实验二 基本功能模块程序设计

**一、实验目的**

1. 掌握Verilog语言框架，编程及调试的方法。
2. 熟悉Verilog的基本语法。
3. 掌握Verilog中时序模块电路的设计方法。
4. 熟悉Verilog中层次结构的设计方法。

**二、实验内容**

1. 利用case语句、always语句等完成编码器、译码器等功能。
2. 利用always语句完成一个触发器电路模块（使能端、复位）的设计。
3. 利用always语句完成寄存器和移位寄存器（循环移位）电路模块的设计。
4. 完成各种数字（个人学号末两位）进制的计数器。
5. 掌握以上电路的程序结构和风格。
6. 观察和分析仿真波形，注重输入输出之间的时序关系。

**三、实验要求**

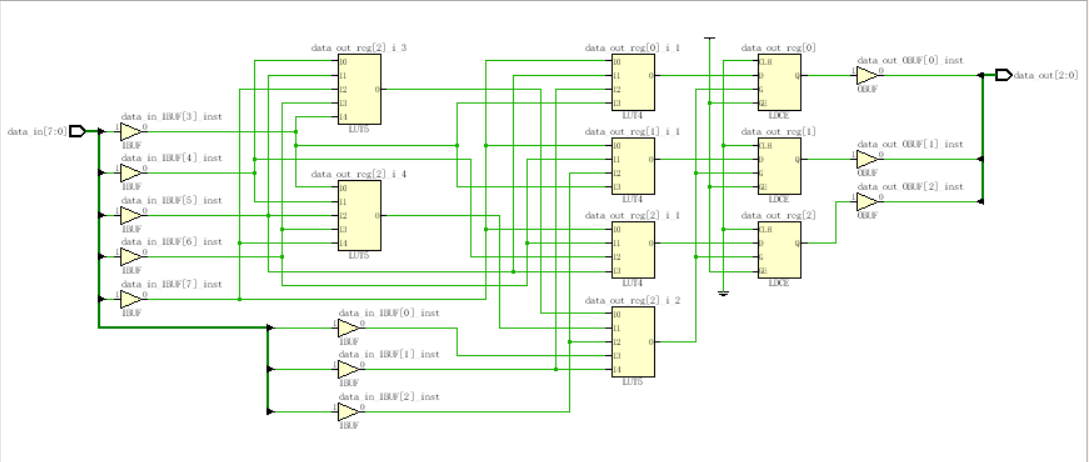
1. 画出模块的电路图。
2. 分析电路的仿真波形。
3. 记录设计和调试过程。

**四、实验代码及结果**

**1.3-8编码器、译码器：**

**编码器：**

**电路图：**



**仿真图：**



**器件源码：**

module encoder(data\_in,data\_out);

input [7:0]data\_in;

output [2:0]data\_out;

reg [2:0]data\_out;

always@(data\_in)

begin

case(data\_in)

8'b00000001:data\_out=3'b000;

8'b00000010:data\_out=3'b001;

8'b00000100:data\_out=3'b010;

8'b00001000:data\_out=3'b011;

8'b00010000:data\_out=3'b100;

8'b00100000:data\_out=3'b101;

8'b01000000:data\_out=3'b110;

8'b10000000:data\_out=3'b111;

endcase

end

endmodule

**仿真源码：**

module sim\_en83();

reg [7:0]in;

wire [2:0]out;

initial begin

in=8'b00001000;

#50

in=8'b00100000;

#50

in=8'b00000100;

#50

in=8'b10000000;

#50

in=8'b00000000;

#50

in=8'b00000001;

