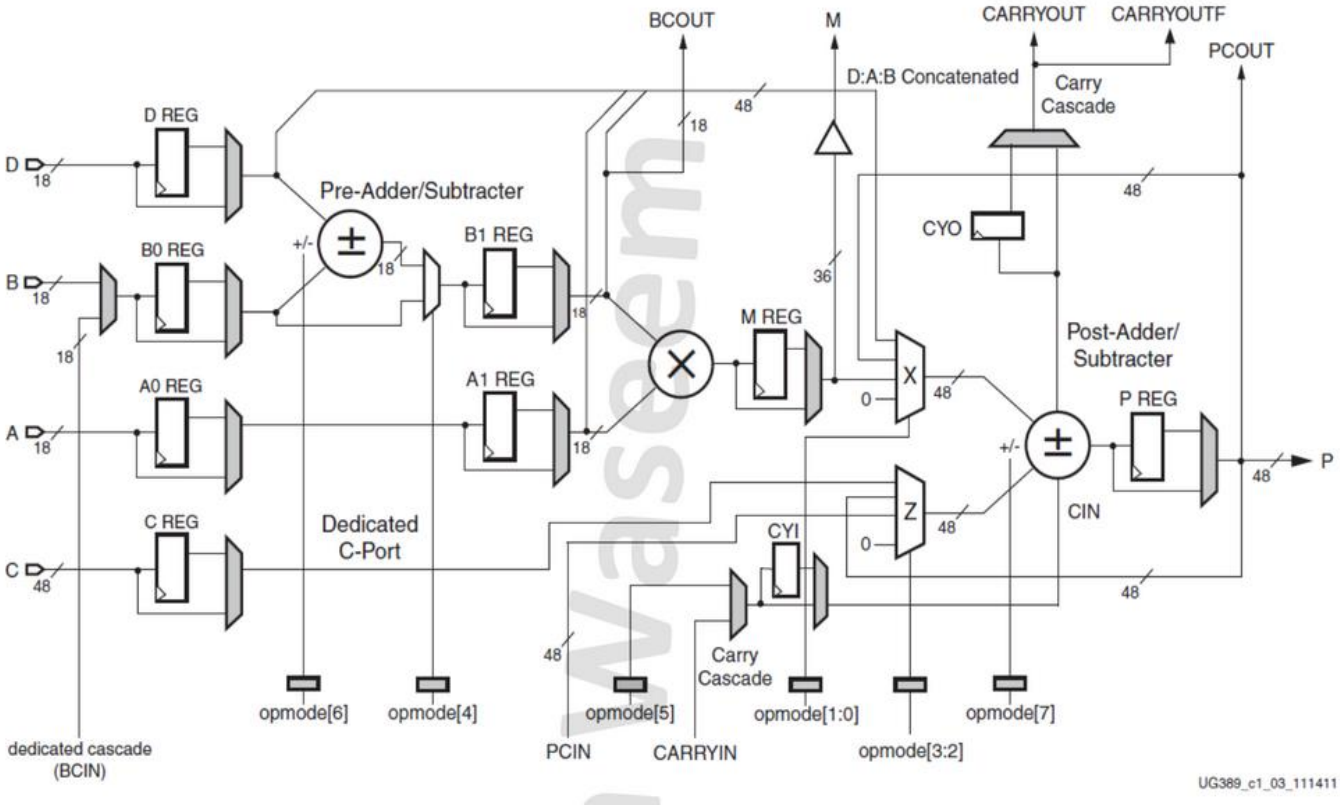
DSP48A1-Spartan6 Project

By

Karim Yehia

DSP Schematic:



RTL Code:

Register Mux Building Block:

```
module reg_mux(clk, clk_enable, s, d, rst, out);
parameter N = 1, RSTTYPE = "SYNC";
input clk, clk_enable, rst, s;
input [N-1:0] d;
output [N-1:0] out;

reg [N-1:0] out_reg;

generate
    if(RSTTYPE == "SYNC") begin
        always @(posedge clk) begin
            if(rst)
                out_reg <= 0;
            else if(clk_enable)
                out_reg <= d;
            end
    end

    else if(RSTTYPE == "ASYNC") begin
      always @(posedge clk or posedge rst) begin
            if(rst)
                out_reg <= 0;
            else if(clk_enable)
                out_reg <= d;
        end
    end
endgenerate

assign out = s ? out_reg : d;
endmodule
```

