硬件描述语言及 FPGA 设计平日作业

姓名： 学号： 专业：

一、单选题：

1. SignaltapII 的采样时钟的频率必须:（ ）。 A．小于被测信号频率；

B．大于被测信号频率；

C．SignaltapII 和 Modelsim 一样，都是软仿真工具；

D． 以上都不对。

1. 片上可编程器件的英文缩写是( )。 A、SOP

B、SOPC

C、SoC D、SPI

1. 下面的软件中，专门用于 FPGA 仿真的是： ( )

A. QuartusII B.Modelsim

C.protues D.Symplify

1. 下列关于同步和异步复位描述正确的是（ ）。 A、同步复位是不受时钟影响；

B、使用 FPGA 设计时芯片的异步复位和同步复位可随意替换使用； C、同步复位需要在时钟沿来临的时候才会对整个系统进行复位；

D、同步复位最大的优点是，数据通路可以不依赖于时钟而复位可用。 5．下面说法正确的是：（ ）

A. Modelssim 不能进行后仿真 B. 不可综合语句是没有用的

C. Verilog HDL 没有包含大量不可综合的语句 D. QuartussII9.0 支持仿真

1. 下列设计流程中，哪一步不能在 QarutusII 上实现：（ ） A.RTL 级描述 B.逻辑综合

C.门级网表 D.芯片级版图

1. 在下列程序中，always 状态将描述一个带同步 Nset 输入端的上升沿触发器，则空括号内应填入

（ ）。

always @（ ） if（！Nset）

Q<=1;

else

Q<=D;

A、 posedge Clock B、 negedge Nset

C、negedge Nset or posedge Clock D、negedge Nset or negedge Clock

1. INTEL FPGA 芯片不包括（ ）。 A、Cyclone V B、Kintex-7

C、Arria D、Stratix

1. 下列哪些语句不可以被综合成电路 ( )。 A、initial

B、always C、assign D、for

二、填空题：

1. 目前世界上符合 IEEE 标准的硬件描述语言有： 。
2. initial 块执行 次。
3. 行为级建模中的过程赋值语句用于对 类型的变量赋值。
4. 若条件分支比较多，使用 语句比较方便。
5. 顺序块使用关键字 来表示。
6. Verilog HDL 程序是由 构成。
7. Verilog HDL 模块中的语句位于关键字 之中。
8. 函数语句使用关键字 来定义。
9. always 块可以表示 逻辑或 逻辑。
10. 状态机分为： 和 。
11. NiosII 的内核分为三个工作模式： 、 和经济型。
12. 状态机的置位和复位分为： 和 。
13. 按照可综合模块设计原则，用 always 块对组合逻辑建模时，用 。
14. Verilog 模型的行为描述包括： 、 、 。
15. 同一模块中多个 always 语句执行顺序是： 。
16. SignalTapII 的测试接口是 JTAG，其设计文件后缀名是 。
17. 可编程逻辑单元 LE 主要包括 3 个部分： 、 、输出寄存器逻辑。
18. Quartus 的输入方法中，采用原理图输入法建立的文件后缀为 。
19. 目前 Altera FPGA 主要： 、 、 Cyclone。
20. 目前最具代表性的软核嵌入式系统处理器包括： 、 。
21. 目前 Altera FPGA 的配置数据主要下载方式包括： 、 。
22. 根据仿真逻辑意义的不同，仿真可分为三类： 、 、 RTL 级仿真。
23. 结构化的过程块包括： 、 。
24. always 块执行次数： 。
25. assign 语句用于对 类型的变量赋值。
26. 行为级条件语句使用关键词 来表示。
27. 块语句按照语句执行时序可分为： 、 。
28. 并行块使用关键字 来表示。
29. Verilog HDL 最基本的 4 种数据类型包括： 、 、 、和 integer。
30. 任务使用关键字 来定义。
31. 带有 posedge 或 negedge 关键字的事件表达式表示 的时序逻辑。
32. Verilog 模型的描述包括：行为描述、 和 级 。
33. 同一模块中多个 initial 语句执行顺序是： 。
34. 在 quartusII 中，图形文件的扩展名是 。
35. 目前主要的 FPGA 公司： 、 、Microsemi（Actel）。
36. Altera FPGA 内部的 CPU 分为： 、 、固核。
37. 目前世界上符合 IEEE 标准的硬件描述语言有： 、 。三、简答题：