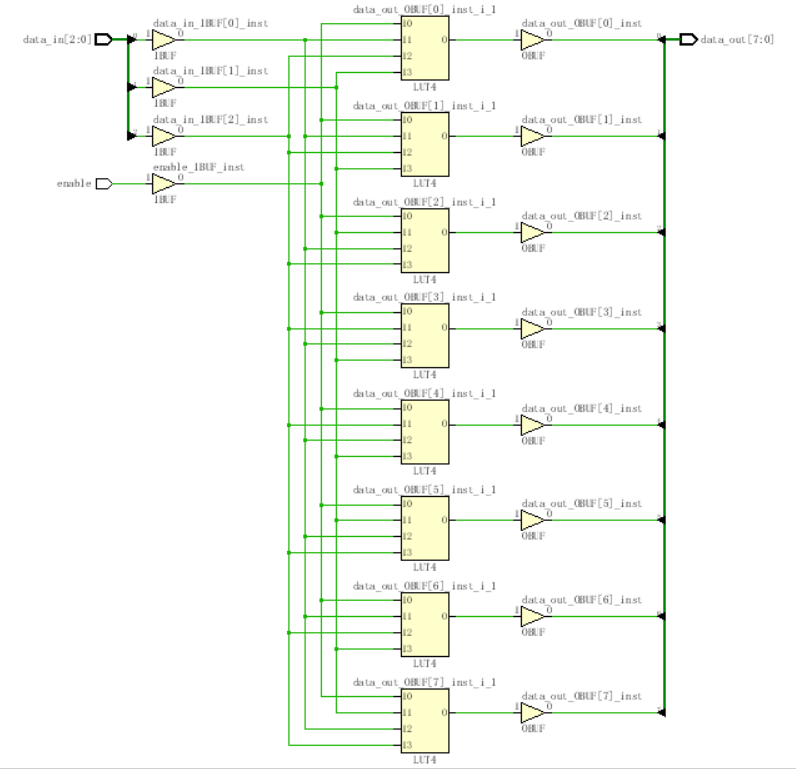
end

encoder en(in,out);

