ELL201 EXPERIMENT 6

Kanishk Singhal
2019MT10698

May 2021

1 Question 1

1.1 State Table

Q_3^n	Q_2^n	Q_1^n	Q_0^n	Q_3^{n+1}	Q_2^{n+1}	Q_1^{n+1}	Q_0^{n+1}
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	1
0	0	1	1	0	0	1	0
0	0	1	0	0	1	1	0
0	1	1	0	0	1	1	1
0	1	1	1	0	1	0	1
0	1	0	1	0	1	0	0
0	1	0	0	1	1	0	0
1	1	0	0	1	1	0	1
1	1	0	1	1	1	1	1
1	1	1	1	1	1	1	0
1	1	1	0	1	0	1	0
1	0	1	0	1	0	1	1
1	0	1	1	1	0	0	1
1	0	0	1	1	0	0	0
1	0	0	0	0	0	0	0

1.2 Number of SR Flip Flops

The number of bits required are $log_2(16)$, which is equal to 4, hence the number of SR Flip Flops required are 4

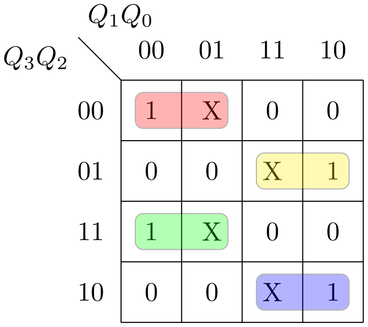
1.3 Assigning value to SR Flip Flops

Q$_3^n$	Q$_2^n$	Q$_1^n$	Q$_0^n$	Q$_3^{n+1}$	Q$_2^{n+1}$	Q$_1^{n+1}$	Q$_0^{n+1}$	S$_3$	R$_3$	S$_2$	R$_2$	S$_1$	R$_1$	S$_0$	R$_0$
0	0	0	0	0	0	0	1	0	X	0	X	0	X	1	0
0	0	0	1	0	0	1	1	0	X	0	X	1	0	X	0
0	0	1	1	0	0	1	0	0	X	0	X	X	0	0	1
0	0	1	0	0	1	1	0	0	X	1	0	X	0	0	X
0	1	1	0	0	1	1	1	0	X	X	0	X	0	1	0
0	1	1	1	0	1	0	1	0	X	X	0	0	1	X	0
0	1	0	1	0	1	0	0	0	X	X	0	0	X	0	1
0	1	0	0	1	1	0	0	1	0	X	0	0	X	0	X
1	1	0	0	1	1	0	1	X	0	X	0	0	X	1	0
1	1	0	1	1	1	1	1	X	0	X	0	1	0	X	0
1	1	1	1	1	1	1	0	X	0	X	0	X	0	0	1
1	1	1	0	1	0	1	0	X	0	0	1	X	0	0	X
1	0	1	0	1	0	1	1	X	0	0	X	X	0	1	0
1	0	1	1	1	0	0	1	X	0	0	X	0	1	X	0
1	0	0	1	1	0	0	0	X	0	0	X	0	X	0	1
1	0	0	0	0	0	0	0	0	1	0	X	0	X	0	X

1.4 Using K-Maps

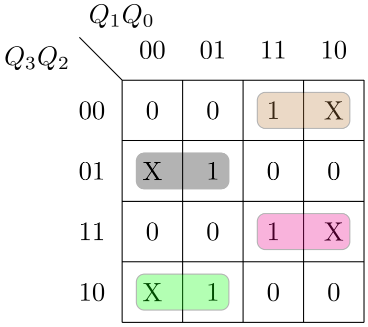
The values of:

- $S_3 = Q_2Q_1'Q_0'$
- $S_3 = Q_2Q_1'Q_0'$
- $R_3 = Q_2'Q_1'Q_0'$
- $S_2 = Q_3'Q_1Q_0'$
- $R_2 = Q_3Q_1Q_0'$
- $S_1 = Q_3'Q_2'Q_0 + Q_3Q_2Q_0$
- $R_1 = Q_3Q_2'Q_0 + Q_3'Q_2Q_0$
- $S_0 = Q_3'Q_2Q_1 + Q_3'Q_2'Q_1' + Q_3Q_2Q_1' + Q_3Q_2'Q_1$
- $R_0 = Q_3'Q_2Q_1' + Q_3'Q_2'Q_1 + Q_3Q_2Q_1 + Q_3Q_2'Q_1'$



