

# **Project 1**

## **Group members**

**Ahmed Awad-Allah Mohamed Ali**

**Ibrahim Khaled Mohamed**

**Farah Ahmed Fathy**

# Spartan6 DSP48A1

## Design 1(actual design):

```
module LOGIC_INPUT1(D,clk,CE,Q,rst);
parameter WIDTH=1;
parameter DREG=1;
parameter RSTTYPE="SYNC";
input [WIDTH-1:0] D;
input clk,rst,CE;
output reg [WIDTH-1:0] Q;
generate
    if(DREG==1) begin
        if(RSTTYPE == "SYNC") begin
            always @(posedge clk) begin
                if(rst)
                    Q <= 0;
                else if (CE)
                    Q <= D;
            end
        end

        else if (RSTTYPE == "ASYNC")
            begin
                always @(posedge clk or posedge rst)
                    begin
                        if (rst)
                            Q <= 0;
                        else if(CE)
                            Q <= D;
                    end
            end
    end

    else if(DREG==0)
        begin
            always @(*)
                Q = D;
        end
end

endgenerate
endmodule
//*****
module MUX (in0,in1,in2,in3,sel,out);
parameter INPUT_SIZE=4,
            SEL_SIZE=2,
            IO_WIDTH=48;

input      [IO_WIDTH-1:0] in0,in1,in2,in3;
```

```

input      [SEL_SIZE-1:0] sel;
output reg [IO_WIDTH-1:0] out;
generate
if(INPUT_SIZE=='d4 && SEL_SIZE=='d2 && IO_WIDTH=='d48)
always@(*)begin
case (sel)
    2'b00 : out = in0;
    2'b01 : out = in1;
    2'b10 : out = in2;
    2'b11 : out = in3;
endcase
end

else if (INPUT_SIZE=='d2 && SEL_SIZE=='d1 && IO_WIDTH=='d18)
always@(*)begin
case (sel)
    1'b0 : out = in0;
    1'b1 : out = in1;
    default : out=in0;
endcase
end
endgenerate
endmodule

//*****

module MUX_B (in0,in1,out);
parameter B_INPUT="DIRECT";
input [17:0]in0,in1;
output [17:0]out;

case(B_INPUT)
"DIRECT" : assign out=in0;
"CASCADE": assign out=in1;
default  : assign out=0;
endcase
endmodule

//*****

module MUX_C (in0,in1,out);
parameter CARRYINSEL="OPMODE5";
input in0,in1;
output out;

case(CARRYINSEL)
"OPMODE5": assign out=in0;
"CARRYIN": assign out=in1;
default  : assign out=1'b0;
endcase
endmodule

```

```

//*****

module ADD_SUB_pre (A,B,SEL,OUT);
input [17:0] A,B;
input SEL ;
output reg [17:0] OUT;

always@(*)begin
    if(SEL==0)
OUT=A+B;
    else
OUT=A-B;
end
endmodule

//*****

module ADD_SUB_post (G2,G1,CIN,SEL,OUT,CARRY_OUT);
input [47:0] G2,G1;
input SEL ,CIN;
output reg [47:0] OUT;
output reg CARRY_OUT;

always@(*) begin
    if(SEL==0)
{CARRY_OUT,OUT}=G2+G1+CIN;
    else
{CARRY_OUT,OUT}=G2-(G1+CIN);
end
endmodule

//*****

module
TOP_MODULE(D,B,BCIN,A,C,PCIN,clk,CARRYIN,M,P,CARRYOUT,CARRYOUTF,OPMODE,CEA,CEB,CEC,CECARRYIN,
CED,CEM,CEOPMODE,CEP,RSTA,RSTB,RSTC,RSTCARRYIN,RSTD,RSTM,RSTOPMODE,RSTP,BCOUT,PCOUT);
parameter A0REG=0,
A1REG=1,
B0REG=0,
B1REG=1,
CREG=1,
DREG=1,
MREG=1,
PREG=1,
CARRYINREG=1,
CARRYOUTREG=1,
OPMODEREG=1,
CARRYINSEL="OPMODE5",
B_INPUT="DIRECT",
RSTTYPE="SYNC";

```

```

//*****
input [17:0] A,B,D,BCIN;
input [47:0] C,PCIN;
input [7:0] OPMODE;
input clk,CARRYIN,RSTA,RSTB,RSTM,RSTP,RSTC,RSTD,RSTCARRYIN,
      RSTOPMODE,CEA,CEB,CEM,CEP,CEC,CED,CECARRYIN,CEOPMODE;
//*****
output [17:0]BCOUT;
output [47:0] PCOUT,P;
output [35:0] M;
output CARRYOUT,CARRYOUTF;
//*****
wire [17:0] IN1,A1,A2,A3,B1,C1,D1,D2;
wire [47:0] G1,G2,H1,CONC,EXTIN,A4;
wire [7:0] OpBUF;
wire [35:0] E1,F1;
wire IN2,H2,O1,CIN;
//*****
MUX_B #(.B_INPUT(B_INPUT))
      M0 (B,BCIN,IN1);

LOGIC_INPUT1 #(.WIDTH('d18),.DREG(DREG),.RSTTYPE(RSTTYPE))
             M1 (D,clk,CED,A1,RSTD);

LOGIC_INPUT1 #(.WIDTH('d18),.DREG(B0REG),.RSTTYPE(RSTTYPE))
             M2 (IN1,clk,CEB,A2,RSTB);

LOGIC_INPUT1 #(.WIDTH('d18),.DREG(A0REG),.RSTTYPE(RSTTYPE))
             M3 (A,clk,CEA,A3,RSTA);

LOGIC_INPUT1 #(.WIDTH('d48),.DREG(CREG),.RSTTYPE(RSTTYPE))
             M4 (C,clk,CEC,A4,RSTC);

LOGIC_INPUT1 #(.WIDTH('d8),.DREG(OPMODEREG),.RSTTYPE(RSTTYPE))
             M5 (OPMODE,clk,CEOPMODE,OpBUF,RSTOPMODE);

ADD_SUB_pre M12 (A1,A2,OpBUF[6],B1);

MUX #(.INPUT_SIZE('d2),.SEL_SIZE('d1),.IO_WIDTH('d18))
     M13 (A2,B1,18'b0,18'b0,OpBUF[4],C1);

LOGIC_INPUT1 #(.WIDTH('d18),.DREG(B1REG),.RSTTYPE(RSTTYPE))
             M6 (C1,clk,CEB,D1,RSTB);

LOGIC_INPUT1 #(.WIDTH('d18),.DREG(A1REG),.RSTTYPE(RSTTYPE))
             M7 (A3,clk,CEA,D2,RSTA);

assign E1= D1*D2;

LOGIC_INPUT1 #(.WIDTH('d36),.DREG(MREG),.RSTTYPE(RSTTYPE))

```

```

        M8 (E1,clk,CEM,F1,RSTM);

assign EXTIN={{12{F1[35]}},F1[35:0]};

LOGIC_INPUT1 #(.WIDTH('d1),.DREG(CARRYINREG),.RSTTYPE(RSTTYPE))
    M11 (IN2,clk,CECARRYIN,CIN,RSTCARRYIN);

MUX_C #(.CARRYINSEL(CARRYINSEL))
    M10 (OpBUF[5],CARRYIN,IN2);

MUX #(.INPUT_SIZE('d4),.SEL_SIZE('d2),.IO_WIDTH('d48))
    M16 (48'b0,EXTIN,P,CONC,OpBUF[1:0],G1);

assign CONC={D[11:0],D2,D1};

MUX #(.INPUT_SIZE('d4),.SEL_SIZE('d2),.IO_WIDTH('d48))
    M17 (48'b0,PCIN,P,A4,OpBUF[3:2],G2);

ADD_SUB_post M15 (G2,G1,CIN,OpBUF[7],H1,H2);

LOGIC_INPUT1 #(.WIDTH('d1),.DREG(CARRYOUTREG),.RSTTYPE(RSTTYPE))
    M18 (H2,clk,CECARRYIN,CARRYOUT,RSTCARRYIN);

LOGIC_INPUT1 #(.WIDTH('d48),.DREG(PREG),.RSTTYPE(RSTTYPE))
    M19 (H1,clk,CEP,P,RSTP);

//*****

assign BCOUT=D1;
assign M=F1;
assign CARRYOUTF=CARRYOUT;
assign PCOUT=P;

endmodule

