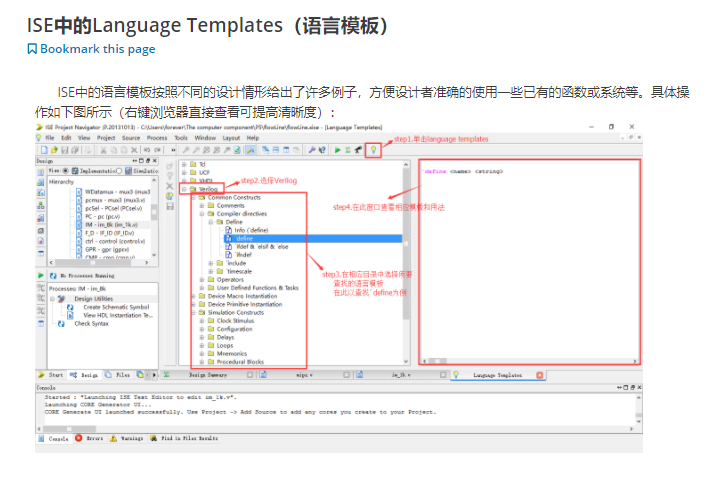
**Verilog HDL**

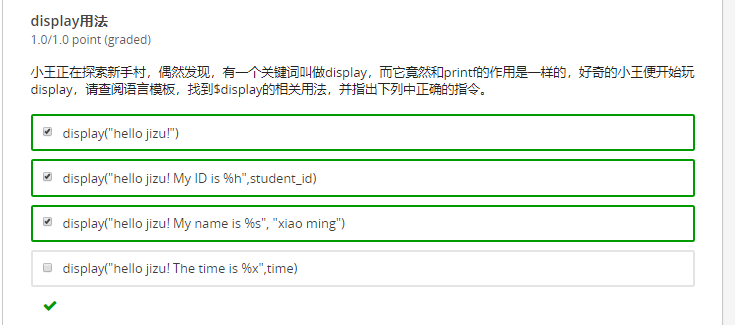
端口设置练习：

****

**注意最后一行定义不要带 “，”**

Language Template:





**$display("<string\_and/or\_variables>", <functions\_or\_signals>);**

**$monitor("<string\_and/or\_variables>", <functions or signals>);**

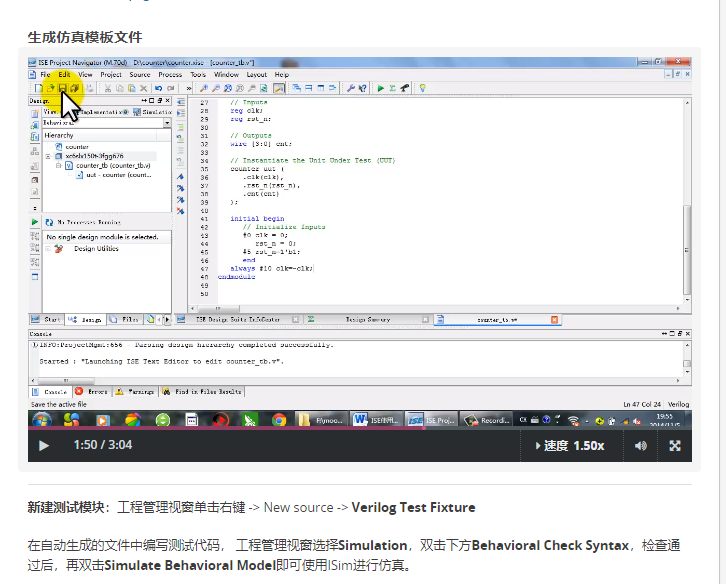


**reg [<memory\_width>] <reg\_name> [<memory\_depth>];**

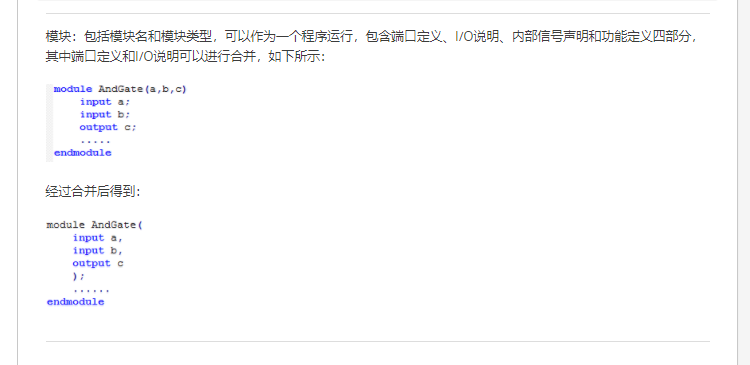
**initial**

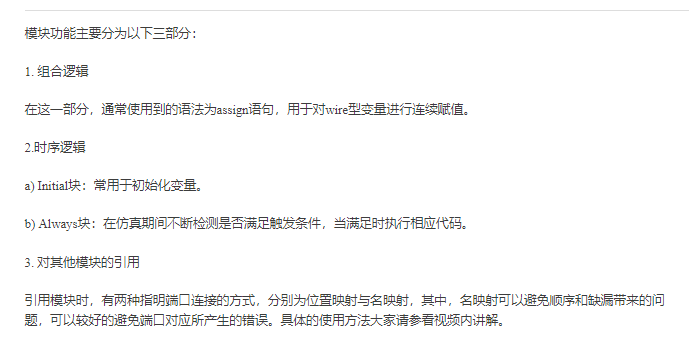
**$readmemh ("<file\_name>", <reg\_name>, <start\_address>, <end\_address>);**