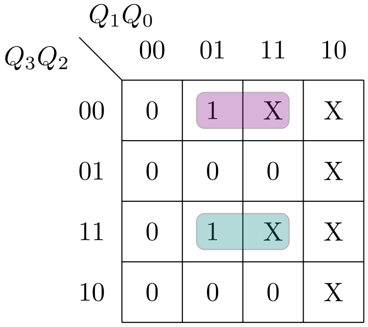
Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	1	X	0	0
01	0	0	X	1
11	1	X	0	0
10	0	0	X	1

(a) K Map for S_0



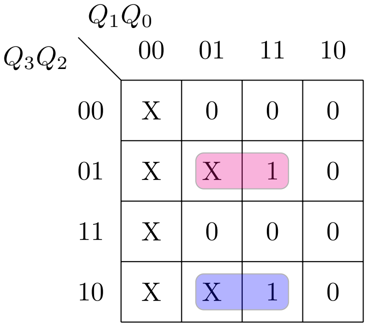
Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	0	0	1	X
01	X	1	0	0
11	0	0	1	X
10	X	1	0	0

(b) K Map for R_0



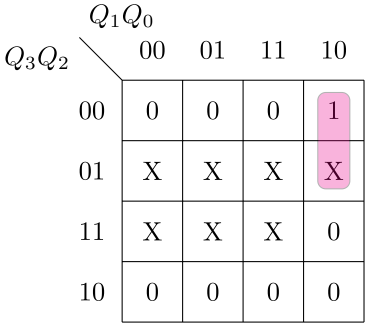
Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	0	1	X	X
01	0	0	0	X
11	0	1	X	X
10	0	0	0	X

(c) K Map for S_1



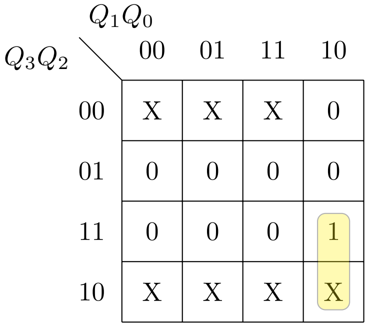
Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	X	0	0	0
01	X	X	1	0
11	X	0	0	0
10	X	X	1	0

(d) K Map for R_1



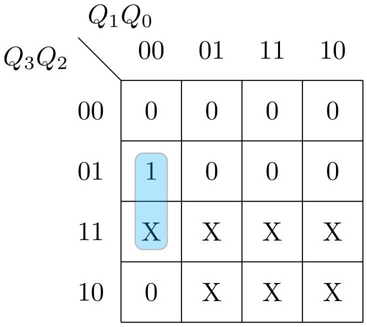
Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	0	0	0	1
01	X	X	X	X
11	X	X	X	0
10	0	0	0	0

(e) K Map for S_2



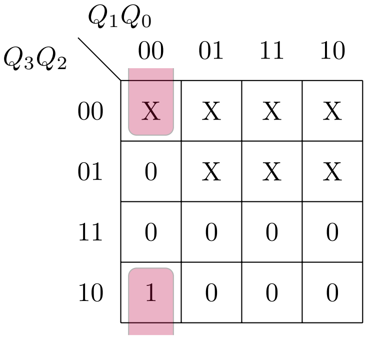
Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	X	X	X	0
01	0	0	0	0
11	0	0	0	1
10	X	X	X	X

(f) K Map for R_2



Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	0	0	0	0
01	1	0	0	0
11	X	X	X	X
10	0	X	X	X

(g) K Map for S_3



Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	X	X	X	X
01	0	X	X	X
11	0	0	0	0
10	1	0	0	0

