

Digital Design using Verilog

Project 1: DSP48A1

Name	Sohaila Ahmed Mohamed
-------------	-----------------------

1. RTL code

```
1  module DSP48A1(CLK,CEA,CEB,CEM,CEP,CEC,CED,CECARRYIN,CEOPMODE,  
2      RSTA,RSTB,RSTC,RSTCARRYIN,RSTD,RSTM,RSTOPMODE,RSTP,  
3      CARRYIN,OPMODE,A,B,D,C,BCIN,PCIN,M,P,CARRYOUT,CARRYOUTF,BCOUT,PCOUT);  
4  
5      //Parameters Definitions  
6      parameter A0REG = 0;  
7      parameter A1REG = 1;  
8      parameter B0REG = 0;  
9      parameter B1REG = 1;  
10  
11     parameter CREG = 1;  
12     parameter DREG = 1;  
13     parameter MREG = 1;  
14     parameter PREG = 1;  
15     parameter CARRYINREG = 1;  
16     parameter CARRYOUTREG = 1;  
17     parameter OPMODEREG = 1;  
18  
19     parameter CARRYINSEL = "OPMODE5";  
20  
21     parameter B_INPUT = "DIRECT";  
22  
23     parameter RSTTYPE = "SYNC";  
24  
25     //Inputs and Outputs Definitions  
26     input CLK,CEA,CEB,CEM,CEP,CEC,CED,CECARRYIN,CEOPMODE;  
27     input RSTA,RSTB,RSTC,RSTCARRYIN,RSTD,RSTM,RSTOPMODE,RSTP;  
28     input CARRYIN;  
29     input [7:0] OPMODE;  
30  
31     input [17:0] A,B,D;  
32     input [47:0] C;  
33  
34     input [17:0] BCIN;  
35     input [47:0] PCIN;  
36  
37     output reg [35:0] M;  
38     output reg [47:0] P;  
39  
40     output reg CARRYOUT,CARRYOUTF;
```

```

41
42  output reg [17:0] BCOUT;
43  output reg [47:0] PCOUT;
44
45  //regs & wires Definitions
46  reg [17:0] A0reg,B0reg,Dreg,B1reg,A1reg,A0mux,B0mux,Dmux,B1mux,A1mux;
47  reg [47:0] Creg,Cmux;
48  reg [7:0] OPMODEreg,OPMODEmux;
49  reg [17:0] preAS,preASmux;
50  reg [36:0] mul_out,Mreg,Mmux;
51  reg CImux,CYIreg,CYImux;
52  reg CY0,CY0reg;
53  reg [47:0] Xmux,Zmux;
54  reg [47:0] POUT;
55  reg [47:0] postAS;
56  reg [47:0] Preg;
57
58  /*RTL Code*/
59
60  generate
61      //Sync RST/////
62      if (RSTTYPE == "SYNC") begin
63          always @(posedge CLK) begin
64              if (RSTA) begin
65                  A0reg <= 0;
66              end
67              else if (CEA) begin
68                  A0reg <= A;
69              end
70              if (RSTB) begin
71                  B0reg <= 0;
72              end
73              else if (CEB) begin
74                  case(B_INPUT)
75                      "DIRECT": B0reg <= B;
76                      "CASCADE": B0reg <= BCIN;
77                      default: B0reg <= B;
78                  endcase
79              end
80              if (RSTC) begin

```

```

81         Creg <= 0;
82     end
83     else if (CEC) begin
84         Creg <= C;
85     end
86     if (RSTD) begin
87         Dreg <= 0;
88     end
89     else if (CED) begin
90         Dreg <= D;
91     end
92     if (RSTOPMODE) begin
93         OPMODEreg <= 0;
94     end
95     else if (CEOPMODE) begin
96         OPMODEreg <= OPMODE;
97     end
98 end
99
100 always @(*) begin
101     if (A0REG) begin
102         A0mux = A0reg;
103     end
104     else begin
105         A0mux = A;
106     end
107     if (B0REG) begin
108         B0mux = B0reg;
109     end
110     else begin
111         case(B_INPUT)
112             "DIRECT": B0mux = B;
113             "CASCADE": B0mux = BCIN;
114             default: B0mux = B;
115         endcase
116     end
117     if (CREG) begin
118         Cmux = Creg;
119     end
120     else begin

```

```

121         Cmux = C;
122     end
123     if (DREG) begin
124         Dmux = Dreg;
125     end
126     else begin
127         Dmux = D;
128     end
129     if (OPMODEREG) begin
130         OPMODEmux = OPMODEreg;
131     end
132     else begin
133         OPMODEmux = OPMODE;
134     end
135     if (OPMODEmux[6]) begin
136         preAS = Dmux - B0mux;
137     end
138     else begin
139         preAS = Dmux + B0mux;
140     end
141     if (OPMODEmux[4]) begin
142         preASmux = preAS;
143     end
144     else begin
145         preASmux = B0mux;
146     end
147 end
148
149 always @(posedge CLK) begin
150     if (RSTB) begin
151         B1reg <= 0;
152     end
153     else if (CEB) begin
154         B1reg <= preASmux;
155     end
156     if (RSTA) begin
157         A1reg <= 0;
158     end
159     else if (CEA) begin
160         A1reg <= A0mux;

