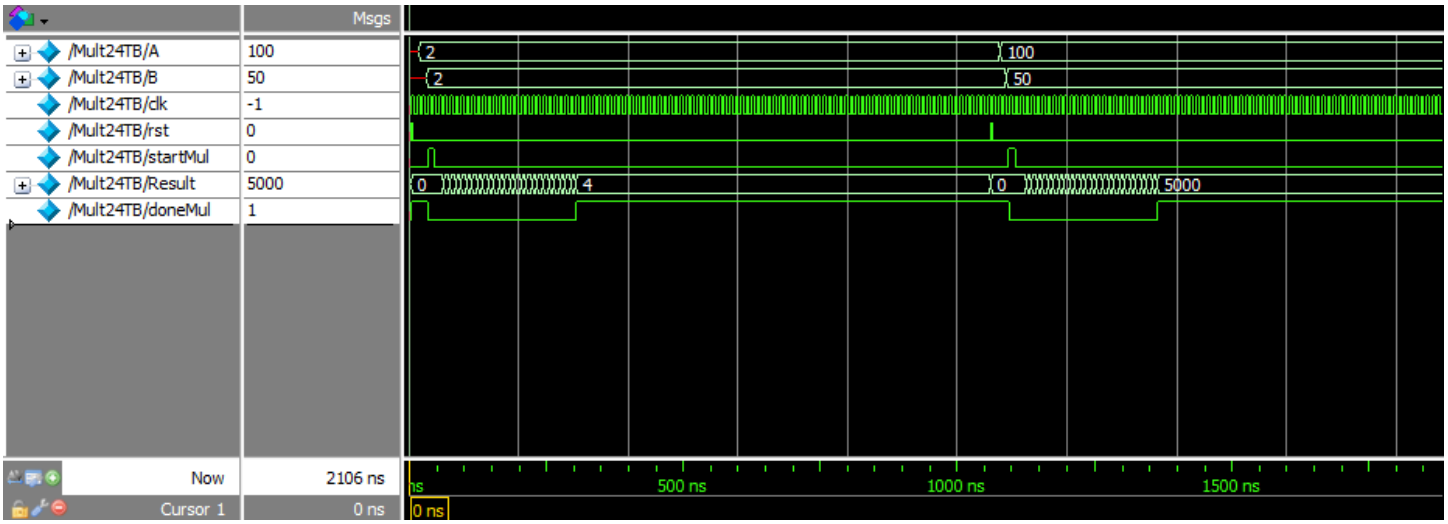
Verilog description:

```
module Mult24Top(input clk,rst,start,input [23:0]A,B,output [47:0] ResultBus,output
ready);
    wire A0;
    wire loadA,shiftA,loadB,loadP,initP,Bsel;
```

```
    Mult24DP dp(clk,rst,loadA,loadB,loadP,shiftA,initP,Bsel,A,B,ResultBus,A0);
    Mult24CU co(clk,rst,start,A0,loadA,shiftA,loadB,loadP,initP,Bsel,ready);

endmodule
```

```
module Mult24TB();
    reg [23:0]A,B;
    reg clk = 0, rst = 0 , startMul = 0;
    wire [47:0]Result;
    wire doneMul;
    Mult24Top UUT(clk,rst,startMul,A,B,Result,doneMul);
    always #5 clk <= ~clk;
    initial begin
        #3 rst = 1;
        #3 rst = 0;
        #13 A = 23'd2;
        #13 B = 23'd2;
        #3 startMul = 1;
        #13 startMul = 0;
        #1013 rst = 1;
        #3 rst = 0;
        #13 A = 23'd100;
        #13 B = 23'd50;
        #3 startMul = 1;
        #13 startMul = 0;
        #1000 $stop;
    end
endmodule
```

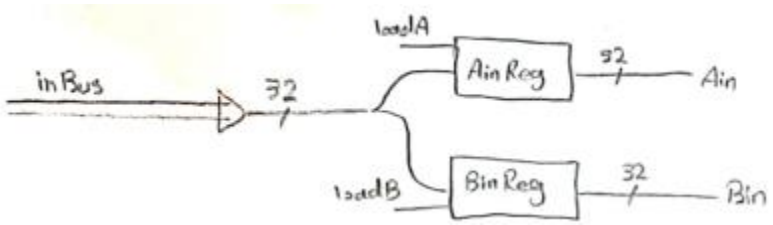
D. Waveform:



Input wrapper. File: WrapperIn.v

A. Datapath

Block-Diagram:

