基本数字逻辑单元HDL描述

主讲：何宾

Email：hebin@mail.buct.edu.cn

2018.08

基本数字逻辑单元HDL描述
--有限自动状态机HDL描述

有限自动状态机FSM（Finate State Machine）的设计是复杂数字系统中非常重要的一部分，是实现高效率高可靠性逻辑控制的重要途径。

- 大部分数字系统都是由控制单元和数据单元组成的。
- 数据单元负责数据的处理和传输，而控制单元主要是控制数据单元的操作的顺序。
- 在数字系统中，控制单元往往是通过使用有限状态机实现的，有限状态机接受外部信号以及数据单元产生的状态信息，产生控制信号序列。

有限自动状态机HDL描述
--FSM设计原理

有限状态机可以由标准数学模型定义。此模型包括一组状态、状态之间的一组转换以及和状态转换有关的一组动作。有限状态机可以表示为：

$$M = (I, O, S, f, h)$$

其中：

$S = \{S_i\}$ 表示一组状态的集合

$I = \{I_j\}$ 表示一组输入信号

$O = \{O_k\}$ 表示一组输出信号

$f(S_i, I_j) : S \times I \rightarrow S$ 为状态转移函数

$h(S_i, I_j) : S \times I \rightarrow O$ 为输出函数

有限自动状态机HDL描述
--FSM设计原理

从上面的数学模型可以看出，如果在数字系统中实现有限状态机，则应该包含三部分：

- 状态寄存器；
- 下状态转移逻辑；
- 输出逻辑。

有限自动状态机HDL描述
--状态定义及编码规则

状态变量定义的Verilog HDL描述

```
reg[2:0] present_state,next_state;

   parameter s0=3’b000, s1=3’b001, s2=3’b010, s3=3’b011,

    s4=3’b100;
```

有限自动状态机HDL描述
--状态定义及编码规则

Xilinx ISE提供了One_Hot、Gray、Compact、Johnson、Sequential、Speed1、User的编码方式。

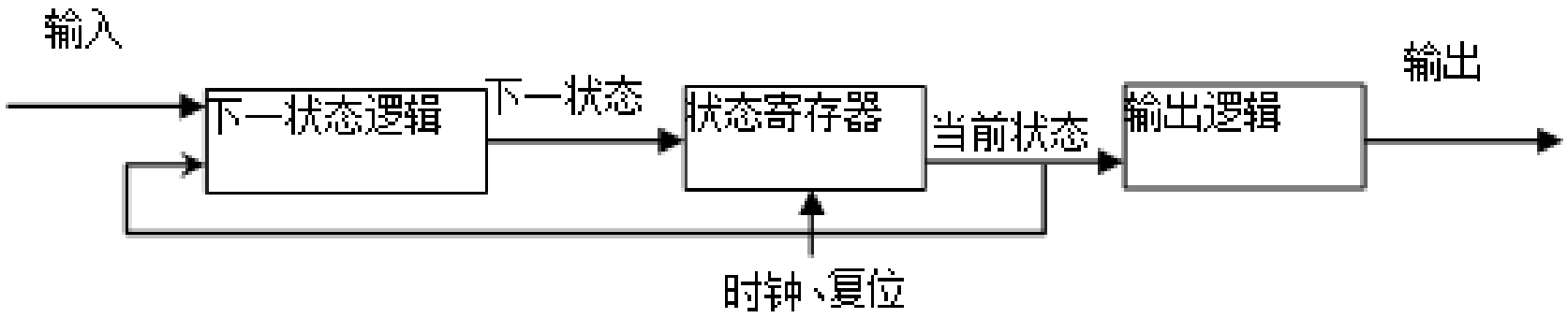
典型编码格式

十进制数	二进制码	Gray码	Johnson码	One-hot吗
0	000	000	000	**001**
1	001	001	001	**010**
2	010	011	011	**100**
3	011	010	111	**1000**
4	100	110		
5	101	111		
6	110	101		
7	**111**	**100**		

