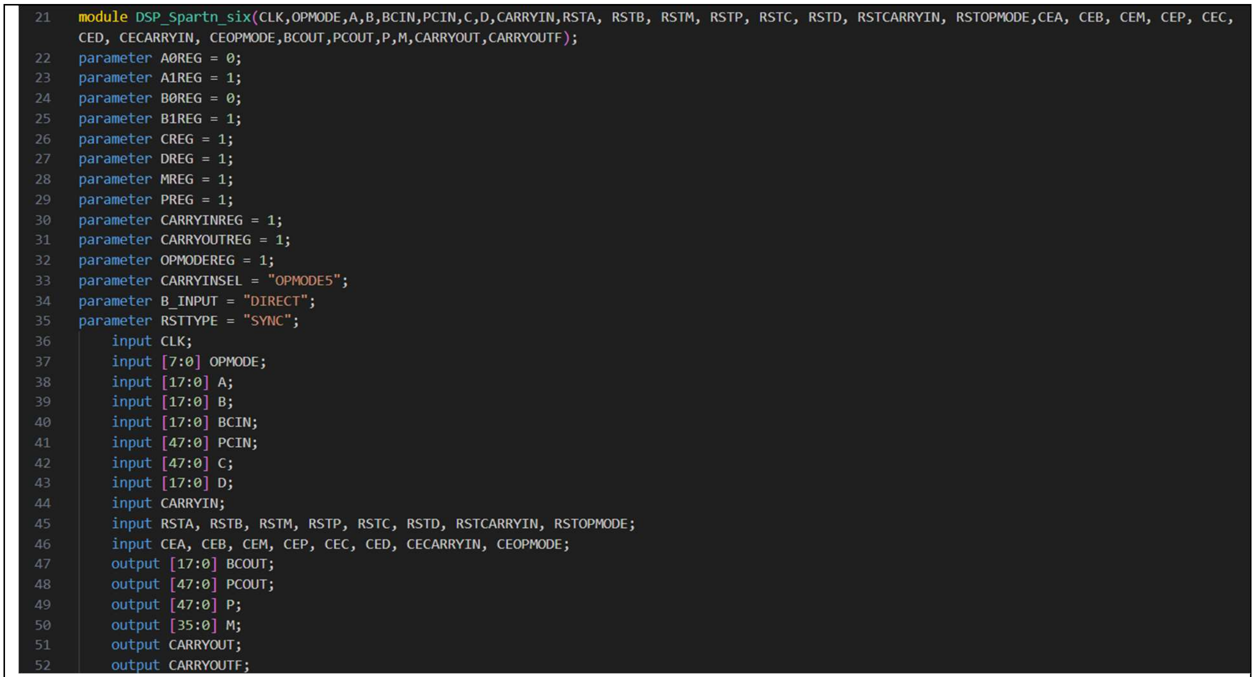
# DSP48A1

**by Ahmed Khalaf Mohamed Ali**

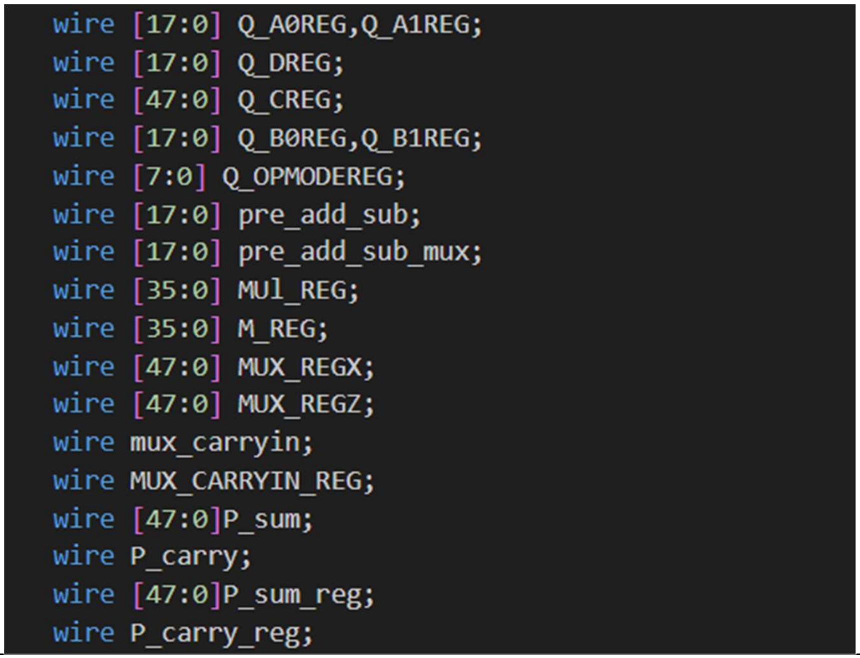
Under the guidance of **Eng. Kareem Waseem**

# DSP48A1:

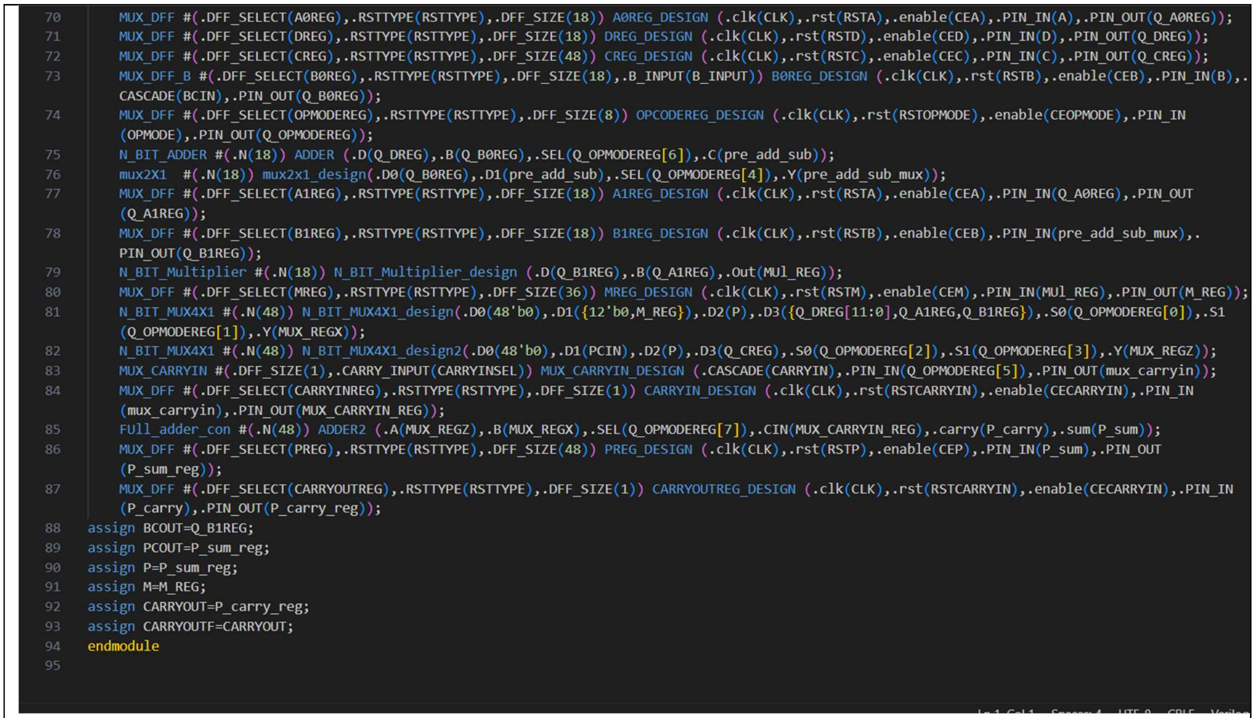
I make this project with structural modeling Make instance to each block. i will insert all instance codes in end of report



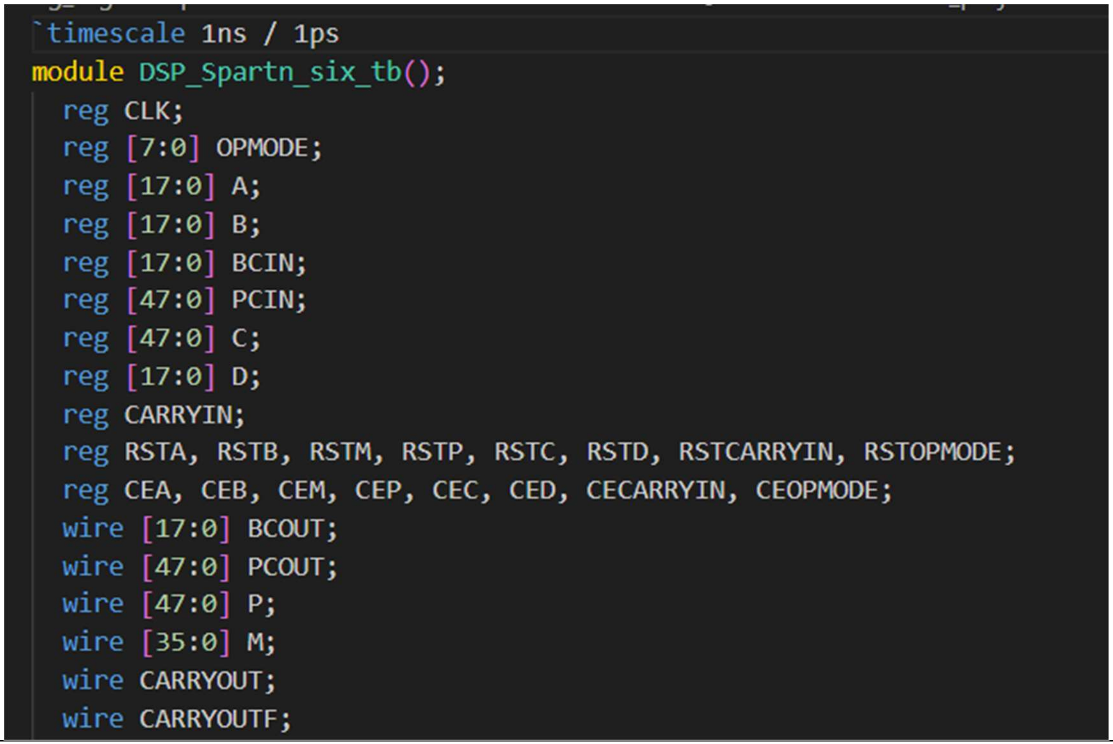
```
module DSP_Spartn_six(CLK,OPMODE,A,B,BCIN,PCIN,C,D,CARRYIN,RSTA, RSTB, RSTM, RSTP, RSTC, RSTD, RSTCARRYIN, RSTOPMODE,CEA, CEB, CEM, CEP, CEC,
CED, CECARRYIN, CEOPMODE,BCOUT,PCOUT,P,M,CARRYOUT,CARRYOUTF);
parameter A0REG = 0;
parameter A1REG = 1;
parameter B0REG = 0;
parameter B1REG = 1;
parameter 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 CLK;
    input [7:0] OPMODE;
    input [17:0] A;
    input [17:0] B;
    input [17:0] BCIN;
    input [47:0] PCIN;
    input [47:0] C;
    input [17:0] D;
    input CARRYIN;
    input RSTA, RSTB, RSTM, RSTP, RSTC, RSTD, RSTCARRYIN, RSTOPMODE;
    input CEA, CEB, CEM, CEP, CEC, CED, CECARRYIN, CEOPMODE;
    output [17:0] BCOUT;
    output [47:0] PCOUT;
    output [47:0] P;
    output [35:0] M;
    output CARRYOUT;
    output CARRYOUTF;
```



