Spartan6 DSP48A1

FPGA Flow

Created by : Yousef Gamal

Table of Contents

1. Spartan-6 DSP48A1 Overview

1.1. Description

The DSP48A1 slice in the Spartan-6 FPGA family is a versatile, high-performance DSP block optimized for digital signal processing applications. It is integrated into the FPGA fabric to enhance computational efficiency and performance.

1.2. Key Features

- **Arithmetic Operations:** Supports 18x18-bit multiply-accumulate (MAC) operations, enabling efficient processing of complex arithmetic functions.

- **Data Widths:** Can handle data widths of up to 48 bits, accommodating a wide range of signal processing tasks.

- **Operation Modes**: Capable of performing various operations such as multiplication, accumulation, and other arithmetic functions.

- **Resource Efficiency:** Designed to deliver high performance while minimizing the usage of FPGA resources, making it suitable for applications with stringent resource constraints.

1.3. Applications

The DSP48A1 slice is ideal for applications requiring intensive arithmetic processing, including:

- Digital filters
- Fast Fourier Transforms (FFT)
- Signal modulation and demodulation

1.4. Configuration and Usage

The DSP48A1 block can be configured using Xilinx's design tools (e.g., Vivado) to match the specific needs of the application. It offers flexibility in terms of input and output data width and supports various mathematical functions.

DSP48A1 Slice in Detail

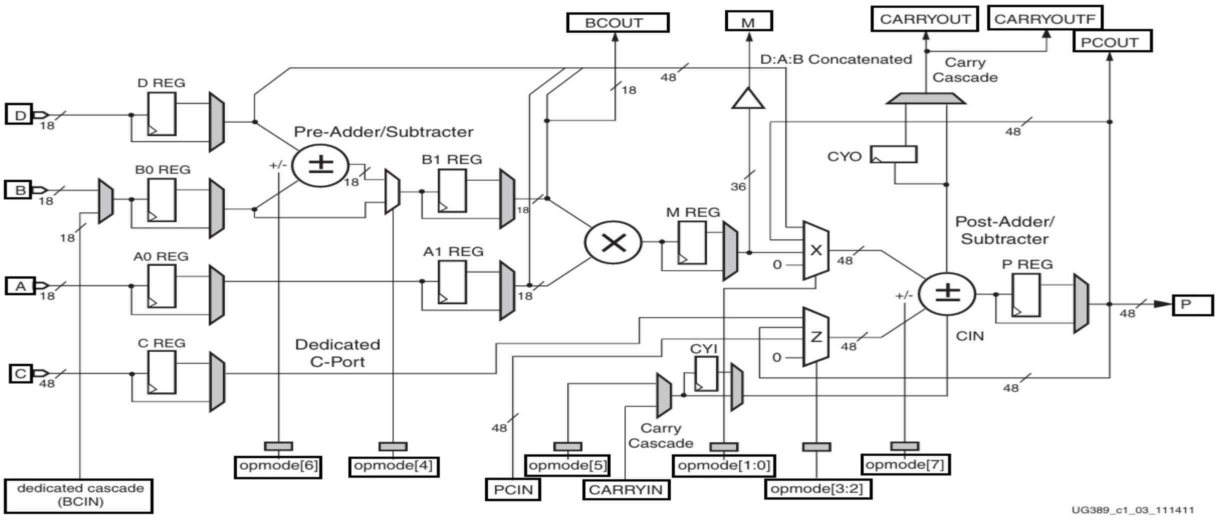


Figure 1-3: **DSP48A1 Slice**

2. RTL Code

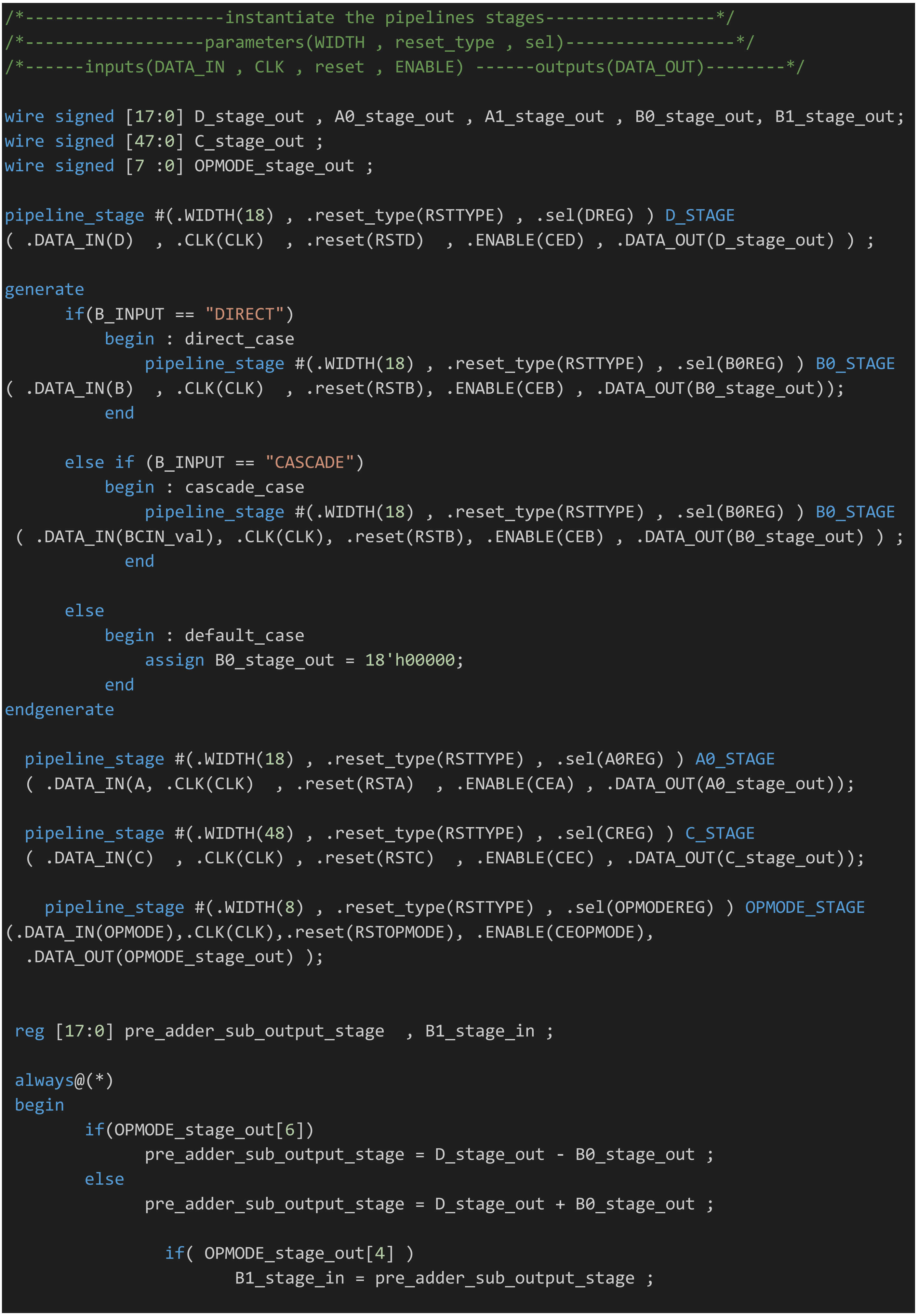
2.1. Pipeline Stage Module

```
module pipeline_stage  #(
    parameter WIDTH = 8,
    parameter reset_type = "ASYNC",
    parameter sel = 1 )
(
    input [WIDTH-1:0] DATA_IN,
    input CLK,
    input reset,
    input ENABLE,
    output reg [WIDTH-1:0] DATA_OUT
);

generate
    if (sel)
        begin
            // Registering Data
            if (reset_type == "SYNC")
                begin
                        always @(posedge CLK)
                        begin
                            if (reset)
                                DATA_OUT <= {WIDTH{1'b0}};
                            else if (ENABLE)
                                DATA_OUT <= DATA_IN;
                        end
                end

            else if (reset_type == "ASYNC")
                begin
                    always @(posedge CLK or posedge reset)
                    begin
                        if (reset)
                            DATA_OUT <= {WIDTH{1'b0}};
                        else if (ENABLE)
                                DATA_OUT <= DATA_IN;
                    end
                end
        end

    else
        begin
            // Bypassing Data
            always @(*)
                begin
                    DATA_OUT = DATA_IN;
                end
        end
endgenerate
endmodule
```

2.2. DSP Top Module

```
module DSP_top_module
#(
    /*-----specify the number of pipeline registers for input paths.-----*/

    parameter A0REG       = 0 ,  /* no register  */
    parameter A1REG       = 1 ,  /* one register */
    parameter B0REG       = 0 ,  /* no register  */
    parameter B1REG       = 1 ,  /* one register */
    parameter CREG        = 1 ,  /* one register */
    parameter DREG        = 1 ,  /* one register */
    parameter MREG        = 1 ,  /* one register */
    parameter PREG        = 1 ,  /* one register */
    parameter CARRYINREG  = 1 ,  /* one register */
    parameter CARRYOUTREG = 1 ,  /* one register */
    parameter OPMODEREG   = 1 ,  /* one register */

    /*---determine the carry cascade input source, Values CARRYIN or opcode[5].---*/
    /*---If neither "CARRYIN" nor "OPMODE5" is set, the output is tied to 0.---*/
    parameter CARRYINSEL = "OPMODE5" , /* default value */

    /*----specifies the source of the input to the B port------*/
    /*----DIRECT: The B port gets its input directly from the B input of the slice.---*/
    /*---CASCADE: The B port gets its input from the BCIN (B cascaded input) of the
          previous DSP48A1 slice.--*/
    /*------ else ==> the mux output should be Zero ------*/
     parameter B_INPUT = "DIRECT" ,

     /*--determines the reset for the DSP48A1 slice is synchronous or asynchronous--*/
     /*--ASYNC: Resets occur asynchronously----- */
     /*--SYNC : Resets occur synchronously------*/
    parameter RSTTYPE = "SYNC" ,

    /*parameters for cascading DSP48A1*/
    parameter BCIN_val  = 18'd2024
)

(
    /*--------------------input and output ports--------------------*/

    input CLK      , /*--DSP clock--*/
    input [7: 0]OPMODE ,
 /*--Control input to select the arithmetic operations of the DSP48A1 slice--*/

    input [17:0] A ,
/* input to multiplier && optionally to post_adder/subtracter depending on OPMODE[1:0].*/

    input [17:0] B ,  /* pre-adder/subtractor input/ multiplier based on OPMODE[4]/ post-
adder/subtractor based on OPMODE[1:0].*/
```

```
 input [17:0] D ,
 /* pre-adder/subtracter input |  D[11:0] are concatenated with A and B and optionally */
/*sent to post-adder/subtracter depending on the value of OPMODE[1:0]. */

    input [47:0] C , /* input to post-adder/subtracter*/
    input CARRYIN  , /* carry input to the post-adder/subtracter*/

    input CEA  ,   /*Clock enable for the A port registers (A0REG & A1REG)*/
    input CEB  ,   /*Clock enable for the B port registers (B0REG & B1REG)*/
    input CEC  ,   /*Clock enable for the C port registers (CREG) */
    input CED  ,   /*Clock enable for the D port registers (DREG) */
    input CEM  ,   /*Clock enable for the multiplier registers (MREG). */
    input CEP  ,   /*Clock enable for the P output port registers: (PREG = 1). */
    input CEOPMODE ,   /*Clock enable for the opmode register (OPMODEREG).*/
    input CECARRY_IN_OUT ,
 /*Clock enables the carry-in register and the carry-out register.*/


/*--All resets are active high. sync or async depending on the parameter RSTTYPE--*/

    input RSTA            ,   /*Reset for the A registers: (A0REG & A1REG).*/
    input RSTB            ,   /*Reset for the B registers: (B0REG & B1REG).*/
    input RSTC            ,   /*Reset for the C registers: (CREG).*/
    input RSTCARRY_IN_OUT ,/*Reset for the carry-in register and the carry-out register*/
    input RSTD             ,/*Reset for the D register (DREG)*/
    input RSTM             ,/*Reset for the multiplier register (MREG)*/
    input RSTOPMODE        ,/*Reset for the opmode register (OPMODEREG).*/
    input RSTP             ,/*Reset for the P output registers (PREG = 1).*/

    input [47:0] PCIN      ,/*Cascade input for Port P.*/
    input [17:0] BCIN      ,/*Cascade input for Port B.*/

    output  [17:0]BCOUT ,   /*Cascade output for Port B.*/
    output  [47:0]PCOUT ,   /*Cascade output for Port P.*/

    output  [35:0] M ,    /* buffered multiplier data output*/
                         /*  MREG = 1 ==> output of multiplier is registered */
                         /*  MREG = 0 ==> Direct output of multiplier */

    output  [47:0] P , /*output of post adder/subtractor */
                         /*  PREG = 1 ==> output is registered */
                         /*  PREG = 0 ==> Direct output  */

    output  CARRYOUT,/*carry out signal from post adder /  subtracter */
                        /*  CARRYOUTREG = 1 ==> output is registered */
                        /*  CARRYOUTREG = 0 ==> Direct output  */

    output  CARRYOUTF  /*It is a copy of the CARRYOUT signal */

);
```

