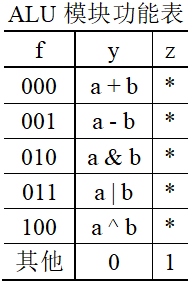
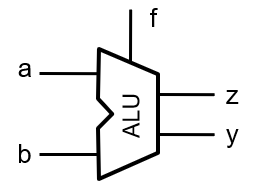
# Lab01-运算器及其应用 实验报告

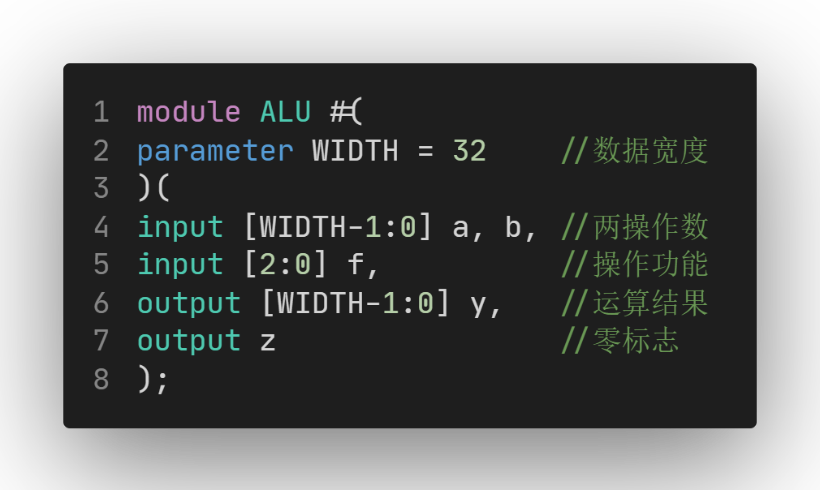