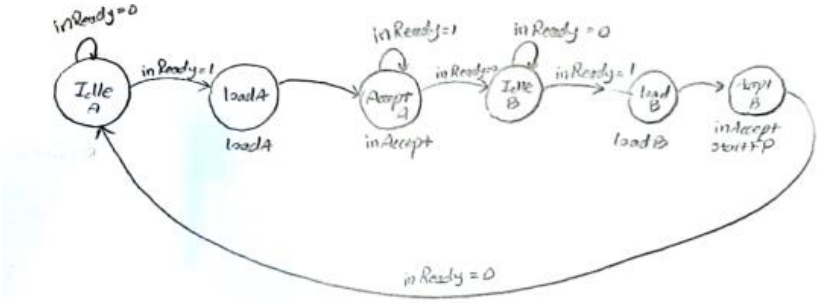


Verilog description:

```
`timescale 1ns/1ns
module WrapperIn_DP(input clk,rst,ldA,ldB,input[31:0] inBus,output reg[31:0] Ain,Bin);
    //Aregister
    always @(posedge clk,posedge rst) begin
        if(rst) Ain <= 32'b0;
        else if(ldA) Ain <= inBus;
    end
    //Bregister
    always @(posedge clk,posedge rst) begin
        if(rst) Bin <= 32'b0;
        else if(ldB) Bin <= inBus;
    end
endmodule
```

B. Controller

State-Diagram:



Verilog description:

```
module WrapperIn_CU(input clk,rst,inReady,output reg inAccept,ldA,ldB,startFP);
    reg [2:0] pstate,nstate;
    parameter Idle1 = 0, Load1 = 1, Accept1 = 2, Idle2 = 3, Load2 = 4, Accept2 = 5;

    always @(pstate,inReady) begin
        nstate = 0;
        {ldA,ldB,inAccept,startFP} = 4'b0;
        case(pstate)
```

```
            Idle1: begin nstate = inReady ? Load1 : Idle1; end
            Load1: begin nstate = Accept1; ldA = 1'b1; end
            Accept1: begin nstate = inReady ? Accept1 : Idle2 ; inAccept = 1'b1; end
            Idle2: begin nstate = inReady ? Load2 : Idle2; end
            Load2: begin nstate = Accept2; ldB = 1'b1; end
            Accept2: begin nstate = inReady ? Accept2 : Idle1; inAccept = 1'b1; startFP =
1'b1; end
        endcase
    end

    always @(posedge clk,posedge rst) begin
        if(rst) pstate <= Idle1;
        else pstate <= nstate;
    end
endmodule
```

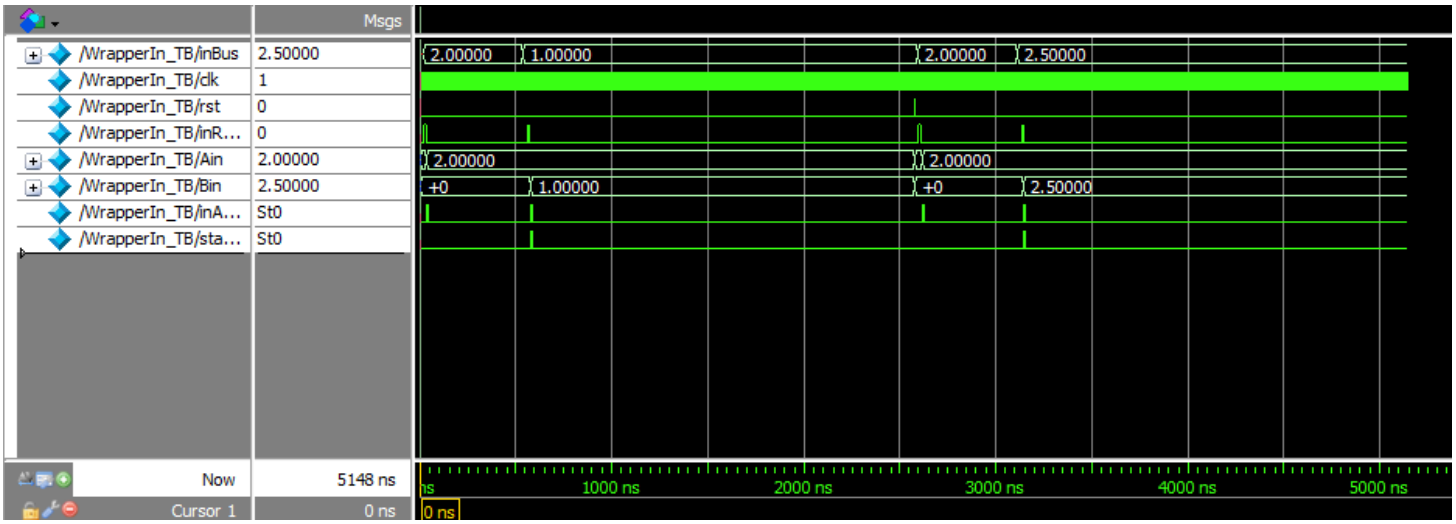
E. Top-Level and Testbench:

Verilog description:

```
module Wrapper32In_Top(input clk,rst,inReady,input[31:0] inBus,output[31:0]
Ain,Bin,output inAccept,startFP);
    wire ldA,ldB;
    WrapperIn_DP dp1(clk,rst,ldA,ldB,inBus,Ain,Bin);
    WrapperIn_CU cu1(clk,rst,inReady,inAccept,ldA,ldB,startFP);
endmodule
```

```
module WrapperIn_TB();
    reg [31:0]inBus;
    reg clk = 0, rst = 0 , inReady = 0;
    wire [31:0]Ain,Bin;
    Wrapper32In_Top UUT(clk,rst,inReady,inBus,Ain,Bin,inAccept,startFP);
    always #5 clk <= ~clk;
    initial begin
        #3 rst = 1;
        #3 rst = 0;
        #13 inBus = 32'b01000000000000000000000000000000;
        #3 inReady = 1;
        #13 inReady = 0;
        #503 inBus = 32'b00111111100000000000000000000000;
        #23 inReady = 1;
        #13 inReady = 0;
        #2003 rst = 1;
        #3 rst = 0;
        #13 inBus = 32'b01000000000000000000000000000000;
        #3 inReady = 1;
        #13 inReady = 0;
        #503 inBus = 32'b01000000001000000000000000000000;
        #23 inReady = 1;
        #13 inReady = 0;
        #2000 $stop;
    end
endmodule
```

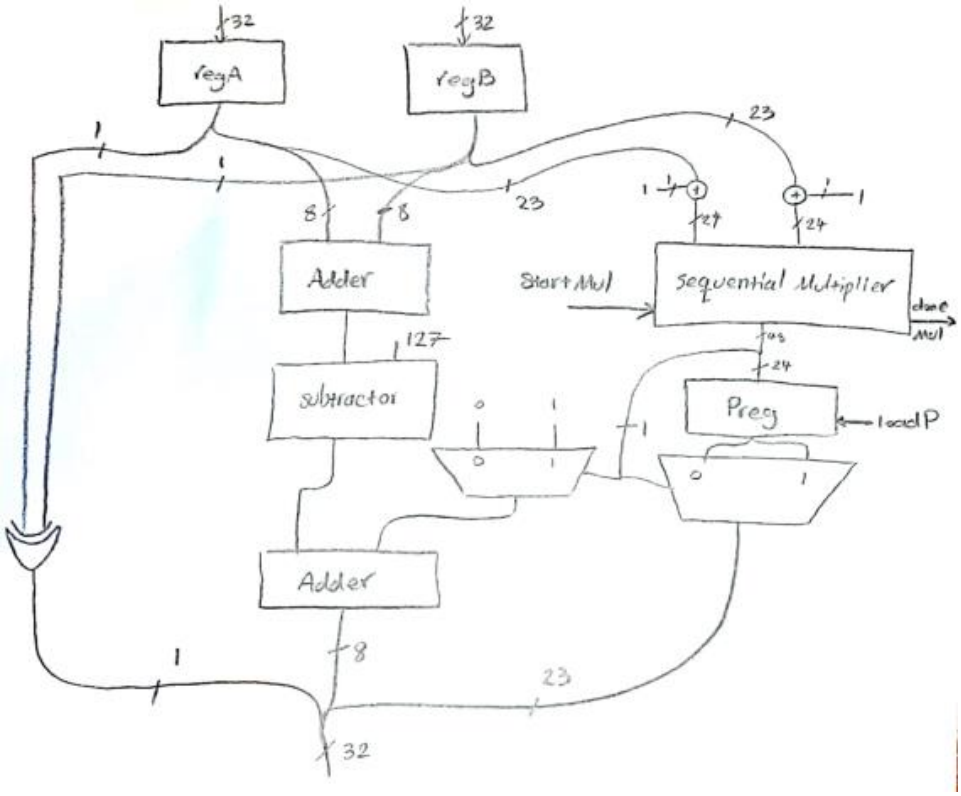
F. Waveform:


Msgs
/WrapperIn_TB/inBus
/WrapperIn_TB/clk
/WrapperIn_TB/rst
/WrapperIn_TB/inR...
/WrapperIn_TB/Ain
/WrapperIn_TB/Bin
/WrapperIn_TB/inA...
/WrapperIn_TB/sta...
2.50000
1
0
0
2.00000
2.50000
St0
St0
2.00000
1.00000
2.00000
2.50000
+0
1.00000
+0
2.50000
Now
5148 ns
Cursor 1
0 ns
1000 ns
2000 ns
3000 ns
4000 ns
5000 ns
0 ns

FP multiplication. File: MultiplierFP_32bit.v

A. Datapath

Block-Diagram:



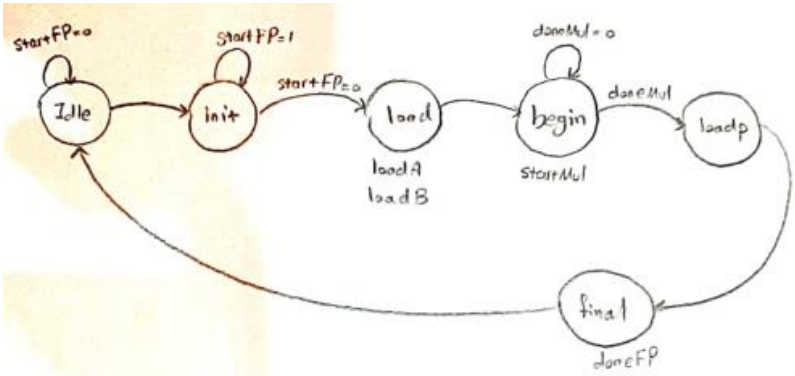
Verilog description:

```
`timescale 1ns/1ns
module MultFP32_DP(input clk,rst,loadA,loadB,initP,loadP,startMul,input[31:0]
Ain,Bin,output[31:0] Result,output P47,doneMul);
    reg[31:0] Areg,Breg,Preg;
    reg[7:0] Aexp,Bexp;
    wire[47:0] mul24Bus;
    Mult24Top MUL(clk,rst,startMul,{1'b1,Areg[22:0]},{1'b1,Breg[22:0]},mul24Bus,doneMul);

    //Aregister
    always @(posedge clk,posedge rst) begin
        if(rst) Areg <= 32'b0;
        else if(loadA) Areg <= Ain;
    end
    //Bregister
    always @(posedge clk,posedge rst) begin
        if(rst) Breg <= 32'b0;
        else if(loadB) Breg <= Bin;
    end
    always @(posedge clk,posedge rst) begin
        if(rst) Preg <= 24'b0;
        else begin
            if(initP) Preg <= 24'b0;
            else if(loadP) Preg <= mul24Bus[46:23];
        end
```

```
    end

    assign Result[31] = (Areg[31] ^ Breg[31]);
    assign P47 = mul24Bus[47];
    assign Aexp = Areg[30:23];
    assign Bexp = Breg[30:23];
    assign Result[30:23] = (Aexp + Bexp) - 8'd127 + P47;
    assign Result[22:0] = P47 ? mul24Bus[46:24] : mul24Bus[45:23];

endmodule
```

B. Controller

State-Diagram:



Verilog description:

```
module MultFP32_CU(input clk,rst,startFP,doneMul,P47,output reg
loadA,loadB,initP,loadP,startMul,doneFP);
    reg [2:0] pstate,nstate;
    parameter Idle = 0, Init = 1, Load1 = 2, Mul24 = 3, Load2 = 4, Calc = 5;

    always @(pstate,doneMul,startFP,P47) begin
        nstate = 0;
        {loadA,loadB,initP,loadP,startMul,doneFP} = 6'b0;
        case(pstate)
            Idle: begin nstate = startFP ? Init : Idle;  end
            Init: begin nstate = startFP ? Init : Load1; initP = 1'b1; end
            Load1: begin nstate = Mul24; loadA = 1'b1; loadB = 1'b1; end
            Mul24: begin nstate = doneMul ? Load2 : Mul24; startMul = 1'b1; end
            Load2: begin nstate = Calc; loadP = 1'b1; end
            Calc: begin nstate = Idle; doneFP = 1'b1; end
        endcase
    end

    always @(posedge clk,posedge rst) begin
        if(rst) pstate <= Idle;
        else pstate <= nstate;
    end
endmodule
```

