UNIVERSITY OF TEHRAN

Electrical and Computer Engineering Department

Digital Logic Design, ECE 367, Spring 1399-1400

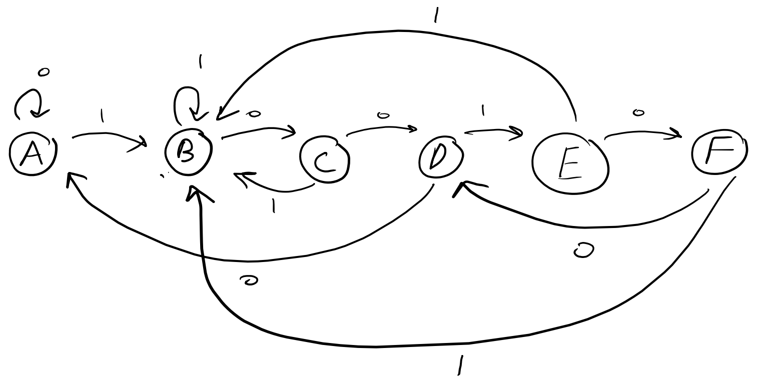
Computer Assignment 5

Mohamad Abaeiani

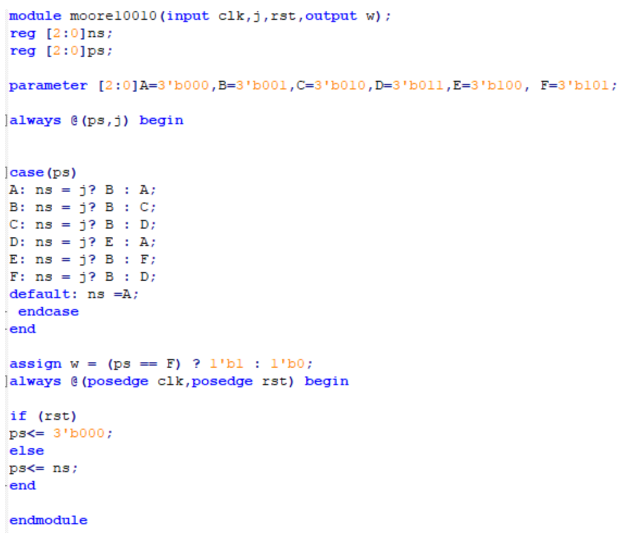
810198432

a.

Moore 10010 state diagram:

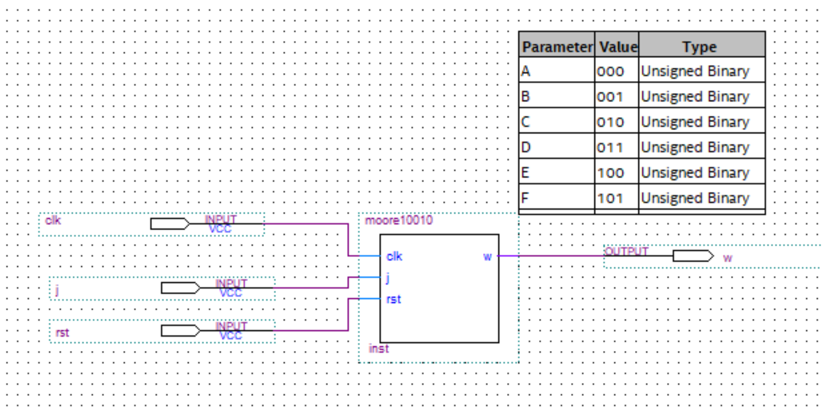


Code:



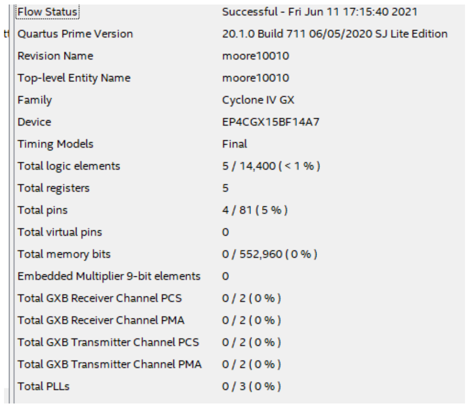
```
module moore10010(input clk,j,rst,output w);
reg [2:0]ns;
reg [2:0]ps;

parameter [2:0]A=3'b000,B=3'b001,C=3'b010,D=3'b011,E=3'b100, F=3'b101;

always @(ps,j) begin


case(ps)
A: ns = j? B : A;
B: ns = j? B : C;
C: ns = j? B : D;
D: ns = j? E : A;
E: ns = j? B : F;
F: ns = j? B : D;
default: ns =A;
 endcase
end

assign w = (ps == F) ? 1'b1 : 1'b0;
always @(posedge clk,posedge rst) begin

if (rst)
ps<= 3'b000;
else
ps<= ns;
end

endmodule
```

