ELP201 Lab Report 3

Adit Malhotra

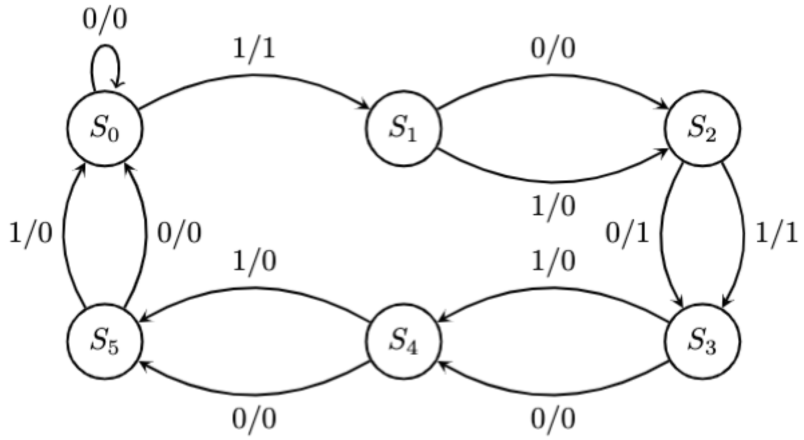
2020EE10458

1 Introduction

According to my entry number ($2020EE10458$), the sequence to be generated at output side is $1, 0, 1, 0, 0, 0$.

2 State diagram

In total there would be 6 states, let them S_i, where $i \in \{0, 1, 2, 3, 4, 5\}$. Let S_0 be the idle state. Then according to the statement of question the state diagram would be:



Since total number of states are 6 and $6 \leqslant 2^3$, hence total of 3 flipflops would be needed. Let's assign each state S_i the binary value of i.

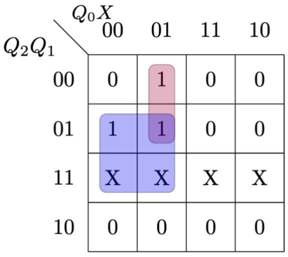
3 State table

The outputs $Q_2Q_1Q_0$, of the D-flipflops would represent the state at any time t. And since the characteristic equation for the D-flipflop is $Q(t+1) = D$, then the state table would be:

$Q_2(t)$	$Q_1(t)$	$Q_0(t)$	$X(in)$	$Q_2(t+1)$	$Q_1(t+1)$	$Q_0(t+1)$	$Y(out)$	D_2	D_1	D_0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	1	0	0	1
0	0	1	0	0	1	0	0	0	1	0
0	0	1	1	0	1	0	0	0	1	0
0	1	0	0	0	1	1	1	0	1	1
0	1	0	1	0	1	1	1	0	1	1
0	1	1	0	1	0	0	0	1	0	0
0	1	1	1	1	0	0	0	1	0	0
1	0	0	0	1	0	1	0	1	0	1
1	0	0	1	1	0	1	0	1	0	1
1	0	1	0	0	0	0	0	0	0	0
1	0	1	1	0	0	0	0	0	0	0

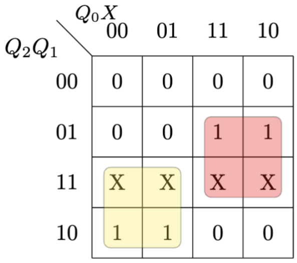
4 Simplification using K-maps

From the state table, the k-maps for *Output* $(Y), D_2, D_1, D_0$ are as follows:



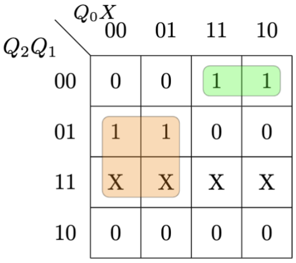
Q_2Q_1 \ Q_0X	00	01	11	10
00	0	1	0	0
01	1	1	0	0
11	X	X	X	X
10	0	0	0	0

Figure 1: K-Map for *Output* (Y)



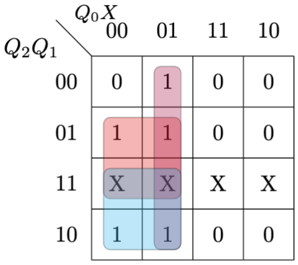
Q_2Q_1 \ Q_0X	00	01	11	10
00	0	0	0	0
01	0	0	1	1
11	X	X	X	X
10	1	1	0	0