```

## Design 2(Golden):

```
module pre_adder_subtractor(A1,A2,opmode,B1);
input [17:0] A1,A2;
input opmode;
output reg [17:0] B1;
always @(*) begin
    if(opmode == 0) begin
        B1 = A1 + A2;
    end
    else begin
        B1 = A1 - A2;
    end
end
endmodule

module post_adder_subtractor(G1,G2,opmode,H1,CIN,COUT);
input [47:0] G1,G2;
input opmode,CIN;
output reg [47:0] H1;
output reg COUT;
always @(*) begin
    if(opmode == 0) begin
        {COUT,H1} = G1 + G2 + CIN;
    end
    else begin
        {COUT,H1} = G2 - (G1 + CIN);
    end
end
endmodule

module multiplier(D1,D2,E1);
input [17:0] D1,D2;
output [35:0] E1;
assign E1 = D1 * D2;
endmodule

module MUX2x1(in0,in1,out,sel);
parameter INOUT_WIDTH=1;
input [INOUT_WIDTH-1:0] in0,in1;
input sel;
output [INOUT_WIDTH-1:0] out;
assign out = (sel == 0)?in0:in1;
endmodule
```

```

module MUX4x1(in0,in1,in2,in3,out,sel);
parameter INOUT_WIDTH=1;
input [INOUT_WIDTH-1:0] in0,in1,in2,in3;
input [1:0] sel;
output [INOUT_WIDTH-1:0] out;
assign out = (sel == 2'b00)?in0:(sel == 2'b01)?in1:(sel == 2'b10)?in2:in3;
endmodule

```

```

module LOGIC_INPUT(D,clk,CE,Q,rst);
parameter WIDTH=1;
parameter DREG=1;
parameter RSTTYPE="SYNC";
input [WIDTH-1:0] D;
input clk,rst,CE;
output reg [WIDTH-1:0] Q;

```

```

generate
    if(DREG) begin
        if(RSTTYPE == "SYNC") begin
            always @(posedge clk) begin
                if(rst)
                    Q <= 0;
                else if (CE)
                    Q <= D;
            end
        end

        else if (RSTTYPE == "ASYNC") begin
            always @(posedge clk or posedge rst) begin
                if (rst) begin
                    Q <= 0;
                end
                else if(CE) begin
                    Q <= D;
                end
            end
        end
    end
    else begin
        always @(*) begin
            Q = D;
        end
    end
endgenerate
endmodule

```



```

module Spartan6_DSP48A1(D,B,BCIN,A,C,PCIN,CLK,CARRYIN,M,P,CARRYOUT,CARRYOUTF,
                        OPMODE,CEA,CEB,CEC,CECARRYIN,CED,CEM,CEOPMODE,CEP,RSTA,RSTB,RSTC,
                        RSTCARRYIN,RSTD,RSTM,RSTOPMODE,RSTP,BCOUT,PCOUT);

parameter A0REG = 0;
parameter A1REG = 1;
parameter B0REG = 0;
parameter B1REG = 1;
parameter CREG = 1;
parameter DREG = 1;
parameter MREG = 1;
parameter PREG = 1;
parameter CARRYINREG = 1;
parameter CARRYOUTREG = 1;
parameter OPMODEREG = 1;
parameter CARRYINSEL = "OPMODE5";
parameter B_INPUT = "DIRECT" ;
parameter RSTTYPE = "SYNC" ;

input [17:0] A,B,D,BCIN;
input [47:0] PCIN,C;
input [7:0] OPMODE;
input CLK,CARRYIN,CEA,CEB,CEC,CECARRYIN,CED,CEM,CEOPMODE,
      CEP,RSTA,RSTB,RSTC,RSTCARRYIN,RSTD,RSTM,RSTOPMODE,RSTP;
output [35:0] M;
output [47:0] P,PCOUT;
output [17:0] BCOUT;
output CARRYOUT,CARRYOUTF;
wire [7:0] OPMODE_wire;
wire [17:0] IN1,A1,A2,A3,B1,C1,D1,D2;
wire [47:0] A4,G1,G2,H1,PCOUT_wire;
wire IN2,A5,H2;
wire [35:0] E1,F1;

generate
    if(B_INPUT == "DIRECT")
        assign IN1 = B;
    else if (B_INPUT == "CASCADE")
        assign IN1 = BCIN;
    else
        assign IN1 = 0;
endgenerate

generate
    if(CARRYINSEL == "OPMODE5")
        assign IN2 = OPMODE_wire[5];
    else if (CARRYINSEL == "CARRYIN")
        assign IN2 = CARRYIN;
    else
        assign IN2 = 0;
endgenerate

```

```

assign BCOUT = D1;
assign M = F1;
assign CARRYOUTF = CARRYOUT;
assign PCOUT = P;
assign PCOUT_wire = PCOUT;

pre_adder_subtractor PAS1(.A1(A1),.A2(A2),.opmode(OPMODE_wire[6]),.B1(B1));
post_adder_subtractor
PAS2(.G1(G1),.G2(G2),.opmode(OPMODE_wire[7]),.H1(H1),.CIN(A5),.COUT(H2));
multiplier MUL1(.D1(D1),.D2(D2),.E1(E1));

MUX2x1 #(.INOUT_WIDTH(18)) mux0(.in0(A2),.in1(B1),.out(C1),.sel(OPMODE_wire[4]));
MUX4x1 #(.INOUT_WIDTH(48)) muxX(.in0(48'b0),.in1({ {12{F1[35]}}}, F1[35:0]
}),.in2(PCOUT_wire),.in3({A1[11:0],D2[17:0],D1[17:0]}),.out(G1),.sel(OPMODE_wire[1:0]));
MUX4x1 #(.INOUT_WIDTH(48))
muxZ(.in0(48'b0),.in1(PCIN),.in2(PCOUT_wire),.in3(A4),.out(G2),.sel(OPMODE_wire[3:2]));

LOGIC_INPUT #(.WIDTH(18),.DREG(DREG),.RSTTYPE(RSTTYPE))
L1(.D(D),.clk(CLK),.Q(A1),.rst(RSTD),.CE(CED));
LOGIC_INPUT #(.WIDTH(18),.DREG(A0REG),.RSTTYPE(RSTTYPE))
L3(.D(A),.clk(CLK),.Q(A3),.rst(RSTA),.CE(CEA));
LOGIC_INPUT #(.WIDTH(48),.DREG(CREG),.RSTTYPE(RSTTYPE))
L4(.D(C),.clk(CLK),.Q(A4),.rst(RSTC),.CE(CEC));
LOGIC_INPUT #(.WIDTH(18),.DREG(B0REG),.RSTTYPE(RSTTYPE))
L2(.D(IN1),.clk(CLK),.Q(A2),.rst(RSTB),.CE(CEB));
LOGIC_INPUT #(.WIDTH(1),.DREG(CARRYINREG),.RSTTYPE(RSTTYPE))
L6(.D(IN2),.clk(CLK),.Q(A5),.rst(RSTCARRYIN),.CE(CECARRYIN));
LOGIC_INPUT #(.WIDTH(8),.DREG(OPMODEREG),.RSTTYPE(RSTTYPE))
L5(.D(OPMODE),.clk(CLK),.Q(OPMODE_wire),.rst(RSTOPMODE),.CE(CEOPMODE));
LOGIC_INPUT #(.WIDTH(18),.DREG(B1REG),.RSTTYPE(RSTTYPE))
L7(.D(C1),.clk(CLK),.Q(D1),.rst(RSTB),.CE(CEB));
LOGIC_INPUT #(.WIDTH(18),.DREG(A1REG),.RSTTYPE(RSTTYPE))
L8(.D(A3),.clk(CLK),.Q(D2),.rst(RSTA),.CE(CEA));
LOGIC_INPUT #(.WIDTH(36),.DREG(MREG),.RSTTYPE(RSTTYPE))
L9(.D(E1),.clk(CLK),.Q(F1),.rst(RSTM),.CE(CEM));
LOGIC_INPUT #(.WIDTH(1),.DREG(CARRYOUTREG),.RSTTYPE(RSTTYPE))
L10(.D(H2),.clk(CLK),.Q(CARRYOUT),.rst(RSTCARRYIN),.CE(CECARRYIN));
LOGIC_INPUT #(.WIDTH(48),.DREG(PREG),.RSTTYPE(RSTTYPE))
L11(.D(H1),.clk(CLK),.Q(P),.rst(RSTP),.CE(CEP));
endmodule

