第8章　硬件描述语言及其应用

§8.1　概述（Verilog HDL）

关于硬件描述语言

- 1983年由GDA公司开发
- 1989年GDA公司被Cadence公司收购
- 1995年，Verilog HDL被制定为IEEE标准
- 用文本描述数字电路结构和行为的语言；
- 语法与C语言相似；
- 推荐参考书：The Verilog Hardware Description Language

第8章　硬件描述语言及其应用

§8.1　概述（Verilog HDL）

关于硬件描述语言

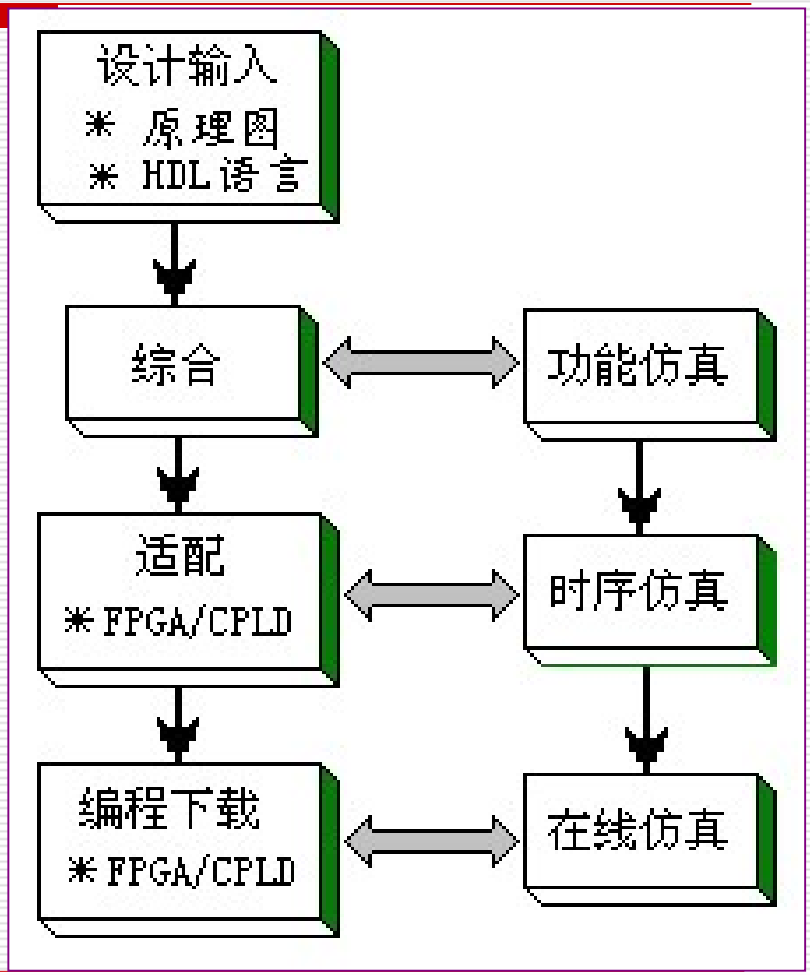
数字系统设计过程：

1、设计输入：原理图或HDL；

2、综合：将原理图/HDL转换为电路网表；

3、适配：将电路网表配置于具体目标器件；

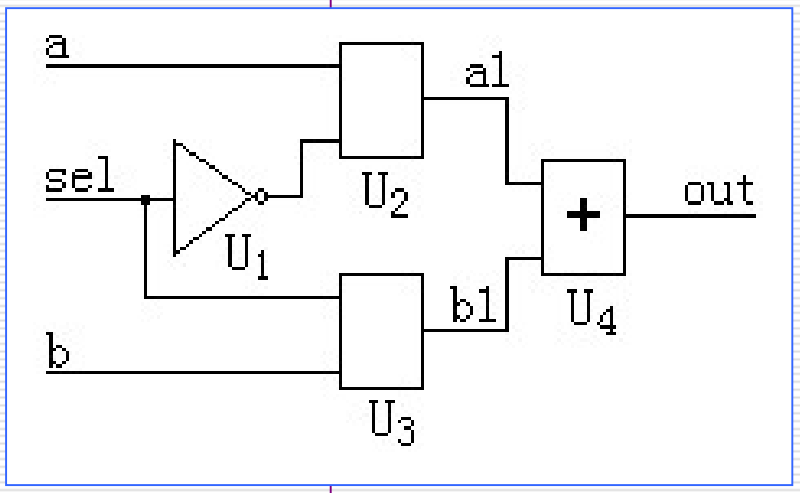
4、编程：将二进制文件载入PLD器件中。



第8章　硬件描述语言及其应用

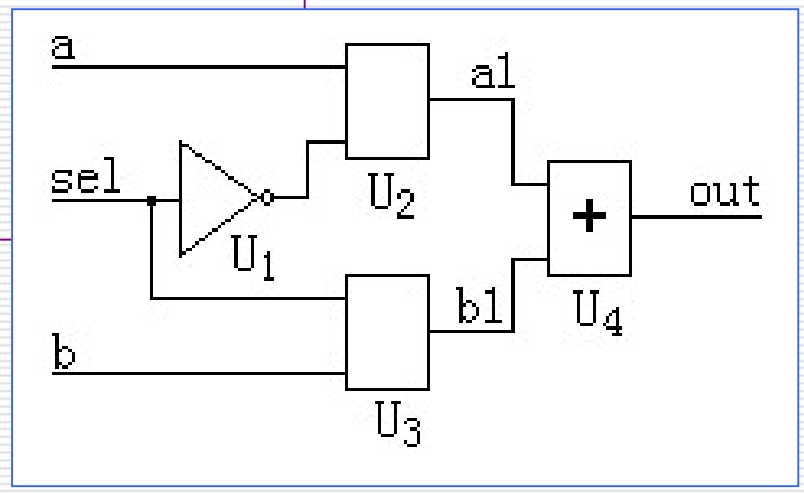
§8.1　概述（Verilog HDL）


```
module mux2to1_GL(a,b,sel,out);
   input    a,b,sel;
   output out;
   wire     selnot,a1,b1;
   not  U1(selnot,sel);
   and  U2(a1,a,selnot);
   and  U3(b1,b,sel);
   or    U4(out,a1,b1);
endmodule        //  门级模型
```



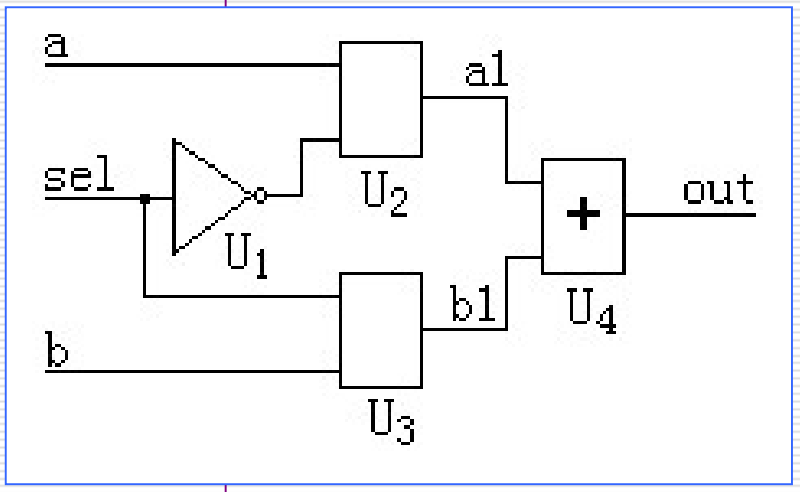