```

```

161         end
162     end
163
164     always @(*) begin
165         if (B1REG) begin
166             B1mux = B1reg;
167         end
168         else begin
169             B1mux = preASmux;
170         end
171         BCOUT = B1mux;
172         if (A1REG) begin
173             A1mux = A1reg;
174         end
175         else begin
176             A1mux = A0mux;
177         end
178         mul_out = B1mux * A1mux;
179     end
180
181     always @(posedge CLK) begin
182         if (RSTM) begin
183             Mreg <= 0;
184         end
185         else if (CEM) begin
186             Mreg <= mul_out;
187         end
188     end
189
190     always @(*) begin
191         if (MREG) begin
192             Mmux = Mreg;
193         end
194         else begin
195             Mmux = mul_out;
196         end
197         M = Mmux;
198     end
199
200     always @(*) begin

```

```

201         case(CARRYINSEL)
202             "OPMODE5": CImux = OPMODEmux[5];
203             "CARRYIN": CImux = CARRYIN;
204             default: CImux = OPMODEmux[5];
205         endcase
206     end
207
208     always @(posedge CLK) begin
209         if (RSTCARRYIN) begin
210             CYIreg <= 0;
211         end
212         else if (CECARRYIN) begin
213             CYIreg <= CImux;
214         end
215     end
216
217     always @(*) begin
218         if (CARRYINREG) begin
219             CYImux = CYIreg;
220         end
221         else begin
222             CYImux = CImux;
223         end
224     end
225
226     always @(*) begin
227         POUT = PCOUT;
228         case(OPMODEmux[1:0])
229             2'b00: Xmux = 48'b0;
230             2'b01: Xmux = Mmux;
231             2'b10: Xmux = POUT;
232             2'b11: Xmux = {Dmux[11:0], A1mux, B1mux};
233         endcase
234
235         case(OPMODEmux[3:2])
236             2'b00: Zmux = 48'b0;
237             2'b01: Zmux = PCIN;
238             2'b10: Zmux = POUT;
239             2'b11: Zmux = Cmux;
240         endcase

```

```

241
242         if (OPMODEmux[7]) begin
243             {CY0,postAS} = Zmux - Xmux - CYImux;
244         end
245         else begin
246             {CY0,postAS} = Zmux + Xmux + CYImux;
247         end
248     end
249
250     always @(posedge CLK) begin
251         if (RSTCARRYIN) begin
252             CYOreg <= 0;
253         end
254         else if (CECARRYIN) begin
255             CYOreg <= CY0;
256         end
257     end
258
259     always @(*) begin
260         if (CARRYOUTREG) begin
261             CARRYOUT = CYOreg;
262             CARRYOUTF = CYOreg;
263         end
264         else begin
265             CARRYOUT = CY0;
266             CARRYOUTF = CY0;
267         end
268     end
269
270     always @(posedge CLK) begin
271         if (RSTP) begin
272             Preg <= 0;
273         end
274         else if (CEP) begin
275             Preg <= postAS;
276         end
277     end
278
279     always @(*) begin
280         if (PREG) begin

```



```

281         P = Preg;
282         PCOUT = Preg;
283     end
284     else begin
285         P = postAS;
286         PCOUT = postAS;
287     end
288 end
289 end
290
291 //Async RST//
292 else if (RSTTYPE == "ASYNC") begin
293     always @(posedge CLK or posedge RSTA or posedge RSTB or posedge RSTC or posedge RSTD or posedge RSTOPMODE) begin
294         if (RSTA) begin
295             A0reg <= 0;
296         end
297         else if (CEA) begin
298             A0reg <= A;
299         end
300         if (RSTB) begin
301             B0reg <= 0;
302         end
303         else if (CEB) begin
304             case(B_INPUT)
305                 "DIRECT": B0reg <= B;
306                 "CASCADE": B0reg <= BCIN;
307                 default: B0reg <= B;
308             endcase
309         end
310         if (RSTC) begin
311             Creg <= 0;
312         end
313         else if (CEC) begin
314             Creg <= C;
315         end
316         if (RSTD) begin
317             Dreg <= 0;
318         end
319         else if (CED) begin
320             Dreg <= D;

```

```

321     end
322     if (RSTOPMODE) begin
323         OPMODEreg <= 0;
324     end
325     else if (CEOPMODE) begin
326         OPMODEreg <= OPMODE;
327     end
328 end
329
330 always @(*) begin
331     if (A0REG) begin
332         A0mux = A0reg;
333     end
334     else begin
335         A0mux = A;
336     end
337     if (B0REG) begin
338         B0mux = B0reg;
339     end
340     else begin
341         case(B_INPUT)
342             "DIRECT": B0mux = B;
343             "CASCADE": B0mux = BCIN;
344             default: B0mux = B;
345         endcase
346     end
347     if (CREG) begin
348         Cmux = Creg;
349     end
350     else begin
351         Cmux = C;
352     end
353     if (DREG) begin
354         Dmux = Dreg;
355     end
356     else begin
357         Dmux = D;
358     end
359     if (OPMODEREG) begin
360         OPMODEmux = OPMODEreg;

```

```

361         end
362     else begin
363         OPMODEmux = OPMODE;
364     end
365     if (OPMODEmux[6]) begin
366         preAS = Dmux - B0mux;
367     end
368     else begin
369         preAS = Dmux + B0mux;
370     end
371     if (OPMODEmux[4]) begin
372         preASmux = preAS;
373     end
374     else begin
375         preASmux = B0mux;
376     end
377 end
378
379 always @(posedge CLK or posedge RSTB or posedge RSTA) begin
380     if (RSTB) begin
381         B1reg <= 0;
382     end
383     else if (CEB) begin
384         B1reg <= preASmux;
385     end
386     if (RSTA) begin
387         Alreg <= 0;
388     end
389     else if (CEA) begin
390         Alreg <= A0mux;
391     end
392 end
393
394 always @(*) begin
395     if (B1REG) begin
396         B1mux = B1reg;
397     end
398     else begin
399         B1mux = preASmux;
400     end