有限自动状态机HDL描述
--FSM的分类及描述

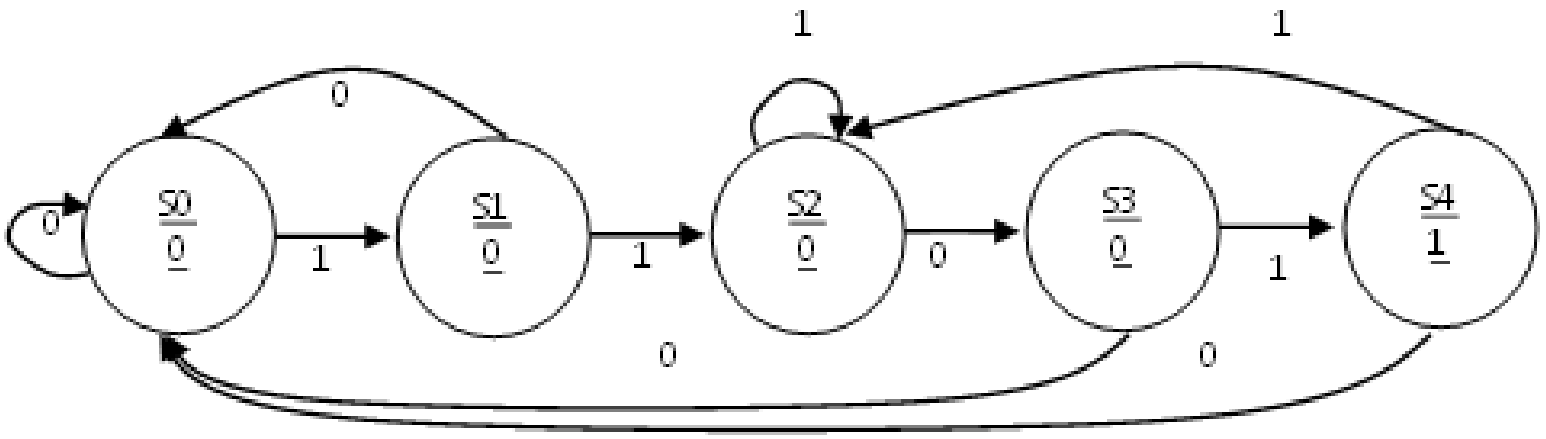
Moore型状态机

Moore型状态机的输出仅与状态机的状态有关，与状态机的输入无关。



有限自动状态机HDL描述
--FSM的分类及描述

设计序列检测器。该序列检测器将检测序列“1101”，当检测到该序列时，状态机的输出z为1。

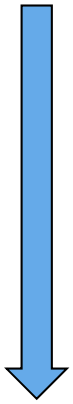


思考：分析该状态机工作原理

有限自动状态机HDL描述
--FSM的分类及描述

Moore型序列检测器Verilog HDL描述

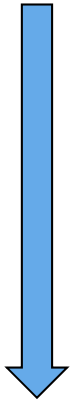
```
module moore(
input wire clk,
input wire clr,
input wire din,
output reg dout
);
reg[2:0] present_state, next_state;
parameter  S0 = 3'b000, S1 = 3'b001, S2 = 3'b010,
                  S3 = 3'b011, S4 = 3'b100;
always @(posedge clk or posedge clr)
  begin
     if (clr == 1)
```



继续下一页

有限自动状态机HDL描述
--FSM的分类及描述

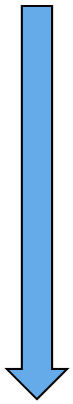
```
        present_state <= S0;
    else
        present_state <= next_state;
    end

always @(*)
begin
    case(present_state)
        S0: if(din == 1)
                next_state <= S1;
            else
                next_state <= S0;
        S1: if(din == 1)
            next_state <= S2;
```



继续下一页

有限自动状态机HDL描述
--FSM的分类及描述

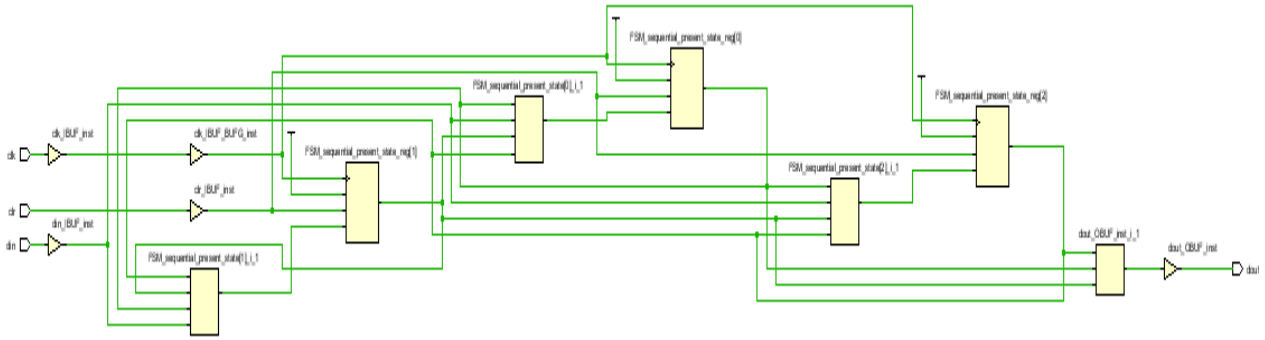
```
        else
            next_state <= S0;
S2: if(din == 0)
            next_state <= S3;
        else
            next_state <= S2;
S3: if(din == 1)
            next_state <= S4;
        else
            next_state <= S0;
S4: if(din == 0)
            next_state <= S0;
        else
            next_state <= S2;
```