§8.1　概述（Verilog HDL）


```
module mux2to1_GL(a,b,sel,out);
   input    a,b,sel;
   output   out;
   assign   out=sel? b: a;
endmodule
//  数据流模型
```



§8.1　概述（Verilog HDL）


```
module mux2to1_GL(a,b,sel,out);
   input    a,b,sel;
   output   out;
   reg      out;
   always   @(sel or a or b)
       if(sel==1) out=b;
       else       out=a;
endmodule
// 行为级模型
```



第8章　硬件描述语言及其应用

§8.2　基本语法规则

词法：

1、间隔符：　空格、换行、TAB

2、标识符、关键字：字母或下划线开头，数字、字母、下划线、$符号组成

3、注释：　　//单行；　/*多行*/

逻辑值：

0，1，x(不定)、z（高阻）

常量：

3’b101　　12’h13x

4’d10

变量数据类型：

```
wire  a, b;  //线网
reg   c;     //寄存器
[m:n]        //定义位宽
```

第8章　硬件描述语言及其应用

§8.3　基本运算

算术运算：

运算符：+、-、*、/、%

[例]：4’b101x+4’b0111，结果：4’bxxxx

相等运算：

==（相等）、！=（不等）：　运算结果为0、1、x。

===（全等）、！==（不全等）：运算结果为0、1。

逻辑运算：

运算结果：0、1;

