DIGITAL SYSTEM DESIGN APPLICATION – EHB 436E

Project 2

Yiğit Bektaş GÜRSOY

040180063

Class Lecturer: Sıddıka Berna Örs Yalçın

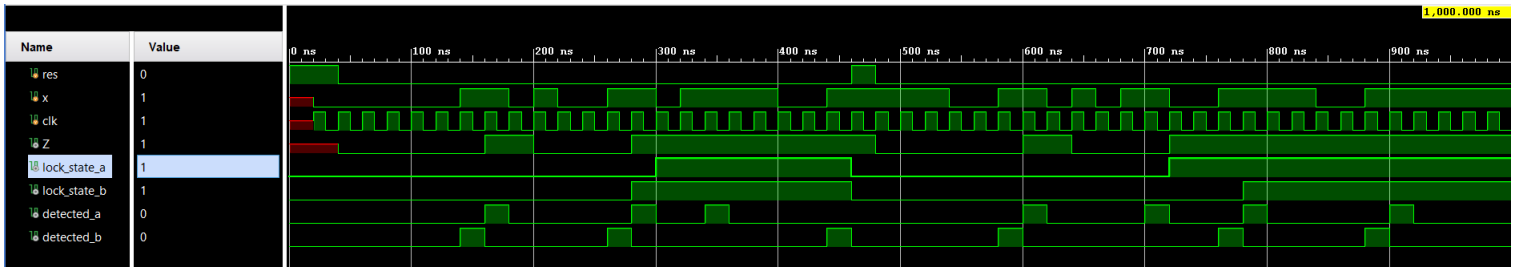
Class Assistant:
Serdar Duran
Yasin Fırat Kula
Mehmet Onur Demirtürk

- Explanation of Algorithm and Circuit
  I defined a state for each state. My school number is 040180063, from here I defined it as A = 6 and B = 3. Then I designed 2 FSMs that detect pattern "0110" and pattern "0011". Then, for the lock state, I made LOCKSTATE for A = 1 for A at any time when "0110" comes up again. Based on these, I designed 2 FSMs. I have designed 4 FSMs in total. I used binary coding for coding. This circuit I designed is Mealy circuit. Below are the boolean equations of the circuit I designed.

  Project2 Verilog Code

```
`timescale 1ns / 1ps

module PROJECT2(
    input res,
    input clk,
    input x,

    output reg Z
    );

    wire [7:0] Q;
    reg [7:0] q;
    wire detected_a, detected_b, lock_state_a, lock_state_b, z;

    // KARNAUGH MAP ASSIGN
    assign Q[0] = ~x || ( q[1] & ~q[0] ) ;
    assign Q[1] =  x & (q[1] ^ q[0]) ;

    assign Q[2] = ( ~x & ~q[3] & ~q[2] ) || ( x & q[3] & ~q[2] ) || ( ~x & q[3] & q[2] ) ;
    assign Q[3] = ( ~x & ~q[3] & q[2] ) || ( q[3] & ~q[2] ) ;

    assign Q[4] = ( detected_a & ~q[5] & ~q[4]) || ( ~detected_a & q[4] ) ;
    assign Q[5] = ( detected_a & q[4] ) || q[5] ;

    assign Q[6] = ( detected_b & ~q[7] & ~q[6]) || ( ~detected_b & q[6] ) ;
    assign Q[7] = ( detected_b & q[6] ) || q[7] ;

    // DETECTED ASSIGN FOR A AND B
    assign detected_a = x & q[1] & q[0] ;
    assign detected_b = x & q[3] & q[2] ;

    // LOCK STATE ASSIGN
    assign lock_state_a = q[5];
    assign lock_state_b = q[7];

    // OUTPUT ASSIGN
    assign z = lock_state_a || lock_state_b || detected_a || detected_b;

    always@ (posedge clk or posedge res)
    begin
        if (res)
        begin
            q[7:0] <= 8'b0000_0000;
        end

        else
        begin
            q[0] <= Q[0];
            q[1] <= Q[1];
            q[2] <= Q[2];
            q[3] <= Q[3];
            q[4] <= Q[4];
            q[5] <= Q[5];
            q[6] <= Q[6];
            q[7] <= Q[7];
            Z <= z;
        end
    end

endmodule
```

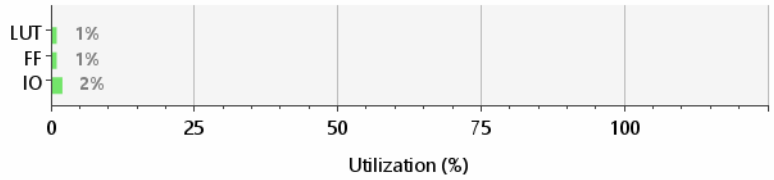
- Post-Implementation Timing Simulation
  The simulation result is shown below. The first thing to notice is that the detect a and detect b sections are detected one clock cycle early. This is due to the moore machine. By putting DFF at the output of the circuit, we made it return to the mealy machine, then our output gave the correct result. As you can see, when "0011" and "0110" are in the circuit, our circuit gives the result of 1.