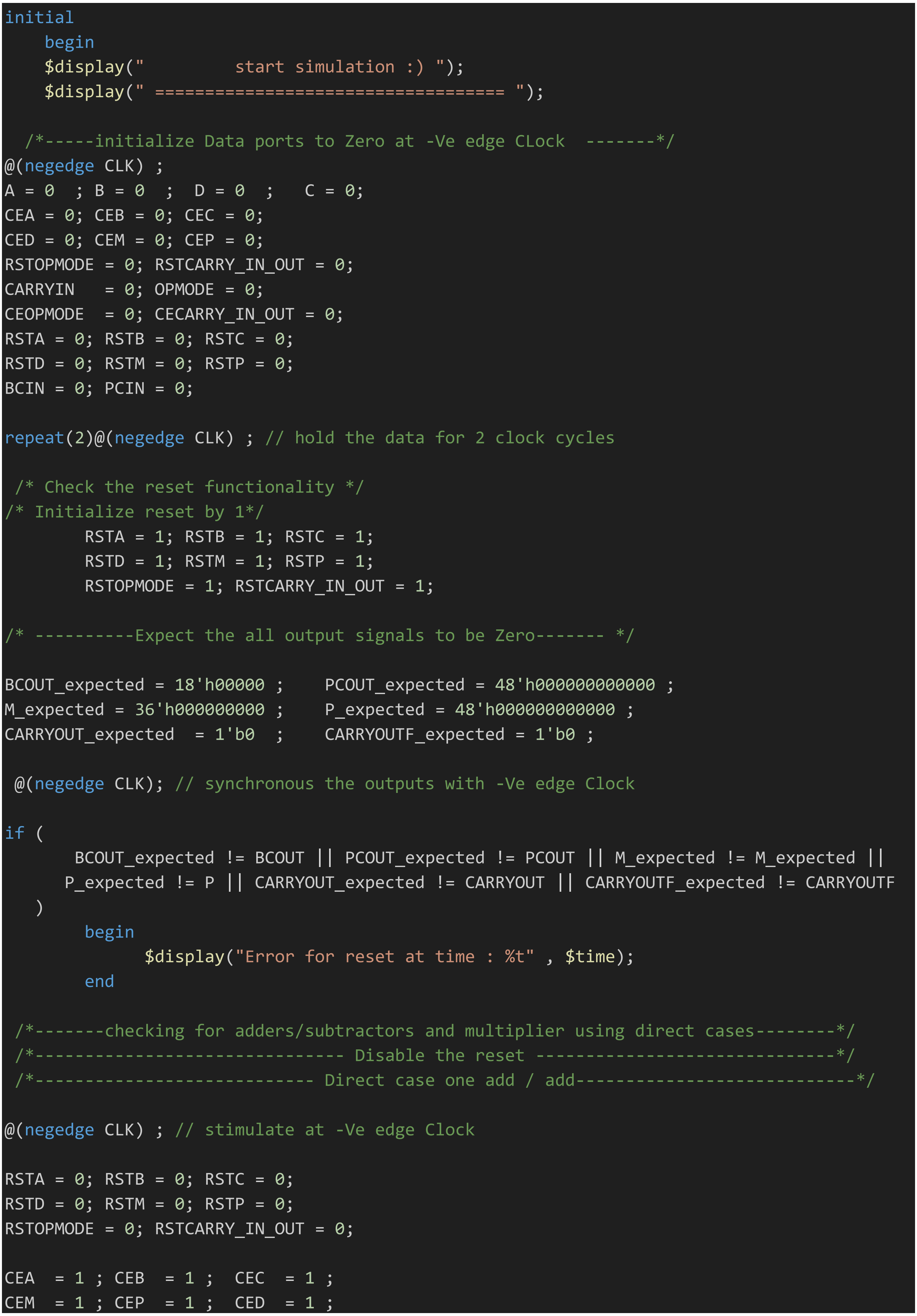


```
/*--------------------instantiate the pipelines stages-----------------*/
/*------------------parameters(WIDTH , reset_type , sel)-----------------*/
/*------inputs(DATA_IN , CLK , reset , ENABLE) ------outputs(DATA_OUT)--------*/

wire signed [17:0] D_stage_out , A0_stage_out , A1_stage_out , B0_stage_out, B1_stage_out;
wire signed [47:0] C_stage_out ;
wire signed [7 :0] OPMODE_stage_out ;

pipeline_stage #(.WIDTH(18) , .reset_type(RSTTYPE) , .sel(DREG) ) D_STAGE
( .DATA_IN(D)  , .CLK(CLK)  , .reset(RSTD)  , .ENABLE(CED) , .DATA_OUT(D_stage_out) ) ;

generate
      if(B_INPUT == "DIRECT")
          begin : direct_case
              pipeline_stage #(.WIDTH(18) , .reset_type(RSTTYPE) , .sel(B0REG) ) B0_STAGE
( .DATA_IN(B)  , .CLK(CLK)  , .reset(RSTB), .ENABLE(CEB) , .DATA_OUT(B0_stage_out));
          end

      else if (B_INPUT == "CASCADE")
          begin : cascade_case
              pipeline_stage #(.WIDTH(18) , .reset_type(RSTTYPE) , .sel(B0REG) ) B0_STAGE
 ( .DATA_IN(BCIN_val), .CLK(CLK), .reset(RSTB), .ENABLE(CEB) , .DATA_OUT(B0_stage_out) ) ;
            end

      else
          begin : default_case
              assign B0_stage_out = 18'h00000;
          end
endgenerate

  pipeline_stage #(.WIDTH(18) , .reset_type(RSTTYPE) , .sel(A0REG) ) A0_STAGE
  ( .DATA_IN(A, .CLK(CLK)  , .reset(RSTA)  , .ENABLE(CEA) , .DATA_OUT(A0_stage_out));

  pipeline_stage #(.WIDTH(48) , .reset_type(RSTTYPE) , .sel(CREG) ) C_STAGE
  ( .DATA_IN(C)  , .CLK(CLK) , .reset(RSTC)  , .ENABLE(CEC) , .DATA_OUT(C_stage_out));

    pipeline_stage #(.WIDTH(8) , .reset_type(RSTTYPE) , .sel(OPMODEREG) ) OPMODE_STAGE
(.DATA_IN(OPMODE),.CLK(CLK),.reset(RSTOPMODE), .ENABLE(CEOPMODE),
  .DATA_OUT(OPMODE_stage_out) );


 reg [17:0] pre_adder_sub_output_stage  , B1_stage_in ;

 always@(*)
 begin
        if(OPMODE_stage_out[6])
              pre_adder_sub_output_stage = D_stage_out - B0_stage_out ;
        else
              pre_adder_sub_output_stage = D_stage_out + B0_stage_out ;

                if( OPMODE_stage_out[4] )
                       B1_stage_in = pre_adder_sub_output_stage ;
                else
                       B1_stage_in = B0_stage_out ;

 end

    pipeline_stage #(.WIDTH(18) , .reset_type(RSTTYPE) , .sel(B1REG) ) B1_STAGE
  ( .DATA_IN(B1_stage_in)  , .CLK(CLK)  , .reset(RSTB)  , .ENABLE(CEB) ,
.DATA_OUT(B1_stage_out) ) ;

  assign BCOUT = B1_stage_out ;

    pipeline_stage #(.WIDTH(18) , .reset_type(RSTTYPE) , .sel(A1REG) ) A1_STAGE
  ( .DATA_IN(A0_stage_out)  , .CLK(CLK)  , .reset(RSTA)  , .ENABLE(CEA) ,
.DATA_OUT(A1_stage_out) ) ;


wire signed [35:0] mutliplier_stage_out  ;

assign mutliplier_stage_out = A1_stage_out * B1_stage_out ;

    pipeline_stage #(.WIDTH(36) , .reset_type(RSTTYPE) , .sel(MREG) ) M_STAGE
 (.DATA_IN(mutliplier_stage_out) , .CLK(CLK), .reset(RSTM), .ENABLE(CEM) ,.DATA_OUT(M));


wire current_carry_in ;

generate

    if(CARRYINSEL == "OPMODE5")
        assign current_carry_in = OPMODE_stage_out[5] ;

    else if(CARRYINSEL == "CARRYIN")
        assign current_carry_in = CARRYIN ;

endgenerate

wire CIN ;

    pipeline_stage #(.WIDTH(1) , .reset_type(RSTTYPE) , .sel(CARRYINREG) ) CYI_STAGE
  ( .DATA_IN(current_carry_in)  , .CLK(CLK)  , .reset(RSTCARRY_IN_OUT)  ,
.ENABLE(CECARRY_IN_OUT) , .DATA_OUT(CIN) ) ;

  reg [47:0] Z_mux_stage_out, X_mux_stage_out;
  reg [47:0] post_adder_sub_output_stage;
  reg [48:0] post_adder_sub_temp ;
  reg CYO_in;
```

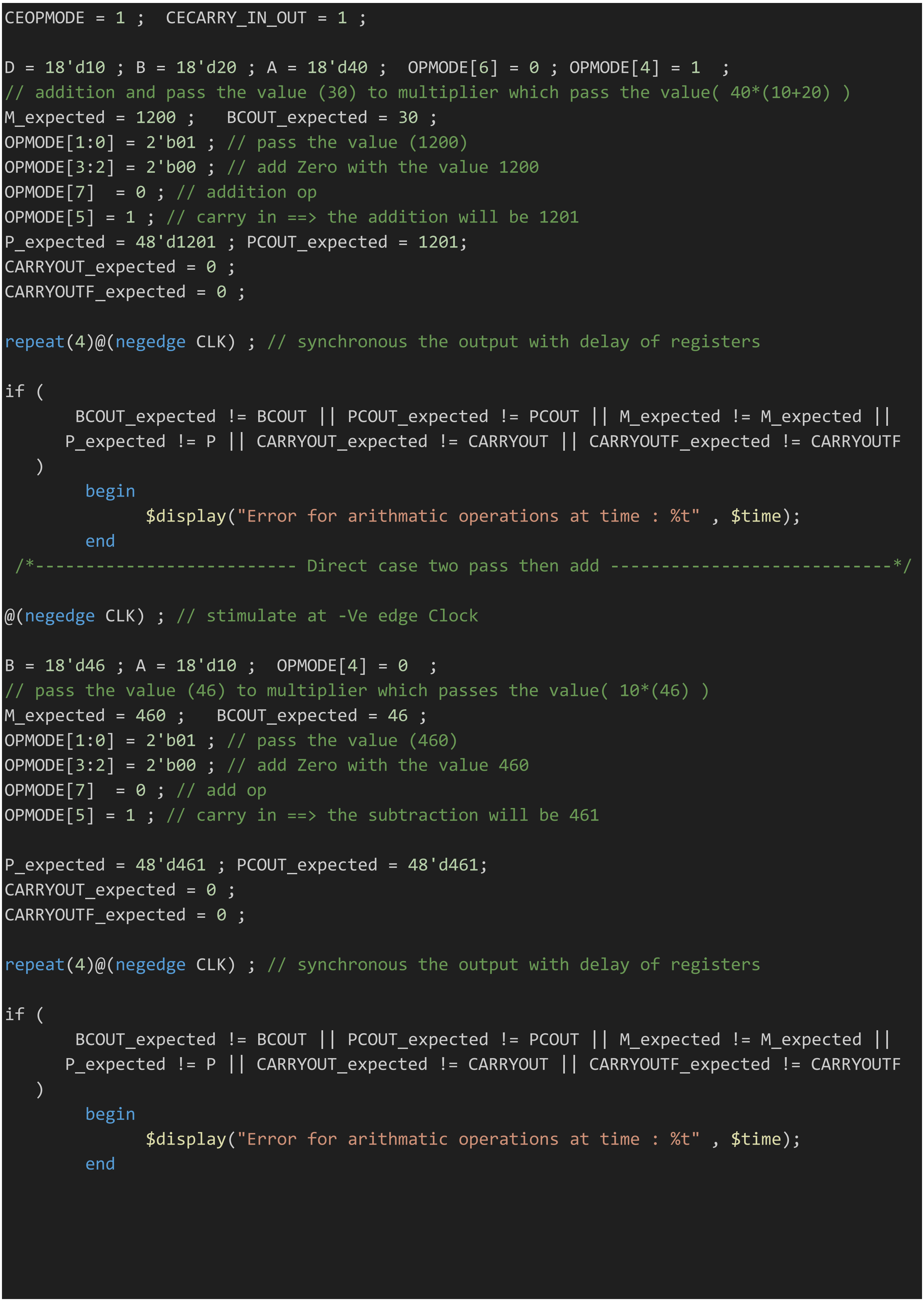
```
always@(*)
  begin
        case( {OPMODE_stage_out[3] , OPMODE_stage_out[2] } )

            2'b00: Z_mux_stage_out = 48'h000000000000 ;
            2'b01: Z_mux_stage_out = PCIN ;
            2'b10: Z_mux_stage_out = P ;
            2'b11: Z_mux_stage_out = C_stage_out ;

        endcase

        case( {OPMODE_stage_out[1] , OPMODE_stage_out[0] } )

            2'b00: X_mux_stage_out = 48'h000000000000 ;
            2'b01: X_mux_stage_out = { { 12{mutliplier_stage_out[35]} } ,
mutliplier_stage_out } ;
            2'b10: X_mux_stage_out =  P ;
            2'b11: X_mux_stage_out = { D_stage_out[11:0] ,  A1_stage_out[17:0]
,  B1_stage_out[17:0] } ;

        endcase

       if( OPMODE_stage_out[7] )

            post_adder_sub_temp  = Z_mux_stage_out - (X_mux_stage_out + CIN)  ;
     else
            post_adder_sub_temp = Z_mux_stage_out + X_mux_stage_out + CIN  ;

            post_adder_sub_output_stage = post_adder_sub_temp[47 : 0] ;
            CYO_in  = post_adder_sub_temp[48] ;

  end

    pipeline_stage #(.WIDTH(1) , .reset_type(RSTTYPE) , .sel(CARRYOUTREG) ) CYO_STAGE
  ( .DATA_IN(CYO_in)  , .CLK(CLK)  , .reset(RSTCARRY_IN_OUT)  , .ENABLE(CECARRY_IN_OUT) ,
.DATA_OUT(CARRYOUT) ) ;

    assign CARRYOUTF = CARRYOUT ;

    pipeline_stage #(.WIDTH(48) , .reset_type(RSTTYPE) , .sel(PREG) ) P_STAGE
  ( .DATA_IN(post_adder_sub_output_stage)  , .CLK(CLK)  , .reset(RSTP)  , .ENABLE(CEP) ,
.DATA_OUT(P) ) ;

  assign PCOUT = P ;


endmodule
```