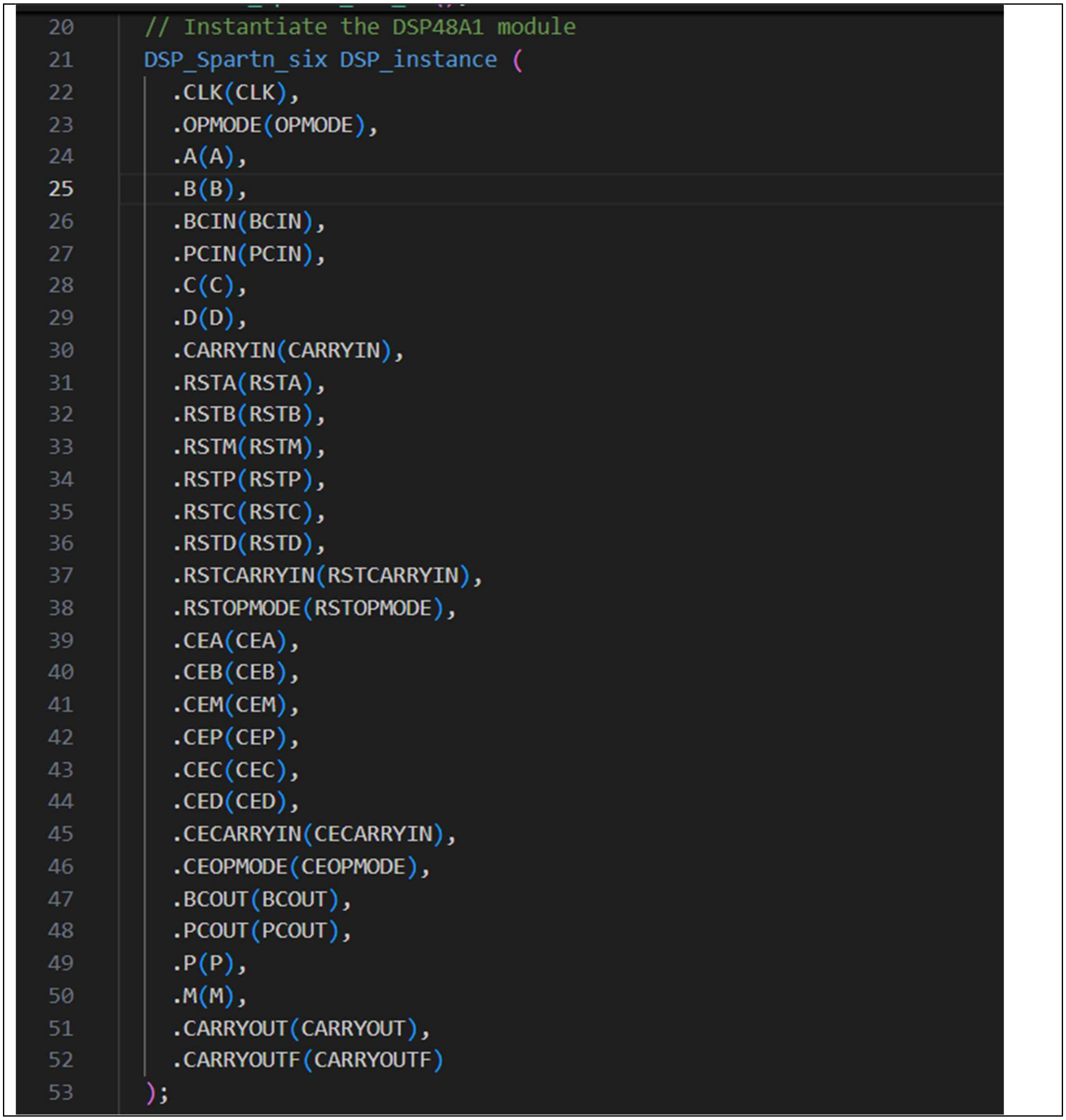
```
  wire [17:0] Q_A0REG,Q_A1REG;
  wire [17:0] Q_DREG;
  wire [47:0] Q_CREG;
  wire [17:0] Q_B0REG,Q_B1REG;
  wire [7:0] Q_OPMODEREG;
  wire [17:0] pre_add_sub;
  wire [17:0] pre_add_sub_mux;
  wire [35:0] MUl_REG;
  wire [35:0] M_REG;
  wire [47:0] MUX_REGX;
  wire [47:0] MUX_REGZ;
  wire mux_carryin;
  wire MUX_CARRYIN_REG;
  wire [47:0]P_sum;
  wire P_carry;
  wire [47:0]P_sum_reg;
  wire P_carry_reg;
```



