

中国科学技术大学计算机学院  
《数字电路实验》报告



实验题目：使用 Vivado 进行仿真

学生姓名：张郑飞扬

学生学号：PB21071416

完成日期：2022. 11. 10

计算机实验教学中心制

2020 年 09 月

## 【实验题目】

使用 Vivado 进行仿真

## 【实验目的】

熟悉 Vivado 软件的下载、安装及使用

学习使用 Verilog 编写仿真文件

学习使用 Verilog 进行仿真，查看并分析波形文件

## 【实验环境】

PC 一台

vlab.ustc.edu.cn

Vivado 工具

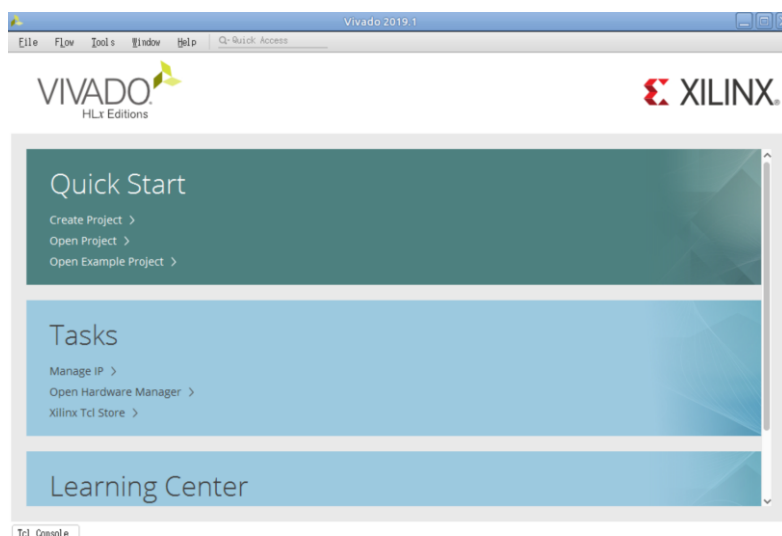
## 【实验过程】

Step1. 下载并安装 Vivado 环境

本人直接采用 VLAB 系统中已经配置好的实验环境进行实验。

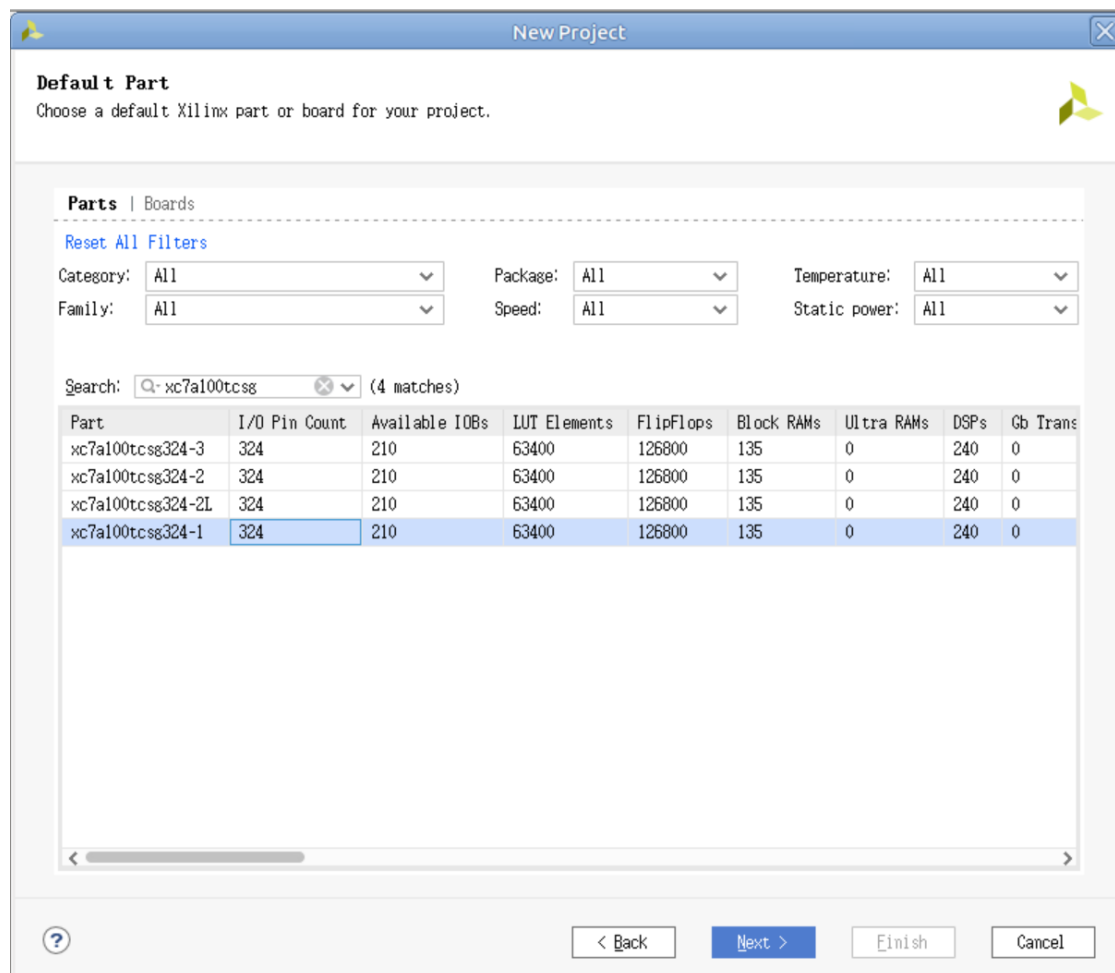
Step2. 建立 Vivado 工程

选择 2019.1 版本的 Vivado，进入主界面



根据手册导引创建工程。在 default part 页面选择

xc7a100tcsg324-1 型号的器件。



Step3. 添加 Verilog 设计文件

根据实验手册，选择 Add or create design source 选项，在生成的 Verilog 文件中，输入如下代码：

```
23 module Test(  
24   input [3:0] a,b,c,d,  
25   input [1:0] sel,  
26   output reg [3:0] o  
27 );  
28 always@(*)  
29 begin  
30   case(sel)  
31     2'b00: o = a;  
32     2'b01: o = b;  
33     2'b10: o = c;  
34     2'b11: o = d;  
35     default: o = 4'h0;  
36   endcase  
37 end  
38 endmodule  
39
```

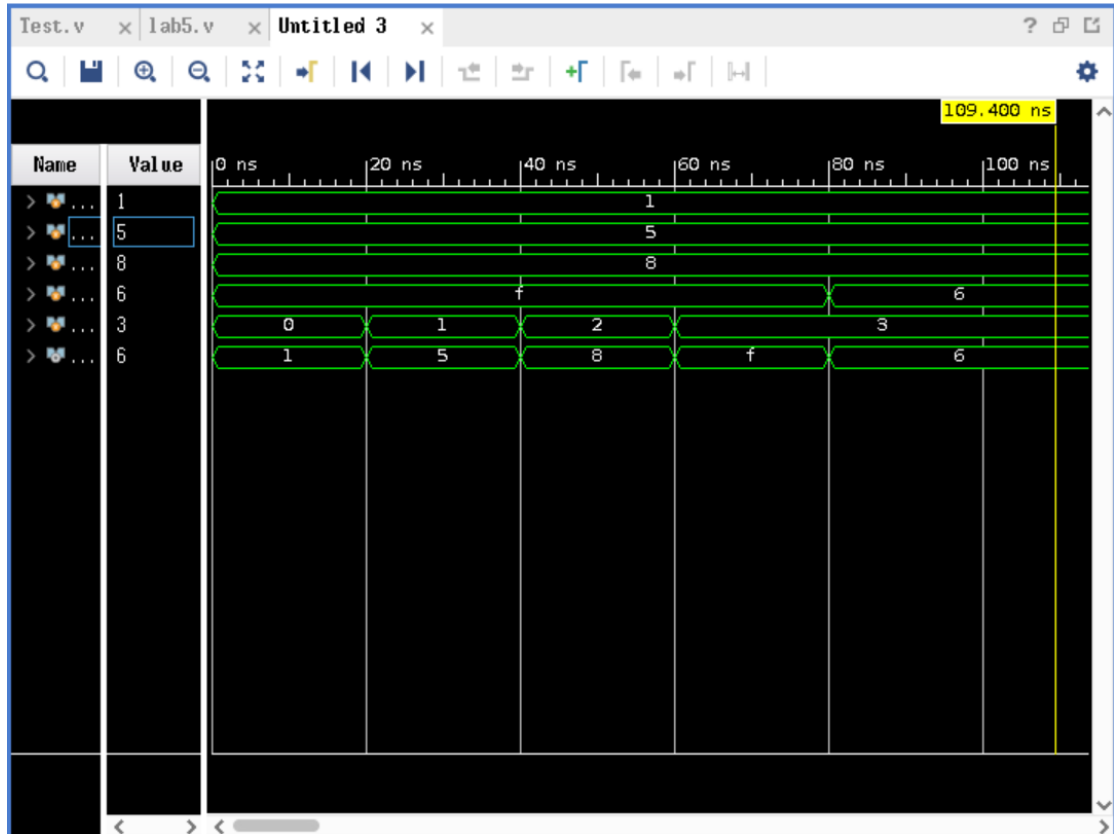
#### Step4. 添加仿真文件

根据实验手册，选择 Add or create simulation sources 选项，输入如下代码：

```
23 module lab5();  
24   reg [3:0] a,b,c,d;  
25   reg [1:0] sel;  
26   wire [3:0] o;  
27   Test Test(.a(a),.b(b),.c(c),.d(d),.sel(sel),.o(o));  
28   initial  
29   begin  
30     a = 4'h1; b = 4'h5; c = 4'h8; d = 4'hF; sel = 2'h0;  
31     #20 a = 4'h1; b = 4'h5; c = 4'h8; d = 4'hF; sel = 2'h1;  
32     #20 a = 4'h1; b = 4'h5; c = 4'h8; d = 4'hF; sel = 2'h2;  
33     #20 a = 4'h1; b = 4'h5; c = 4'h8; d = 4'hF; sel = 2'h3;  
34     #20 a = 4'h1; b = 4'h5; c = 4'h8; d = 4'h6; sel = 2'h3;  
35     #20 $finish;  
36   end  
37 endmodule  
38
```