DSP48A!:

```
module DSP48A1(A, B, D, C, CLK, CARRYIN, OPMODE, BCIN, RSTA, RSTB, RSTM, RSTP, RSTC, RSTD,
               RSTCARRYIN, RSTOPMODE, CEA, CEB, CEM, CEP, CEC, CED, CECARRYIN, CEOPMODE, PCIN,
               PCOUT, BCOUT, P, M, CARRYOUT, CARRYOUTF);
parameter A0REG = 0, B0REG = 0;
parameter A1REG = 1, B1REG = 1, CREG = 1, DREG = 1, MREG = 1, PREG = 1, CARRYINREG = 1, CARRYOUTREG = 1, OPMODEREG = 1;
parameter CARRYINSEL = "OPMODE5"; //valid values are "OPMODE5" & "CARRYIN"
parameter B_INPUT = "DIRECT"; // valid values are "DIRECT" & "CASCADE"
parameter RSTTYPE = "SYNC"; // valid values are "SYNC" & "ASYNC"
input [17:0] A, B, D, BCIN;
input [47:0] C, PCIN;
input [7:0] OPMODE;
input CLK, CARRYIN, RSTA, RSTB, RSTC, RSTM, RSTP, RSTD, RSTCARRYIN, RSTOPMODE;
input CEA, CEB, CEC, CEM, CEP, CED, CECARRYIN, CEOPMODE;
output [17:0] BCOUT;
output [47:0] PCOUT, P;
output [35:0] M;
output CARRYOUT, CARRYOUTF;

// First Stage
wire [17:0] d_reg_out;
reg_mux #(.N(18), .RSTTYPE(RSTTYPE)) D_REGISTER(CLK, CED, DREG, D, RSTD, d_reg_out);

wire [17:0] b0_reg_in;
wire [17:0] b0_reg_out;
assign b0_reg_in = (B_INPUT == "DIRECT") ? B : BCIN;
reg_mux #(.N(18), .RSTTYPE(RSTTYPE)) B0_REGISTER(CLK, CEB, B0REG, b0_reg_in, RSTB, b0_reg_out);

wire [17:0] a0_reg_out;
reg_mux #(.N(18), .RSTTYPE(RSTTYPE)) A0_REGISTER(CLK, CEA, A0REG, A, RSTA, a0_reg_out);

wire [47:0] c_reg_out;
reg_mux #(.N(48), .RSTTYPE(RSTTYPE)) C_REGISTER(CLK, CEC, CREG, C, RSTC, c_reg_out);

// opcode stages
wire op6_out;
reg_mux #(.N(1), .RSTTYPE(RSTTYPE)) OP6_REGISTER(CLK, CEOPMODE, OPMODEREG, OPMODE[6], RSTOPMODE, op6_out);

wire op4_out;
reg_mux #(.N(1), .RSTTYPE(RSTTYPE)) OP4_REGISTER(CLK, CEOPMODE, OPMODEREG, OPMODE[4], RSTOPMODE, op4_out);

wire op5_out;
reg_mux #(.N(1), .RSTTYPE(RSTTYPE)) OP5_REGISTER(CLK, CEOPMODE, OPMODEREG, OPMODE[5], RSTOPMODE, op5_out);

wire op7_out;
reg_mux #(.N(1), .RSTTYPE(RSTTYPE)) OP7_REGISTER(CLK, CEOPMODE, OPMODEREG, OPMODE[7], RSTOPMODE, op7_out);

wire [1:0] op1to0_out;
reg_mux #(.N(2), .RSTTYPE(RSTTYPE)) OP1TO0_REGISTER(CLK, CEOPMODE, OPMODEREG, OPMODE[1:0], RSTOPMODE, op1to0_out);

wire [1:0] op3to2_out;
reg_mux #(.N(2), .RSTTYPE(RSTTYPE)) OP3TO2_REGISTER(CLK, CEOPMODE, OPMODEREG, OPMODE[3:2], RSTOPMODE, op3to2_out);

// second stage
wire [17:0] pre_add_substract;
assign pre_add_substract = (op6_out) ? (d_reg_out - b0_reg_out) : (d_reg_out + b0_reg_out);

wire [17:0] b1_reg_in;
assign b1_reg_in = (op4_out) ? (pre_add_substract) : (b0_reg_out);

wire [17:0] b1_reg_out;
reg_mux #(.N(18), .RSTTYPE(RSTTYPE)) B1_REGISTER(CLK, CEB, B1REG, b1_reg_in, RSTB, b1_reg_out);

wire [17:0] a1_reg_out;
reg_mux #(.N(18), .RSTTYPE(RSTTYPE)) A1_REGISTER(CLK, CEA, A1REG, a0_reg_out, RSTA, a1_reg_out);

assign BCOUT = b1_reg_out;

wire [35:0] m_reg_in, m_reg_out;
assign m_reg_in = a1_reg_out * b1_reg_out;
reg_mux #(.N(36), .RSTTYPE(RSTTYPE)) M_REGISTER(CLK, CEM, MREG, m_reg_in, RSTM, m_reg_out);
assign M = m_reg_out;

wire cyi_in, CIN;
assign cyi_in = (CARRYINSEL == "OPMODE5") ? op5_out : CARRYIN;
reg_mux #(.N(1), .RSTTYPE(RSTTYPE)) CYI_REGISTER(CLK, CECARRYIN, CARRYINREG, cyi_in, RSTCARRYIN, CIN);

// X & Z multiplexers
// X mux
wire [47:0] X, Z;
assign X = (op1to0_out == 3) ? ({d_reg_out[11:0], a1_reg_out, b1_reg_out}) :
                        (op1to0_out == 2) ? (P) :
                        (op1to0_out == 1) ? ({{12{1'b0}}, m_reg_out}) :
                        0;
assign Z = (op3to2_out == 3) ? (c_reg_out) :
           (op3to2_out == 2) ? (P) :
           (op3to2_out == 1) ? (PCIN) :
             0;
wire [47:0] post_add_sub_out;
wire cyo_in;
assign {cyo_in, post_add_sub_out} = (op7_out) ? (Z - (X + CIN)) : (Z + X + CIN);

reg_mux #(.N(48), .RSTTYPE(RSTTYPE)) P_REGISTER(CLK, CEP, PREG, post_add_sub_out, RSTP, P);
assign PCOUT = P;

reg_mux #(.N(1), .RSTTYPE(RSTTYPE)) CYO_REGISTER(CLK, CECARRYIN, CARRYOUTREG, cyo_in, RSTCARRYIN, CARRYOUT);
assign CARRYOUTF = CARRYOUT;

endmodule
```

