数字IC设计经典笔试题之【verilog篇】

**Verilog**

# 1：HDL语言的层次概念？

HDL语言是分层次的、类型的，最常用的层次概念有系统与标准级、功能模块级，行为级，寄存器传输级和门级。

系统级，算法级，RTL级(行为级)，门级，开关级

# 2：设计一个自动饮料售卖机，饮料10分钱，硬币有5分和10分两种，并考虑找零。

a.画出fsm（有限状态机）

b.用verilog编程，语法要符合FPGA设计的要求

c.设计工程中可使用的工具及设计大致过程？

设计过程：

a、首先确定输入输出，A=1表示投入10分，B=1表示投入5分，Y=1表示弹出饮料，Z=1表示找零。

b、确定电路的状态，S0表示没有进行投币，S1表示已经有5分硬币。

c、画出状态转移图。

module sell(clk,rst,a,b,y,z);

input clk,rst,a,b;

output y,z;

parameter s0=0,s1=1;

reg state,next\_state;

always@(posedge clk)

begin

if(!rst)

state<=s0;

else

state<=next\_state;

end

always@(a or b or cstate)begin

y=0;z=0;

case(state)

s0: if(a==1&&b==0) begin

next\_state=s1;

end

else if(a==0&&b==1) begin

next\_state=s0; y=1;

end

else begin

next\_state=s0;

end

s1: if(a==1&&b==0)begin

next\_state=s0;y=1;

end

else if(a==0&&b==1)begin

next\_state=s0; y=1;z=1;

end

else begin

next\_state=s0;

end

default: next\_state=s0;

endcase

end

endmodule

# 3：用D触发器做个二分频的电路？画出逻辑电路？

module div2(clk,rst,clk\_out);

input clk,rst;

output reg clk\_out;

always@(posedge clk)begin

if(!rst)begin

clk\_out <=0;

end

else begin

clk\_out <=~ clk\_out;

end

end

endmodule

现实工程设计中一般不采用这样的方式来设计，二分频一般通过DCM来实现。通过DCM得到的分频信号没有相位差,或者是从Q端引出加一个反相器。

# 4：用你熟悉的设计方式设计一个可预置初值的7进制循环计数器,15进制的呢？

module counter7(clk,rst,load,data,cout);

input clk,rst,load;

input [2:0] data;

output reg [2:0] cout;

always@(posedge clk)begin

if(!rst)begin

cout<=3’d0;

end

else if(load)begin

cout<=data;

end

else if(cout>=3’d6)begin

cout<=3’d0;

end

else begin

cout<=cout+3’d1;

end

end

endmodule

# 5：用Verilog或VHDL写一段代码,实现消除一个glitch（毛刺）？

将传输过来的信号经过两级触发器就可以消除毛刺。（这是我自己采用的方式：这种方式消除毛刺是需要满足一定条件的，并不能保证一定可以消除）

module glitch(clk,data,q\_out)

input clk,data;

output reg q\_out;

reg q1;

always@(posedge clk)begin

q1<=data;

q\_out<=q1;

end

endmodule

# 6：画出DFF的结构图,用verilog实现之。

module dff(clk,d,qout);

input clk,d;

output qout;

reg qout;

always@(posedge clk)begin

if(!reset)begin

qout<=0;

end

else begin

qout<=d;

end

end

endmodule

# 7：请用HDL描述四位的全加法器、5分频电路。

module adder4(a,b,ci,s,co);

input ci;

input [3:0] a,b;

output co;

output [3:0] s;

assign {co,s}=a+b+ci;

endmodule

// 这个5分频不对，实际是10分频

module div5(clk,rst,clk\_out);

input clk,rst;

output clk\_out;

reg [3:0] count;

always@(posedge clk)begin

if(!rst) begin

count<=0;

clk\_out=0;

end

else if(count==3’d5)begin

count<=0;

clk\_out=~clk\_out;

end

else begin

count<=count+1;

end

end

endmodule

# 8：实现奇数倍分频且占空比为50%的情况。

module div7 ( clk, reset\_n, clkout );

input clk,reset\_n;

output clkout;

reg [3:0] count;

reg div1;

reg div2;

always @( posedge clk )

begin

if ( ! reset\_n )

count <= 3'b000;

else

case ( count )

3'b000 : count <= 3'b001;

3'b001 : count <= 3'b010;

3'b010 : count <= 3'b011;

3'b011 : count <= 3'b100;

3'b100 : count <= 3'b101;

3'b101 : count <= 3'b110;

3'b110 : count <= 3'b000;

default :

count <= 3'b000;

endcase

end

always @( posedge clk )

begin

if ( ! reset\_n )

div1 <= 1'b0;

else if ( count == 3'b000 )

div1 <= ~ div1;

end

always @( negedge clk )

begin

if ( ! reset\_n )

div2 <= 1'b0;

else if ( count == 3'b100 )

div2 <= ~ div2;

end

assign clkout = div1 ^ div2;

endmodule

# 9：用VERILOG或VHDL写一段代码，实现10进制计数器。

module counter10(clk,rst,count);

input clk,rst;

output [3:0] count;

reg [3:0] count;

always@(posedge clk)begin

if(!rst)begin

count<=0;

end

else if(count>=4’d9)begin

count<=0;

end

else begin

count<=count+1;

end

end

endmodule

**-END-**