```

## testbench:

```
module Spartan6_DSP48A1_tb();
parameter A0REG = 0;
parameter A1REG = 1;
parameter B0REG = 0;
parameter B1REG = 1;
parameter CREG = 1;
parameter DREG = 1;
parameter MREG = 1;
parameter PREG = 1;
parameter CARRYINREG = 1;
parameter CARRYOUTREG = 1;
parameter OPMODEREG = 1;
parameter CARRYINSEL = "OPMODE5";
parameter B_INPUT = "DIRECT" ;
parameter RSTTYPE = "SYNC" ;

parameter A0REG2 = 1;
parameter A1REG2 = 1;
parameter B0REG2 = 1;
parameter B1REG2 = 1;
parameter CREG2 = 1;
parameter DREG2 = 1;
parameter MREG2 = 1;
parameter PREG2 = 1;
parameter CARRYINREG2 = 1;
parameter CARRYOUTREG2 = 1;
parameter OPMODEREG2 = 1;

reg [17:0] A,B,D,BCIN;
reg [47:0] PCIN,C;
reg [7:0] OPMODE;
reg
CLK,CARRYIN,CEA,CEB,CEC,CECARRYIN,CED,CEM,CEOPMODE,CEP,RSTA,RSTB,RSTC,RSTCARRYIN,RSTD,RSTM,RSTOPMODE,RSTP;
wire [35:0] M1,M2,M1_GOLD,M2_GOLD;
wire [47:0] P1,PCOUT1,P2,PCOUT2,P1_GOLD,PCOUT1_GOLD,P2_GOLD,PCOUT2_GOLD;
wire [17:0] BCOUT1,BCOUT2,BCOUT1_GOLD,BCOUT2_GOLD;
wire
CARRYOUT1,CARRYOUTF1,CARRYOUT2,CARRYOUTF2,CARRYOUT1_GOLD,CARRYOUTF1_GOLD,CARRYOUT2_GOLD,CARRYOUTF2_GOLD;
integer i;
```

```

Spartan6_DSP48A1 #(A0REG,A1REG,B0REG,B1REG,CREG,DREG,MREG,PREG,
    CARRYINREG,CARRYOUTREG,OPMODEREG,CARRYINSEL,B_INPUT,RSTTYPE)
    DSP1(D,B,BCIN,A,C,PCIN,CLK,CARRYIN,M1,P1,CARRYOUT1,CARRYOUTF1,OPMODE,
        CEA,CEB,CEC,CECARRYIN,CED,CEM,CEOPMODE,CEP,RSTA,RSTB,RSTC,
        RSTCARRYIN,RSTD,RSTM,RSTOPMODE,RSTP,BCOUT1,PCOUT1);

Spartan6_DSP48A1 #(A0REG2,A1REG2,B0REG2,B1REG2,CREG2,DREG2,MREG2,PREG2,
    CARRYINREG2,CARRYOUTREG2,OPMODEREG2,CARRYINSEL,B_INPUT,RSTTYPE)
    DSP3(D,B,BCIN,A,C,PCIN,CLK,CARRYIN,M2,P2,CARRYOUT2,CARRYOUTF2,OPMODE,CEA,
        CEB,CEC,CECARRYIN,CED,CEM,CEOPMODE,CEP,RSTA,RSTB,RSTC,
        RSTCARRYIN,RSTD,RSTM,RSTOPMODE,RSTP,BCOUT2,PCOUT2);

TOP_MODULE #(A0REG,A1REG,B0REG,B1REG,CREG,DREG,MREG,PREG,CARRYINREG,
    CARRYOUTREG,OPMODEREG,CARRYINSEL,B_INPUT,RSTTYPE)
    DSP1R(D,B,BCIN,A,C,PCIN,CLK,CARRYIN,M1_GOLD,P1_GOLD,CARRYOUT1_GOLD,
        CARRYOUTF1_GOLD,OPMODE,CEA,CEB,CEC,CECARRYIN,CED,CEM,CEOPMODE,CEP,RSTA,
        RSTB,RSTC,RSTCARRYIN,RSTD,RSTM,RSTOPMODE,RSTP,BCOUT1_GOLD,PCOUT1_GOLD);

TOP_MODULE #(A0REG2,A1REG2,B0REG2,B1REG2,CREG2,DREG2,MREG2,PREG2
    ,CARRYINREG2,CARRYOUTREG2,OPMODEREG2,CARRYINSEL,B_INPUT,RSTTYPE)
    DSP3R(D,B,BCIN,A,C,PCIN,CLK,CARRYIN,M2_GOLD,P2_GOLD,CARRYOUT2_GOLD,
        CARRYOUTF2_GOLD,OPMODE,CEA,CEB,CEC,CECARRYIN,CED,CEM,CEOPMODE,CEP,RSTA,RSTB,RSTC,
        RSTCARRYIN,RSTD,RSTM,RSTOPMODE,RSTP,BCOUT2_GOLD,PCOUT2_GOLD);

initial begin
    CLK=0;
    forever begin
        #1 CLK = ~CLK;
    end
end

```

```
initial begin
    A=1;
    B=2;
    D=3;
    BCIN=4;
    PCIN=5;
    C=6;
    OPMODE=7;
    RSTA=1;
    RSTB=1;
    RSTC=1;
    RSTD=1;
    RSTCARRYIN=1;
        CARRYIN=0;
    RSTM=1;
    RSTOPMODE=1;
    RSTP=1;
    CEP=0;
    CEA=0;
    CEB=0;
    CEC=0;
    CED=0;
    CECARRYIN=0;
    CEM=0;
    CEOPMODE=0;
    CEP=0;
    #2;
    CEP=1;
    CEA=1;
    CEB=1;
    CEC=1;
    CED=1;
    CECARRYIN=1;
    CEM=1;
    CEOPMODE=1;
    CEP=1;
    #2;
    RSTA=0;
    RSTB=0;
    RSTC=0;
    RSTD=0;
    RSTCARRYIN=0;
    RSTM=0;
    RSTOPMODE=0;
    RSTP=0;
    OPMODE=0;
    #2;
```

```

A=10;
B=20;
D=30;
BCIN=40;
PCIN=50;
C=60;

for(i=0;i<256;i=i+1) begin
    OPMODE=OPMODE+1;
    #8;

    $display("check time " , $time);

    if(M1 != M1_GOLD || P1 != P1_GOLD || CARRYOUT1 != CARRYOUT1_GOLD ||
BCOUT1 != BCOUT1_GOLD || PCOUT1 != PCOUT1_GOLD) begin
        $stop;
        $display("ERROR!");
    end

    if(M2 != M2_GOLD || P2 != P2_GOLD || CARRYOUT2 != CARRYOUT2_GOLD ||
BCOUT2 != BCOUT2_GOLD || PCOUT2 != PCOUT2_GOLD) begin
        $stop;
        $display("ERROR!");
    end

    #2;
end
$stop;
end
endmodule

