数字逻辑实验——序列检测器

2016011058 漆耘含

一、 实验目的

1. 掌握有限状态机的实现原理和方法
2. 掌握序列检测的方法

二、 设计方案

1. 有限状态机

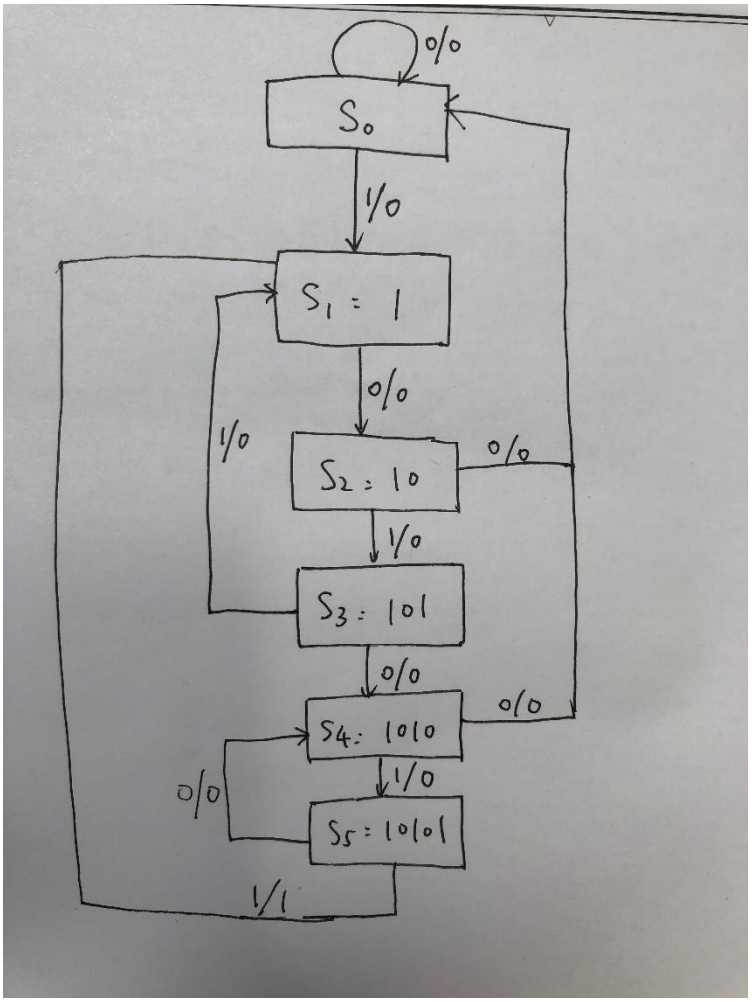
采用摩尔机的设计方案，对序列“101011”的顺序序列进行编码：

序列	编号	编码（LED 显示）
0	S0	000
1	S1	001
10	S2	010
101	S3	011
1010	S4	100
10101	S5	101

摩尔机的输出有 LED 三位显示和是否检测到序列“101011”，则可以列出次态表：

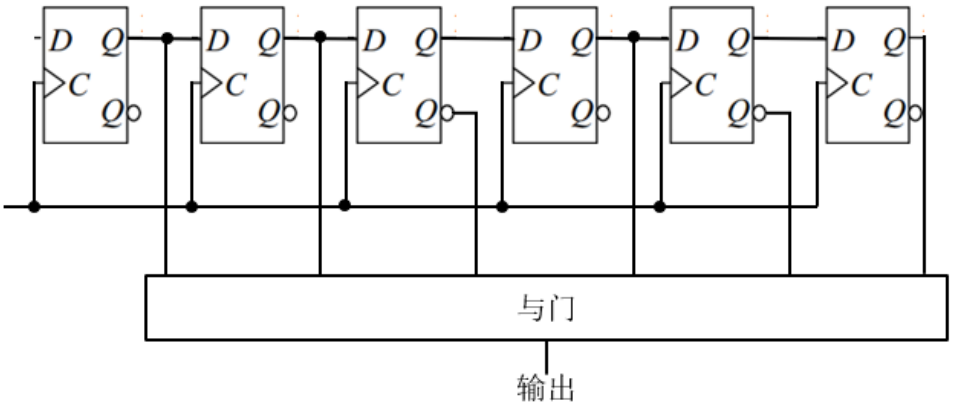
输入序列	现态	次态		输出	
		X=0	X=1	X=0	X=1
无	S0	S0	S1	0	0
1	S1	S2	S1	0	0
10	S2	S0	S3	0	0
101	S3	S4	S1	0	0
1010	S4	S0	S5	0	0
10101	S5	S4	S1	0	1

