浙江大学 2015－2016 学年 夏 学期

《HDL 语言 》课程期末考试试卷

课程号：_____，开课学院：_____生仪学院_____

考试形式：闭卷 √、开卷（请在选定项上打 √）

考试日期：_2016_年_6_月_25_日，考试时间：_120_分钟

诚信考试，沉着应考，杜绝违纪。

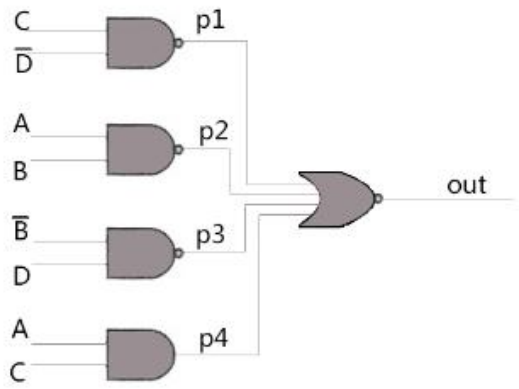
考生姓名：_____学号：_____所属院系：_____

题序	一	二	三	总 分
得分				
评卷人				

考生注意：本试卷共 11 张，3 道大题，分 19 道小题，请检查一下有无缺页。

一、 填空题（每空 2 分，共 30 分）

1、请补充如下图所示的门级电路的**结构描述**的 HDL 语言：




```
module DUT ( );
    wire    p1,p2,p3,p4,out;
    reg     A,B,C,D;
    nand
        g1(p1,C,～D);
        g2(p2,A,B);
        g3(p3,～B,D);
    _____;
    _____;

endmodule
```

2、Verilog 有四种基本循环语句，包括 for、 forever、repeat 和_____。

verilog 的本质是_____触发的 HDL 语言。

3、状态机按照输出逻辑可以分为两种，一种是_____，另一种是_____，

二者的区别是：_____。

4、状态机的状态变量一般采用的编码方式有(至少写两种)_____，

其各自的优点是_____。

5、以下代码用于生成斐波纳挈数列，其语法上有两处错误，请指出。

_____ _____

```
module top;
    wire   [15:0]   num;
    wire   [15:0]   num_out;
    number_gen   ng     (num);
    figNumCalc   fnc    (num,num_out);
endmodule

module number_gen
(output reg   [15:0]   num = 0);

    always   begin
        #50 num = num + 1;
        #50 -> ready;
        end
endmodule

module figNumCalc
(input      [15:0]   startingValue,
output reg [15:0]   fibNum);

    reg [15:0] count,oldNum;
    wire [15:0] temp;
    always   begin
        @ng.ready
        count = startingValue;
        oldNum = 1;
        for(fibNum = 0;count != 0; count = count - 1)   begin
            temp = fibNum;
            fibNum = fibNum + oldNum;
            oldNum = temp;
            end
        end
endmodule
```

6、以下代码实现时是否有问题，如何优化？_____

```
module test(
input     a,
input     data_in,
output    data_out);
      always @(a or data_in)   begin
            case(a)
            0:    data_out <= data_in;
            endcase
      end
endmodule
```

7、以下两段代码的行为有什么不一样？_____

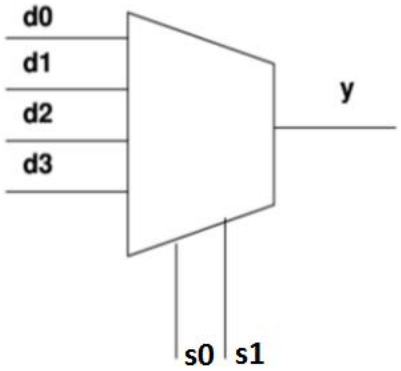
```
...
@(posedge exp)
      #1 statement1;
...
```

```
...
wait(exp)
      #1 statement1;
...
```

8、用 verilog 设计一个带异步复位、同步置位，时钟上升沿触发的 D 触发器。

```
module DFF(
input clk,
input clk_enable, //为高电平时 clk 输入有效
input d,
input set_n,
input rst_n,
output q);









endmodule
```

9、用行为描述设计以下四选一数据选择器，要求四路输入有相同优先级：




```
module mux_4_to_1(
input d0,
input d1,
input d2,
input d3,
input s0,
input s1,
output y);
```

