《数字电路与 EDA》测试题

一、选择题（黑色加粗为答案）

1、EDA 技术是设计实现电路系统的一种技术，该技术以（）为目标器件。

A、PLD 器件　B、ASIC 器件　**C、PLD 或 ASIC**　D、FPGA/CPLD

2、某摩尔（Moore）型有限状态机，状态变量 state，输入变量 x，三过程设计，输出逻辑模块 always 敏感列表准确的是（）。

A、always@(state,x)　B、always@(x)

C、always@(state)　D、always@(*)

3、随着更大规模的 FPGA/CPLD 器件的推出，哪种技术的出现，使得 SOPC 技术进入实用化？

A、嵌入式微处理器软核　B、嵌入式 DSP 软核

C、嵌入式微控制器软核　D、嵌入式 AD 软核

4、如果某电子系统设计人员，先调用设计库中的基本元件（门电路等）组成子系统，再逐步扩大到整个系统的设计思路是（）设计。

A、Bottom-Down　B、Top-Down　C、IP reuse　**D、Bottom-Up**

5、IP 核按照移植灵活性上讲，由高到低的顺序是（）。

A、硬核、固核、软核　B、硬核、软核、固核

C、软核、固核、硬核　D、软核、硬核、固核

6、数字系统设计流程中，输入有两种方式（）。

A、HDL 和文本　B、原理图和 Verilog HDL

C、原理图和 VHDL　**D、原理图和文本**

7、有别于软件的编译，EDA 设计中的综合指的是（）。

A、将门电路转化成 HDL 代码　B、将 HDL 代码转化成门电路

C、将较高层次逻辑描述转化成较低层次电路描述

D、将较低层次电路描述转化成较高层次逻辑描述

8、从集成度上来划分，下列器件都属于复杂 PLD 器件的是（）。

A、PROM 和 GAL　**B、CPLD 和 FPGA**

C、CPLD 和 PLA　D、FPGA 和 PROM

9、CPLD 器件和 FPGA 器件从结构和原理上来看，分别基于（）。

A、查找表和 LUT　**B、乘积项和 LUT**

C、与或阵列和 RAM　D、MC 和 LB

10、下图 1-1 为某一电路模块，则下列描述正确的是（）。

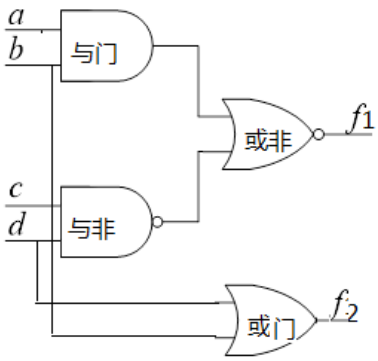
<table>
<tr>
<td> 

图 1-1</td>
<td>A、assign f1= ~ ((a&b) | (~(c&d)))
assign f2= b+d
B、assign f1= ~ ((a&b) || (~(c&d)))
assign f2= b || d
C、assign f1=~ ($\overline{cd}$ +ab)
assign f2= b + d
D、assign f1= ~ ((a&b) | (~(c&d)))
assign f2= b | d</td>
</tr>
</table>

11、下列模块声明中，哪个模块名不合法（）？

A、module Always（a,b,c）； B、module Test（a,b,c）；

C、module @t1（a,b,c）； D、module always_1（a,b,c）；

12、下列变量定义中，正确的是（）。

A、*a1 B、56sin **C、count** D、$_test

13、假设 reg[6:0] a=6’haf； reg[4:0] b=4’d10 ；reg[3:0] c；c={a[6:5]，b[2:0]}，c 的值为（）。

A、2’b01 B、4’d2 **C、4’d10** D、2’h10

14、下面定义一个容量为 20，字长为 1 的存储器 a，正确的是（）。

A、reg[19:0] a; B、reg[19:0] a[19:0] **C、reg a[19:0]** D、reg a[20]

15、假设 a=4’b1001，b=6’d57,那么 reg c=a&&b,reg[5:0] d=a&b 的结果为（）。(原始题目有问题)

A、c=6’b001001 d=6’d9 B、c=1’h1 d=6’b9

C、c=1’b1 d=6’d9 D、c=6’b001001 d=6’d1001

二、填空题

(3)	output reg[6:0] c;或 output [6:0] c;	**(4)**	output [6:0] d;
(5)	reg[3:0] e	**(6)**	parameter WIDTH=4;
(7)	always@(posedge clk,negedge clr)	**(8)**	qout<={qout,~qout[3]}
(9)	task my_task	**(10)**	my_task(a,b,c)
(11)	**cout1 | cout2**	**(12)**	**posedge clk 或 negedge clk**
(13)	**negedge clk 或 posedge clk**	**(14)**	**negedge KEY[1]**
(15)	**(sig1==0)?CLK1:((sig1==1)?CLK100:CLK1000) 或相同逻辑的其他描述**		