x：操作数包含不定态。

其他：

位运算、缩位运算、关系运算

移位运算、位拼接运算

第8章　硬件描述语言及其应用

§8.4　门级建模

基本门元件

Verilog HDL内置12个基本门级元件

元件符号	说明	元件符号	说明
and	多输入与门	nand	多输入与非门
or	多输入或门	nor	多输入或非门
xor	多输入异或门	xnor	多输入异或非门
buf	多输出缓冲器	not	多输出非门
bufif1	高有效三态门	notif1	高有效三态反相器
bufif0	低有效三态门	notif0	低有效三态反相器

§8.4　门级建模

基本门的运用

调用形式：

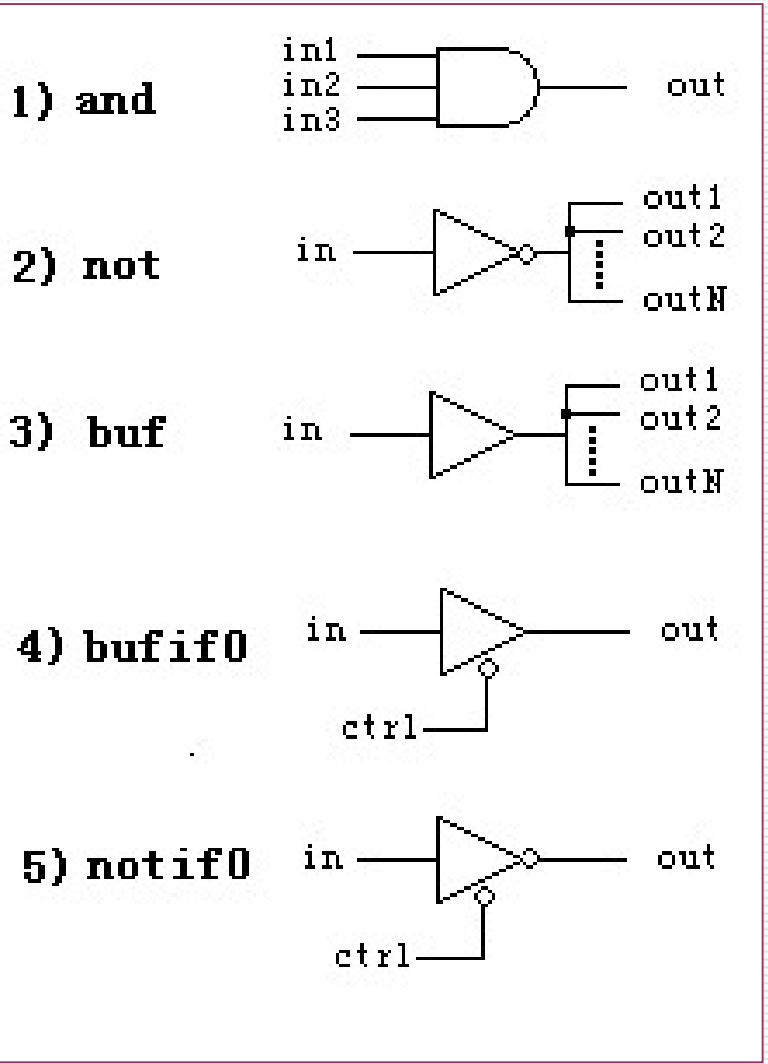
1、**and** A1(out,in1,in2,in3)

2、**not**　N1(out1,out2,..,in)

3、**buf**　B1(out1,ou2,..,in)

4、**bufif0**　B1(out,in,ctrl)