```

```

401      BCOUT = B1mux;
402 ▼      if (A1REG) begin
403          A1mux = A1reg;
404      end
405 ▼      else begin
406          A1mux = A0mux;
407      end
408      mul_out = B1mux * A1mux;
409  end
410
411 ▼      always @(posedge CLK or posedge RSTM) begin
412 ▼          if (RSTM) begin
413              Mreg <= 0;
414          end
415 ▼          else if (CEM) begin
416              Mreg <= mul_out;
417          end
418      end
419
420 ▼      always @(*) begin
421 ▼          if (MREG) begin
422              Mmux = Mreg;
423          end
424 ▼          else begin
425              Mmux = mul_out;
426          end
427          M = Mmux;
428      end
429
430 ▼      always @(*) begin
431 ▼          case(CARRYINSEL)
432              "OPMODE5": CImux = OPMODEmux[5];
433              "CARRYIN": CImux = CARRYIN;
434              default: CImux = OPMODEmux[5];
435          endcase
436      end
437
438 ▼      always @(posedge CLK or posedge RSTCARRYIN) begin
439 ▼          if (RSTCARRYIN) begin
440              CYIreg <= 0;

```

```

441         end
442     ▼     else if (CECARRYIN) begin
443         CYIreg <= CImux;
444     end
445     end
446
447     always @(*) begin
448     ▼         if (CARRYINREG) begin
449             CYImux = CYIreg;
450         end
451     ▼         else begin
452             CYImux = CImux;
453         end
454     end
455
456     always @(*) begin
457         POUT = PCOUT;
458     ▼         case(OPMODEmux[1:0])
459             2'b00: Xmux = 48'b0;
460             2'b01: Xmux = Mmux;
461             2'b10: Xmux = POUT;
462             2'b11: Xmux = {Dmux[11:0],A1mux,B1mux};
463         endcase
464
465     ▼         case(OPMODEmux[3:2])
466             2'b00: Zmux = 48'b0;
467             2'b01: Zmux = PCIN;
468             2'b10: Zmux = POUT;
469             2'b11: Zmux = Cmux;
470         endcase
471
472     ▼         if (OPMODEmux[7]) begin
473             {CY0,postAS} = Zmux - Xmux - CYImux;
474         end
475     ▼         else begin
476             {CY0,postAS} = Zmux + Xmux + CYImux;
477         end
478     end
479
480     ▼     always @(posedge CLK or posedge RSTCARRYIN) begin

```

```
481         if (RSTCARRYIN) begin
482             CY0reg <= 0;
483         end
484         else if (CECARRYIN) begin
485             CY0reg <= CY0;
486         end
487     end
488     always @(*) begin
489         if (CARRYOUTREG) begin
490             CARRYOUT = CY0reg;
491             CARRYOUTF = CY0reg;
492         end
493         else begin
494             CARRYOUT = CY0;
495             CARRYOUTF = CY0;
496         end
497     end
498
499     always @(posedge CLK or posedge RSTP) begin
500         if (RSTP) begin
501             Preg <= 0;
502         end
503         else if (CEP) begin
504             Preg <= postAS;
505         end
506     end
507     always @(*) begin
508         if (PREG) begin
509             P = Preg;
510             PCOUT = Preg;
511         end
512         else begin
513             P = postAS;
514             PCOUT = postAS;
515         end
516     end
517 end
518 endgenerate
519
520 endmodule
```