Symbol:



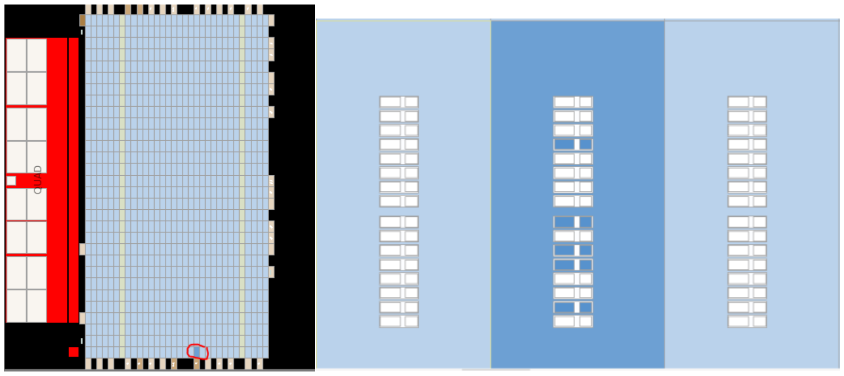
Parameter	Value	Type
A	000	Unsigned Binary
B	001	Unsigned Binary
C	010	Unsigned Binary
D	011	Unsigned Binary
E	100	Unsigned Binary
F	101	Unsigned Binary

Synthesis report:



Flow Status	Successful - Fri Jun 11 17:15:40 2021
Quartus Prime Version	20.1.0 Build 711 06/05/2020 SJ Lite Edition
Revision Name	moore10010
Top-level Entity Name	moore10010
Family	Cyclone IV GX
Device	EP4CGX15BF14A7
Timing Models	Final
Total logic elements	5 / 14,400 (< 1 %)
Total registers	5
Total pins	4 / 81 (5 %)
Total virtual pins	0
Total memory bits	0 / 552,960 (0 %)
Embedded Multiplier 9-bit elements	0
Total GXB Receiver Channel PCS	0 / 2 (0 %)
Total GXB Receiver Channel PMA	0 / 2 (0 %)
Total GXB Transmitter Channel PCS	0 / 2 (0 %)
Total GXB Transmitter Channel PMA	0 / 2 (0 %)
Total PLLs	0 / 3 (0 %)

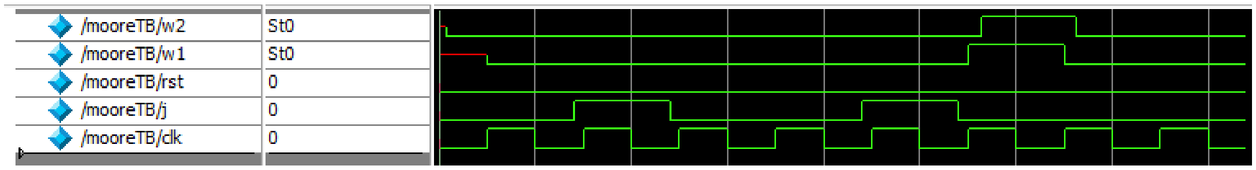
Floor plan:



As we could see the design used 5 cells (which include logic elements and registers) to implement the functionality which is the same as compilation report. It has also used some ports to get and send signals.

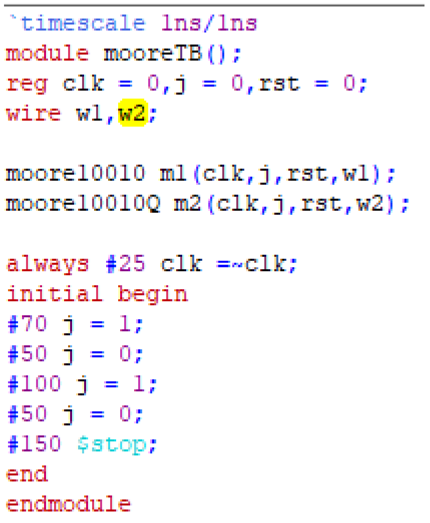
Pre and post synthesis waveform:

(w2 is post synthesis and w1 is pre synthesis)



As we could see the post synthesis output triggers a bit after the clock posedge.