3. Testbench Code

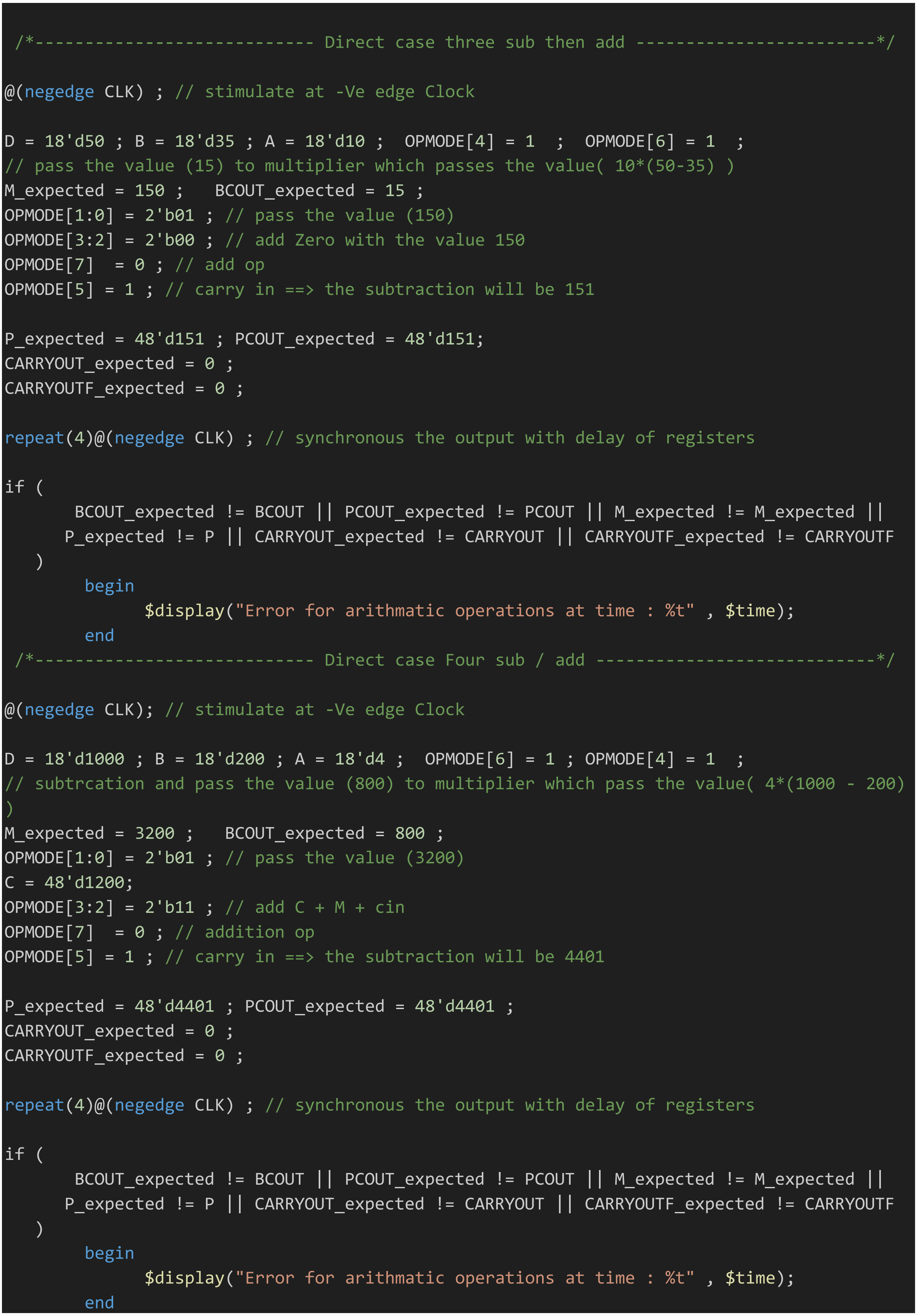
```
/* This testbench is done for some default value */

/* if we need to change it pass the required paramter to DUT and change the delay */
/* Defaults ==> A0  , B0 are not exist */
/* Defaults ==> carry in is opmode [5] */
/* Defaults ==> disable cascading for port B*/
/* Defaults ==> synchronous reset*/

module DSP_tb ;

    reg CLK ;
    reg [7: 0]OPMODE ;
    reg [17:0] A  , B , D ;
    reg [17:0] BCIN  ;
    reg [47:0] PCIN  ;
    reg [47:0] C ;
    reg CARRYIN  ;
    reg CEA ,CEB , CEC , CED , CEM ,  CEP ;
    reg CEOPMODE ;
    reg CECARRY_IN_OUT ;
    reg RSTA , RSTB , RSTC , RSTD , RSTM , RSTP ;
    reg RSTCARRY_IN_OUT  ;
    reg RSTOPMODE ;

    wire  [17:0]BCOUT ;
    wire  [47:0]PCOUT ;
    wire  [35:0] M ;
    wire  [47:0] P ;
    wire  CARRYOUT ;
    wire  CARRYOUTF ;

    reg  [17:0]BCOUT_expected ;
    reg  [47:0]PCOUT_expected ;
    reg  [35:0] M_expected ;
    reg  [47:0] P_expected ;
    reg  CARRYOUT_expected ;
    reg  CARRYOUTF_expected ;

DSP_top_module DUT (.*) ;

always
    begin
        CLK = 0 ;
        #10 ;
        CLK = 1 ;
        #10 ;
    end
```



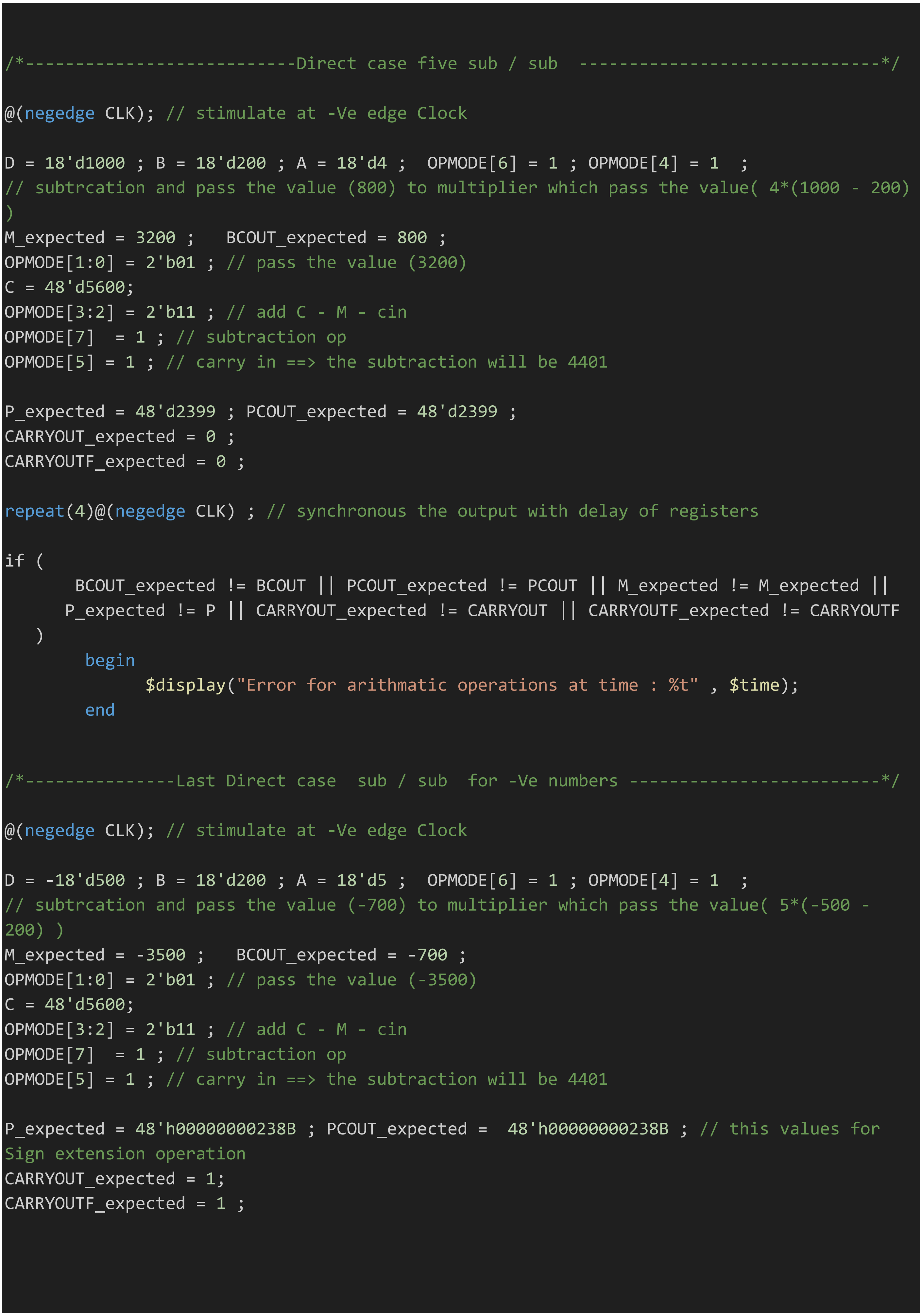
```
initial
    begin
    $display("         start simulation :) ");
    $display(" =================================== ");

  /*-----initialize Data ports to Zero at -Ve edge CLock  -------*/
@(negedge CLK) ;
A = 0  ; B = 0  ;  D = 0  ;   C = 0;
CEA = 0; CEB = 0; CEC = 0;
CED = 0; CEM = 0; CEP = 0;
RSTOPMODE = 0; RSTCARRY_IN_OUT = 0;
CARRYIN   = 0; OPMODE = 0;
CEOPMODE  = 0; CECARRY_IN_OUT = 0;
RSTA = 0; RSTB = 0; RSTC = 0;
RSTD = 0; RSTM = 0; RSTP = 0;
BCIN = 0; PCIN = 0;

repeat(2)@(negedge CLK) ; // hold the data for 2 clock cycles

 /* Check the reset functionality */
/* Initialize reset by 1*/
        RSTA = 1; RSTB = 1; RSTC = 1;
        RSTD = 1; RSTM = 1; RSTP = 1;
        RSTOPMODE = 1; RSTCARRY_IN_OUT = 1;

/* ----------Expect the all output signals to be Zero------- */

BCOUT_expected = 18'h00000 ;    PCOUT_expected = 48'h000000000000 ;
M_expected = 36'h000000000 ;    P_expected = 48'h000000000000 ;
CARRYOUT_expected  = 1'b0  ;    CARRYOUTF_expected = 1'b0 ;

 @(negedge CLK); // synchronous the outputs with -Ve edge Clock

if (
       BCOUT_expected != BCOUT || PCOUT_expected != PCOUT || M_expected != M_expected ||
      P_expected != P || CARRYOUT_expected != CARRYOUT || CARRYOUTF_expected != CARRYOUTF
   )
        begin
              $display("Error for reset at time : %t" , $time);
        end

 /*-------checking for adders/subtractors and multiplier using direct cases--------*/
 /*------------------------------- Disable the reset ------------------------------*/
 /*---------------------------- Direct case one add / add----------------------------*/

@(negedge CLK) ; // stimulate at -Ve edge Clock

RSTA = 0; RSTB = 0; RSTC = 0;
RSTD = 0; RSTM = 0; RSTP = 0;
RSTOPMODE = 0; RSTCARRY_IN_OUT = 0;

CEA  = 1 ; CEB  = 1 ;  CEC  = 1 ;
CEM  = 1 ; CEP  = 1 ;  CED  = 1 ;
```