```

## **DO FILE:**

**quit -sim**

**vlib work**

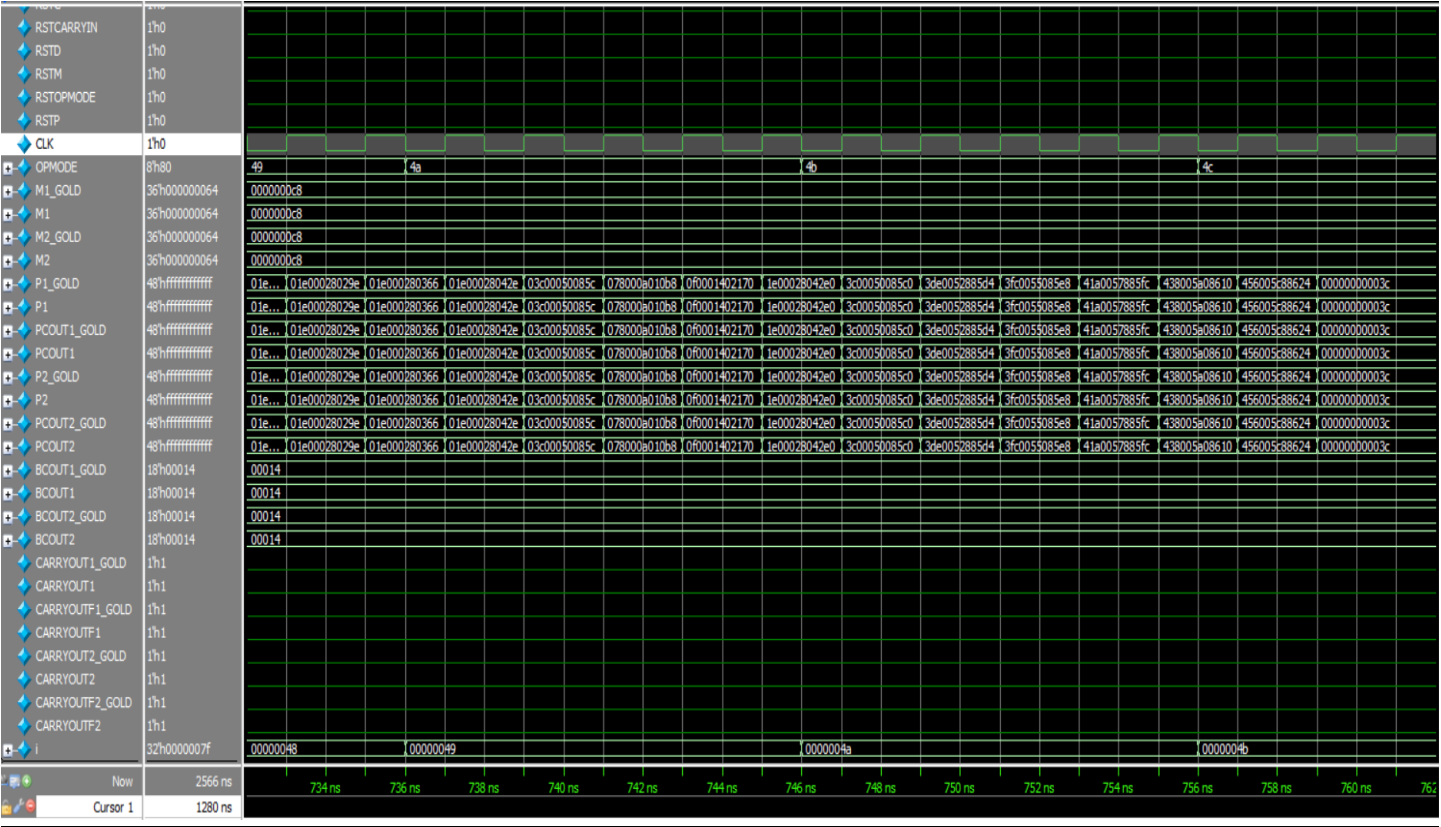
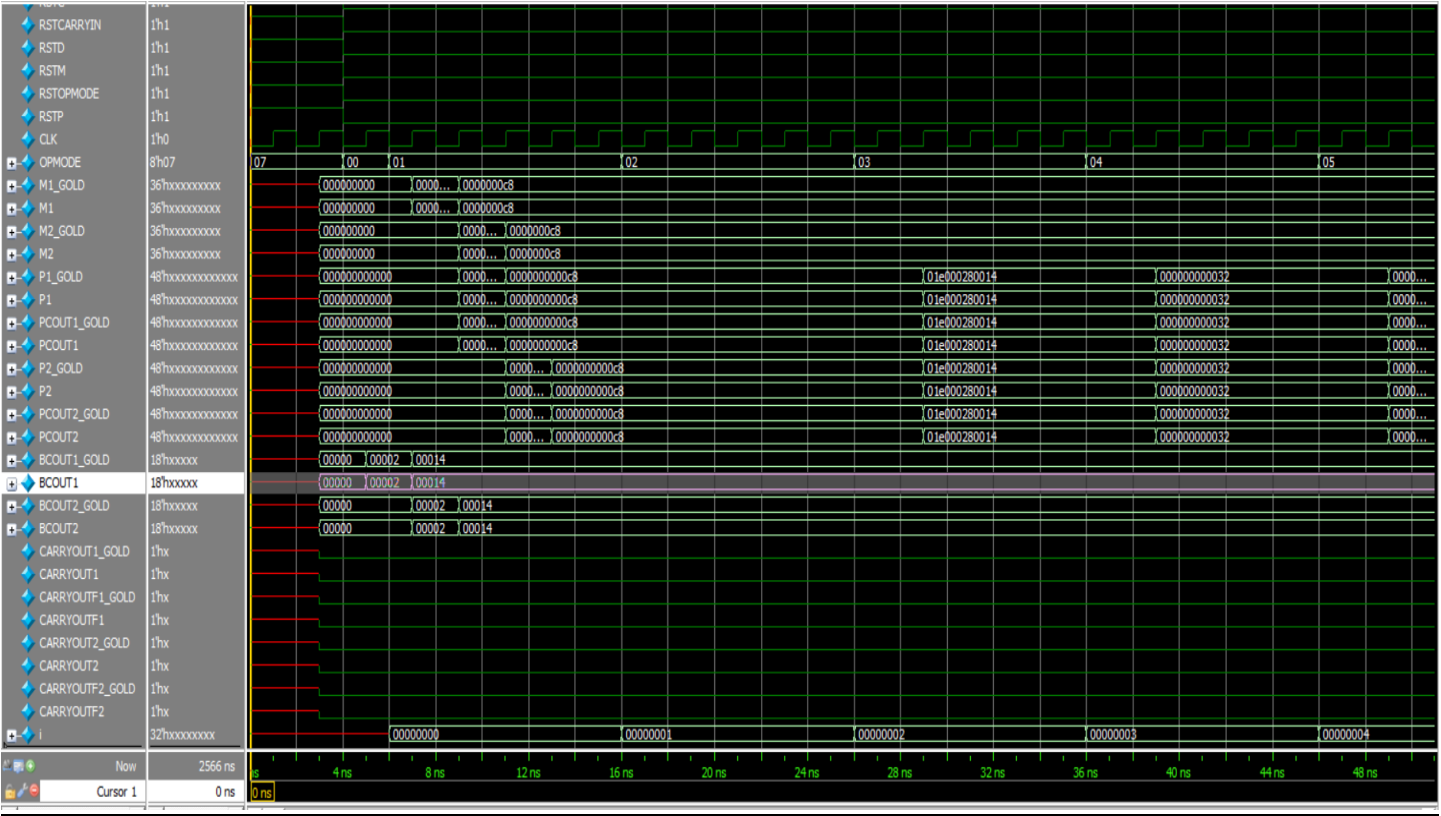
**vlog Spartan6\_DSP48A1\_tb.v Spartan6\_DSP48A1.v test1.v**