2. Testbench code

```
1  module DSP48A1_TB();
2
3  reg CLK,CEA,CEB,CEM,CEP,CEC,CED,CECARRYIN,CEOPMODE;
4  reg RSTA,RSTB,RSTC,RSTCARRYIN,RSTD,RSTM,RSTOPMODE,RSTP;
5  reg CARRYIN;
6  reg [7:0] OPMODE;
7
8  reg [17:0] A,B,D;
9  reg [47:0] C;
10
11 reg [17:0] BCIN;
12 reg [47:0] PCIN;
13
14 wire [35:0] M;
15 wire [47:0] P;
16
17 wire CARRYOUT,CARRYOUTF;
18
19 wire [17:0] BCOUT;
20 wire [47:0] PCOUT;
21
22 //DUT instantiation
23 DSP48A1 DUT(CLK,CEA,CEB,CEM,CEP,CEC,CED,CECARRYIN,CEOPMODE,
24             RSTA,RSTB,RSTC,RSTCARRYIN,RSTD,RSTM,RSTOPMODE,RSTP,
25             CARRYIN,OPMODE,A,B,D,C,BCIN,PCIN,M,P,CARRYOUT,CARRYOUTF,BCOUT,PCOUT);
26
27 //clk generation
28 initial begin
29     CLK = 0;
30     forever
31         #1 CLK = ~CLK;
32 end
33
34 //test
35 initial begin
36     //2.1. Reset Functionality
37     RSTA = 1;RSTB = 1;RSTC = 1;
38     RSTCARRYIN = 1;RSTD = 1;
39     RSTM = 1;RSTOPMODE = 1;RSTP = 1;
40
41     CLK = $random;CEA = $random;
42     CEB = $random;CEM = $random;
43     CEP = $random;CEC = $random;
44     CED = $random;CECARRYIN = $random;
45     CEOPMODE = $random;
46
47     CARRYIN = $random;OPMODE = $random;
48
49     A = $random;B = $random;
50     C = $random;D = $random;
51
52     BCIN = $random;PCIN = $random;
53
54     @(negedge CLK);
```

```

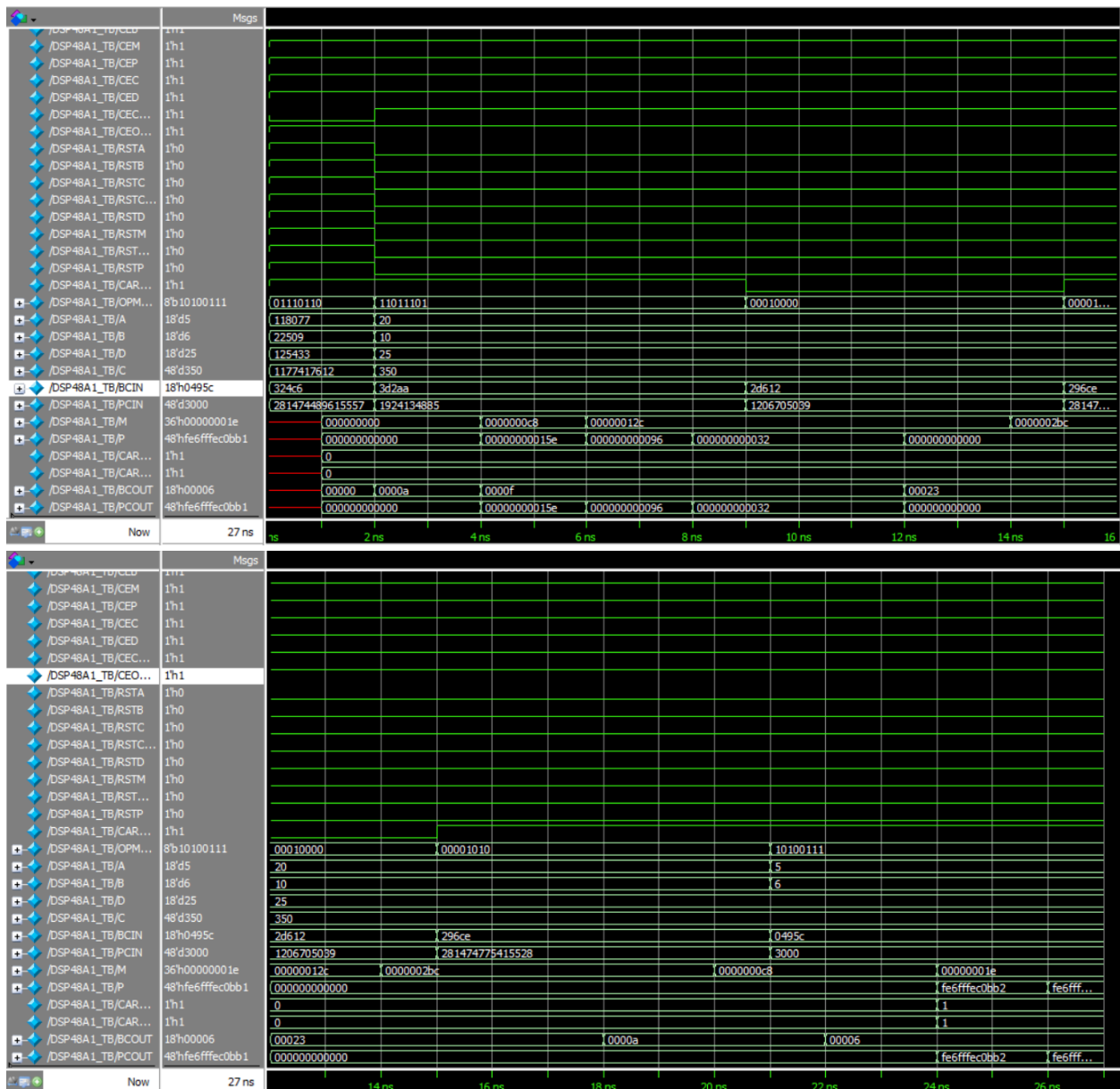
55
56     if ((M != 0) || (P != 0) || (CARRYOUT != 0) || (CARRYOUTF != 0) || (PCOUT != 0) || (BCOUT != 0)) begin
57         $display("error");
58         $stop;
59     end
60
61     RSTA = 0;RSTB = 0;RSTC = 0;
62     RSTCARRYIN = 0;RSTD = 0;
63     RSTM = 0;RSTOPMODE = 0;RSTP = 0;
64
65     CLK = 1;CEA = 1;
66     CEB = 1;CEM = 1;
67     CEP = 1;CEC = 1;
68     CED = 1;CECARRYIN = 1;
69     CEOPMODE = 1;
70
71     //2.2. DSP path 1
72     A = 18'd20;B = 18'd10;
73     C = 48'd350;D = 18'd25;
74
75     OPMODE = 8'b11011101;
76
77     BCIN = $random;PCIN = $random;CARRYIN = $random;
78
79     repeat(4) @(negedge CLK);
80
81     if ((M != 36'h12c) || (P != 48'h32) || (CARRYOUT != 0) || (CARRYOUTF != 0) || (PCOUT != 48'h32) || (BCOUT != 18'hf)) begin
82         $display("error");
83         $stop;
84     end
85
86     //2.3. DSP path 2
87     OPMODE = 8'b00010000;
88
89     BCIN = $random;PCIN = $random;CARRYIN = $random;
90
91     repeat(3) @(negedge CLK);
92
93     if ((M != 36'h2bc) || (P != 48'h0) || (CARRYOUT != 0) || (CARRYOUTF != 0) || (PCOUT != 48'h0) || (BCOUT != 18'h23)) begin
94         $display("error");
95         $stop;
96     end
97
98     //2.4. DSP path 3
99     OPMODE = 8'b00001010;
100
101     BCIN = $random;PCIN = $random;CARRYIN = $random;
102
103     repeat(3) @(negedge CLK);
104
105     if ((M != 36'hc8) || (P != 48'h0) || (CARRYOUT != 0) || (CARRYOUTF != 0) || (PCOUT != 48'h0) || (BCOUT != 18'ha)) begin
106         $display("error");
107         $stop;
108     end
109
110     //2.5. DSP path 4
111     A = 18'd5;B = 18'd6;
112
113     OPMODE = 8'b10100111;PCIN = 48'd3000;
114
115     BCIN = $random;CARRYIN = $random;
116
117     repeat(3) @(negedge CLK);
118
119     if ((M != 36'h1e) || (P != 48'hfe6fffec0bb1) || (CARRYOUT != 1) || (CARRYOUTF != 1) || (PCOUT != 48'hfe6fffec0bb1) || (BCOUT != 18'h6)) be
120         $display("error");
121         $stop;
122     end
123     $stop;
124 end
125
126 endmodule