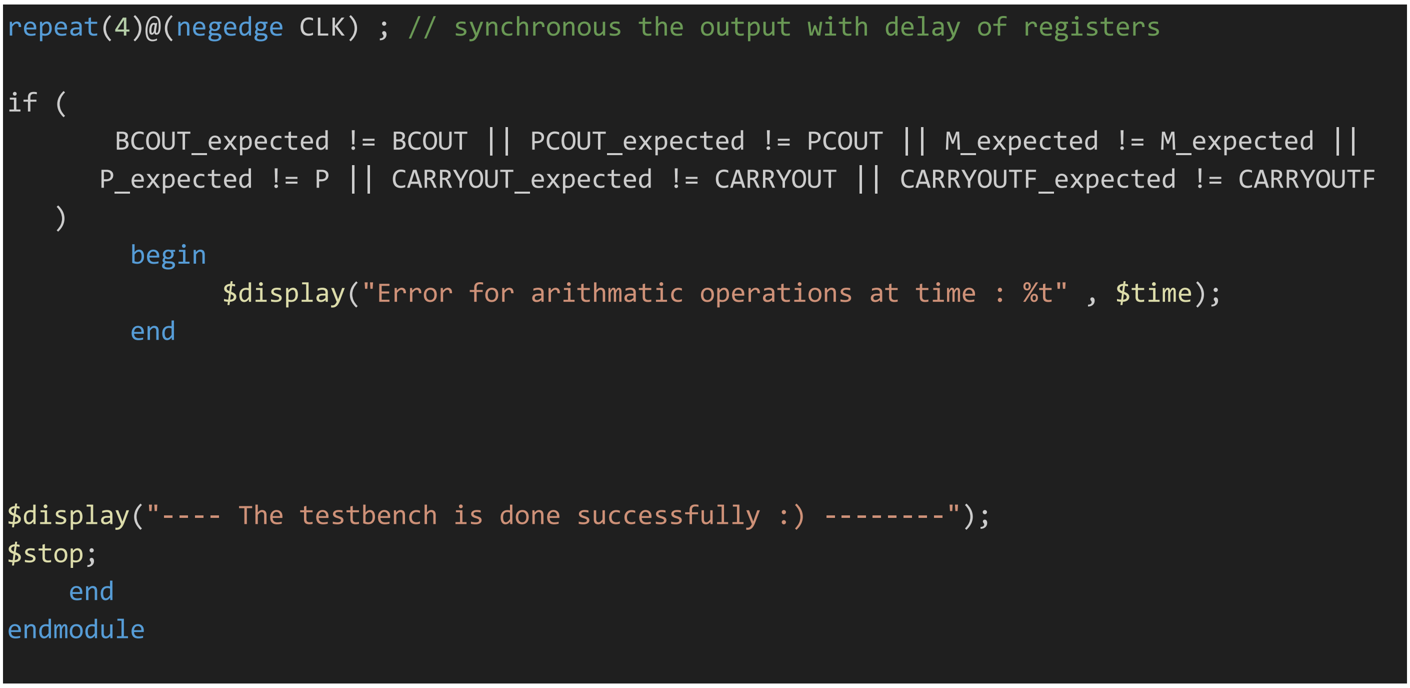
```
CEOPMODE = 1 ;  CECARRY_IN_OUT = 1 ;

D = 18'd10 ; B = 18'd20 ; A = 18'd40 ;  OPMODE[6] = 0 ; OPMODE[4] = 1  ;
// addition and pass the value (30) to multiplier which pass the value( 40*(10+20) )
M_expected = 1200 ;   BCOUT_expected = 30 ;
OPMODE[1:0] = 2'b01 ; // pass the value (1200)
OPMODE[3:2] = 2'b00 ; // add Zero with the value 1200
OPMODE[7]  = 0 ; // addition op
OPMODE[5] = 1 ; // carry in ==> the addition will be 1201
P_expected = 48'd1201 ; PCOUT_expected = 1201;
CARRYOUT_expected = 0 ;
CARRYOUTF_expected = 0 ;

repeat(4)@(negedge CLK) ; // synchronous the output with delay of registers

if (
       BCOUT_expected != BCOUT || PCOUT_expected != PCOUT || M_expected != M_expected ||
      P_expected != P || CARRYOUT_expected != CARRYOUT || CARRYOUTF_expected != CARRYOUTF
   )
        begin
              $display("Error for arithmatic operations at time : %t" , $time);
        end
 /*-------------------------- Direct case two pass then add ----------------------------*/

@(negedge CLK) ; // stimulate at -Ve edge Clock

B = 18'd46 ; A = 18'd10 ;  OPMODE[4] = 0  ;
// pass the value (46) to multiplier which passes the value( 10*(46) )
M_expected = 460 ;   BCOUT_expected = 46 ;
OPMODE[1:0] = 2'b01 ; // pass the value (460)
OPMODE[3:2] = 2'b00 ; // add Zero with the value 460
OPMODE[7]  = 0 ; // add op
OPMODE[5] = 1 ; // carry in ==> the subtraction will be 461

P_expected = 48'd461 ; PCOUT_expected = 48'd461;
CARRYOUT_expected = 0 ;
CARRYOUTF_expected = 0 ;

repeat(4)@(negedge CLK) ; // synchronous the output with delay of registers

if (
       BCOUT_expected != BCOUT || PCOUT_expected != PCOUT || M_expected != M_expected ||
      P_expected != P || CARRYOUT_expected != CARRYOUT || CARRYOUTF_expected != CARRYOUTF
   )
        begin
              $display("Error for arithmatic operations at time : %t" , $time);
        end
```



```
 /*----------------------------- Direct case three sub then add ------------------------*/

@(negedge CLK) ; // stimulate at -Ve edge Clock

D = 18'd50 ; B = 18'd35 ; A = 18'd10 ;  OPMODE[4] = 1  ;  OPMODE[6] = 1  ;
// pass the value (15) to multiplier which passes the value( 10*(50-35) )
M_expected = 150 ;   BCOUT_expected = 15 ;
OPMODE[1:0] = 2'b01 ; // pass the value (150)
OPMODE[3:2] = 2'b00 ; // add Zero with the value 150
OPMODE[7]  = 0 ; // add op
OPMODE[5] = 1 ; // carry in ==> the subtraction will be 151

P_expected = 48'd151 ; PCOUT_expected = 48'd151;
CARRYOUT_expected = 0 ;
CARRYOUTF_expected = 0 ;

repeat(4)@(negedge CLK) ; // synchronous the output with delay of registers

if (
       BCOUT_expected != BCOUT || PCOUT_expected != PCOUT || M_expected != M_expected ||
      P_expected != P || CARRYOUT_expected != CARRYOUT || CARRYOUTF_expected != CARRYOUTF
   )
        begin
              $display("Error for arithmatic operations at time : %t" , $time);
        end
 /*---------------------------- Direct case Four sub / add ----------------------------*/

@(negedge CLK); // stimulate at -Ve edge Clock

D = 18'd1000 ; B = 18'd200 ; A = 18'd4 ;  OPMODE[6] = 1 ; OPMODE[4] = 1  ;
// subtrcation and pass the value (800) to multiplier which pass the value( 4*(1000 - 200)
)
M_expected = 3200 ;   BCOUT_expected = 800 ;
OPMODE[1:0] = 2'b01 ; // pass the value (3200)
C = 48'd1200;
OPMODE[3:2] = 2'b11 ; // add C + M + cin
OPMODE[7]  = 0 ; // addition op
OPMODE[5] = 1 ; // carry in ==> the subtraction will be 4401

P_expected = 48'd4401 ; PCOUT_expected = 48'd4401 ;
CARRYOUT_expected = 0 ;
CARRYOUTF_expected = 0 ;

repeat(4)@(negedge CLK) ; // synchronous the output with delay of registers

if (
       BCOUT_expected != BCOUT || PCOUT_expected != PCOUT || M_expected != M_expected ||
      P_expected != P || CARRYOUT_expected != CARRYOUT || CARRYOUTF_expected != CARRYOUTF
   )
        begin
              $display("Error for arithmatic operations at time : %t" , $time);
        end
```

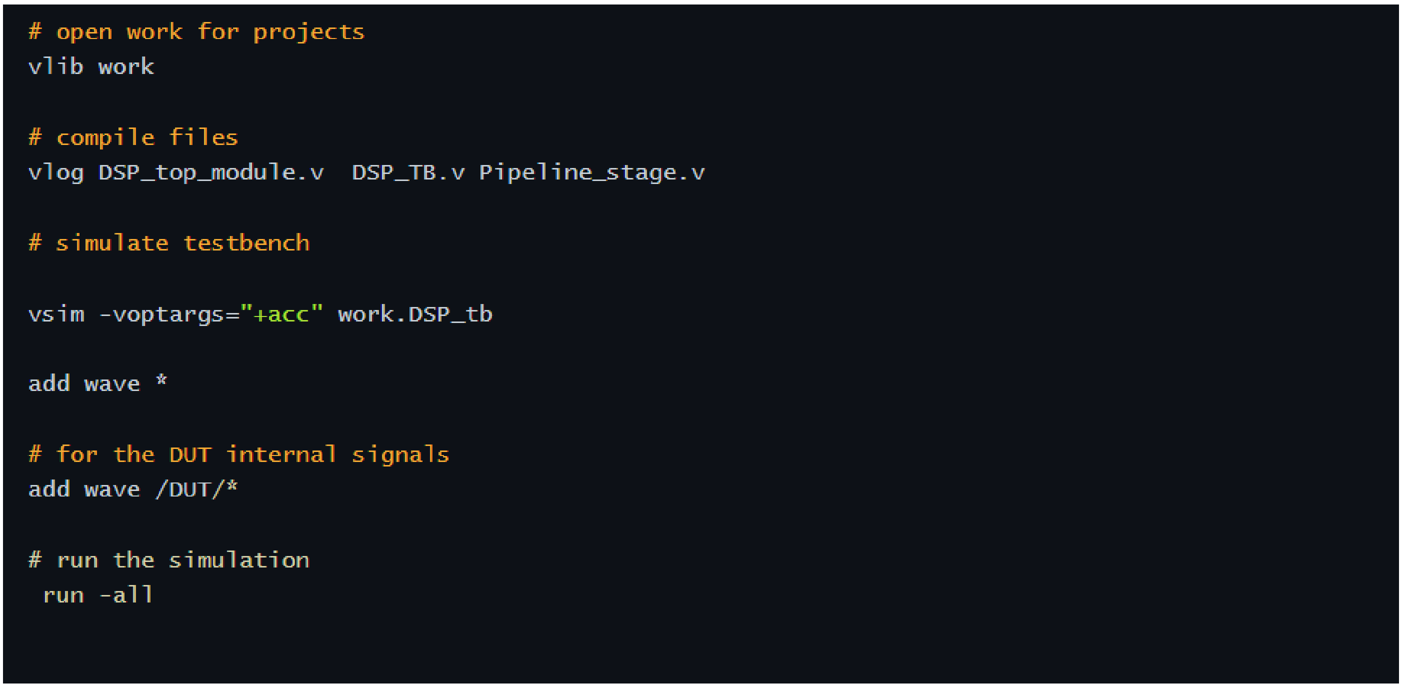


```
/*---------------------------Direct case five sub / sub  ------------------------------*/

@(negedge CLK); // stimulate at -Ve edge Clock

D = 18'd1000 ; B = 18'd200 ; A = 18'd4 ;  OPMODE[6] = 1 ; OPMODE[4] = 1  ;
// subtrcation and pass the value (800) to multiplier which pass the value( 4*(1000 - 200)
)
M_expected = 3200 ;   BCOUT_expected = 800 ;
OPMODE[1:0] = 2'b01 ; // pass the value (3200)
C = 48'd5600;
OPMODE[3:2] = 2'b11 ; // add C - M - cin
OPMODE[7]  = 1 ; // subtraction op
OPMODE[5] = 1 ; // carry in ==> the subtraction will be 4401

P_expected = 48'd2399 ; PCOUT_expected = 48'd2399 ;
CARRYOUT_expected = 0 ;
CARRYOUTF_expected = 0 ;

repeat(4)@(negedge CLK) ; // synchronous the output with delay of registers

if (
       BCOUT_expected != BCOUT || PCOUT_expected != PCOUT || M_expected != M_expected ||
      P_expected != P || CARRYOUT_expected != CARRYOUT || CARRYOUTF_expected != CARRYOUTF
   )
        begin
              $display("Error for arithmatic operations at time : %t" , $time);
        end


/*--------------Last Direct case  sub / sub  for -Ve numbers ------------------------*/

@(negedge CLK); // stimulate at -Ve edge Clock

D = -18'd500 ; B = 18'd200 ; A = 18'd5 ;  OPMODE[6] = 1 ; OPMODE[4] = 1  ;
// subtrcation and pass the value (-700) to multiplier which pass the value( 5*(-500 -
200) )
M_expected = -3500 ;   BCOUT_expected = -700 ;
OPMODE[1:0] = 2'b01 ; // pass the value (-3500)
C = 48'd5600;
OPMODE[3:2] = 2'b11 ; // add C - M - cin
OPMODE[7]  = 1 ; // subtraction op
OPMODE[5] = 1 ; // carry in ==> the subtraction will be 4401

P_expected = 48'h00000000238B ; PCOUT_expected =  48'h00000000238B ; // this values for
Sign extension operation
CARRYOUT_expected = 1;
CARRYOUTF_expected = 1 ;
```