## 实验过程

### 32bit ALU

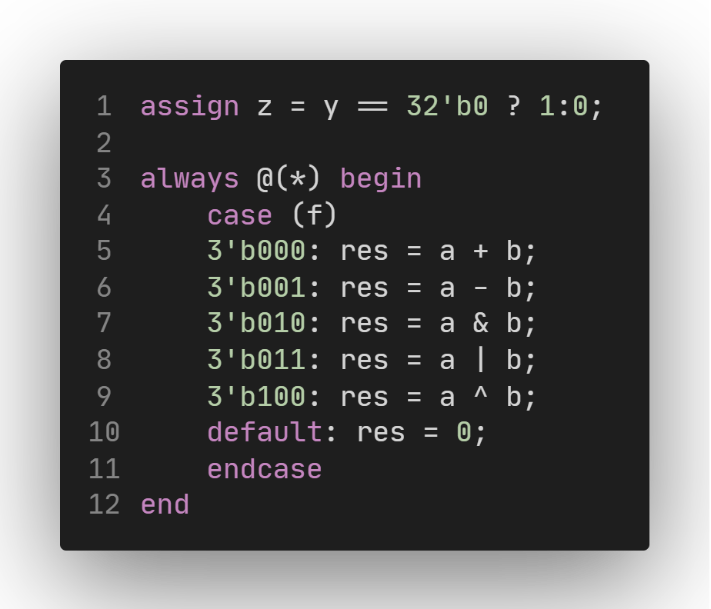
1. Datapath & function



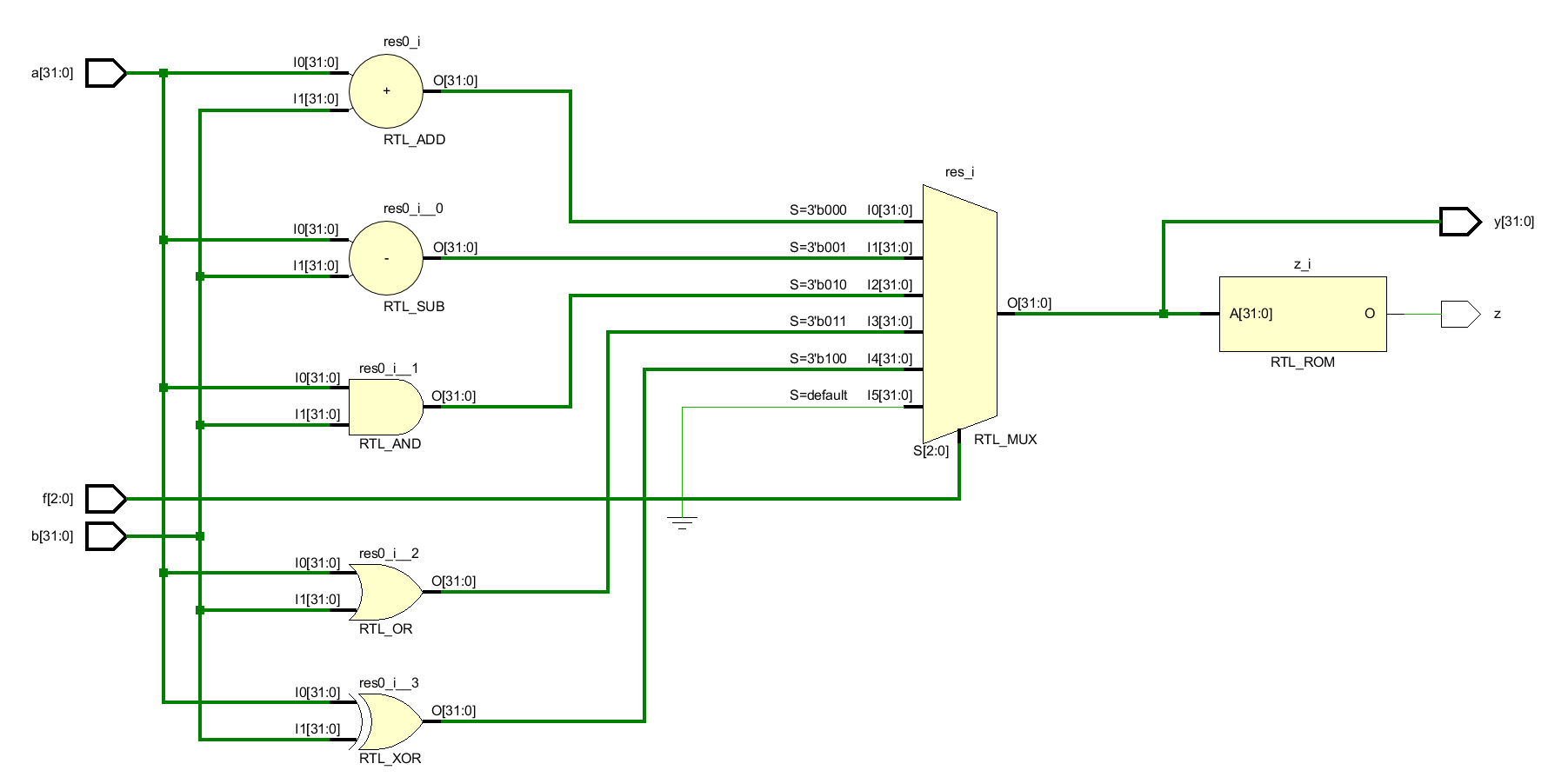