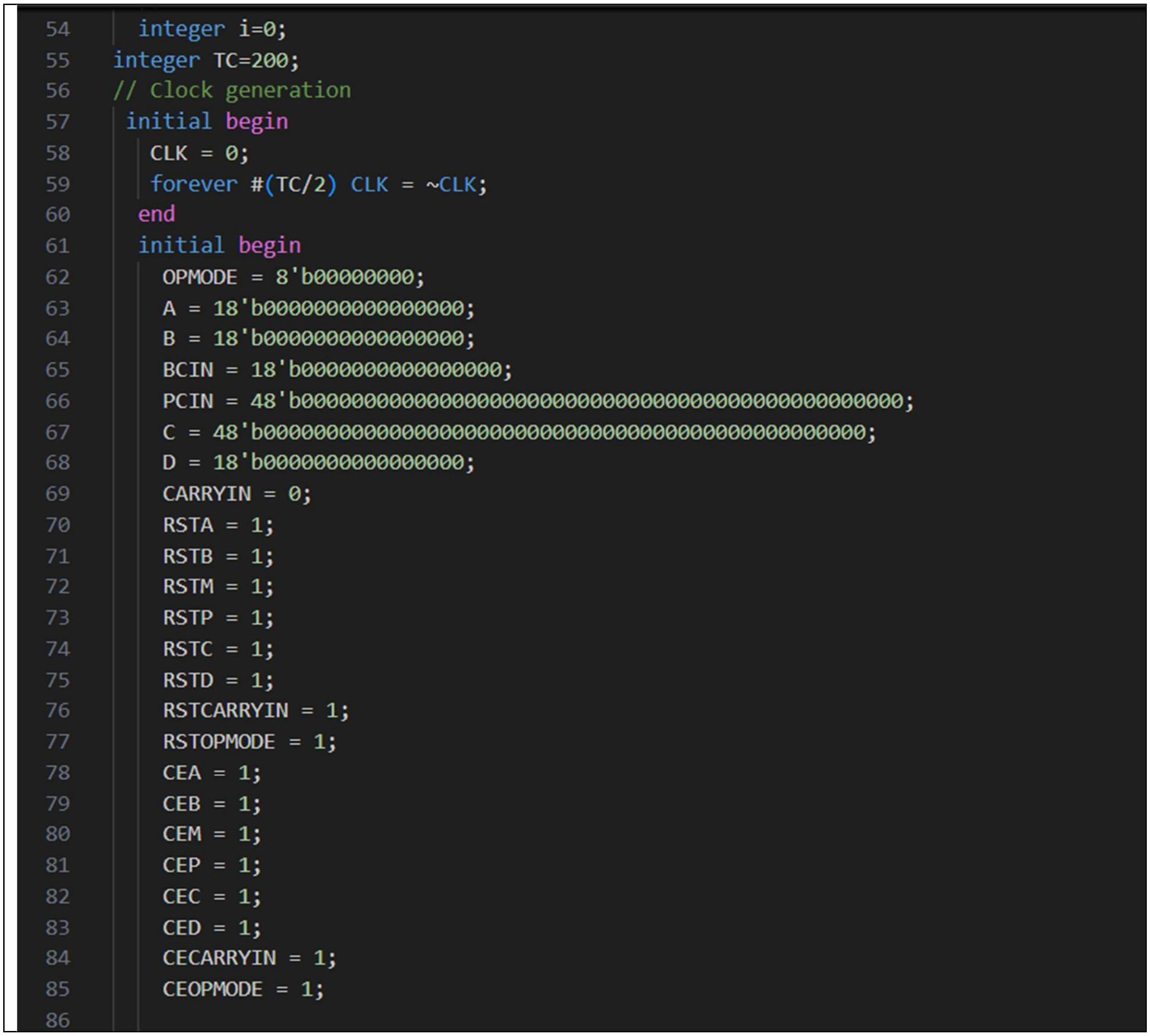
```verilog
    MUX_DFF #(.DFF_SELECT(A0REG),.RSTTYPE(RSTTYPE),.DFF_SIZE(18)) A0REG_DESIGN (.clk(CLK),.rst(RSTA),.enable(CEA),.PIN_IN(A),.PIN_OUT(Q_A0REG));
    MUX_DFF #(.DFF_SELECT(DREG),.RSTTYPE(RSTTYPE),.DFF_SIZE(18)) DREG_DESIGN (.clk(CLK),.rst(RSTD),.enable(CED),.PIN_IN(D),.PIN_OUT(Q_DREG));
    MUX_DFF #(.DFF_SELECT(CREG),.RSTTYPE(RSTTYPE),.DFF_SIZE(48)) CREG_DESIGN (.clk(CLK),.rst(RSTC),.enable(CEC),.PIN_IN(C),.PIN_OUT(Q_CREG));
    MUX_DFF_B #(.DFF_SELECT(B0REG),.RSTTYPE(RSTTYPE),.DFF_SIZE(18),.B_INPUT(B_INPUT)) B0REG_DESIGN (.clk(CLK),.rst(RSTB),.enable(CEB),.PIN_IN(B),.
    CASCADE(BCIN),.PIN_OUT(Q_B0REG));
    MUX_DFF #(.DFF_SELECT(OPMODEREG),.RSTTYPE(RSTTYPE),.DFF_SIZE(8)) OPCODEREG_DESIGN (.clk(CLK),.rst(RSTOPMODE),.enable(CEOPMODE),.PIN_IN
    (OPMODE),.PIN_OUT(Q_OPMODEREG));
    N_BIT_ADDER #(.N(18)) ADDER (.D(Q_DREG),.B(Q_B0REG),.SEL(Q_OPMODEREG[6]),.C(pre_add_sub));
    mux2X1  #(.N(18)) mux2x1_design(.D0(Q_B0REG),.D1(pre_add_sub),.SEL(Q_OPMODEREG[4]),.Y(pre_add_sub_mux));
    MUX_DFF #(.DFF_SELECT(A1REG),.RSTTYPE(RSTTYPE),.DFF_SIZE(18)) A1REG_DESIGN (.clk(CLK),.rst(RSTA),.enable(CEA),.PIN_IN(Q_A0REG),.PIN_OUT
    (Q_A1REG));
    MUX_DFF #(.DFF_SELECT(B1REG),.RSTTYPE(RSTTYPE),.DFF_SIZE(18)) B1REG_DESIGN (.clk(CLK),.rst(RSTB),.enable(CEB),.PIN_IN(pre_add_sub_mux),.
    PIN_OUT(Q_B1REG));
    N_BIT_Multiplier #(.N(18)) N_BIT_Multiplier_design (.D(Q_B1REG),.B(Q_A1REG),.Out(MUl_REG));
    MUX_DFF #(.DFF_SELECT(MREG),.RSTTYPE(RSTTYPE),.DFF_SIZE(36)) MREG_DESIGN (.clk(CLK),.rst(RSTM),.enable(CEM),.PIN_IN(MUl_REG),.PIN_OUT(M_REG));
    N_BIT_MUX4X1 #(.N(48)) N_BIT_MUX4X1_design(.D0(48'b0),.D1({12'b0,M_REG}),.D2(P),.D3({Q_DREG[11:0],Q_A1REG,Q_B1REG}),.S0(Q_OPMODEREG[0]),.S1
    (Q_OPMODEREG[1]),.Y(MUX_REGX));
    N_BIT_MUX4X1 #(.N(48)) N_BIT_MUX4X1_design2(.D0(48'b0),.D1(PCIN),.D2(P),.D3(Q_CREG),.S0(Q_OPMODEREG[2]),.S1(Q_OPMODEREG[3]),.Y(MUX_REGZ));
    MUX_CARRYIN #(.DFF_SIZE(1),.CARRY_INPUT(CARRYINSEL)) MUX_CARRYIN_DESIGN (.CASCADE(CARRYIN),.PIN_IN(Q_OPMODEREG[5]),.PIN_OUT(mux_carryin));
    MUX_DFF #(.DFF_SELECT(CARRYINREG),.RSTTYPE(RSTTYPE),.DFF_SIZE(1)) CARRYIN_DESIGN (.clk(CLK),.rst(RSTCARRYIN),.enable(CECARRYIN),.PIN_IN
    (mux_carryin),.PIN_OUT(MUX_CARRYIN_REG));
    FUll_adder_con #(.N(48)) ADDER2 (.A(MUX_REGZ),.B(MUX_REGX),.SEL(Q_OPMODEREG[7]),.CIN(MUX_CARRYIN_REG),.carry(P_carry),.sum(P_sum));
    MUX_DFF #(.DFF_SELECT(PREG),.RSTTYPE(RSTTYPE),.DFF_SIZE(48)) PREG_DESIGN (.clk(CLK),.rst(RSTP),.enable(CEP),.PIN_IN(P_sum),.PIN_OUT
    (P_sum_reg));
    MUX_DFF #(.DFF_SELECT(CARRYOUTREG),.RSTTYPE(RSTTYPE),.DFF_SIZE(1)) CARRYOUTREG_DESIGN (.clk(CLK),.rst(RSTCARRYIN),.enable(CECARRYIN),.PIN_IN
    (P_carry),.PIN_OUT(P_carry_reg));
assign BCOUT=Q_B1REG;
assign PCOUT=P_sum_reg;
assign P=P_sum_reg;
assign M=M_REG;
assign CARRYOUT=P_carry_reg;
assign CARRYOUTF=CARRYOUT;
endmodule
```



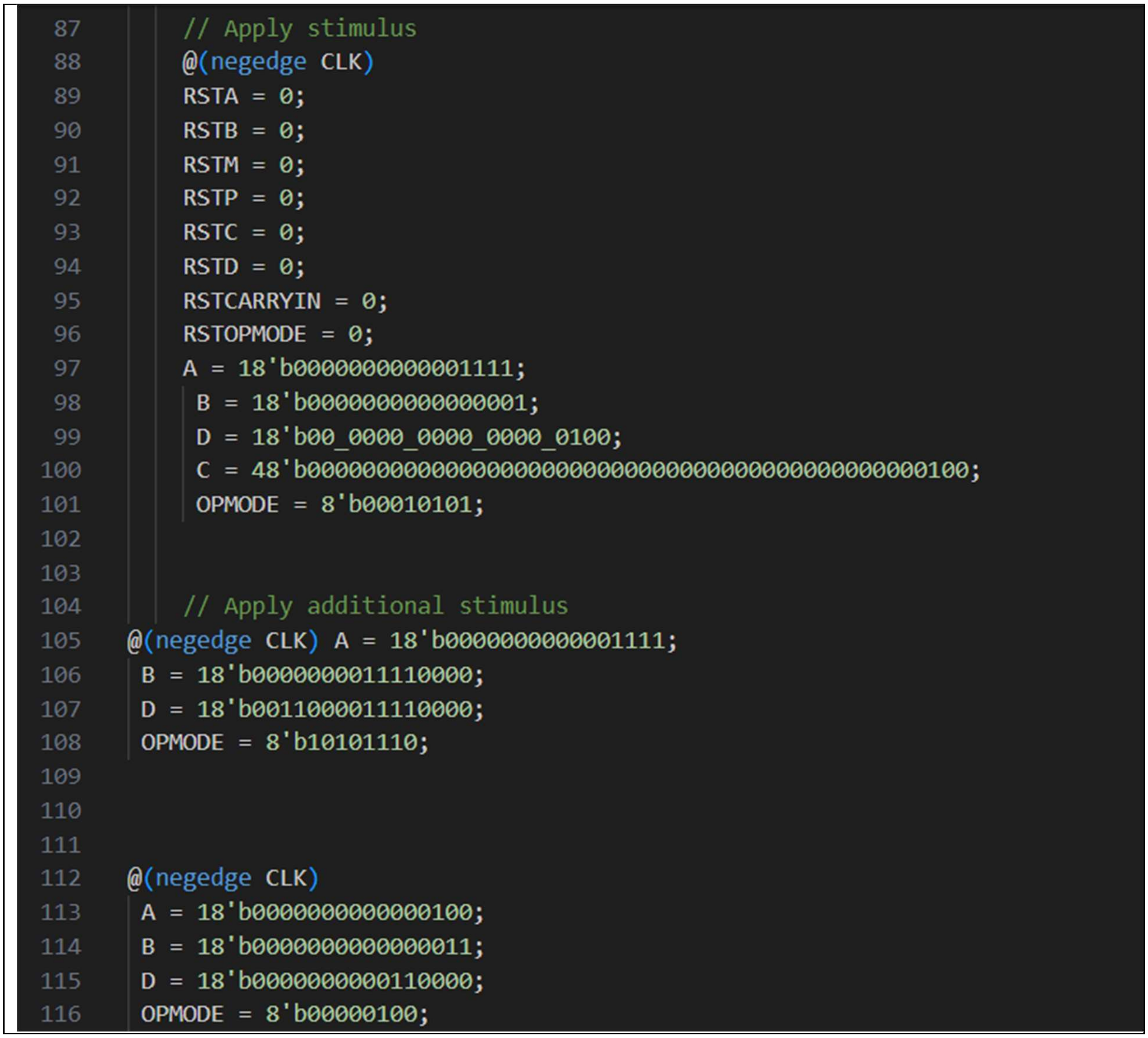
```verilog
`timescale 1ns / 1ps
module DSP_Spartn_six_tb();
  reg CLK;
  reg [7:0] OPMODE;
  reg [17:0] A;
  reg [17:0] B;
  reg [17:0] BCIN;
  reg [47:0] PCIN;
  reg [47:0] C;
  reg [17:0] D;
  reg CARRYIN;
  reg RSTA, RSTB, RSTM, RSTP, RSTC, RSTD, RSTCARRYIN, RSTOPMODE;
  reg CEA, CEB, CEM, CEP, CEC, CED, CECARRYIN, CEOPMODE;
  wire [17:0] BCOUT;
  wire [47:0] PCOUT;
  wire [47:0] P;
  wire [35:0] M;
  wire CARRYOUT;
  wire CARRYOUTF;
```