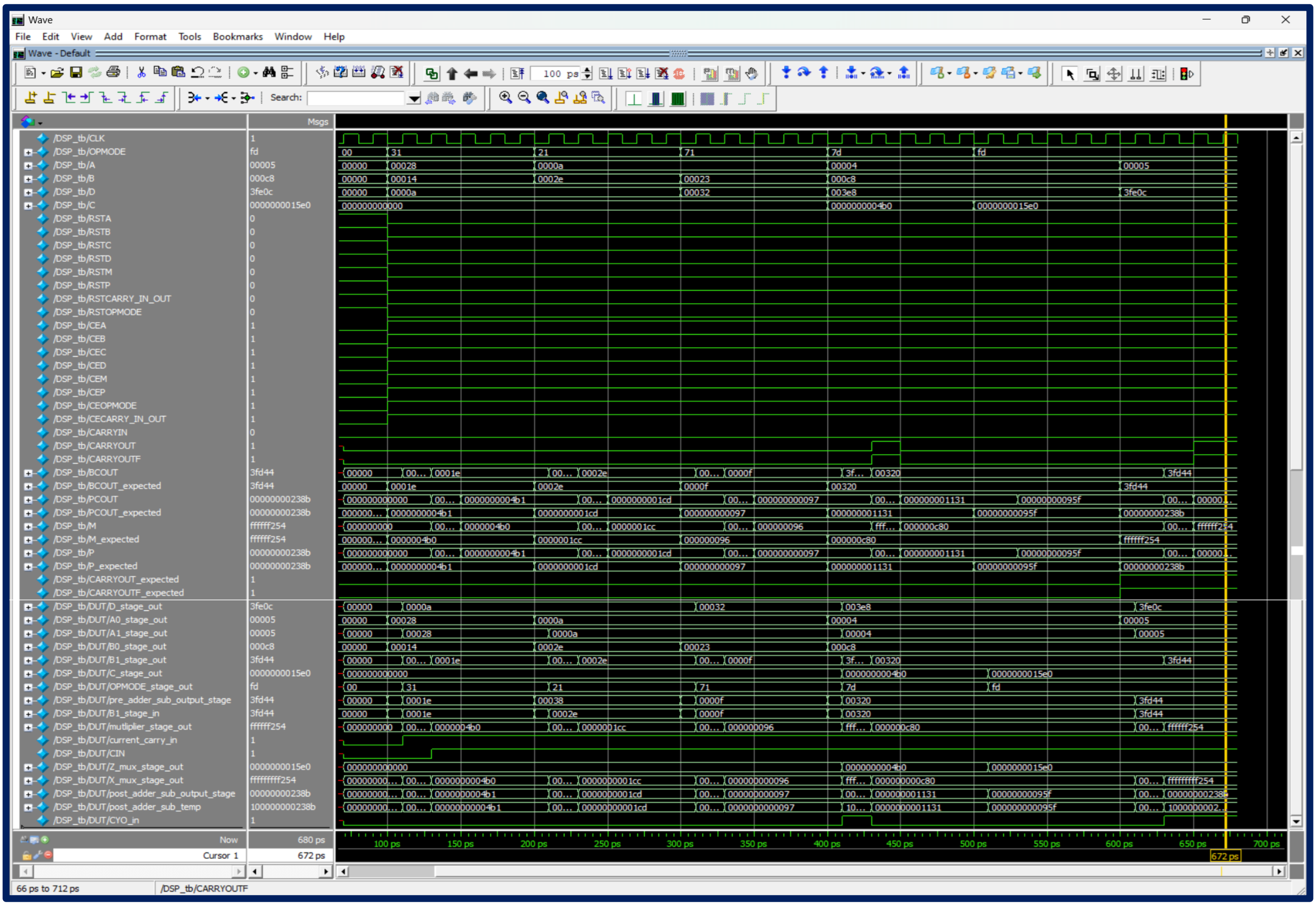
```
repeat(4)@(negedge CLK) ; // synchronous the output with delay of registers

if (
       BCOUT_expected != BCOUT || PCOUT_expected != PCOUT || M_expected != M_expected ||
      P_expected != P || CARRYOUT_expected != CARRYOUT || CARRYOUTF_expected != CARRYOUTF
   )
        begin
              $display("Error for arithmatic operations at time : %t" , $time);
        end



$display("---- The testbench is done successfully :) --------");
$stop;
    end
endmodule
```

4. Do File



```
# open work for projects
vlib work

# compile files
vlog DSP_top_module.v  DSP_TB.v Pipeline_stage.v

# simulate testbench

vsim -voptargs="+acc" work.DSP_tb

add wave *

# for the DUT internal signals
add wave /DUT/*

# run the simulation
 run -all
```

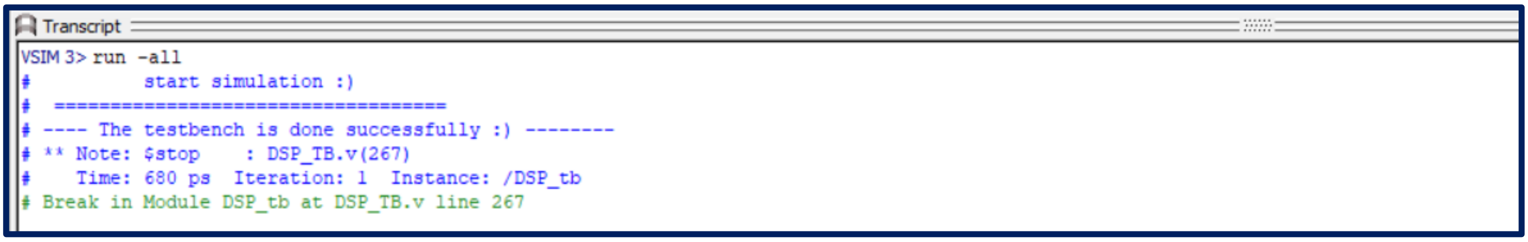
5. Waveform



	Msgs
/DSP_tb/CLK	1
/DSP_tb/OPMODE	fd
/DSP_tb/A	00005
/DSP_tb/B	000c8
/DSP_tb/D	3fe0c
/DSP_tb/C	0000000015e0
/DSP_tb/RSTA	0
/DSP_tb/RSTB	0
/DSP_tb/RSTC	0
/DSP_tb/RSTD	0
/DSP_tb/RSTM	0
/DSP_tb/RSTP	0
/DSP_tb/RSTCARRY_IN_OUT	0
/DSP_tb/RSTOPMODE	0
/DSP_tb/CEA	1
/DSP_tb/CEB	1
/DSP_tb/CEC	1
/DSP_tb/CED	1
/DSP_tb/CEM	1
/DSP_tb/CEP	1
/DSP_tb/CEOPMODE	1
/DSP_tb/CECARRY_IN_OUT	1
/DSP_tb/CARRYIN	0
/DSP_tb/CARRYOUT	1
/DSP_tb/CARRYOUTF	1
/DSP_tb/BCOUT	3fd44
/DSP_tb/BCOUT_expected	3fd44
/DSP_tb/PCOUT	00000000238b
/DSP_tb/PCOUT_expected	00000000238b
/DSP_tb/M	ffffff254
/DSP_tb/M_expected	ffffff254
/DSP_tb/P	00000000238b
/DSP_tb/P_expected	00000000238b
/DSP_tb/CARRYOUT_expected	1
/DSP_tb/CARRYOUTF_expected	1
/DSP_tb/DUT/D_stage_out	3fe0c
/DSP_tb/DUT/A0_stage_out	00005
/DSP_tb/DUT/A1_stage_out	00005
/DSP_tb/DUT/B0_stage_out	000c8
/DSP_tb/DUT/B1_stage_out	3fd44
/DSP_tb/DUT/C_stage_out	0000000015e0
/DSP_tb/DUT/OPMODE_stage_out	fd
/DSP_tb/DUT/pre_adder_sub_output_stage	3fd44
/DSP_tb/DUT/B1_stage_in	3fd44
/DSP_tb/DUT/mutliplier_stage_out	ffffff254
/DSP_tb/DUT/current_carry_in	1
/DSP_tb/DUT/CIN	1
/DSP_tb/DUT/Z_mux_stage_out	0000000015e0
/DSP_tb/DUT/X_mux_stage_out	fffffffff254
/DSP_tb/DUT/post_adder_sub_output_stage	00000000238b
/DSP_tb/DUT/post_adder_sub_temp	100000000238b
/DSP_tb/DUT/CYO_in	1
Now	680 ps
Cursor 1	672 ps

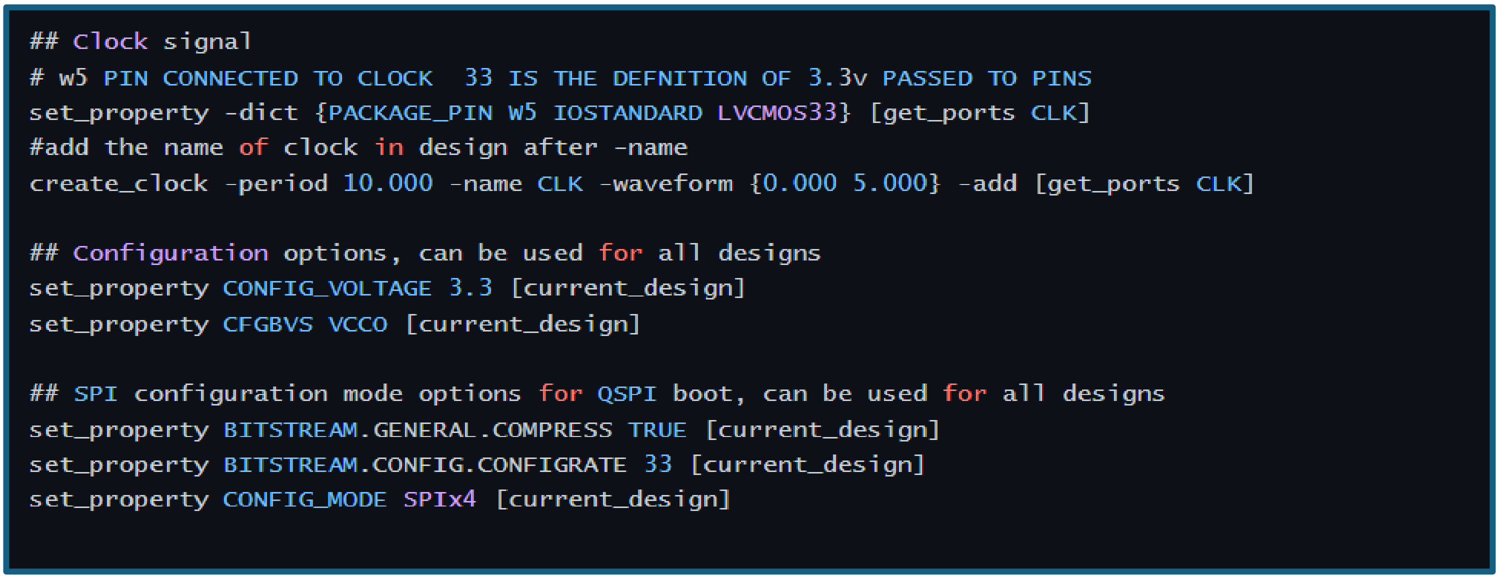
66 ps to 712 ps	/DSP_tb/CARRYOUTF

6. Transcript



```
VSIM 3> run -all
#         start simulation :)
#   =====================================
# ---- The testbench is done successfully :) --------
# ** Note: $stop    : DSP_TB.v(267)
#    Time: 680 ps  Iteration: 1  Instance: /DSP_tb
# Break in Module DSP_tb at DSP_TB.v line 267
```

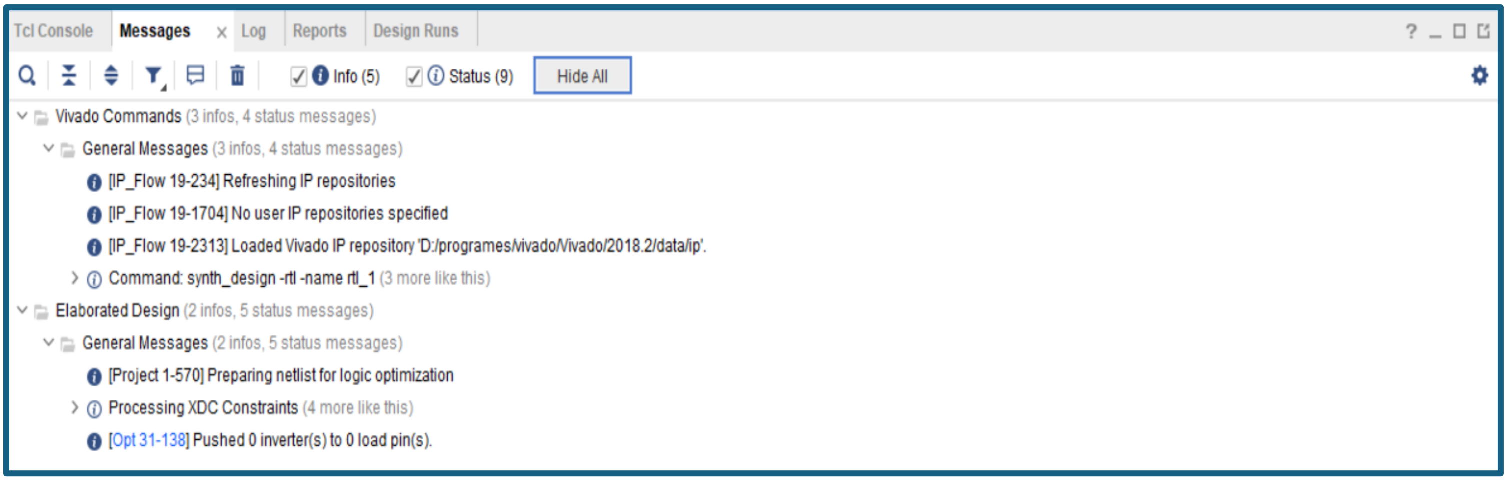
7. Constraints



```
## Clock signal
# w5 PIN CONNECTED TO CLOCK  33 IS THE DEFNITION OF 3.3v PASSED TO PINS
set_property -dict {PACKAGE_PIN W5 IOSTANDARD LVCMOS33} [get_ports CLK]
#add the name of clock in design after -name
create_clock -period 10.000 -name CLK -waveform {0.000 5.000} -add [get_ports CLK]

## Configuration options, can be used for all designs
set_property CONFIG_VOLTAGE 3.3 [current_design]
set_property CFGBVS VCCO [current_design]

## SPI configuration mode options for QSPI boot, can be used for all designs
set_property BITSTREAM.GENERAL.COMPRESS TRUE [current_design]
set_property BITSTREAM.CONFIG.CONFIGRATE 33 [current_design]
set_property CONFIG_MODE SPIx4 [current_design]
```