```

3. Do file

```
1 vlib work
2 vlog DSP48A1.v DSP48A1_TB.v
3 vsim -voptargs=+acc work.DSP48A1_TB
4 add wave *
5 run -all
6 #quit -sim
```

4. QuestaSim Snippets

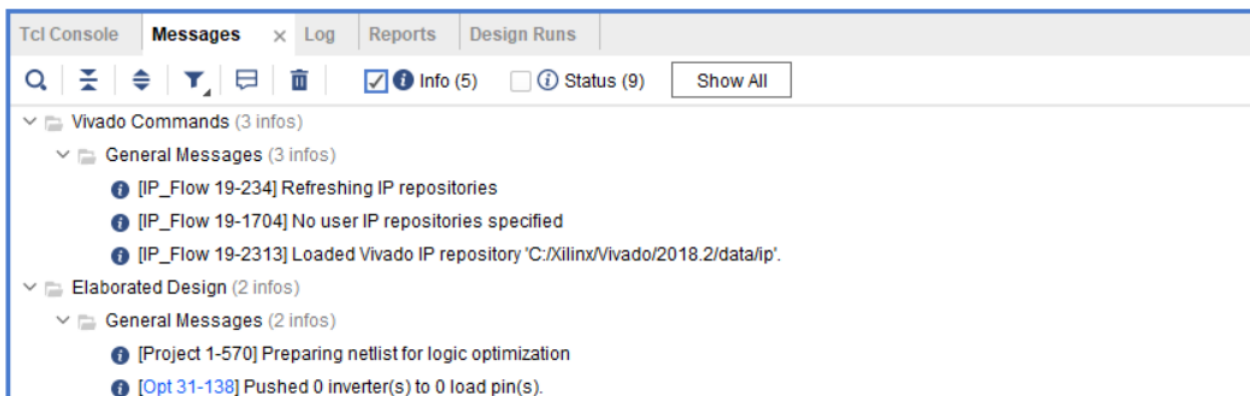


5. Constraint File

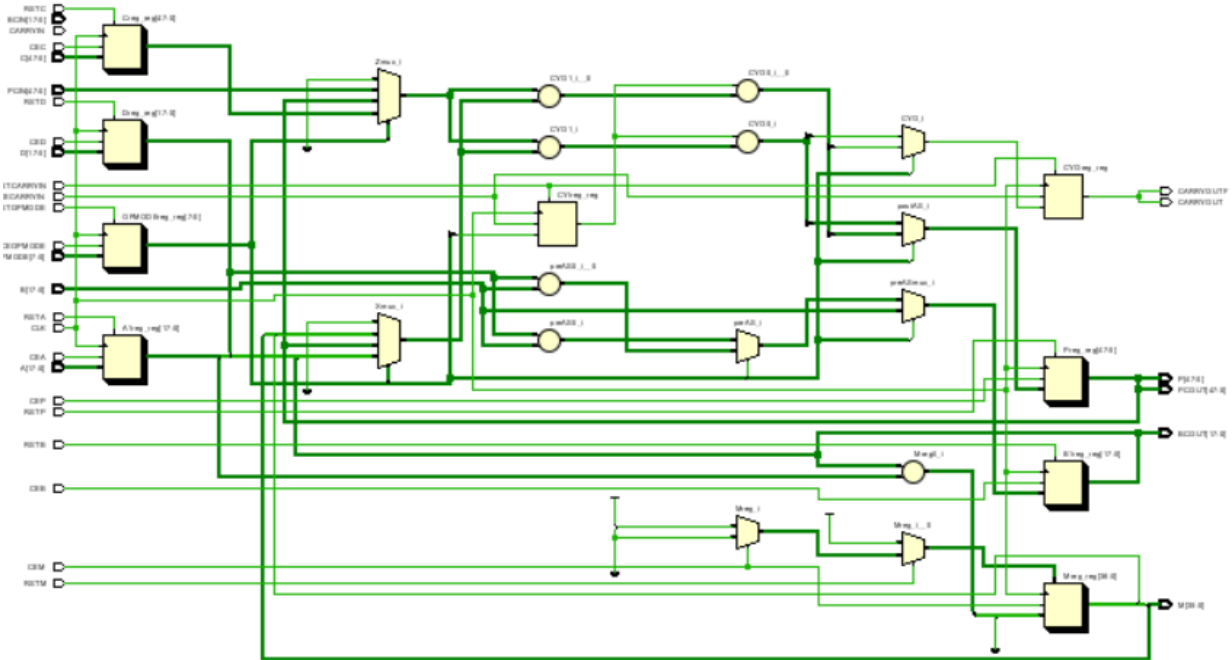
```
1  ## This file is a general .xdc for the Basys3 rev B board
2  ## To use it in a project:
3  ## - uncomment the lines corresponding to used pins
4  ## - rename the used ports (in each line, after get_ports) according to the top level signal names in the project
5
6  # Clock signal
7  set_property -dict { PACKAGE_PIN W5    IOSTANDARD LVCMOS33 } [get_ports clk]
8  create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clk]
9
10
11 ## Switches
12 #set_property -dict { PACKAGE_PIN V17    IOSTANDARD LVCMOS33 } [get_ports {sw[0]}]
13 #set_property -dict { PACKAGE_PIN V16    IOSTANDARD LVCMOS33 } [get_ports {sw[1]}]
14 #set_property -dict { PACKAGE_PIN W16    IOSTANDARD LVCMOS33 } [get_ports {sw[2]}]
15 #set_property -dict { PACKAGE_PIN W17    IOSTANDARD LVCMOS33 } [get_ports {sw[3]}]
16 #set_property -dict { PACKAGE_PIN W15    IOSTANDARD LVCMOS33 } [get_ports {sw[4]}]
17 #set_property -dict { PACKAGE_PIN V15    IOSTANDARD LVCMOS33 } [get_ports {sw[5]}]
18 #set_property -dict { PACKAGE_PIN W14    IOSTANDARD LVCMOS33 } [get_ports {sw[6]}]
19 #set_property -dict { PACKAGE_PIN W13    IOSTANDARD LVCMOS33 } [get_ports {sw[7]}]
20 #set_property -dict { PACKAGE_PIN V2     IOSTANDARD LVCMOS33 } [get_ports {sw[8]}]
21 #set_property -dict { PACKAGE_PIN T3     IOSTANDARD LVCMOS33 } [get_ports {sw[9]}]
22 #set_property -dict { PACKAGE_PIN T2     IOSTANDARD LVCMOS33 } [get_ports {sw[10]}]
23 #set_property -dict { PACKAGE_PIN R3     IOSTANDARD LVCMOS33 } [get_ports {sw[11]}]
24 #set_property -dict { PACKAGE_PIN W2     IOSTANDARD LVCMOS33 } [get_ports {sw[12]}]
25 #set_property -dict { PACKAGE_PIN U1     IOSTANDARD LVCMOS33 } [get_ports {sw[13]}]
26 #set_property -dict { PACKAGE_PIN T1     IOSTANDARD LVCMOS33 } [get_ports {sw[14]}]