继续下一页

有限自动状态机HDL描述
--FSM的分类及描述

```
        default next_state <= S0;
    endcase
  end

always @(*)
    begin
        if (present_state == S4)
                dout <= 1;
        else
                dout <= 0;
        end
endmodule
```



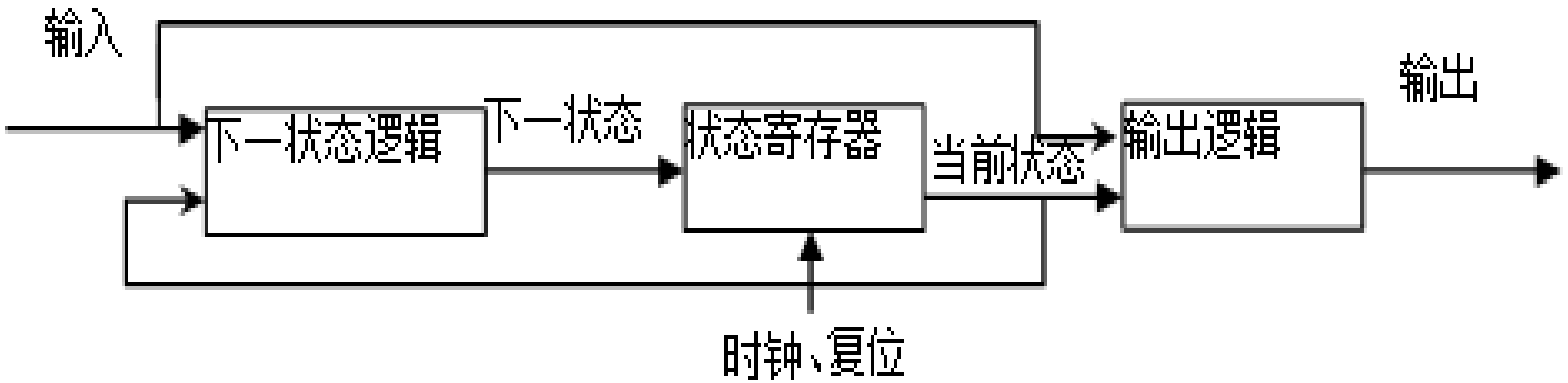
思考与练习1：根据综合后的结果，分析该状态机结构。

本设计保存在本书\eda_verilog\example6_33目录下

有限自动状态机HDL描述
--FSM的分类及描述

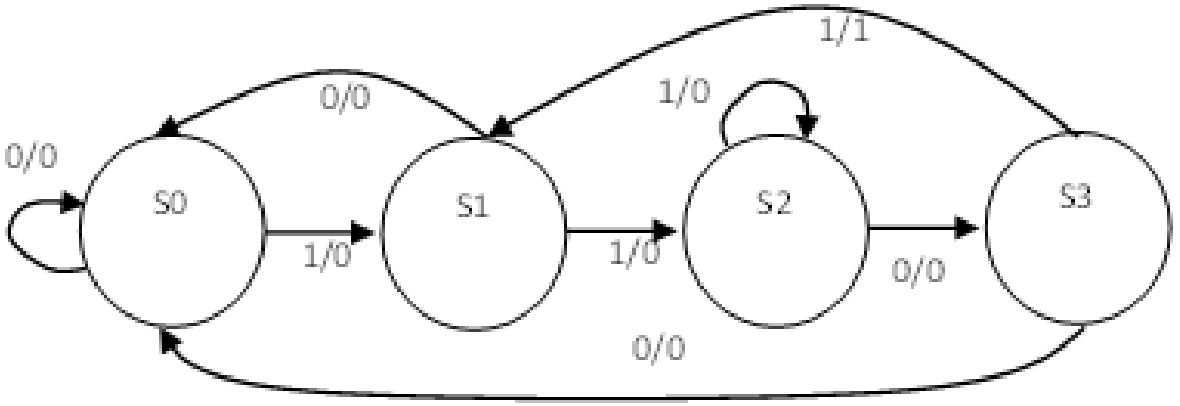
Mealy型状态机

Mealy型状态机的输出由状态机的输入和状态机的状态共同决定；



有限自动状态机HDL描述
--FSM的分类及描述

该序列检测器将检测序列“1101”，当检测到该序列时，状态机的输出z为1。



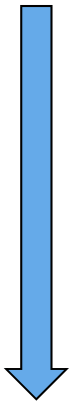
有限自动状态机HDL描述
--FSM的分类及描述

- Moore状态机检测序列时，使用了5个状态，当为状态S4时，输出为1。
- 可以使用Mealy状态机检测序列。当为状态S3时，且输入为1时，输出z为1。
  - 状态迁移的条件表示为当前输入/当前输出。比如当为状态S3时（接收到序列110），输入为1，输出将变为1。下一个时钟沿有效时，状态变化到S1，输出z变为0。
  - 为了让z成为寄存的输出（也就是说状态变化为S1时，输出仍然被保持1），为输出添加寄存器。即Mealy状态机的输出和D触发器连接，这样下一个时钟有效时，状态仍然变化到S1，但输出被锁存为1（被保持）。

有限自动状态机HDL描述
--FSM的分类及描述

Mealy型序列检测器Verilog HDL描述

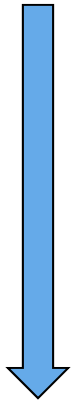
```
module seqdetb(
   input wire clr,
   input wire clk,
   input wire din,
   output reg dout
);
reg[1:0] present_state, next_state;
parameter S0 = 2'b00, S1 = 2'b01, S2 = 2'b10, S3 = 2'b11;
