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：

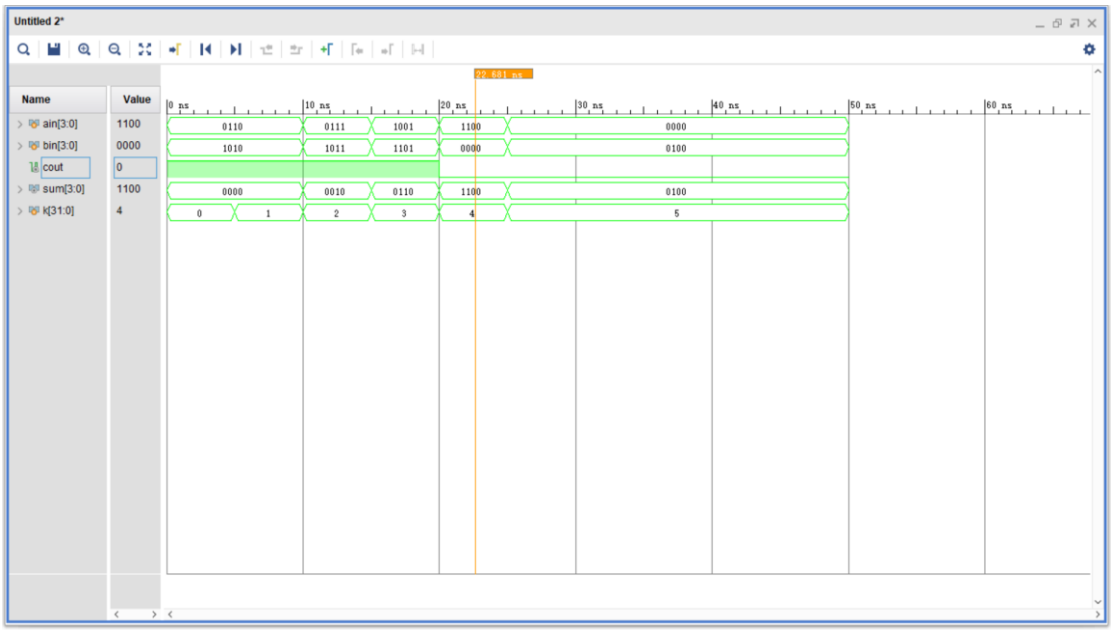
1. 学习 Verilog 中任务(task)与函数(function)的语法，体会两者的异同
2. 利用 task 与 function 训练模块化设计风格，提高程序可读性并开发可重用性代码
3. 学习 Verilog 中测试台(testbench)代码的编写，利用其验证硬件模型的功能正确性，并学会通过设定惯性延迟产生特定波形
4. 复习各种类型的锁存器与带有控制信号的触发器的时序电路基础
5. 模拟实现锁存器与触发器功能，初步开发时序逻辑电路

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

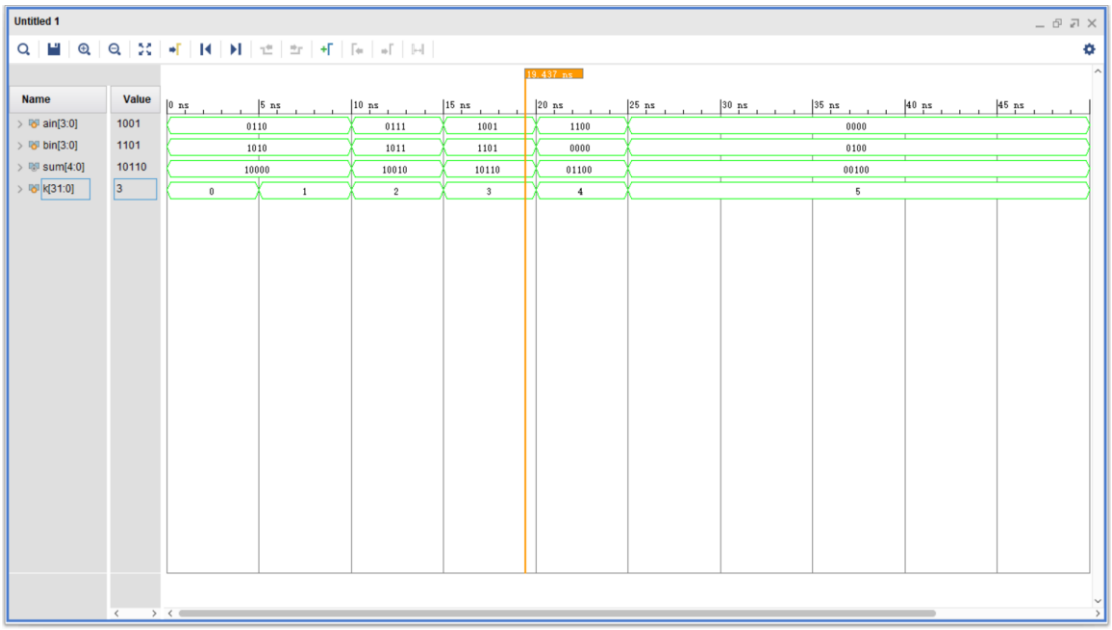
```
# run 50ns
ain=0110, bin=1010, cout=x, sum=xxxx at time=                0
ain=0110, bin=1010, cout=1, sum=0000 at time=             5000
ain=0111, bin=1011, cout=1, sum=0000 at time=            10000
ain=1001, bin=1101, cout=1, sum=0010 at time=            15000
ain=1100, bin=0000, cout=1, sum=0110 at time=            20000
ain=0000, bin=0100, cout=0, sum=1100 at time=            25000
Simulation Done
```