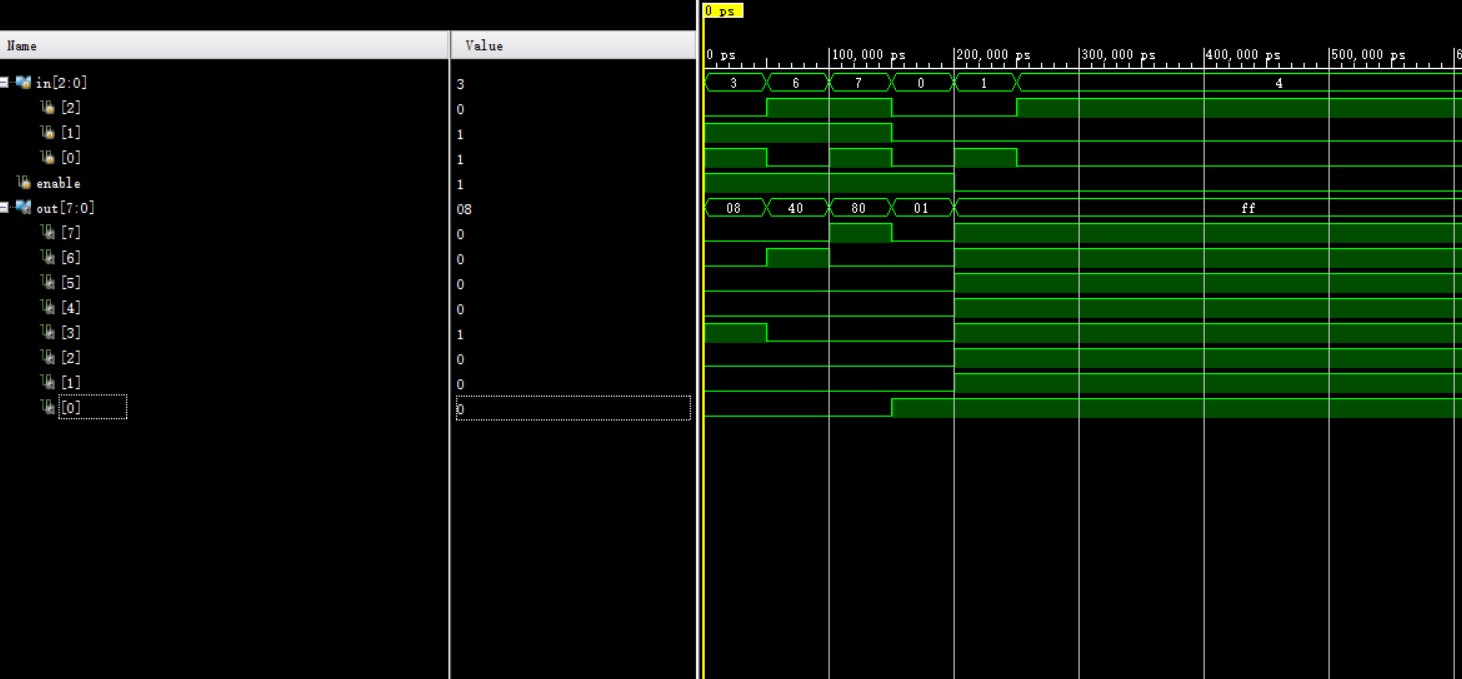
endmodule

**译码器：**

**电路图：**



**仿真波形图：**



**器件源码：**

module decoder(data\_in,enable,data\_out);

input [2:0] data\_in;

input enable;

output [7:0]data\_out;

reg [7:0]data\_out;

always@(data\_in)

begin

if(enable==1)

case(data\_in)

3'b000: data\_out = 8'b00000001;

3'b001: data\_out = 8'b00000010;

3'b010: data\_out = 8'b00000100;

3'b011: data\_out = 8'b00001000;

3'b100: data\_out = 8'b00010000;

3'b101: data\_out = 8'b00100000;

3'b110: data\_out = 8'b01000000;

3'b111: data\_out = 8'b10000000;

default:data\_out = 8'bxxxxxxxx;

endcase

else

data\_out=8'b11111111;

end

endmodule

**仿真源码：**

module sim\_2();

reg [2:0] in;

reg enable;

wire[7:0] out;

initial begin

enable=1;

in=3'b011;

#50

enable=1;

in=3'b110;

#50

enable=1;

in=3'b111;

#50

enable=1;

in=3'b000;

#50

enable=0;

in=3'b001;

#50

enable=0;

in=3'b100;

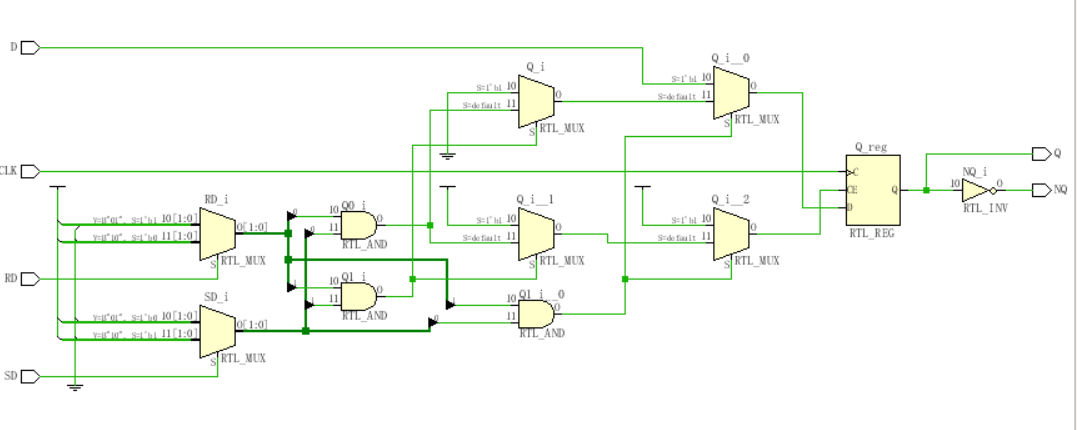
end

decoder de(in,enable,out);

endmodule

**2.D触发器**

**电路图：**



**仿真图：**



**器件源码：**

module dff(CLK ,RD,SD,Q,D,NQ);

input CLK;

input RD;

input SD;

input D;

output Q;

output NQ;

reg Q;

assign NQ=~Q;

always@(posedge CLK)

begin

if(RD==1&&SD==0)

Q<=1;

if(RD==0&&SD==1)

Q<=0;

if(RD==1&&SD==1)

Q<=1'bx;

if(RD==0&&SD==0)