always @(posedge clr or posedge clk)
```



继续下一页

有限自动状态机HDL描述
--FSM的分类及描述

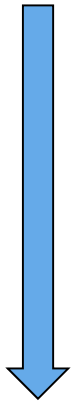
```
begin
    if(clr == 1)
        present_state <= S0;
    else
        if((present_state == S3) & din == 1)
        begin
            dout <= 1;
            present_state <= next_state;
        end
        else
        begin
            dout <= 0;
            present_state <= next_state;
```



继续下一页

有限自动状态机HDL描述
--FSM的分类及描述

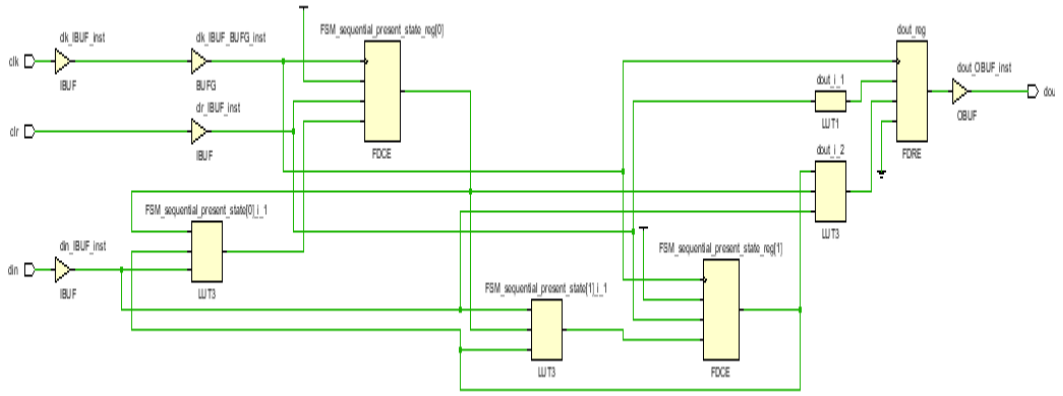
```
        end
    end
always @(*)
begin
    case(present_state)
        S0: if(din == 1)
                next_state <= S1;
            else
                next_state <= S0;
        S1: if(din == 1)
                next_state <= S2;
            else
                next_state <= S0;
```



继续下一页

有限自动状态机HDL描述
--FSM的分类及描述

```
        S2: if(din == 0)
                next_state <= S3;
            else
                next_state <= S2;
        S3: if(din == 1)
                next_state <= S1;
            else
                next_state <= S0;
        default next_state <= S0;
    endcase
end
endmodule
```