- Utilization Report

Summary

Resource	Utilization	Available	Utilization %
LUT	7	20800	0.03
FF	9	41600	0.02
IO	4	210	1.90



- Setup Delays

Unconstrained Paths - NONE - NONE - Setup

Name	Slack	Levels	Routes	High Fanout	From	To	Total ...	Logic Delay	Net Delay	Requirement	Source Clock	Destir
Path 1	∞	2	1	1	Z_reg/C	Z	4.740	3.063	1.677	∞		
Path 2	∞	3	2	10	x	Z_reg/D	3.489	1.438	2.051	∞	input port clock	
Path 3	∞	2	2	10	x	q_reg[1]/D	3.084	1.080	2.004	∞	input port clock	
Path 4	∞	2	2	9	res	Z_reg/CE	2.796	1.076	1.720	∞	input port clock	
Path 5	∞	2	1	10	x	q_reg[3]/D	2.715	1.102	1.612	∞	input port clock	
Path 6	∞	2	1	10	x	q_reg[5]/D	2.713	1.108	1.604	∞	input port clock	
Path 7	∞	2	1	10	x	q_reg[2]/D	2.693	1.080	1.612	∞	input port clock	
Path 8	∞	2	1	10	x	q_reg[4]/D	2.685	1.080	1.604	∞	input port clock	
Path 9	∞	2	1	10	x	q_reg[0]/D	2.488	1.080	1.408	∞	input port clock	
Path 10	∞	2	1	10	x	q_reg[7]/D	2.365	1.109	1.256	∞	input port clock	

- Hold Delays

Unconstrained Paths - NONE - NONE - Hold

Name	Slack	Levels	Routes	High Fanout	From	To	Total ...	Logic Delay	Net Delay	Requirement	Source Clock
Path 19	∞	1	1	9	res	q_reg[0]/CLR	0.553	0.181	0.372	-∞	input port clock
Path 20	∞	1	1	9	res	q_reg[1]/CLR	0.553	0.181	0.372	-∞	input port clock
Path 18	∞	2	1	5	q_reg[2]/C	q_reg[2]/D	0.395	0.209	0.186	-∞	
Path 17	∞	2	1	5	q_reg[2]/C	q_reg[3]/D	0.393	0.207	0.186	-∞	
Path 16	∞	2	1	2	q_reg[4]/C	q_reg[5]/D	0.363	0.212	0.151	-∞	
Path 15	∞	2	1	2	q_reg[6]/C	q_reg[7]/D	0.362	0.213	0.149	-∞	
Path 14	∞	2	1	2	q_reg[4]/C	q_reg[4]/D	0.360	0.209	0.151	-∞	
Path 13	∞	2	1	2	q_reg[6]/C	q_reg[6]/D	0.358	0.209	0.149	-∞	
Path 12	∞	2	1	5	q_reg[1]/C	q_reg[0]/D	0.330	0.246	0.084	-∞	
Path 11	∞	2	1	5	q_reg[2]/C	Z_reg/D	0.314	0.209	0.105	-∞	

Maximum Clock Frequency of the circuit is 1/max delay. f = 1/4.74 ns = 216 MHz

- Timing Constrait
  Maximum delay is set to 4.6ns but delays did not change. Results are given below.

```
set_max_delay -from [get_ports {clk res x}] -to [get_ports Z] 4.600
```

Unconstrained Paths - NONE - NONE - Setup

Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	De
Path 1	∞	2	1	1	Z_reg/C	Z	4.762	3.085	1.677	∞		
Path 2	∞	3	2	10	x	Z_reg/D	3.266	1.460	1.806	∞	input port clock	
Path 3	∞	2	2	9	res	Z_reg/CE	2.668	1.094	1.575	∞	input port clock	
Path 4	∞	2	2	10	x	q_reg[1]/D	2.554	1.100	1.453	∞	input port clock	
Path 5	∞	2	1	10	x	q_reg[0]/D	2.397	1.100	1.297	∞	input port clock	
Path 6	∞	2	1	10	x	q_reg[7]/D	2.347	1.129	1.218	∞	input port clock	
Path 7	∞	2	1	10	x	q_reg[6]/D	2.318	1.100	1.218	∞	input port clock	
Path 8	∞	1	1	9	res	q_reg[0]/CLR	2.254	0.970	1.285	∞	input port clock	
Path 9	∞	1	1	9	res	q_reg[1]/CLR	2.254	0.970	1.285	∞	input port clock	
Path 10	∞	1	1	9	res	q_reg[2]/CLR	2.254	0.970	1.285	∞	input port clock	

- State Machine Type
  As I mentioned above, when using the Moore machine, the pattern a and b was determined by 1 clock cycle, and when we converted it to the Mealy machine, this situation was corrected. I realized this situation by adding DFF to the output of our circuit.