27 #set_property -dict { PACKAGE_PIN R2     IOSTANDARD LVCMOS33 } [get_ports {sw[15]}]
28
29
30 ## LEDs
31 #set_property -dict { PACKAGE_PIN U16    IOSTANDARD LVCMOS33 } [get_ports {led[0]}]
32 #set_property -dict { PACKAGE_PIN E19    IOSTANDARD LVCMOS33 } [get_ports {led[1]}]
33 #set_property -dict { PACKAGE_PIN U19    IOSTANDARD LVCMOS33 } [get_ports {led[2]}]
34 #set_property -dict { PACKAGE_PIN V19    IOSTANDARD LVCMOS33 } [get_ports {led[3]}]
35 #set_property -dict { PACKAGE_PIN W18    IOSTANDARD LVCMOS33 } [get_ports {led[4]}]
36 #set_property -dict { PACKAGE_PIN U15    IOSTANDARD LVCMOS33 } [get_ports {led[5]}]
```

6. Elaboration (“Messages” tab & Schematic snippets)

- **Messages**



- **Schematic**



7. Synthesis (“Messages” tab, Utilization report, timing report & Schematic snippets)

- **Messages**

Tcl Console
Messages
Log
Reports
Design Runs
Debug

☒ Warning (46)
☒ Info (31)
☐ Status (19)
Show All

▼ Synthesis (46 warnings)

- [Synth 8-2490] overwriting previous definition of module DSP48A1 [\[DSP48A1.v:1\]](#)
- > [Synth 8-6014] Unused sequential element A0reg_reg was removed. [\[DSP48A1.v:65\]](#) (3 more like this)
- [Synth 8-3936] Found unconnected internal register 'Dmux_reg' and it is trimmed from '18' to '12' bits. [\[DSP48A1.v:124\]](#)
- > [Synth 8-3331] design DSP48A1 has unconnected port CARRYIN (37 more like this)
 - [Synth 8-3332] Sequential element (Mreg_reg[36]) is unused and will be removed from module DSP48A1.
 - [Constraints 18-5210] No constraint will be written out.

- **Utilization report**

The screenshot shows the 'Utilization' tab in the Xilinx Vivado IDE. The 'Hierarchy' sub-tab is selected, displaying a table of resource utilization. The table has columns for 'Name', 'Slice LUTs (134600)', 'Slice Registers (269200)', 'DSPs (740)', 'Bonded IOB (500)', and 'BUFGCTRL (32)'. The 'DSP48A1' resource is highlighted in orange, showing a value of 208 in the 'Slice LUTs' column. The left sidebar shows the project hierarchy with 'utilization_1' selected.