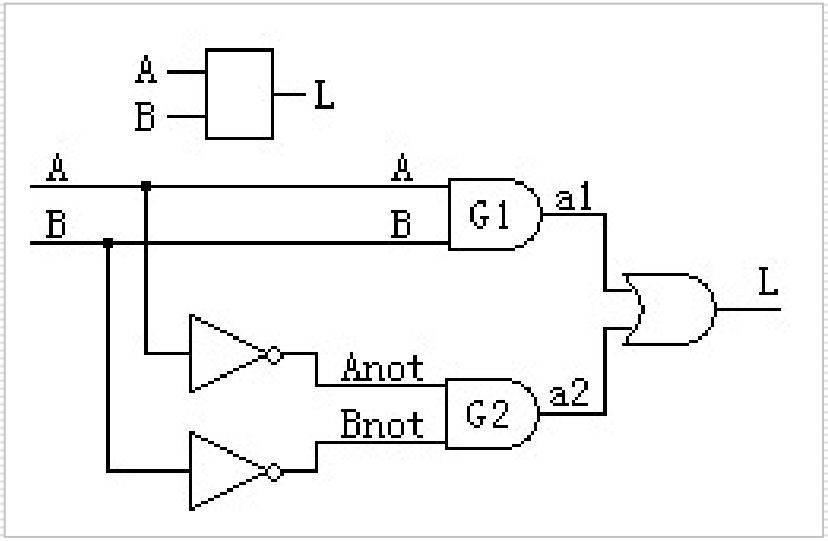
5、**notif0**　N1(out,in,ctrl)



§8.4　门级建模

基本门的运用

[例子]




```
module  circuit(A,  B,  L);
    input    A,   B;
    output   L;
    wire     a1,a2,Anot,Bnot;
    and   G1(a1, A, B);
    and   G2(a2, Anot, Bnot);
    not  G3(Anot, A);
    not  G4(Bnot, B);
    or    G5(L, a1, a2);
endmodule
```

§8.5　Verilog HDL行为级建模

连续赋值语句

assign连续赋值语句：

```
module   M1(A,B,L1,L2,L3);
    input[3:0]  A,B;
    output  L1,L2,L3;
    wire tmp = (A<B);
    assign
      L1=(A<B);
      L2=(A>B);
      L3=(A==B);
endmodule
```

§8.5　Verilog HDL行为级建模

行为级建模基础

always过程语句：

```
always @(敏感信号表达式)
    begin：块名
  变量定义
  过程赋值
  if-else  条件语句
  case, casex, casez  分支语句
  while, repeat, for 循环语句
    end
```

§8.5　Verilog HDL行为级建模

行为级建模基础

[例]对4-2优先编码的建模

```
module  priority_encode(in,out)
    input[3:0]  in;
    output[1:0]     out;
    wire[3:0]    in;
    reg[1:0]     out;
```

```
always @(in)
  begin
     casez(in)
        4'b1???: out=2'b11;
        4'b01??: out=2'b10;
        4'b001?: out=2'b01;
        4'b0001: out=2'b00;
        default: out=2'b00;
     endcase
  end
endmodule
```

§8.5　Verilog HDL行为级建模

循环语句

[例]　8位乘法器实现

```
module
    param
        // 变
    input[
    output
    wire[S
    reg[LO
```

```
always @(opA  or opB)      //过程语句
  begin:  mult                  //语句块名称
    integer  index;     //32位带符号整型变量
    result=0;
    for(index=0;index<SIZE;index=index+1)
        if(opB[index]= =1)
            result = result + (opA<<index);   //移位操作
  end
endmodule
```

第8章　硬件描述语言及其应用

§8.5　Verilog HDL行为级建模

关于事件控制

★电平触发：

always @(opA or opB)

★边沿触发：

always @(posedge clk　or negedge clr)

posedge:　保留字，上升沿

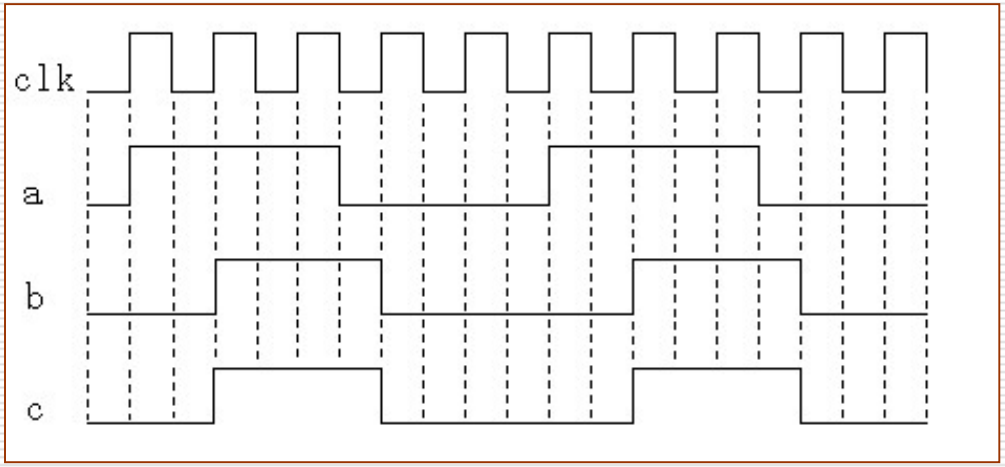
negedge：保留字，下降沿

§8.5　Verilog HDL行为级建模

阻塞赋值和非阻塞赋值

★阻塞赋值方式：

```
module block(c, b, a, clk);
   input    clk, a;
   output   c, b;
   reg      c, b;
   always @(posedge  clk)
         begin
            b=a;
            c=b;
         end
endmodule
```