**vsim -voptargs=+acc work.Spartan6\_DSP48A1\_tb**

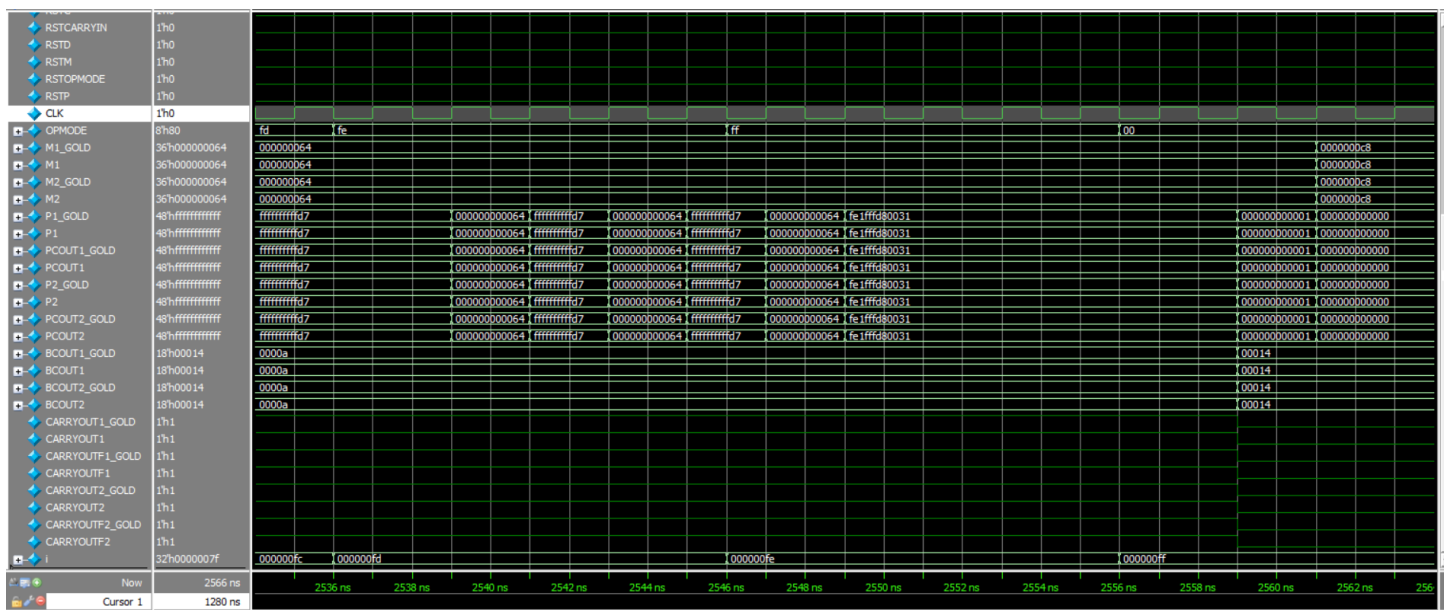
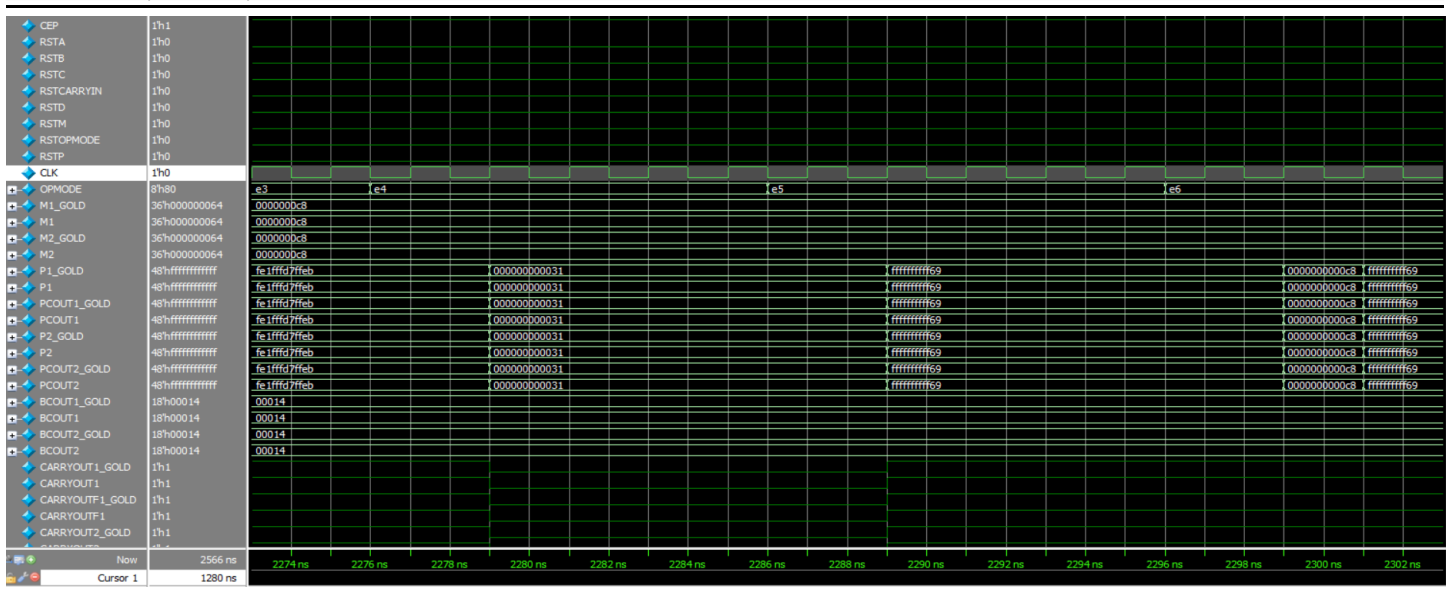
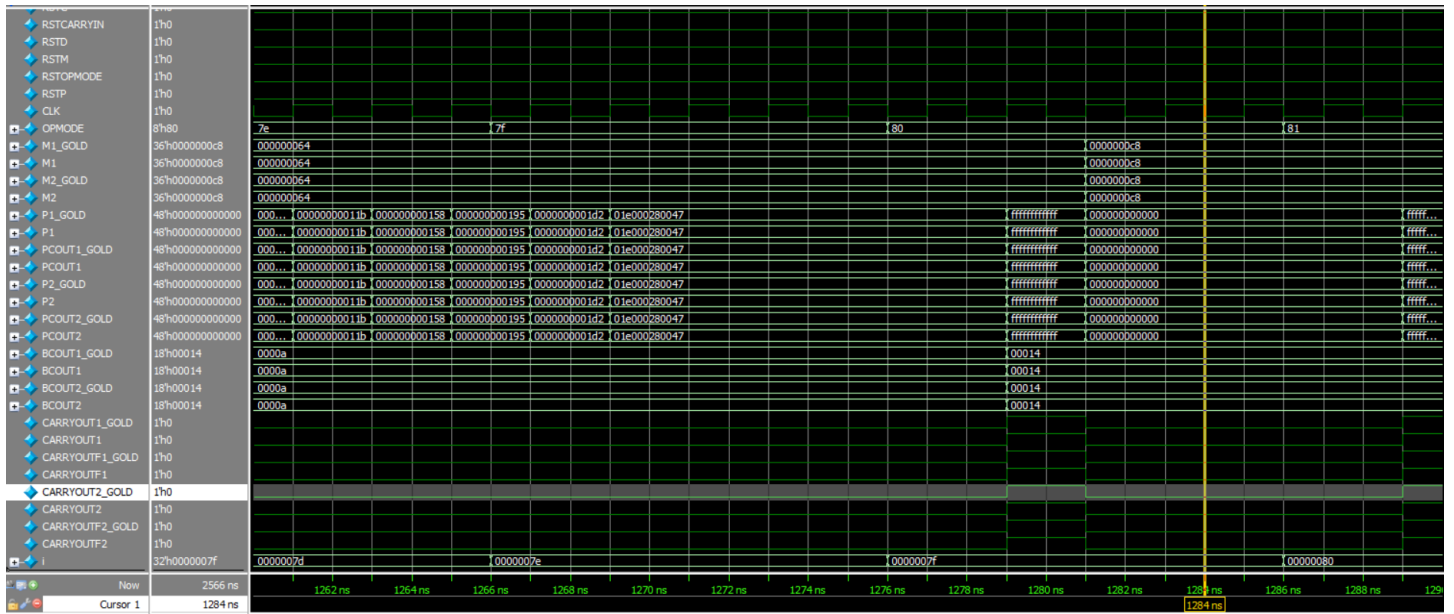
**add wave \***

**run -all**

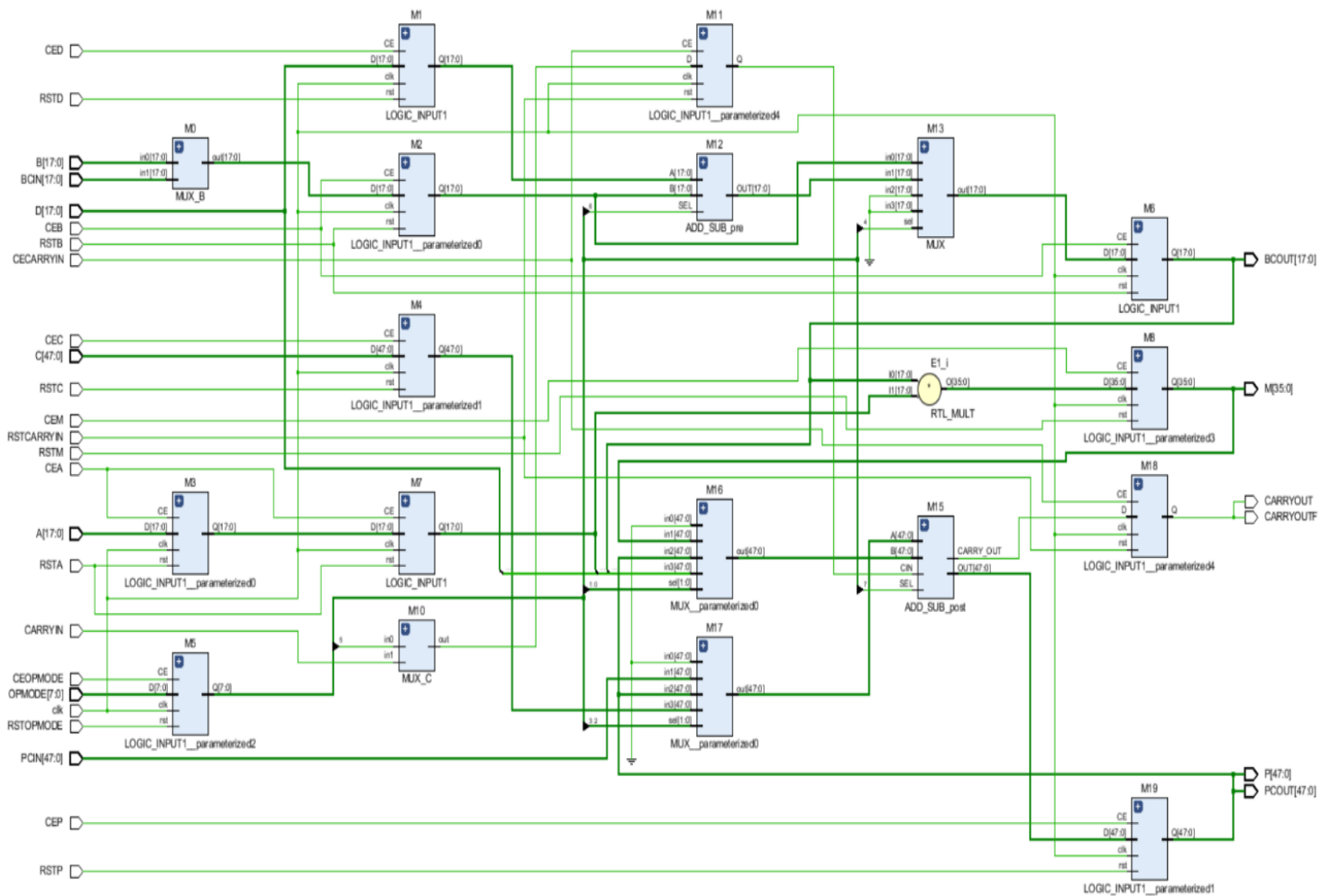
simulation:







## schematic after the elaboration:



## messages after the elaboration:

Tcl Console Messages x Log Reports Design Runs

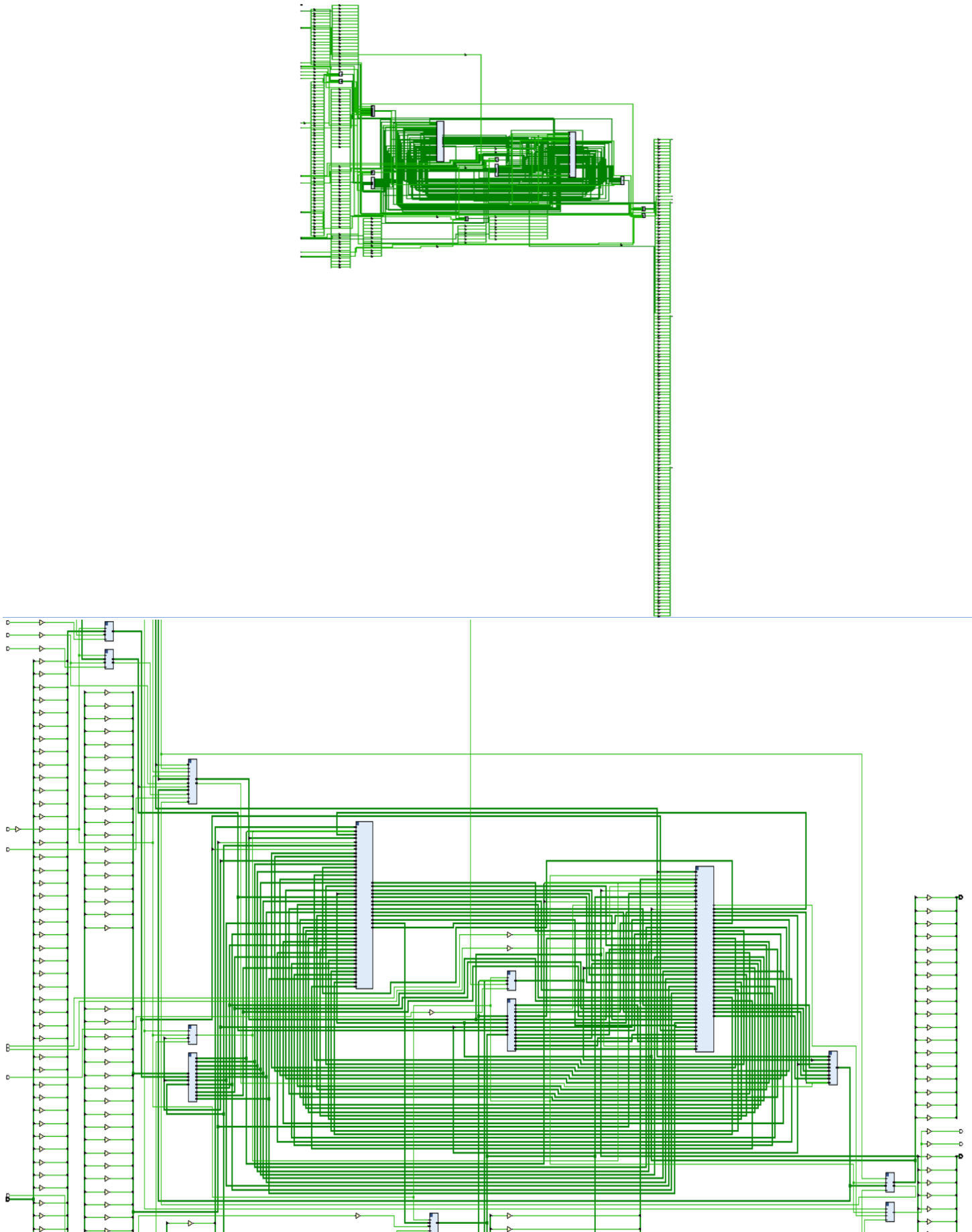
Warning (58) Info (33) Status (11) Hide All

Vivado Commands (3 infos, 6 status messages)

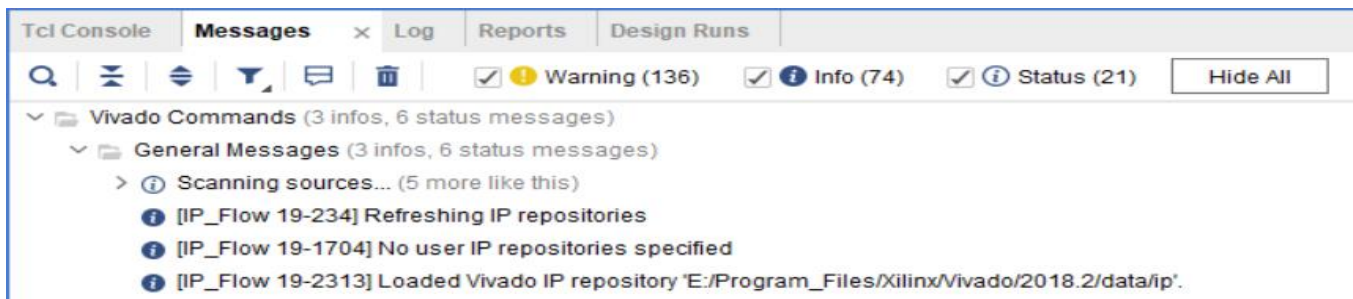
General Messages (3 infos, 6 status messages)

- Scanning sources... (5 more like this)
- [IP\_Flow 19-234] Refreshing IP repositories
- [IP\_Flow 19-1704] No user IP repositories specified
- [IP\_Flow 19-2313] Loaded Vivado IP repository 'E:/Program\_Files/Xilinx/Vivado/2018.2/data/ip'.