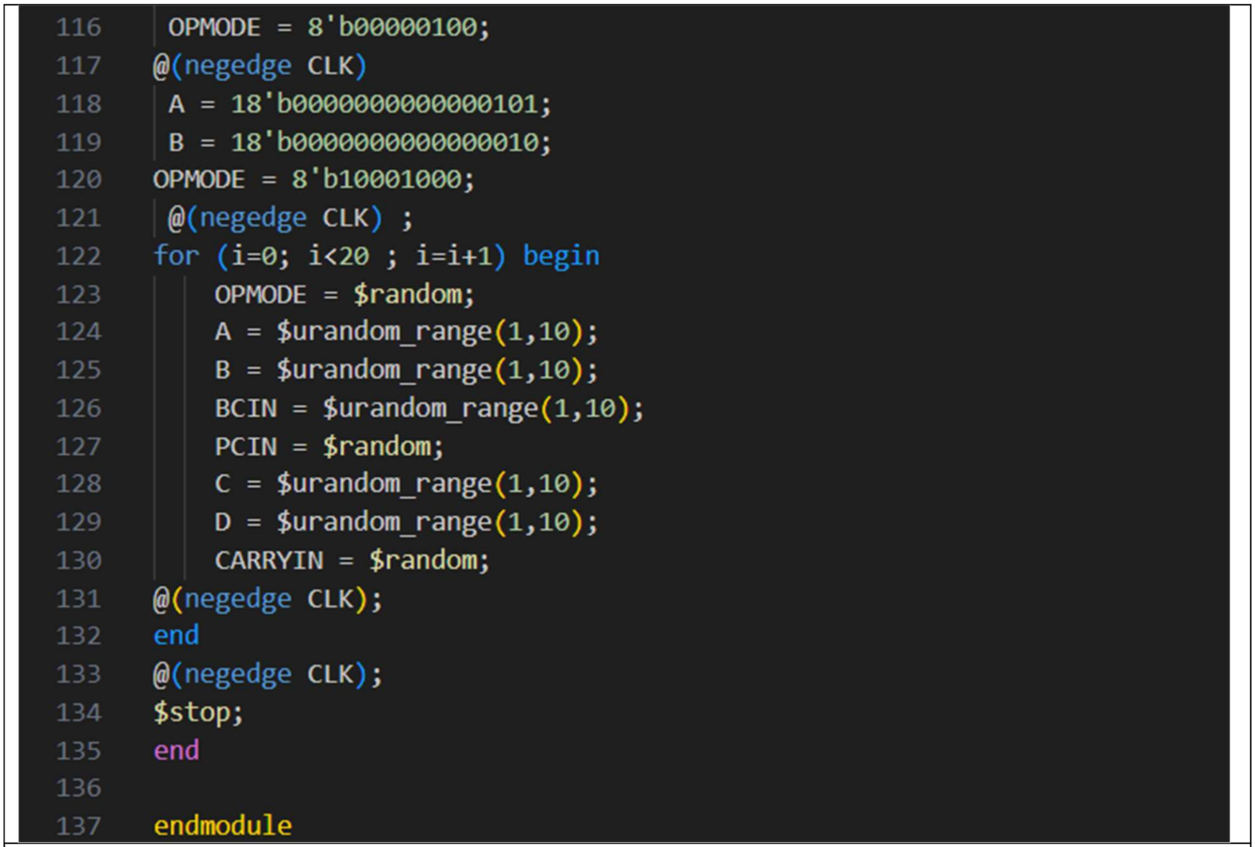
```verilog
// Instantiate the DSP48A1 module
DSP_Spartn_six DSP_instance (
  .CLK(CLK),
  .OPMODE(OPMODE),
  .A(A),
  .B(B),
  .BCIN(BCIN),
  .PCIN(PCIN),
  .C(C),
  .D(D),
  .CARRYIN(CARRYIN),
  .RSTA(RSTA),
  .RSTB(RSTB),
  .RSTM(RSTM),
  .RSTP(RSTP),
  .RSTC(RSTC),
  .RSTD(RSTD),
  .RSTCARRYIN(RSTCARRYIN),
  .RSTOPMODE(RSTOPMODE),
  .CEA(CEA),
  .CEB(CEB),
  .CEM(CEM),
  .CEP(CEP),
  .CEC(CEC),
  .CED(CED),
  .CECARRYIN(CECARRYIN),
  .CEOPMODE(CEOPMODE),
  .BCOUT(BCOUT),
  .PCOUT(PCOUT),
  .P(P),
  .M(M),
  .CARRYOUT(CARRYOUT),
  .CARRYOUTF(CARRYOUTF)
);
```



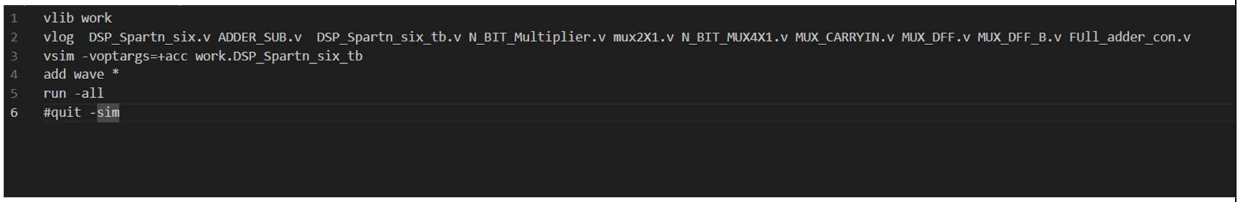
```
  integer i=0;
integer TC=200;
// Clock generation
 initial begin
   CLK = 0;
   forever #(TC/2) CLK = ~CLK;
  end
  initial begin
    OPMODE = 8'b00000000;
    A = 18'b0000000000000000;
    B = 18'b0000000000000000;
    BCIN = 18'b0000000000000000;
    PCIN = 48'b000000000000000000000000000000000000000000000000;
    C = 48'b000000000000000000000000000000000000000000000000;
    D = 18'b0000000000000000;
    CARRYIN = 0;
    RSTA = 1;
    RSTB = 1;
    RSTM = 1;
    RSTP = 1;
    RSTC = 1;
    RSTD = 1;
    RSTCARRYIN = 1;
    RSTOPMODE = 1;
    CEA = 1;
    CEB = 1;
    CEM = 1;
    CEP = 1;
    CEC = 1;
    CED = 1;
    CECARRYIN = 1;
    CEOPMODE = 1;

```



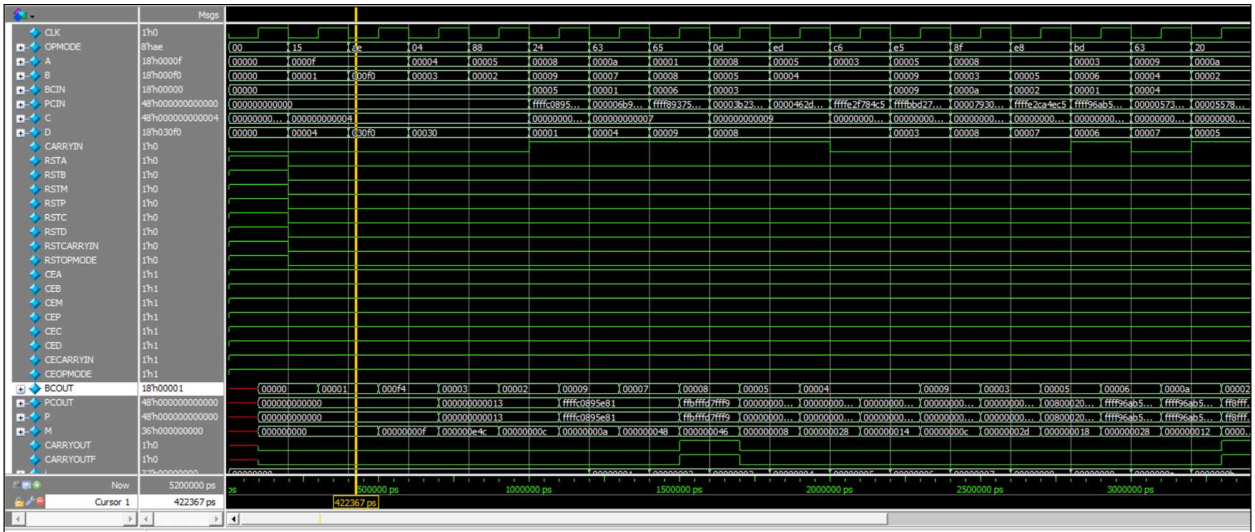
```
    // Apply stimulus
    @(negedge CLK)
    RSTA = 0;
    RSTB = 0;
    RSTM = 0;
    RSTP = 0;
    RSTC = 0;
    RSTD = 0;
    RSTCARRYIN = 0;
    RSTOPMODE = 0;
    A = 18'b0000000000001111;
     B = 18'b0000000000000001;
     D = 18'b00_0000_0000_0000_0100;
     C = 48'b000000000000000000000000000000000000000000000100;
     OPMODE = 8'b00010101;


    // Apply additional stimulus
@(negedge CLK) A = 18'b0000000000001111;
 B = 18'b0000000011110000;
 D = 18'b0011000011110000;
 OPMODE = 8'b10101110;



@(negedge CLK)
 A = 18'b0000000000000100;
 B = 18'b0000000000000011;
 D = 18'b0000000000110000;
 OPMODE = 8'b00000100;
```