(h) K Map for R_3

1.5 Verilog code

```verilog
module srff(input S, input R, input clk, input rst, output reg Q, output Qbar);
always @(posedge clk)
if (!rst)
    Q <= 0;
else begin
    if(!S & !R)
        Q <= Q;
    else if(!S & R)
        Q <= 0;
    else if(S & !R)
        Q <= 1;
end
assign Qbar = !Q;
endmodule
module graycode(input rst, input clk, output [3:0] out);
wire r0,r1,r2,r3,s0,s1,s2,s3,s4,q0,q1,q2,q3,qn0,qn1,qn2,qn3;
assign s0 = (!q3 & !q2 & !q1) | (!q3 & q2 & q1) | (q3 & !q2 & q1) | (q3 & q2 & !q1);
assign r0 = (!q3 & !q2 & q1) | (!q3 & q2 & !q1) | (q3 & !q2 & !q1) | (q3 & q2 & q1);

assign s1 = (!q3 & !q2 & q0) | (q3 & q2 & q0);
assign r1 = (!q3 & q2 & q0) | (q3 & !q2 & q0);
assign s2 = (!q3 & q1 & !q0);
assign r2 = (q3 & q1 & !q0);
assign s3 = (q2 & !q1 & !q0);
assign r3 = (!q2 & !q1 & !q0);

srff sr3 (s3, r3, clk, rst, q3, qn3);
srff sr2 (s2, r2, clk, rst, q2, qn2);
srff sr1 (s1, r1, clk, rst, q1, qn1);
srff sr0 (s0, r0, clk, rst, q0, qn0);

assign out = {q3, q2, q1, q0};

endmodule

//The code written below is for test bench
module tb_graycode;
    reg clk;
    reg rstn;
    wire [3:0] out;
    graycode counter (rstn, clk, out);
    always #5 clk = ~clk;
    initial begin
      $dumpfile("graycode.vcd");
                        $dumpvars(0, tb_graycode);
      $monitor($time,"␣%b", out);
      rstn <= 0;
      clk <= 0;
      repeat (1) @ (posedge clk);
      rstn <= 1;
      repeat (17) @ (posedge clk);
      $finish;
   end
endmodule
```

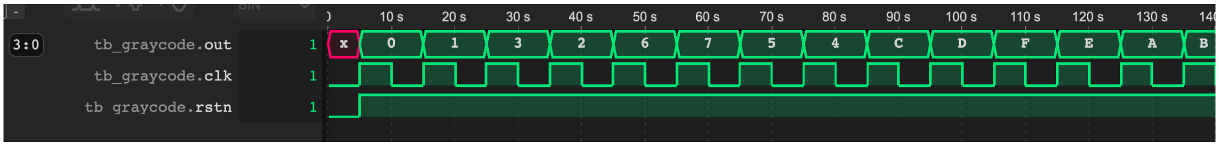


Figure 2: Waveform of graycode

2 Question 2

2.1 Number of D Flip Flops

The number of DFF required are 4. As, the number of states are 15, so number of DFF = $\lceil log_2 15 \rceil$ which is equal to 4

2.2 Assigning value to D

Q3	Q2	Q1	Q0	D3	D2	D1	D0
0	0	0	1	1	0	0	0
1	0	0	0	0	1	0	0
0	1	0	0	0	0	1	0
0	0	1	0	1	0	0	1
1	0	0	1	1	1	0	0
1	1	0	0	0	1	1	0
0	1	1	0	1	0	1	1
1	0	1	1	0	1	0	1
0	1	0	1	1	0	1	0
1	0	1	0	1	1	0	1
1	1	0	1	1	1	1	0
1	1	1	0	1	1	1	1
1	1	1	1	0	1	1	1
0	1	1	1	0	0	1	1
0	0	1	1	0	0	0	1

The last digit of my entry number is 8, thus the counter starts from 1000. Hence, the transition would be from 1000 → 0100 → 0010 → 1001 → 1100 → 0110 → 1011 → 0101 → 1010 → 1101 → 1110 → 1111 → 0111 → 0011 → 0001 → 1000 and then repeating itself

No, the counter doesn't cover all the 16 states. The state 0000 is not covered. The counter is behaving as a mod 15 counter, as it is repeating the states after 15 transitions. It doesn't depend on the initial state, as it repeats itself after 15 transitions covering all the 15 states.

