Lab7

姓名 学号

孙昊哲 PB20000277

实验目的

- 熟悉 FPGAOL
- 在线实验平台结构及使用
- 掌握 FPGA 开发各关键环节
- 学会使用 IP 核（知识产权核）

实验平台

Vivado2019

Windows11

T1

使用下表的方式对ROM进行初始化

Num	Bin
0	11111100
1	01100000
2	11011010
3	11110010
4	01100110
5	10110110
6	10111110
7	11100000
8	11111110
9	11110110
A	11101110
B	00111110
C	10011100
D	01111010
E	10011110

Num	Bin
F	10001110

那么我们的代码将会很简单为：

```
`timescale 1ns / 1ps

module T1(
input [3:0]sw,
output [7:0]led);
dist_mem_gen_0 dist_mem_gen_0(sw, led);
endmodule
```

T2

本题中采用分时复用的方式，由于FPGAOL上的管脚数量有限，我们对数码管进行了一定的简化：在使能方面，不再通过8个管脚分别使能8个数位，而是仅使能由`AN[2:0]`所表示的二进制数所对应的数位；在显示的数字方面，不再通过`SEG`信号独立控制每个段（segment），而是直接显示`D[3:0]`形成的16进制数。例如，若`AN=3'b010, D=4'b1010`，则在下标为2的数位上显示`A`。

和现实中的数码管一样，我们建议的扫描频率为50Hz，也就是说，若要驱动8个数码管，需要一个400Hz的时钟。[1]

因此我们采用如下代码的方式来实现显示两个十六进制数字：

```
reg [32:0] hexplay_cnt;
always@(posedge clk)
begin
    if (hexplay_cnt >= (2000000 / 8))
        hexplay_cnt <= 0;
    else
        hexplay_cnt <= hexplay_cnt + 1;
end

always@(posedge clk)
begin
    if(hexplay_cnt==0)
    begin
        if(an==3'b000)
            an<=3'b001;
        else
            an<=3'b000;
    end
end
```

完整代码如下：

```
`timescale 1ns / 1ps

module T2(
input [7:0]sw,
input clk,
```

```
output reg [2:0]an,
output reg [3:0]hexplay_data);

reg [32:0] hexplay_cnt;
always@(posedge clk)
begin
    if (hexplay_cnt >= (2000000 / 8)) //400Hz的扫描频率
        hexplay_cnt <= 0;
    else
        hexplay_cnt <= hexplay_cnt + 1;
end

always@(posedge clk)
begin
    if(hexplay_cnt==0)
    begin
        if(an==3'b000)
            an<=3'b001;
        else
            an<=3'b000;
    end
end

always@(posedge clk)
begin
    if(an==3'b000)
        hexplay_data<=sw[3:0];
    else
        hexplay_data<=sw[7:4];
end
endmodule
```

T3

在本题中data1,2,3,4依次表示十分之一秒，秒的个位，秒的十位，分钟，所以我们只需要处理好进位所需要的条件即可：

```
always@(posedge clk or posedge rst)
begin
    if(rst)
    begin
        data1 <= 4;
        data2 <= 3;
        data3 <= 2;
        data4 <= 1;
    end
    else if(cnt2 == 0)
    begin
        if(data1 == 9)
        begin
            data1 <= 0;
            if(data2 == 9)
            begin
                data2 <= 0;
                if(data3 == 5)
```

```
                begin
                    data3 <= 0;
                    if(data4 == 15)
                        data4 <= 0;
                    else
                        data4 <= data4 + 1;
                end
                else
                    data3 <= data3 + 1;
            end
            else
                data2 <= data2 + 1;
        end
        else
            data1 <= data1 + 1;
    end
    else
    begin
        data1 <= data1;
        data2 <= data2;
        data3 <= data3;
        data4 <= data4;
    end
end
```

完整代码如下：

```
`timescale 1ns / 1ps

module T3(
input clk, rst,
output reg [2:0]an,
output reg [3:0]data);

reg [31:0] cnt1;
reg [31:0] cnt2;
reg [3:0] data1;
reg [3:0] data2;
reg [3:0] data3;
reg [3:0] data4;
always@(posedge clk)
begin
    if (cnt1 >= (2000000 / 8)) //400Hz
        cnt1 <= 0;
    else
        cnt1 <= cnt1 + 1;
end

always@(posedge clk)
begin
    if (cnt2 >= (10000000)) //10Hz
        cnt2 <= 0;
    else
        cnt2 <= cnt2 + 1;
end
```

```
always@(posedge clk)
begin
    if(cnt1 == 0)
    begin
        if(an == 3'b011)
            an <= 3'b000;
        else
            an <= an + 3'b001;
    end
end

always@(posedge clk)
begin
    if(an == 3'b000)
        data <= data1;
    else if(an == 3'b001)
        data <= data2;
    else if(an == 3'b010)
        data <= data3;
    else
        data <= data4;
end

always@(posedge clk or posedge rst)
begin
    if(rst)
    begin
        data1 <= 4;
        data2 <= 3;
        data3 <= 2;
        data4 <= 1;
    end
    else if(cnt2 == 0)
    begin
        if(data1 == 9)
        begin
            data1 <= 0;
            if(data2 == 9)
            begin
                data2 <= 0;
                if(data3 == 5)
                begin
                    data3 <= 0;
                    if(data4 == 15)
                        data4 <= 0;
                    else
                        data4 <= data4 + 1;
                end
                else
                    data3 <= data3 + 1;
            end
            else
                data2 <= data2 + 1;
        end
```

```
        else
            data1 <= data1 + 1;
    end
    else
    begin
        data1 <= data1;
        data2 <= data2;
        data3 <= data3;
        data4 <= data4;
    end
end
endmodule
```

总结与思考

- 通过本次实验熟悉了IP核的使用方法，熟悉了调试verilog的若干方法
- 实验难度适中
- 任务量适中
- 无

---

1. he xu / FPGAOL-Examples · GitLab (ustc.edu.cn) ↩