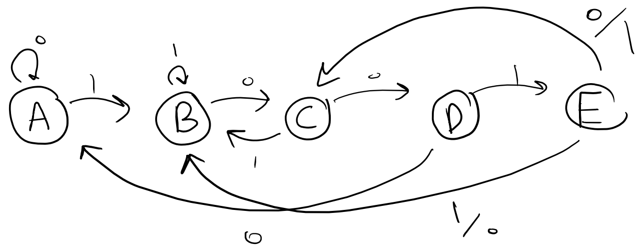
Test Bench Code:



```
`timescale 1ns/1ns
module mooreTB();
reg clk = 0,j = 0,rst = 0;
wire w1,w2;

moore10010 m1(clk,j,rst,w1);
moore10010Q m2(clk,j,rst,w2);

always #25 clk =~clk;
initial begin
#70 j = 1;
#50 j = 0;
#100 j = 1;
#50 j = 0;
#150 $stop;
end
endmodule
```

b.

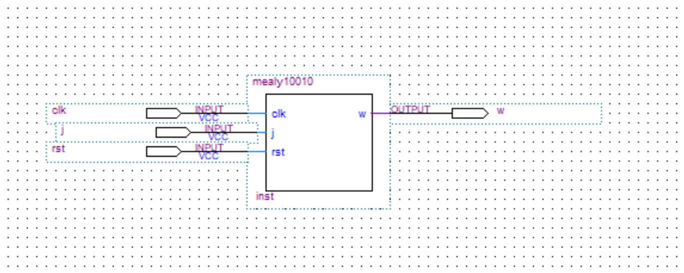
state diagram:



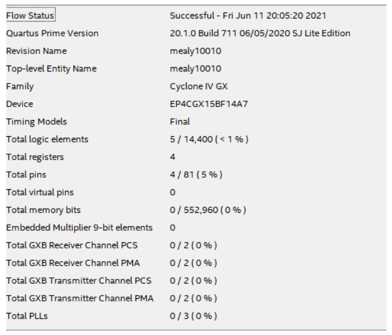
Code:

```
module mealy10010(input clk,j,rst,output w);
reg [2:0]ns;
reg [2:0]ps;

always @(ps,j) begin

ns = 3'b000;

case(ps)

3'b000: ns = j? 3'b001 : 3'b000;
3'b001: ns = j? 3'b001 : 3'b010;
3'b010: ns = j? 3'b001 : 3'b011;
3'b011: ns = j? 3'b100 : 3'b000;
3'b100: ns = j? 3'b001 : 3'b010;
 endcase
end

assign w = (ps == 3'b100) ?~j : 1'b0;

always @(posedge clk,posedge rst) begin

if (rst)
ps<= 3'b000;
else
ps<= ns;
end

endmodule
```

Symbol:

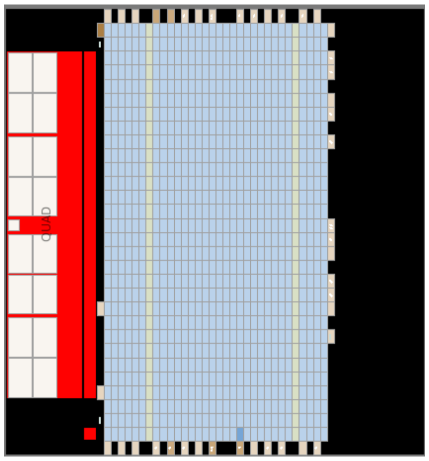


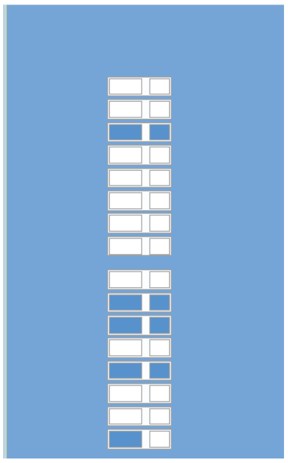
Synthesis compile report:



Flow Status	Successful - Fri Jun 11 20:05:20 2021
Quartus Prime Version	20.1.0 Build 711 06/05/2020 SJ Lite Edition
Revision Name	mealy10010
Top-level Entity Name	mealy10010
Family	Cyclone IV GX
Device	EP4CGX15BF14A7
Timing Models	Final
Total logic elements	5 / 14,400 (< 1 %)
Total registers	4
Total pins	4 / 81 (5 %)
Total virtual pins	0
Total memory bits	0 / 552,960 (0 %)
Embedded Multiplier 9-bit elements	0
Total GXB Receiver Channel PCS	0 / 2 (0 %)
Total GXB Receiver Channel PMA	0 / 2 (0 %)
Total GXB Transmitter Channel PCS	0 / 2 (0 %)
Total GXB Transmitter Channel PMA	0 / 2 (0 %)
Total PLLs	0 / 3 (0 %)