1. Interface

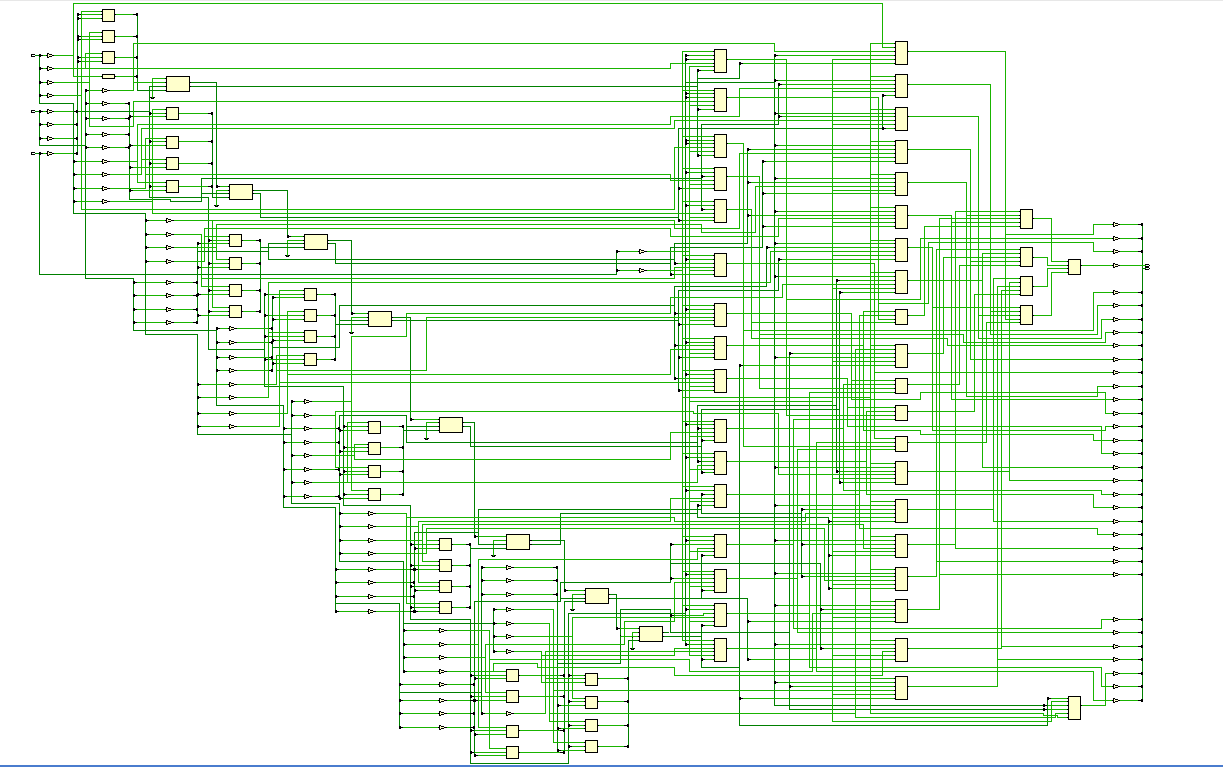


1. 核心

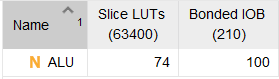


1. 电路