2、（共 6 分）某电路设计要求为： a、b 为两个 4 位输入， c、d 为 7 位输出。

module MD(a,b,c,d); ****下面为端口、数据类型声明部分********** input[3:0] a,b; //输入 a，b （3） //输出 c 声明 （4） //输出 d 声明 （5） // 变量 e 的定义 ***端口、数据类型声明部分********** always@ (a , b) ***//过程块 1***	begin *c=。。。//与 a、b 的逻辑描述，略* *e=。。。。。//具体语句，略;* end Show s1(e，d)；*//模块调用，e 在 Show 模块中作为输入变量，位宽为 4；d 为 7 位输出。* endmodule

3、（共 6 分）下面代码为异步复位模为 8 的 Johnson 计数器，其逻辑为最高位求反的循环左移，即 0000→0001→0011→0111→1111→1110→1100→1000→0000。。。。。

module johnson(clk,clr,qout); (6) //定义符号常量 WIDTH input clk,clr; //clk 为时钟信号；clr 为异步复位信号，低电平有效 output reg [(WIDTH-1):0] qout; always @((7))//时钟为上升沿、clr 为异步复位	begin if(~clr) qout<=0; else begin (8) //采用拼接运算符实现高位求反的循环左移逻辑。 end end endendmodule

4、（共 4 分）任务定义，根据说明将合适的代码填入对应位置。

module alutask(code,a,b,c); ……//其他代码，略 （9） //任务定义，任务名为 my_task input[3:0] a,b; output[5:0] out; integer i; begin for(i=3;i>=0;i=i-1) out[i]=a[i] | b[i]; end endtask	always@（*） begin case（code） 2’b00: (10) //任务调用，输入为 a，b，输出为 c 2’b01:……//代码，略 Endcase end endmodule

5、（共 6 分）下面是一个占空比 50%的 7 分频电路，采用同步复位（reset），请将代码补齐。

```
module count7(reset,clk,cout);
input clk,reset;   output cout;
reg[2:0] m,n;      reg cout1,cout2;
assign cout=    （11）    ;
always @(   （12）   )//过程 1
  begin
  if(!reset)
    begin   cout1<=0;   m<=0;   end
  else   begin
    if(m==6)   m<=0;
    else   m<=m+1;
    if(m<3) cout1<=1; else cout1<=0;
    end
  end
always @(    （13）    )
  //代码逻辑与过程 1 相同，产生时钟信号 cout2
 endmodule
```

6、（共 4 分）下面为实验中计时电路的设计代码，为了观测方便，通过 KEY[1]按键改变计时器的速度，根据提示补齐代码。

```
module Timer(KEY,CLOCK_50,HEX7);
……代码略
divclk1000hz c1(1,CLOCK_50,CLK1000);//产生 1000hz 频率
divclk100hz c0(1,CLOCK_50,CLK100);////产生 100hz 频率
divclk1hz    c2(1,CLOCK_50,CLK1);////产生 1hz 频率
always@(   （14）   )//按键 KEY[1]控制 sig1 值
begin
  if(sig1==2)   sig1=0;
  else          sig1=sig1+1;
end
// sig1=0 时 1s 计时，sig1=1 时 0.01s 计时，sig2=2 时 0.001 秒计时。
assign clk1=      （15）   ;//根据 sig1 选择时钟，用条件运算符（？：）实现
always@(posedge clk1,negedge KEY[0])
begin
 //计时逻辑，略
end
endmodule
```

三、程序改错题

(1)	output [6:0] HEX4,HEX3,HEX2,HEX1,HEX0;
(2)	always@(posedge clk1,posedge SW[0]) 或 always@(posedge clk1)
(3)	**else hex={hex[6:0],hex[34:7]};**
(4)	module adder1(SW,CLOCK_50, HEX2,HEX1,HEX0);
(5)	**tempa<=ina[7:4]; tempb<=inb[7:4]**
(6)	{cout,sum[7:4]}=tempa+tempb+firstc
(7)	**default:out=1'bx;**
(8)	HEX1=show（sum/10）;HEX0=show（sum%10）;删除
(9)	Function【6:0】 show；
(10)	**show=out；**

1、（共 6 分）下面代码是“HELLO”在 5 个七段管上移位的代码，SW[0]为同步控制信号，SW[0]=1 进行循环左移，SW[0]=0 为循环右移。修改错误代码。

```
module lab4_1(SW,CLOCK_50, HEX4,HEX3,HEX2,HEX1,HEX0);
input[1:0] SW;
input CLOCK_50;//板上的 50M 时钟
**********错误（1）**********
output reg[6:0] HEX4,HEX3,HEX2,HEX1,HEX0;
**********错误（1）**********
reg[34:0] hex=35'b0001001_0000110_1000111_1000111_1000000;
divclk1hz c1(CLOCK_50,clk1);//分频模块，由 50M 时钟产生 1hz 时钟
assign { HEX4,HEX3,HEX2,HEX1,HEX0}=hex;
**********错误（2）**********
```

```
always@(posedge clk1,SW[0])
**************错误（2）**************


begin
    if(SW[0])    hex={hex,hex[55:49]};
    **************错误（3）**************
    else         hex={hex[6:0],hex};
    **************错误（3）**************
end
endmodule
```