Q<=D;

end

endmodule

**仿真源码：**

module sim\_dff( );

reg RD;

reg SD;

reg D;

reg CLK;

wire Q;

wire NQ;

initial CLK=1'b1;

always begin

#10 CLK<=~CLK;

end

initial begin

RD=1;

SD=0;

D=1;

#30

RD=0;

SD=1;

D=1;

#30

RD=1;

SD=1;

D=1;

#30

RD=0;

SD=0;

D=1;

#30

RD=0;

SD=0;

D=0;

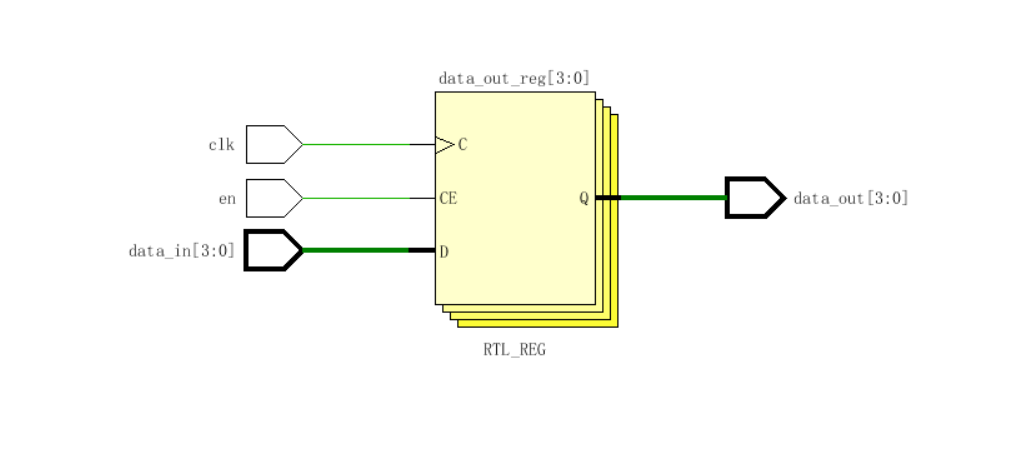
end

dff df(CLK,RD,SD,Q,D,NQ);

endmodule

**3.寄存器和移位寄存器**

**寄存器电路图：**



**仿真图：**



**器件源代码：**

module register(data\_in,en,clk,data\_out );

input[3:0] data\_in;

input en,clk;

output [3:0] data\_out;

reg [3:0] data\_out;

always@(posedge clk)

begin

if(en==1)

data\_out<=data\_in;

end

endmodule

**仿真源代码：**

module sim\_reg();

reg clk,en;

reg [3:0]in;

wire [3:0]out;

initial clk=1'b0;

always #10 clk<=~clk;

initial begin

en=0;

in=4'b0110;

#30

en=1;

in=4'b1110;

#30

en=0;

in=4'b1000;