#### Step5. 波形仿真

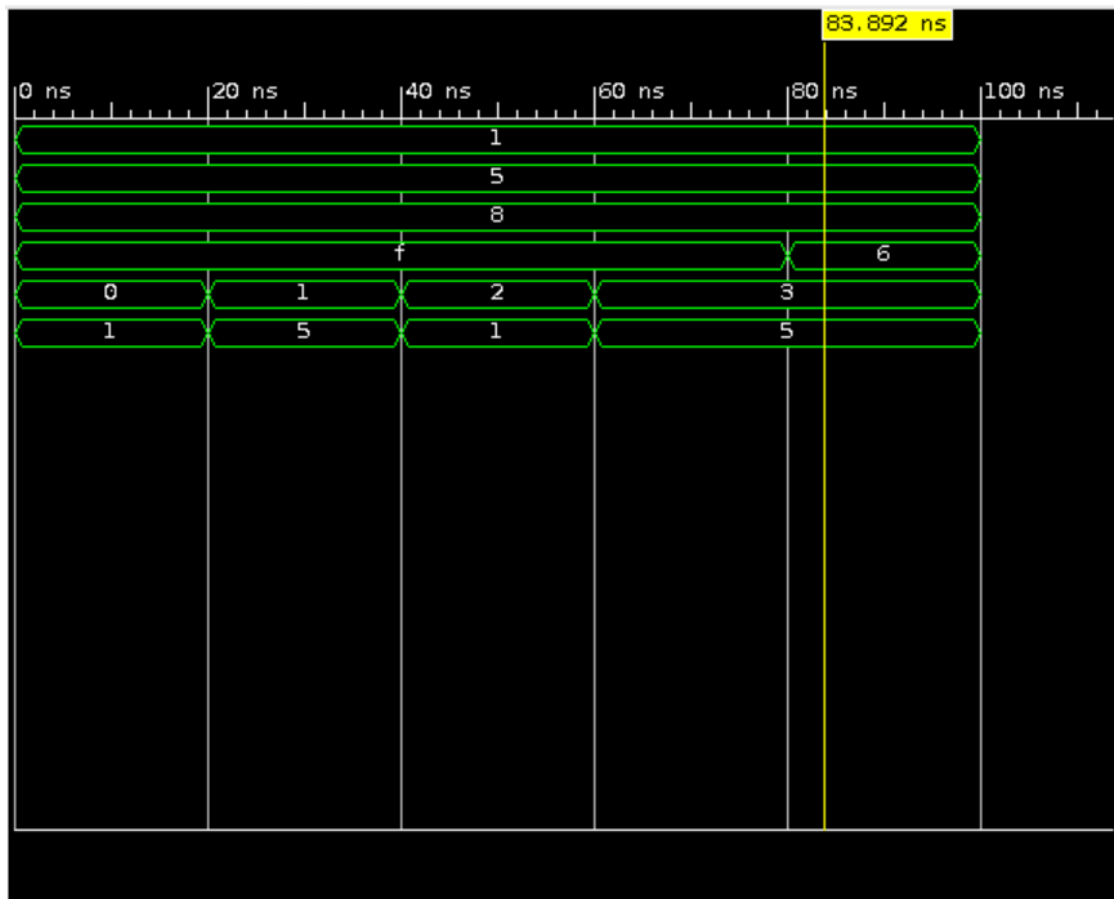
点击“Run Simulation”运行仿真工具，得到波形如下



通过观察波形我们可以发现，该电路的仿真波形符合四选一选

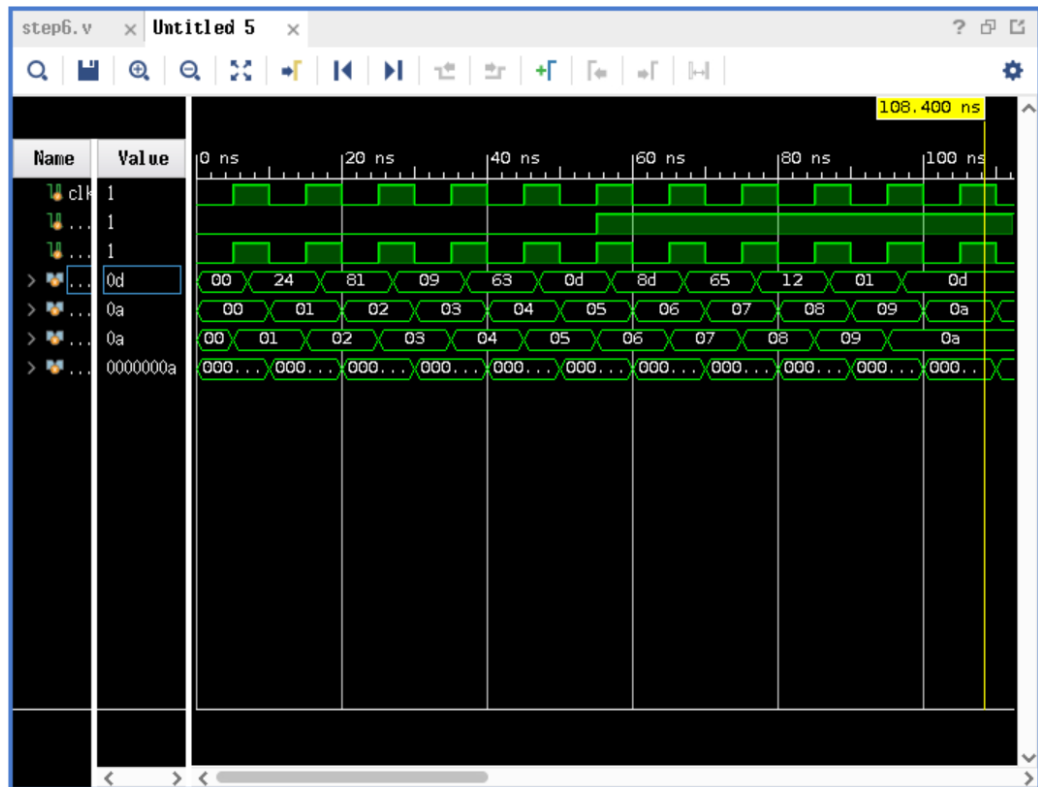
择器的行为特性， Verilog 代码设计正确。

下面关闭波形仿真窗口，打开前面的 Verilog 设计文件，将其中的“input [1:0] sel,”改成“input sel,” 重新进行仿真，观察波形结果。我们会发现其波形不符合四选一选择器的行为。