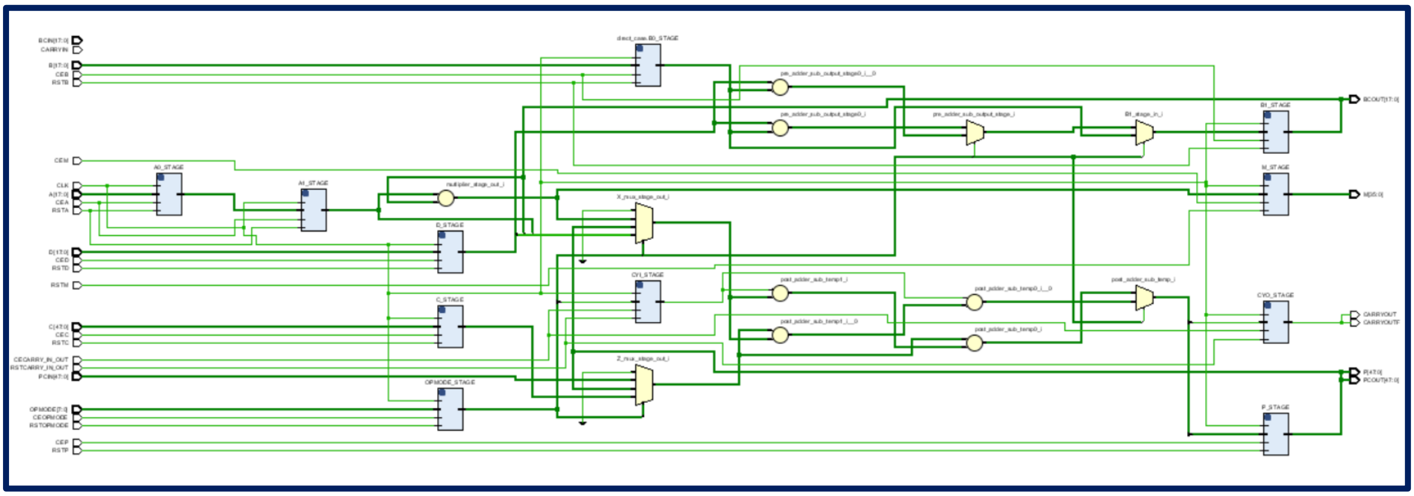
8. Elaboration

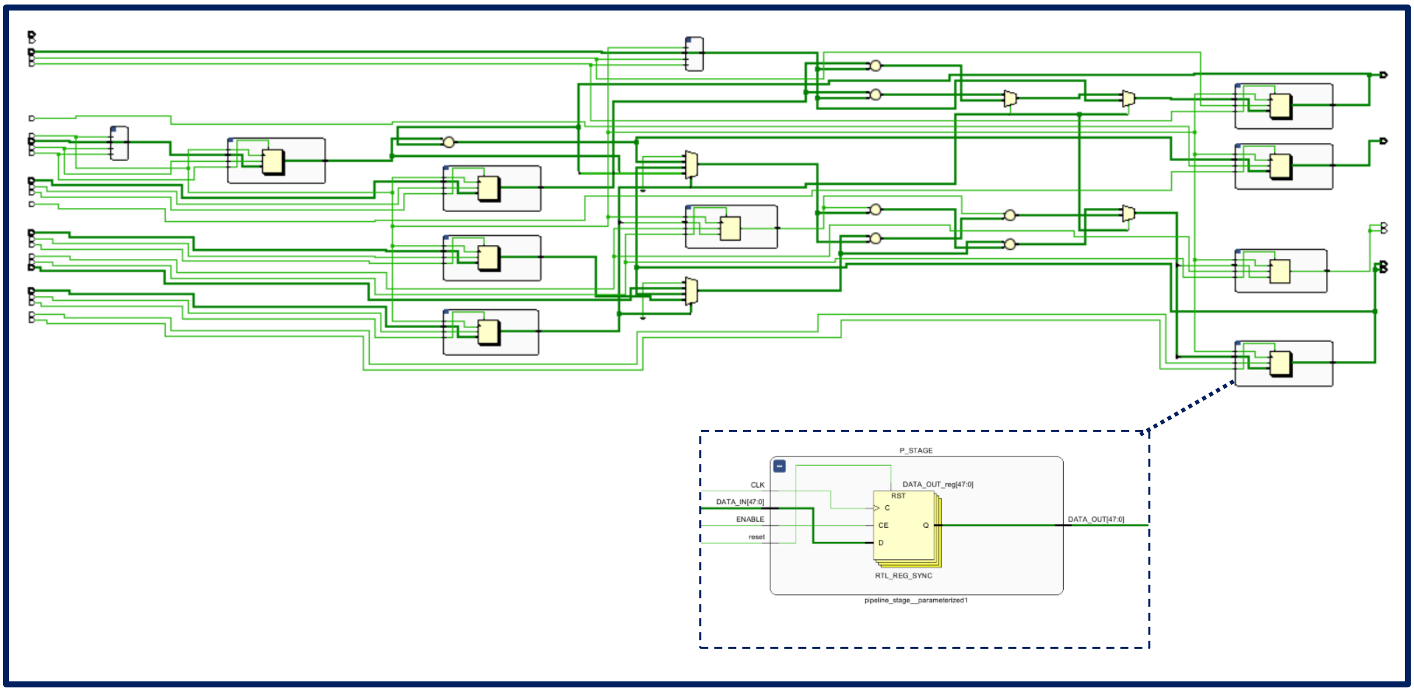
8.1. Messages tab



```
Vivado Commands (3 infos, 4 status messages)
  General Messages (3 infos, 4 status messages)
    [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:/programes/vivado/Vivado/2018.2/data/ip'.
    Command: synth_design -rtl -name rtl_1 (3 more like this)
Elaborated Design (2 infos, 5 status messages)
  General Messages (2 infos, 5 status messages)
    [Project 1-570] Preparing netlist for logic optimization
    Processing XDC Constraints (4 more like this)
    [Opt 31-138] Pushed 0 inverter(s) to 0 load pin(s).
```

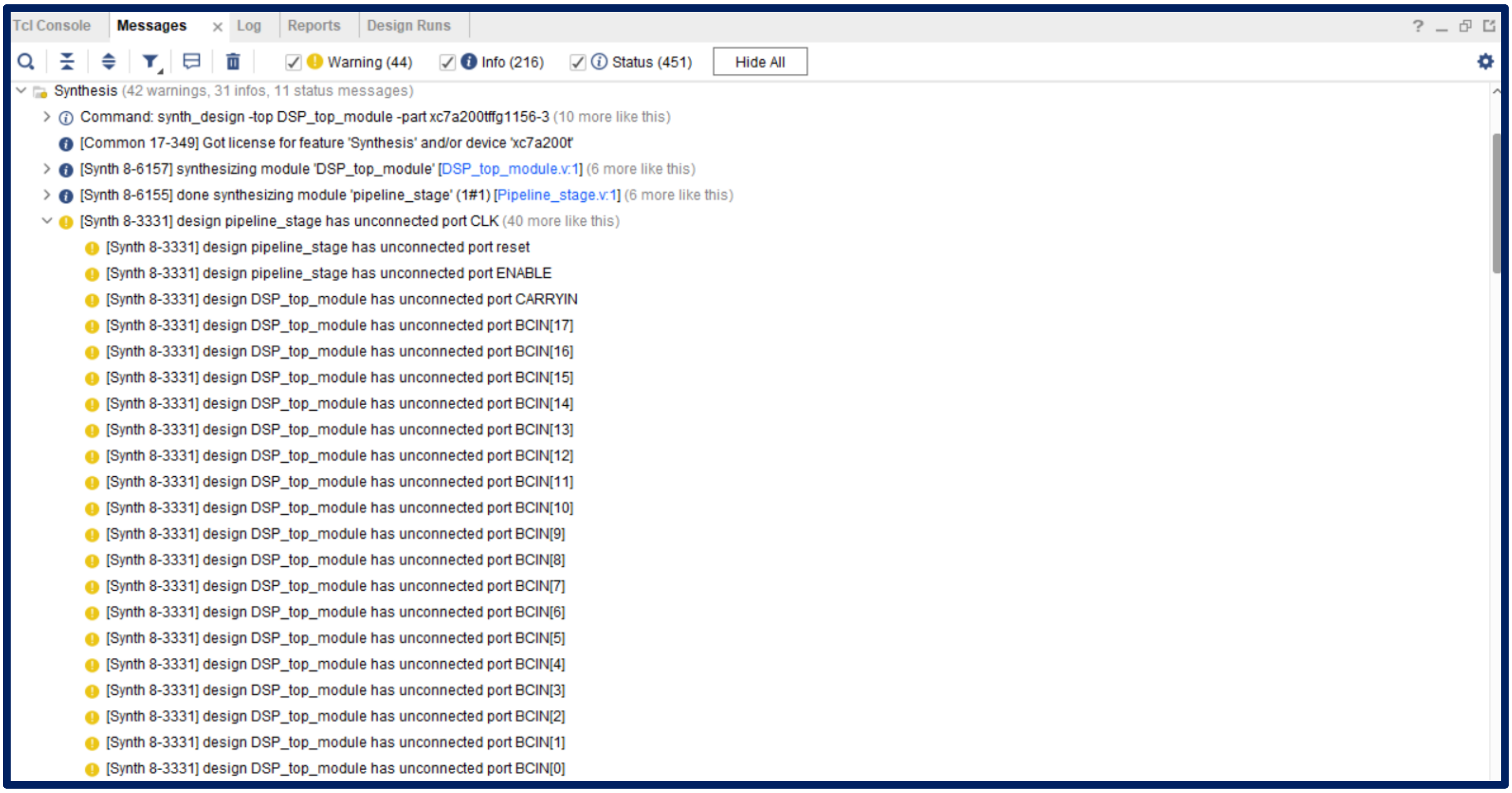
8.2. Schematic





9. Synthesis

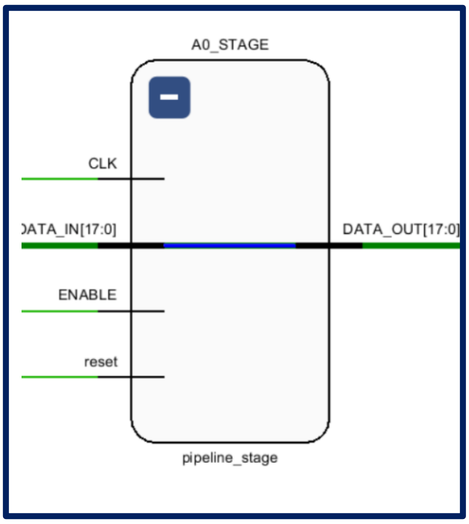
9.1. Messages tab



```
Synthesis (42 warnings, 31 infos, 11 status messages)
  Command: synth_design -top DSP_top_module -part xc7a200tffg1156-3 (10 more like this)
  [Common 17-349] Got license for feature 'Synthesis' and/or device 'xc7a200t'
  [Synth 8-6157] synthesizing module 'DSP_top_module' [DSP_top_module.v:1] (6 more like this)
  [Synth 8-6155] done synthesizing module 'pipeline_stage' (1#1) [Pipeline_stage.v:1] (6 more like this)
  [Synth 8-3331] design pipeline_stage has unconnected port CLK (40 more like this)
    [Synth 8-3331] design pipeline_stage has unconnected port reset
    [Synth 8-3331] design pipeline_stage has unconnected port ENABLE
    [Synth 8-3331] design DSP_top_module has unconnected port CARRYIN
    [Synth 8-3331] design DSP_top_module has unconnected port BCIN[17]
    [Synth 8-3331] design DSP_top_module has unconnected port BCIN[16]
    [Synth 8-3331] design DSP_top_module has unconnected port BCIN[15]
    [Synth 8-3331] design DSP_top_module has unconnected port BCIN[14]
    [Synth 8-3331] design DSP_top_module has unconnected port BCIN[13]
    [Synth 8-3331] design DSP_top_module has unconnected port BCIN[12]
    [Synth 8-3331] design DSP_top_module has unconnected port BCIN[11]
    [Synth 8-3331] design DSP_top_module has unconnected port BCIN[10]
    [Synth 8-3331] design DSP_top_module has unconnected port BCIN[9]
    [Synth 8-3331] design DSP_top_module has unconnected port BCIN[8]
    [Synth 8-3331] design DSP_top_module has unconnected port BCIN[7]
    [Synth 8-3331] design DSP_top_module has unconnected port BCIN[6]
    [Synth 8-3331] design DSP_top_module has unconnected port BCIN[5]
    [Synth 8-3331] design DSP_top_module has unconnected port BCIN[4]
    [Synth 8-3331] design DSP_top_module has unconnected port BCIN[3]
    [Synth 8-3331] design DSP_top_module has unconnected port BCIN[2]
    [Synth 8-3331] design DSP_top_module has unconnected port BCIN[1]
    [Synth 8-3331] design DSP_top_module has unconnected port BCIN[0]
```