以上是最简状态的次态表，下面画出状态转移图：



2. 移位寄存器

采用六个 D 触发器级联形成，每个触发器对应一个 LED 灯输出，判断是否检测到序列“101011”为每个 D 触发器中储存的状态的组合逻辑。



三、 关键代码及文件清单

1. 文件清单

序列检测器：

fsm.v

fsm_tb.v

移位寄存器实现：

check_d.v

check_d_tb.v

2. 关键代码

a）Fsm:

```
1   module check(clk,reset,in_put,flag,led);
2   input clk,reset,in_put;
3   output flag;
4   output [2:0]led;
5
6   fsm f(.clk(clk),.reset(reset),.in_put(in_put),.flag(flag),.led(led));
7   endmodule
8
9
10  module fsm(clk,reset,in_put,flag,led);
11  input clk,reset,in_put;
12  output flag;
13  output [2:0]led;
14  reg flag;
15  reg [2:0]led;
```

```
17  always @(negedge clk or negedge reset)
18  begin
19    if(~reset)
20      begin
21        led<=3'b000;
22        flag<=0;
23      end
24    else
25      begin
26        case(led)
27          3'b000:
28            if(in_put)
29              begin
30                led<=3'b001;
31                flag<=0;
32              end
33            else
34              begin
35                flag<=0;
36                led<=3'b000;
37              end
38          3'b001:
39            if(~in_put)
40              begin
41                flag<=0;
42                led<=3'b010;
43              end
44            else
45              begin
46                flag<=0;
47                led<=3'b001;
48              end
49          3'b010:
50            if(in_put)
51              begin
52                flag<=0;
53                led<=3'b011;
54              end
55            else
56              begin
57                flag<=0;
58                led<=3'b000;
59              end
60          3'b011:
61            if(in_put)
62              begin
63                flag<=0;
64                led<=3'b001;
65              end
66            else
67              begin
68                flag<=0;
69                led<=3'b100;
70              end
71          3'b100:
72            if(in_put)
73              begin
74                flag<=0;
75                led<=3'b101;
76              end
77            else
78              begin
79                flag<=0;
80                led<=3'b000;
81              end
```

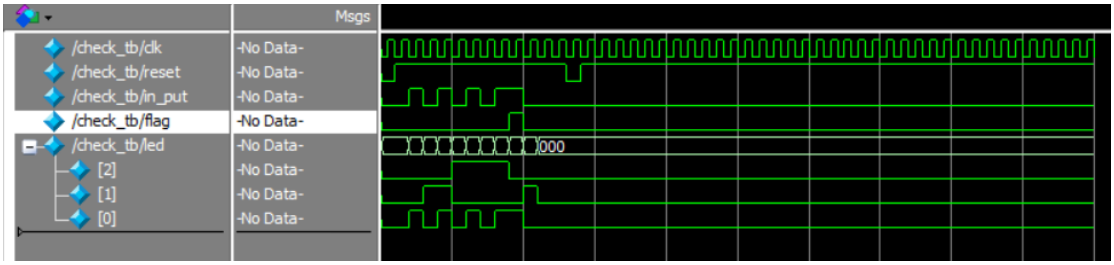
```
82            3'b101:
83              if(in_put)
84                begin
85                  flag<=1;
86                  led<=3'b001;
87                end
88              else
89                begin
90                  flag<=0;
91                  led<=3'b100;
92                end
93          endcase
94        end
95
96    end
97
98    endmodule
```

b) Check_d:

```
1   module check_d(clk,d,r,flag,q);
2     input clk,d,r;
3     output flag;
4     output [5:0]q;
5
6     wire [5:0]q;
7     wire flag;
8
9     D d1(.clk(clk),.d(d),.r(r),.q(q[5]));
10    D d2(.clk(clk),.d(q[5]),.r(r),.q(q[4]));
11    D d3(.clk(clk),.d(q[4]),.r(r),.q(q[3]));
12    D d4(.clk(clk),.d(q[3]),.r(r),.q(q[2]));
13    D d5(.clk(clk),.d(q[2]),.r(r),.q(q[1]));
14    D d6(.clk(clk),.d(q[1]),.r(r),.q(q[0]));
15    and and1(flag,q[0],~q[1],q[2],~q[3],q[4],q[5]);
16  endmodule
17
18  module D(clk,d,r,q);
19    input clk,d,r;
20    output q;
21    reg q;
22
23    always @(posedge clk or negedge r)
24    begin
25       if(~r)
26          begin
27          q<=0;
28        end
29       else
30       begin
31        q<=d;
32      end
33    end
34  endmodule
```