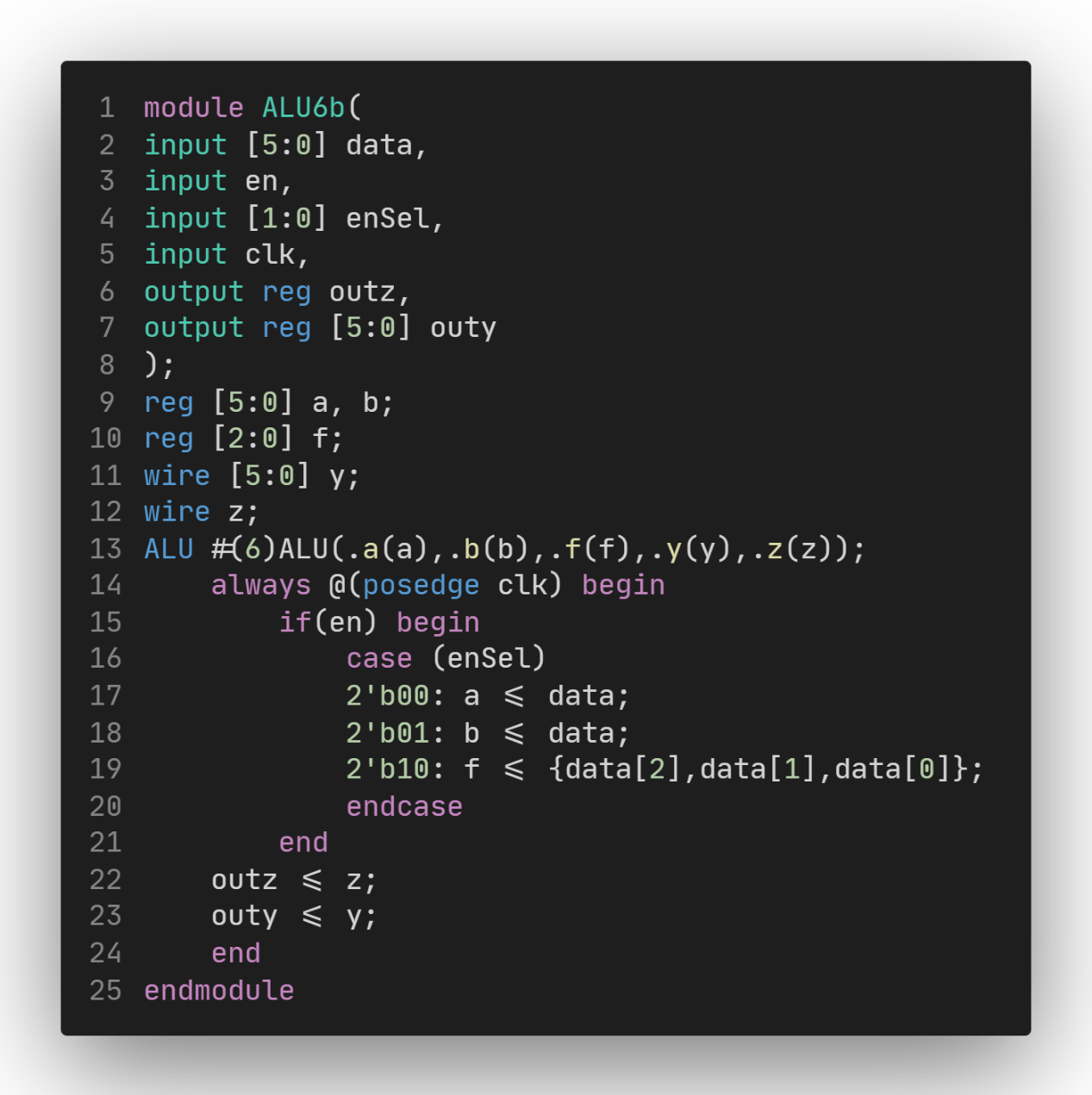


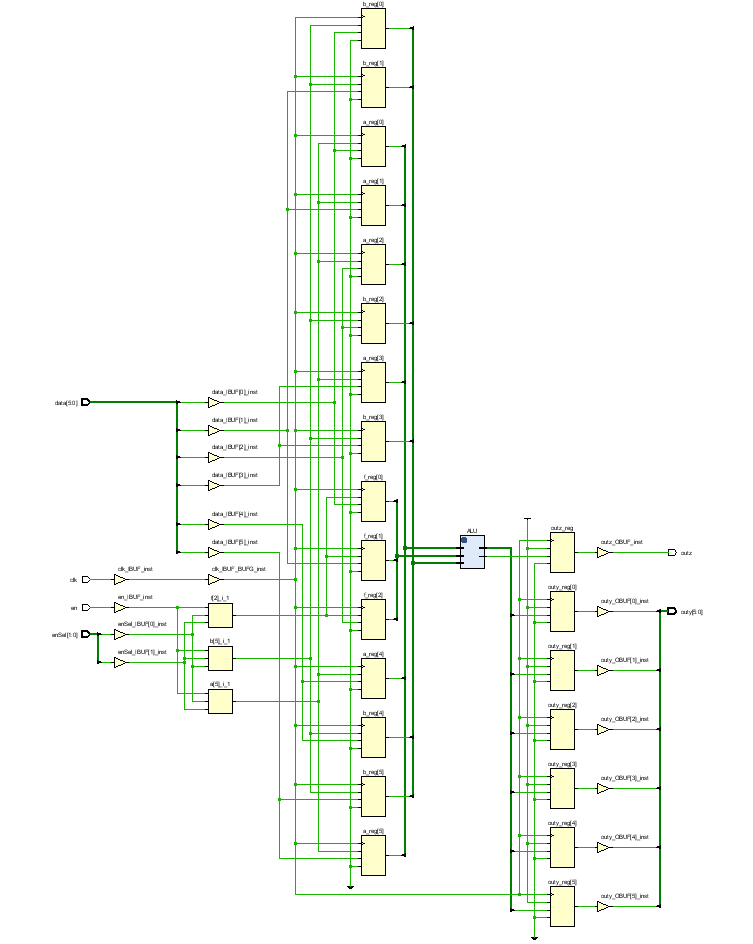
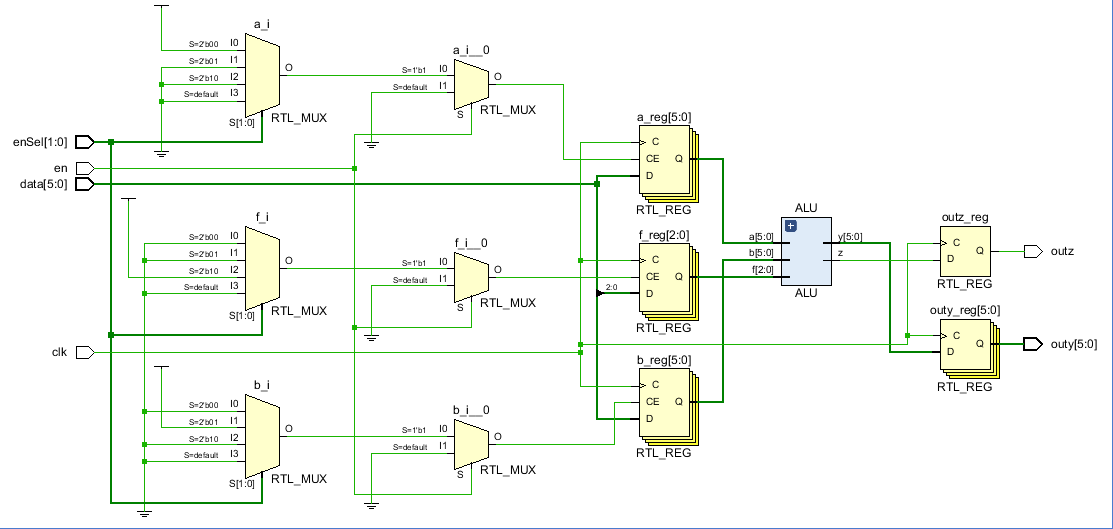