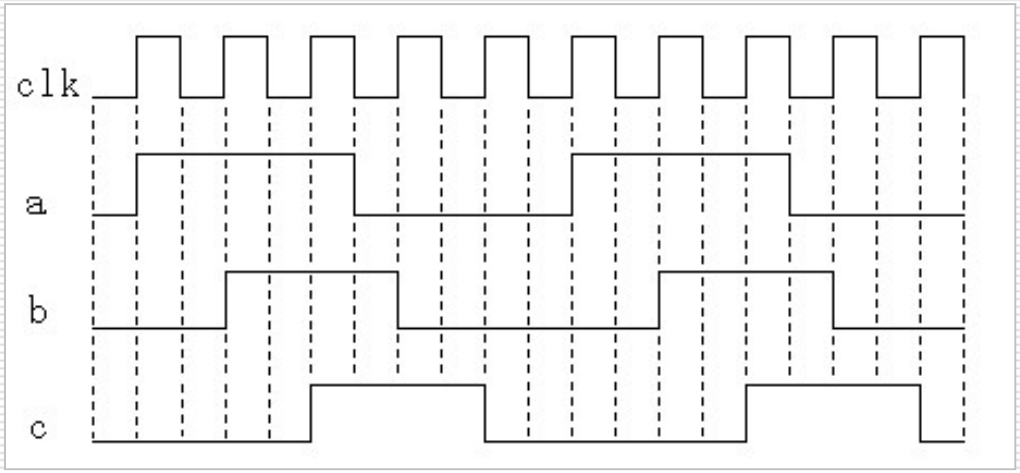


§8.5　Verilog HDL行为级建模

阻塞赋值和非阻塞赋值

★非阻塞赋值方式：

```
module block(c, b, a, clk);
   input    clk, a;
   output   c, b;
   reg      c, b;
   always @(posedge  clk)
         begin
            b<=a;
            c<=b;
         end
endmodule
```



§8.5　Verilog HDL行为级建模

并发执行

★两个always语句块的并发执行：

```
module block(c, b, a, clk);
   input    clk, a;
   output  c, b;
   reg       c, b;
   always @(posedge  clk)
         begin   b=a;   end
    always @(posedge  clk)
         begin   c=b;   end
endmodule
```

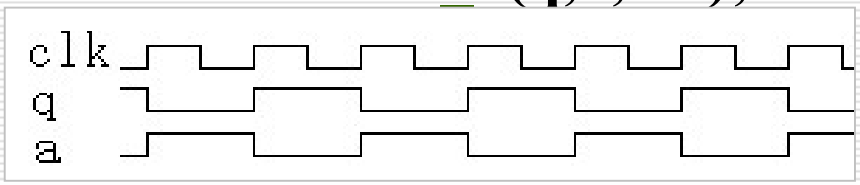
两个always语句块，用阻塞赋值实现非阻塞赋值功能。

§8.5　Verilog HDL行为级建模

并发执行

★模块1：

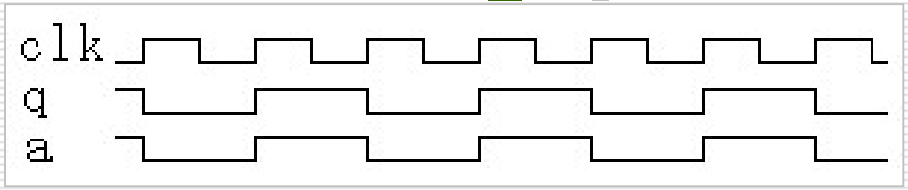
```
module serial_1(q,a,clk);
```




```
reg      q, a;
always @(posedge  clk)
      begin
         q=~q;
         a=~q;
      end
endmodule
```