二、 简述题（每题 5 分，共 25 分）

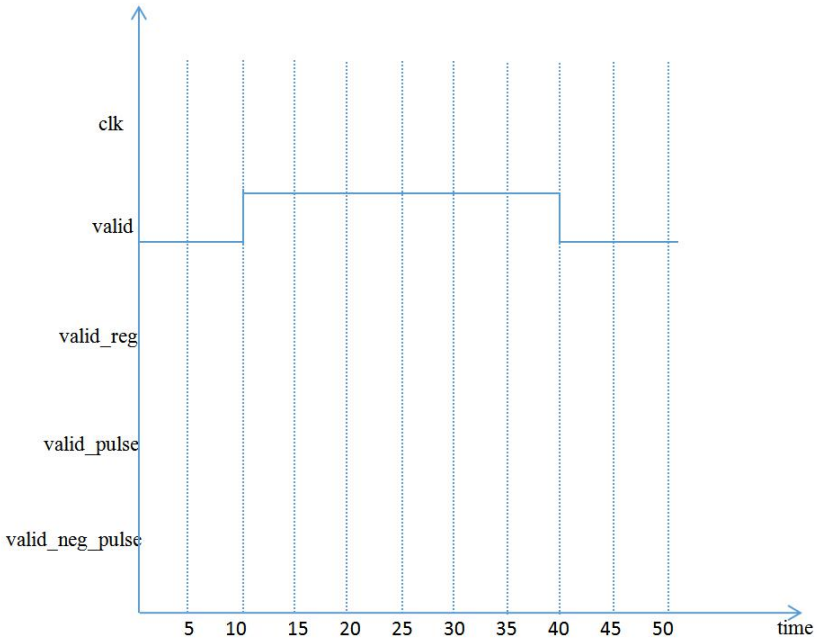
10、试简述阻塞赋值（=）和非阻塞赋值（<=）的区别。并解析下面两段代码在时序上有什么不同。

```
begin
      a = #50 1; b = #100 1; c= #150 1;
end
```

```
begin
      a <= #50 1; b <= #100 1; #100 c = 1;
end
```

11、根据已知的 valid 波形，画出下面程序中变量 clk、valid_reg、valid_pulse、valid_neg_pulse 的仿真波形(从 0ns 到 50ns)。

```
`timescale 1ns/1ns
reg     valid_reg;
wire    clk;
wire    valid_pulse;
wire    valid_neg_pulse;

always    begin
      #3    clk = 0;
      #7    clk = 1;
      end

always @(posedge clk)
      valid_reg <= valid;

assign valid_pulse = valid &(~valid_reg);
assign valid_neg_pulse = (~valid )&valid_reg;
```



12、系统任务$monitor 与$display 有什么区别？$stop 与$finish 有什么区别？

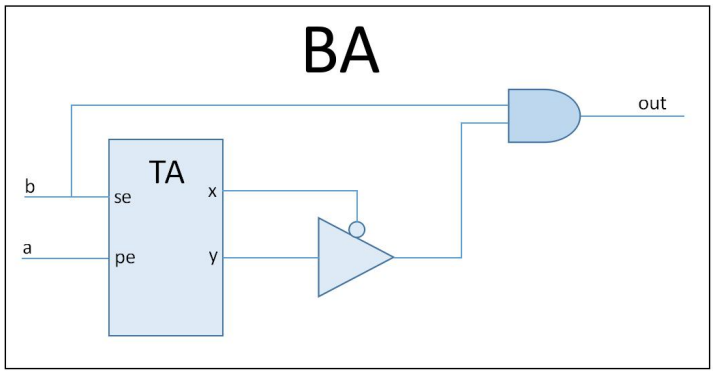
13、always 语句和 initial 语句的关键区别是什么？能否相互嵌套？

14、用 function 实现计算一个 8 位变量的奇校验位（如果 1 的个数为偶数，奇校验位为 1），并简述 function 与 task 的区别(至少三点）。

三、 设计题（共 45 分）

15、TA 模块程序如下，写出引用 TA 模块实现 BA 模块的行为级描述 verilog 代码。（5 分）

```
module TA(
input se,
input pe,
output x,
output y);

assign {x,y} = se + pe;
endmodule
```