1、 基于数字系统设计流程包括哪些步骤 ?各步骤的功能？答：

2、 请描述使用 modelsim 进行仿真的基本流程。答

3、FPGA 与 ARM、DSP 从资源配置来看的区别有哪些。答

4、FPGA 与 ARM、DSP 开发语言本身的区别有哪些。答：

5、FPGA 系统结构和资源包括哪些？画出分布图。答：

6、NIOS II 硬件开发流程包括哪些？答：

1. 名词解释：

1、SoC

答：片上系统，又称芯片系统、系统芯片，是指把系统集成在一片芯片上

2、LUT

答：可编程逻辑结构，查找表

3、JTAG

答：联合测试行动组

4、ISP

答：系统内编程

5、下载

答：将设计好的配置数据传输到FPGA中以实现特定功能的过程

6、FPGA

答：现场可编程门阵列

1. wire 型

答：表示物理连线的数据类型

1. reg 型

答：表示寄存器类型的数据

1. 布局布线

答：将逻辑综合后的网表映射到具体物理资源并完成互连的过程

1. 逻辑综合

答：将高级硬件描述语言（HDL）转换为低层门级网表的过程

1. RTL 设计

答：寄存器传输级设计，用HDL描述数字电路在寄存器之间的数据流动和处理方式

12、时序约束

答：对设计中信号路径的延迟设置时间要求，以确保满足系统时钟频率等性能指标

13、Verilong HDL

答：一种常用的硬件描述语言，用于建模和仿真数字逻辑电路

14、时序分析

答：对电路中信号传播路径的延迟进行分析，判断是否满足设定的时序要求

15、综合优化

答：在逻辑综合过程中对电路进行面积、速度和功耗等方面的优化处理

16、CPLD

答：复杂可编程逻辑器件，一种基于乘积项结构的非易失性可编程逻辑器件

17、IP 核

答：知识产权核，预先设计好的、具有特定功能的可重用模块或代码组件

18、ISP 下载

答：通过在系统编程接口对目标器件进行程序烧录的过程

19、EDA

答：电子设计自动化，使用计算机软件工具进行电路设计、仿真和验证的技术

20、ASIC

答：专用集成电路，为特定应用需求专门设计制造的集成电路

21、FIFO

答：先进先出队列，一种常用于数据缓冲的数据结构或硬件模块

22、时序逻辑

答：输出不仅依赖于当前输入，还依赖于电路过去状态的逻辑电路

23、约束文件

答：包含设计所需的时序、引脚分配、电压等限制条件的文本文件

24、行为级建模

答：从算法和功能角度对电路进行建模，不涉及具体硬件结构

25、综合优化

答：对综合结果进行性能和资源使用的优化

26、时序路径

答：从起点（如触发器或输入端口）到终点（如触发器或输出端口）的信号传播路径

27、综合约束

答：在综合阶段指定的设计目标，如最大延迟、最小周期等

28、Testbench

答：测试平台，用于对被测模块进行功能验证的仿真环境

1. 组合逻辑

答：输出仅取决于当前输入的逻辑电路，不具有记忆功能

1. 硬件仿真

答：使用仿真工具对硬件设计的功能和时序行为进行模拟验证

五、电路设计题

1. 设计实现下表 1 功能的电路程序(模块名称 dff）

表 1 功能表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| clk | clr | reset | d | q | qn |
| ↑ | 0 | x | x | 0 | 1 |
| ↑ | 1 | 0 | x | 1 | 0 |
| ↑ | 1 | 1 | ０ | ０ | １ |
| ↑ | 1 | 1 | １ | １ | ０ |

module dff(

input clk,

input clr,

input reset,

input d,

output reg q,

output qn

);

// qn 始终是 q 的非

assign qn = ~q;

always @(posedge clk) begin

if (clr == 1'b0) begin

q <= 1'b0;

end else if (reset == 1'b0) begin

q <= 1'b1;

end else begin

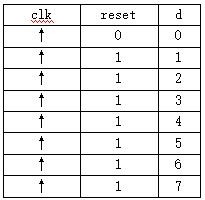
q <= d;

