数电实验EXP1

刘永鹏 191220070 计科 693901492@qq.com 2020.9.12

1. 实验目的

用 case 语句实现一个 2 位 4 选 1 的选择器，如图 1 5所示，选择器有 5 个 2 位输入端，分别为 X0, X1, X2, X3 和 Y，输出端为 F；X0, X1, X2, X3 是四个 2 位的输入变量。输出 F 端受控制端 Y 的控制，选择其中的一个 X 输出，当 Y = 00 时，输出端输出 X0，即 F = X0；当 Y = 01 时，输出端输出 X1，即 F = X1； 以此类推。

2. 实验原理

本实验即为普通4选1选择器的扩展，将输入输出全部由1位变为2位。

其真值表与普通选择器基本一致：通过**k位选择输入控制2^k位的输出。**

输出\输入	**00**	**01**	**10**	**11**
LEDR0/LEDR1	SW2/SW3	SW4/SW5	SW6/SW7	SW8/SW9

3. 实验环境/器材

1. input：**[9:0] SW**
2. output: **[9:0] LEDR**

(事实上输出只需两个LED灯，这里使用SystemBuilder快捷建立工程。)

4. 程序代码

本实验较为简单，不再绘制流程图，以下是CASE部分的代码：

```
always @ (cin)
begin
    case(cin)
        0: begin LEDR[0] = SW[2];LEDR[1] = SW[3]; end
        1: begin LEDR[0] = SW[4];LEDR[1] = SW[5]; end
        2: begin LEDR[0] = SW[6];LEDR[1] = SW[7]; end
        3: begin LEDR[0] = SW[8];LEDR[1] = SW[9]; end
        default: begin LEDR[0] = SW[2];LEDR[1] = SW[3]; end
    endcase
end
```

接着是激励代码的赋值部分。

```
begin
// code executes for every event on sensitivity list
// insert code here --> begin
SW[2] = 0; SW[3] = 0; SW[4] = 0; SW[5] = 1; SW[6] = 1; SW[7] = 0; SW[8] =
1; SW[9] = 1; #0;
SW[0] = 0; SW[1] = 0; #10;
SW[0] = 1; SW[1] = 0; #10;
SW[0] = 0; SW[1] = 1; #10;
SW[0] = 1; SW[1] = 1; #10;

@eachvec;
// --> end
end
```