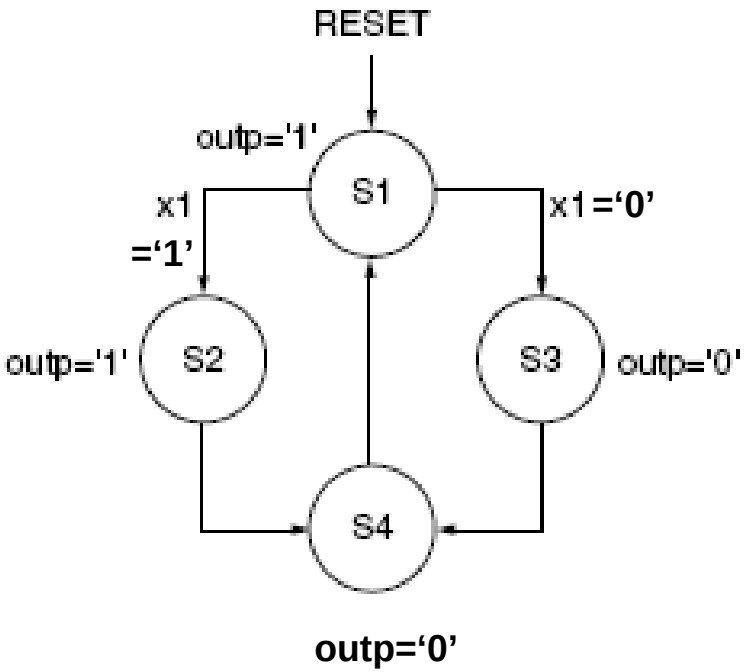


思考与练习1：根据综合后的结果，分析该状态机结构。

本设计保存在本书\eda_verilog\example6_34目录下

有限自动状态机HDL描述
--FSM的描述

状态机描述规则



状态机描述方式：

- 三进程；
- 两进程；
- 单进程；

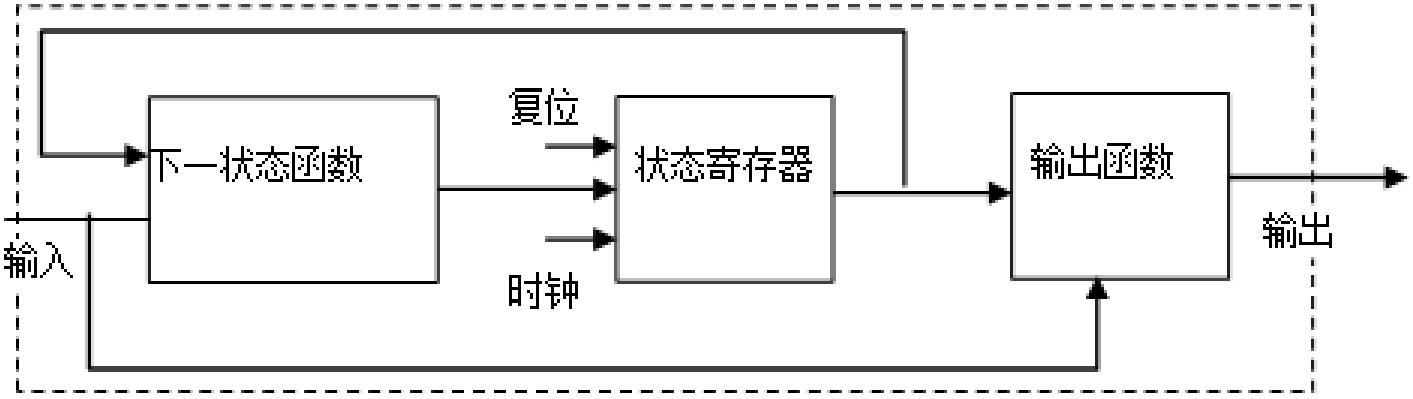
该状态图包含：

- 四个状态：s1，s2，s3，s4;
- 5个转移；
- 1个输入“x1”;
- 1个输出“outp”；

有限自动状态机HDL描述
--FSM的描述

单进程状态机的实现方法

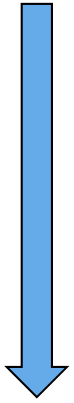
采用单进程状态机描述时，状态的变化、状态寄存器和输出功能描述用一个进程进行描述。