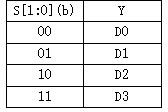
end

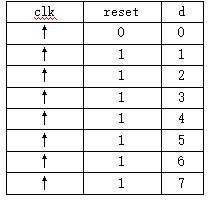
end

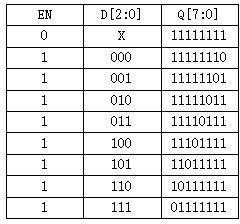
endmodule

|  |  |  |
| --- | --- | --- |
| EN | D2D1D0(d) | Q[7:0](b) |
| 0 | x | 11111111 |
| 1 | 0 | 11111110 |
| 1 | 1 | 11111101 |
| 1 | 2 | 11111011 |
| 1 | 3 | 11110111 |
| 1 | 4 | 11101111 |
| 1 | 5 | 11011111 |
| 1 | 6 | 10111111 |
| 1 | 7 | 01111111 |









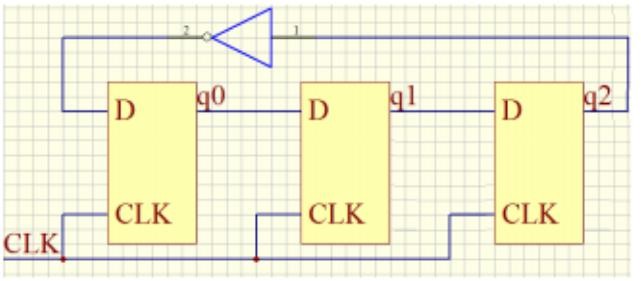


图 1 电路图

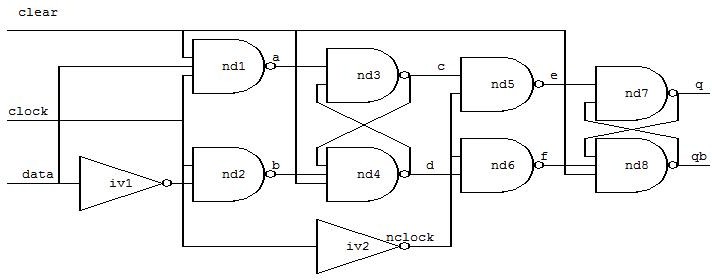


图 2 门级 D 触发器

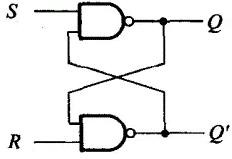


图 3 电路图

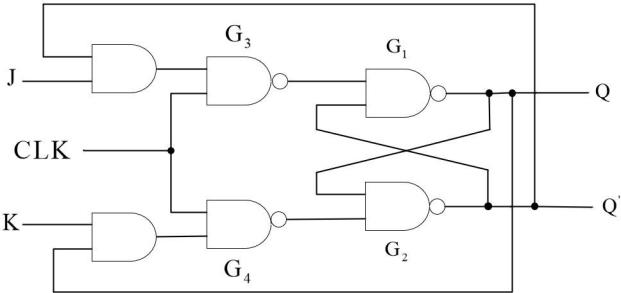


图 4 电路图

11.设计下图 5 电路图的程序(模块名 muxtwo)。

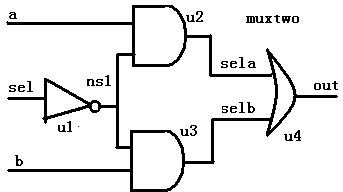


图 5 电路图

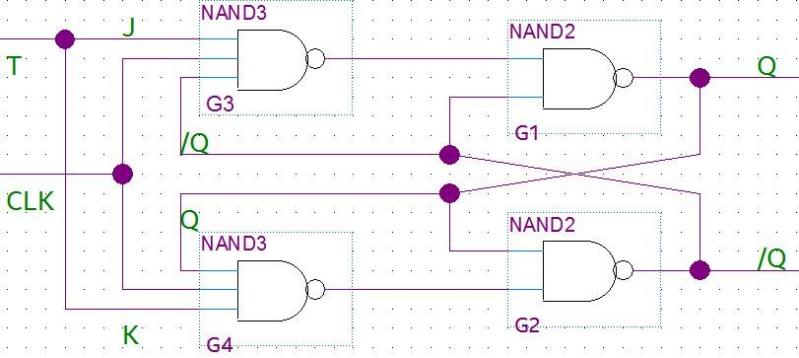


图 6 T 触发器电路图

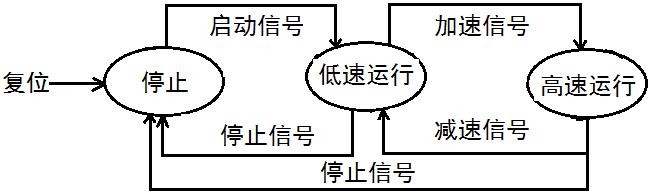


图 7 电机控制图

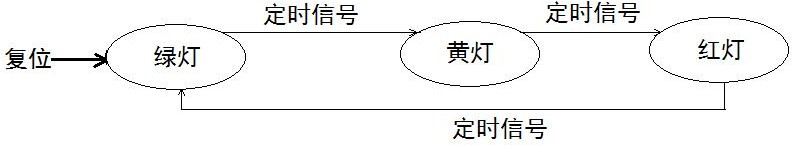


图 8 序列检测电路图

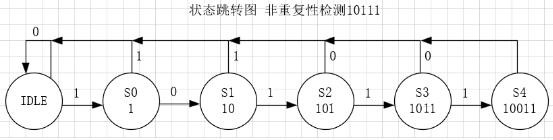


图 9 序列检测电路

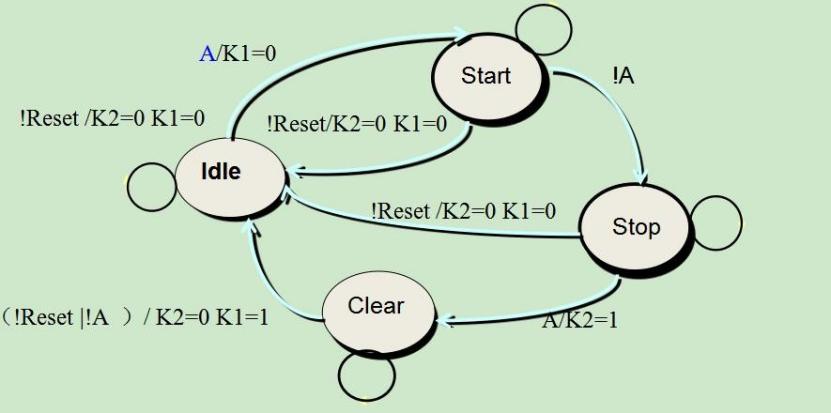


图 10 状态转移电路图

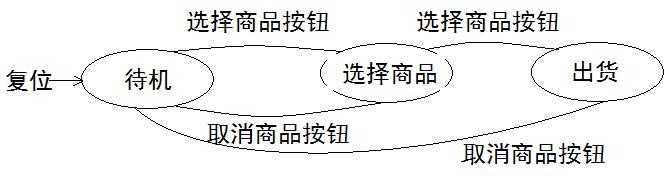


图 11 自动售货机电路

六、综合题

1.试分析下面程序，找出程序中存在的语法错误，写出正确的完整程序。

//对 CPA 四分频，且占空比为 1/3 module div4(CPA，CPB)

output CPB； input CPA;

Reg CPB;