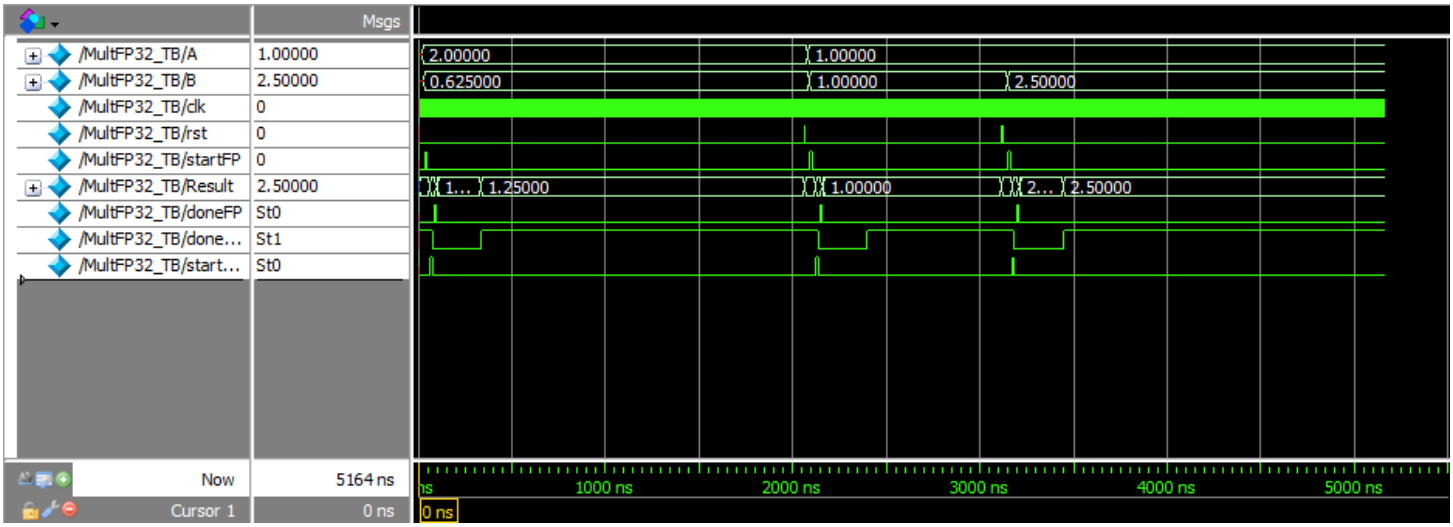
C. Top-Level and Testbench:

Verilog description:

```
module MultFP32_Top(input clk,rst,startFP,input[31:0] A,B,output[31:0] Result,output
doneFP,doneMul,startMul);
    wire P47;
    wire loadA,loadB,loadP,initP;

    MultFP32_DP dp(clk,rst,loadA,loadB,initP,loadP,startMul,A,B,Result,P47,doneMul);
    MultFP32_CU cu(clk,rst,startFP,doneMul,P47,loadA,loadB,initP,loadP,startMul,doneFP);
endmodule
```

```
module MultFP32_TB();
    reg [31:0]A,B;
    reg clk = 0, rst = 0 , startFP = 0;
    wire [31:0]Result;
    wire doneFP,doneMul,startMul;
    MultFP32_Top UUT(clk,rst,startFP,A,B,Result,doneFP,doneMul,startMul);
    always #5 clk <= ~clk;
    initial begin
        #3 rst = 1;
        #3 rst = 0;
        #13 A = 32'b01000000000000000000000000000000;
        #13 B = 32'b00111111100100000000000000000000;
        #3 startFP = 1;
        #13 startFP = 0;
        #2013 rst = 1;
        #3 rst = 0;
        #13 A = 32'b00111111100000000000000000000000;
        #13 B = 32'b00111111100000000000000000000000;
        #3 startFP = 1;
        #13 startFP = 0;
        #1013 rst = 1;
        #3 rst = 0;
        #13 A = 32'b00111111100000000000000000000000;
        #13 B = 32'b01000000001000000000000000000000;
        #3 startFP = 1;
        #13 startFP = 0;
        #2000 $stop;
    end
endmodule
```

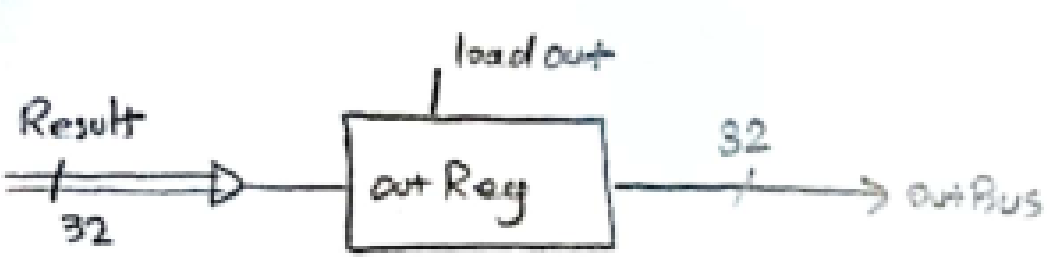
D. Waveform:



Output wrapper. File: WrapperOut.v

A. Datapath

Block-Diagram:

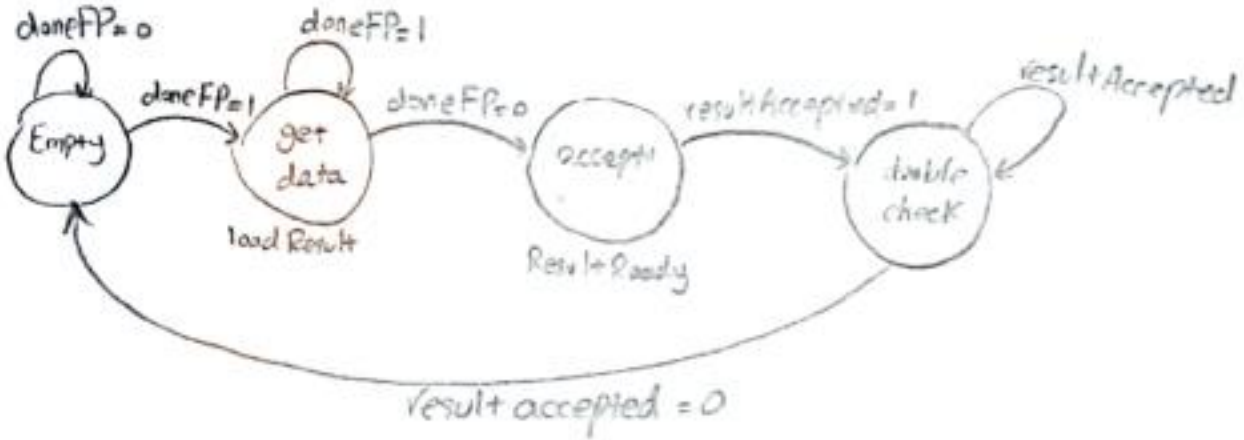


Verilog description:

```
`timescale 1ns/1ns
module WrapperOut_DP(input clk,rst,loadResult,input[31:0] Result,output reg[31:0]
OutBus);
    //Aregister
    always @(posedge clk,posedge rst) begin
        if(rst) OutBus <= 32'b0;
        else if(loadResult) OutBus <= Result;
    end
endmodule
```

B. Controller

State-Diagram:



Verilog description:

```
module WrapperOut_CU(input clk,rst,doneFP,resultAccepted,output reg
resultReady,loadResult);
    reg [1:0] pstate,nstate;
    parameter Empty = 0, Receive = 1, waitAccept = 2, sureAccept = 3;

    always @(pstate,doneFP,resultAccepted) begin
        nstate = 0;
```

```
        {loadResult,resultReady} = 2'b0;
        case(pstate)
            Empty: begin nstate = doneFP ? Receive : Empty; end
            Receive: begin nstate = doneFP ? Receive : waitAccept; loadResult = 1'b1; end
            waitAccept: begin nstate = resultAccepted ? sureAccept : waitAccept ;
resultReady = 1'b1; end
            sureAccept: begin nstate = resultAccepted ? sureAccept : Empty; end

        endcase
    end

    always @(posedge clk,posedge rst) begin
        if(rst) pstate <= Empty;
        else pstate <= nstate;
    end
endmodule
```

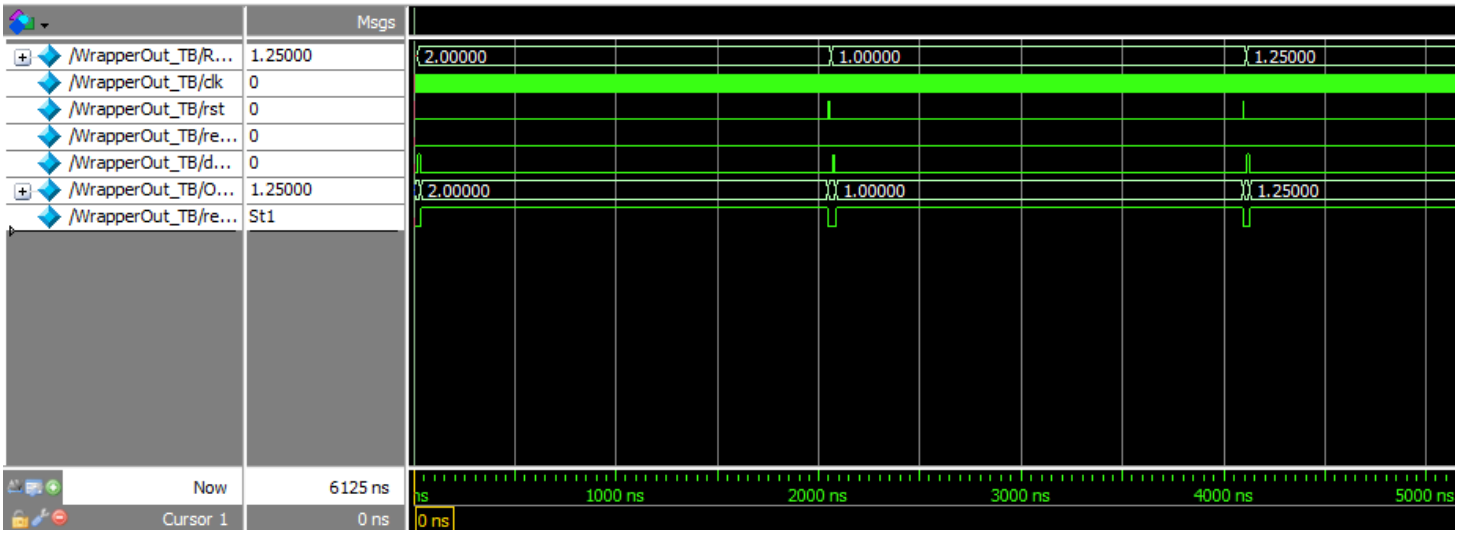
A. Testbench:

Verilog description:

```
module WrapperOut32_Top(input clk,rst,resultAccepted,doneFP,input[31:0]
Result,output[31:0] OutBus,output resultReady);
    wire loadResult;
    WrapperOut_DP dp1(clk,rst,loadResult,Result,OutBus);
    WrapperOut_CU cu1(clk,rst,doneFP,resultAccepted,resultReady,loadResult);
endmodule
```

```
module WrapperOut_TB();
    reg [31:0]Result;
    reg clk = 0, rst = 0 , resultAccepted = 0, doneFP = 0;
    wire [31:0]OutBus;
    WrapperOut32_Top UUT(clk,rst,resultAccepted,doneFP,Result,OutBus,resultReady);
    always #5 clk <= ~clk;
    initial begin
        #3 rst = 1;
        #3 rst = 0;
        #13 Result = 32'b01000000000000000000000000000000;
        #3 doneFP = 1;
        #13 doneFP = 0;
        #2013 rst = 1;
        #3 rst = 0;
        #13 Result = 32'b00111111100000000000000000000000;
        #3 doneFP = 1;
        #13 doneFP = 0;
        #2013 rst = 1;
        #3 rst = 0;
        #13 Result = 32'b00111111101000000000000000000000;
        #3 doneFP = 1;
        #13 doneFP = 0;
        #2000 $stop;
    end
endmodule
```

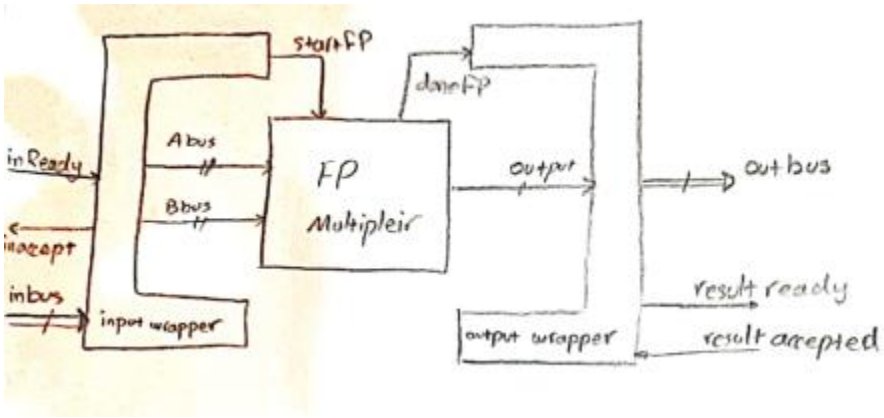
B. Waveform:


Msgs
/WrapperOut_TB/R...
/WrapperOut_TB/clk
/WrapperOut_TB/rst
/WrapperOut_TB/re...
/WrapperOut_TB/d...
/WrapperOut_TB/O...
/WrapperOut_TB/re...
1.25000
0
0
0
1.25000
St1
2.00000
1.00000
1.25000
2.00000
1.00000
1.25000
Now
6125 ns
Cursor 1
0 ns
1000 ns
2000 ns
3000 ns
4000 ns
5000 ns
0 ns

Final FP multiplication. File: FinalFP32bit.v

A. Top-Level

Block-Diagram:



Verilog description:

```
`timescale 1ns/1ns
module Final_Top(input clk,rst,inReady,resultAccepted,input[31:0] inBus,output[31:0]
OutBus,output inAccept,startFP,doneFP,doneMul,startMul,resultReady);
    wire P47;
    wire ldA,ldB,loadA,loadB,loadP,initP,loadResult;
    wire[31:0]Ain,Bin,Result;
    WrapperIn_DP dp1(clk,rst,ldA,ldB,inBus,Ain,Bin);
    WrapperIn_CU cu1(clk,rst,inReady,inAccept,ldA,ldB,startFP);
    MultFP32_DP dp2(clk,rst,loadA,loadB,initP,loadP,startMul,Ain,Bin,Result,P47,doneMul);
    MultFP32_CU cu2(clk,rst,startFP,doneMul,P47,loadA,loadB,initP,loadP,startMul,doneFP);
    WrapperOut_DP dp3(clk,rst,loadResult,Result,OutBus);
    WrapperOut_CU cu3(clk,rst,doneFP,resultAccepted,resultReady,loadResult);
endmodule
```