Testbench:

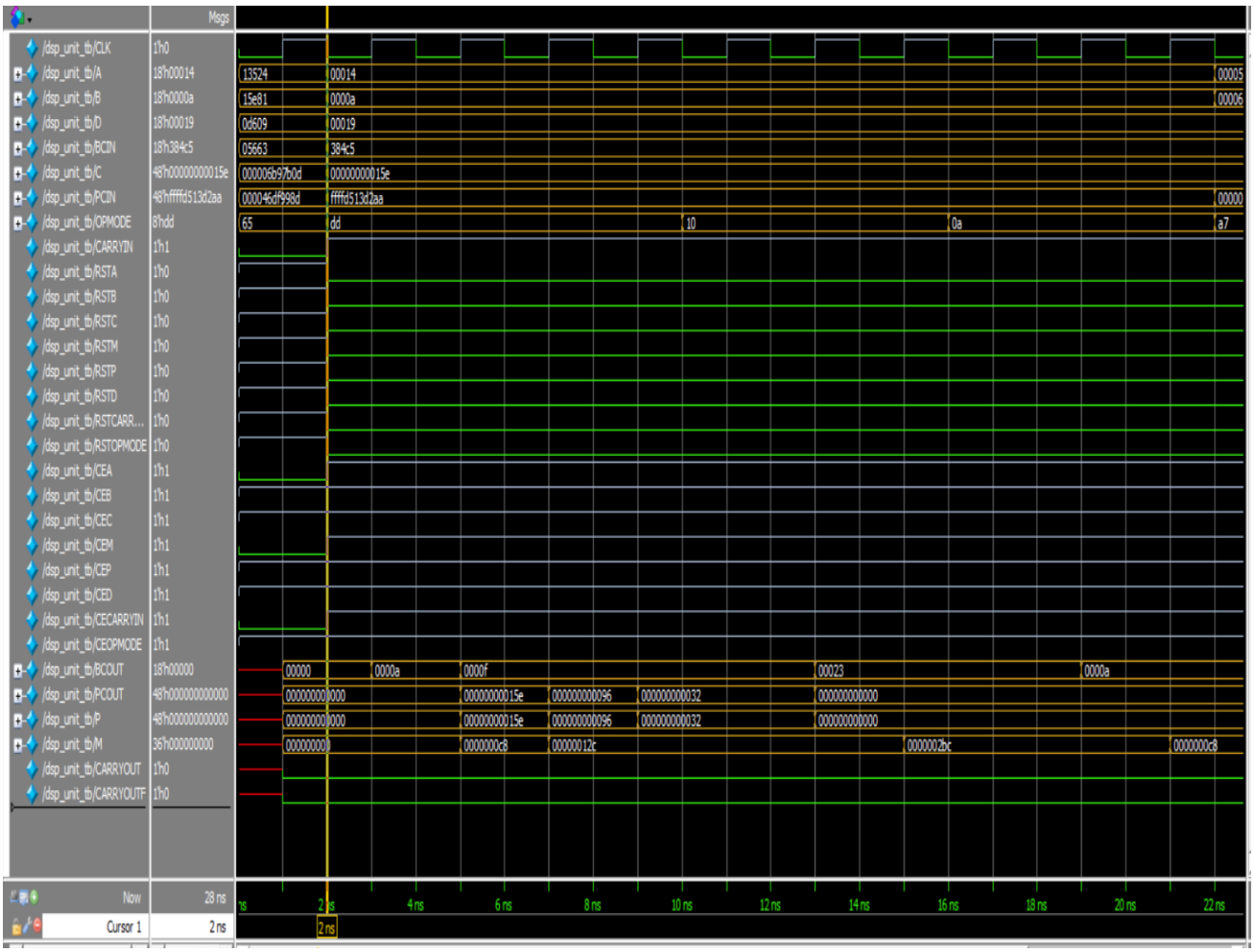
```
module dsp_unit_tb();
reg [17:0] A, B, D, BCIN;
reg [47:0] C, PCIN;
reg [7:0] OPMODE;
reg CLK, CARRYIN, RSTA, RSTB, RSTC, RSTM, RSTP, RSTD, RSTCARRYIN, RSTOPMODE;
reg CEA, CEB, CEC, CEM, CEP, CED, CECARRYIN, CEOPMODE;
wire [17:0] BCOUT;
wire [47:0] PCOUT, P;
wire [35:0] M;
wire CARRYOUT, CARRYOUTF;

DSP48A1 dut(A, B, D, C, CLK, CARRYIN, OPMODE, BCIN, RSTA, RSTB, RSTM, RSTP, RSTC, RSTD,
               RSTCARRYIN, RSTOPMODE, CEA, CEB, CEM, CEP, CEC, CED, CECARRYIN, CEOPMODE, PCIN,
               PCOUT, BCOUT, P, M, CARRYOUT, CARRYOUTF);
initial begin
    CLK = 0;
   forever #1 CLK = ~CLK;
end

initial begin
    RSTA = 1; RSTB = 1; RSTC = 1; RSTCARRYIN = 1; RSTM = 1; RSTP = 1; RSTD = 1; RSTOPMODE = 1;
    A = $random; B = $random; D = $random; BCIN = $random;
    C = $random; PCIN = $random;
    OPMODE = $random;
    CEA = $random; CEB = $random; CEC = $random; CEM = $random; CEP = $random; CED = $random;
    CECARRYIN = $random; CEOPMODE = $random; CARRYIN = $random;
    @(negedge CLK);
    // path1 test