Figure 2: 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

Figure 3: 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

Figure 4: K-Map for D_0

And from the k-maps the simplified expressions are as follows:

$$Y = \bar{Q}_2\bar{Q}_0X \ + \ Q_1\bar{Q}_0$$

$$D_2 = Q_2\bar{Q}_0 \ + \ Q_0Q_1$$

$$D_1 = \bar{Q}_2\bar{Q}_1Q_0 \ + \ Q_1\bar{Q}_0$$

$$D_0 = \bar{Q}_0X \ + \ Q_1\bar{Q}_0 \ + \ Q_2\bar{Q}_0$$

5 Verilog Simulation

5.1 Code

The code file (design.v) for the implementation of the above finite state machine using D-flipflops is: The D-flipflop designed is a postive edge triggered one and has a reset pin which is active low.

```verilog
module DFlipFlop(D,clock,preset,Q,Qbar);
input D,clock,preset;
output reg Q,Qbar;

initial
begin
  if(~preset)
  begin
    Q=1;Qbar=0;
  end
  else
  begin
    Q=0;Qbar=1;
  end
end

always@(posedge clock)
begin
  Q <= D;
  Qbar <= ~D;
end
endmodule

module FSM(in,out,clock,q2,q1,q0,q2bar,q1bar,q0bar);
input in,clock;
output wire q2,q1,q0,q2bar,q1bar,q0bar;
output reg out=0;

DFlipFlop ff1((q0bar & in)|(q1 & q0bar)|(q2 & q0bar),clock,1'b1,q0,q0bar);
DFlipFlop ff2((q2bar & q1bar & q0)|(q1 & q0bar),clock,1'b1,q1,q1bar);
DFlipFlop ff3((q2 & q0bar)|(q1 &q0),clock,1'b1,q2,q2bar);
always @(posedge clock )
begin
  assign out=(q1 & q0bar)|(q2bar & q0bar & in);
end
endmodule
```

The code file for the testbench (main.v) is: Intitially the input (X) is kept 1 for atleast 6 clock cycles and when it reaches the idle state again, it is then made 0 for again atleast 6 clock cycles, and finally it is again made 1 for one clock cycle so that machine again becomes active then immediately made 0 for again atleast 6 clock cycles. The D-flipflop designed is a postive edge triggered one and has a reset pin which is active low.

```verilog
`timescale 1s/100ms
`include "design.v"

module main();

reg clk,in;
wire q0,q1,q2,q2bar,q1bar,q0bar,out;
```

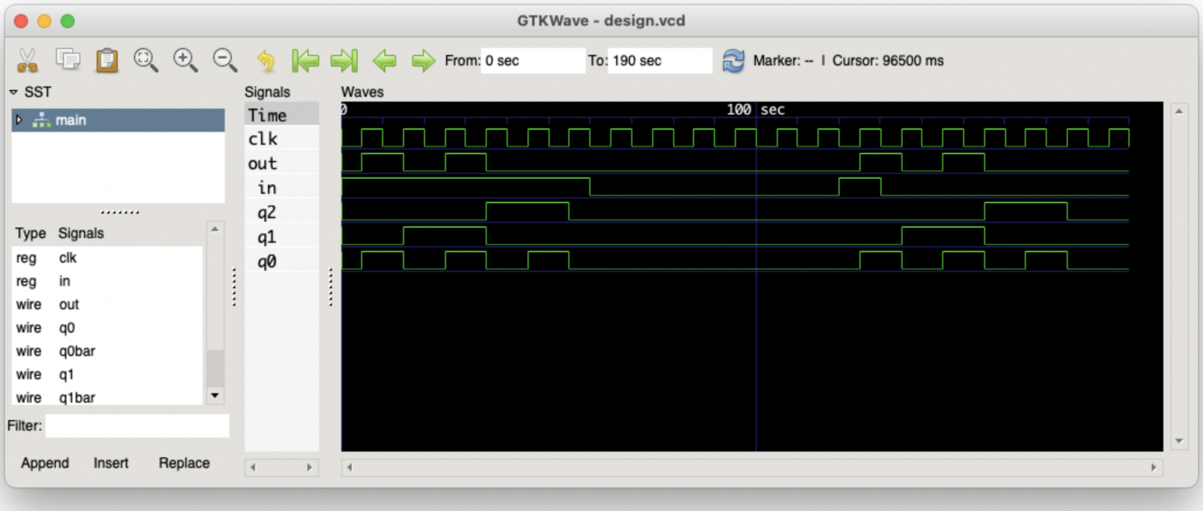
```

//Instantating the finite state machine
FSM func(in,out,clk,q2,q1,q0,q2bar,q1bar,q0bar);

initial
begin
$monitor("CLK=%b,out=%b,in=%b,q2=%b,q1=%b,q0=%b",clk,out,in,q2,q1,q0);
$dumpfile("design.vcd");
$dumpvars(0,main);
  clk=0; //initially clock=0;
  in=1;#60 //initially input=1;
  in=0;#60; //after running the simulation for 60s, input is made 0;
  in=1;#10;in=0;
  #60;
  $finish;
end
always
  #5 clk = ~clk; //changing clock after every 5s
endmodule
```

5.2 Waveform

The gtkwave plot obtained is:



Hence it can be observed from the plot, that the FSM once starts generating the sequence, keep generating it no matter whatever is the value of X is and also it remains in the idle state as long as X is kept 0. And at last the sequence generated is $1, 0, 1, 0, 0, 0$ at the output side. Hence, the FSM is working perfectly as required.

Download link for the code files

The code files can be downloaded from **here**.