

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： FPGA 原理和 vivado 综合  
学生姓名： 牛庆源  
学生学号： PB21111733  
完成日期： 2022. 11. 17

### 【实验题目】

1. 请通过实验中给出的可编程逻辑单元、交叉互连矩阵及 IOB 电路图，实现如下代码，并将其输出到引脚 B 上。给出配置数据和电路截图。

```
module test(input clk,output reg a);  
always@(posedge clk)  
a <= a ^ 1'b1;  
endmodule
```

2. 实验中的开关和 LED 的对应关系是相反的，即最左侧的开关控制最右侧的 LED，最右侧的开关控制最左侧的 LED，请修改实验中给出的 XDC 文件，使开关和 LED 一一对应（最左侧的开关控制最左侧的 LED）。

3. 设计一个 30 位计数器，每个时钟周期加 1，用右侧的 8 个 LED 表示计数器的高 8 位，观察实际运行结果。将该计数器改成 32 位，将高 8 位输出到 LED，与前面的运行结果进行对比，分析结果及时钟信号在其中所起的作用。

### 【实验目的】

了解 FPGA 工作原理

了解 Verilog 文件和约束文件在 FPGA 开发中的作用

学会使用 Vivado 进行 FPGA 开发的完整流程

### 【实验环境】

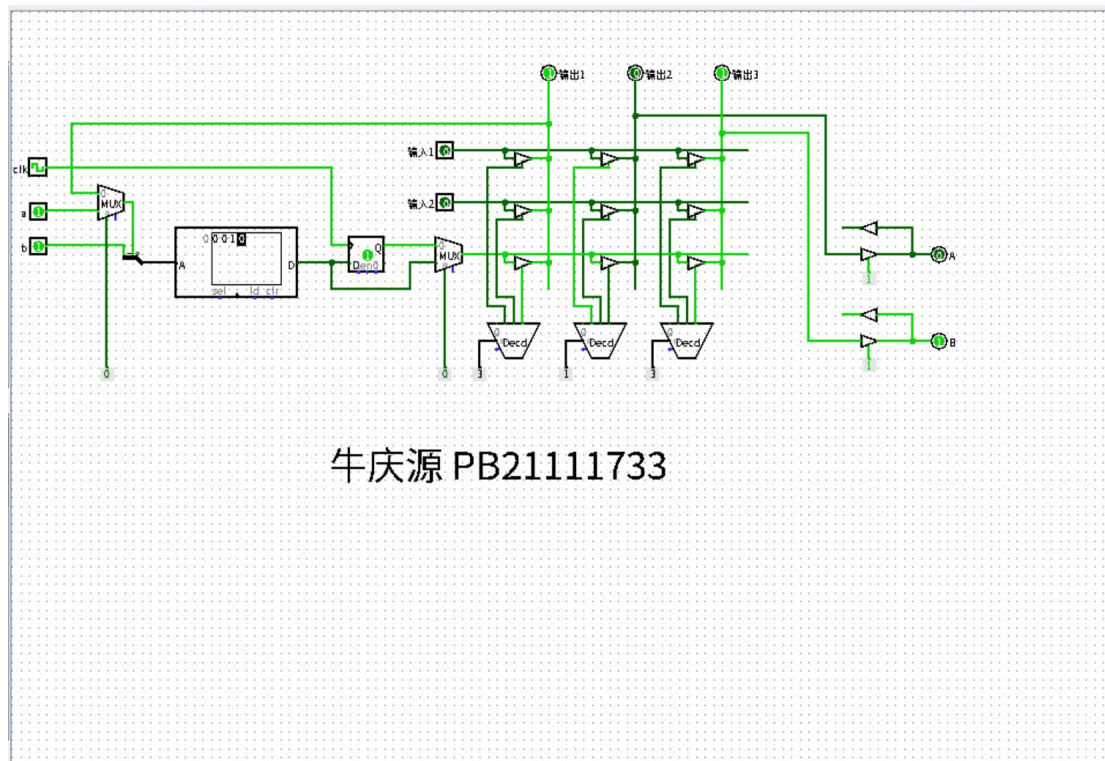
vlab.ustc.edu.cn

Vivado 软件

fpgaol.ustc.edu.cn

## 【手册操作】

Step4:



Step5:

控制文件

```
23 module test(  
24     input      clk,  
25     input      rst,  
26     input  [7:0] sw,  
27     output reg [7:0] led;  
28     always@(posedge clk or posedge rst)  
29     begin  
30         if(rst)  
31             led <= 8'haa;  
32         else  
33             led <= {sw[0], sw[1], sw[2], sw[3], sw[4], sw[5], sw[6], sw[7]};  
34     end  
35 endmodule  
36
```

约束文件

```

1  set_property -dict { PACKAGE_PIN E3      IOSTANDARD LVCMOS33 } [get_ports {clk}];
2  set_property -dict { PACKAGE_PIN B18     IOSTANDARD LVCMOS33 } [get_ports {rst}];
3  set_property -dict { PACKAGE_PIN C17     IOSTANDARD LVCMOS33 } [get_ports {led[0]}];
4  set_property -dict { PACKAGE_PIN D18     IOSTANDARD LVCMOS33 } [get_ports {led[1]}];
5  set_property -dict { PACKAGE_PIN E18     IOSTANDARD LVCMOS33 } [get_ports {led[2]}];
6  set_property -dict { PACKAGE_PIN G17     IOSTANDARD LVCMOS33 } [get_ports {led[3]}];
7  set_property -dict { PACKAGE_PIN D17     IOSTANDARD LVCMOS33 } [get_ports {led[4]}];
8  set_property -dict { PACKAGE_PIN E17     IOSTANDARD LVCMOS33 } [get_ports {led[5]}];
9  set_property -dict { PACKAGE_PIN F18     IOSTANDARD LVCMOS33 } [get_ports {led[6]}];
10 set_property -dict { PACKAGE_PIN G18     IOSTANDARD LVCMOS33 } [get_ports {led[7]}];
11
12 set_property -dict { PACKAGE_PIN D14     IOSTANDARD LVCMOS33 } [get_ports {sw[0]}];
13 set_property -dict { PACKAGE_PIN F16     IOSTANDARD LVCMOS33 } [get_ports {sw[1]}];
14 set_property -dict { PACKAGE_PIN G16     IOSTANDARD LVCMOS33 } [get_ports {sw[2]}];
15 set_property -dict { PACKAGE_PIN H14     IOSTANDARD LVCMOS33 } [get_ports {sw[3]}];
16 set_property -dict { PACKAGE_PIN E16     IOSTANDARD LVCMOS33 } [get_ports {sw[4]}];
17 set_property -dict { PACKAGE_PIN F13     IOSTANDARD LVCMOS33 } [get_ports {sw[5]}];
18 set_property -dict { PACKAGE_PIN G13     IOSTANDARD LVCMOS33 } [get_ports {sw[6]}];
19 set_property -dict { PACKAGE_PIN H16     IOSTANDARD LVCMOS33 } [get_ports {sw[7]}];

```

Step6:

烧写 FPGA

Bitstream File

Select file

example bitstream ▾

C:\fakepath\test.bit

Program!

Program success!

FPGA interface

uart show>

FPGAOL UART xterm.js 1.1

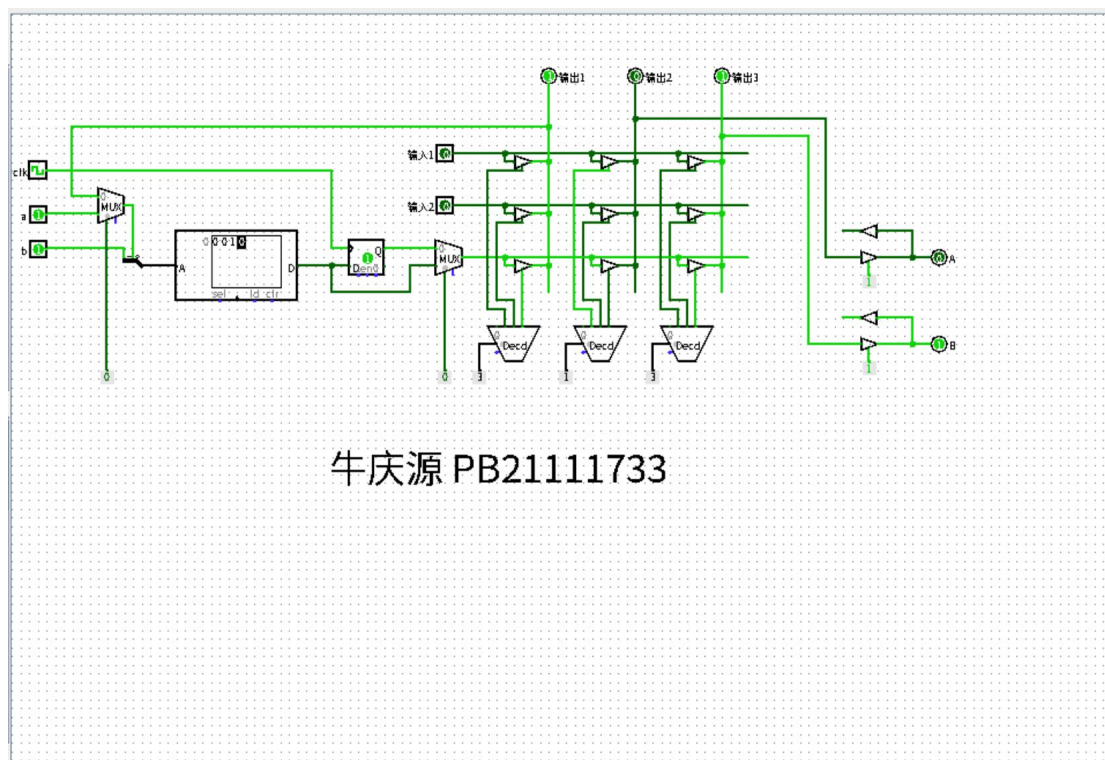
The screenshot shows a web-based FPGA programming interface. The top section, 'Bitstream File', includes a 'Select file' button, a dropdown menu showing 'example bitstream', a text input field with 'C:\fakepath\test.bit', and a 'Program!' button. Below this, a green bar indicates 'Program success!'. The bottom section, 'FPGA interface', features a hardware diagram of an XC7A100t-CSG324-1 FPGA. The diagram shows 8 LEDs (led7 to led0) and 8 switches (sw7 to sw0) connected to specific pins (G18, F18, E17, D17, G17, E18, D18, C17 on top; H16, G13, F13, E16, H14, G16, F16, D14 on bottom). To the right of the diagram is a terminal window titled 'uart show>' displaying 'FPGAOL UART xterm.js 1.1'.

【实验练习】

题目一

设置配置区域为 0 0 3 1 3，RAM 配置为 0010，a 和 b 输入为 1 1。

另 RAM 接入的 2bit 信号为 {b, a}, a 的信号由 0,1 变化则选择 RAM 的后两位中的一位输出, 经过选择器输入到交叉互连矩阵中, 作为输入。输出 1 为电路的输出, 输出 2 和输出 3 经过控制模块 IOB。输出 3 与输出 1 相同, 输出 2 与输出 1 不同。输出 1 回到 a 端, 与 a 信号处理后当做 RAM 接入的低位 bit 输入。得到电路图如下。



## 题目二

只需要将 step5 中的 sw 全部反接即可。

约束文件如下：（仅将 sw 的高位与低位一一调换即可）



Project Summary x lab06.2.v x lab06\_2\_con.xdc x

C:/Users/niu/lab06.2/lab06.2.srscs/constrs\_1/new/lab06\_2\_con.xdc

Q [Icons]

```
1 set_property -dict { PACKAGE_PIN E3 IOSTANDARD LVCMOS33 } [get_ports {c1k}}];
2 set_property -dict { PACKAGE_PIN B18 IOSTANDARD LVCMOS33 } [get_ports {rst}}];
3 set_property -dict { PACKAGE_PIN C17 IOSTANDARD LVCMOS33 } [get_ports {led[0}}];
4 set_property -dict { PACKAGE_PIN D18 IOSTANDARD LVCMOS33 } [get_ports {led[1}}];
5 set_property -dict { PACKAGE_PIN E18 IOSTANDARD LVCMOS33 } [get_ports {led[2}}];
6 set_property -dict { PACKAGE_PIN G17 IOSTANDARD LVCMOS33 } [get_ports {led[3}}];
7 set_property -dict { PACKAGE_PIN D17 IOSTANDARD LVCMOS33 } [get_ports {led[4}}];
8 set_property -dict { PACKAGE_PIN E17 IOSTANDARD LVCMOS33 } [get_ports {led[5}}];
9 set_property -dict { PACKAGE_PIN F18 IOSTANDARD LVCMOS33 } [get_ports {led[6}}];
10 set_property -dict { PACKAGE_PIN G18 IOSTANDARD LVCMOS33 } [get_ports {led[7}}];
11
12 set_property -dict { PACKAGE_PIN D14 IOSTANDARD LVCMOS33 } [get_ports {sw[7}}];
13 set_property -dict { PACKAGE_PIN F16 IOSTANDARD LVCMOS33 } [get_ports {sw[6}}];
14 set_property -dict { PACKAGE_PIN G16 IOSTANDARD LVCMOS33 } [get_ports {sw[5}}];
15 set_property -dict { PACKAGE_PIN H14 IOSTANDARD LVCMOS33 } [get_ports {sw[4}}];
16 set_property -dict { PACKAGE_PIN E16 IOSTANDARD LVCMOS33 } [get_ports {sw[3}}];
17 set_property -dict { PACKAGE_PIN F13 IOSTANDARD LVCMOS33 } [get_ports {sw[2}}];
18 set_property -dict { PACKAGE_PIN G13 IOSTANDARD LVCMOS33 } [get_ports {sw[1}}];
19 set_property -dict { PACKAGE_PIN H16 IOSTANDARD LVCMOS33 } [get_ports {sw[0}}];
```