    RSTA = 0; RSTB = 0; RSTC = 0; RSTCARRYIN = 0; RSTM = 0; RSTP = 0; RSTD = 0; RSTOPMODE = 0;
    CEA = 1; CEB = 1; CEC = 1; CEM = 1; CEP = 1; CED = 1; CECARRYIN = 1; CEOPMODE = 1;
    OPMODE = 8'b11011101;
    A = 20; B = 10; C = 350; D = 25;
    BCIN = $random; PCIN = $random; CARRYIN = $random;
    repeat(4) @(negedge CLK);
    if((BCOUT != 18'hf) || (M != 36'h12c) || (P != 48'h32) || (CARRYOUT != 0)) begin
      $display("Error");
      $stop;
    end
    //path2 test
    OPMODE = 8'b00010000;
    repeat(3) @(negedge CLK);
    if((BCOUT != 18'h23) || (M != 36'h2bc) || (P != 0) || (CARRYOUT != 0)) begin
      $display("Error");
      $stop;
    end
    //path3 test
    OPMODE = 8'b00001010;
    repeat(3) @(negedge CLK);
    if((BCOUT != 18'ha) || (M != 36'hc8) || (P != PCOUT) || (CARRYOUT != CARRYOUTF)) begin
      $display("Error");
      $stop;
    end
    //path4 test
    OPMODE = 8'b10100111;
    A = 5; B = 6; C = 350; D = 25; PCIN = 3000;
    repeat(3) @(negedge CLK);
    if((BCOUT != 18'h6) || (M != 36'h1e) || (P != 48'hfe6fffec0bb1) || (CARRYOUT != 1)) begin
      $display("Error");
      $stop;
    end
    $stop;
    end
endmodule
```