**schematic after the synthesis:**



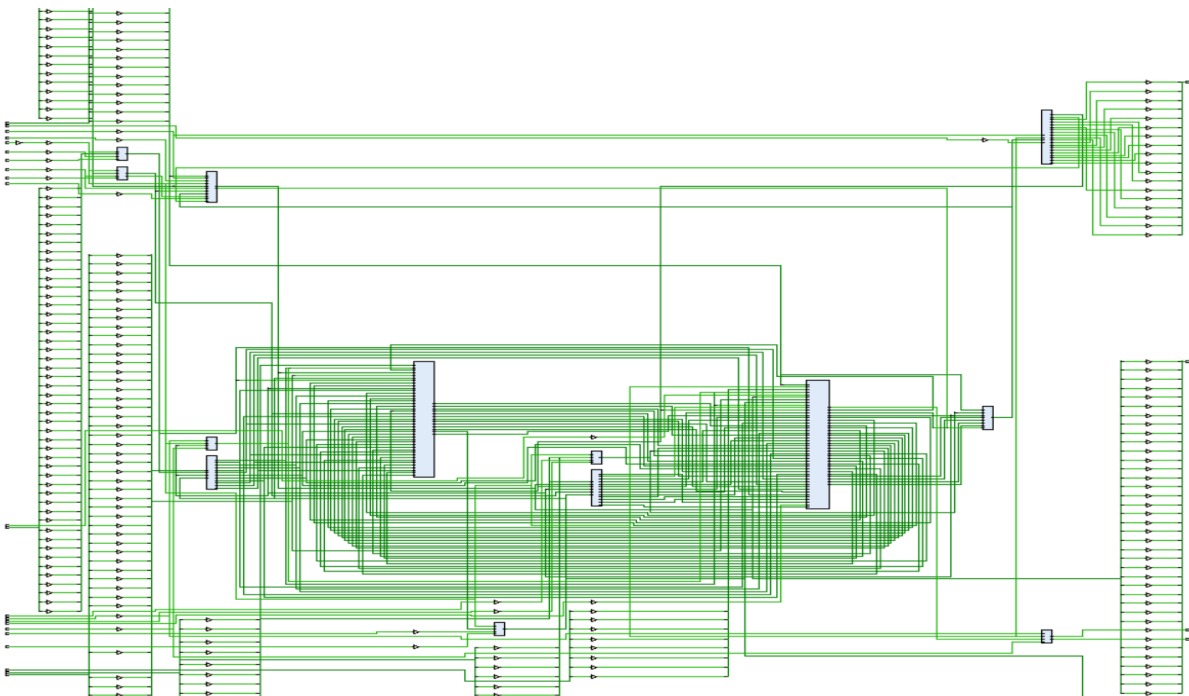
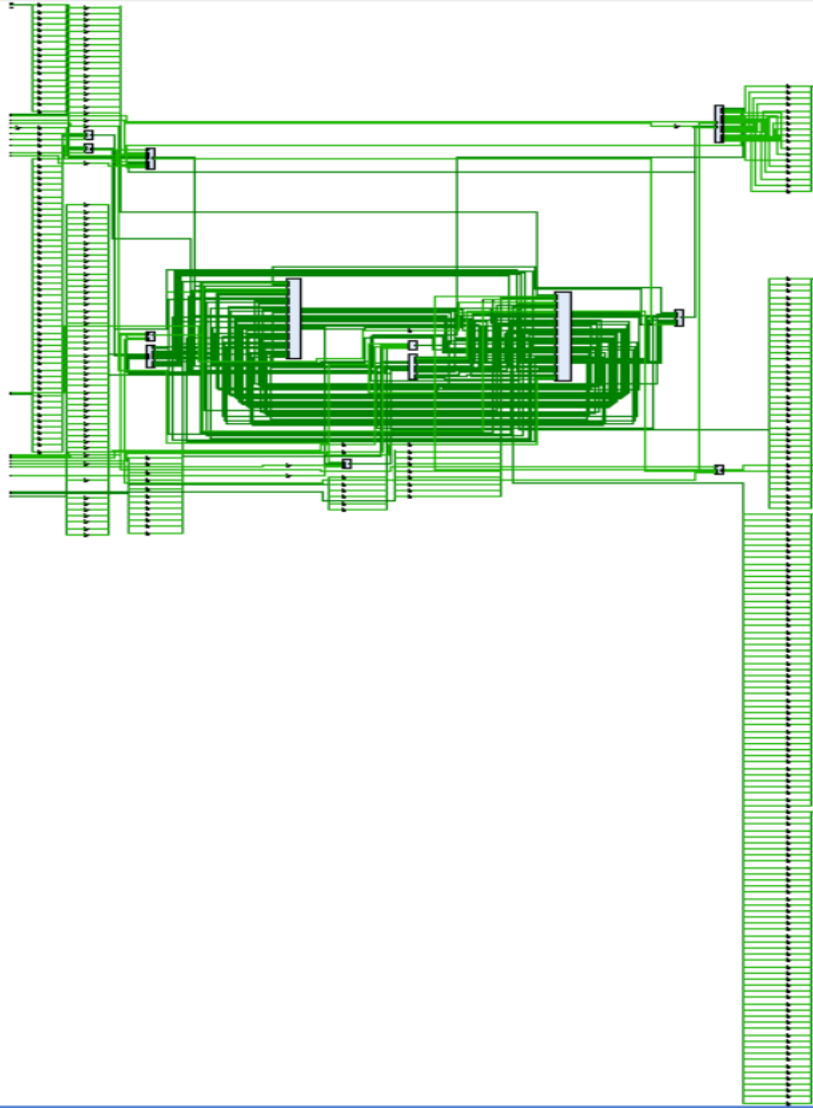
## messages after the synthesis:



The screenshot shows the Vivado Messages window. The top bar includes tabs for 'Tcl Console', 'Messages' (active), 'Log', 'Reports', and 'Design Runs'. Below the tabs is a toolbar with icons for search, expand/collapse, filter, and other actions. To the right of the toolbar, there are checkboxes and counts for 'Warning (136)', 'Info (74)', and 'Status (21)', along with a 'Hide All' button. The main area displays a tree view of messages:

- ▼ Vivado Commands (3 infos, 6 status messages)
  - ▼ General Messages (3 infos, 6 status messages)
    - > ⓘ Scanning sources... (5 more like this)
      - ⓘ [IP\_Flow 19-234] Refreshing IP repositories
      - ⓘ [IP\_Flow 19-1704] No user IP repositories specified
      - ⓘ [IP\_Flow 19-2313] Loaded Vivado IP repository 'E:/Program\_Files/Xilinx/Vivado/2018.2/data/ip'.

## schematic after the implementation:



## messages after the implementation:

Tcl Console Messages x Log Reports Design Runs

Warning (148) Info (222) Status (381) Hide All

Vivado Commands (3 infos, 6 status messages)

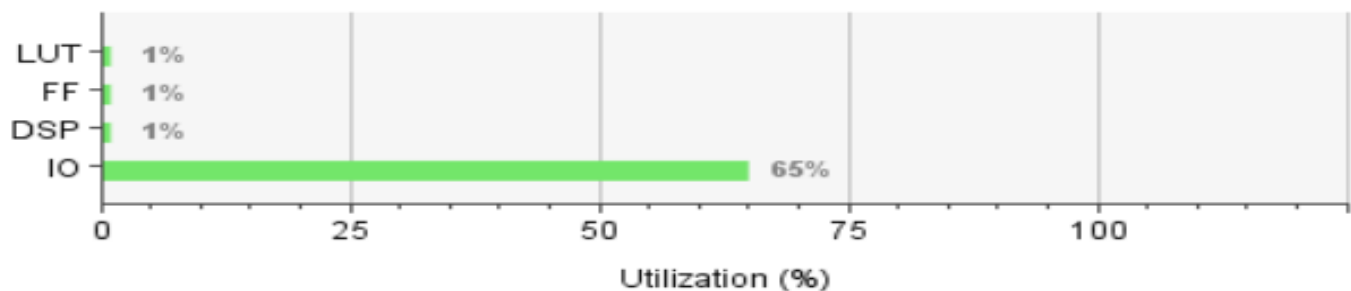
General Messages (3 infos, 6 status messages)

Scanning sources... (5 more like this)

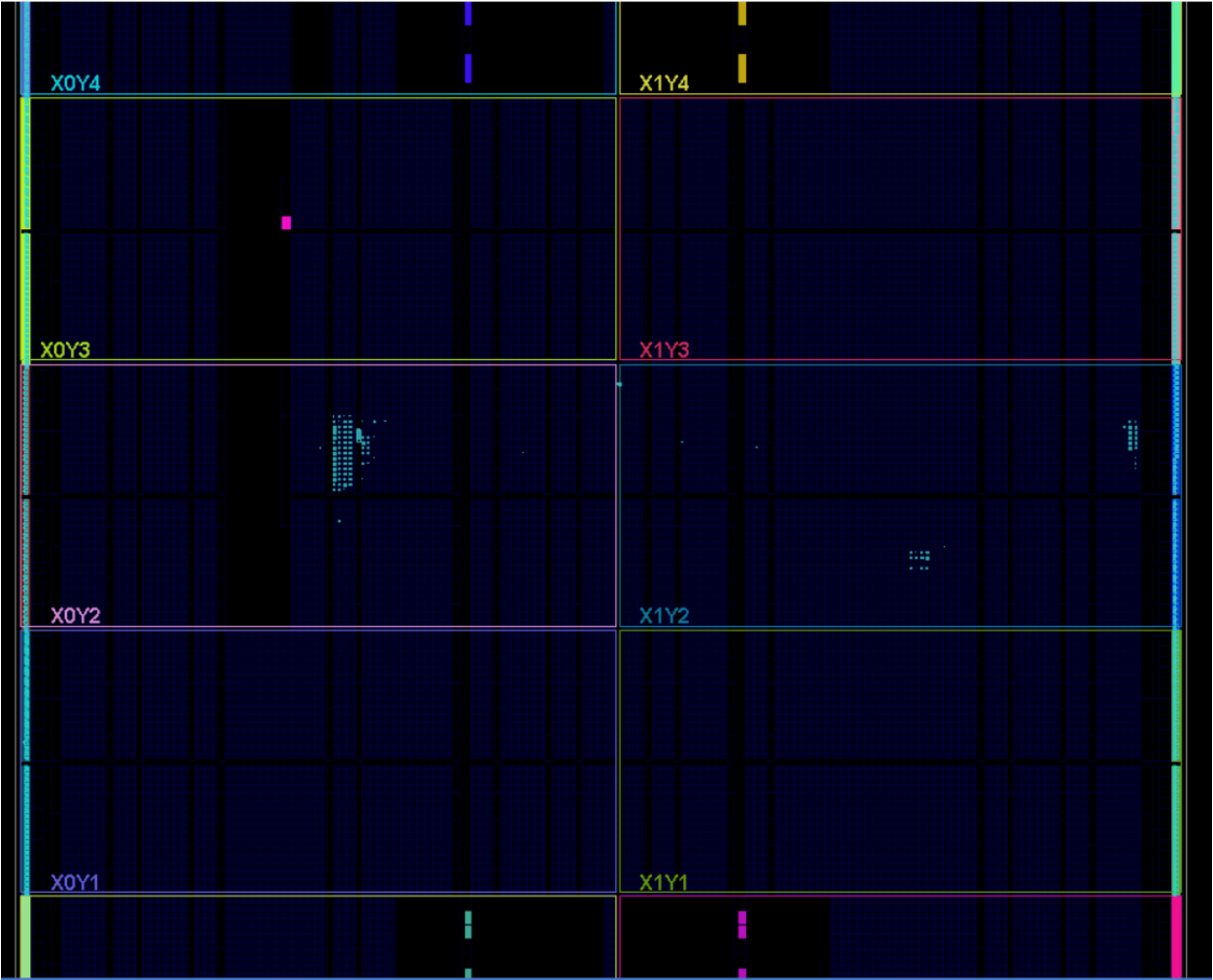
- [IP\_Flow 19-234] Refreshing IP repositories
- [IP\_Flow 19-1704] No user IP repositories specified
- [IP\_Flow 19-2313] Loaded Vivado IP repository 'E:/Program\_Files/Xilinx/Vivado/2018.2/data/ip'.