有限自动状态机HDL描述
--FSM的描述

单进程状态机Verilog HDL描述

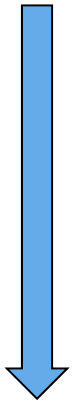
```
module fsm_1(
    input clk,
    input reset,
    input x1,
    output reg outp
    );
reg [1:0] state;
parameter s1 = 2'b00, s2 = 2'b01, s3 = 2'b10, s4 = 2'b11;
initial begin
     state = 2'b00;
```



继续下一页

有限自动状态机HDL描述
--FSM的描述

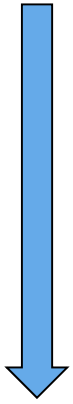
```
 end

always @(posedge clk or posedge reset)
begin
    if(reset)
      begin
        state <= s1;
        outp <= 1'b1;
      end
    else
      begin
         case(state)
            s1: begin
                if(x1 == 1'b1)
```



继续下一页

有限自动状态机HDL描述
--FSM的描述

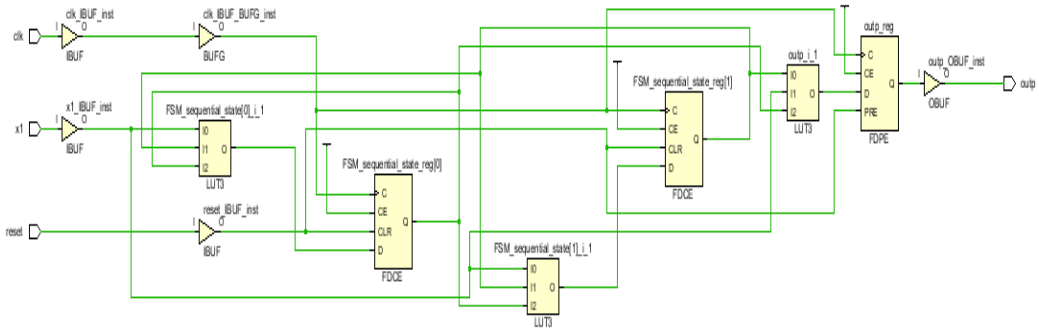
```
    begin
        state <= s2;
        outp <= 1'b1;
    end
  else
    begin
        state <= s3;
        outp <= 1'b0;
    end
  end
s2: begin
        state <= s4;
        outp <= 1'b1;
```



继续下一页

有限自动状态机HDL描述
--FSM的描述

```
                end
            s3: begin
                    state <=s4;
                    outp <= 1'b0;
                end
            s4: begin
                    state <= s1;
                    outp <= 1'b0;
                end
        endcase
    end
end
endmodule
```



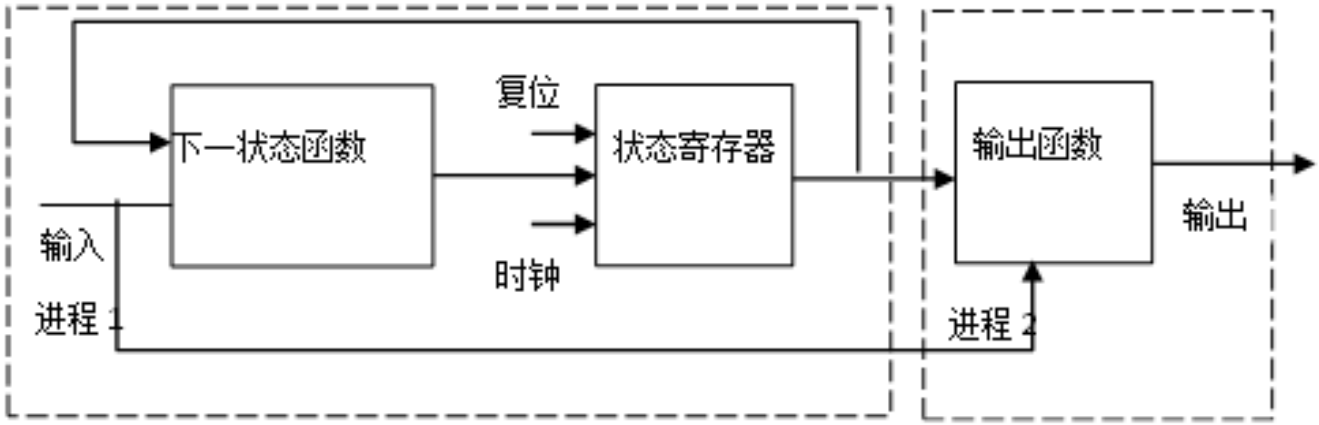
思考与练习1：根据综合后的结果，分析该状态机结构。

本设计保存在本书\eda_verilog\example6_35目录下

有限自动状态机HDL描述
--FSM的描述

双进程状态机的实现方法

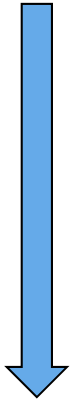
与单进程状态机不同的是，采用双进程状态机时，输出函数用一个进程描述，而状态寄存器和下一状态函数用另一个进程描述。