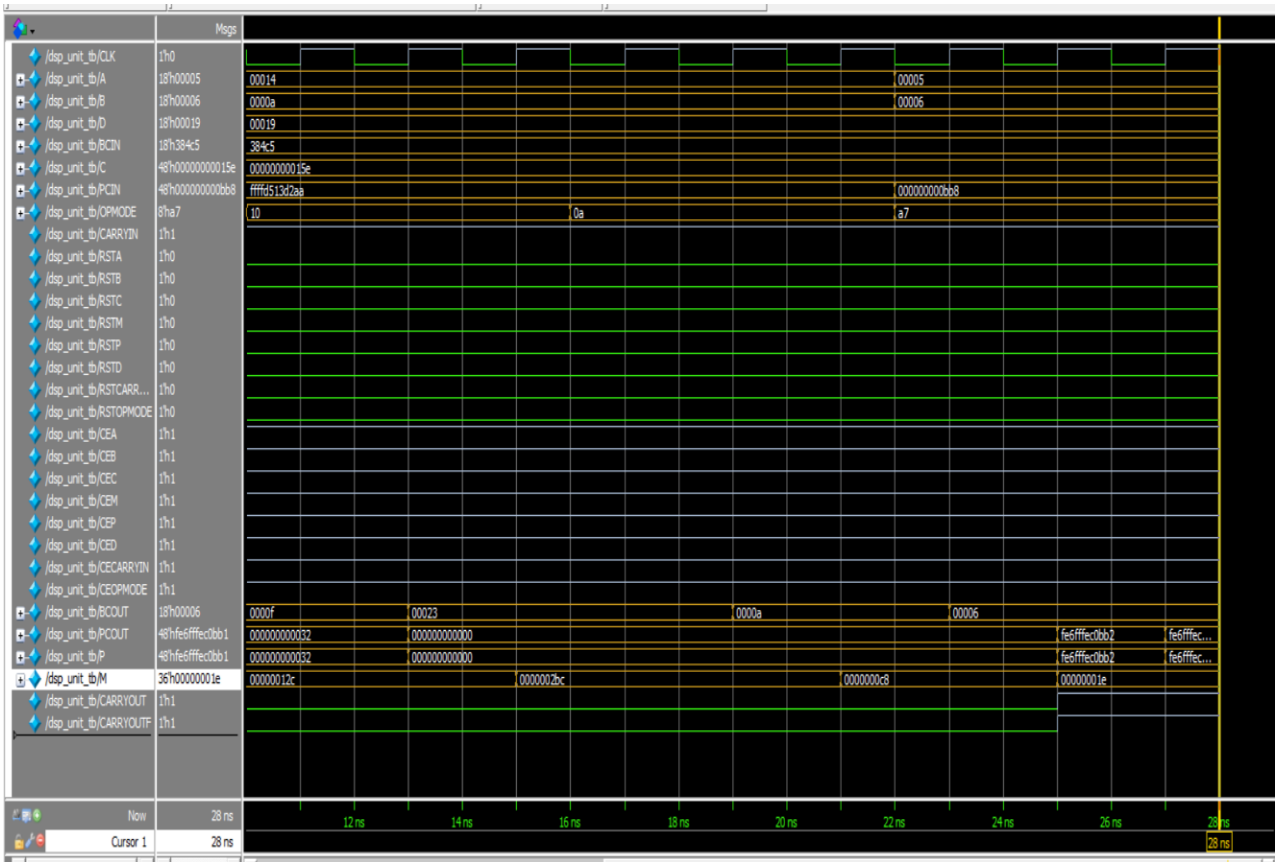
Do File:

run_tb.do

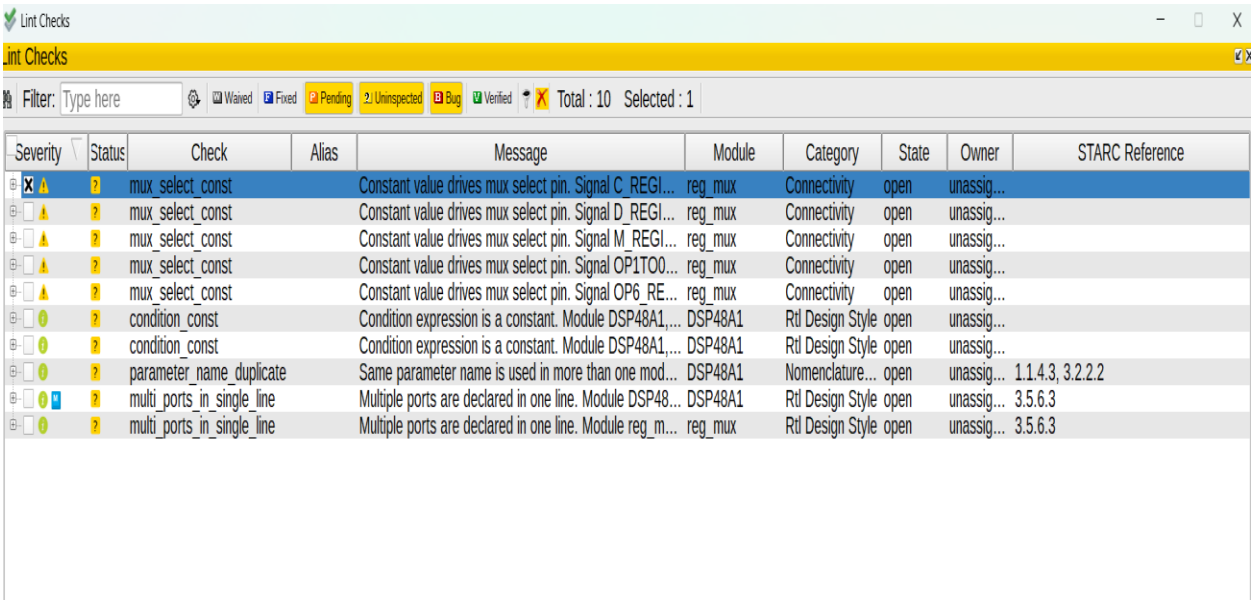
```
1   vlib work
2   vlog reg_mux.v dsp48a1.v dsp48a1_tb.v
3   vsim -voptargs=+acc work.dsp_unit_tb
4   add wave *
5   run -all
6   #quit -sim
```