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

```
# run 50ns
ain=0110, bin=1010, sum=xxxxx at time=                0
ain=0110, bin=1010, sum=10000 at time=             5000
ain=0111, bin=1011, sum=10000 at time=            10000
ain=1001, bin=1101, sum=10010 at time=            15000
ain=1100, bin=0000, sum=10110 at time=            20000
ain=0000, bin=0100, sum=01100 at time=            25000
Simulation Done
```

截图 4-lab4_2_1 仿真 B

Lab4_3_2

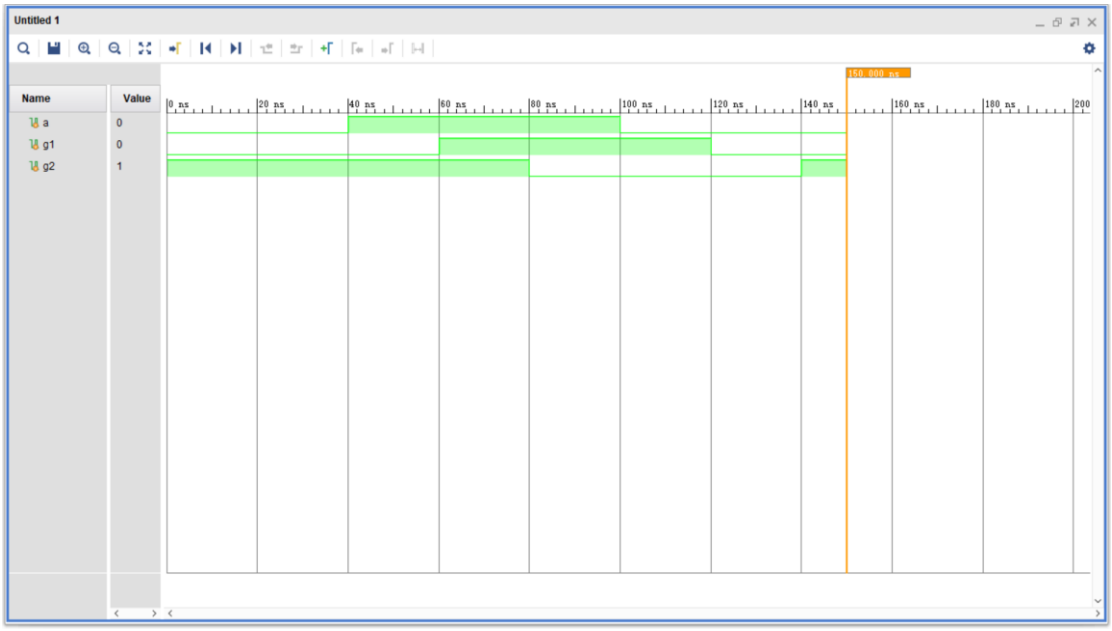
代码 lab4_3_2_tb.v

```
`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
```

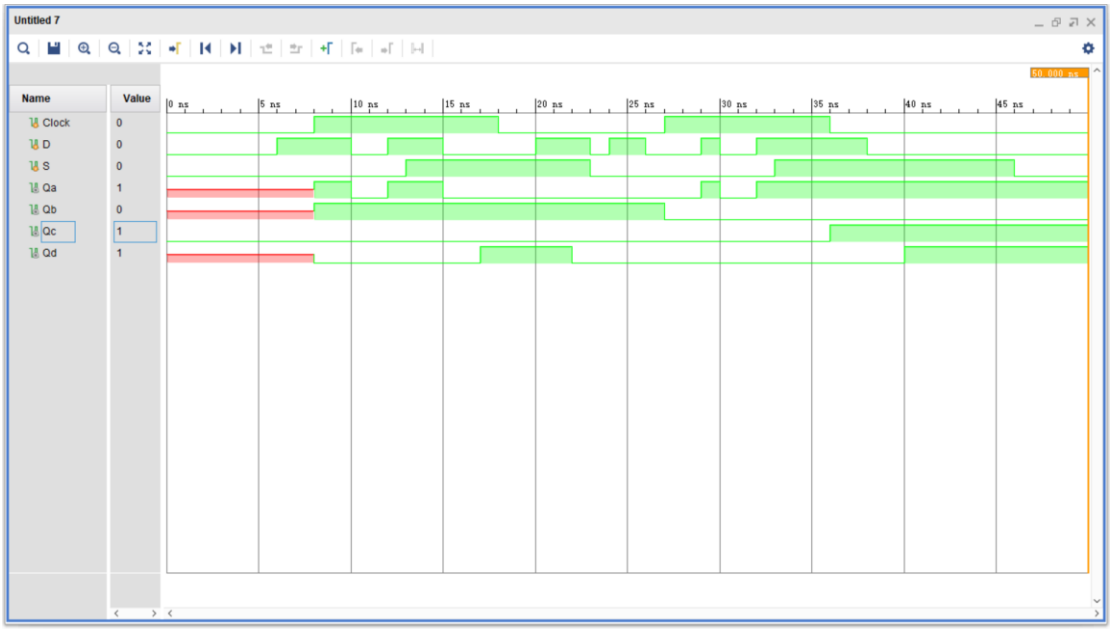
代码 testbench.v

```
`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。