Momen Mostafa Mohamed Elzaghawy

Lab1

Q3:64 bit ladner fischer parallel prefix adder

1. RTL the Verilog/VHDL code implements the assigned question and comment your design well upload all the code files and attach the link to the report

```
module carry_op_2bit(
output[1:0]go,po,input [1:0]g,p
);
 assign go[1]=g[1]|(g[0]&p[1]);
 assign po[1]=p[0]&p[1];
 assign go[0]=g[0];
 assign po[0]=p[0];
 endmodule
```

```
module fourbit(
output [3:0]go,po,input [3:0]g,p
);
 wire [3:0]gw,pw;

 carry_op_2bit C0(gw[1:0],pw[1:0],g[1:0],p[1:0]);
 carry_op_2bit C1(gw[3:2],pw[3:2],g[3:2],p[3:2]);
 assign go[1:0]=gw[1:0];
 assign po[1:0]=pw[1:0];
 assign go[2]=gw[2]|(gw[1]&pw[2]);
 assign go[3]=gw[3]|(gw[2]&pw[3]);
 assign po[2]=pw[2]&pw[1];
 assign po[2]=pw[2]&pw[1];
 endmodule
```

```
module eightbit (
output [7:0]go,po,input[7:0]g,p
);
 wire [7:0]gw,pw;
 fourbit F0(gw[3:0],pw[3:0],g[3:0],p[3:0]);
 fourbit F1(gw[7:4],pw[7:4],g[7:4],p[7:4]);
 assign go[3:0]=g[3:0];
 assign po[3:0]=p[3:0];
 assign go[4]=gw[4]|(gw[3]&pw[4]);
 assign go[5]=gw[5]|(gw[3]&pw[5]);
 assign go[6]=gw[6]|(gw[3]&pw[6]);
 assign go[7]=gw[7]|(gw[3]&pw[7]);
 assign po[4]=pw[4]&pw[3];
 assign po[5]=pw[5]&pw[3];
 assign po[6]=pw[6]&pw[3];
 assign po[7]=pw[7]&pw[3];
 endmodule
```

```
module sixteenbit(
output [15:0]go,po,input [15:0]g,p
);
 wire [15:0]gw,pw;
 eightbit E0(gw[7:0],pw[7:0],g[7:0],p[7:0]);
 eightbit E1(gw[15:8],pw[15:8],g[15:8],p[15:8]);
 assign go[7:0]=gw[7:0];
 assign po[7:0]=pw[7:0];
 assign go[8]=gw[8]|(gw[7]&pw[8]);
 assign go[9]=gw[9]|(gw[7]&pw[9]);
 assign go[10]=gw[10]|(gw[7]&pw[10]);
 assign go[11]=gw[11]|(gw[7]&pw[11]);
 assign go[12]=gw[12]|(gw[7]&pw[12]);
 assign go[13]=gw[13]|(gw[7]&pw[13]);
 assign go[14]=gw[14]|(gw[7]&pw[14]);
 assign go[15]=gw[15]|(gw[7]&pw[15]);
 assign po[8]=pw[8]&pw[7];
 assign po[9]=pw[9]&pw[7];
 assign po[10]=pw[10]&pw[7];
 assign po[11]=pw[11]&pw[7];
 assign po[12]=pw[12]&pw[7];
 assign po[13]=pw[13]&pw[7];
 assign po[14]=pw[14]&pw[7];
 assign po[15]=pw[15]&pw[7];
 endmodule
```