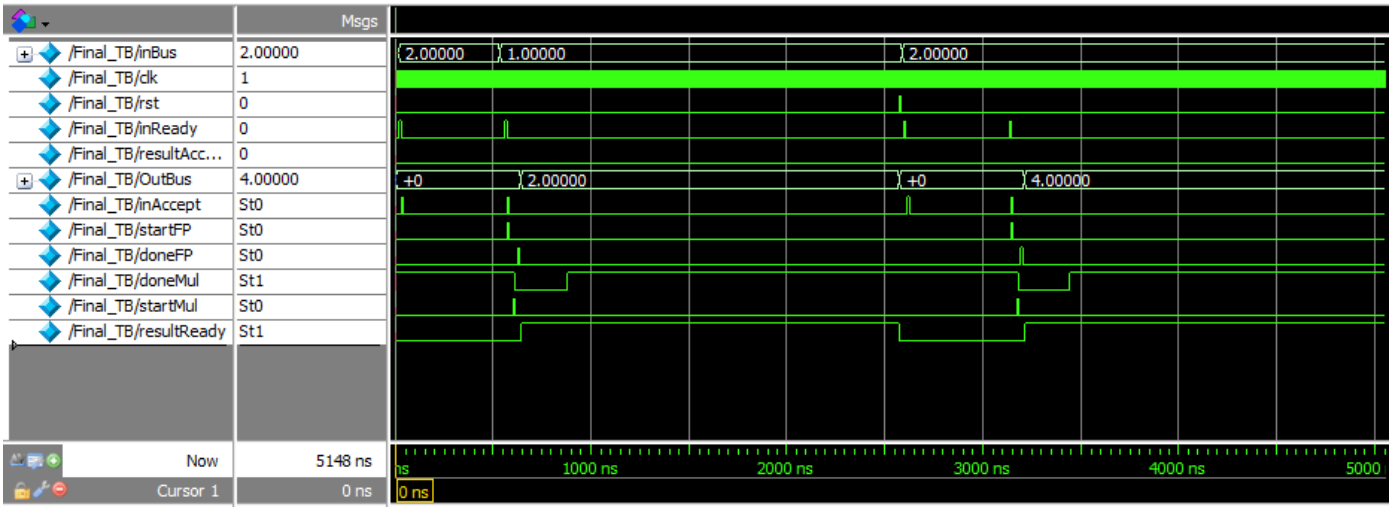
B. Testbench:

Verilog description:

```
module Final_TB();
    reg [31:0]inBus;
    reg clk = 0, rst = 0 , inReady = 0 ,resultAccepted = 0;
    wire [31:0]OutBus;
    wire inAccept,startFP,doneFP,doneMul,startMul,resultReady;
    Final_Top
UUT(clk,rst,inReady,resultAccepted,inBus,OutBus,inAccept,startFP,doneFP,doneMul,startMul,
resultReady);

    always #5 clk <= ~clk;
    initial begin
        #3 rst = 1;
        #3 rst = 0;
        #13 inBus = 32'b01000000000000000000000000000000;
        #3 inReady = 1;
        #13 inReady = 0;
        #503 inBus = 32'b00111111100000000000000000000000;
```

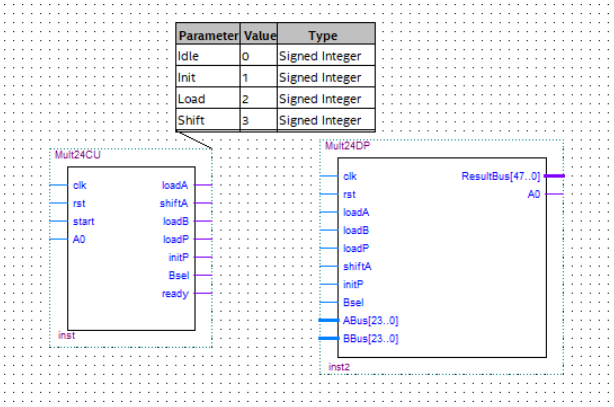
```
        #23 inReady = 1;
        #13 inReady = 0;
        #2003 rst = 1;
        #3 rst = 0;
        #13 inBus = 32'b01000000000000000000000000000000;
        #3 inReady = 1;
        #13 inReady = 0;
        #503 inBus = 32'b01000000000000000000000000000000;
        #23 inReady = 1;
        #13 inReady = 0;
        #2000 $stop;
    end
endmodule
```

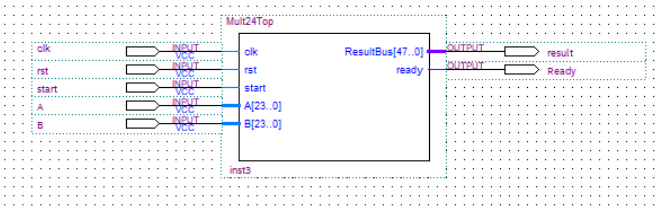
C. Waveform:



Synthesize.

24bit-seqental multiplier:





Input wrapper:

FP multiplication:

Output wrapper:

Final FP multiplication:

Synthesizes: