

实验四 使用 FPGA 实现时序逻辑电路

第 1 步：5 分

实验目的：（简要写出）

- (1) 掌握时序逻辑电路基本元件的 verilog 实现方法。
- (2) 掌握编写简单仿真文件对设计文件进行功能仿真的方法。
- (3) 掌握使用 verilog 语言基于计数器结构描述时序逻辑电路的方法。

第 2 步：5 分

实验原理：（简要写出）

使用 FPGA 实现含异步复位功能的 D 触发器，功能表如图 4-1 所示。使用 EGO1 板卡上的 P17 管脚的 100MHz 时钟信号作为时钟，拨码开关 SW0 作为输入，拨码开关 SW1 作为复位信号，LED 灯 LD2_0 作为输出。

真值表：

reset	CLK	D	Q*
1	x	x	0
0	↑	1	1
0	↑	0	0

使用阻塞赋值“=”时，赋值语句立即把当前值赋给变量；使用非阻塞赋值“<=”时，赋值语句要等到 always 块结束后，才完成对变量的赋值操作。

always 块中的语句是顺序执行，而 always 块之间以及与 assign 语句之间是并发执行。

若使用其余引脚接入时钟信号（比如使用按键 S1 产生时钟信号），则可能需要在约束文件里添加：set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets clk_IBUF]。因为芯片有部分引脚为专用的时钟引脚，如果不采用专用的时钟引脚引入时钟 vivado 就会报错。通过这句约束可以将这个错误降级为 WARNING，屏蔽 vivado 对此规则的检测，从而实现从普通 IO 引脚引入时钟。具体设计时请谨慎使用。

第 3 步：8 分

预习题（拍照或截图，可上传多张图片）

实验四 使用 FPGA 实现时序逻辑电路

专业-班级: 自动化1班学号: 210320111姓名: 吕家昊

实验检查记录表:

预习	一	二	三	必做完成时间	四 (选做)

1. 填空: EGO1 实验板搭载了 100 M Hz 的时钟芯片, 时钟信号接入了 FPGA 的全局时钟引脚 P17。
2. 填空: 阻塞赋值符号“三”, 赋值语句 立即 完成对变量的赋值; 非阻塞赋值符号“≤”, 赋值语句 等到 always 块结束后 完成对变量的赋值。
3. 填空: always 块中的语句是 顺序 执行, 而 always 块之间以及与 assign 语句之间是 并发 执行。
4. 问答: 请阅读表 4-3 的仿真文件, 简述文件中包含几个语句块, 每个语句块分别是什么作用?
diff-reset UUT ... 实例化被测模块
initial begin clk=1'b0; ... end 用于初始化输入, 并产生 reset 信号
always begin clk=1'b0; #5 clk=1'b1; #5; end 产生 5ns 翻转一次的 clk 信号
always begin d=1'b0; #8 d=1'b1; #8; end 产生 8ns 翻转一次的 d 信号
5. 问答: 如果使用按键 S1 产生时钟信号, 则需要在约束文件中添加什么语句? 其作用是什么?
set-property CLOCK_DEDICATED_ROUTE FALSE [get_nets clk_IBUF]
作用: 屏蔽 vivado 对不采用专用时钟引脚引入时钟的报错。
6. 问答: always@(posedge clk) 语句是什么含义?
当 clk 发生变化时, 执行 always 内语句。
产生上升沿

第 5 步: 20 分

4.5.1 设计含异步复位和同步使能的 D 触发器

简述实验现象。并对仿真波形进行分析。

截图或拍照：Verilog 源代码、约束文件、仿真文件源代码、仿真波形（可上传多张图片）。

实验现象：

当 SW1 拨至高电平时，LED 灯始终为灭。

SW1 为低电平：

若 SW2 为低电平，改变 SW0 对灯亮灭无影响。

若 SW2 为高电平，SW0 为高电平时灯亮，SW0 为低电平时灯灭。

仿真分析：

clk 每 10ns 产生一次上升沿，d 每 8ns 翻转一次。

en=0, reset=1 时，q 被置 0。

en=0, reset=0 时，由于前 50ms 内 q 被置 0，因此 q 保持为 0。

en=1, reset=0 时，clk 上升沿到达时，q 状态与此时 d 相同。

en=1, reset=1 时，q 被置 0。

```
module d_trigger(  
    input reset, clk, en, d,  
    output q  
);  
    reg q_reg;  
always@(posedge clk, posedge reset)  
begin  
    if(reset) q_reg <= 0;  
    else if(en) q_reg = d;  
end  
assign q = q_reg;  
endmodule
```

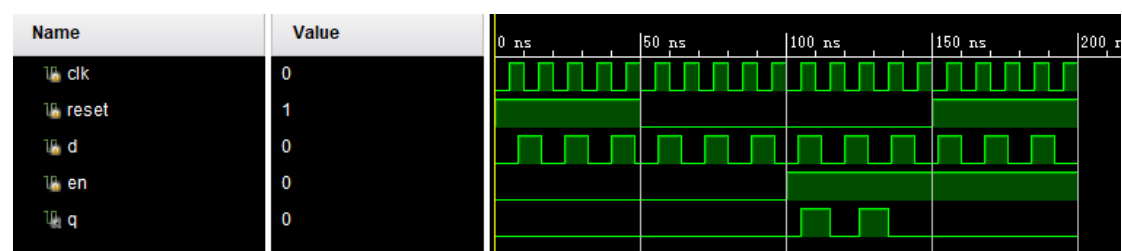
```
set_property PACKAGE_PIN R11 [get_ports clk];  
set_property PACKAGE_PIN N4 [get_ports reset];  
set_property PACKAGE_PIN R1 [get_ports d];  
set_property PACKAGE_PIN M4 [get_ports en];  
set_property PACKAGE_PIN K2 [get_ports q];  
set_property IOSTANDARD LVCOS33 [get_ports clk];  
set_property IOSTANDARD LVCOS33 [get_ports reset];  
set_property IOSTANDARD LVCOS33 [get_ports d];  
set_property IOSTANDARD LVCOS33 [get_ports en];  
set_property IOSTANDARD LVCOS33 [get_ports q];  
set_property BITSTREAM.CONFIG.UNUSEDPIN PULLNONE [current_design];  
set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets clk_IBUF];
```

```

module simu(
);
    reg clk;
    reg reset;
    reg d;
    reg en;
    wire q;
    d_trigger trig(.clk(clk), .reset(reset), .d(d), .en(en), .q(q));

    initial begin
        clk = 0;
        d = 0;
        {en, reset} = 2'b01;
        #50 {en, reset} = 2'b00;
        #50 {en, reset} = 2'b10;
        #50 {en, reset} = 2'b11;
        #50 $finish; //50ns 后结束仿真
    end
    always begin
        clk = 1'b0;
        #5 clk = 1'b1;
        #5;
    end
    always begin
        d = 1'b0;
        #8 d = 1'b1;
        #8;
    end
endmodule

```



第 6 步： 20 分

4.5.2 使用 FPGA 以 1Hz 频率实现十进制计数器并通过 LED 进行输出

简述实验现象。

截图或拍照：Verilog 源代码、约束文件（可上传多张图片）。

按键保持按下时，led 灯一次表示二进制下 0~9；
按键松开，led 灯显示不变。

```
module counter_LED(  
    input clk,  
    input enable,  
    output [3:0] out  
);  
    reg[26:0] divclk;  
    reg[3:0] reg4;  
    reg cout_1s;  
    //对 100M 时钟分频产生 1Hz 时钟信号  
    always@(posedge clk)  
    begin  
        if (divclk< 100000000-1)  
            begin  
                divclk<=divclk+1;  
                cout_1s<=1'b0;  
            end  
        else  
            begin  
                divclk<=0;  
                cout_1s<=1'b1;  
            end  
        end  
    end  
    //1Hz 时钟频率的十进制计数器  
    always@(posedge clk)  
    begin  
        if (enable && cout_1s)  
            begin  
                if (reg4<9) reg4<=reg4+1;  
                else reg4<=0;  
            end  
        else  
            reg4<=reg4;  
        end  
    end  
    //reg 值赋值给输出端口
```

```

1  set_property PACKAGE_PIN P17 [get_ports clk];
2  set_property PACKAGE_PIN R11 [get_ports enable];
3  set_property PACKAGE_PIN K2 [get_ports out[0]];
4  set_property PACKAGE_PIN J2 [get_ports out[1]];
5  set_property PACKAGE_PIN J3 [get_ports out[2]];
6  set_property PACKAGE_PIN H4 [get_ports out[3]];
7  set_property IOSTANDARD LVCMOS33 [get_ports clk];
8  set_property IOSTANDARD LVCMOS33 [get_ports enable];
9  set_property IOSTANDARD LVCMOS33 [get_ports out[0]];
10 set_property IOSTANDARD LVCMOS33 [get_ports out[1]];
11 set_property IOSTANDARD LVCMOS33 [get_ports out[2]];
12 set_property IOSTANDARD LVCMOS33 [get_ports out[3]];
13 set_property BITSTREAM.CONFIG.UNUSEDPIN PULLNONE [current_design];

```

第 7 步： 20 分

4.5.3 设计十字路口交通灯系统

简述实验现象。

截图或拍照：Verilog 源代码、约束文件（可上传多张图片）。

led 灯每 8s 执行一次如下循环：

- 南北红灯、东西绿灯亮，其它灯灭（3s）
- 南北红灯、东西黄灯亮，其它灯灭（1s）
- 南北绿灯、东西红灯亮，其它灯灭（3s）
- 南北黄灯、东西红灯亮，其它灯灭（1s）

```

22
23 module Light(
24     input clk,
25     output[2:0] ns,
26     output[2:0] we
27 );
28     reg[26:0] divclk;
29     reg[2:0] cnt;
30     reg[2:0] ns_reg;
31     reg[2:0] we_reg;
32     //对 100M 时钟分频产生 1Hz 时钟信号
33     always@(posedge clk)
34     begin
35         if (divclk < 100000000-1)
36         begin
37             divclk <= divclk + 1;
38         end
39         else
40         begin
41             divclk <= 0;
42             if(cnt == 3'b111) cnt <= 0;
43             else cnt <= cnt + 1;
44             case(cnt)
45                 0: {ns_reg, we_reg} = 6'b100001;
46                 1: {ns_reg, we_reg} = 6'b100001;
47                 2: {ns_reg, we_reg} = 6'b100001;
48                 3: {ns_reg, we_reg} = 6'b100010;
49                 4: {ns_reg, we_reg} = 6'b001100;
50                 5: {ns_reg, we_reg} = 6'b001100;
51                 6: {ns_reg, we_reg} = 6'b001100;
52                 7: {ns_reg, we_reg} = 6'b010100;
53                 default: {ns_reg, we_reg} = 6'b111111;
54             endcase
55         end
56     end
57     assign ns = ns_reg;
58     assign we = we_reg;
59 endmodule
60

```

```
1  set_property PACKAGE_PIN P17 [get_ports clk];
2  set_property PACKAGE_PIN J3 [get_ports ns[2]];
3  set_property PACKAGE_PIN J2 [get_ports ns[1]];
4  set_property PACKAGE_PIN K2 [get_ports ns[0]];
5  set_property PACKAGE_PIN L1 [get_ports we[2]];
6  set_property PACKAGE_PIN M1 [get_ports we[1]];
7  set_property PACKAGE_PIN K3 [get_ports we[0]];
8  set_property IOSTANDARD LVCOS33 [get_ports clk];
9  set_property IOSTANDARD LVCOS33 [get_ports ns[2]];
10 set_property IOSTANDARD LVCOS33 [get_ports ns[1]];
11 set_property IOSTANDARD LVCOS33 [get_ports ns[0]];
12 set_property IOSTANDARD LVCOS33 [get_ports we[2]];
13 set_property IOSTANDARD LVCOS33 [get_ports we[1]];
14 set_property IOSTANDARD LVCOS33 [get_ports we[0]];
15 set_property BITSTREAM.CONFIG.UNUSEDPIN PULLNONE [current_design];
16
```

第 8 步： 10 分

4.5.4* （选做）二进制码转换十进制并用数码管显示

简述实验现象。

截图或拍照：Verilog 源代码、约束文件（可上传多张图片）。


```

23 module BintOct(
24     input key,
25     input clk,
26     input[3:0] din,
27     output[1:0] cs,
28     output[6:0] out5, out1
29 );
30
31     reg[6:0] out5_reg = 0, out1_reg = 0;
32     reg cs_reg = 0;
33     always@(posedge key)
34     begin
35         case(din)
36             0: {out1_reg, out5_reg} <= 14'b11111101111110;
37             1: {out1_reg, out5_reg} <= 14'b11111100110000;
38             2: {out1_reg, out5_reg} <= 14'b11111101101101;
39             3: {out1_reg, out5_reg} <= 14'b11111101111001;
40             4: {out1_reg, out5_reg} <= 14'b11111100110011;
41             5: {out1_reg, out5_reg} <= 14'b11111101011011;
42             6: {out1_reg, out5_reg} <= 14'b11111101011111;
43             7: {out1_reg, out5_reg} <= 14'b11111101110000;
44             8: {out1_reg, out5_reg} <= 14'b11111101111111;
45             9: {out1_reg, out5_reg} <= 14'b11111101111011;
46             10: {out1_reg, out5_reg} <= 14'b01100001111110;
47             11: {out1_reg, out5_reg} <= 14'b01100000110000;
48             12: {out1_reg, out5_reg} <= 14'b01100001101101;
49             13: {out1_reg, out5_reg} <= 14'b01100001111001;
50             14: {out1_reg, out5_reg} <= 14'b01100000110011;
51             15: {out1_reg, out5_reg} <= 14'b01100001011011;
52             default: {out1_reg, out5_reg} <= 14'b00000010000001;
53         endcase
54     end
55
56     always@(posedge clk)
57     begin
58         if(cs_reg == 0) cs_reg <= 1;
59         else cs_reg <= 0;
60     end
61
62     assign cs = {cs_reg, !cs_reg};
63     assign out5 = out5_reg;
64     assign out1 = out1_reg;
65 endmodule
66

```

```

1  set_property PACKAGE_PIN R11 [get_ports key];
2  set_property PACKAGE_PIN P17 [get_ports clk];
3  set_property PACKAGE_PIN R2 [get_ports din[3]];
4  set_property PACKAGE_PIN M4 [get_ports din[2]];
5  set_property PACKAGE_PIN N4 [get_ports din[1]];
6  set_property PACKAGE_PIN R1 [get_ports din[0]];
7  set_property PACKAGE_PIN D2 [get_ports out5[0]];
8  set_property PACKAGE_PIN E2 [get_ports out5[1]];
9  set_property PACKAGE_PIN F3 [get_ports out5[2]];
10 set_property PACKAGE_PIN F4 [get_ports out5[3]];
11 set_property PACKAGE_PIN D3 [get_ports out5[4]];
12 set_property PACKAGE_PIN E3 [get_ports out5[5]];
13 set_property PACKAGE_PIN D4 [get_ports out5[6]];
14 set_property PACKAGE_PIN B2 [get_ports out1[0]];
15 set_property PACKAGE_PIN B3 [get_ports out1[1]];
16 set_property PACKAGE_PIN A1 [get_ports out1[2]];
17 set_property PACKAGE_PIN B1 [get_ports out1[3]];
18 set_property PACKAGE_PIN A3 [get_ports out1[4]];
19 set_property PACKAGE_PIN A4 [get_ports out1[5]];
20 set_property PACKAGE_PIN B4 [get_ports out1[6]];
21 set_property PACKAGE_PIN G1 [get_ports cs[1]];
22 set_property PACKAGE_PIN G2 [get_ports cs[0]];
23
24 set_property IOSTANDARD LVCMOS33 [get_ports key];
25 set_property IOSTANDARD LVCMOS33 [get_ports clk];
E 26 set_property IOSTANDARD LVCMOS33 [get_ports din[3]];
E 27 set_property IOSTANDARD LVCMOS33 [get_ports din[2]];
E 28 set_property IOSTANDARD LVCMOS33 [get_ports din[1]];
E 29 set_property IOSTANDARD LVCMOS33 [get_ports din[0]];
E 30 set_property IOSTANDARD LVCMOS33 [get_ports out5[6]];
E 31 set_property IOSTANDARD LVCMOS33 [get_ports out5[5]];
E 32 set_property IOSTANDARD LVCMOS33 [get_ports out5[4]];
E 33 set_property IOSTANDARD LVCMOS33 [get_ports out5[3]];
E 34 set_property IOSTANDARD LVCMOS33 [get_ports out5[2]];
E 35 set_property IOSTANDARD LVCMOS33 [get_ports out5[1]];
E 36 set_property IOSTANDARD LVCMOS33 [get_ports out5[0]];
E 37 set_property IOSTANDARD LVCMOS33 [get_ports out1[6]];
E 38 set_property IOSTANDARD LVCMOS33 [get_ports out1[5]];
E 39 set_property IOSTANDARD LVCMOS33 [get_ports out1[4]];
40 set_property IOSTANDARD LVCMOS33 [get_ports out1[3]];
41 set_property IOSTANDARD LVCMOS33 [get_ports out1[2]];
42 set_property IOSTANDARD LVCMOS33 [get_ports out1[1]];
43 set_property IOSTANDARD LVCMOS33 [get_ports out1[0]];
44 set_property IOSTANDARD LVCMOS33 [get_ports cs[1]];
45 set_property IOSTANDARD LVCMOS33 [get_ports cs[0]];
46
47 set_property BITSTREAM.CONFIG.UNUSEDPIN PULLNONE [current_design];
48 set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets key_IBUF]

```

```

23 module BintOct(
24     input key,
25     input clk,
26     input[3:0] din,
27     output[1:0] cs,
28     output[6:0] out5
29 );
30
31     reg[6:0] out5_reg = 0, out1_reg = 0; //out1_reg out5_reg实际对应DK5, DK6
32     reg cs_reg = 0;
33     reg[17:0] divclk = 0;
34     always@(posedge key)
35     begin
36         case(din)
37             0: {out1_reg, out5_reg} <= 14'b1111101111110;
38             1: {out1_reg, out5_reg} <= 14'b1111100110000;
39             2: {out1_reg, out5_reg} <= 14'b1111101101101;
40             3: {out1_reg, out5_reg} <= 14'b1111101111001;
41             4: {out1_reg, out5_reg} <= 14'b1111100110011;
42             5: {out1_reg, out5_reg} <= 14'b1111101011011;
43             6: {out1_reg, out5_reg} <= 14'b1111101011111;
44             7: {out1_reg, out5_reg} <= 14'b1111101110000;
45             8: {out1_reg, out5_reg} <= 14'b1111101111111;
46             9: {out1_reg, out5_reg} <= 14'b1111101111011;
47             10: {out1_reg, out5_reg} <= 14'b0110000111110;
48             11: {out1_reg, out5_reg} <= 14'b01100000110000;
49             12: {out1_reg, out5_reg} <= 14'b01100001101101;
50             13: {out1_reg, out5_reg} <= 14'b01100001111001;
51             14: {out1_reg, out5_reg} <= 14'b01100000110011;
52             15: {out1_reg, out5_reg} <= 14'b01100001011011;
53             default: {out1_reg, out5_reg} <= 14'b00000010000001;
54         endcase
55     end
56
57     always@(posedge clk) //降低片选更改频率, 防止数码管来不及切换导致被覆盖
58     begin
59         if(divclk < 100000 - 1)
60         begin
61             divclk = divclk + 1;
62         end
63         else
64         begin
65             divclk = 0;
66             if(cs_reg == 0) cs_reg <= 1;
67             else cs_reg <= 0;
68         end
69     end
70
71     assign cs = cs_reg ? 2'b10 : 2'b01;
72     assign out5 = cs_reg ? out1_reg : out5_reg;
73 endmodule