生成仿真文件：

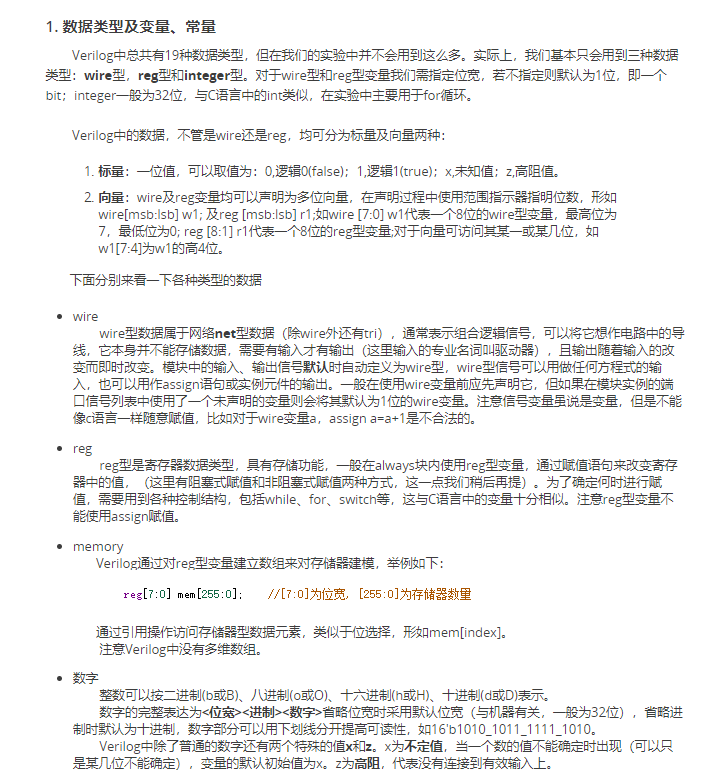


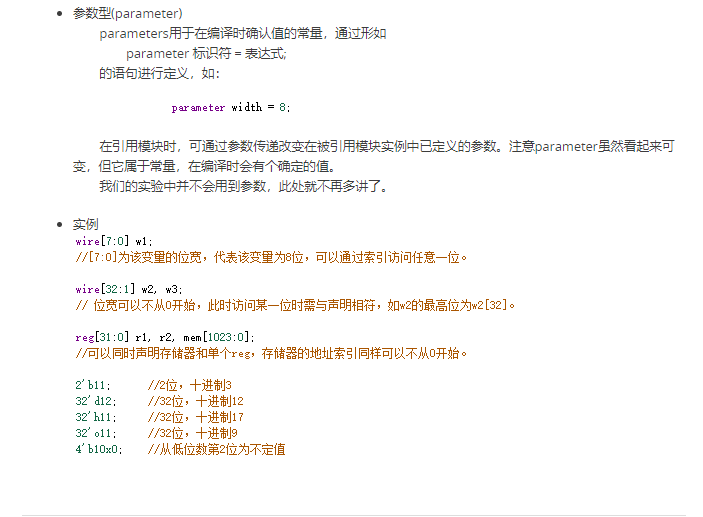
生成模块：





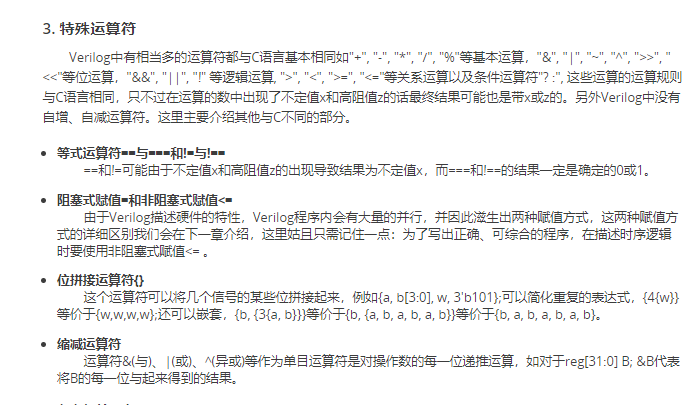
基础语法知识：

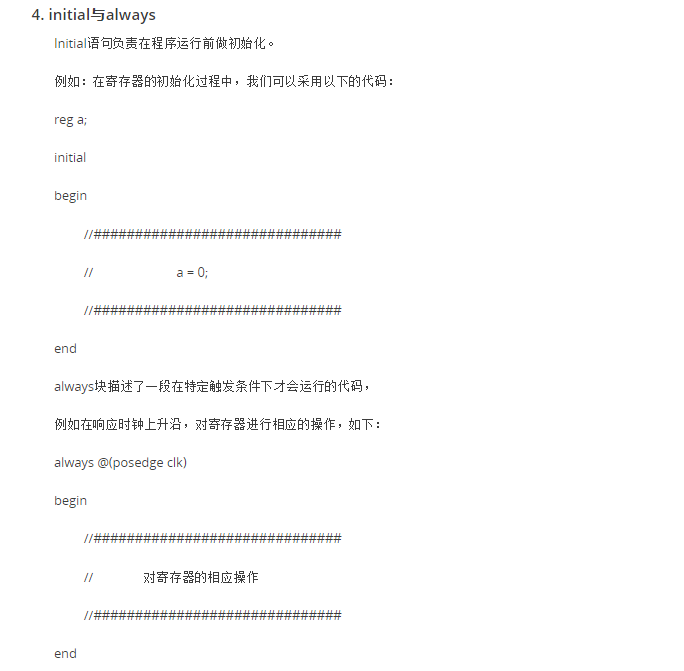


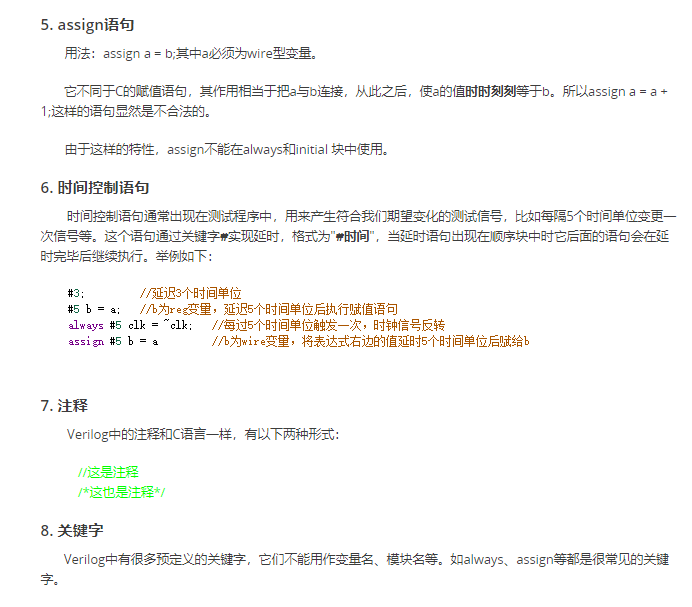


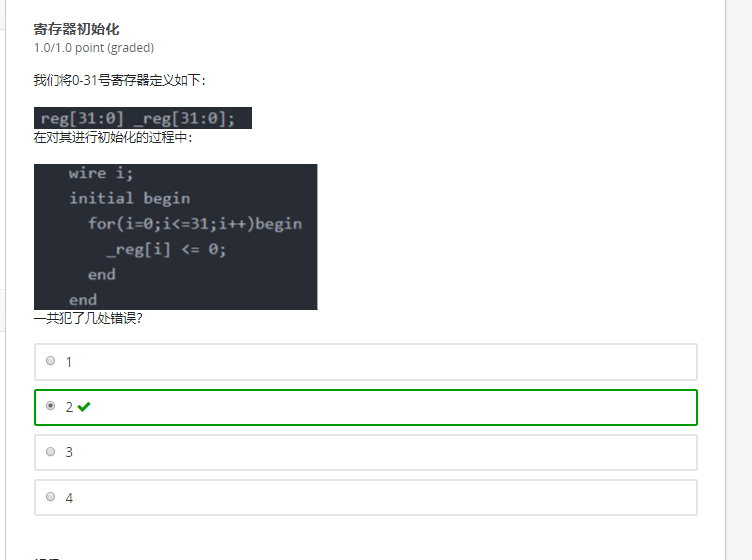












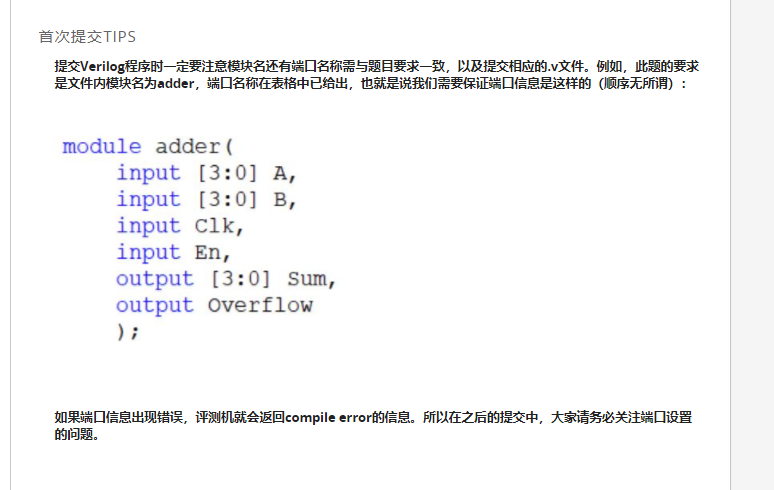
**i作为for的计数应该声明为integer型，然后只能i=i+1**

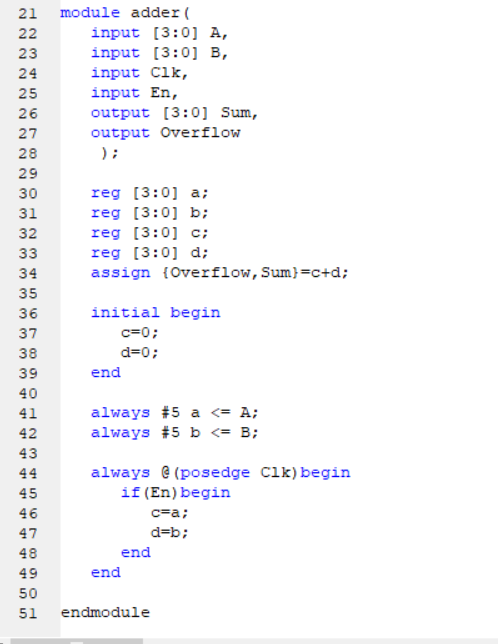


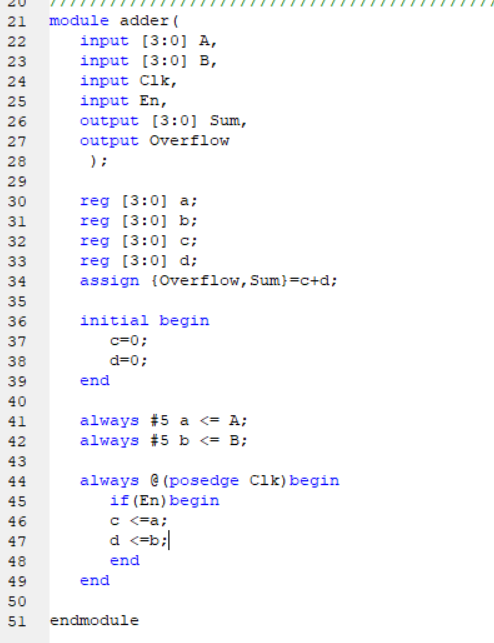
尽可能短

构建加法器：

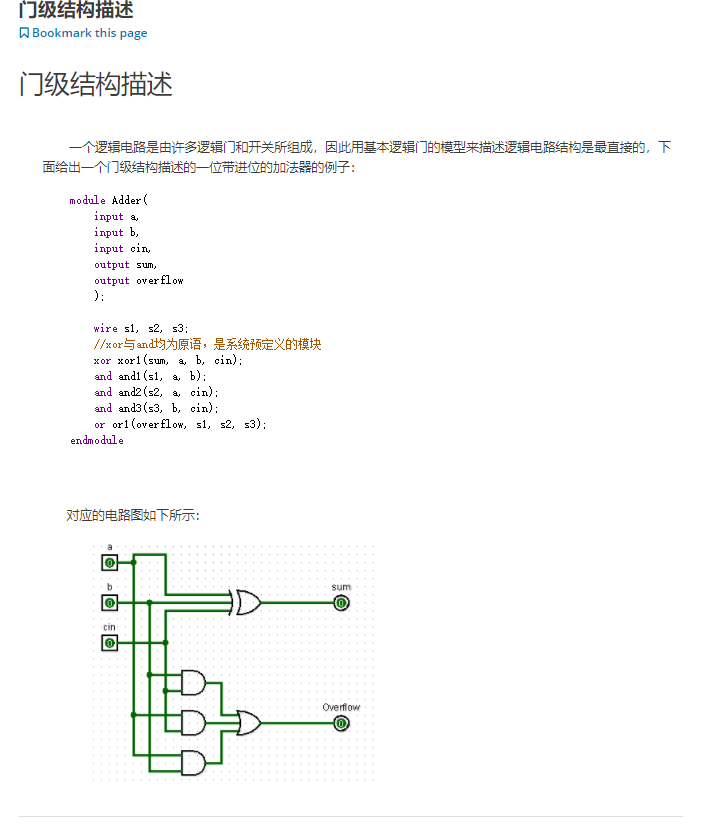




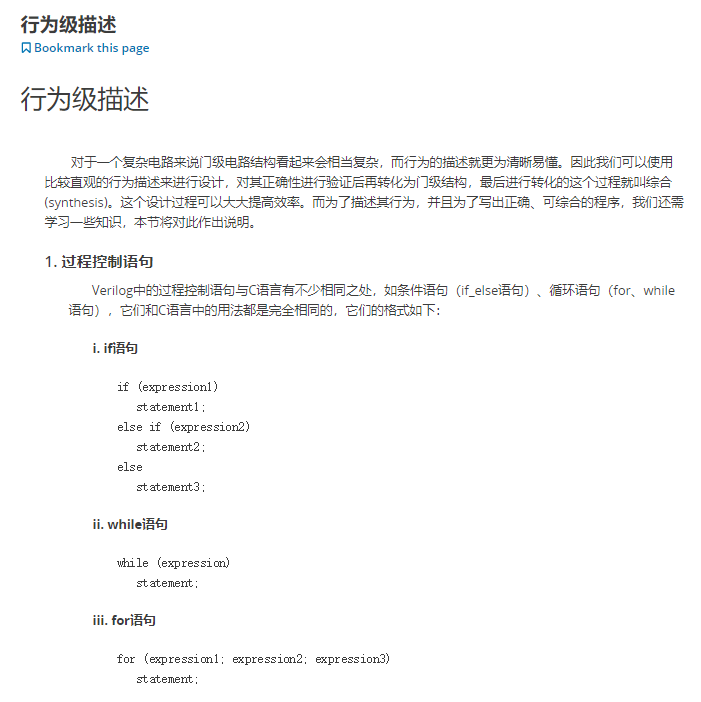


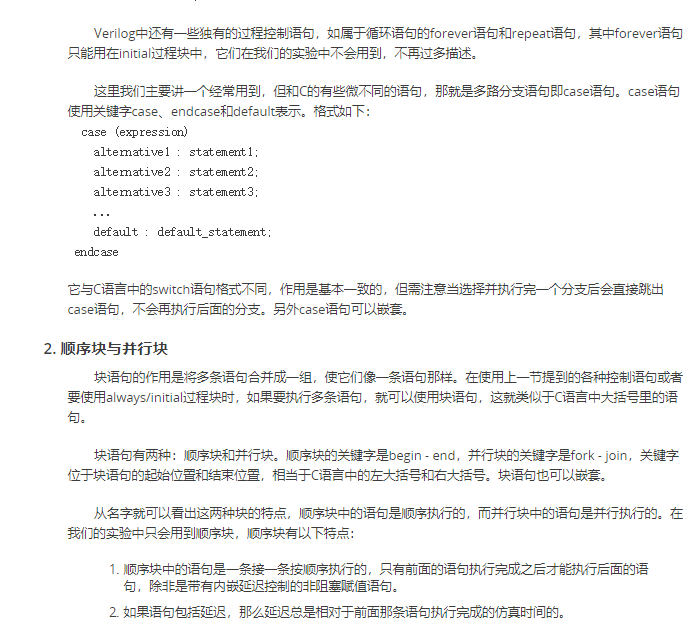


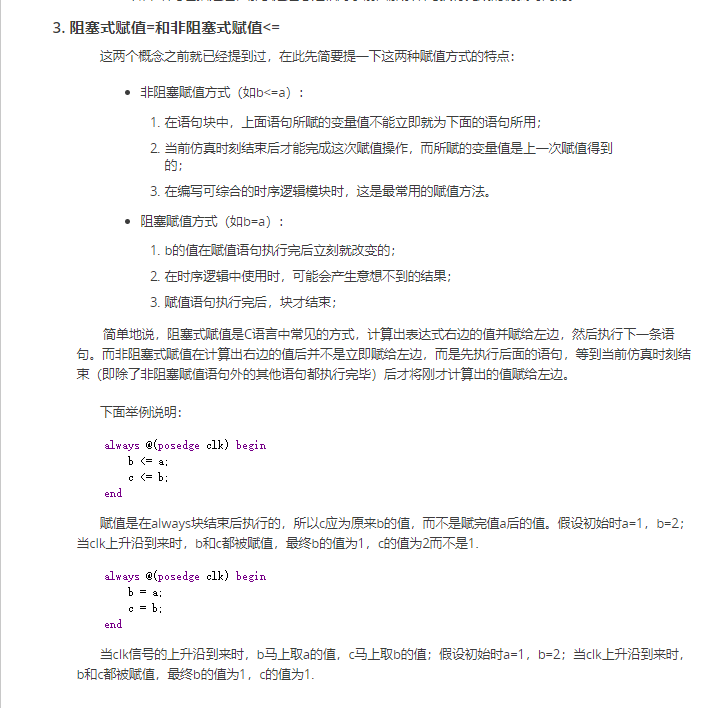
门级结构描述：



行为级描述：

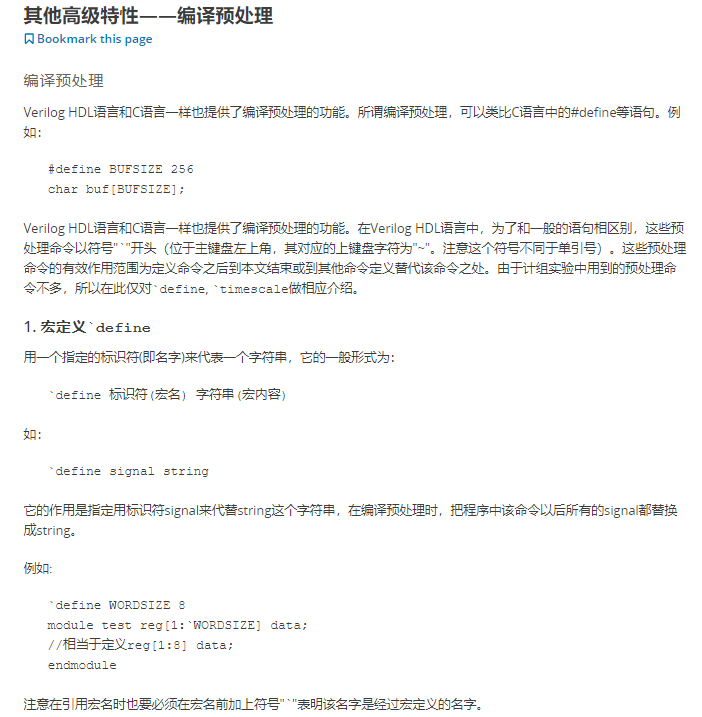


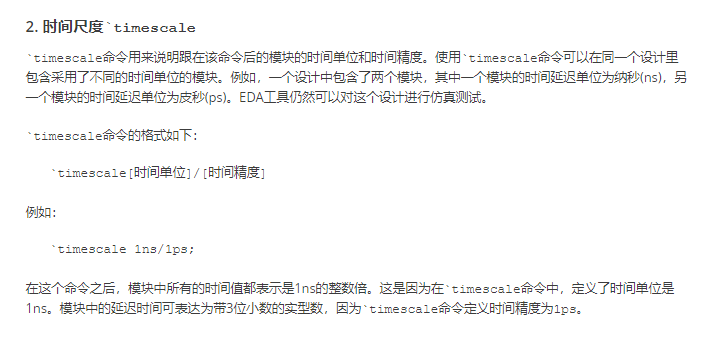




**它与C语言中的switch语句格式不同，作用是基本一致的，但需注意当选择并执行完一个分支后会直接跳出case语句，不会再执行后面的分支。另外case语句可以嵌套。**

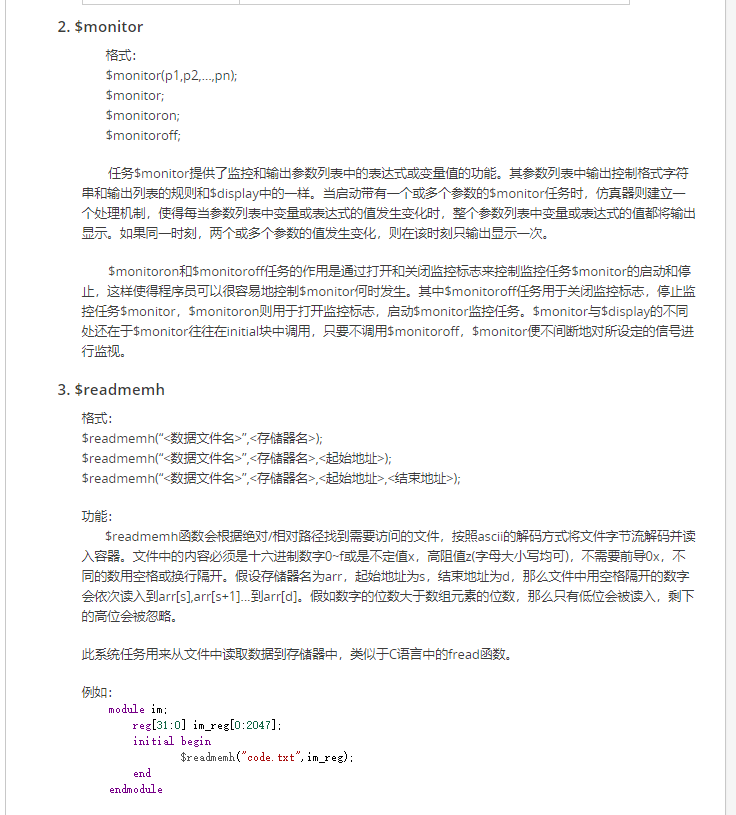
编译预处理：





系统任务：



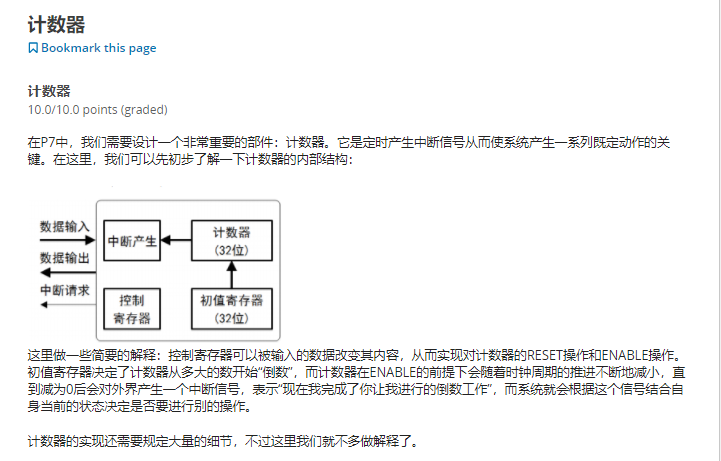




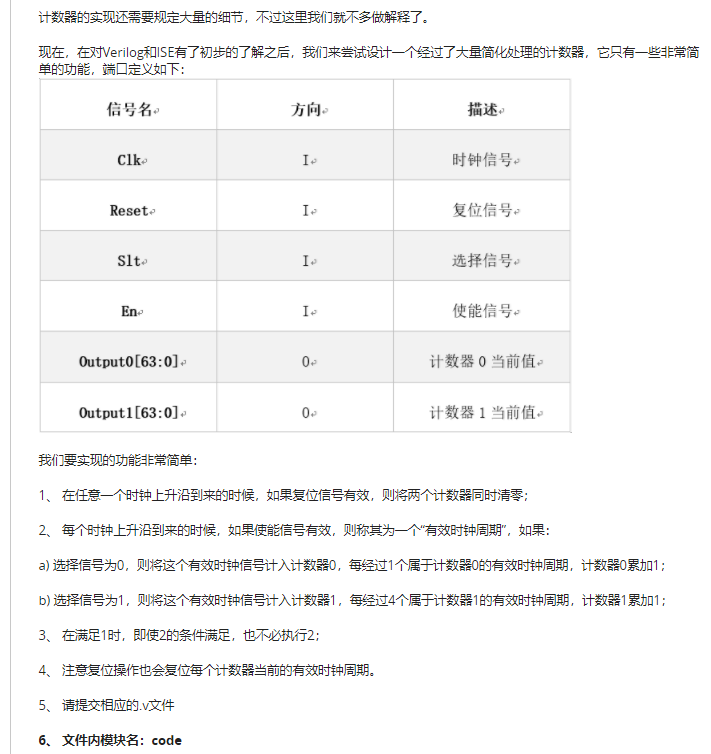
**P5需要从外部文件为设计的CPU读入写好的指令**

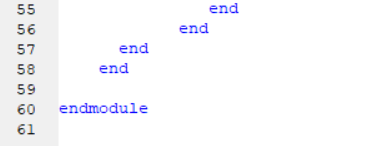
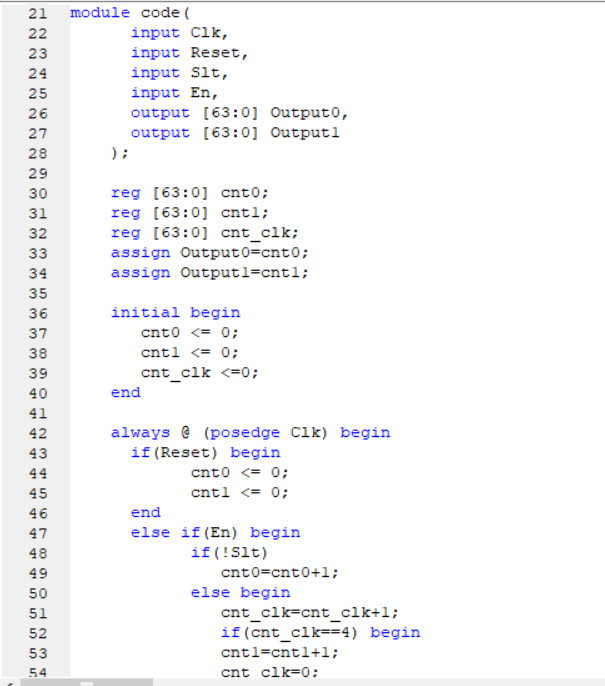


**monitor的输出，找最频繁那个即可**

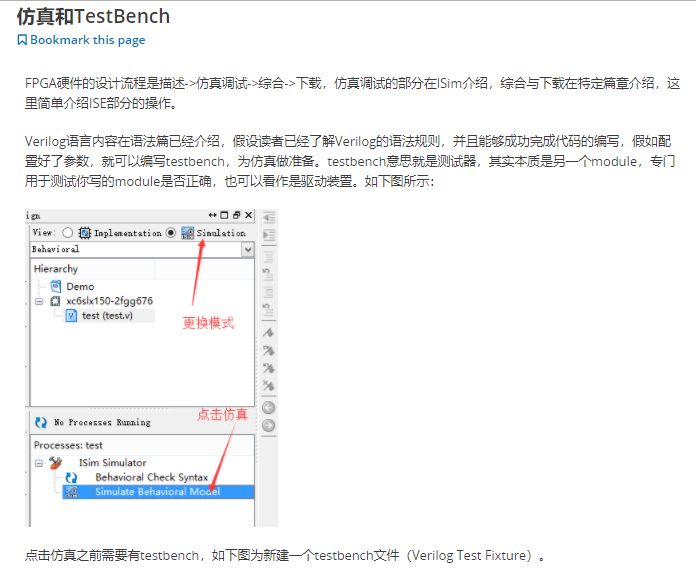


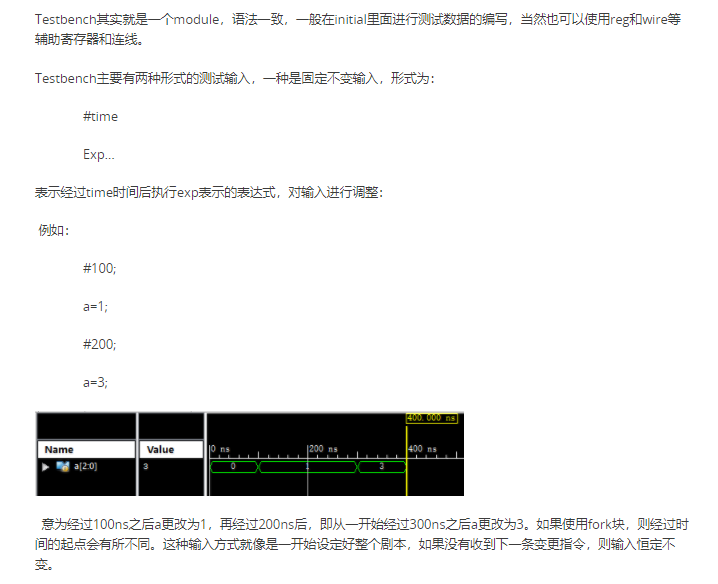
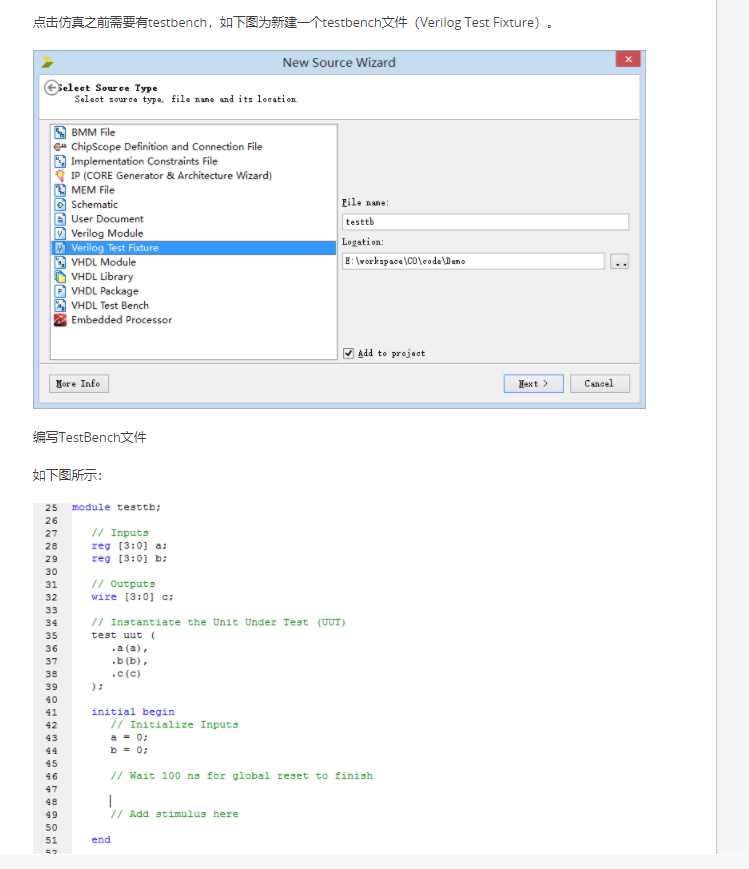
**P7设计计数器**





**ISIM的仿真和测试：**







ISIM的部分报错和解决方案：

**组合逻辑模块全用 =**

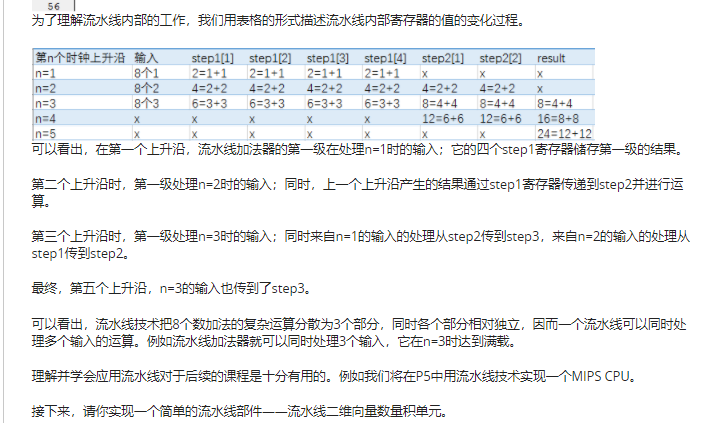
**时序逻辑模块全用 <=**

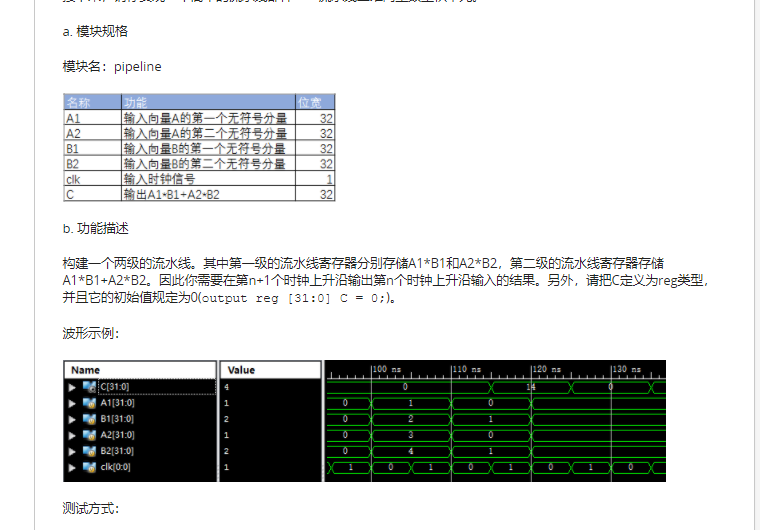
**注意 <=不能分开**

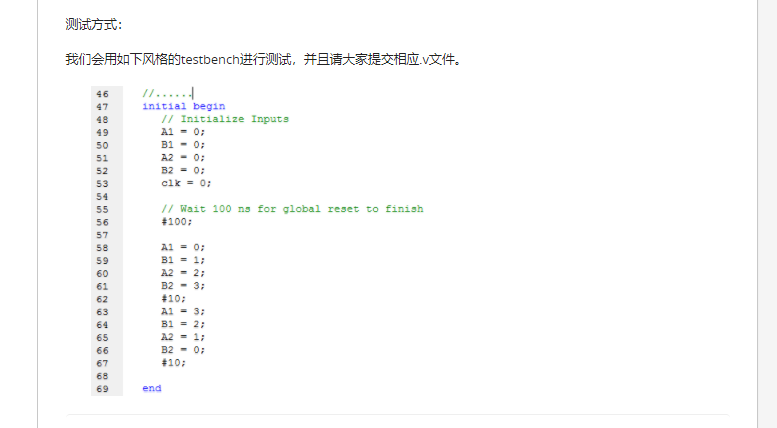
**wire只能被assign赋值，reg只能在always和initial中赋值**

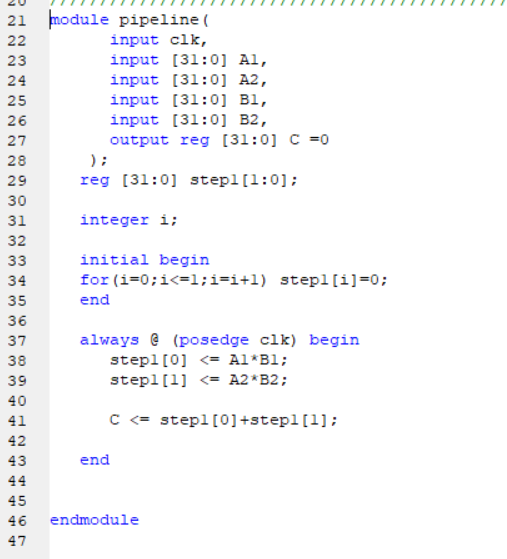
流水线线路初步：

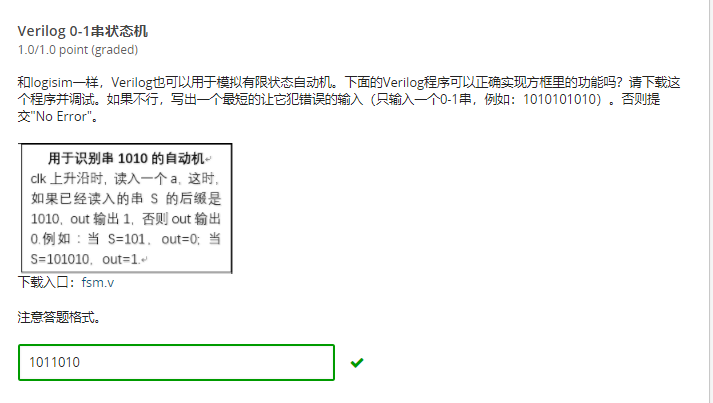


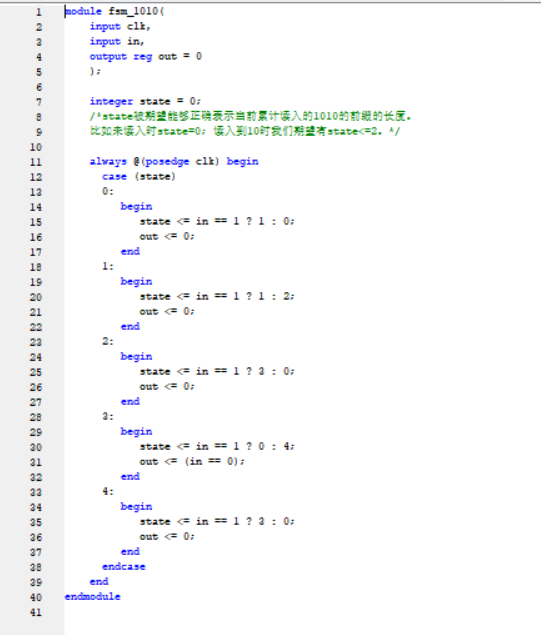




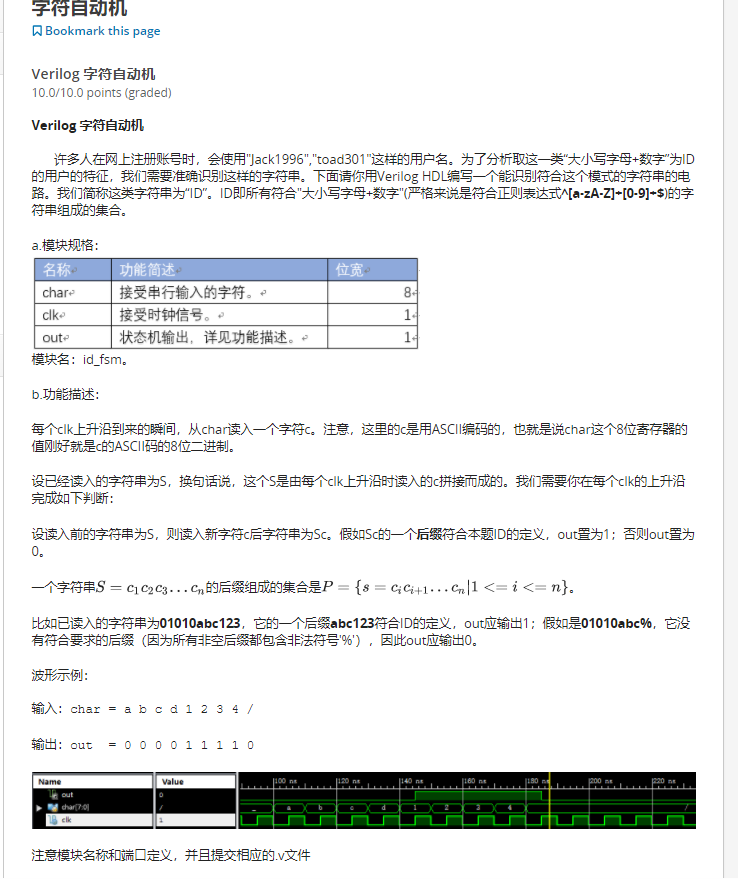






****

**用状态机去检查串**





**module id\_fsm(**

**input [7:0] char,**

**input clk,**

**output reg out=0**

**);**

**reg alpha = 0;**

**always @ (posedge clk) begin**

**if((65<=char && char<=90)||(97<=char && char<=122))begin //如果是字母**

**alpha <=1;**

**out <=0;**

**end**

**else begin**

**if(alpha && char>=48 && char<=57) //如果前面是字母，当前是数字**

**out <= 1;**

**else begin //如果前面不是字母，即使输入数字或者其他都是0**

**alpha <= 0;**

**out <=0;**

**end**

**end**

**end**

**注意设置的寄存器的位数**

**四选一多路选择器：**

**module four\_bit\_selector\_p72(**

**output reg out, //将output设置成reg，输出端口被声明为寄存器类型变量**

**input i0,i1,i2,i3,**

**input s0,s1**

**);**

**//若输入信号改变，则重新计算输出信号out**

**//造成输出信号out重新计算的所有输入信号必须写入always @(…)的电平敏感列表**

**always @ (s1 or s0 or i0 or i1 or i2 or i3)begin**

**case({s1,s0})**

**2'b00:**

**out <= i0;**

**2'b01:**

**out <= i1;**

**2'b10:**

**out <= i2;**

**2'b11:**

**out <= i3;**

**default:**

**out <= 0;**

**endcase**

**end**

**endmodule**

**四位计数器：**

**module four\_bit\_counter\_p73(**

**output reg [3:0] Q = 0, //输出变量Q被定义为寄存器类型，别忘记初始化，不然波形输出xxxx**

**input clock,**

**input clear**

**);**

**always @ (posedge clear or negedge clock)begin**

**if(clear)**

**Q <= 4'b0; //为了能生成诸如触发器一类的时序逻辑，建议使用非阻塞赋值**

**else begin**

**Q <= Q + 1; //Q是一个四位寄存器，计数超过15后会归零，因此模16没有必要**

**end**

**end**

**Endmodule**

**基本RS锁存器：**

**module basic\_RS\_latch(**

**input SDN,RDN,**

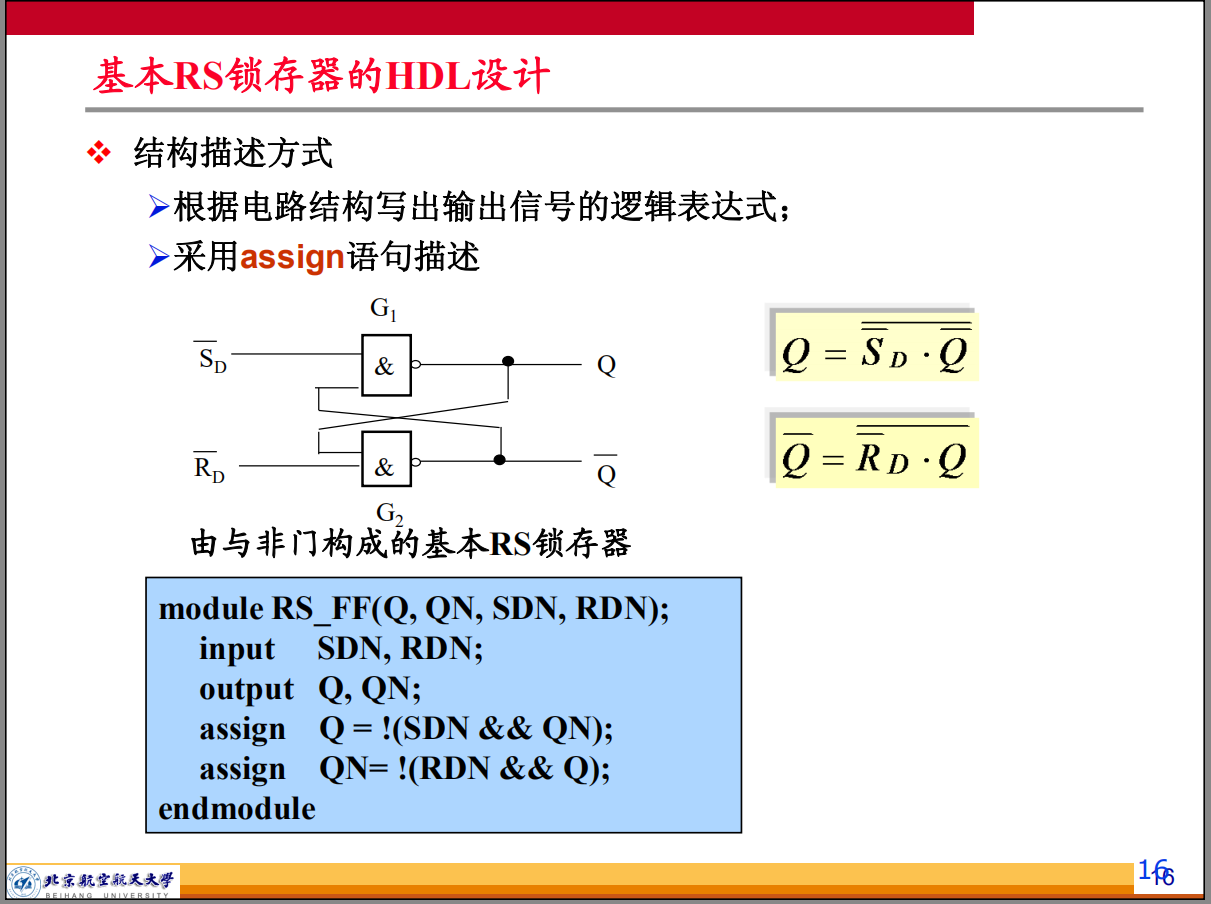
**output Q,QN**

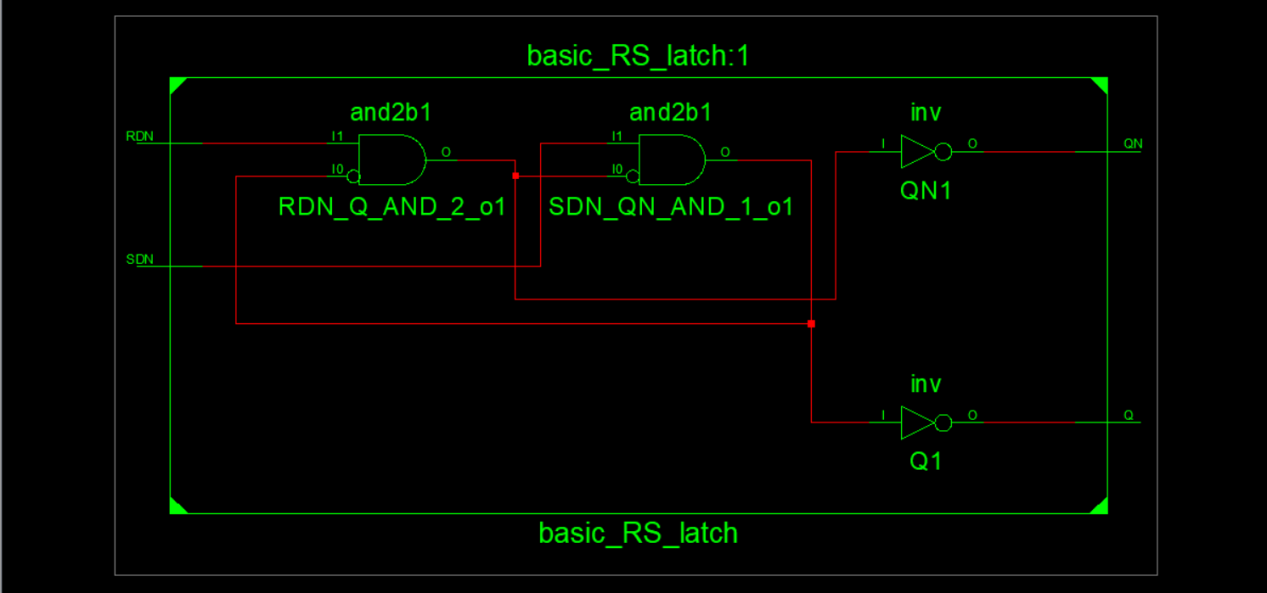
**);**

**assign Q = !(SDN && QN);**

**assign QN = !(RDN && Q);**

**endmodule**





**基本RS锁存器：**

**module basic\_RS\_latch(**

**input SDN,RDN,**

**output Q,QN**

**);**

**assign Q = !SDN || !QN;**

**assign QN = !RDN || !Q;**

**endmodule**