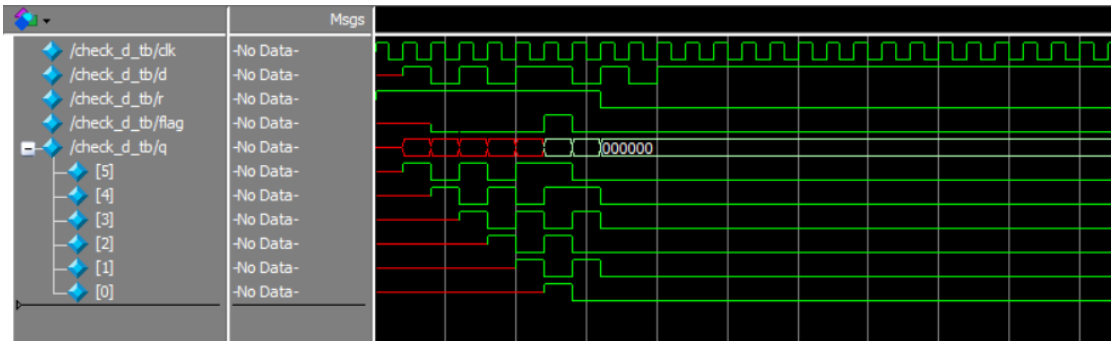
四、 仿真结果及分析

1. Fsm:



由图可以看出，在输入序列达到“101011”的时候，flag 输出为 1，同时状态转到 S1（即“001”）。由 LED 输出可以看出状态转移的过程，是符合状态转移图中的流程的。

2. Check_d:



从输入来看，一次输入序列“101011”，从输出 LED 来看很明显是一个移位寄存器，当输入“101011”时，flag 输出“1”，满足要求

五、 综合情况

1. 有限状态机

Flow Summary	
Flow Status	Successful - Sun May 06 15:22:58 2018
Quartus II 64-Bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Web Edition
Revision Name	check
Top-level Entity Name	check
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Total logic elements	5 / 33,216 (< 1 %)
Total combinational functions	5 / 33,216 (< 1 %)
Dedicated logic registers	4 / 33,216 (< 1 %)
Total registers	4
Total pins	7 / 475 (1 %)
Total virtual pins	0
Total memory bits	0 / 483,840 (0 %)
Embedded Multiplier 9-bit elements	0 / 70 (0 %)
Total PLLs	0 / 4 (0 %)

Slow Model Fmax Summary

	Fmax	Restricted Fmax	Clock Name	Note
1	517.06 MHz	450.05 MHz	clk_50M	limit due to minimum period restriction (max I/O toggle rate)

Clock to Output Times

	Data Port	Clock Port	Rise	Fall	Clock Edge	Clock Reference
1	flag	clk_50M	7.461	7.461	Fall	clk_50M
2	led[*]	clk_50M	7.474	7.474	Fall	clk_50M
1	led[0]	clk_50M	7.474	7.474	Fall	clk_50M
2	led[1]	clk_50M	7.448	7.448	Fall	clk_50M
3	led[2]	clk_50M	7.442	7.442	Fall	clk_50M

2. D 触发器

Flow Summary

Flow Status	Successful - Sun May 06 15:31:04 2018
Quartus II 64-Bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Web Edition
Revision Name	check_d
Top-level Entity Name	check_d
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Total logic elements	6 / 33,216 (< 1 %)
Total combinational functions	2 / 33,216 (< 1 %)
Dedicated logic registers	6 / 33,216 (< 1 %)
Total registers	6
Total pins	10 / 475 (2 %)
Total virtual pins	0
Total memory bits	0 / 483,840 (0 %)
Embedded Multiplier 9-bit elements	0 / 70 (0 %)
Total PLLs	0 / 4 (0 %)

Slow Model Fmax Summary

	Fmax	Restricted Fmax	Clock Name	Note
1	921.66 MHz	450.05 MHz	clk_50M	limit due to minimum period restriction (max I/O toggle rate)

Clock to Output Times

	Data Port	Clock Port	Rise	Fall	Clock Edge	Clock Reference
1	flag	clk_50M	8.919	8.919	Rise	clk_50M
2	⌄ q[*]	clk_50M	7.761	7.761	Rise	clk_50M
1	q[0]	clk_50M	7.691	7.691	Rise	clk_50M
2	q[1]	clk_50M	7.761	7.761	Rise	clk_50M
3	q[2]	clk_50M	7.706	7.706	Rise	clk_50M
4	q[3]	clk_50M	7.475	7.475	Rise	clk_50M
5	q[4]	clk_50M	7.474	7.474	Rise	clk_50M
6	q[5]	clk_50M	7.516	7.516	Rise	clk_50M

六、 问题及解决方案

整个设计过程出现了一个问题，就是在设计有限状态机的时候，因为最后一个状态输入 1 之后，flag 会输出 1，并且跳转到 S1 状态，我第一次设计直接就到 S0 状态了，这就导致后来不能识别连续的输入“101011”序列，经过检查改正之后，能准确无误地识别序列“101011”。