**实验报告**

实验题目： 任务、函数与测试台；锁存器与触发器

日期： 2018 年 11 月 2 日

姓名: 罗晏宸 学号: PB17000297 成绩:\_\_\_\_\_\_\_\_\_\_\_\_

**实验目的：**

1. 学习Verilog中任务(task)与函数(function)的语法，体会两者的异同
2. 利用task与function训练模块化设计风格，提⾼程序可读性并开发可重⽤性代码
3. 学习Verilog中测试台(testbench)代码的编写，利用其验证硬件模型的功能正确性，并学会通过设定惯性延迟产生特定波形
4. 复习各种类型的锁存器与带有控制信号的触发器的时序电路基础
5. 模拟实现锁存器与触发器功能，初步开发时序逻辑电路

**实验内容**（截图、照片与代码）

Lab4\_1\_1

代码 add\_two\_values\_task.v

`timescale 1ns / 1ps

module add\_two\_values\_task(

    input [3:0] ain,bin,

output reg cout,

output reg [3:0] sum

);

task add\_two\_values;

    input [3:0] ain,bin;

    output cout;

    output [3:0] sum;

    reg [4:0] temp;

    begin

        temp = ain+bin;

        cout = temp[4];

        sum = temp[3:0];

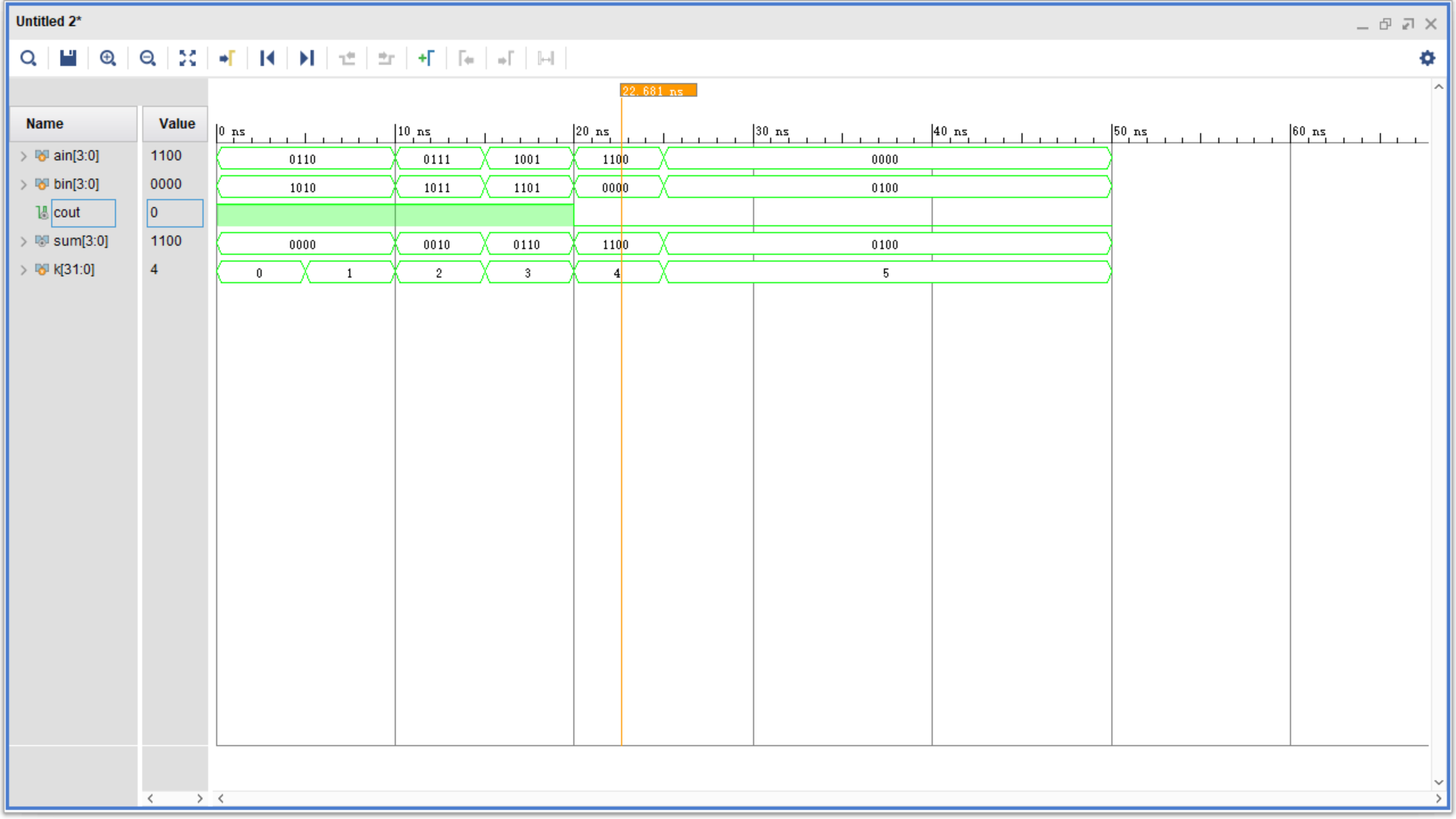
    end

endtask

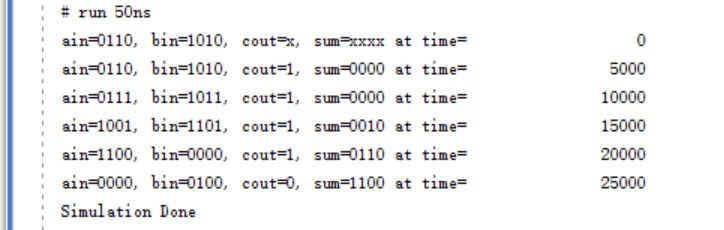
always @(\*)

    add\_two\_values (ain,bin,cout,sum);

endmodule



截图 1-lab4\_1\_1仿真A



截图 2-lab4\_1\_1仿真B

Lab4\_2\_1

代码 add\_two\_values\_function.v

`timescale 1ns / 1ps

module add\_two\_values\_function(

input [3:0] ain,

input [3:0] bin,

output reg [4:0] sum

);

function [4:0] add\_two\_values;

    input [3:0] ain,bin;

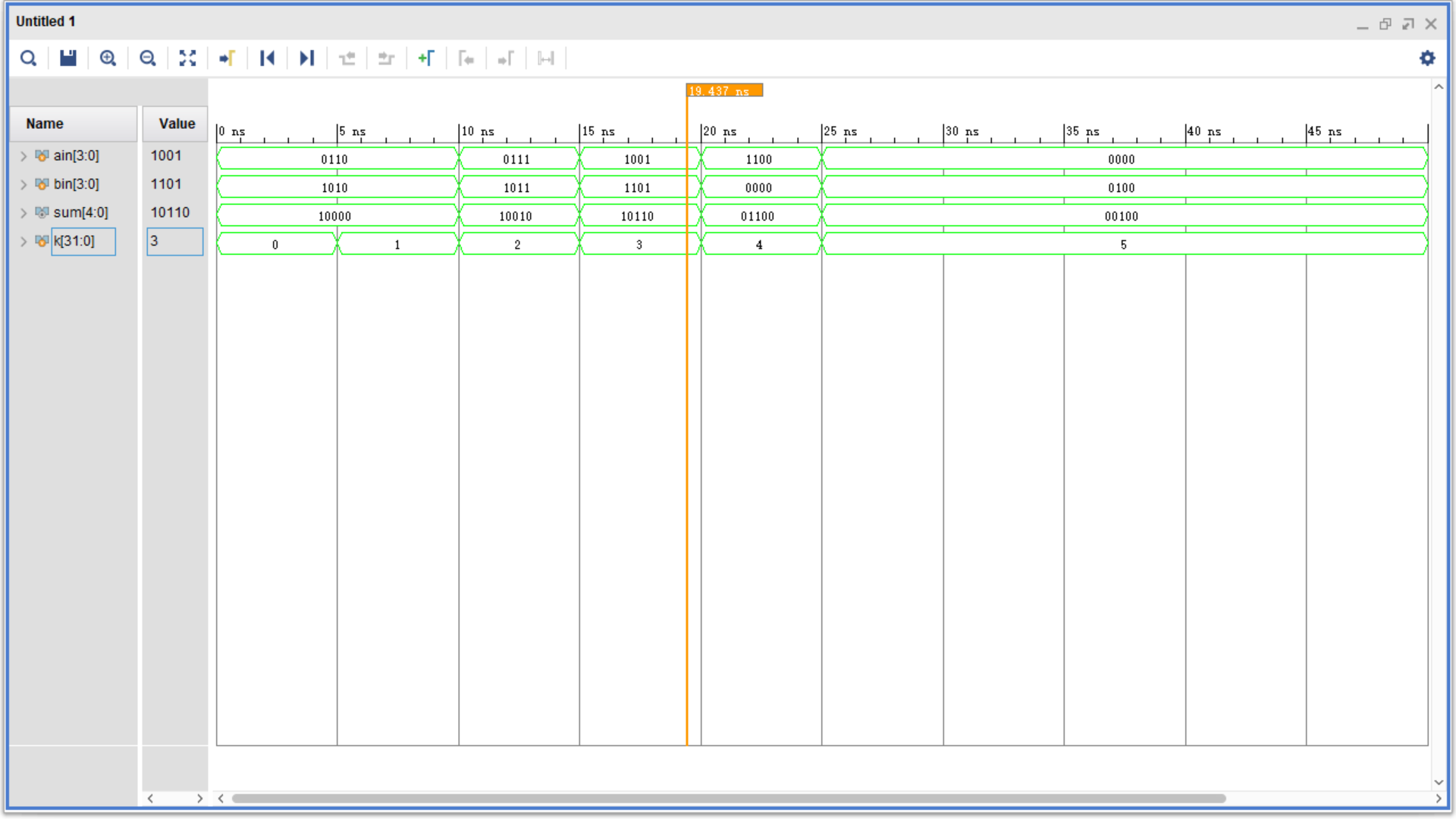
    add\_two\_values = ain + bin;

endfunction

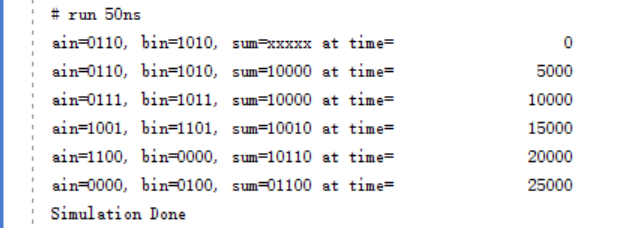
always @(\*)

    sum = add\_two\_values(ain,bin);

endmodule



截图 3-lab4\_2\_1仿真A



截图 4-lab4\_2\_1仿真B

Lab4\_3\_2

代码lab4\_3\_2\_tb.v

`timescale 1ns / 1ps

module lab4\_3\_2\_tb(

);

reg a,g1,g2;

lab4\_3\_2 DUT (.a(a),.g1(g1),.g2(g2));

initial

begin

    a=0;g1=0;g2=1;

    #40 a=1;

    #20 g1=1;

    #20 g2=0;

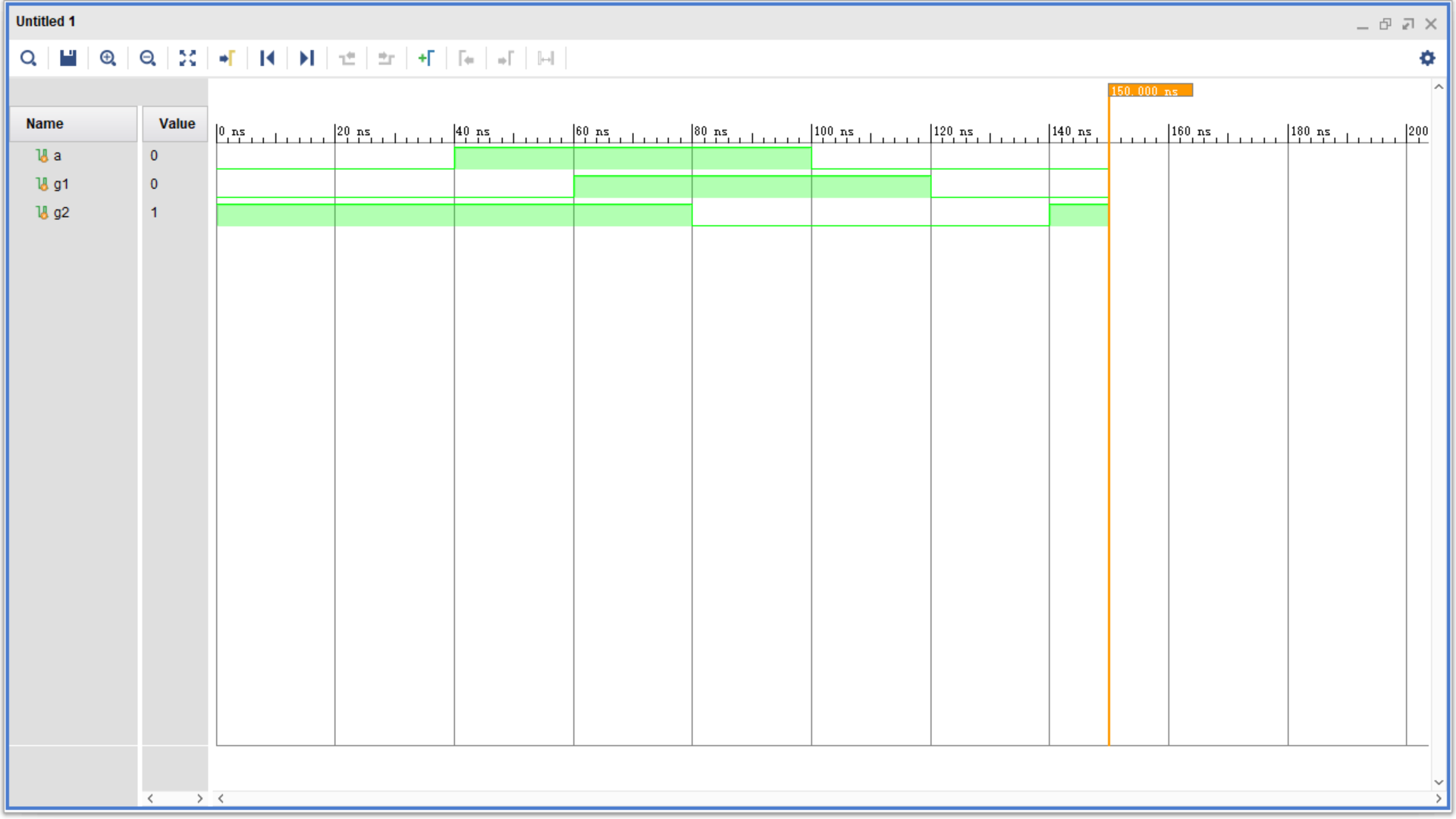
    #20 a=0;

    #20 g1=0;

    #20 g2=1;

end

endmodule



截图 5-lab4\_3\_2仿真

Lab5\_2\_2

代码D\_latch\_behavior.v

`timescale 1ns / 1ps

module D\_latch\_behavior (input D, input Enable, output reg Q, output reg Qbar);

always @ (D or Enable)

    if(Enable)

    begin

        Q <= D;

        Qbar <= ~D;

    end

endmodule

代码D\_ff\_posedge\_behavior.v

`timescale 1ns / 1ps

module D\_ff\_posedge\_behavior (input D, input Clk, output reg Q);

    always @ (posedge Clk)

        if(Clk)

        begin

            Q <= D;

        end

endmodule

代码D\_ff\_negedge\_behavior.v

`timescale 1ns / 1ps

module D\_ff\_negedge\_behavior (input D, input Clk, output reg Q);

    always @ (negedge Clk)

        if(~Clk)

        begin

            Q <= D;

        end

endmodule

代码SR\_latch\_dataflow.v

`timescale 1ns / 1ps

module SR\_latch\_dataflow(

    input R,

    input S,

    output Q,

    output Qbar

    );

//  assign #2 Q\_i = Q;

//  assign #2 Qbar\_i = Qbar;

    assign #2 Qbar = ~ (S | Q);

    assign #2 Q = ~ (R | Qbar);

endmodule

代码testbench.v

`timescale 1ns / 1ps

module testbench(

);

reg Clock,D,S;

wire Qa,Qb,Qc,Qd;

D\_latch\_behavior DUT1 (.D(D), .Enable(Clock), .Q(Qa), .Qbar());

D\_ff\_posedge\_behavior DUT2 (.D(D), .Clk(Clock), .Q(Qb));

D\_ff\_negedge\_behavior DUT3 (.D(D), .Clk(Clock), .Q(Qc));

    SR\_latch\_dataflow DUT4 (.R(D),.S(S),.Q(Qd),.Qbar());

    initial

    begin

        Clock=0;D=0;S=0;

        #6 D=1;

        #2 Clock=1;

        #2 D=0;

        #2 D=1;

        #1 S=1;

        #2 D=0;

        #3 Clock=0;

        #2 D=1;

        #3 D=0;S=0;

        #1 D=1;

        #2 D=0;

        #1 Clock=1;

        #2 D=1;

        #1 D=0;

        #2 D=1;

        #1 S=1;

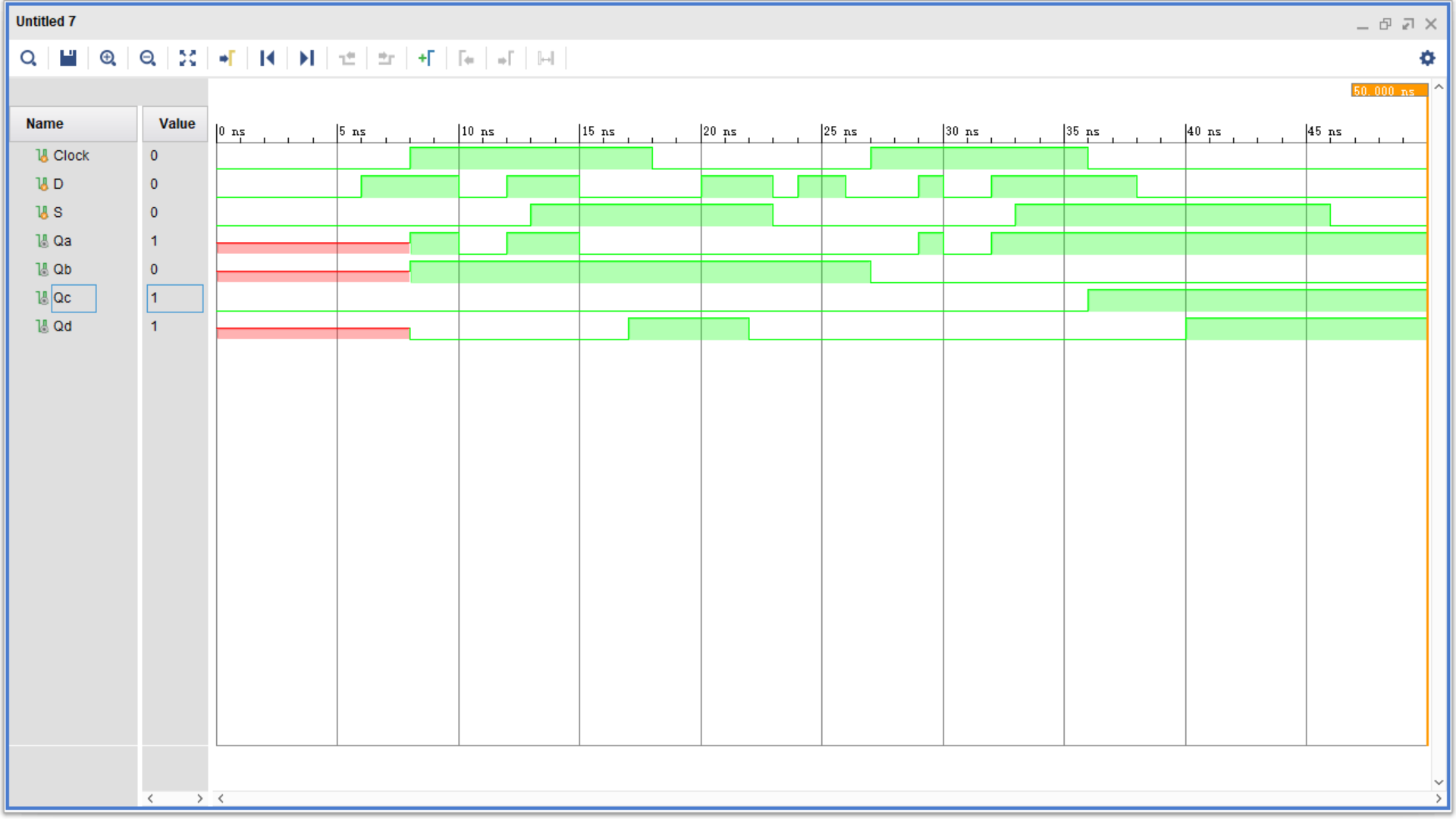
        #3 Clock=0;

        #2 D=0;

        #8 S=0;

    end

endmodule



截图 6-lab5\_2\_2仿真

**实验总结：**

在本次实验的lab4部分中，学习了程序模块task与function的使用，进一步锻炼了自身程序设计的能力，培养良好的代码风格，提高了开发效率；通过对testbench的学习，包括对新的系统函数$display与$time的了解，掌握了验证硬件模型与模拟波形的有利工具，并初步观察利用TCL console 控制台的输出；在lab5部分中，通过对D锁存器，RS锁存器，D触发器（上升沿与下降沿）等基础储存器的模拟与实现，开始练习设计开发时序性逻辑电路，为之后的实验与学习提供了工具与知识储备。