## utilization report:

Resource	Utilization	Available	Utilization %
LUT	262	134600	0.19
FF	179	269200	0.07
DSP	1	740	0.14
IO	327	500	65.40



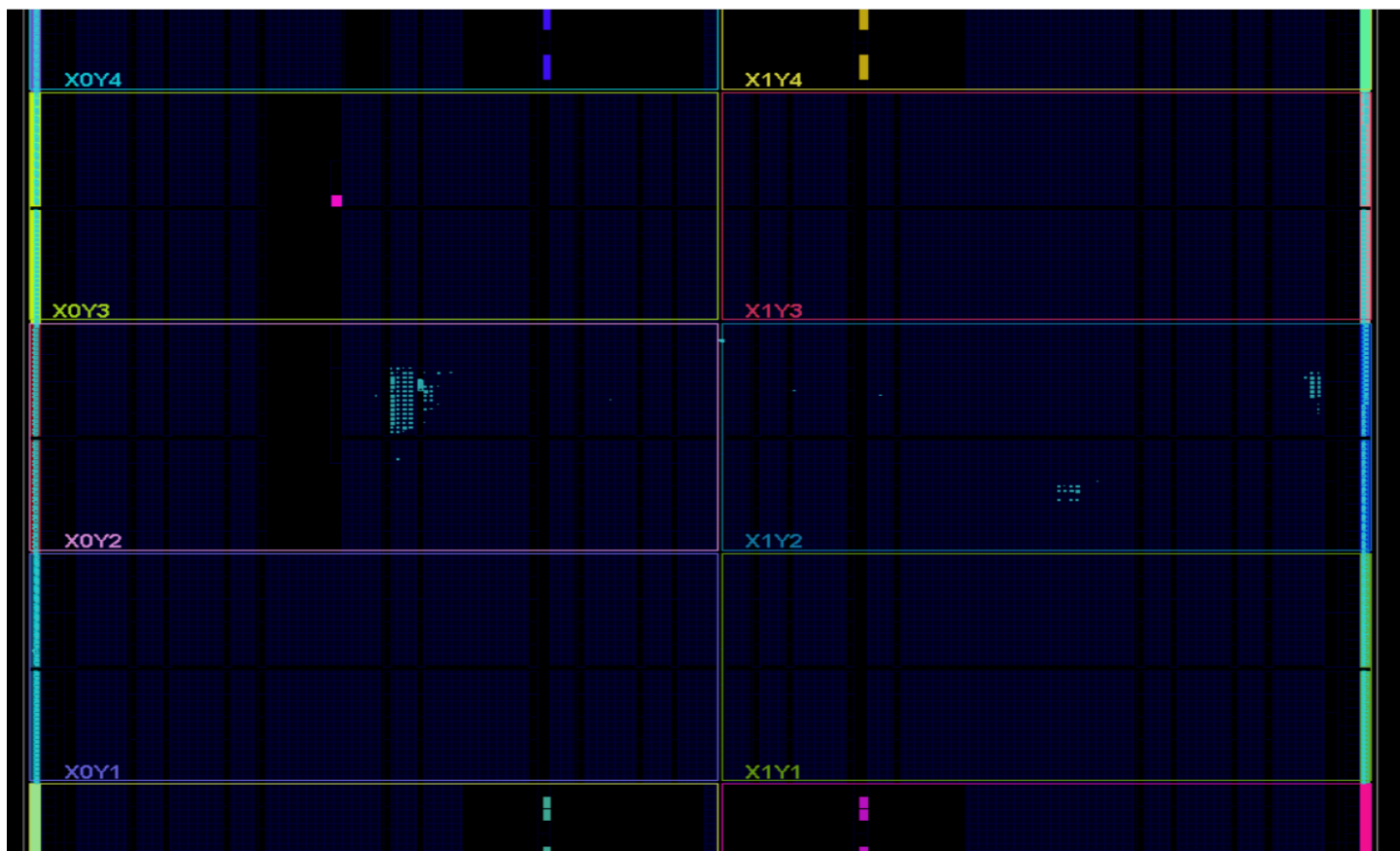
device:





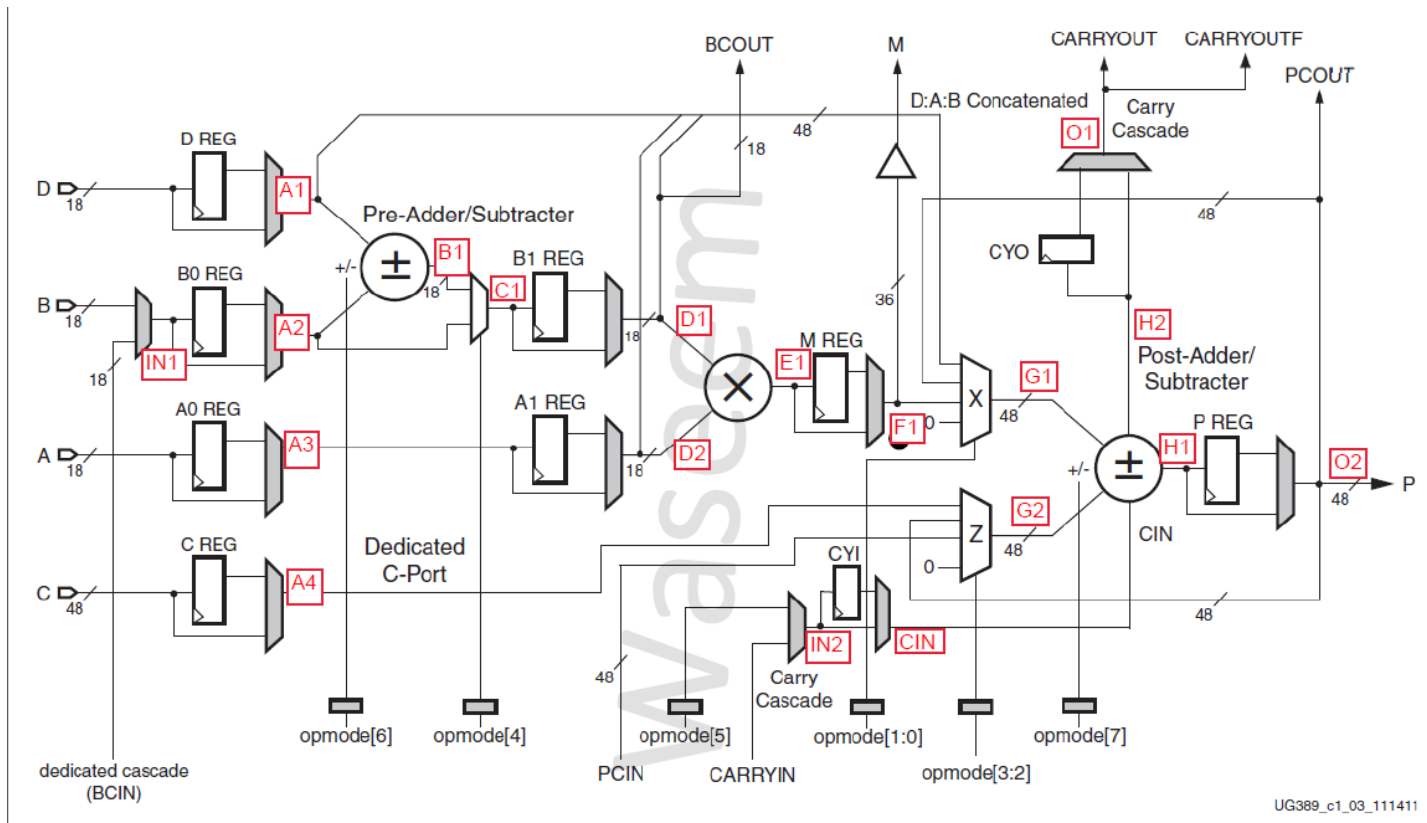


Q\_reg (DSP48E1)





## NOTES:



- These are wires names used in the two codes.
- We don't test the full combinational case,  
Bec: it may cause a combinational loop ∞