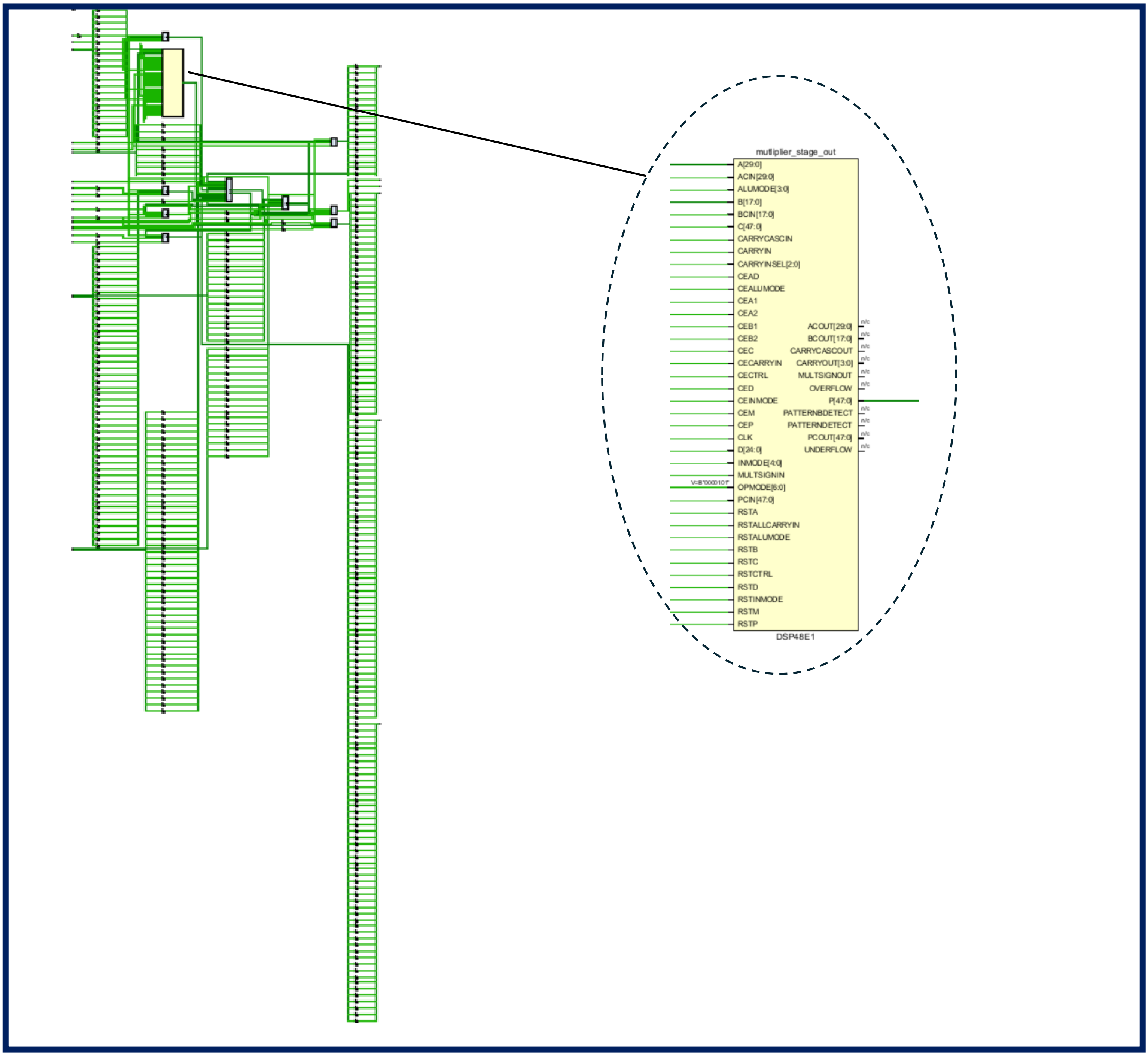
9.1.1. Comments on Warnings

At compile time, we can select whether a stage is to be registered or combinational. For stages designated as combinational, the Pipeline Stage functions as a direct connection between the input and output. This configuration results in other inputs, such as Enable and Clock, being left unconnected.

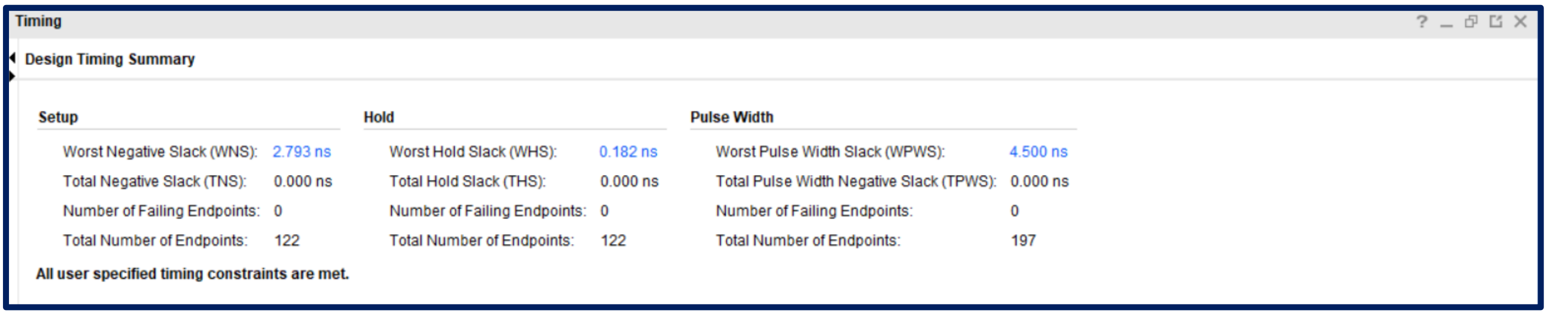


Regarding the warning about the B Cascaded Input (BCIN), this is because the port is not driven. This occurs as we have designed a single stage of the DSP48A1 slice and have not implemented cascading slices.

9.2. Netlist Generation



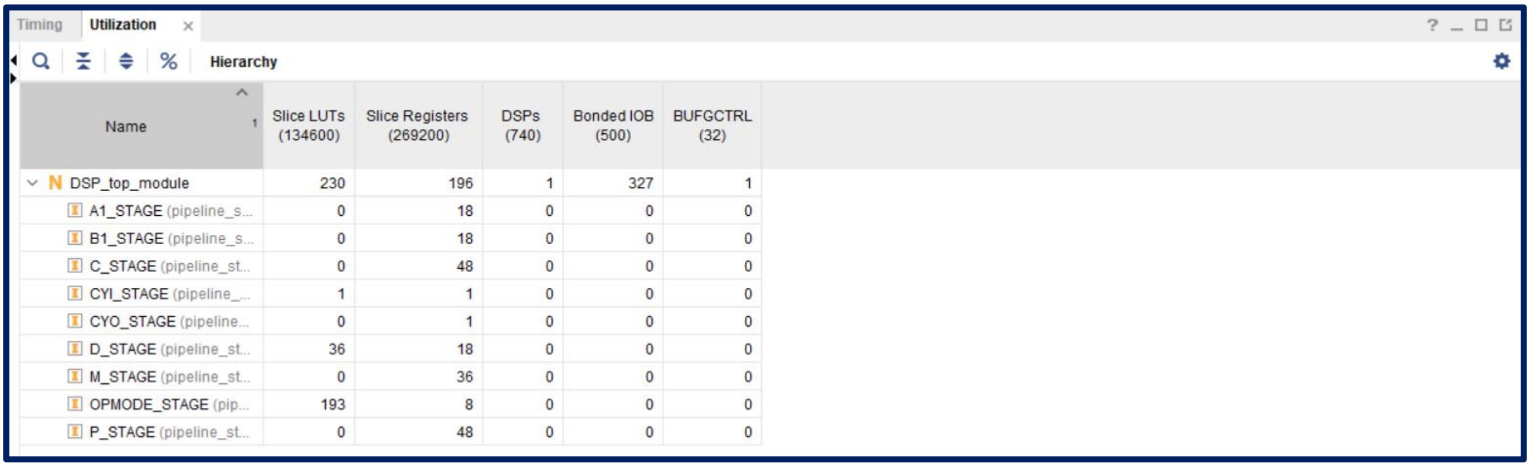
9.3. Report timing summary



Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	2.793 ns	Worst Hold Slack (WHS):	0.182 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	122	Total Number of Endpoints:	122	Total Number of Endpoints:	197

All user specified timing constraints are met.

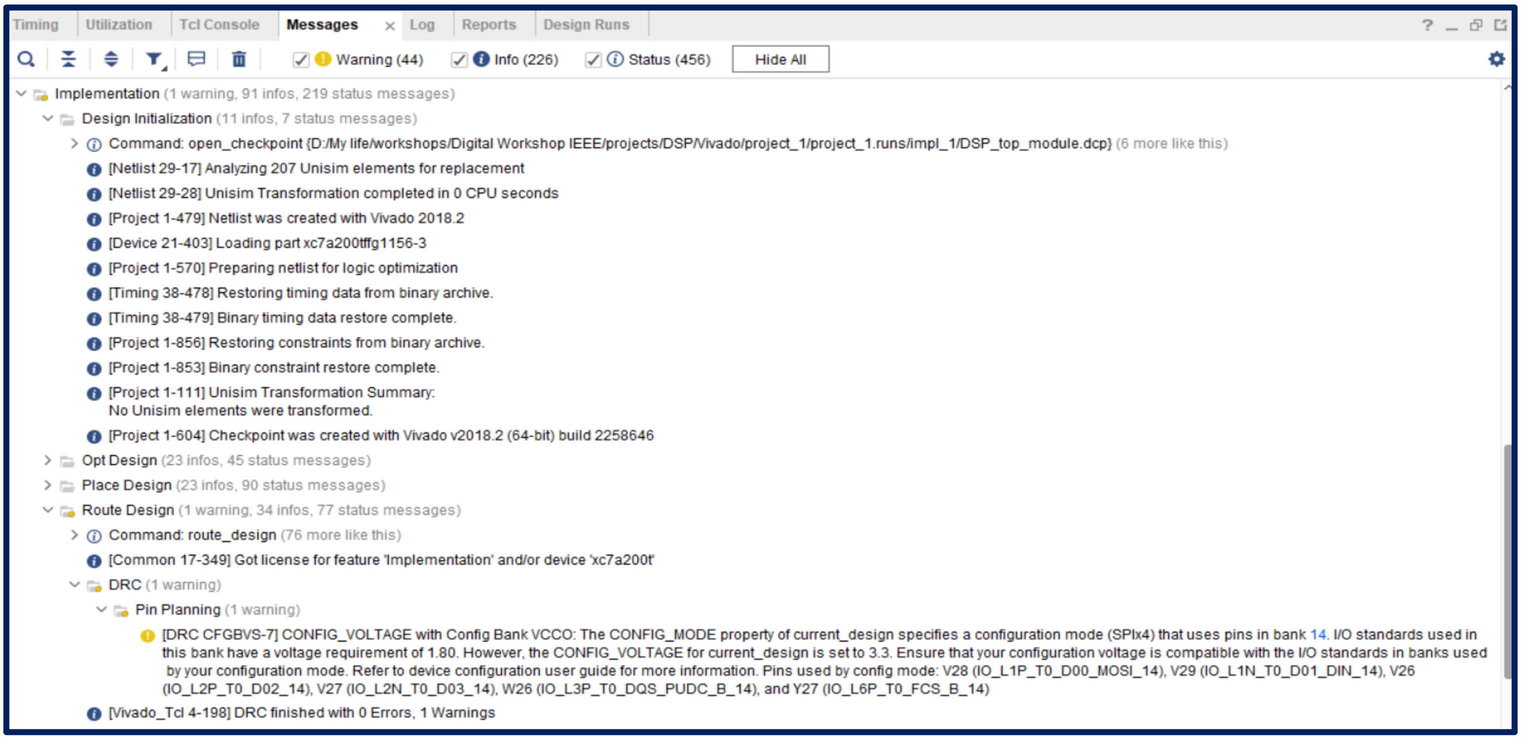
9.4. Utilization Report



Name	Slice LUTs (134600)	Slice Registers (269200)	DSPs (740)	Bonded IOB (500)	BUFGCTRL (32)
DSP_top_module	230	196	1	327	1
A1_STAGE (pipeline_s...	0	18	0	0	0
B1_STAGE (pipeline_s...	0	18	0	0	0
C_STAGE (pipeline_st...	0	48	0	0	0
CYI_STAGE (pipeline_...	1	1	0	0	0
CYO_STAGE (pipeline...	0	1	0	0	0
D_STAGE (pipeline_st...	36	18	0	0	0
M_STAGE (pipeline_st...	0	36	0	0	0
OPMODE_STAGE (pip...	193	8	0	0	0
P_STAGE (pipeline_st...	0	48	0	0	0