2、（共 6 分）下面是一个 2 级流水线设计的 8 为加法器。SW[16:9]和 SW[8:1]为两个 8 为输入操作数，SW[0]为前级的进位。HEX2-HEX0 以 16 进制形式显示最后结果。CLOCK_50 为 50M 输入时钟。修改错误的代码。

```
**************错误（4）**************
module adder1(SW,CLOCK_50, HEX2,HEX1,HEX0,clk);
**************错误（4）**************
input[16:0] SW;   input CLOCK_50;   output reg[6:0] HEX2,HEX1,HEX0;
reg[3:0] tempa,tempb,firsts; //tempa 和 tempb 为第 1 级寄存器，缓存未参与运算的高 4 位 a、b 的数据
reg firstc;//firstcz 为第 1 级寄存器，缓存第一级运算的进位
always @(posedge clk)
**************错误（5），漏掉语句，请补齐**************
begin   {firstc,firsts}=ina[3:0]+inb[3:0]+cin;
**************错误（5）**************
end
always @(posedge clk)
begin
**************错误（6）**************
    {cout,sum}=tempa+tempb+firstc;
**************错误（6）**************
    sum[3:0]=firsts;
end
divclk1hz   d1(CLOCK_50,clk);//分频电路 clk 为 1hz 时钟
```

```
endmodule
```

3、(共 2 分)下面是一个组合逻辑模块，功能是 4 选 1 的数据选择器，按要求进行修改。

```
module mux4_1(out,in0,in1,in2,in3,sel);
   output reg out;
   input in0,in1,in2,in3;
   input[1:0]   sel;
always @( sel,in0,in1,in2,in3)
**********错误（7）有遗漏请补齐**********
case(sel)
        2'b00: out=in0;
        2' b01: out=in1;
        2'b10: out=in2;
        2'b11: out=in3;
   endcase
**********错误（7）**********
endmodule
```

4、（共 6 分）下面为一个计时电路，当 SW=0 时， HEX1,HEX0 显示年份“15”,SW[0]=1 显示秒（0-59）计时。

```
module (CLOCK_50,HEX0,HEX1,KEY,SW);
input[1:0] KEY;//异步复位信号
input[1:0] SW; input CLOCK_50;   output[6:0] HEX1,HEX0;
reg[5:0] sum=0;
divclk1hz   d1(CLOCK_50,CLK);//分频模块，CLK 为 1hz，具体实现略
assign HEX1=(SW[0])?show(sum/10):show(1);
assign HEX0=(SW[0])?show(sum%10):show(5);
**********错误（8）**********
always@(posedge CLK)
begin
   if(!KEY[0]) sum=0;
   else   if(SW[0])
           begin
                if(sum==59) sum=0; else   sum=sum+1;
```

```
            end
            HEX1=show（sum/10）;HEX0=show（sum%10）;
*************错误（8）**************
end
 *************错误（9）**************
    function show；//函数，七段管输出
    *************错误（9）**************
      input[3:0] in4;
       *************错误（10）,代码遗漏，补齐**************
        case(in4)
              4'H0:out7=7'b1000000;   //0
              4'H1:out7=7'b1111001;   //1
              ……..
              4'H9:out7=7'b0011000; //9
              default:out7=7'b1111111;//不亮
         endcase
        *************错误（10）**************
      endfunction
endmodule
```

四、程序设计题（共 2 题，共 20 分）

1、（10分）设计一个BCD码计数器，模24（计数值0-23）。

```
module BCD_count(clk，reset，S1，S0，J0);//模块声明
input clk,reset;
output reg[3:0] S1,S0;
output reg J0;

always@(posedge clk,posedge reset)
begin
        if(S1<2)
              begin
                 if(S0==9)
                          begin S0<=0;   S1<=S1+1; J0<=0end
                 else begin S0<=S0+1;J0<=0;
              end
        else//S1=2
```

```
            bgein
               if(S0==3) begin S0<=0;S1<=0;J0<=1;end
               else begin S0<=S0+1;J0<=0;end
            end
end
endmodule
```

2、（10分）在计算机和数据通信中，对串行数据流的定界是一项基本操作。如果某个串行数据流x,定界符号为“1001”，请用状态机设计模块实现该定界符的检测。

```
module dect(clk，x，reset，y);
module dect(clk，x，reset，y);
input clk,x,reset;
output reg y;
reg[4:0] state;
parameter s0=5’b00001, s1=5’b00010, s2=5’b00100, s3=5’b01000, s4=5’b10000;

always@(posedge clk,posedge reset)
begin
if(reset) state<=s0;
else
    case(state)
       s0:begin if(x)   state<=s1;   else     state<=s0;   y<=0;end
       s1: begin if(x)   state<=s1;   else     state<=s2;   y<=0;end
       s2: begin if(x)   state<=s1;   else     state<=s3;   y<=0;end
       s3: begin if(x)   state<=s4;   else     state<=s0;   y<=0;end
       s4: begin if(x)   state<=s1;   else     state<=s0;   y<=1;end
       default: begin state<=s0; y<=0;end
end
endmodule
```