★模块2：

```
module serial_2(q,a,clk);
```



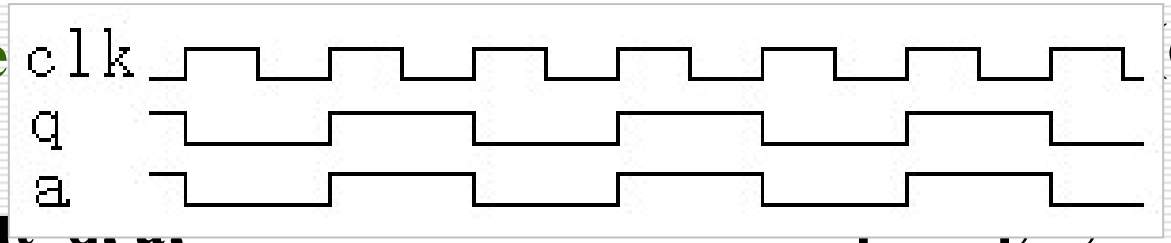

```
reg      q, a;
always @(posedge  clk)
      begin
         a=~q;
         q=~q;
      end
endmodule
```

§8.5　Verilog HDL行为级建模

并发执行

★并行模块1：

```
module
  input
  outpu
  reg      q, a;
  always @(posedge  clk)
     begin  q=~q;  end
  always @(posedge  clk)
     begin  a=~q;  end
endmodule
```



★并行模块2：

```
q,a,clk);
  reg      q, a;
  always @(posedge  clk)
     begin  a=~q;  end
  always @(posedge  clk)
     begin  q=~q;  end
endmodule
```

§8.5　Verilog HDL行为级建模

并发执行

[例]：

块语句：
```
begin
    B=A;
    C=B+1;
end
```
与：
```
begin
    B<=A;
    C<=B+1;
end
```
的区别？

在同一块语句中不能同时使用阻塞赋值语句和非阻塞赋值语句；

数字电路包括组合逻辑电路与组合时序电路，时序电路建议用非阻塞赋值语句。

§8.5　Verilog HDL行为级建模

行为级建模

［例］　J-K触发器行为级建模：

```
module  JK_FF(j, k, clk, Q, Qnot)
   output  Q, Qnot;
   input   j, k, clk;
   reg     Q;
   assign  Qnot=~Q;
   always  @(posedge  clk)
      case ( {j, k} )
           2'b00:  Q<=Q;          2'b01:  Q<=1'b0;
           2'b10:  Q<=1'b1;       2'b11:  Q<=~Q;
      endcase
   endmodule
```