Waveform:



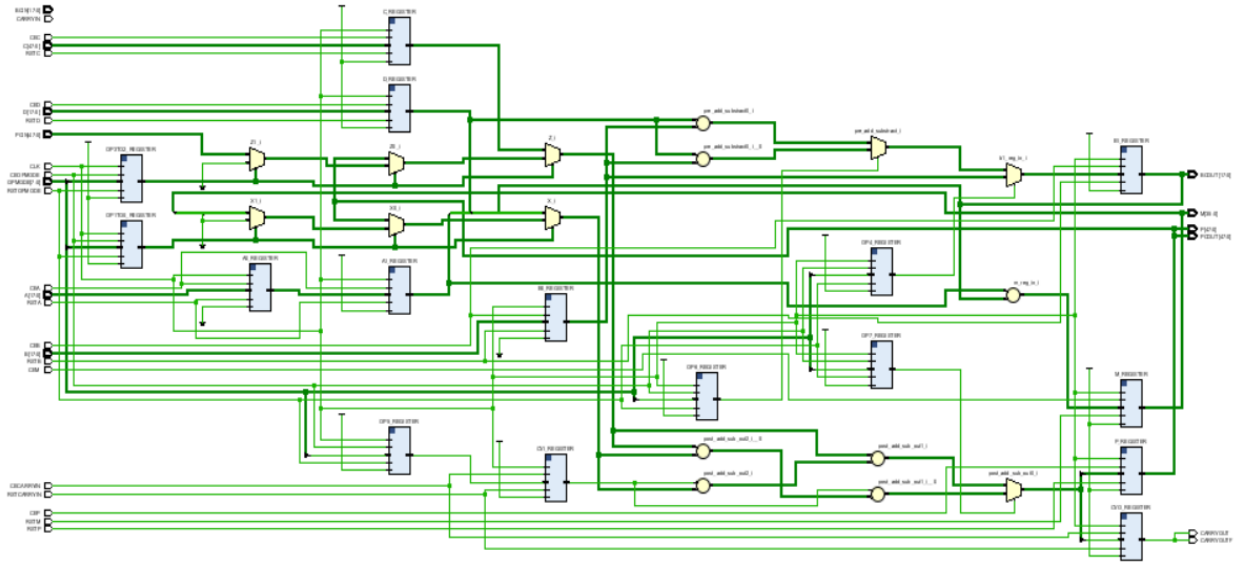


Lint: (No Errors or Critical Warnings):