1. 性能

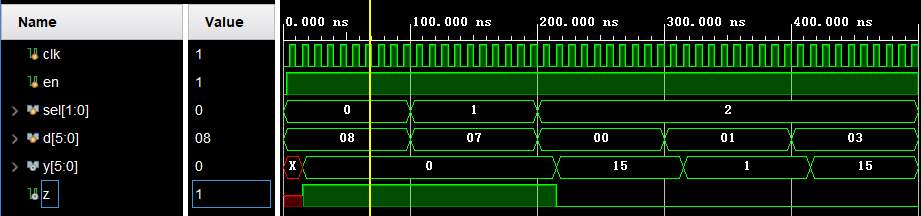


### 6bit ALU

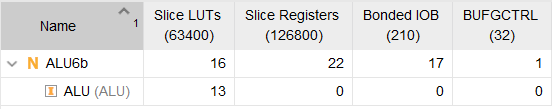
1. Code
2. RTL & 综合仿真

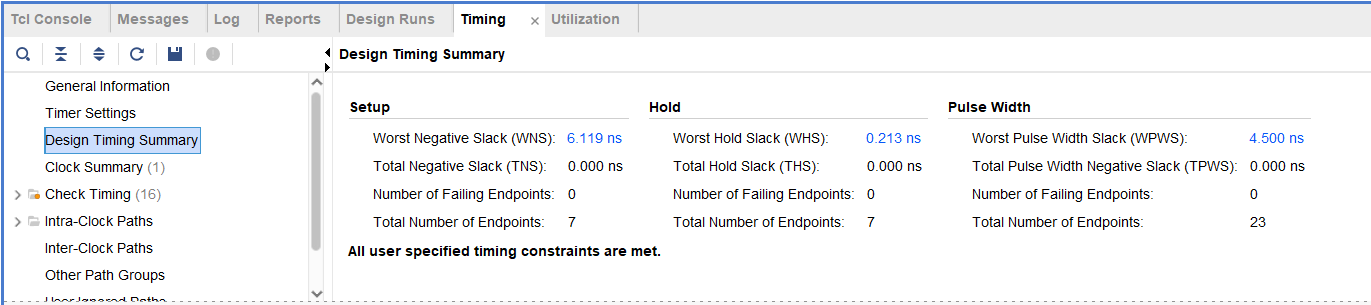


1. 仿真

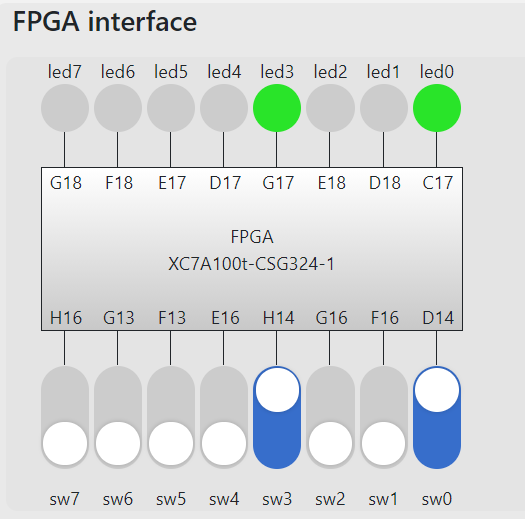


1. 性能

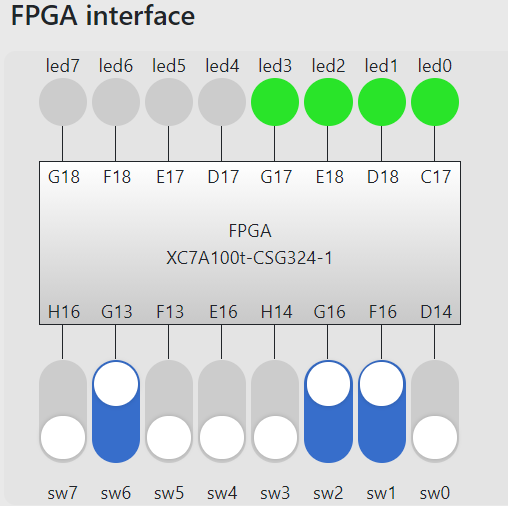