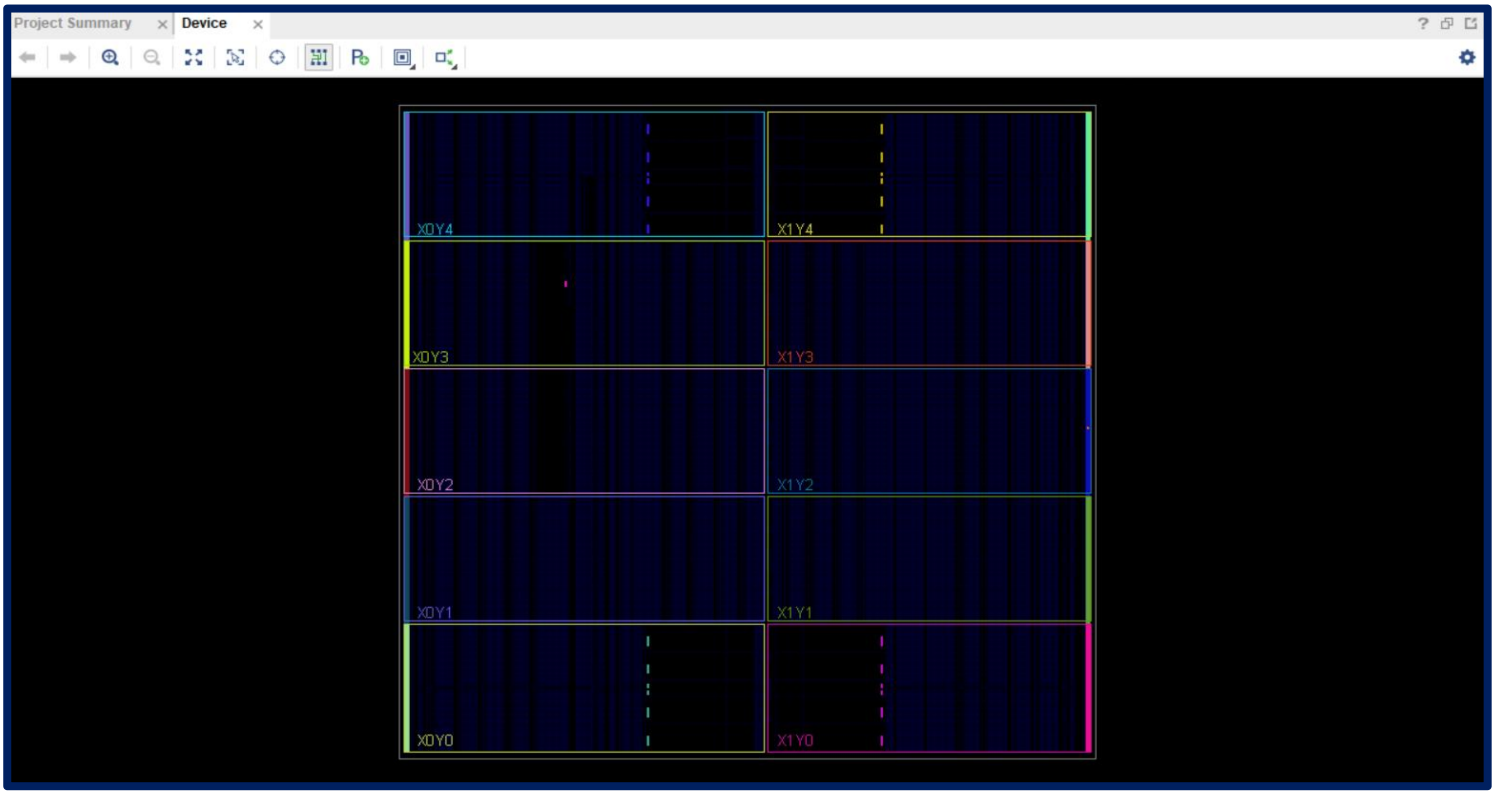
10. Implementation

10.1. Message tab

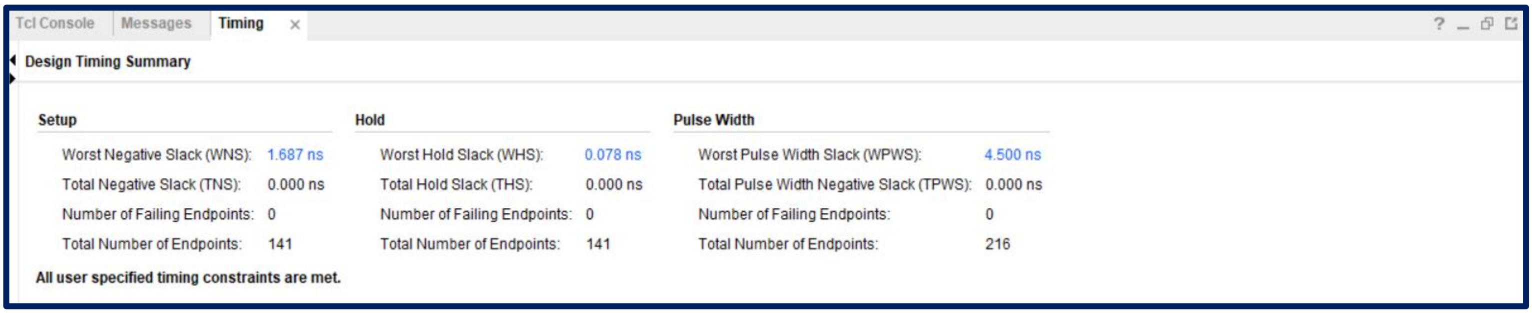


```
Implementation (1 warning, 91 infos, 219 status messages)
  Design Initialization (11 infos, 7 status messages)
    Command: open_checkpoint {D:/My life/workshops/Digital Workshop IEEE/projects/DSP/Vivado/project_1/project_1.runs/impl_1/DSP_top_module.dcp} (6 more like this)
    [Netlist 29-17] Analyzing 207 Unisim elements for replacement
    [Netlist 29-28] Unisim Transformation completed in 0 CPU seconds
    [Project 1-479] Netlist was created with Vivado 2018.2
    [Device 21-403] Loading part xc7a200tffg1156-3
    [Project 1-570] Preparing netlist for logic optimization
    [Timing 38-478] Restoring timing data from binary archive.
    [Timing 38-479] Binary timing data restore complete.
    [Project 1-856] Restoring constraints from binary archive.
    [Project 1-853] Binary constraint restore complete.
    [Project 1-111] Unisim Transformation Summary:
    No Unisim elements were transformed.
    [Project 1-604] Checkpoint was created with Vivado v2018.2 (64-bit) build 2258646
  Opt Design (23 infos, 45 status messages)
  Place Design (23 infos, 90 status messages)
  Route Design (1 warning, 34 infos, 77 status messages)
    Command: route_design (76 more like this)
    [Common 17-349] Got license for feature 'Implementation' and/or device 'xc7a200t'
    DRC (1 warning)
      Pin Planning (1 warning)
        [DRC CFGBVS-7] CONFIG_VOLTAGE with Config Bank VCCO: The CONFIG_MODE property of current_design specifies a configuration mode (SPIx4) that uses pins in bank 14. I/O standards used in
        this bank have a voltage requirement of 1.80. However, the CONFIG_VOLTAGE for current_design is set to 3.3. Ensure that your configuration voltage is compatible with the I/O standards in banks used
         by your configuration mode. Refer to device configuration user guide for more information. Pins used by config mode: V28 (IO_L1P_T0_D00_MOSI_14), V29 (IO_L1N_T0_D01_DIN_14), V26
        (IO_L2P_T0_D02_14), V27 (IO_L2N_T0_D03_14), W26 (IO_L3P_T0_DQS_PUDC_B_14), and Y27 (IO_L6P_T0_FCS_B_14)
    [Vivado_Tcl 4-198] DRC finished with 0 Errors, 1 Warnings
```

10.2. Device



10.3. Design timing summary



Design Timing Summary

Setup

Worst Negative Slack (WNS): 1.687 ns
Total Negative Slack (TNS): 0.000 ns
Number of Failing Endpoints: 0
Total Number of Endpoints: 141

Hold

Worst Hold Slack (WHS): 0.078 ns
Total Hold Slack (THS): 0.000 ns
Number of Failing Endpoints: 0
Total Number of Endpoints: 141

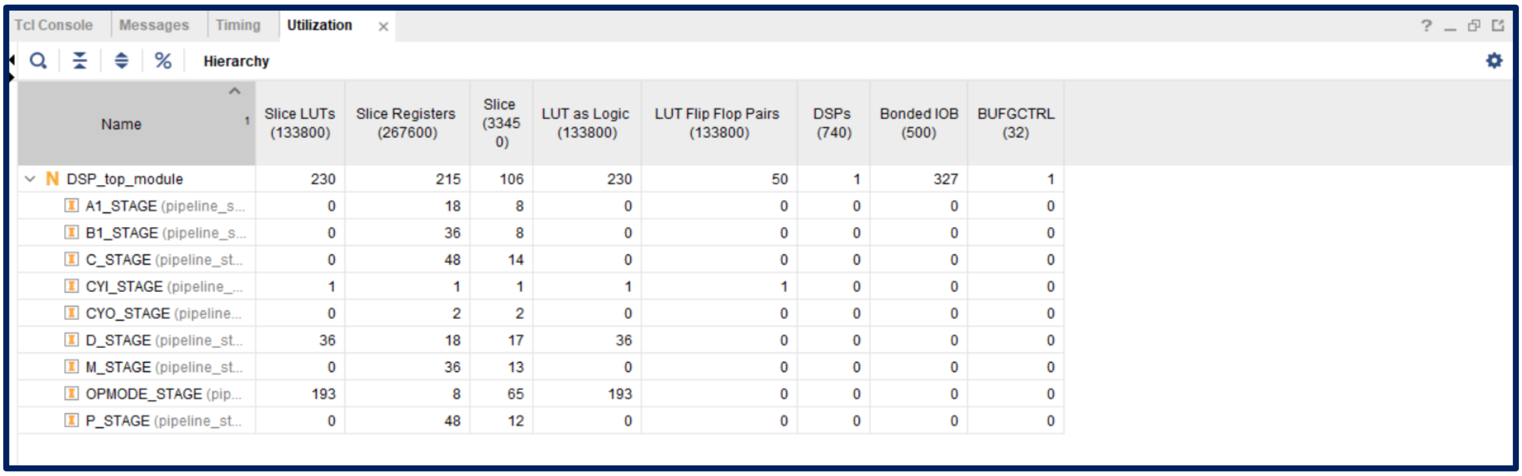
Pulse Width

Worst Pulse Width Slack (WPWS): 4.500 ns
Total Pulse Width Negative Slack (TPWS): 0.000 ns
Number of Failing Endpoints: 0
Total Number of Endpoints: 216

All user specified timing constraints are met.

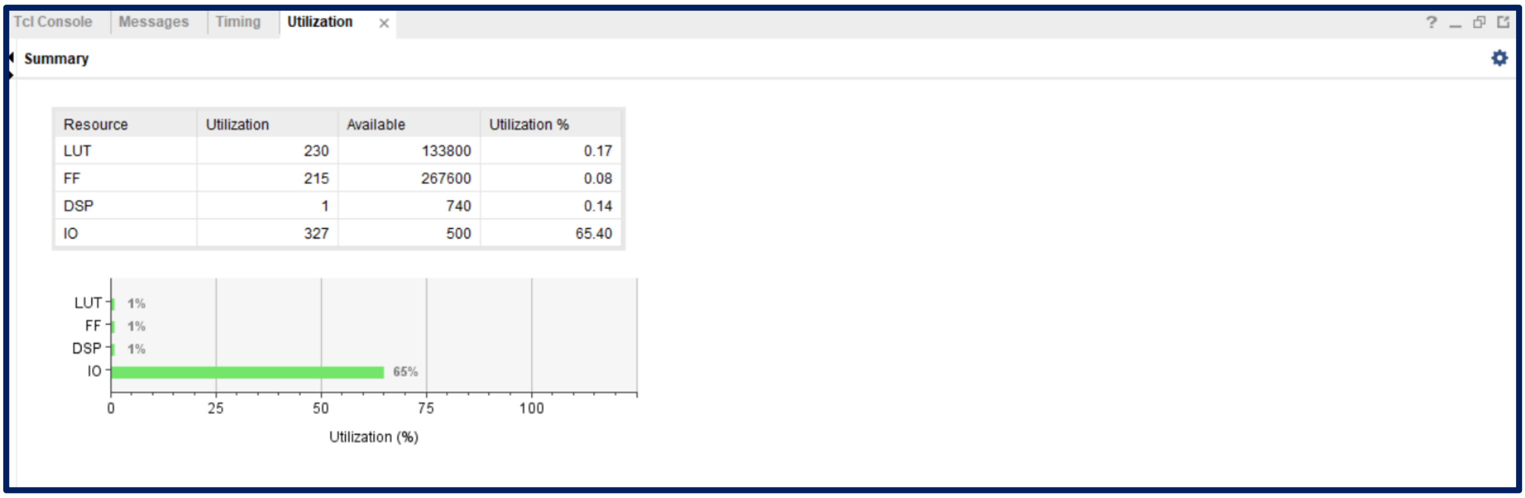
10.4. Utilization Report

10.4.1. Hierarchy



Name	Slice LUTs (133800)	Slice Registers (267600)	Slice (33450)	LUT as Logic (133800)	LUT Flip Flop Pairs (133800)	DSPs (740)	Bonded IOB (500)	BUFGCTRL (32)
N DSP_top_module	230	215	106	230	50	1	327	1
A1_STAGE (pipeline_s...	0	18	8	0	0	0	0	0
B1_STAGE (pipeline_s...	0	36	8	0	0	0	0	0
C_STAGE (pipeline_st...	0	48	14	0	0	0	0	0
CYI_STAGE (pipeline_...	1	1	1	1	1	0	0	0
CYO_STAGE (pipeline...	0	2	2	0	0	0	0	0
D_STAGE (pipeline_st...	36	18	17	36	0	0	0	0
M_STAGE (pipeline_st...	0	36	13	0	0	0	0	0
OPMODE_STAGE (pip...	193	8	65	193	0	0	0	0
P_STAGE (pipeline_st...	0	48	12	0	0	0	0	0

10.4.2. Summary



Resource	Utilization	Available	Utilization %
LUT	230	133800	0.17
FF	215	267600	0.08
DSP	1	740	0.14
IO	327	500	65.40