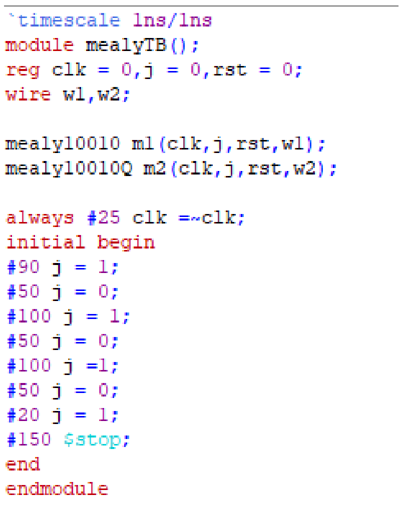
Floor plan:





As we could conclude there are 4 registers and 5 logic elements used in this synthesis, so it’s the same as the cmpile report(also there are some ports used to deliver data as the previous part)

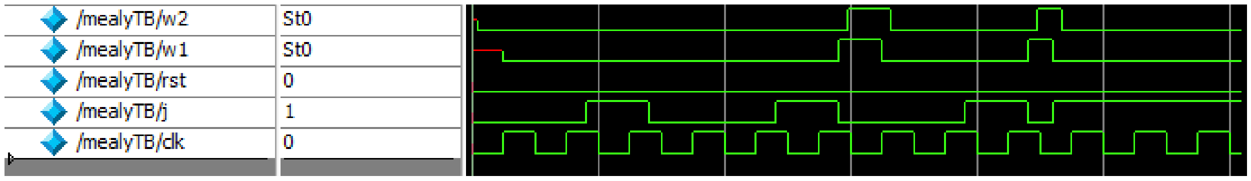
Test Bench:



```
`timescale 1ns/1ns
module mealyTB();
reg clk = 0,j = 0,rst = 0;
wire w1,w2;

mealy10010 m1(clk,j,rst,w1);
mealy10010Q m2(clk,j,rst,w2);

always #25 clk =~clk;
initial begin
#90 j = 1;
#50 j = 0;
#100 j = 1;
#50 j = 0;
#100 j =1;
#50 j = 0;
#20 j = 1;
#150 $stop;
end
endmodule
```

Wave Form:

(w2 is post synthesis and w1 is pre synthesis)



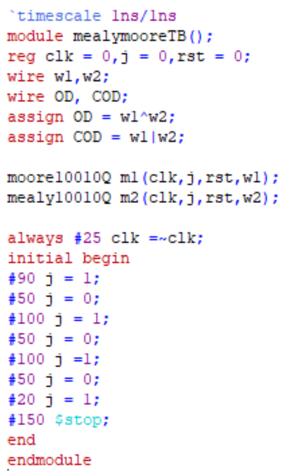
As we could see the post synthesis output triggers a bit after the clock posedge.

Also, as it is expected from mealy behavior the output followed j signal and became 0 a bit sooner than the next clock posedge, leading to smaller on-time than j input.

And it may cause glitches by invalid input changes.

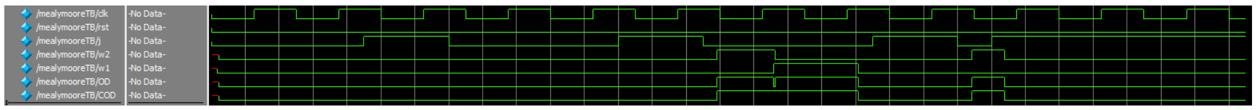
C.

Code(testbench):



```
`timescale 1ns/1ns
module mealymooreTB();
reg clk = 0,j = 0,rst = 0;
wire w1,w2;
wire OD, COD;
assign OD = w1^w2;
assign COD = w1|w2;

moore10010Q m1(clk,j,rst,w1);
mealy10010Q m2(clk,j,rst,w2);

always #25 clk =~clk;
initial begin
#90 j = 1;
#50 j = 0;
#100 j = 1;
#50 j = 0;
#100 j =1;
#50 j = 0;
#20 j = 1;
#150 $stop;
end
endmodule
```

Waveform:



OD is for output difference signal and COD is for clean output difference signal.

In the first transition of OD the output difference is due to difference in moore and mealy timing so it can be ignored, but the next transition is where mealy missed the clock and gave faulty output, and this should be considered.

By looking precisely at the OD we see that the is a small gap between mealy and moore transitions, and it is where both outputs are 1 because of gate delay differences, so buy changing xor with or this problem would be solved and we get to COD output.

By looking at COD we can conclude that if the ON-time is less than one clock cycle hen there is a fault in mealy and it should be considered