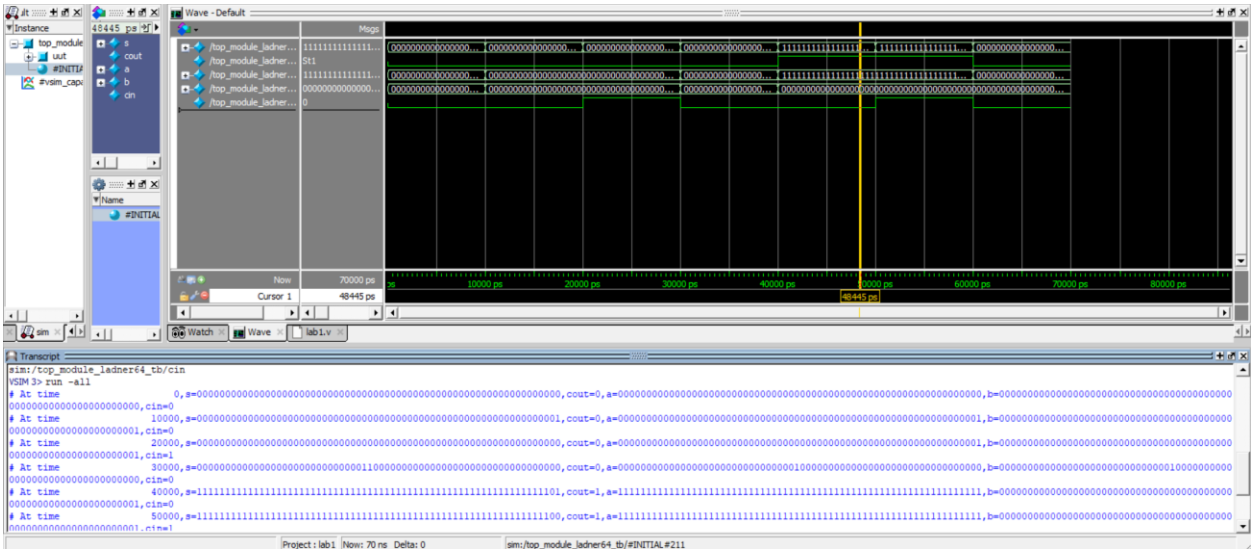
```
module thirtytwobit(
output [31:0]go,po,input [31:0]g,p
);
wire [31:0]gw,pw;
sixteenbit L0(gw[15:0],pw[15:0],g[15:0],p[15:0]);
sixteenbit L1(gw[31:16],pw[31:16],g[31:16],p[31:16]);
assign go[15:0]=gw[15:0];
assign po[15:0]=pw[15:0];
assign go[16]=gw[16]|(gw[15]&pw[16]);
assign go[17]=gw[17]|(gw[15]&pw[17]);
assign go[18]=gw[18]|(gw[15]&pw[18]);
assign go[19]=gw[19]|(gw[15]&pw[19]);
assign go[20]=gw[20]|(gw[15]&pw[20]);
assign go[21]=gw[21]|(gw[15]&pw[21]);
assign go[22]=gw[22]|(gw[15]&pw[22]);
assign go[23]=gw[23]|(gw[15]&pw[23]);
assign go[24]=gw[24]|(gw[15]&pw[24]);
assign go[25]=gw[25]|(gw[15]&pw[25]);
assign go[26]=gw[26]|(gw[15]&pw[26]);
assign go[27]=gw[27]|(gw[15]&pw[27]);
assign go[28]=gw[28]|(gw[15]&pw[28]);
assign go[29]=gw[29]|(gw[15]&pw[29]);
assign go[30]=gw[30]|(gw[15]&pw[30]);
assign go[31]=gw[31]|(gw[15]&pw[31]);
assign po[16]=pw[16]&pw[15];
assign po[17]=pw[17]&pw[15];
assign po[18]=pw[18]&pw[15];
assign po[19]=pw[19]&pw[15];
assign po[20]=pw[20]&pw[15];
assign po[21]=pw[21]&pw[15];
assign po[22]=pw[22]&pw[15];
assign po[23]=pw[23]&pw[15];
assign po[24]=pw[24]&pw[15];
assign po[25]=pw[25]&pw[15];
assign po[26]=pw[26]&pw[15];
assign po[27]=pw[27]&pw[15];
assign po[28]=pw[28]&pw[15];
assign po[29]=pw[29]&pw[15];
assign po[30]=pw[30]&pw[15];
assign po[31]=pw[31]&pw[15];
endmodule
```

```
module sixtyfourbit(
output [63:0]go,po,input [63:0]g,p
);
wire [63:0]gw,pw;
thirtytwobit M0(gw[31:0],pw[31:0],g[31:0],p[31:0]);
thirtytwobit M1(gw[63:32],pw[63:32],g[63:32],p[63:32]);
assign go[31:0]=gw[31:0];
assign po[31:0]=pw[31:0];
assign go[32]=gw[32]|(gw[31]&pw[32]);
assign go[33]=gw[33]|(gw[31]&pw[33]);
assign go[34]=gw[34]|(gw[31]&pw[34]);
assign go[35]=gw[35]|(gw[31]&pw[35]);
assign go[36]=gw[36]|(gw[31]&pw[36]);
assign go[37]=gw[37]|(gw[31]&pw[37]);
assign go[38]=gw[38]|(gw[31]&pw[38]);
assign go[39]=gw[39]|(gw[31]&pw[39]);
assign go[40]=gw[40]|(gw[31]&pw[40]);
assign go[41]=gw[41]|(gw[31]&pw[41]);
assign go[42]=gw[42]|(gw[31]&pw[42]);
assign go[43]=gw[43]|(gw[31]&pw[43]);
assign go[44]=gw[44]|(gw[31]&pw[44]);
assign go[45]=gw[45]|(gw[31]&pw[45]);
assign go[46]=gw[46]|(gw[31]&pw[46]);
assign go[47]=gw[47]|(gw[31]&pw[47]);
assign go[48]=gw[48]|(gw[31]&pw[48]);
assign go[49]=gw[49]|(gw[31]&pw[49]);
assign go[50]=gw[50]|(gw[31]&pw[50]);
assign go[51]=gw[51]|(gw[31]&pw[51]);
assign go[52]=gw[52]|(gw[31]&pw[52]);
assign go[53]=gw[53]|(gw[31]&pw[53]);
assign go[54]=gw[54]|(gw[31]&pw[54]);
assign go[55]=gw[55]|(gw[31]&pw[55]);
assign go[56]=gw[56]|(gw[31]&pw[56]);
assign go[57]=gw[57]|(gw[31]&pw[57]);
assign go[58]=gw[58]|(gw[31]&pw[58]);
assign go[59]=gw[59]|(gw[31]&pw[59]);
assign go[60]=gw[60]|(gw[31]&pw[60]);
assign go[61]=gw[61]|(gw[31]&pw[61]);
assign go[62]=gw[62]|(gw[31]&pw[62]);
assign go[63]=gw[63]|(gw[31]&pw[63]);
assign po[32]=pw[32]&pw[31];
assign po[33]=pw[33]&pw[31];
assign po[34]=pw[34]&pw[31];
assign po[35]=pw[35]&pw[31];
assign po[36]=pw[36]&pw[31];
assign po[37]=pw[37]&pw[31];
assign po[38]=pw[38]&pw[31];
assign po[39]=pw[39]&pw[31];
assign po[40]=pw[40]&pw[31];
assign po[41]=pw[41]&pw[31];
assign po[42]=pw[42]&pw[31];
assign po[43]=pw[43]&pw[31];
assign po[44]=pw[44]&pw[31];
assign po[45]=pw[45]&pw[31];
assign po[46]=pw[46]&pw[31];
assign po[47]=pw[47]&pw[31];
assign po[48]=pw[48]&pw[31];
assign po[49]=pw[49]&pw[31];
assign po[50]=pw[50]&pw[31];
assign po[51]=pw[51]&pw[31];
assign po[52]=pw[52]&pw[31];
assign po[53]=pw[53]&pw[31];
assign po[54]=pw[54]&pw[31];
assign po[55]=pw[55]&pw[31];
assign po[56]=pw[56]&pw[31];
assign po[57]=pw[57]&pw[31];
assign po[58]=pw[58]&pw[31];
assign po[59]=pw[59]&pw[31];
assign po[60]=pw[60]&pw[31];
assign po[61]=pw[61]&pw[31];
assign po[62]=pw[62]&pw[31];
assign po[63]=pw[63]&pw[31];
endmodule
```

```
module top_module_ladner64(
output [63:0]s,output cout,input [63:0]a,b,input cin
);
 wire [63:0]g,p;
 wire [63:0]go,po;
 assign g=a^b;
 assign p=a&b;
 sixtyfourbit K0(.go(go),.po(po),.g(g),.p(p));
 assign s[0]=p[0]^cin;
 genvar i;
generate
 for(i=0;i<63;i=i+1)
begin
 assign s[i+1]=p[i+1]^go[i];
end
endgenerate
 assign cout=go[63];
 endmodule
```