J	K	CLK	Q	/Q
0	0	↑	Q	/Q
0	1	↑	0	1
1	0	↑	1	0
1	1	↑	/Q	Q

§8.5　Verilog HDL行为级建模

状态图的行为级建模

有限状态机（Finite State Machine）：

1、摩尔（Moore）型状态机

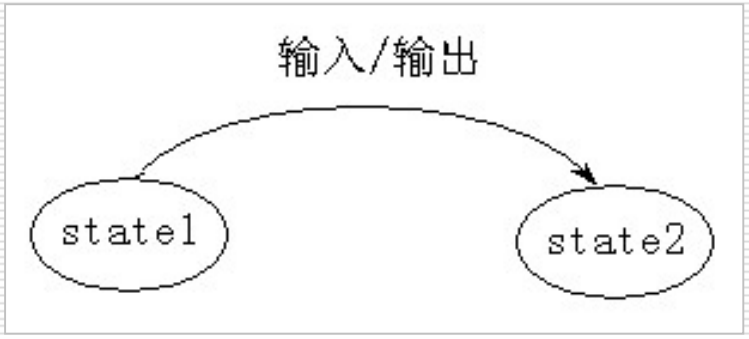
输出状态仅是当前状态的函数

2、米里（Mealy）型状态机

输出状态是当前状态和当前输入状态的函数

状态机的表示：

状态图、状态表

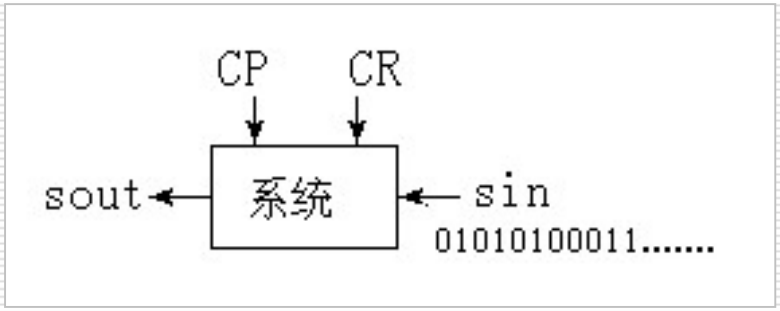


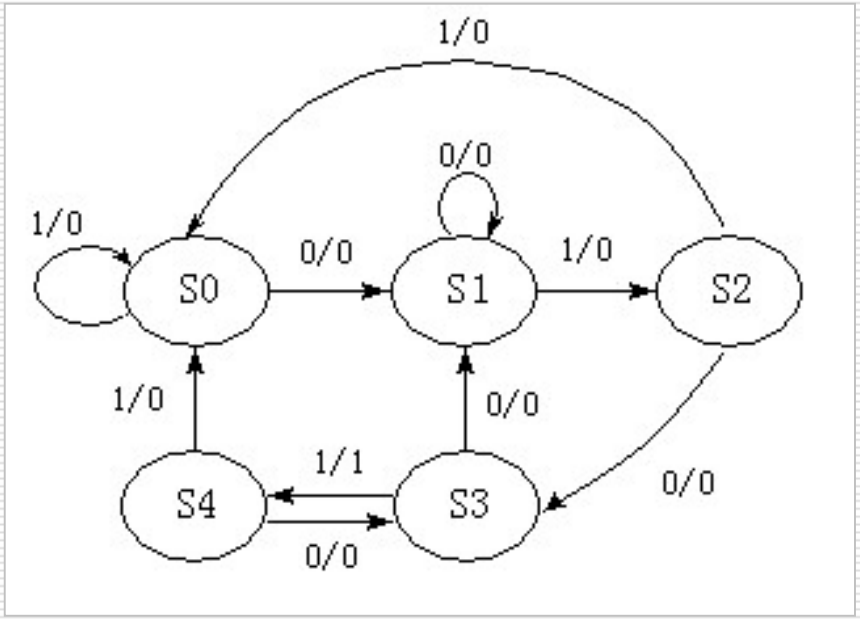
§8.5　Verilog HDL行为级建模

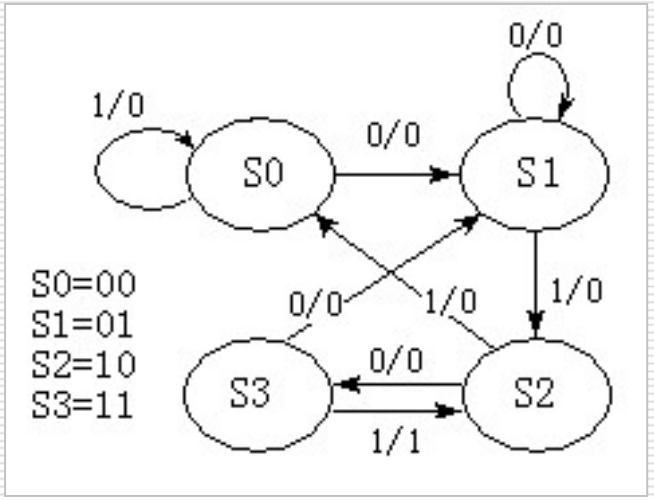
状态图的行为级建模

[例]设计一个序列检测系统，当检测到串行序列0101时，输出1，否则输出0。

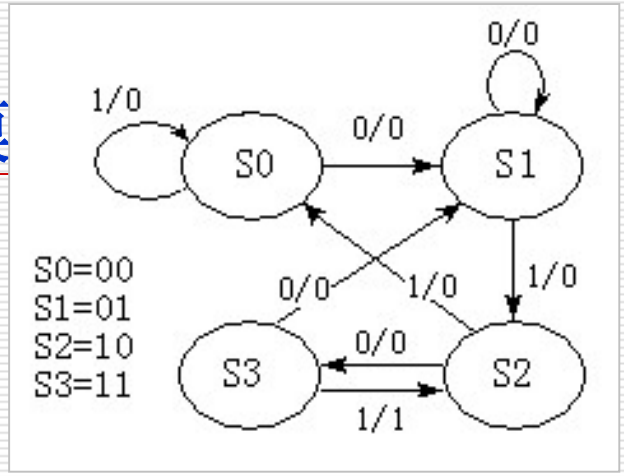
（sin、sout、CP,CR）







状态图的行为级建模

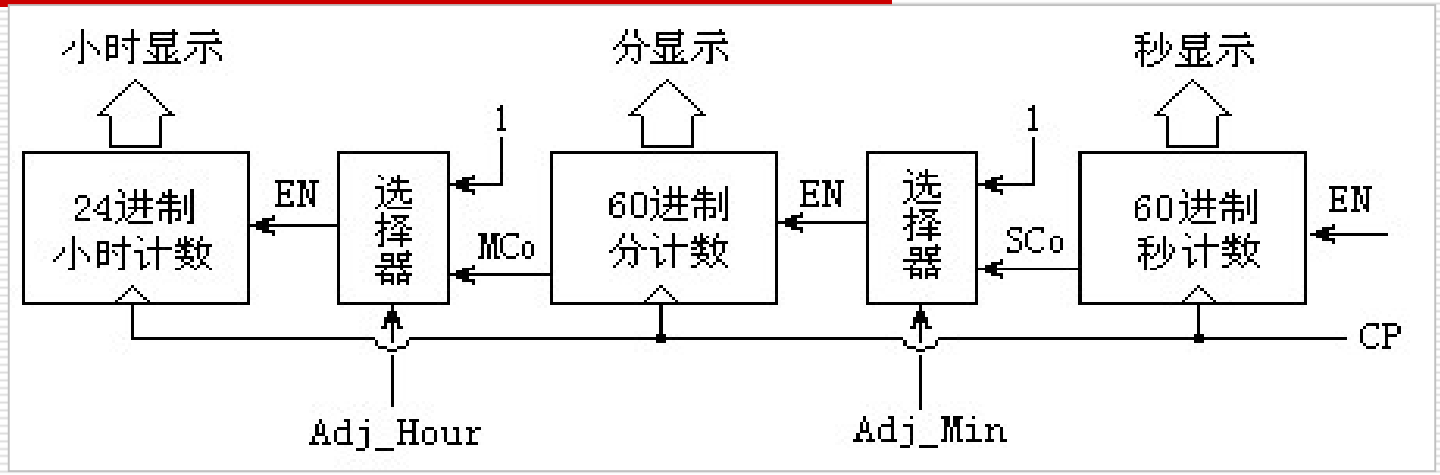



```
always @ (posedge CP)
   begin
       if(~CR)    c_s<=S0;
       else       c_s<=n_s;
   end
```

```
 input      sin,CP,CR;
always @ (c_s  or  sin)
   begin   case(c_s)
        S0: begin sout<=0;n_s<=(sin==1)?S0:S1;end
        S1: begin sout<=0;n_s<=(sin==1)?S2:S1;end
        S2: begin sout<=0;n_s<=(sin==1)?S0:S3;end
             S3: if(sin==1)
                    begin  sout<=1;  n_s<=S2;  end
                 else
                    begin  sout<=0;  n_s<=S1;  end