16、已知输出表达式如下：（10 分）

$$F（A,B,C,D）=\sum m(0,4,6,8,10,11)+\sum d(2,13,15)$$

要求：

- 画出相应的卡诺图；
- 写出输出的最简与或表达式；
- 用 verilog 实现输出表达式。

17、设计一个同步系统实现的秒杀仲裁电路，2 个用户参与秒杀，模块定义如下：（10 分）

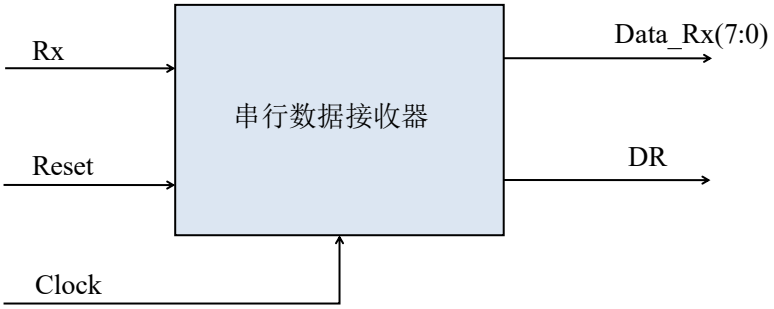
```
module SecKill(
input clock,
input req1,
input req2,
input start,
input clear,
output out1,
output out2);
```

clock:时钟输入
start:初始化为逻辑 0，秒杀开始后为逻辑 1；
req1:用户 1 的秒杀请求输入；
req2:用户 2 的秒杀请求输入；
out1:用户 1 秒杀成功输出；
out2:用户 2 秒杀成功输出；
clear:当 clear 持续一个时钟周期为高电平时，结束秒杀，start 置零。

要求：

- 如果用户在开始秒杀前请求秒杀，则失去秒杀资格；
- 开始秒杀后，第一个输入秒杀请求的用户秒杀成功，其余失败；
- 如果两个秒杀请求同时到达，则用户 1 秒杀成功；
- 可能没有用户秒杀成功。

18、请用 verilog 语言实现一个串行数据接收电路，接收串行数据并输出并行数据，其框图如下，不要求测试文件。（10 分）



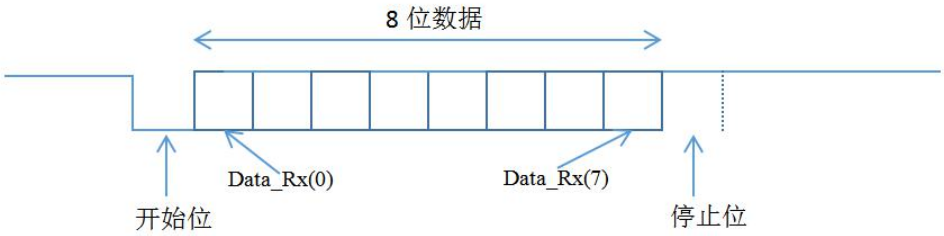
该电路时钟频率为串行数据传输波特率的 16 倍。产生一个 8 位并行输出，和一个状态输出(DR),当 1 字节的数据接收完毕时，该状态输出一个时钟周期的逻辑 1。Rx 输入端的数据格式如下：

① 1 位起始位（逻辑 0）；

② 8 位传输数据位（低位在前、高位在后）；

③ 1 为停止位（逻辑 1）；

④ 无奇偶校验。



当接收电路等待数据时，Rx 为逻辑 1。当 Rx 为逻辑 0 时代表起始位的指示（由外部电路产生）。然后传输 8 位数据位，最后用 1 位停止位（逻辑 1）代表传输结束。

19、设计 verilog 代码实现一个 4 位十进制数组合锁（假设密码为 0420），如果用户成功输入顺序正确的 4 位数，输出 Right 信号，如果任何一位输入错误，则在 4 位数输入完毕后输出 Wrong 信号，并且输入 3 次错误后组合锁锁死，不能再解锁，Lock 信号保持逻辑 1，除非系统通过 reset 信号复位。（10 分）

要求：

- 画出状态转移图；
- 用 verilog 实现状态机，模块定义如下；
- 不要求测试文件。

```
module NumLock(
input clock,
input reset,
input [3:0] Num,
output Right,
output Wrong,
output Lock);

clock:时钟输入；
reset:系统复位输入，高电平有效；
Num:单个数位输入；
Right:解锁成功输出；
Wrong:解锁失败输出；
Lock:锁死输出。
```