```



```

1  set_property PACKAGE_PIN R11 [get_ports key];
2  set_property PACKAGE_PIN P17 [get_ports clk];
3  set_property PACKAGE_PIN R2 [get_ports din[3]];
4  set_property PACKAGE_PIN M4 [get_ports din[2]];
5  set_property PACKAGE_PIN N4 [get_ports din[1]];
6  set_property PACKAGE_PIN R1 [get_ports din[0]];
7  set_property PACKAGE_PIN D2 [get_ports out5[0]];
8  set_property PACKAGE_PIN E2 [get_ports out5[1]];
9  set_property PACKAGE_PIN F3 [get_ports out5[2]];
10 set_property PACKAGE_PIN F4 [get_ports out5[3]];
11 set_property PACKAGE_PIN D3 [get_ports out5[4]];
12 set_property PACKAGE_PIN E3 [get_ports out5[5]];
13 set_property PACKAGE_PIN D4 [get_ports out5[6]];
14 set_property PACKAGE_PIN G1 [get_ports cs[1]];
15 set_property PACKAGE_PIN F1 [get_ports cs[0]];
16
17 set_property IOSTANDARD LVCMOS33 [get_ports key];
18 set_property IOSTANDARD LVCMOS33 [get_ports clk];
19 set_property IOSTANDARD LVCMOS33 [get_ports din[3]];
20 set_property IOSTANDARD LVCMOS33 [get_ports din[2]];
21 set_property IOSTANDARD LVCMOS33 [get_ports din[1]];
22 set_property IOSTANDARD LVCMOS33 [get_ports din[0]];
23 set_property IOSTANDARD LVCMOS33 [get_ports out5[6]];
24 set_property IOSTANDARD LVCMOS33 [get_ports out5[5]];
25 set_property IOSTANDARD LVCMOS33 [get_ports out5[4]];
26 set_property IOSTANDARD LVCMOS33 [get_ports out5[3]];
27 set_property IOSTANDARD LVCMOS33 [get_ports out5[2]];
28 set_property IOSTANDARD LVCMOS33 [get_ports out5[1]];
29 set_property IOSTANDARD LVCMOS33 [get_ports out5[0]];
30 set_property IOSTANDARD LVCMOS33 [get_ports cs[1]];
31 set_property IOSTANDARD LVCMOS33 [get_ports cs[0]];
32
33 set_property BITSTREAM.CONFIG.UNUSEDPIN PULLNONE [current_design];
34 set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets key_IBUF]

```

实验思考

第 1 题： 20 分

(1) 如果 4.5.3 中的时钟使用实验板上的 S1 开关输入。输出端是否会出现因为输入抖动而无法正常工作现象？如何解决？
输入端会因输入抖动而无法正常工作。

解决方式：

- (1) 采用消抖电路，采用较低的时钟频率，若连续几个周期内检测到高电平，视为按键按下。
- (2) 利用时钟，当检测到高电平时开始计时，若一段时间（10~20ms）后仍为高电平，视

为按键按下。

第 2 题： 1 分

(2) 实验过程中以及在线平台使用中遇到哪些问题？实验体会与建议？

在数码管显示时，若希望 DK1~4 或 DK5~8 显示不同内容，片选改变频率不应过高，否则数码管来不及切换导致输出被覆盖（见下图）。

解决此问题可将片选降频（实验中使用 1kHz）