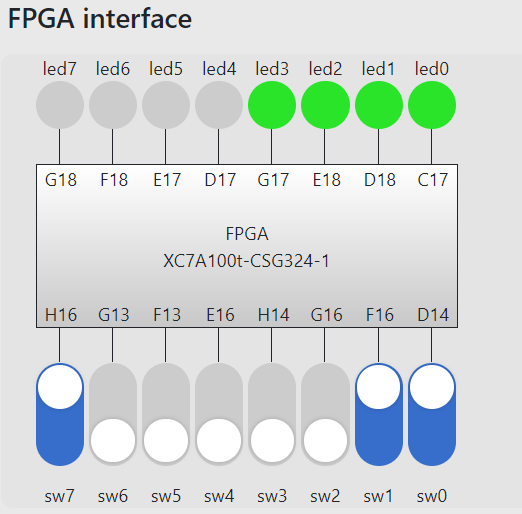
1. 下载测试
2. 输入a



1. 输入b（此时f初始为0，加运算）

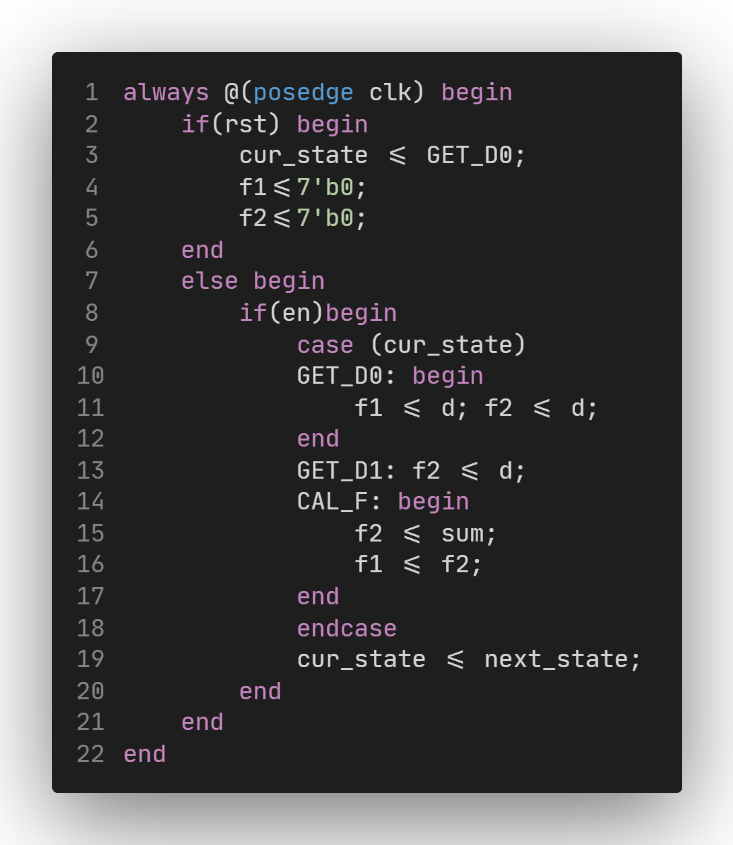
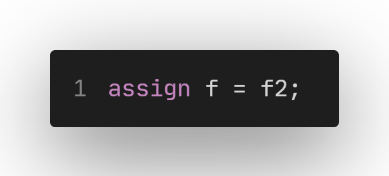
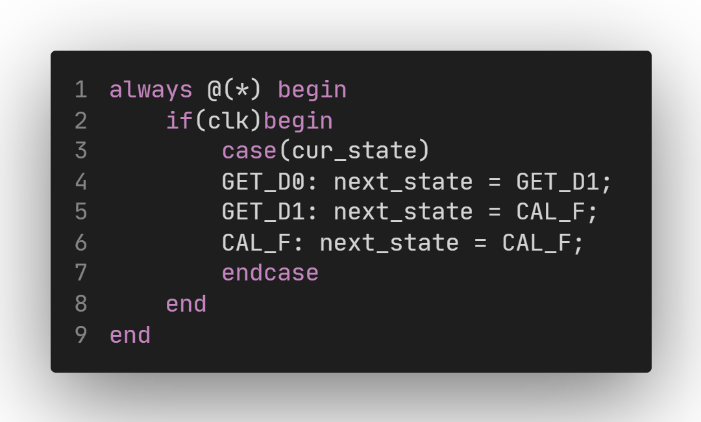


1. 输入f（与运算）

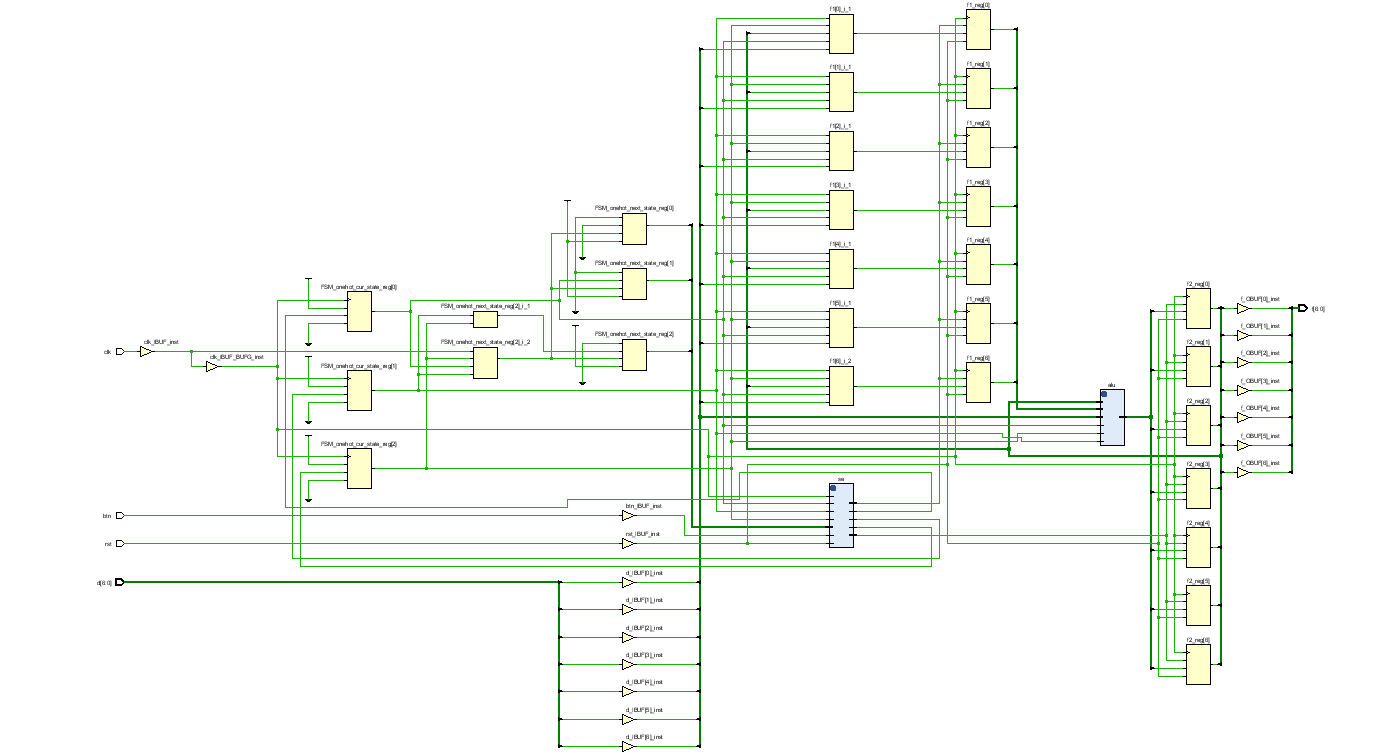
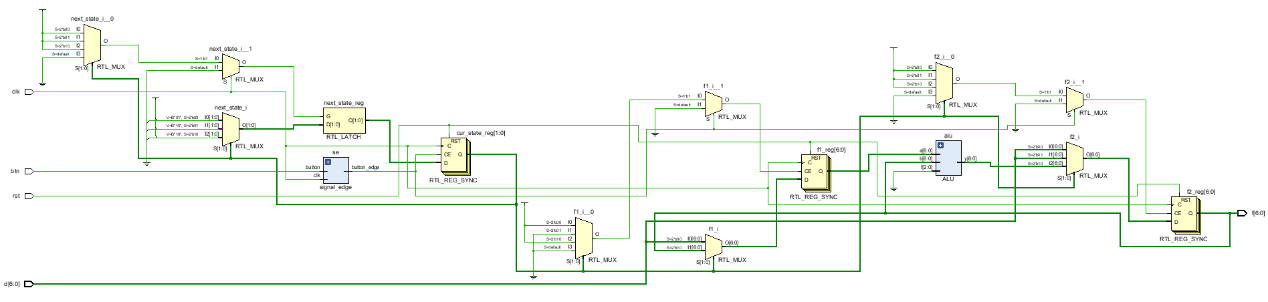