设计思路即按照普通四选一选择器的思路，将输入输出扩展位2位即可。

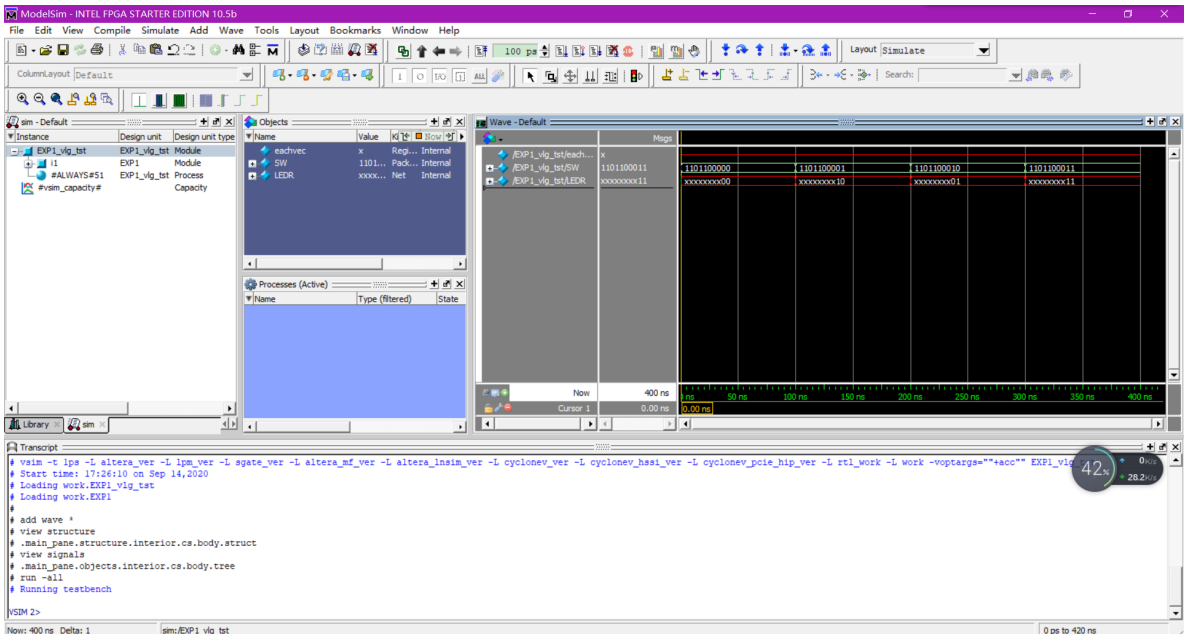
5. 实验过程

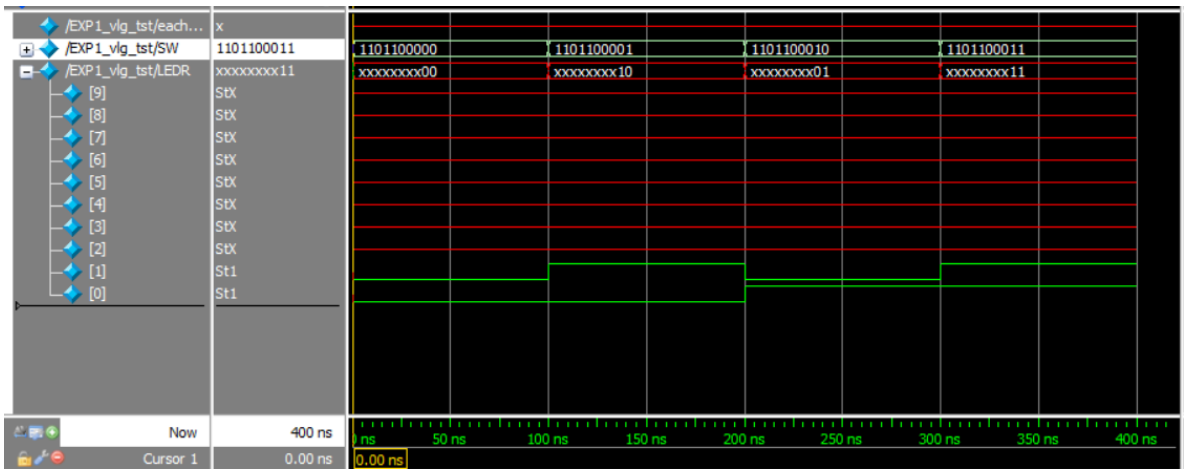
1. Systembuilder自动建立项目。
2. 编写程序代码。
3. 编译通过后自动生成Testbench。
4. 连接Testbench, RTL仿真。
5. 仿真结果无误后烧制代码在板上再次验证。

6. 验证方法

用testbench中的赋值对照RTL仿真结果直接验证。因为输入输出量较小，因此这种方法较为简单。

7. 实验结果





8. 问题与解决方案

本次实验内容上较为简单，问题主要出现在**Quartus软件的使用**上以及**verilog语言的语法**问题。

1. Quartus文件名不当会导致系统无法正确识别的问题。.v, testbench文件以及顶层文件的命名必须严格按照要求进行（亲身体验命名不当带来的严重后果QAQ）。
2. wire, reg类型以及[x:y]在定义中的位置要正确使用。（wire [1:0]cin错写为wire cin[1:0]导致我花了一段时间寻找该错误）
3. 细小的软件与语法问题在万能的百度与各种博客的帮助下顺利完成。

9. 启示

问题分析透彻再动手！基础能力达标再动手！

1. verilog边学边做实验（×）| 基本语法完全掌握再做实验（√）
2. 题目读清楚再做实验！（第一遍用if_else写了一半才发现要求用case）

10. 意见与建议

目前暂无。