Prova 2 / 2017 INF 251 Nome e Matricula _____

1. Projete uma máquina de estados para gerar as seguintes sequências: Se A=0, **9** → **8** → **7** → **8** → 9 → 8 → 7 → 8 …. E Se A=1, **7** → **8** → **9** → 7 → 8 → 9 …. Desenhe o diagrama de estados, a tabela, as equações para as funções de estado e saída, desenhe o circuito com portas lógicas e flipflops. Faça também o desenho com memória e flipflops e preencha a memória.

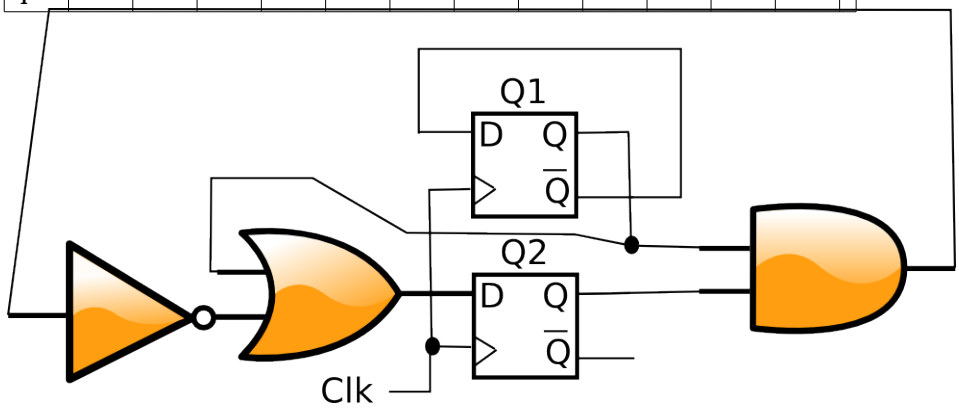
2. Baseado no código Verilog do anexo, projete o modulo de maquina de estados para o contador da questão 1.

3. Suponha a tabela abaixo com o contéudo da memória de uma máquina de estados, onde os 2 primeiros bits são o próximo estado e os três ultimos são as saídas. O endereço da memória é a concatenação do estado atual E1, E0 e a entrada A. Por exemplo, na linha 2 (010 em binário), o estado é 01 e a entrada A tem o valor 0. Desenhe o diagrama de estados da máquina. O estado inicial é 00.

					Numero da linha
1	0	1	0	1	0
0	1	1	0	1	1
0	1	0	1	1	2 (010) onde E1E0 = 01, A = 0
0	0	0	1	1	3
1	0	1	0	1	4
0	1	1	0	1	5
1	0	0	0	1	6
0	1	0	0	1	7

4. Suponha que Q1 e Q2 sejam inicialmente 0. Preencha a tabela com os valores de Q1 e Q2 para o circuito abaixo.

clk	0	1	0	1	0	1	0	1	0	1	0	1	
q1	0												
q2	0												



```
module fsm_m (clk, reset,End , ch , done, chout, rst);
input clk, reset, End, ch;
output done, chout, rst;
reg done, chout, rst;
reg [1:0] state;
parameter Initial = 2'b00, Count = 2'b01, Done = 2'b10, Change = 2'b11;
/* State register (synchronous reset) */
always @(posedge clk or posedge reset)
begin
  if (reset)
    state <= Initial;
  else
    case (state)
    Initial:
      state <= Count;
    Count:
      if (End && !ch)  state <= Done;
      else if (End && ch ) state <= Change;
      else state <= Count;
    Done:
      state <= Initial;
    Change:
      state <= Initial;
  endcase
end
/* Output logic */
always @(*) begin
  rst = (state == Initial);
  done = (state == Done || state == Change);
  chout = (state == Change);
end
endmodule
//-------------------------------
module fsm_table_cc( din, dout,p );
input [2:0] din; output [2:0] dout; input p;
reg [2:0] dout;
parameter MM = 3'b000, M = 3'b001,P = 3'b010,PP = 3'b011;
parameter M3 = 3'b100, M2 = 3'b101,P2 = 3'b110,P3 = 3'b111;
always @ (*)
begin
  case(din)
    M3: if (p) dout = M2; else dout = M3;
    M2: if (p) dout = MM; else dout = M3;
    MM: if (p) dout = M; else dout = M2;
    M: if (p) dout = P; else dout = MM;
    P: if (p) dout = PP; else dout = M;
    PP: if (p) dout = P2; else dout = P;
    P2: if (p) dout = P3; else dout = PP;
    P3: if (p) dout = P3; else dout = P2;
  endcase
end
endmodule
```