#30

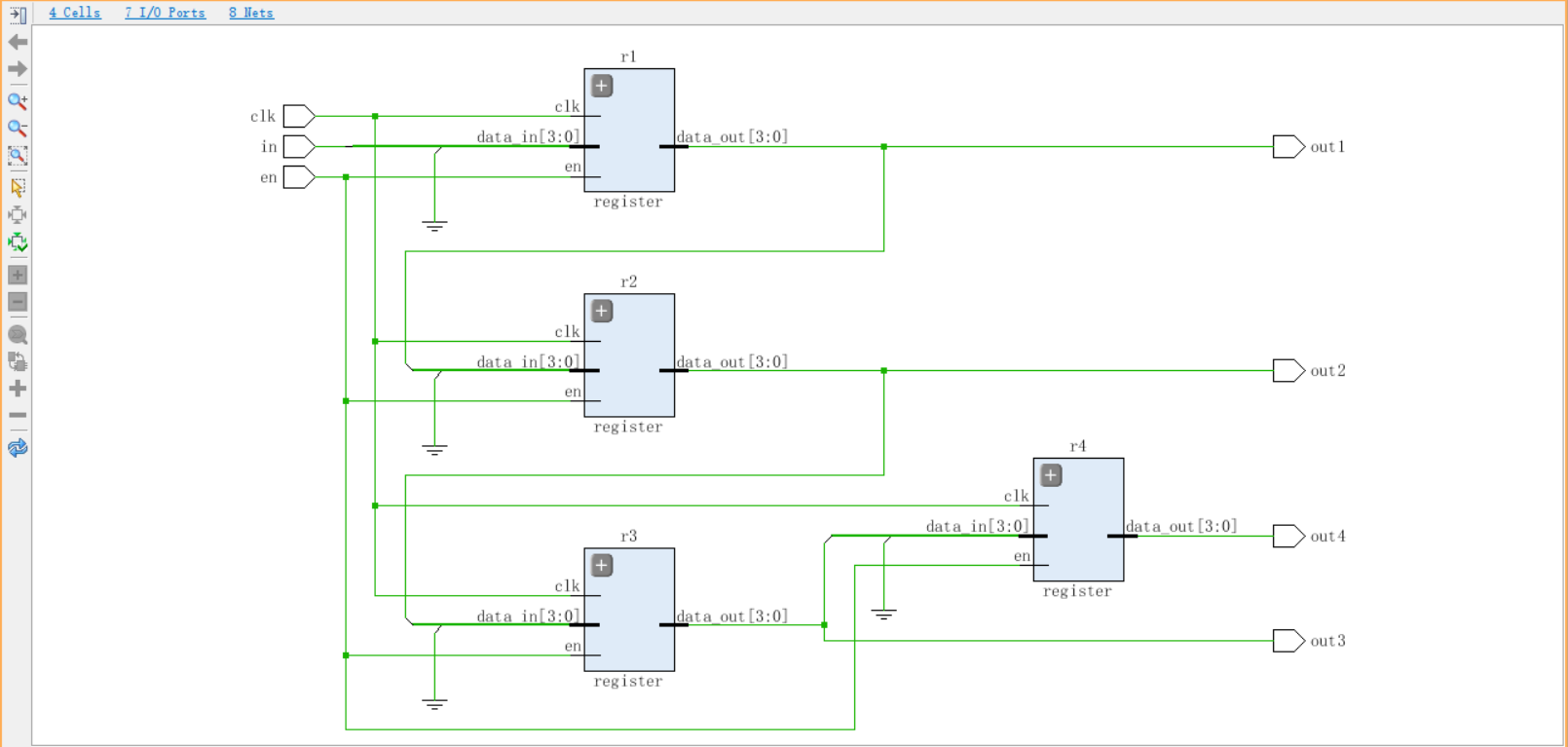
en=1;

in=4'b0011;