            endcase
    end
```

§8.5　Verilog HDL行为级建模

数字钟的分层次设计




```
module  clock (Hour, Min, Sec, CP, nCR, EN, Adj_M, Adj_S)
   input          CP,  nCR,  EN,  Adj_M,  Adj_H;
   output[7:0]    Hour,  Min,  Sec;
   supply1        Vdd;     //连线型变量：wire，supply1,supply0
   wire           MinL_EN,   MinH_EN,   Hour_EN;
   ……
endmodule
```

§8.5　Verilog HDL行为级建模

数字钟的分层次设计

```
counter10    U1(Sec[3:0],nCR,EN,CP);
counter6     U2(Sec[7:4],nCR,(Sec[3:0]==8'h59),CP);
assign  MinL_EN=Adj_M? Vdd: (Sec==8'h59);
assign  MinH_EN=(Adj_M&&(Min[3:0]==4'h9))||
                        (Min[3:0]==4'h9)&&(Sec==8'h59);
counter10  U3(Min[3:0],nCR,MinL_EN,CP);      //低位10进制
counter6   U4(Min[7:4],nCR,MinH_EN,CP);      //高位6进制
assign Hour_EN=Adj_H ? Vdd:
                    ((Min==8'h59)&&(Sec==8'h59));
counter24  U5(Hour[7:4],Hour[3:0],nCR,Hour_EN,CP);
```

第8章　硬件描述语言及其应用

RTL级建模

- 在FPGA设计中，只有寄存器传输（Register Transfer Level，RTL）级以下的建模才是可综合的
- RTL级语法是Verilog HDL的一个子集
- 用于描述数据如何在寄存器之间进行传输、控制和处理
- 需要掌握以下内容：
  - ✓ “硬件意识”
  - ✓RTL级语法
  - ✓目标器件结构等