烧写 FPGA 如下

**Bitstream File**

Select file example bitstream C:\fakepath\lab06\_2.bit Program!

Program success!

**FPGA interface** uart show>

led7 led6 led5 led4 led3 led2 led1 led0

G18 F18 E17 D17 G17 E18 D18 C17

FPGA  
XC7A100t-CSG324-1

H16 G13 F13 E16 H14 G16 F16 D14

sw7 sw6 sw5 sw4 sw3 sw2 sw1 sw0

FPGAOL UART xterm.js 1.1

## 题目三

要设计 30 位计数器，使用 30bits 的信号作为计数器，时钟信号用于触发计数器所计数的增加，每经过一个时钟周期，计数器信号增加，并将高八位赋予 led 的八位信号。设计 rst 同步复位信号，rst 有效时复位为 0。控制文件如下：

```
23 module lab06_3(  
24     input clk, rst,  
25     output reg [7:0] led);  
26     reg [29:0] r;  
27     always@(posedge clk)  
28     begin  
29         if(rst)  
30         begin  
31             r <= 30'b0;  
32             led <= 8'h0;  
33         end  
34         else  
35         begin  
36             r <= r + 30'b1;  
37             led <= {r[29], r[28], r[27], r[26], r[25], r[24], r[23], r[22]};  
38         end  
39     end  
40 endmodule
```

约束文件如下，只需要将 rst 信号，led 信号与管脚一一对应即可：

```
1 set_property -dict { PACKAGE_PIN E3 IOSTANDARD LVCMOS33 } [get_ports {clk}];  
2 set_property -dict { PACKAGE_PIN B18 IOSTANDARD LVCMOS33 } [get_ports {rst}];  
3 set_property -dict { PACKAGE_PIN G18 IOSTANDARD LVCMOS33 } [get_ports {led[0]}};  
4 set_property -dict { PACKAGE_PIN F18 IOSTANDARD LVCMOS33 } [get_ports {led[1]}};  
5 set_property -dict { PACKAGE_PIN E17 IOSTANDARD LVCMOS33 } [get_ports {led[2]}};  
6 set_property -dict { PACKAGE_PIN D17 IOSTANDARD LVCMOS33 } [get_ports {led[3]}};  
7 set_property -dict { PACKAGE_PIN G17 IOSTANDARD LVCMOS33 } [get_ports {led[4]}};  
8 set_property -dict { PACKAGE_PIN E18 IOSTANDARD LVCMOS33 } [get_ports {led[5]}};  
9 set_property -dict { PACKAGE_PIN D18 IOSTANDARD LVCMOS33 } [get_ports {led[6]}};  
10 set_property -dict { PACKAGE_PIN C17 IOSTANDARD LVCMOS33 } [get_ports {led[7]}};
```

32 位计数器只需要将计数的信号改为 32bits,led 取高八位信号，其余相同即可，控制文件如下：

```

23 module lab06_3_2(
24     input clk, rst,
25     output reg [7:0] led);
26     reg [31:0] r;
27     always@(posedge clk)
28     begin
29         if(rst)
30             begin
31                 r <= 32'b0;
32                 led <= 8'h0;
33             end
34         else
35             begin
36                 r <= r + 32'b1;
37                 led <= {r[31], r[30], r[29], r[28], r[27], r[26], r[25], r[24]};
38             end
39     end
40 endmodule
41

```

约束文件与之前文件相同。

烧写 FPGA 发现 32 位计数器中 LED 闪烁的频率低于 30 位计数器。

时钟信号作用为控制计数器增加。

### 【总结与思考】

1. 首先了解了FPGA的工作原理以及Verilog 文件和约束文件在FPGA开发中的作用，其次学会了如何编写约束文件，如何使控制文件和约束文件联动。
2. 难易程度为中等，出现了不少的 bug，一一解决。
3. 任务量中等，完成时间花费中等。
4. 没有较好的改进建议，实验手册详尽，不过希望约束文件可以以文字形式给出例子，可以减少我们重复的工作量。