2. Simulation modify the Testbench from Lab0 to fit with the new width of I/O and run simulation attach screenshots to show the output or waveform

```
1   `timescale 1ns/1ps
2   module top_module_ladner64_tb;
3   wire [63:0]s;
4   wire cout;
5   reg [63:0]a,b;
6   reg cin;
7   top_module_ladner64 uut(.s(s),.cout(cout),.a(a),.b(b),.cin(cin));
8   initial
9   begin
10  a=0;
11  b=0;
12  cin=0;
13  #10 a=64'h0000000000000001; b=64'h0000000000000001;cin=0;
14  #10 a=64'h0000000000000001; b=64'h0000000000000001;cin=1;
15  #10 a=64'h0000000100000000; b=64'h0000000200000000;cin=0;
16  #10 a=64'hffffffffffffffff; b=64'h0000000000000001;cin=0;
17  #10 a=64'hffffffffffffffff; b=64'h0000000000000001;cin=1;
18  #10 a=64'h0000000000000000; b=64'h0000000000000001;cin=0;
19  #10 $stop;
20   end
21   initial begin
22  $monitor("At time %t,s=%b,cout=%b,a=%b,b=%b,cin=%b",
23  $time,s,cout,a,b,cin);
24  end
25  endmodule
```



Codes in google drive folder

https://drive.google.com/file/d/1GLUhS3Ut-hfv-4YCLQIG7MMkEccEDaWk/view?usp=drive_link