有限自动状态机HDL描述
--FSM的描述

双进程状态机Verilog HDL描述

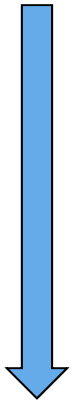
```
module fsm_2(
    input clk,
    input reset,
    input x1,
    output reg outp
    );
reg [1:0] state;
parameter  s1 = 2'b00, s2 = 2'b01, s3 = 2'b10, s4 = 2'b11;
```



继续下一页

有限自动状态机HDL描述
--FSM的描述

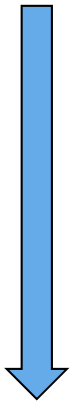
```
initial begin
    state = 2'b00;
end

always @(posedge clk or posedge reset)
begin
  if(reset)
        state <= s1;
  else
  begin
```



继续下一页

有限自动状态机HDL描述
--FSM的描述

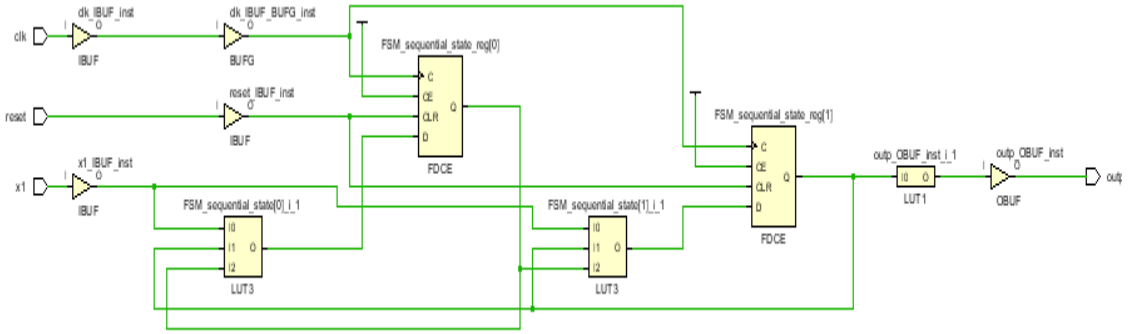
```
        case(state)
            s1: if(x1 == 1'b1)
                    state <= s2;
                else
                    state <= s3;
            s2: state <= s4;
            s3: state <= s4;
            s4: state <= s1;
        endcase
    end
end
```



继续下一页

有限自动状态机HDL描述
--FSM的描述

```
always @(state)
begin
        case (state)
            s1: outp <= 1'b1;
            s2: outp <= 1'b1;
            s3: outp <= 1'b0;
            s4: outp <= 1'b0;
        endcase
 end
endmodule
```