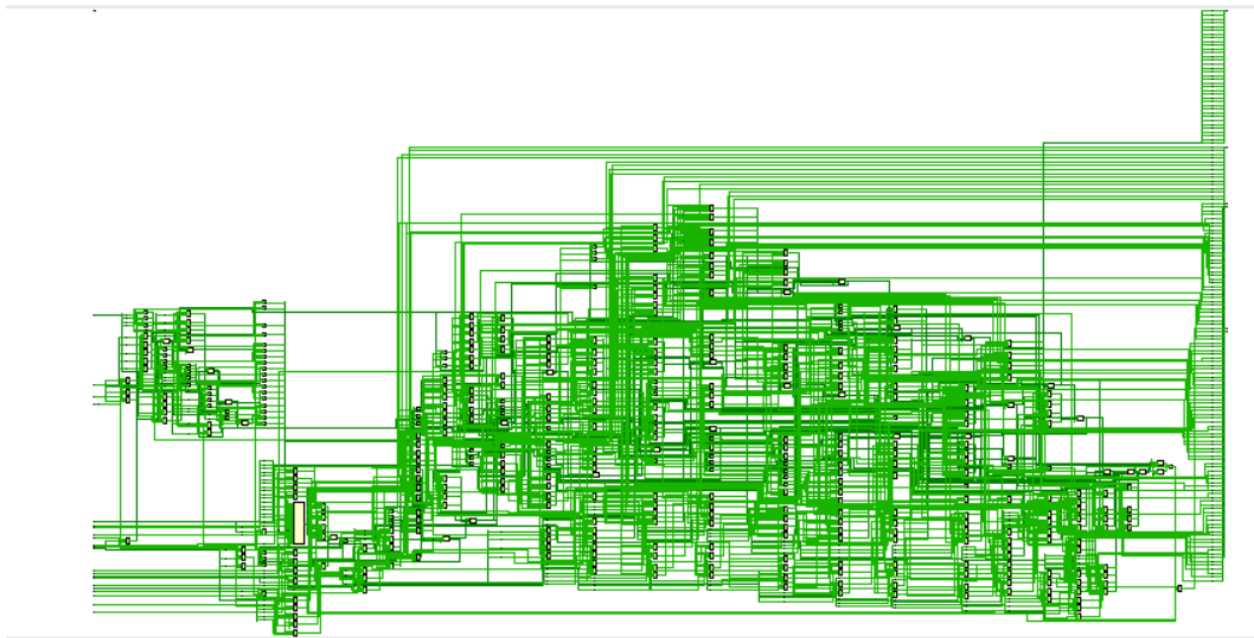
Name	Slice LUTs (134600)	Slice Registers (269200)	DSPs (740)	Bonded IOB (500)	BUFGCTRL (32)
DSP48A1	208	160	1	327	1

- **Timing report**

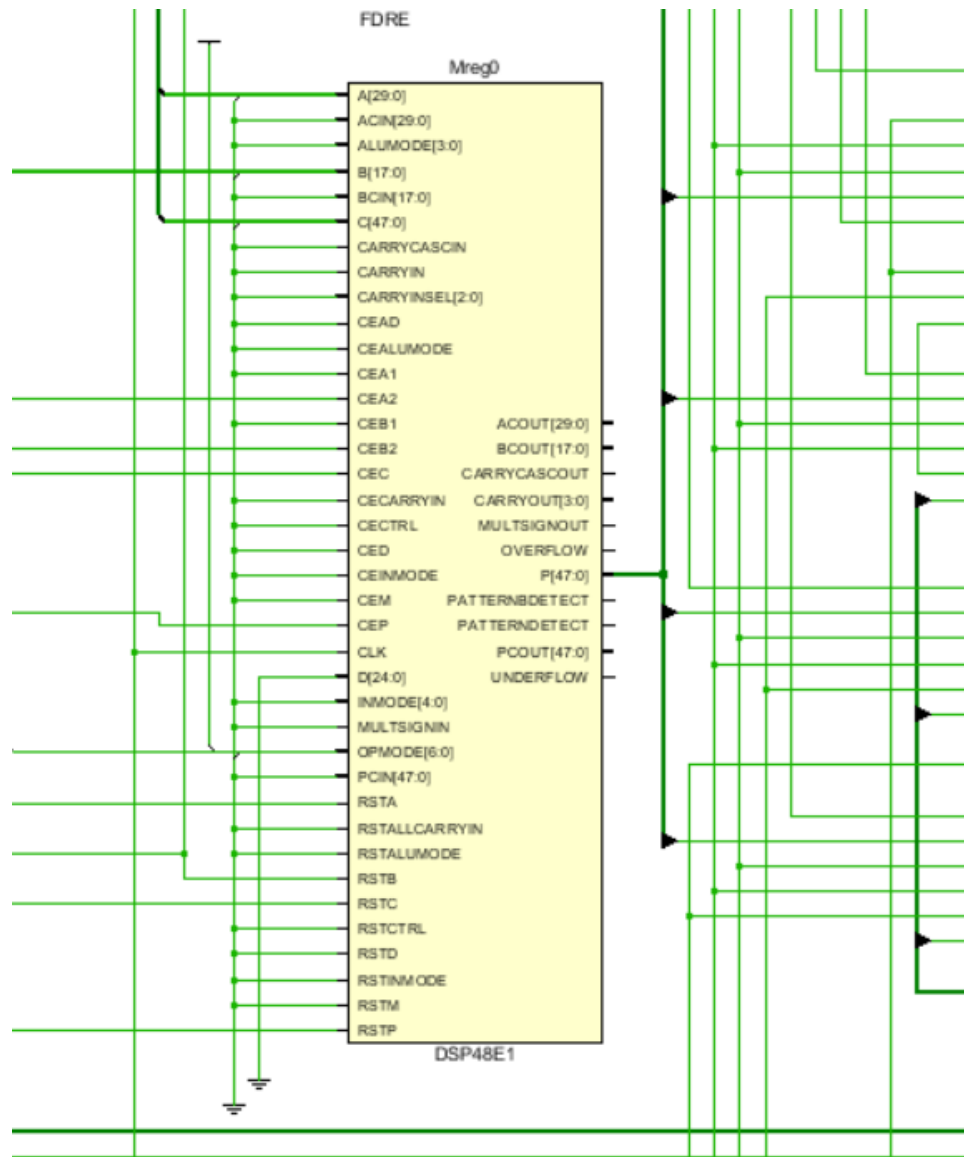
Setup	Hold	Pulse Width
Worst Negative Slack (WNS): 5.794 ns	Worst Hold Slack (WHS): 0.182 ns	Worst Pulse Width Slack (WPWS): 4.500 ns
Total Negative Slack (TNS): 0.000 ns	Total Hold Slack (THS): 0.000 ns	Total Pulse Width Negative Slack (TPWS): 0.000 ns
Number of Failing Endpoints: 0	Number of Failing Endpoints: 0	Number of Failing Endpoints: 0
Total Number of Endpoints: 106	Total Number of Endpoints: 106	Total Number of Endpoints: 162