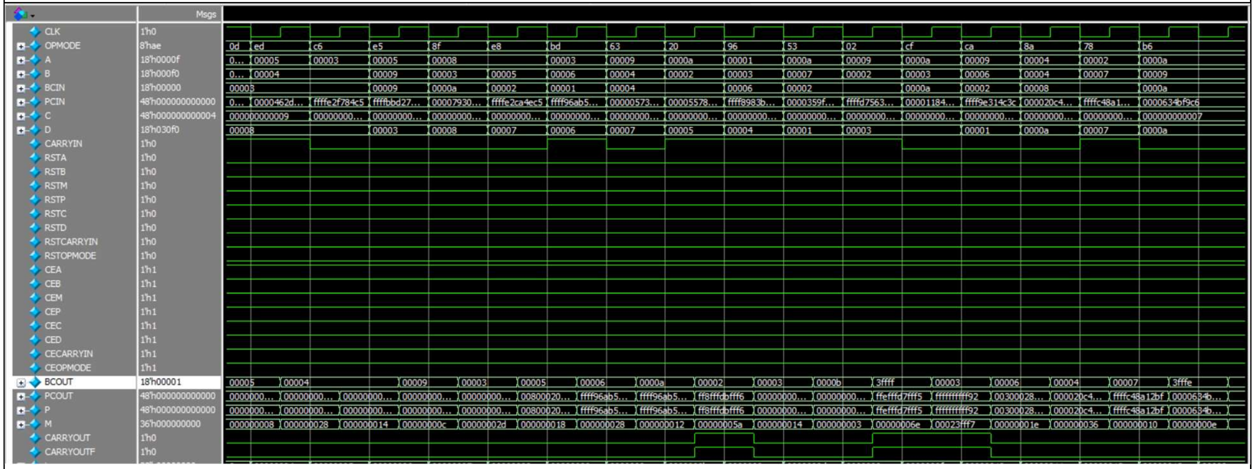


```verilog
 OPMODE = 8'b00000100;
@(negedge CLK)
 A = 18'b000000000000000101;
 B = 18'b000000000000000010;
OPMODE = 8'b10001000;
 @(negedge CLK) ;
for (i=0; i<20 ; i=i+1) begin
    OPMODE = $random;
    A = $urandom_range(1,10);
    B = $urandom_range(1,10);
    BCIN = $urandom_range(1,10);
    PCIN = $random;
    C = $urandom_range(1,10);
    D = $urandom_range(1,10);
    CARRYIN = $random;
@(negedge CLK);
end
@(negedge CLK);
$stop;
end

endmodule
```

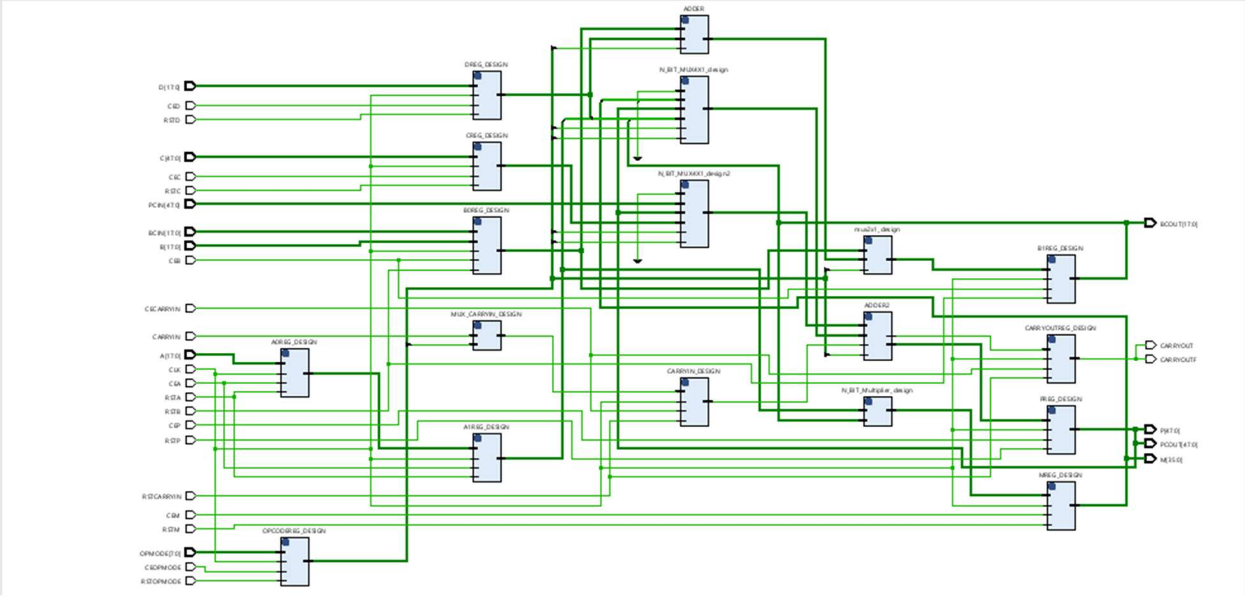


```
vlib work
vlog  DSP_Spartn_six.v ADDER_SUB.v  DSP_Spartn_six_tb.v N_BIT_Multiplier.v mux2X1.v N_BIT_MUX4X1.v MUX_CARRYIN.v MUX_DFF.v MUX_DFF_B.v FUll_adder_con.v
vsim -voptargs=+acc work.DSP_Spartn_six_tb
add wave *
run -all
#quit -sim
```

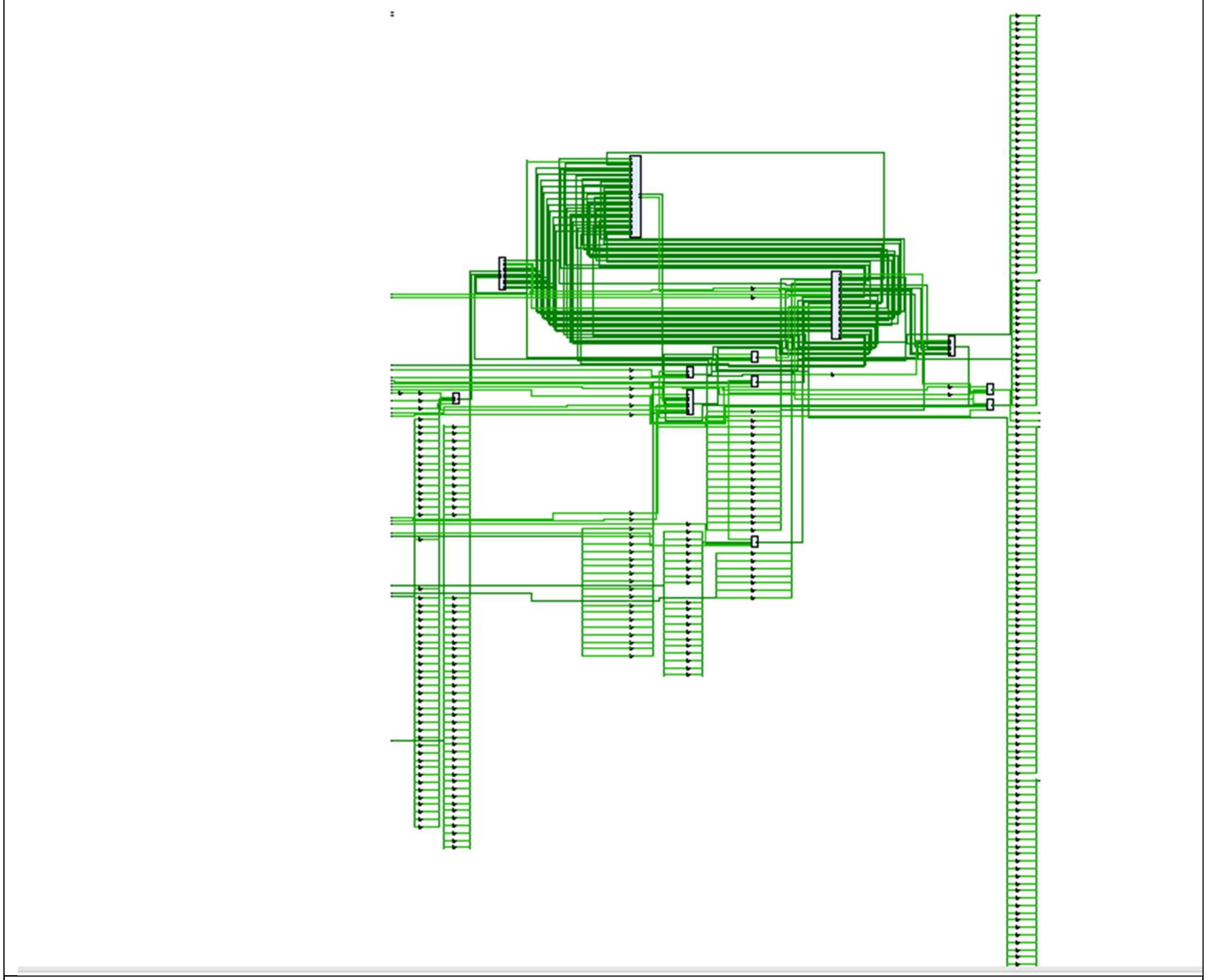

Msgs
CLK 1'h0
OPMODE 8'hae
A 18'h0000f
B 18'h000f0
BCIN 18'h00000
PCIN 48'h000000000000
C 48'h000000000004
D 18'h030f0
CARRYIN 1'h0
RSTA 1'h0
RSTB 1'h0
RSTM 1'h0
RSTP 1'h0
RSTC 1'h0
RSTD 1'h0
RSTCARRYIN 1'h0
RSTOPMODE 1'h0
CEA 1'h1
CEB 1'h1
CEM 1'h1
CEP 1'h1
CEC 1'h1
CED 1'h1
CECARRYIN 1'h1
CEOPMODE 1'h1
BCOUT 18'h00001
PCOUT 48'h000000000000
P 48'h000000000000
M 36'h000000000
CARRYOUT 1'h0
CARRYOUTF 1'h0
Now 5200000 ps
Cursor 1 422367 ps
500000 ps
1000000 ps
1500000 ps
2000000 ps
2500000 ps
3000000 ps
422367 ps