always @ (posedge CPA) if (i==3)

begin i=0；CPB<=b’1；end else

begin i=i+1；CPB<=0；end

end endmodule

//带异步清零的 JK 触发器逻辑功能描述 module JKCFQ(Q,J,K,CLK,RD)

output Q;

input J,K,CLK,RD; always @ (posedge CLK)

begin

if(!RD) Q<=0;

else case(J,K)

2’b00 : Q<=Q;2’b01 : Q<=0;

2’b10 : Q<=1;2’b11 : Q<=~Q;

default : Q<=1’bx; endcase

end Endmodule

//异步清零的 4 位计数器

module Hardreg(d,clk,clrb,q) input clk,clrb;

input [3:0]d;

output [3:0]q;

Reg [3:0]q;

always @ (posedge clk) if (clrb)

begin q<=b’1；end else

begin q<=d+1;end

end endmodule

module Example(clk, data\_in, data\_out); input clk;

input [3:0] data\_in; output data\_out;

reg temp;

always @(posedge clk) begin temp = data\_in[0];

end

always @(data\_in) begin case(data\_in)

4'b0001: data\_out = 1'b1; 4'b0010: data\_out = 1'b0;

endcase

end

assign data\_out = temp;

module Counter(clk, rst, cnt) input clk, rst;

output cnt;

always @(posedge clk) begin if (rst) cnt = 0; else cnt <= cnt + 1;

always @(\*) begin

$display("Current count: %d", cnt); end

initial begin cnt <= 0; end

endmodule

module Mux(input [1:0] sel, input [3:0] D, output Y); reg Y;

always @(sel or D) begin casez(sel)

