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：Verilog 硬件描述语言

学生姓名：_____徐奥_____

学生学号：_____PB20061343_____

完成日期：__2021 年 11 月 4 日__

计算机实验教学中心制

2020 年 09 月

【实验题目】

Verilog 硬件描述语言

【实验目的】

1. 掌握 Verilog HDL 常用语法
2. 能够熟练阅读并理解 Verilog 代码
3. 能够设计较复杂的数字功能电路
4. 能够将 Verilog 代码与实际硬件相对应

【实验环境】

vlab.ustc.edu.cn

【实验过程】

1. Verilog 关键字。包括：module/endmodule、input、output、wire、reg、assign、always、initial、begin/end、posedge、negedge、if、else、case、endcase 等。
2. Verilog 代码基本结构。每个模块都是由关键字 module 开头，由 endmodule 结束。每个模块都应该有一个唯一的模块名，模块名不能使用 Verilog 语法的关键字。模块名后面的括号内是对输入输出信号的定义。模块主体部分只能出现四类语句：内部信号定义、模块实例化、assign 语句、always 语句，每类语句的数量不受限制。
3. Verilog 数据及类型。4 中基本值，常量。
4. Verilog 操作符：算术运算符、关系运算符、逻辑操作符、归约操作符、条件操作符、移位操作符、拼接操作符
5. Verilog 表达式
6. 模块调用

【实验练习】

题目 1. 阅读以下 Verilog 代码，找出其语法错误，并进行修改

语法错误包括：

（1） if、else 用于实现带有优先级的条件分支， 一般出现在 always 语句的过程语句部分，而不能直接在模块内部单独出现。所以应将 if、else 语句放在 always 中。

（2） 在 always 语句中的赋值操作对象，应设置为 reg 型，而非默认的 wire 型。

修改为：

```
module test(
    input a,
    output reg b
);
    always@(*) begin
        if(a) b = 1'b0;
        else b = 1'b1;
    end
endmodule
```

图 1

题目 2. 阅读以下 Verilog 代码，将空白部分补充完整

```
module test(
    input [4:0] a,
    output reg [4:0] b
);
    always@(*)
        b = a;
endmodule
```

图 2

题目 3. 阅读以下 Verilog 代码，写出当 a = 8'b0011_0011, b = 8'b1111_0000 时各输出信号的值

c = 8'b0011_0000

d = 8'b1111_0011

e = 8'b1100_0011

f = 8'b1100_1100

g = 8'b0011_0000

h = 8'b0000_0110

i = 8'b0000_0000

j = 8'b1111_0000

k = 8'b0100_0011

题目 4. 阅读以下 Verilog 代码，找出代码中的语法错误，并修改错误：

(1) assign 赋值是对线网型赋值，不能对 reg 类型赋值
(2) 端口信号可以通过位置或名称进行关联，但两种关联方式不能混用。

修改为：

```
module sub_test(
    input a,b,
    output c
);
    assign c = (a<b)? a : b;
endmodule

module test(
    input a,b,c,
    output o
);
    wire temp;
    sub_test sub_test1(.a(a),.b(b),.c(temp));
    sub_test sub_test2(.a(temp),.b(c),.c(o));
endmodule
```

图 3

题目 5. 阅读以下 Verilog 代码，找出其中的语法错误，说明错误原因,并进行修改。

错误：

（1） 模块实例化不可以在 always 里，Verilog 是硬件设计语言，模块实例化在往电路板上安装芯片（module）。如果模块实例化放在 always 里面了，就相当于“有条件地安装一个芯片”，这显然不是一个正常的硬件逻辑。
（2） 模块名后面的括号内应该包含输入输出信号的定义,不应该把 output 放在括号外。

修改为：

```
module sub_test(
    input a,b;
    output o
);
    assign o = a + b;
endmodule

module test(
    input a,b,
    output c
);
    sub_test sub_test(.a(a),.b(b),.o(c));
endmodule
```

图 4

【总结与思考】

1. 本次实验系统学习了 verilog 语言，对于它的使用有了更深的了解，为之后的实验打下基础。
2. 在复习过程中，对于实验文档的细节知识需要认真学习，其中就包括：模块实例化：实例化的输出端只能接 wire 类型信号
3. 本次实验内容量适中，难度适中，但对于细节知识的考查十分细致。