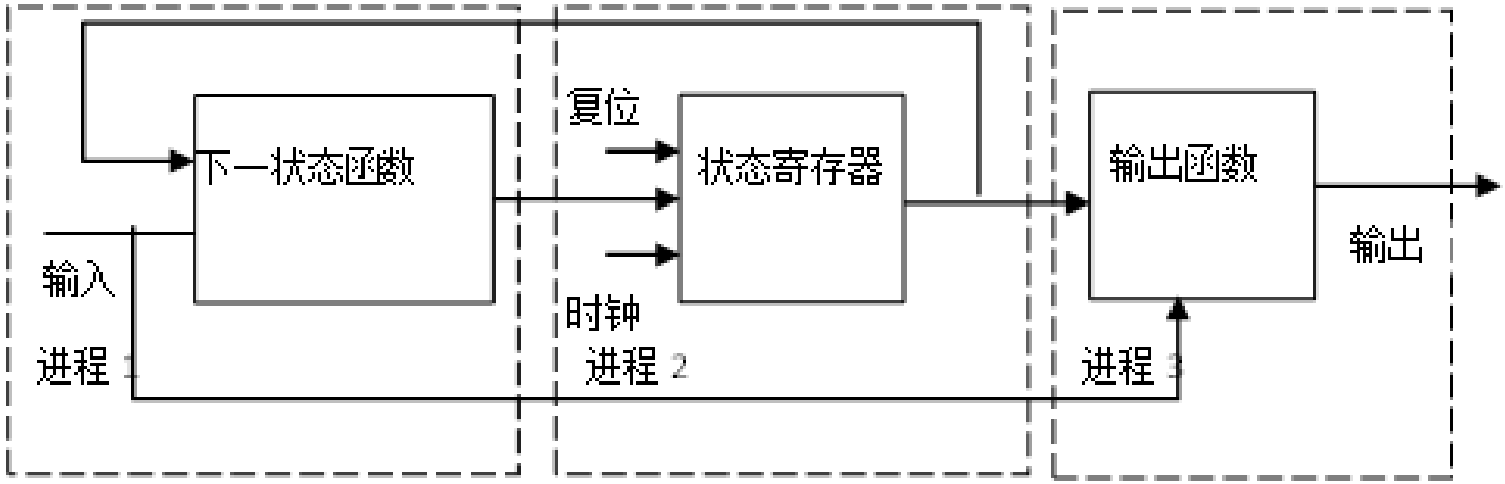
思考与练习**1**：根据综合后的结果，分析该状态机结构。

本设计保存在本书**\eda_verilog\example6_36**目录下

有限自动状态机HDL描述
--FSM的描述

三进程状态机的实现规则

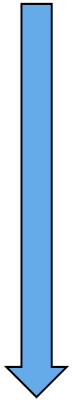
采用三进程状态机时，输出函数用一个进程描述，而状态寄存器和下一状态函数分别用两个进程描述。



有限自动状态机HDL描述
--FSM的描述

三进程状态机Verilog HDL描述

```
module fsm_3(
    input clk,
    input reset,
    input x1,
    output reg outp
    );
reg [1:0] state;
reg [1:0] next_state;
parameter s1 = 2'b00, s2 = 2'b01, s3 = 2'b10, s4 = 2'b11;
```



继续下一页

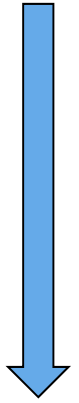
有限自动状态机HDL描述
--FSM的描述

```
initial begin
        state <= 2'b00;
end

always @(posedge clk or posedge reset)
begin
    if (reset)
        state <= s1;
    else
        state <= next_state;
end
```

继续下一页

有限自动状态机HDL描述
--FSM的描述

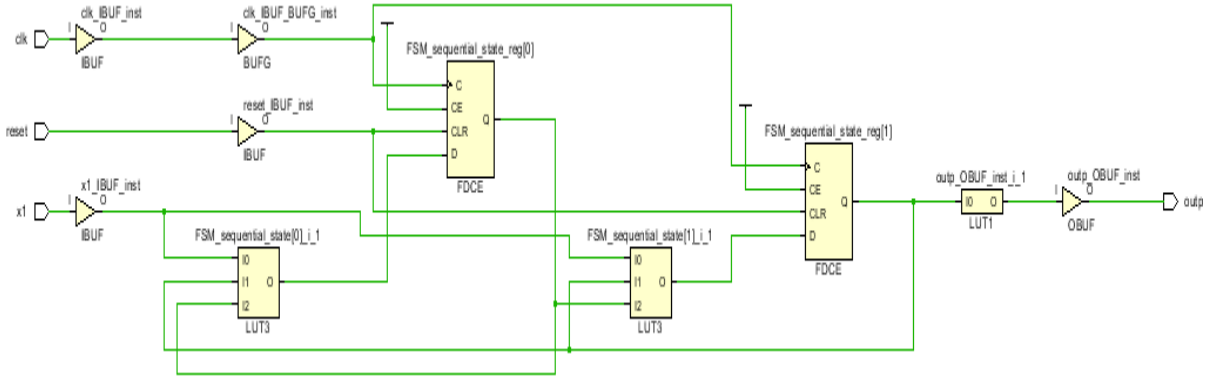
```
always @(state or x1)
begin
   case (state)
      s1: if(x1 == 1'b1)
                next_state = s2;
            else
                next_state = s3;
      s2: next_state = s4;
      s3: next_state = s4;
      s4: next_state = s1;
   endcase
end
```



继续下一页

有限自动状态机HDL描述
--FSM的描述

```
always @(state)
begin
    case (state)
        s1: outp = 1'b1;
        s2: outp = 1'b1;
        s3: outp = 1'b0;
        s4: outp = 1'b0;
    endcase
end
endmodule
```



思考与练习1：根据综合后的结果，分析该状态机结构。

本设计保存在本书\eda_verilog\example6_37目录下