Msgs
CLK 1'h0
OPMODE 8'hae
A 18'h0000f
B 18'h000f0
BCIN 18'h00000
PCIN 48'h000000000000
C 48'h000000000004
D 18'h030f0
CARRYIN 1'h0
RSTA 1'h0
RSTB 1'h0
RSTM 1'h0
RSTP 1'h0
RSTC 1'h0
RSTD 1'h0
RSTCARRYIN 1'h0
RSTOPMODE 1'h0
CEA 1'h1
CEB 1'h1
CEM 1'h1
CEP 1'h1
CEC 1'h1
CED 1'h1
CECARRYIN 1'h1
CEOPMODE 1'h1
BCOUT 18'h00001
PCOUT 48'h000000000000
P 48'h000000000000
M 36'h000000000
CARRYOUT 1'h0
CARRYOUTF 1'h0

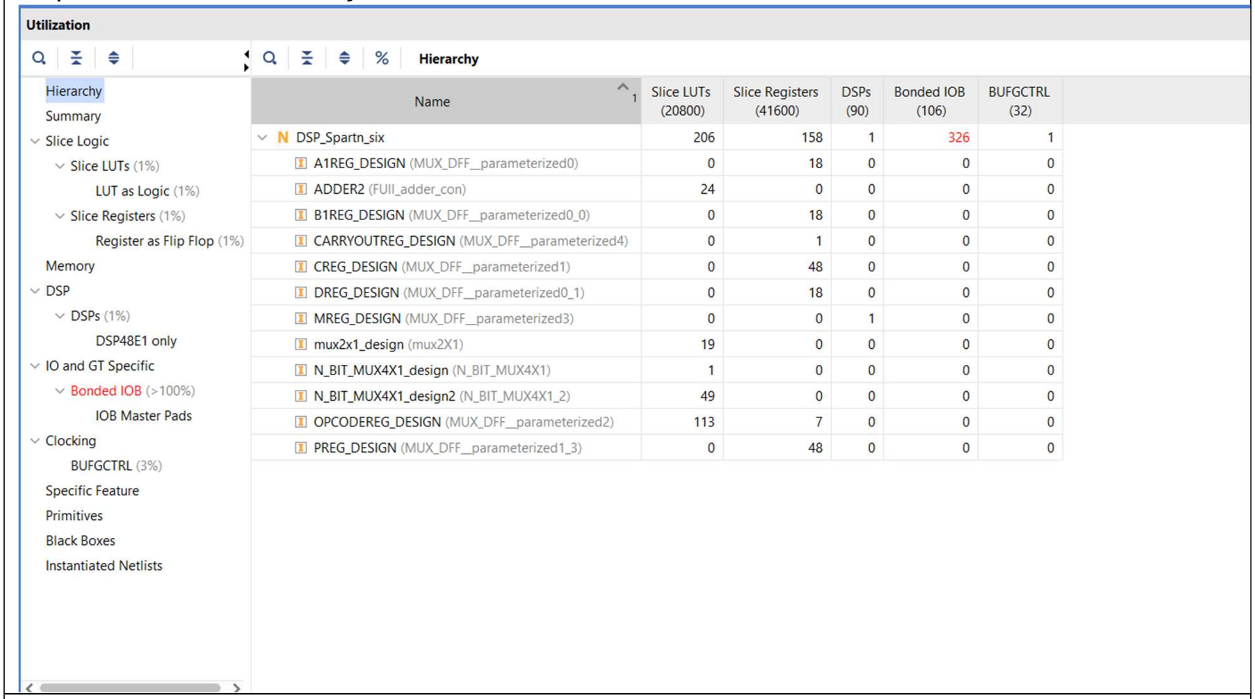
Schematic after elaboration:



Schematic after synthesis:

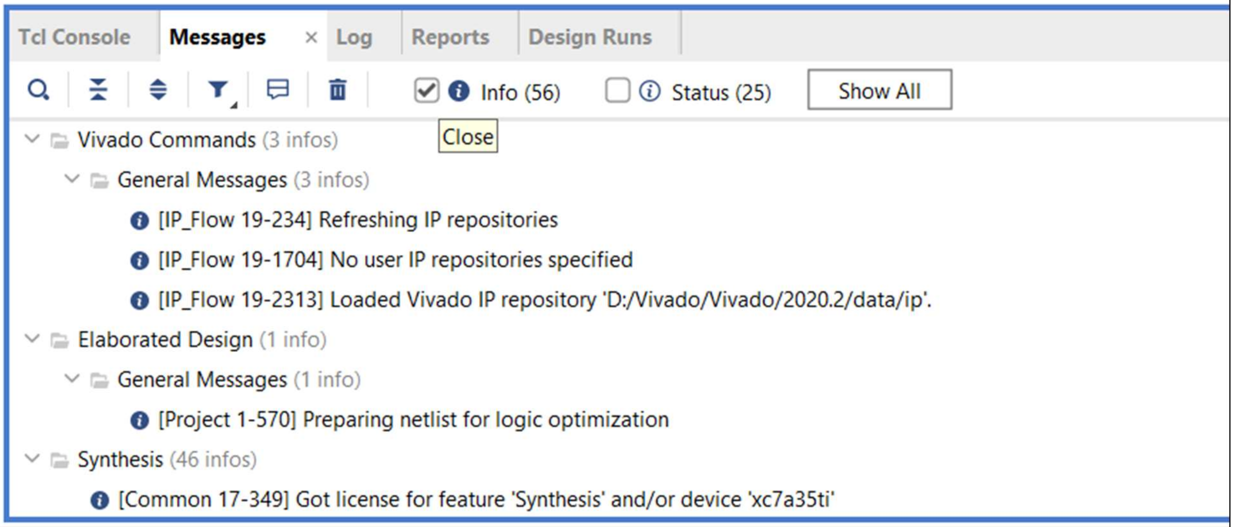


Report utilization after synthesis:



| Name | Slice LUTs (20800) | Slice Registers (41600) | DSPs (90) | Bonded IOB (106) | BUFGCTRL (32) |
|---|---|---|---|---|---|
| N DSP_Spartn_six | 206 | 158 | 1 | 326 | 1 |
| A1REG_DESIGN (MUX_DFF__parameterized0) | 0 | 18 | 0 | 0 | 0 |
| ADDER2 (FUll_adder_con) | 24 | 0 | 0 | 0 | 0 |
| B1REG_DESIGN (MUX_DFF__parameterized0_0) | 0 | 18 | 0 | 0 | 0 |
| CARRYOUTREG_DESIGN (MUX_DFF__parameterized4) | 0 | 1 | 0 | 0 | 0 |
| CREG_DESIGN (MUX_DFF__parameterized1) | 0 | 48 | 0 | 0 | 0 |
| DREG_DESIGN (MUX_DFF__parameterized0_1) | 0 | 18 | 0 | 0 | 0 |
| MREG_DESIGN (MUX_DFF__parameterized3) | 0 | 0 | 1 | 0 | 0 |
| mux2x1_design (mux2X1) | 19 | 0 | 0 | 0 | 0 |
| N_BIT_MUX4X1_design (N_BIT_MUX4X1) | 1 | 0 | 0 | 0 | 0 |
| N_BIT_MUX4X1_design2 (N_BIT_MUX4X1_2) | 49 | 0 | 0 | 0 | 0 |
| OPCODEREG_DESIGN (MUX_DFF__parameterized2) | 113 | 7 | 0 | 0 | 0 |
| PREG_DESIGN (MUX_DFF__parameterized1_3) | 0 | 48 | 0 | 0 | 0 |

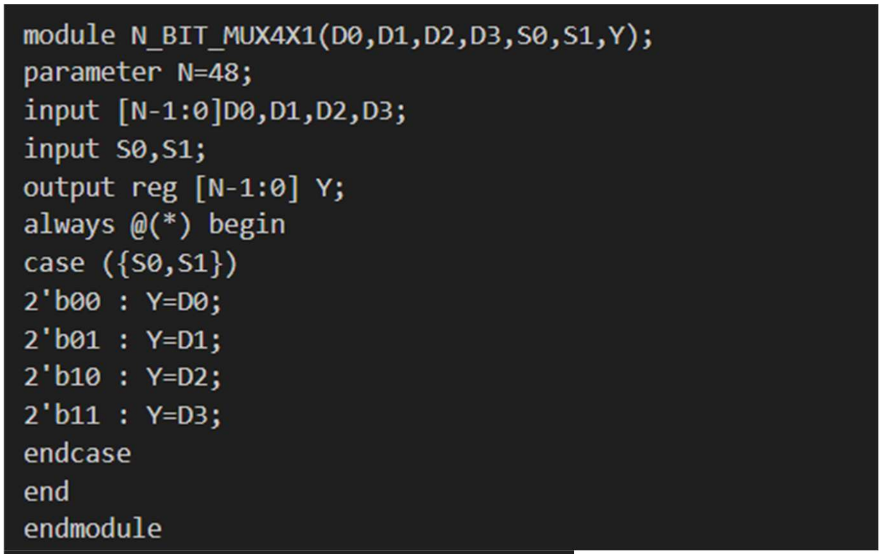
Report messages after synthesis:



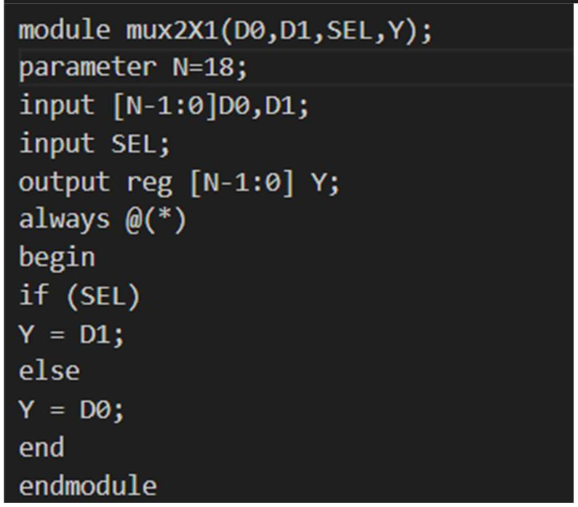
```
Vivado Commands (3 infos)
    General Messages (3 infos)
        [IP_Flow 19-234] Refreshing IP repositories
        [IP_Flow 19-1704] No user IP repositories specified
        [IP_Flow 19-2313] Loaded Vivado IP repository 'D:/Vivado/Vivado/2020.2/data/ip'.
Elaborated Design (1 info)
    General Messages (1 info)
        [Project 1-570] Preparing netlist for logic optimization
Synthesis (46 infos)
    [Common 17-349] Got license for feature 'Synthesis' and/or device 'xc7a35ti'
```

|   |
|---|
|   |
|   |

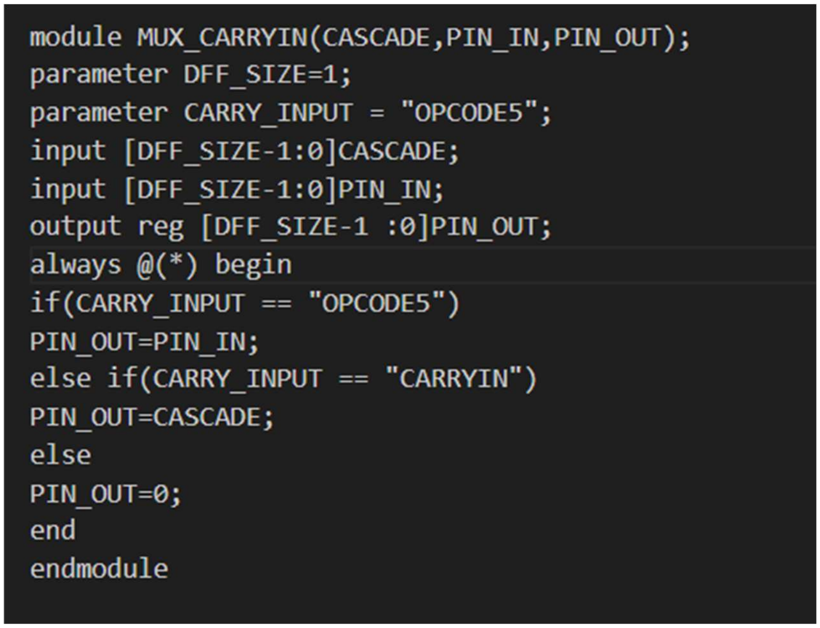
instance codes:



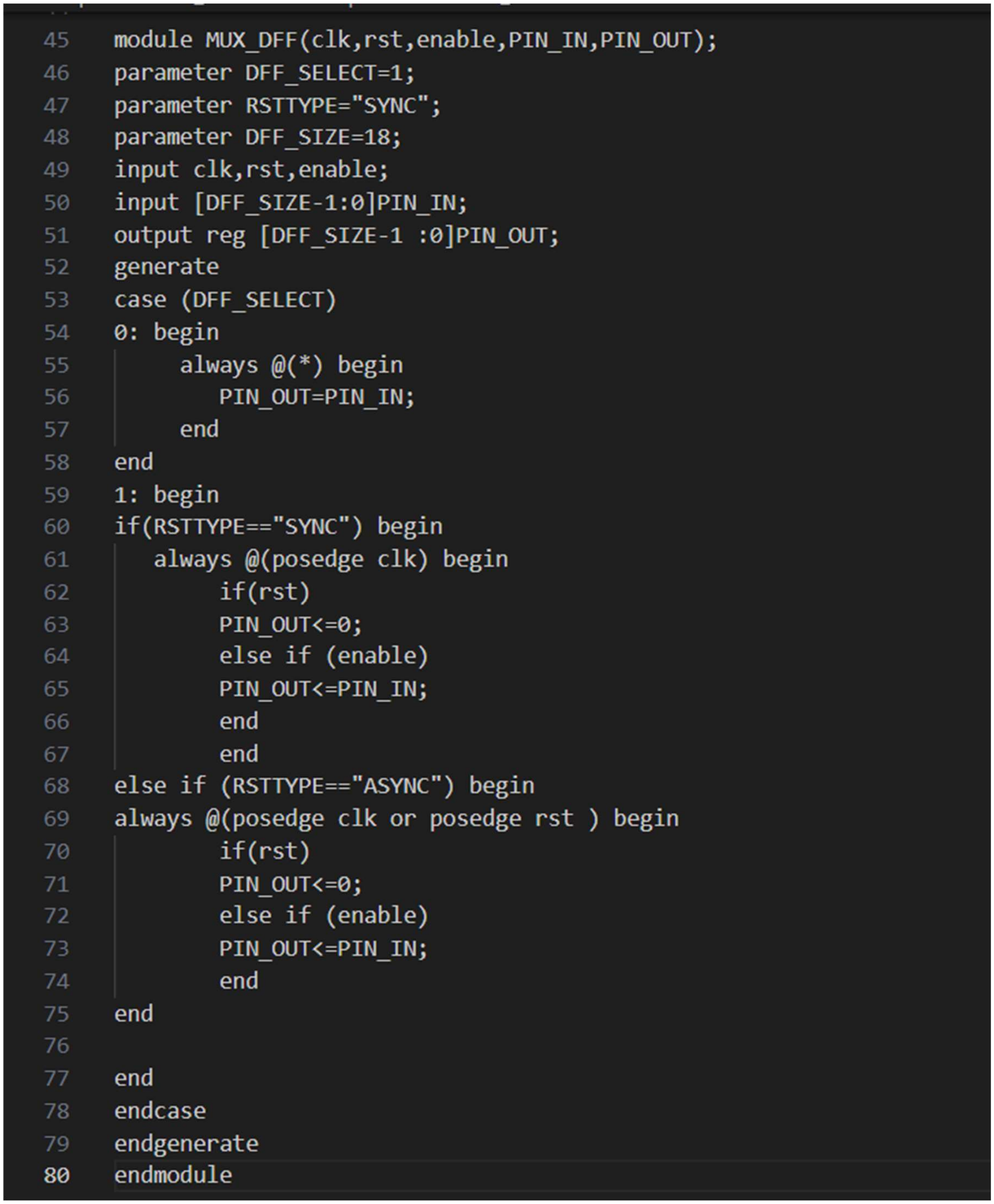
```
module N_BIT_MUX4X1(D0,D1,D2,D3,S0,S1,Y);
parameter N=48;
input [N-1:0]D0,D1,D2,D3;
input S0,S1;
output reg [N-1:0] Y;
always @(*) begin
case ({S0,S1})
2'b00 : Y=D0;
2'b01 : Y=D1;
2'b10 : Y=D2;
2'b11 : Y=D3;
endcase
end
endmodule
```



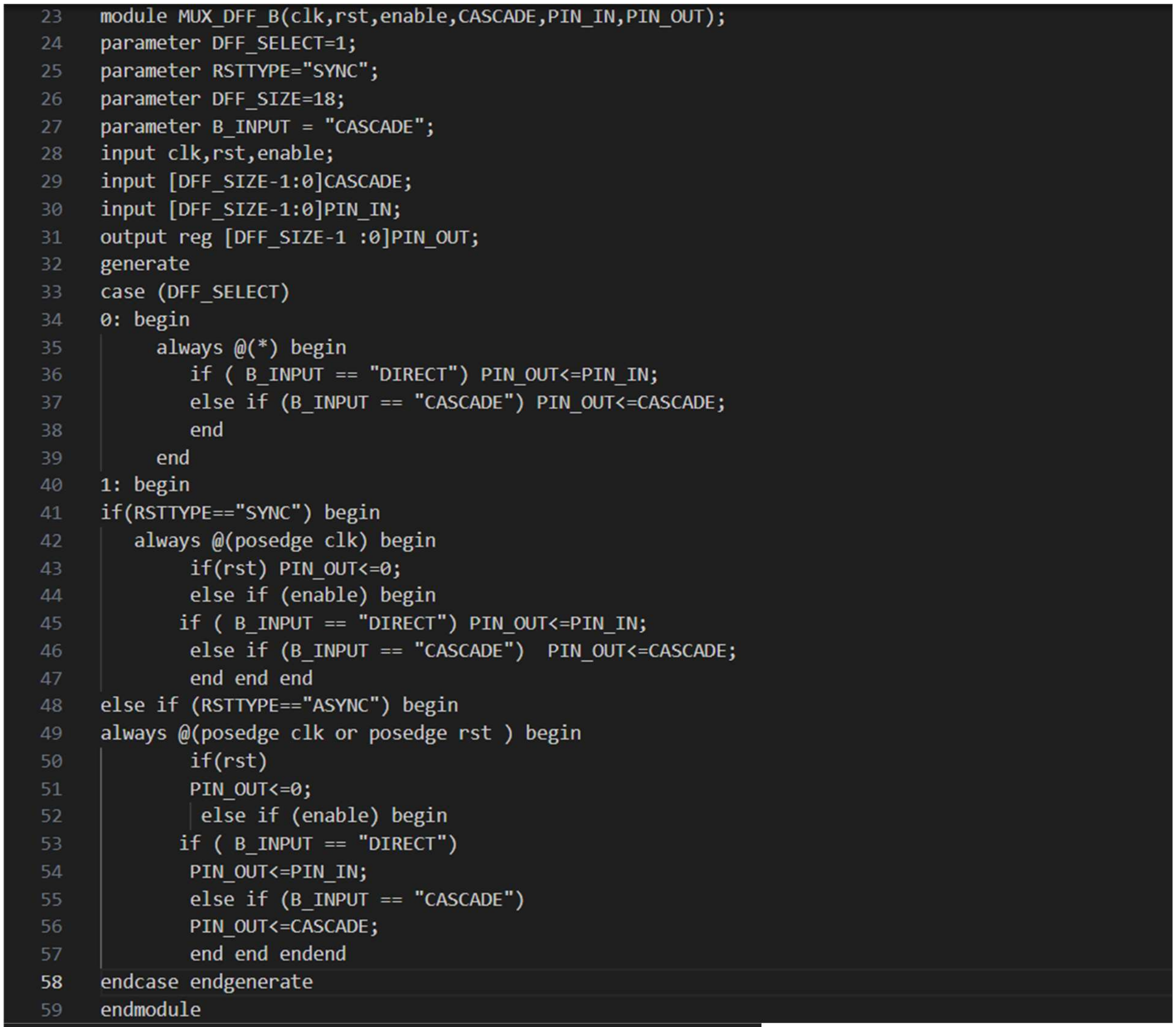
```
module mux2X1(D0,D1,SEL,Y);
parameter N=18;
input [N-1:0]D0,D1;
input SEL;
output reg [N-1:0] Y;
always @(*)
begin
if (SEL)
Y = D1;
else
Y = D0;
end
endmodule
```