The values of:

- $D_3 = Q_0 \oplus Q_1$
- $D_2 = Q_3$
- $D_1 = Q_2$
- $D_0 = Q_1$

Q_1Q_0 Q_3Q_2	00	01	11	10
00	X	0	1	1
01	0	0	1	1
11	0	0	1	1
10	0	0	1	1

(a) K Map for D_0

Q_1Q_0 Q_3Q_2	00	01	11	10
00	X	0	0	0
01	1	1	1	1
11	1	1	1	1
10	0	0	0	0

(b) K Map for D_1

Q_1Q_0 Q_3Q_2	00	01	11	10
00	X	0	0	0
01	0	0	0	0
11	1	1	1	1
10	1	1	1	1

1

(c) K Map for D_2

Q_1Q_0 Q_3Q_2	00	01	11	10
00	X	1	0	1
01	0	1	0	1
11	0	1	0	1
10	0	1	0	1

(d) K Map for D_3

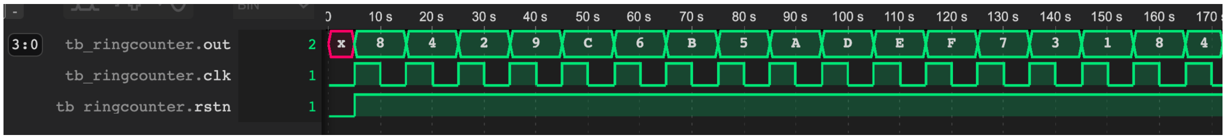


Figure 4: Waveform of ring counter

2.3 Verilog code

```verilog
module dff(input D, input iniv, input clk, input rst, output reg Q, output Qbar);
always @(posedge clk)
if(!rst)
    Q <= iniv;
else
    Q <= D;
assign Qbar = !Q;
endmodule

module ringcounter(input rst, input clk, output [3:0] out);

wire d0,d1,d2,d3,q0,q1,q2,q3,qn0,qn1,qn2,qn3;

assign d0 = q1;
assign d1 = q2;
assign d2 = q3;
assign d3 = q1^q0;

dff df0 (d0, 1'b1, clk, rst, q0, qn0);
dff df1 (d1, 1'b0, clk, rst, q1, qn1);
dff df2 (d2, 1'b0, clk, rst, q2, qn2);
dff df3 (d3, 1'b0, clk, rst, q3, qn3);

assign out = {d3,d2,d1,d0};
endmodule

//the code written below is for test bench
module tb_ringcounter;
    reg clk;
    reg rstn;
    wire [3:0] out;
    ringcounter counter (rstn, clk, out);
    always #5 clk = ~clk;
    initial begin
      $dumpfile("ringcounter.vcd");
                        $dumpvars(0, tb_ringcounter);
      $monitor($time,"␣%b", out);
      rstn <= 0;
      clk <= 0;
      repeat (1) @ (posedge clk);
      rstn <= 1;
      repeat (17) @ (posedge clk);
      $finish;
   end
endmodule
```

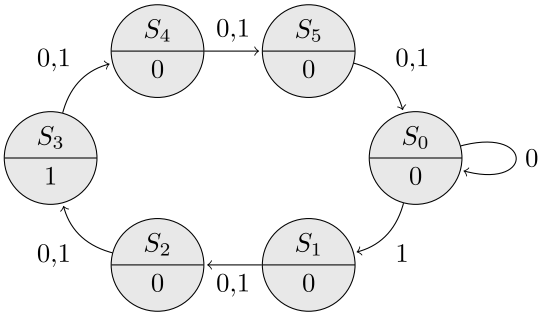
ELL201 EXPERIMENT 7

Kanishk Singhal
2019MT10698

May 2021

1 The reduced state diagram for the Moore Machine

My entry number is 2019MT10698, thus the sequence to be generated is {0,0,0,1,0,0}



2 Number of flip flops

The total number of states in the reduced state diagram are 6, thus the number of flip flops used are $\lceil log_2 6 \rceil$, which is equal to 3.

3 Assigning values to the states

The values of:

- $S_0 = 000$
- $S_1 = 001$
- $S_2 = 010$
- $S_3 = 011$
- $S_4 = 100$
- $S_5 = 101$

4 State Table

Q$_2^n$	Q$_1^n$	Q$_0^n$	X	Q$_2^{n+1}(D_2)$	Q$_1^{n+1}(D_1)$	Q$_0^{n+1}(D_0)$	Y
0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	0
0	0	1	0	0	1	0	0
0	0	1	1	0	1	0	0
0	1	0	0	0	1	1	1
0	1	0	1	0	1	1	1
0	1	1	0	1	0	0	0
0	1	1	1	1	0	0	0
1	0	0	0	1	0	1	0
1	0	0	1	1	0	1	0
1	0	1	0	0	0	0	0
1	0	1	1	0	0	0	0