## Step6. Verilog 仿真文件常用语法

在 Vivado 中新建一个工程，加入实验手册中的仿真文件，进行仿真，观察各信号的波形。



阅读实验手册中对一些 Verilog 新关键字和语法结构的解释。

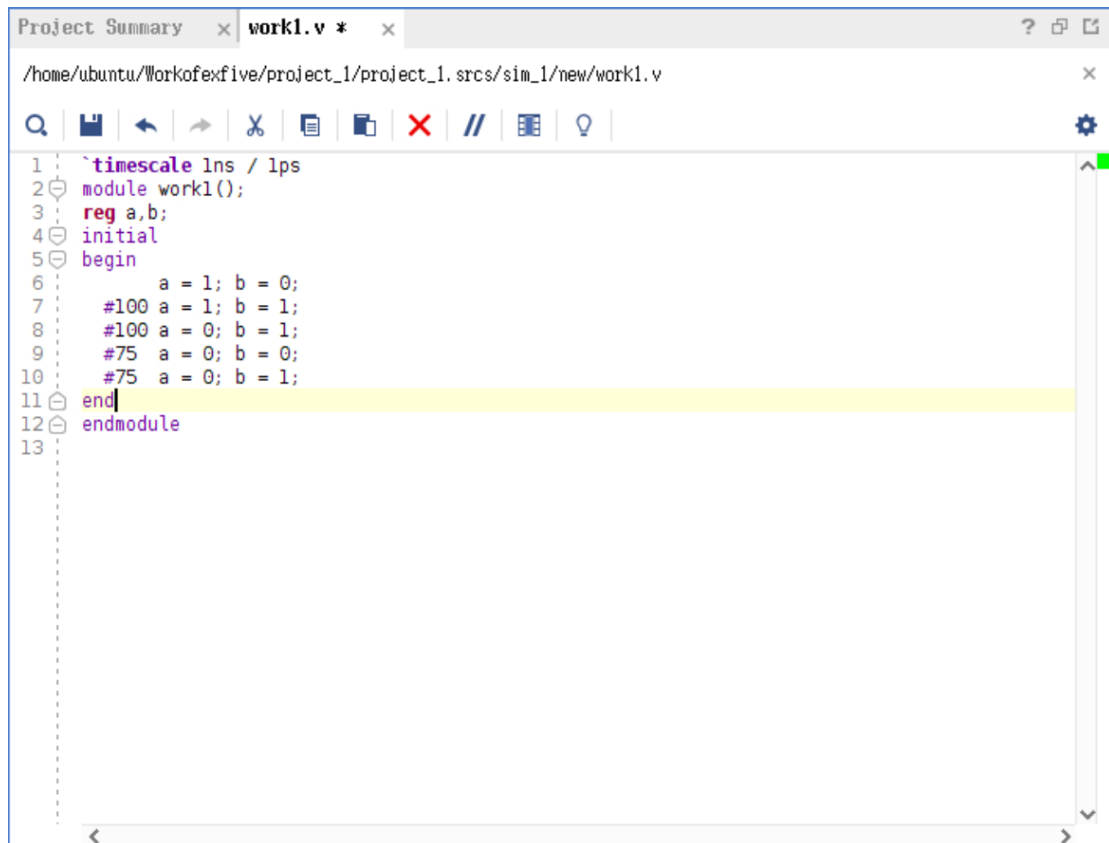
如：

**initial:** 该关键字与 **always** 同为过程语句关键字，但与 **always** 不同的是，**initial** 语句只执行一次，**initial** 语句在模拟开始时执行

## 【实验练习】

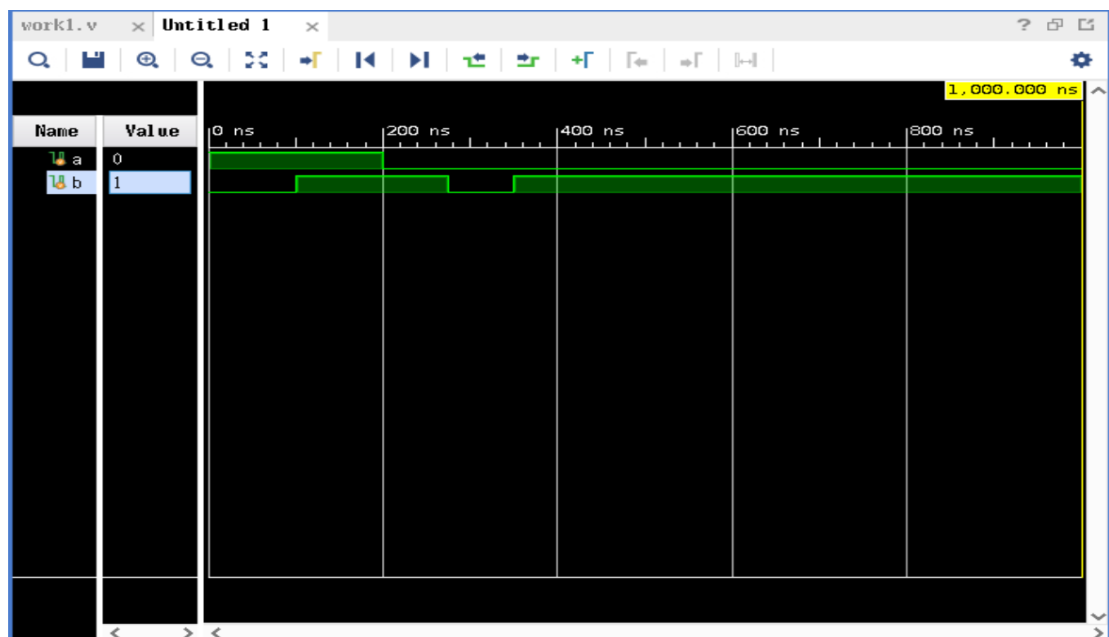
题目 1:

仿真文件如下:



```
1 `timescale 1ns / 1ps
2 module work1();
3   reg a,b;
4   initial
5   begin
6     a = 1; b = 0;
7     #100 a = 1; b = 1;
8     #100 a = 0; b = 1;
9     #75 a = 0; b = 0;
10    #75 a = 0; b = 1;
11  end
12 endmodule
13
```

得到波形图如下:



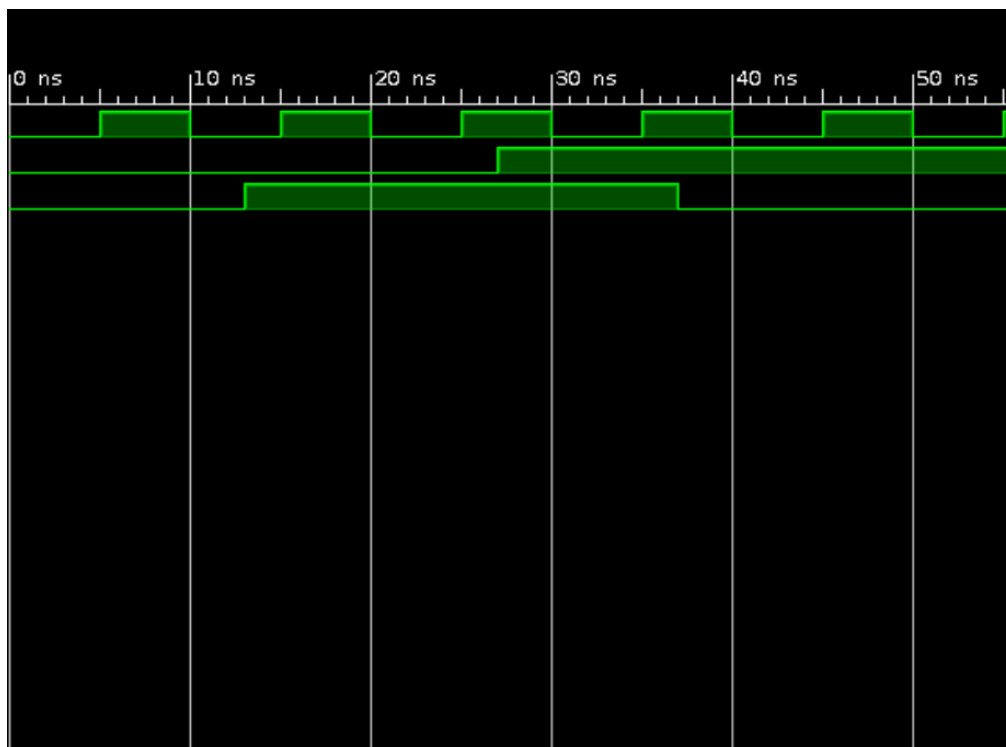
题目 2:

仿真文件如下:

```
test.v
/home/ubuntu/Experiment5/Workofexfive/work2/project_1/project_1.srcs/sim_1/new/test.v

1  timescale 1ns / 1ps
2  module test();
3      reg clk,rst_n,d;
4      initial clk = 0;
5      always #5 clk = ~clk;
6      initial
7      begin
8          rst_n = 0;
9          #27 rst_n = 1;
10     end
11     initial
12     begin
13         d = 0;
14         #13 d = 1;
15         #24 d = 0;
16     end
17 endmodule
18
```

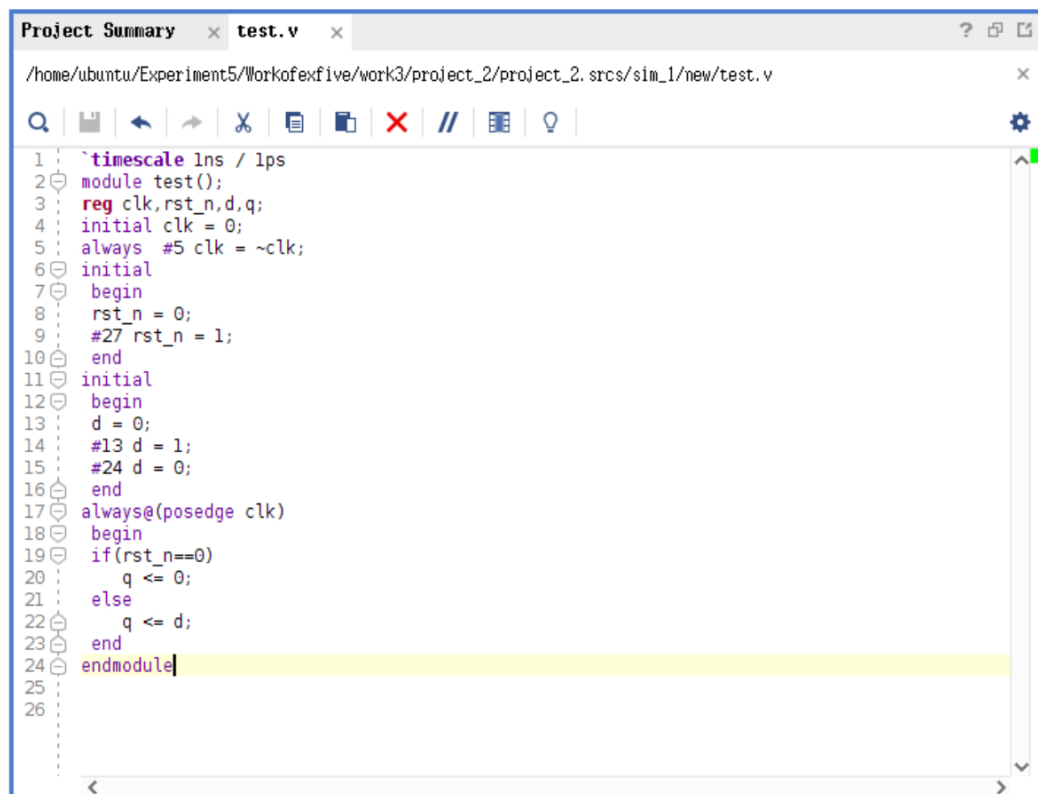
波形图如下:





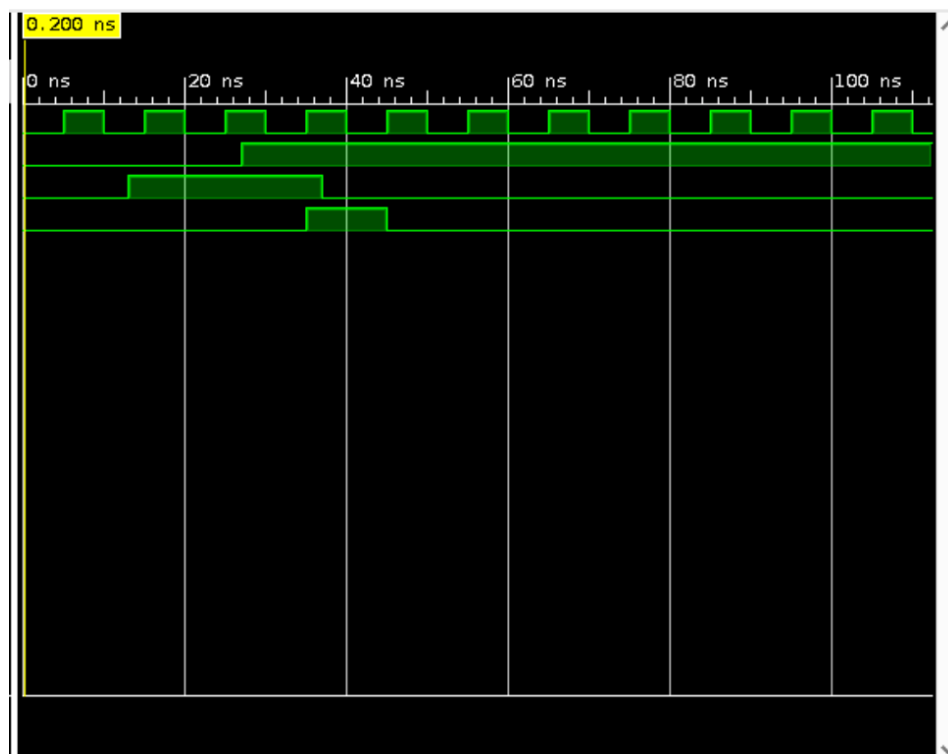
题目 3:

仿真文件如下:



```
1  `timescale 1ns / 1ps
2  module test();
3      reg clk,rst_n,d,q;
4      initial clk = 0;
5      always #5 clk = ~clk;
6      initial
7      begin
8          rst_n = 0;
9          #27 rst_n = 1;
10     end
11     initial
12     begin
13         d = 0;
14         #13 d = 1;
15         #24 d = 0;
16     end
17     always@(posedge clk)
18     begin
19         if(rst_n==0)
20             q <= 0;
21         else
22             q <= d;
23         end
24     endmodule
25
26
```

波形图如下:



题目 4:

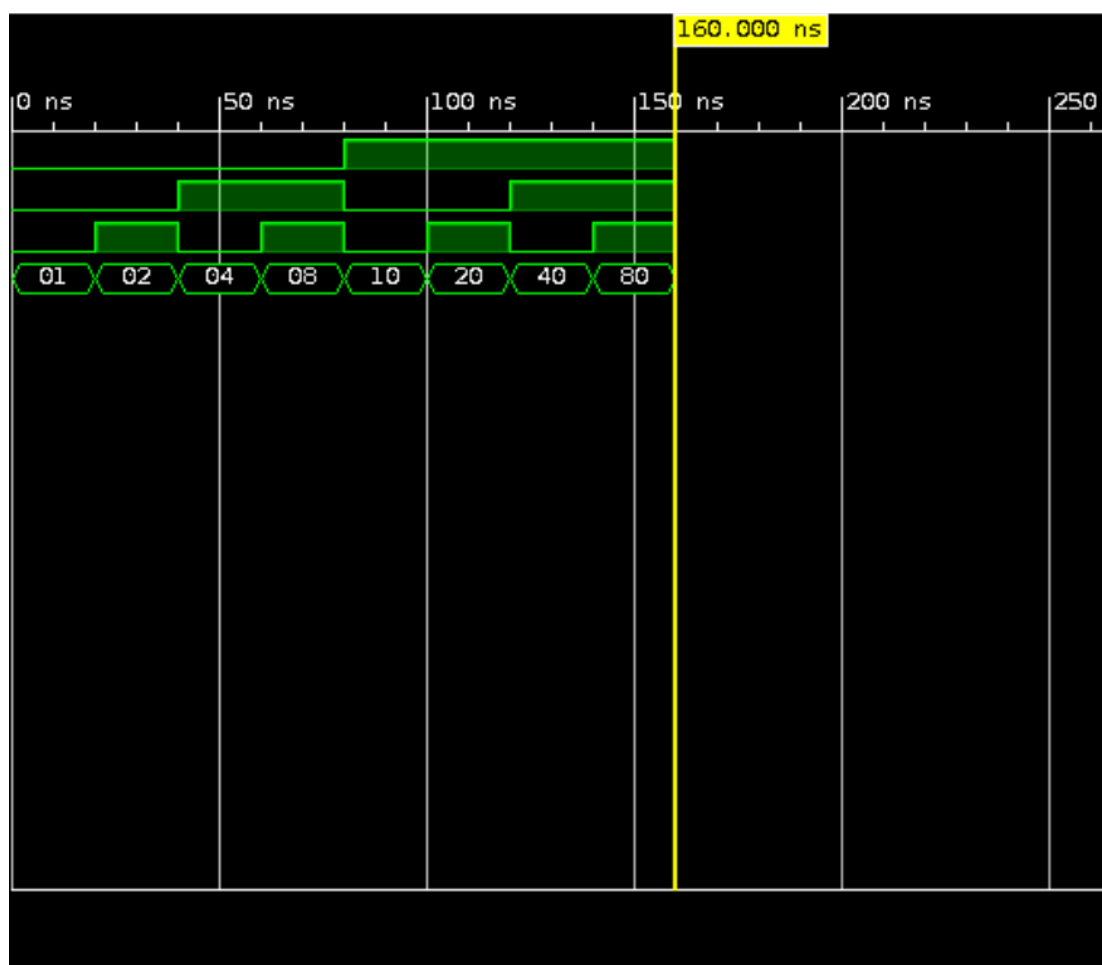
3-8 译码器的设计文件如下:

```
23 module Decoder3_8(  
24     input a1,a2,a3,  
25     output reg [7:0]out  
26 );  
27 always@(*)  
28 begin  
29     case({a1,a2,a3})  
30         3'b000: out = 8'b00000001;  
31         3'b001: out = 8'b00000010;  
32         3'b010: out = 8'b00000100;  
33         3'b011: out = 8'b00001000;  
34         3'b100: out = 8'b00010000;  
35         3'b101: out = 8'b00100000;  
36         3'b110: out = 8'b01000000;  
37         3'b111: out = 8'b10000000;  
38     endcase  
39 end  
40 endmodule  
41
```

3-8 译码器的仿真文件如下:

```
timescale 1ns / 1ps  
module test();  
    reg a1,a2,a3;  
    wire [7:0]out;  
    Decoder3_8 Decoder3_8test(.a1(a1),.a2(a2),.a3(a3),.out(out));  
    initial  
    begin  
        a1=0;a2=0;a3=0;  
        #20;  
        a1=0;a2=0;a3=1;  
        #20;  
        a1=0;a2=1;a3=0;  
        #20;  
        a1=0;a2=1;a3=1;  
        #20;  
        a1=1;a2=0;a3=0;  
        #20;  
        a1=1;a2=0;a3=1;  
        #20;  
        a1=1;a2=1;a3=0;  
        #20;  
        a1=1;a2=1;a3=1;  
        #20;  
        $stop;  
    end  
endmodule
```

波形图如下：



其中 out 为 16 进制表示法，可见符合 3-8 译码器功能，设计合理。

### 【总结与思考】

本次实验是本人第一次接触 vivado 软件并使用其进行仿真，在经过本次实验学习后，本人对 vivado 的功能有了初步的认识，同时在编写 Verilog 代码的过程中也进一步学习了更多的 Verilog 语法和关键字。