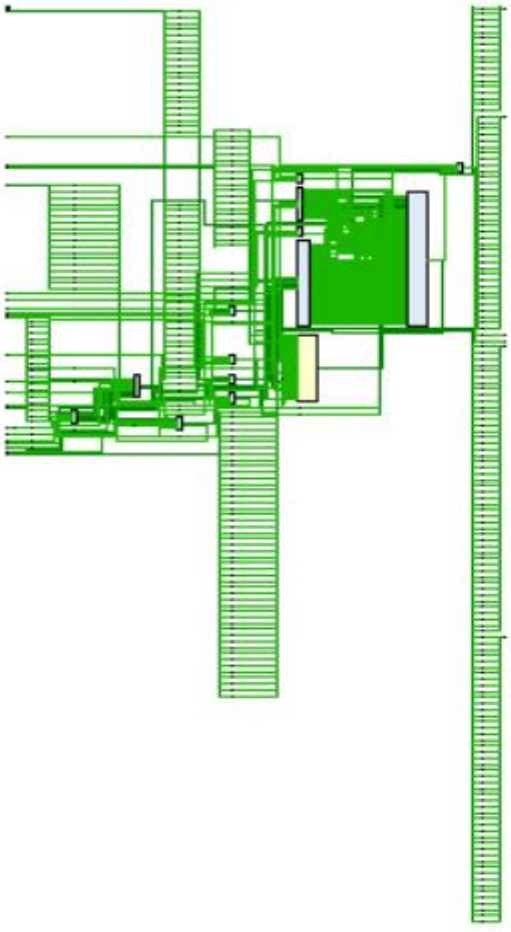


Severity	Status	Check	Alias	Message	Module	Category	State	Owner	STARC Reference
	?	mux_select_const		Constant value drives mux select pin. Signal C_REGI...	reg_mux	Connectivity	open	unassig...	
	?	mux_select_const		Constant value drives mux select pin. Signal D_REGI...	reg_mux	Connectivity	open	unassig...	
	?	mux_select_const		Constant value drives mux select pin. Signal M_REGI...	reg_mux	Connectivity	open	unassig...	
	?	mux_select_const		Constant value drives mux select pin. Signal OP1TO0...	reg_mux	Connectivity	open	unassig...	
	?	mux_select_const		Constant value drives mux select pin. Signal OP6_RE...	reg_mux	Connectivity	open	unassig...	
	?	condition_const		Condition expression is a constant. Module DSP48A1,...	DSP48A1	Rtl Design Style	open	unassig...	
	?	condition_const		Condition expression is a constant. Module DSP48A1,...	DSP48A1	Rtl Design Style	open	unassig...	
	?	parameter_name_duplicate		Same parameter name is used in more than one mod...	DSP48A1	Nomenclature...	open	unassig...	1.1.4.3, 3.2.2.2
	?	multi_ports_in_single_line		Multiple ports are declared in one line. Module DSP48...	DSP48A1	Rtl Design Style	open	unassig...	3.5.6.3
	?	multi_ports_in_single_line		Multiple ports are declared in one line. Module reg_m...	reg_mux	Rtl Design Style	open	unassig...	3.5.6.3

Elaborated Schematic:



Synthesis Schematic:



Synthesis Timing Report:

Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	5.168 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:	87	Total Number of Endpoints:	87	Total Number of Endpoints:	162

All user specified timing constraints are met.

Synthesis Utilization Report:

Name	Slice LUTs (134600)	Slice Registers (269200)	DSPs (740)	Bonded IOB (500)	BUFGCTRL (32)
N DSP48A1	230	160	1	327	1
A1_REGISTER (reg_m...	0	18	0	0	0
B1_REGISTER (reg_m...	1	18	0	0	0
C_REGISTER (reg_m...	0	48	0	0	0
CYI_REGISTER (reg_...	1	1	0	0	0
CYO_REGISTER (reg_...	0	1	0	0	0
D_REGISTER (reg_m...	16	18	0	0	0
OP1TO0_REGISTER (r...	145	2	0	0	0
OP3TO2_REGISTER (r...	48	2	0	0	0
OP4_REGISTER (reg_...	2	1	0	0	0
OP5_REGISTER (reg_...	0	1	0	0	0
OP6_REGISTER (reg_...	17	1	0	0	0
OP7_REGISTER (reg_...	0	1	0	0	0
P_REGISTER (reg_mu...	0	48	0	0	0

Implementation Timing Report:

Design Timing Summary

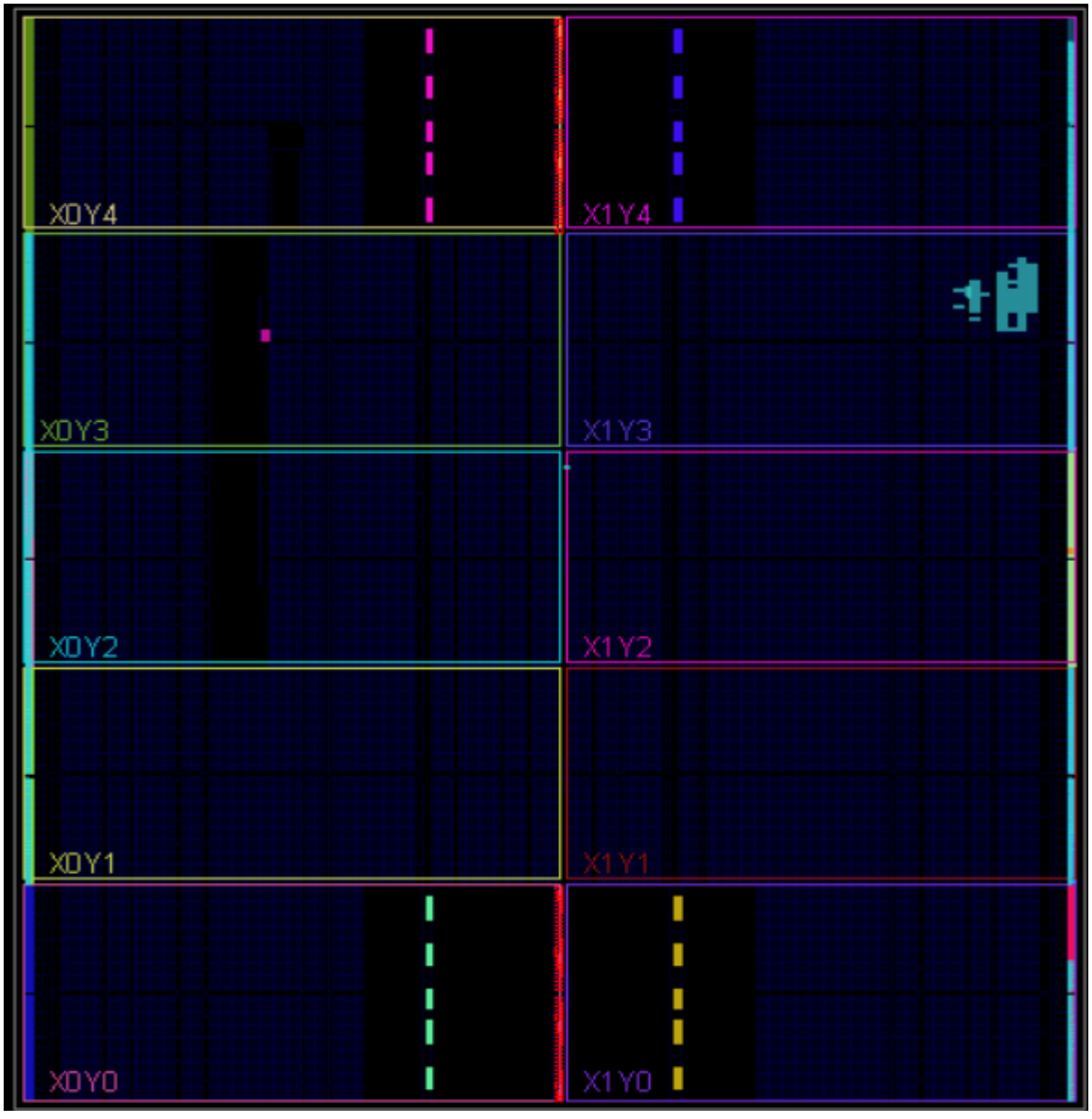
Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	3.815 ns	Worst Hold Slack (WHS):	0.265 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	106	Total Number of Endpoints:	106	Total Number of Endpoints:	181

All user specified timing constraints are met.

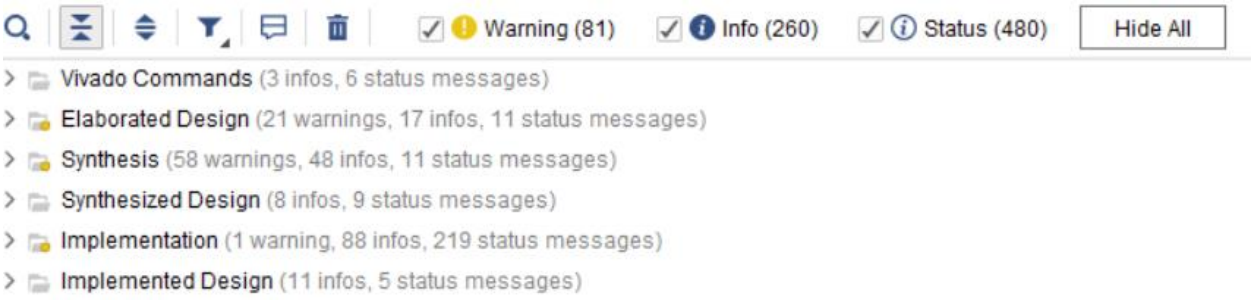
Implementation Utilization Report:

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 DSP48A1	229	179	103	229	51	1	327	1
A1_REGISTER (reg_m...	0	18	7	0	0	0	0	0
B1_REGISTER (reg_m...	0	36	12	0	0	0	0	0
C_REGISTER (reg_m...	0	48	17	0	0	0	0	0
CYI_REGISTER (reg_...	1	1	1	1	1	0	0	0
CYO_REGISTER (reg_...	0	2	2	0	0	0	0	0
D_REGISTER (reg_m...	0	18	10	0	0	0	0	0
OP1TO0_REGISTER (r...	97	2	34	97	0	0	0	0
OP3TO2_REGISTER (r...	96	2	31	96	0	0	0	0
OP4_REGISTER (reg_...	18	1	8	18	0	0	0	0
OP5_REGISTER (reg_...	0	1	1	0	0	0	0	0
OP6_REGISTER (reg_...	17	1	6	17	0	0	0	0
OP7_REGISTER (reg_...	0	1	1	0	0	0	0	0
P_REGISTER (reg_mu...	0	48	12	0	0	0	0	0

Device:



Messages (No Critical Warnings or Errors):



Constraint File:

```
## Clock signal
set_property -dict {PACKAGE_PIN W5 IOSTANDARD LVCMOS33} [get_ports CLK]
create_clock -period 10.000 -name sys_clk_pin -waveform {0.000 5.000} -add [get_ports {CLK}]
```