2'b00: Y = D[0];

2'b01: Y = D[1];

2'b10: Y = D[2];

endcase

end

assign D = {D[2:0], 1'b0}; initial begin

#10 Y <= 0;

end

1. 分析电路描述，画出产生的波形。 module CLK\_tb(clk);

output clk; reg clk; initial

begin

#3 clk = 1;

#4 clk = 0;

#6 clk = 1;

#2 clk = 0;

#5 clk = 1;

#12 clk = 0;

end endmodule

1. 分析电路描述，画出产生的波形。 module Pulse (clk);

output clk; reg clk; initial

fork

#2 clk = 1; #7 clk = 0;

#11 clk = 1;#15 clk = 0;

#16 clk = 1; #21 clk = 0; join

Endmodule

module Clk\_tb (clk); output clk;

reg clk;

initial clk = 1; initial

fork

#2 clk = ~clk; #5 clk = ~clk;

#10 clk = ~clk; #15 clk = ~clk; #20 clk = ~clk; #25 clk = ~clk;

join Endmodule

module WAVE\_tb; reg clk; initial begin

clk = 0;

forever #5 clk = ~clk; end

endmodule

1. 分析下图 12 中的波形，设计一个测试模块，产生同样的波形（`timescale 1us/1us）。

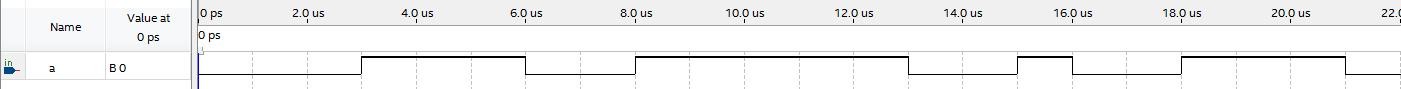


图 12 波形图

module a\_tb(a);

output a;

reg a;

initial

begin

a= 0;

#3 a= 1;

#3 a = 0;

#2 a = 1;

#5 a = 0;

#2 a = 1;

#1 a = 0;

#2 a = 1;

#3 a = 0;

#1 $stop

end endmodule

1. 分析下图 13 中的波形， 设计一个测试模块（ LED\_tb ） ， 产生同样的波形（ `timescale 1us/1us）。

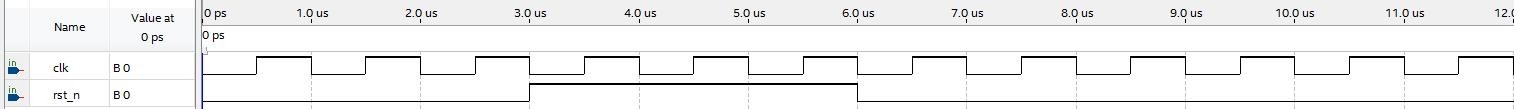


图 13 波形图

module LED\_tb; reg clk;

reg

initial begin

clk = 0;

forever #0.5 clk = ~clk;

end

initial begin

rst\_n = 0;

#3 rst\_n = 1;

#3 rst\_n = 0;

