module ALU(A,B,Cn,OP,F,CLK);

input [3:0] A,B,OP,Cn;

input CLK;

output reg [3:0] F;

always@(posedge CLK or OP)

begin

case(OP)

4'b0000:F = A; //MOV

4'b0001:F = A + B; //ADD

4'b0010:F = A + B + Cn; //ADDC

4'b0011:F = A - B; //SUB

4'b0100:F = A - B - Cn; //SUBB

4'b0101:F = A & B; //AND

4'b0110:F = A | B; //OR

4'b0111:F = ~A; //NOT

4'b1000:F = A^B; //XOR

4'b1001:F = A + 1; //INC

4'b1010:F = A - 1; //DEC

4'b1011:F = A^~B; //XNOR

4'b1100:F = A + 4; //+4

endcase

end

endmodule

`timescale 1ns/1ps

module tb;

reg [3:0] A,B,OP,Cn;

reg CLK = 0;

wire [3:0] F;

ALU uut(.A(A),.B(B),.Cn(Cn),.F(F),.OP(OP),.CLK(CLK));

always #1 CLK = ~CLK;

initial

begin

OP = 0;

A = 4'b0101;B = 4'b0101;

#2

OP = 1;

A = 4'b0101;B = 4'b0101;

#2

OP = 2; Cn = 0;

A = 4'b0101;B = 4'b0101;

#2

OP = 2; Cn = 1;

A = 4'b0101;B = 4'b0101;

#2

OP = 3; Cn = 0;

A = 4'b0101;B = 4'b0101;

#2

OP = 4; Cn = 0;

A = 4'b0101;B = 4'b0101;

#2

OP = 4; Cn = 1;

A = 4'b0101;B = 4'b0101;

#2

OP = 5; Cn = 0;

A = 4'b0101;B = 4'b0101;

#2

OP = 6;

A = 4'b0101;B = 4'b0101;

#2

OP = 7;

A = 4'b0101;B = 4'b0101;

#2

OP = 8;

A = 4'b0101;B = 4'b0101;

#2

OP = 9;

A = 4'b0101;B = 4'b0101;

#2

OP = 10;

A = 4'b0101;B = 4'b0101;

#2

OP = 11;

A = 4'b0101;B = 4'b0101;

#2

OP = 12;

A = 4'b0101;B = 4'b0101;

end

endmodule