### FLS

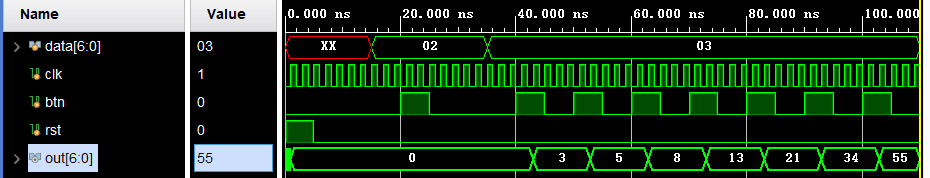
1. Code



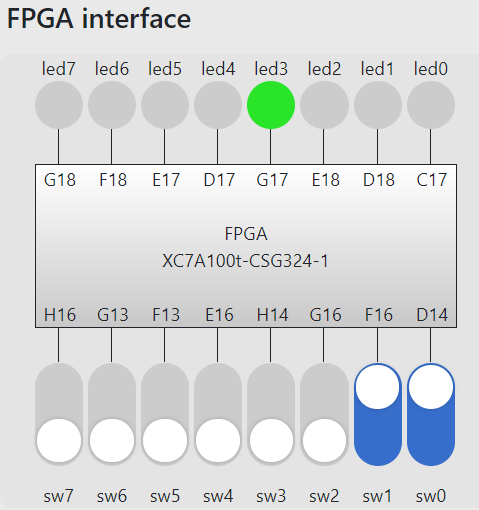
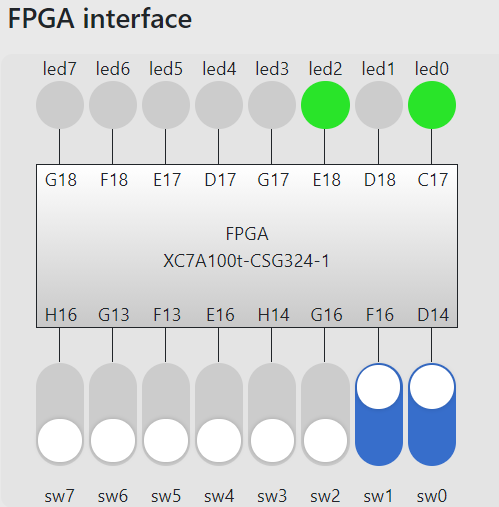
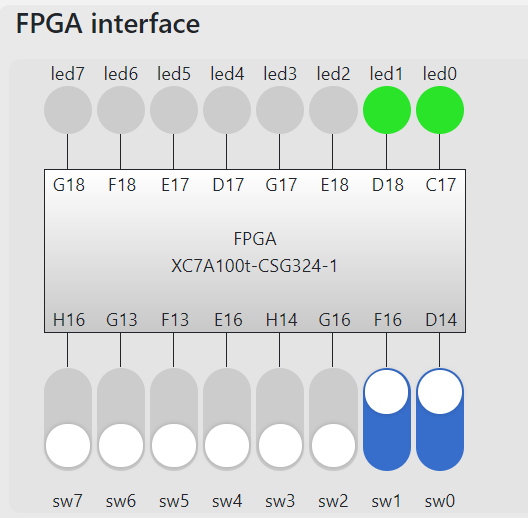
1. 电路