All user specified timing constraints are met.

- **Schematic**

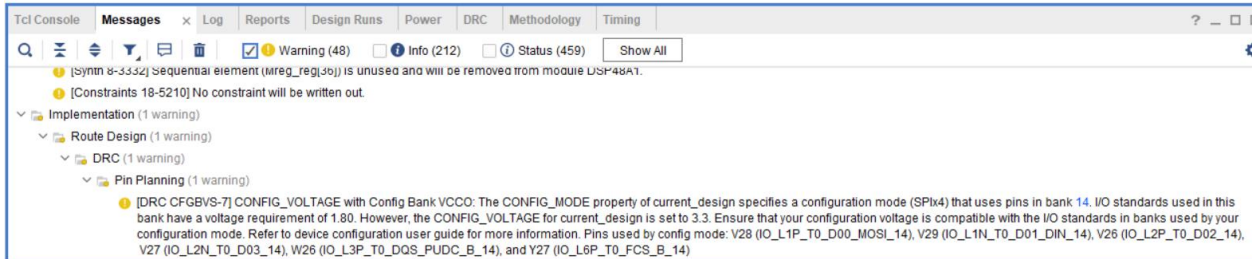


- DSP block



8. Implementation (“Messages” tab, Utilization report, timing report & device snippets)

- Messages



- Utilization report

The Utilization report shows the resource usage for the DSP48A1 device. The table lists various resources and their usage counts.

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)
DSP48A1	207	179	92	207	27	1	327	1

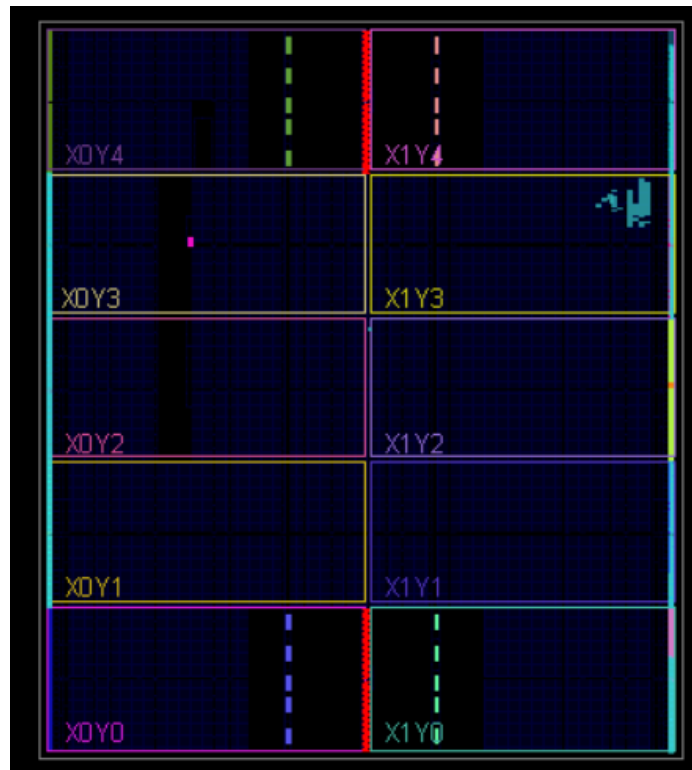
- Timing report

The Timing report shows the Design Timing Summary. The table lists various timing metrics and their values.

Setup	Hold	Pulse Width
Worst Negative Slack (WNS): 4.653 ns	Worst Hold Slack (WHS): 0.278 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: 125	Total Number of Endpoints: 125	Total Number of Endpoints: 181

All user specified timing constraints are met.

- Device



9. Linting (snippets showing no errors)

Transcript

```
#
# Top level modules:
#
#     DSP48A1
#
# End time: 05:18:28 on Aug 02,2025, Elapsed time: 0:00:00
#
# Errors: 0, Warnings: 0
#
```

Visualizer>