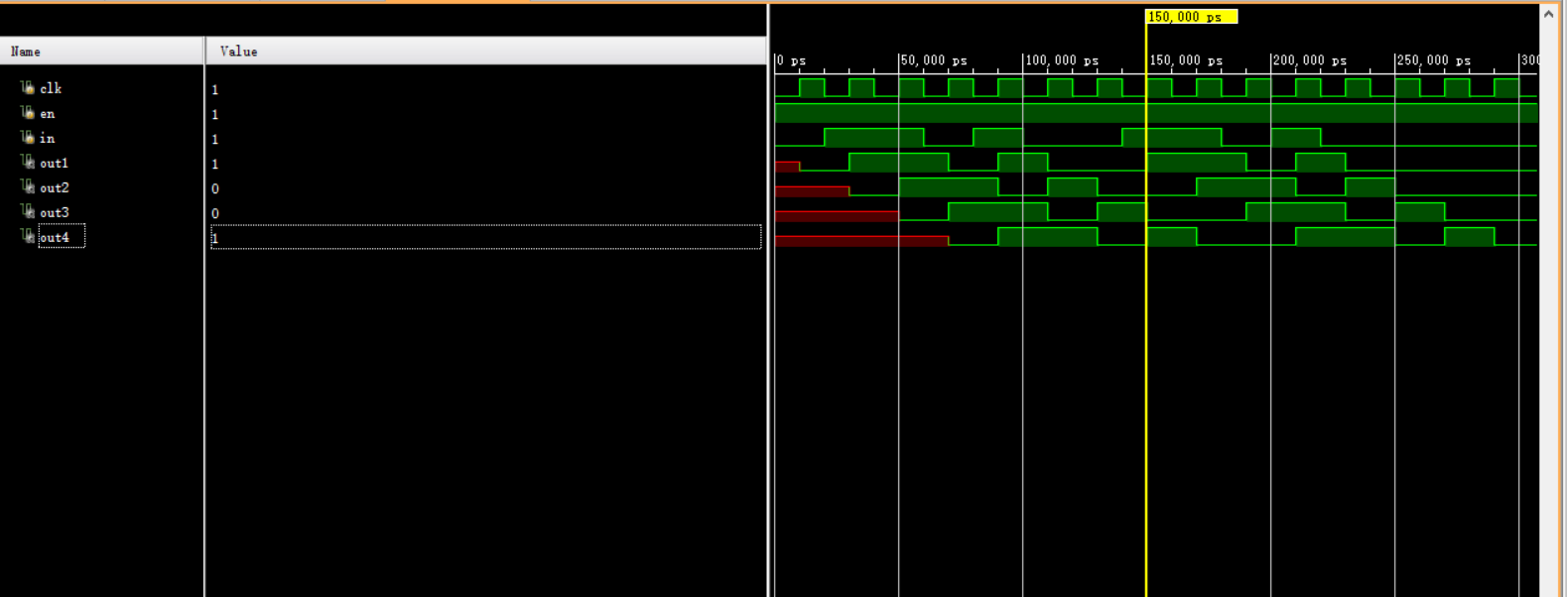
end

register reg2(in,en,clk,out);

endmodule

**移位寄存器电路图：**

**仿真图：**



**器件源代码：**

module fourreg(

in,out1,out2,out3,out4,en,clk

);

input in,en,clk;

output out1,out2,out3,out4;

register r1(in,en,clk,out1);

register r2(out1,en,clk,out2);

register r3(out2,en,clk,out3);

register r4(out3,en,clk,out4);

endmodule

**仿真源代码：**

module sim\_fourreg();

reg clk,en;

reg in;

wire out1,out2,out3,out4;

initial clk=1'b0;

always #10 clk<=~clk;

initial en=1;

initial begin

in=1'b0;

#20

in=1'b1;

#20

in=1'b1;

#20

in=1'b0;

#20

in=1'b1;

#20

in=1'b0;

#20

in=1'b0;

#20

in=1'b1;

#20

in=1'b1;

#20

in=1'b0;

#20

in=1'b1;

#20

in=1'b0;

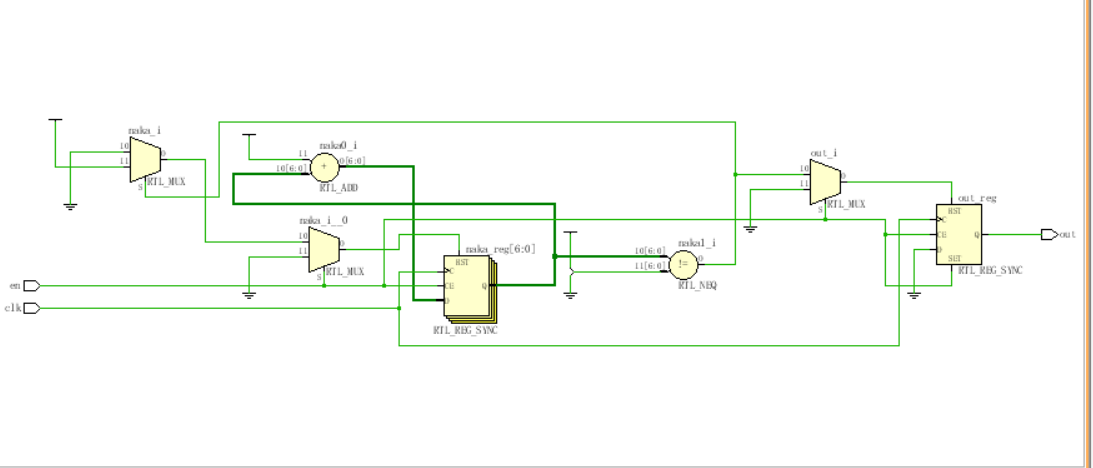
end

fourreg reg2(in,out1,out2,out3,out4,en,clk);