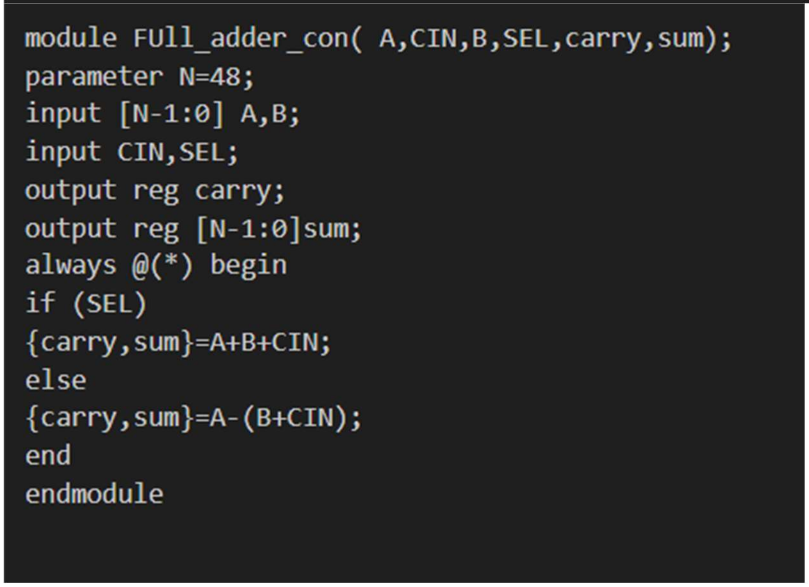
```
module MUX_CARRYIN(CASCADE,PIN_IN,PIN_OUT);
parameter DFF_SIZE=1;
parameter CARRY_INPUT = "OPCODE5";
input [DFF_SIZE-1:0]CASCADE;
input [DFF_SIZE-1:0]PIN_IN;
output reg [DFF_SIZE-1 :0]PIN_OUT;
always @(*) begin
if(CARRY_INPUT == "OPCODE5")
PIN_OUT=PIN_IN;
else if(CARRY_INPUT == "CARRYIN")
PIN_OUT=CASCADE;
else
PIN_OUT=0;
end
endmodule
```



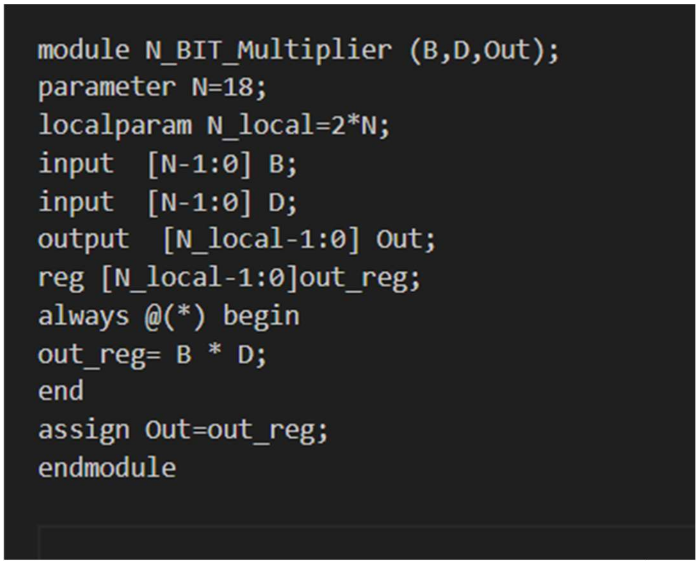
```
module MUX_DFF(clk,rst,enable,PIN_IN,PIN_OUT);
parameter DFF_SELECT=1;
parameter RSTTYPE="SYNC";
parameter DFF_SIZE=18;
input clk,rst,enable;
input [DFF_SIZE-1:0]PIN_IN;
output reg [DFF_SIZE-1 :0]PIN_OUT;
generate
case (DFF_SELECT)
0: begin
     always @(*) begin
        PIN_OUT=PIN_IN;
     end
end
1: begin
if(RSTTYPE=="SYNC") begin
   always @(posedge clk) begin
        if(rst)
        PIN_OUT<=0;
        else if (enable)
        PIN_OUT<=PIN_IN;
        end
        end
else if (RSTTYPE=="ASYNC") begin
always @(posedge clk or posedge rst ) begin
        if(rst)
        PIN_OUT<=0;
        else if (enable)
        PIN_OUT<=PIN_IN;
        end
end

end
endcase
endgenerate
endmodule
```



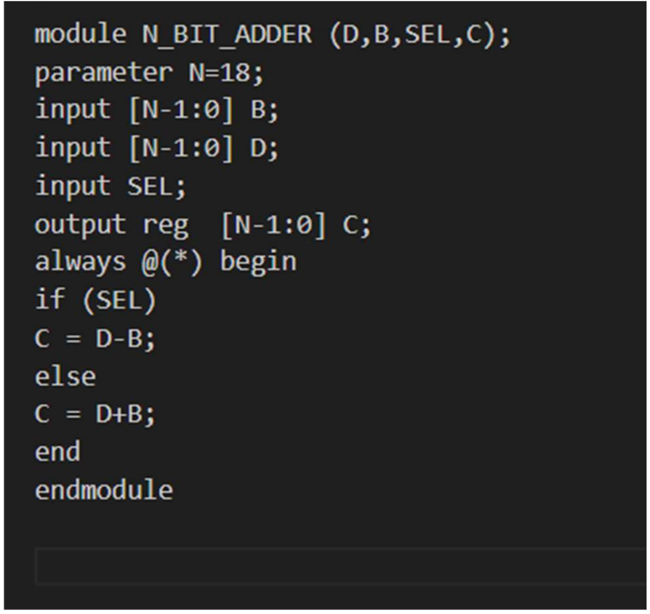
```
module MUX_DFF_B(clk,rst,enable,CASCADE,PIN_IN,PIN_OUT);
parameter DFF_SELECT=1;
parameter RSTTYPE="SYNC";
parameter DFF_SIZE=18;
parameter B_INPUT = "CASCADE";
input clk,rst,enable;
input [DFF_SIZE-1:0]CASCADE;
input [DFF_SIZE-1:0]PIN_IN;
output reg [DFF_SIZE-1 :0]PIN_OUT;
generate
case (DFF_SELECT)
0: begin
     always @(*) begin
        if ( B_INPUT == "DIRECT") PIN_OUT<=PIN_IN;
        else if (B_INPUT == "CASCADE") PIN_OUT<=CASCADE;
        end
     end
1: begin
if(RSTTYPE=="SYNC") begin
   always @(posedge clk) begin
        if(rst) PIN_OUT<=0;
        else if (enable) begin
       if ( B_INPUT == "DIRECT") PIN_OUT<=PIN_IN;
        else if (B_INPUT == "CASCADE")  PIN_OUT<=CASCADE;
        end end end
else if (RSTTYPE=="ASYNC") begin
always @(posedge clk or posedge rst ) begin
        if(rst)
        PIN_OUT<=0;
         else if (enable) begin
       if ( B_INPUT == "DIRECT")
        PIN_OUT<=PIN_IN;
        else if (B_INPUT == "CASCADE")
        PIN_OUT<=CASCADE;
        end end endend
endcase endgenerate
endmodule
```



```
module FUll_adder_con( A,CIN,B,SEL,carry,sum);
parameter N=48;
input [N-1:0] A,B;
input CIN,SEL;
output reg carry;
output reg [N-1:0]sum;
always @(*) begin
if (SEL)
{carry,sum}=A+B+CIN;
else
{carry,sum}=A-(B+CIN);
end
endmodule
```



```
module N_BIT_Multiplier (B,D,Out);
parameter N=18;
localparam N_local=2*N;
input  [N-1:0] B;
input  [N-1:0] D;
output  [N_local-1:0] Out;
reg [N_local-1:0]out_reg;
always @(*) begin
out_reg= B * D;
end
assign Out=out_reg;
endmodule
```



```
module N_BIT_ADDER (D,B,SEL,C);
parameter N=18;
input [N-1:0] B;
input [N-1:0] D;
input SEL;
output reg  [N-1:0] C;
always @(*) begin
if (SEL)
C = D-B;
else
C = D+B;
end
endmodule
```