module rand(a,b,y0,y1,y2,y3, y4,y5,y6,y7,y8,y9,y10,y11.y12,y13,y14.y15,y16.y17,y18,y19,y20,y21.y22,y23,y 24,y25,y26,y27,y28,y29,y30):

input [3:0] a,b;

output reg[3:0] yo,y1,y2.y3.y4,y5,y6.y7,y8,y9,y10,y11.y12,y13.y14,y15.y16.y17,y18,y19,y20,y21.y22,y23,y24.y25, y26 y27y28,y29,y30;

assign a=4b1010;

assign b=4b1101;

Always @(\*)

begin

y0=a&&b;

y1-allb;

y2=!a;

y3=a&b;

y4=alb;

y5-~a;

y6-a^b;

y7=a~^b;

y8-&a;

y9=!a;

y10=~&a;

y11=~a;

y12=^a;

y13=~^a;

y14=a<<b;

y15=a>>b;

y16=a>>>b;

y17=a<<<b;

y18=a>b;

y19=(4>=b);

y20=(a<b);

y21=(a<=b);

y22-(a==b);

y23-(al-b);

y24-(a===b);

y25=(a!==b);

y26=a+b;

y27=a-b;

y28-a\*b;

y29=a/b;

y30=a%b;

end

endodule

Rtl code for alu

module alu(input [7:0]a,b, input (3:0]command\_in,

input oe,

output [15:0]d\_out);

parameter reg [15:0]out;

begin

end

ADD = 4'b0000,

SUB 4'b0010,

DEC = 4b0011,

always@(command\_in)

MUL = 4b0100,

DIV = 4'b0101,

SHL = 4'b0110,

SHR =4'b0111, .

AND =4b1000,

OR =4b1001,

INV = 4b1010,

NAND = 4'b1011,

NOR =4b1100.

XOR =4'b1101,

XNOR = 4'b1110,

case(command\_in)

ADD: out=a+b;

INC: out=a+1;

SUB: out=a-b;

DEC: out=a-1;

MUL: out-a\*b;

DIV: out=a/b;

SHL: out=a<<b;

SHR: out=a>>b;

AND: out-a&<b;

OR: out=ab;

INV: out=~a;

NAND:out=(a&b);

NOR:out=(ab):

endcase

XOR:out=a^b:

XNOR:out=(a^b);

BUF:out=a;

default: out=16h0000;

assign d out = (oe) ? out :16'hzzzz; endmodule