5 K-Maps

The values of:

- $D_2 = Q_2Q_0' + Q_1Q_0$
- $D_1 = Q_1Q_0' + Q_2'Q_1'Q_0$
- $D_0 = Q_0'X + Q_1Q_0' + Q_2Q_0'$
- $Y = Q_1Q_0'$

Q_2Q_1 \ Q_0X	00	01	11	10
00	0	0	0	0
01	0	0	1	1
11	X	X	X	X
10	1	1	0	0

(a) K Map for D_2

Q_2Q_1 \ Q_0X	00	01	11	10
00	0	0	1	1
01	1	1	0	0
11	X	X	X	X
10	0	0	0	0

(b) K Map for D_1

Q_2Q_1 \ Q_0X	00	01	11	10
00	0	1	0	0
01	1	1	0	0
11	X	X	X	X
10	1	1	0	0

(c) K Map for D_0

Q_2 \ Q_1Q_0	00	01	11	10
0	0	0	0	1
1	0	0	X	X

(d) K Map for Y

6 Verilog Code

```verilog
module dff(input D, input iniv, input clk, input rst, output reg Q, output Qbar);
always @(posedge clk)
if(!rst)
    Q <= iniv;
else
    Q <= D;
assign Qbar = !Q;
endmodule

module fsm(input rst, input clk, input X, output Y);

wire d0,d1,d2,q0,q1,q2,q3,qn0,qn1,qn2,qn3;

assign d0 = (!q0 & X) | (q1 & !q0) | (q2 & !q0);
assign d1 = (q1 & !q0) | (!q2 & !q1 & q0);
assign d2 = (q2 & !q0) | (q1 & q0);


dff df0 (d0, 1'b0, clk, rst, q0, qn0);
dff df1 (d1, 1'b0, clk, rst, q1, qn1);
dff df2 (d2, 1'b0, clk, rst, q2, qn2);

assign Y = q1 & !q0;
endmodule

//The code below is for test bench
module tb_fsm;
    reg clk;
    reg rstn;
    reg X;
    wire out;
    fsm counter (rstn, clk, X, out);
    always #5 clk = ~clk;
    always #30 X = !X;
    initial begin
      $dumpfile("fsm.vcd");
                        $dumpvars(0, tb_fsm);
      $monitor($time,"␣%b", out);
      rstn <= 0;
      clk <= 0;
      X <= 1;
      repeat (1) @ (posedge clk);
      rstn <= 1;
      repeat (17) @ (posedge clk);
      $finish;
   end
endmodule
```

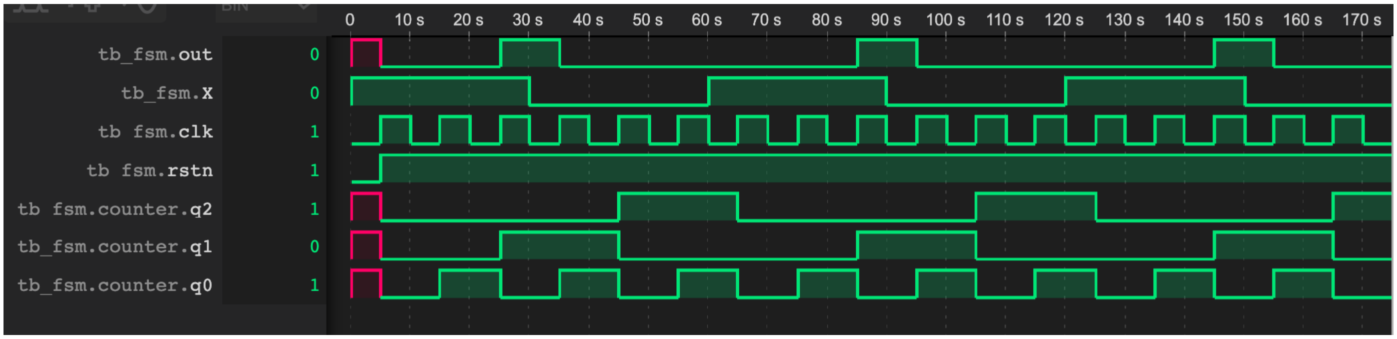


Figure 2: The output waveform