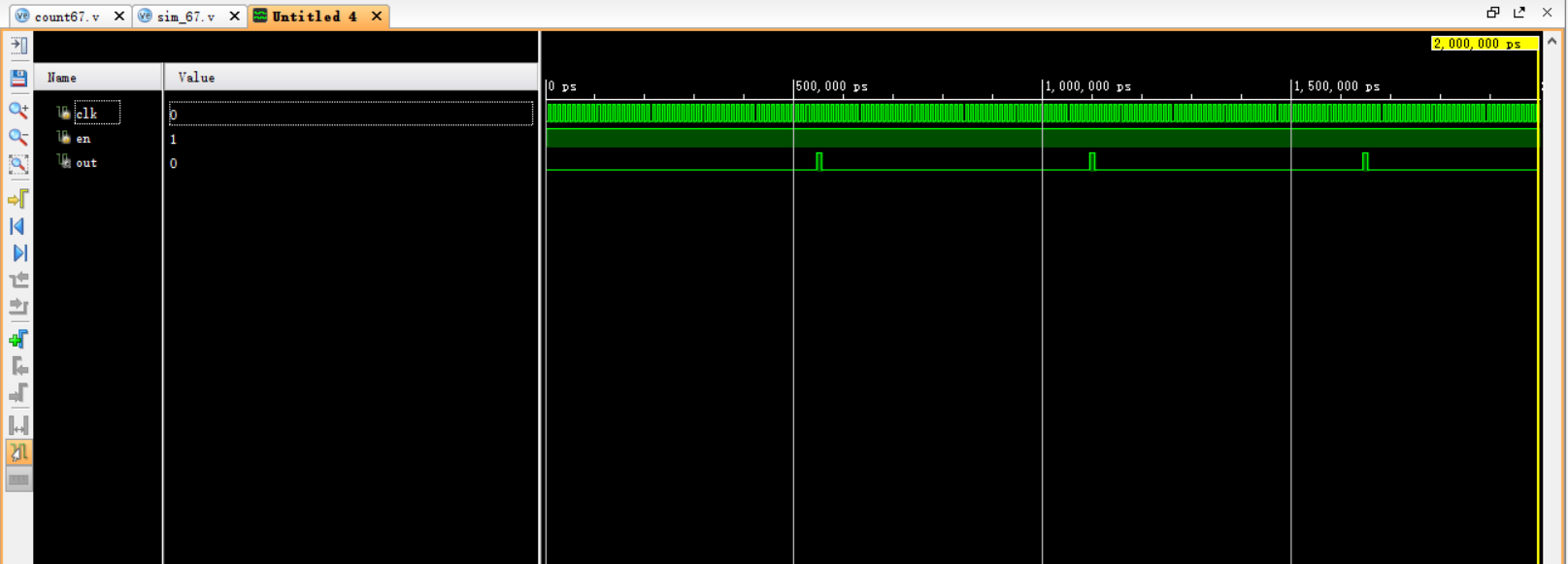
endmodule

**67进制计数器**

**电路图：**



**仿真图：**



**器件源代码：**

module count67(

en,clk,out

);

input en,clk;

output out;

reg out;

reg [6:0]naka;

initial naka<=0;

initial out<=0;

always@(posedge clk)

begin

if(en==1)

begin

if(naka!=7'o66)

begin

naka<=naka+7'o1;

out<=0;

end

else

begin

naka<=7'o0;

out<=1;

end

end

end

**endmodule**

**仿真源代码：**

module sim\_67();

reg clk,en;

wire out;

initial en=1;

initial clk=1'b0;

always #5 clk<=~clk;

count67 c67(

en,

clk,

out

);

endmodule

**五、调试和心得体会**

用Verilog实现了译码器、解码器、触发器、寄存器