Probleme:

1) Sa se proiecteze un automat cu stari finite Moore ce analizeaza un sir de caractere si accepta doar combinatia MURMUR. Alfabetul utilizat este: M R U

```
`timescale 1ns / 1ps
module testp1;

	localparam M=2'd0;
	localparam R=2'd1;
	localparam U=2'd2;

	// Inputs
	reg clk;
	reg clr;
	reg [1:0] din;

	// Outputs
	wire valid;

	// Instantiate the Unit Under Test (UUT)
	problema1 uut (.clk(clk), .clr(clr), .din(din), .valid(valid));

	always #5 clk <= ~clk;

	initial begin
		// Initialize Inputs
		clk = 0;
		clr = 1;
		din = M;
		#10;
		clr = 0;
		#10;
		din = U;
		#10;
		din = R;
		#10;
		din = M;
		#10;
		din = U;
		#10;
		din = R;
		#30;
		$stop;
	end

endmodule
```

2) Sa se proiecteze un automat cu stari finite Moore ce analizeaza un sir de caractere si accepta combinatia FROG cu orice prefix. Alfabetul utilizat este: F G O R

```
`timescale 1ns / 1ps
module testp2;

	localparam F=2'd0;
	localparam G=2'd1;
	localparam O=2'd2;
	localparam R=2'd3;

	// Inputs
	reg clk;
	reg clr;
	reg [1:0] din;

	// Outputs
	wire valid;

	// Instantiate the Unit Under Test (UUT)
	problema2 uut (.clk(clk), .clr(clr), .din(din), .valid(valid));

	always #5 clk <= ~clk;

	initial begin
		// Initialize Inputs
		clk = 0;
		clr = 1;
		din = F;
		#10;
		clr = 0;
		#10;
		din = R;
		#10;
		din = F;
		#10;
		din = R;
		#10;
		din = O;
		#10;
		din = G;
		#30;
		$stop;
	end

endmodule
```

3) Sa se proiecteze un automat cu stari finite Moore ce analizeaza un sir de caractere si accepta combinatiile CAT sau DOG oriunde. Alfabetul utilizat este: A C D G O T

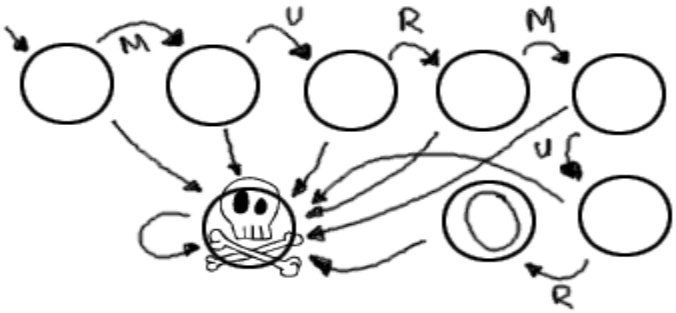
```
`timescale 1ns / 1ps
module testp3;

	localparam A=3'd0;
	localparam C=3'd1;
	localparam D=3'd2;
	localparam G=3'd3;
	localparam O=3'd4;
	localparam T=3'd5;

	// Inputs
	reg clk;
	reg clr;
	reg [2:0] din;

	// Outputs
	wire valid;

	// Instantiate the Unit Under Test (UUT)
	problema3 uut (.clk(clk), .clr(clr), .din(din), .valid(valid));

	always #5 clk <= ~clk;

	initial begin
		// Initialize Inputs
		clk = 0;
		clr = 1;
		din = D;
		#10;
		clr = 0;
		#10;
		din = C;
		#10;
		din = A;
		#10;
		din = D;
		#10;
		din = O;
		#10;
		din = G;
		#30;
		$stop;
	end

endmodule
```

Se cere:

a) Diagram de stari
b) Descrierea in verilog a automatului
c) Formele de unda obtinute cu ajutorul testului descris de codul verilog

Rezolvari:

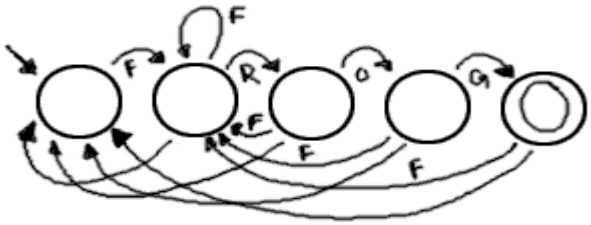
1)




```
`timescale 1ns / 1ps
module problema1(
   input clk,
   input clr,
   input [1:0] din,
   output valid
   );

            localparam M=2'd0;
            localparam R=2'd1;
            localparam U=2'd2;


            reg [2:0] state;

            assign valid = (state==6);

            always@(posedge clk or posedge clr)
                  if (clr) state <= 0;
                  else case (state)
                        3'd0 : if (din==M) state <= 1;
                                          else state <= 7;
                        3'd1 : if (din==U) state <= 2;
                                          else state <= 7;
                        3'd2 : if (din==R) state <= 3;
                                          else state <= 7;
                        3'd3 : if (din==M) state <= 4;
                                          else state <= 7;
                        3'd4 : if (din==U) state <= 5;
                                          else state <= 7;
```

```
                3'd5 : if (din==R) state <= 6;
                                   else state <= 7;
                3'd6 : state <= 7;
                3'd7 : state <= 7;
        endcase

endmodule
```

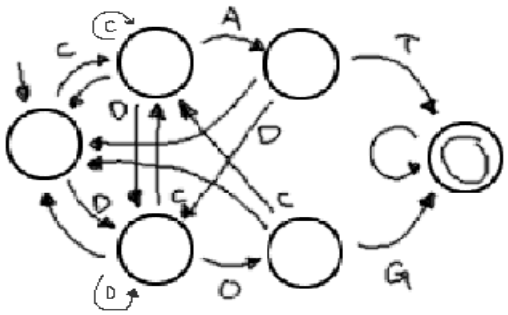
2)




```
`timescale 1ns / 1ps
module problema2(
    input clk,
    input clr,
    input [1:0] din,
    output valid
    );

        localparam F=2'd0;
        localparam G=2'd1;
        localparam O=2'd2;
        localparam R=2'd3;

        reg [2:0] state;

        assign valid = (state==4);

        always@(posedge clk or posedge clr)
            if (clr) state <= 0;
            else case (state)
                3'd0 : if (din==F) state <= 1;
                                   else state <= 0;
                3'd1 : if (din==R) state <= 2;
                                   else begin
                                    if (din==F) state <= 1;
                                    else state <= 0;
                                   end
```

```
                3'd2 : if (din==O) state <= 3;
                                   else begin
                                    if (din==F) state <= 1;
                                    else state <= 0;
                                   end
                3'd3 : if (din==G) state <= 4;
                                   else begin
                                    if (din==F) state <= 1;
                                    else state <= 0;
                                   end
                3'd4 : if (din==F) state <= 1;
                                   else state <= 0;
                default: state <= 0;
        endcase

endmodule
```

3)




```
`timescale 1ns / 1ps
module problema3(
   input clk,
   input clr,
   input [2:0] din,
   output valid
   );

            localparam A=3'd0;
            localparam C=3'd1;
            localparam D=3'd2;
            localparam G=3'd3;
            localparam O=3'd4;
            localparam T=3'd5;
```

```
reg [2:0] state;

assign valid = (state==5);

always@(posedge clk or posedge clr)
        if (clr) state <= 0;
        else case (state)
                3'd0 : if (din==C) state <= 1;
                                   else begin
                                    if (din==D) state <= 2;
                                    else state <= 0;
                                   end
                3'd1 : if (din==A) state <= 3;
                                   else begin
                                    if (din==D) state <= 2;
                                    else begin
                                         if (din==C) state <= 1;
                                         else state <= 0;
                                        end
                                   end
                3'd2 : if (din==O) state <= 4;
                                   else begin
                                    if (din==C) state <= 1;
                                    else begin
                                         if (din==D) state <= 2;
                                         else state <= 0;
                                        end
                                   end
                3'd3 : if (din==T) state <= 5;
                                   else begin
                                    if (din==D) state <= 2;
                                    else state <= 0;
                                   end
                3'd4 : if (din==G) state <= 5;
                                   else begin
                                    if (din==C) state <= 1;
                                    else state <= 0;
                                   end
                3'd5 : state <= 5;
                default: state <= 0;
        endcase

endmodule
```