end

endmodule

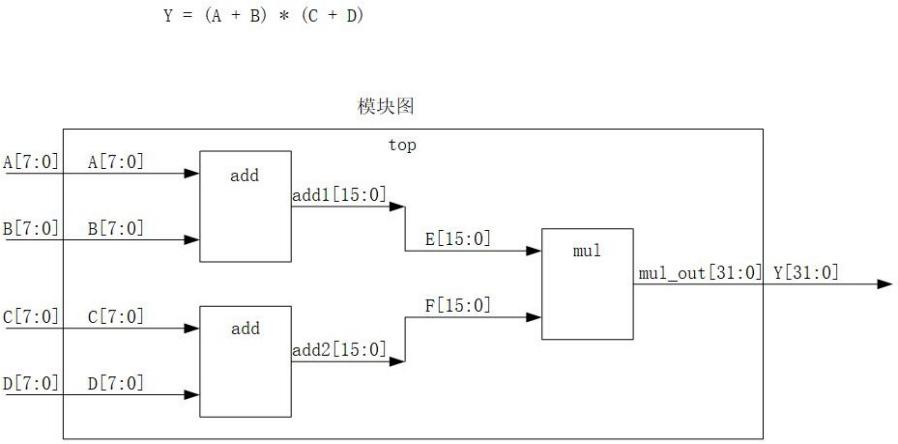


图 14 层次化设计模块图

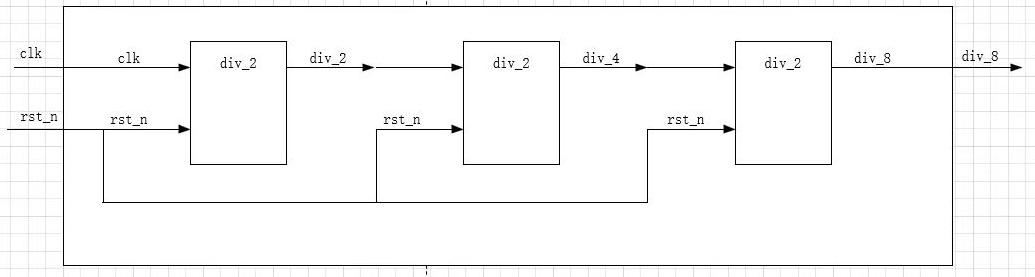


图 15 层次化设计模块图

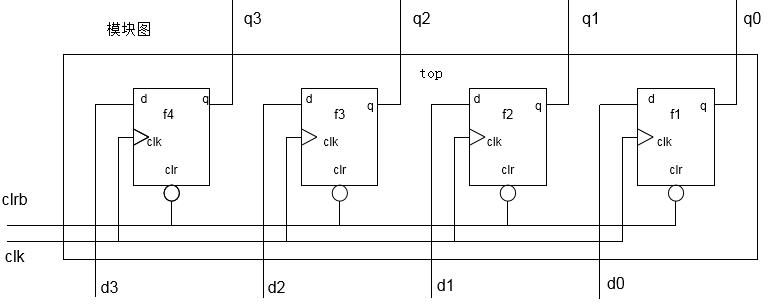


图 16 层次化设计模块图

数器，设计相关电路模块，同时设计测试模块（DIV\_16\_tb）。

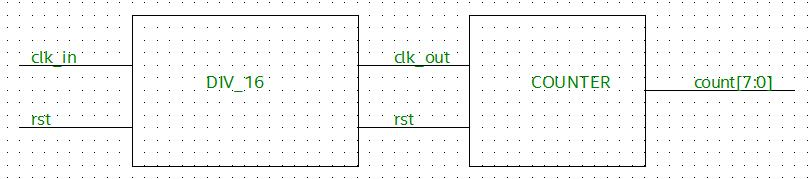


图 17 层次化设计模块图

1. 根据下图 18 层次化模块图设计电路模块及测试模块（ 模块名 ADDER\_8BIT ， 测试模块

ADDER\_8BIT\_tb）。

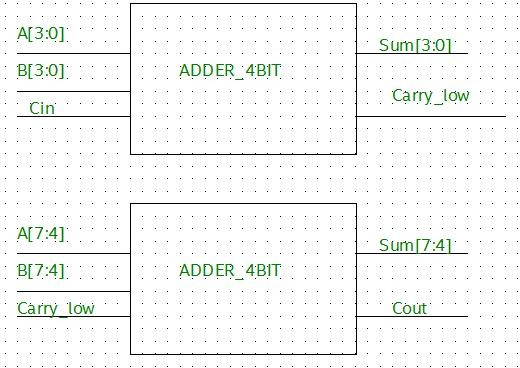


图 18 层次化设计模块图

1. 根据层次化模块图 19 设计电路程序及测试模块（模块名 MULTIPLEXER，测试模块 mux\_tb）。

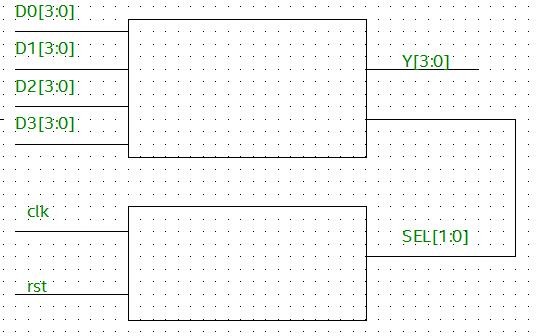


图 19 层次化设计模块图