1. 仿真



1. 下载测试



## 总结

本次实验本身难度较低，给同学们提供了一个回忆verilog的机会。再实验过程中回忆、巩固了verilog相关的基本知识。

## Appendix

代码

1. ALU

module ALU #(

parameter WIDTH = 32    //数据宽度

)(

input [WIDTH-1:0] a, b, //两操作数

input [2:0] f,          //操作功能

output [WIDTH-1:0] y,   //运算结果

output z                //零标志

);

reg [WIDTH-1:0] res;

assign y = res;

assign z = y == 32'b0 ? 1:0;

always @(\*) begin

    case (f)

    3'b000: res = a + b;

    3'b001: res = a - b;

    3'b010: res = a & b;

    3'b011: res = a | b;

    3'b100: res = a ^ b;

    default: res = 0;

    endcase

end

endmodule

1. 6bit ALU

module ALU6b(

input [5:0] data,

input en,

input [1:0] enSel,

input clk,

output reg outz,

output reg [5:0] outy

);

reg [5:0] a, b;

reg [2:0] f;

wire [5:0] y;

wire z;

ALU #(6)ALU(.a(a),.b(b),.f(f),.y(y),.z(z));

    always @(posedge clk) begin

        if(en) begin

            case (enSel)

            2'b00: a <= data;

            2'b01: b <= data;

            2'b10: f <= {data[2],data[1],data[0]};

            endcase

        end

    outz <= z;

    outy <= y;

    end

endmodule

1. FLS

module  FLS(

  input  clk, rst, btn,

  input  [6:0]  d,

  output [6:0]  f

);

parameter GET\_D0 = 2'b00; // To get the 1st number

parameter GET\_D1 = 2'b01; // To get the 2nd number

parameter CAL\_F = 2'b10;  // To calculate the next number

wire en;

signal\_edge se(clk,btn,en);

reg[1:0] cur\_state, next\_state;

reg[6:0] f1, f2;

wire[6:0] sum;

ALU #(7)alu(.a(f1),.b(f2),.f(3'b000), .y(sum));

always @(\*) begin

    if(clk)begin

        case(cur\_state)

        GET\_D0: next\_state = GET\_D1;

        GET\_D1: next\_state = CAL\_F;

        CAL\_F: next\_state = CAL\_F;

        endcase

    end

end

always @(posedge clk) begin

    if(rst) begin

        cur\_state <= GET\_D0;

        f1<=7'b0;

        f2<=7'b0;

    end

    else begin

        if(en)begin

            case (cur\_state)

            GET\_D0: begin

                f1 <= d; f2 <= d;

            end

            GET\_D1: f2 <= d;

            CAL\_F: begin

                f2 <= sum;

                f1 <= f2;

            end

            endcase

            cur\_state <= next\_state;

        end

    end

end

assign f = f2;

endmodule

// 取时钟上升沿

module signal\_edge(

    input clk,

    input button,

    output button\_edge

    );

    reg button\_r1,button\_r2;

    always@(posedge clk)

        button\_r1 <= button;

    always@(posedge clk)

        button\_r2 <= button\_r1;

    assign button\_edge = button\_r1 & (~button\_r2);

endmodule

1. 6bit ALU 仿真

module ALU6\_test();

    reg clk,en;

    reg [1:0] sel;

    reg [5:0] d;

    wire [5:0] y;

    wire z;

    ALU6b ALU6b(.clk(clk),.en(en),.enSel(sel),.data(d),.outy(y),.outz(z));

    initial clk=0;

    always #5 clk=~clk;

    initial

    begin

        en=1'b0;

        #3 en=1'b1;

    end

    // always #1 en=~en;

    initial

    begin

            sel=2'b00;d=6'b01000;

        #100 sel=2'b01;d=6'b000111;

        #100 sel=2'b10;d=6'b000000;

        #100 d=6'b01;

        #100 d=6'b11;

        #100 $finish;

    end

endmodule

1. FLS 仿真

module fls\_testbench();

    reg[6:0] data;

    reg clk,btn,rst;

    wire[6:0] out;

    fls fls(clk,rst,btn,data,out);

    initial clk = 0;

    always #1 clk=~clk;

    initial

    begin

        rst = 1; btn = 0;

    #5 rst = 0;

    #10 data = 7'd2;

    #5  btn = 1; #5  btn =0;

    #10 data = 7'd3;

    #5  btn = 1; #5  btn =0;

    #5  btn = 1; #5  btn =0;

    #5  btn = 1; #5  btn =0;

    #5  btn = 1; #5  btn =0;

    #5  btn = 1; #5  btn =0;

    #5  btn = 1; #5  btn =0;

    #5  btn = 1; #5  btn =0;

    #5 $finish;

    end

endmodule