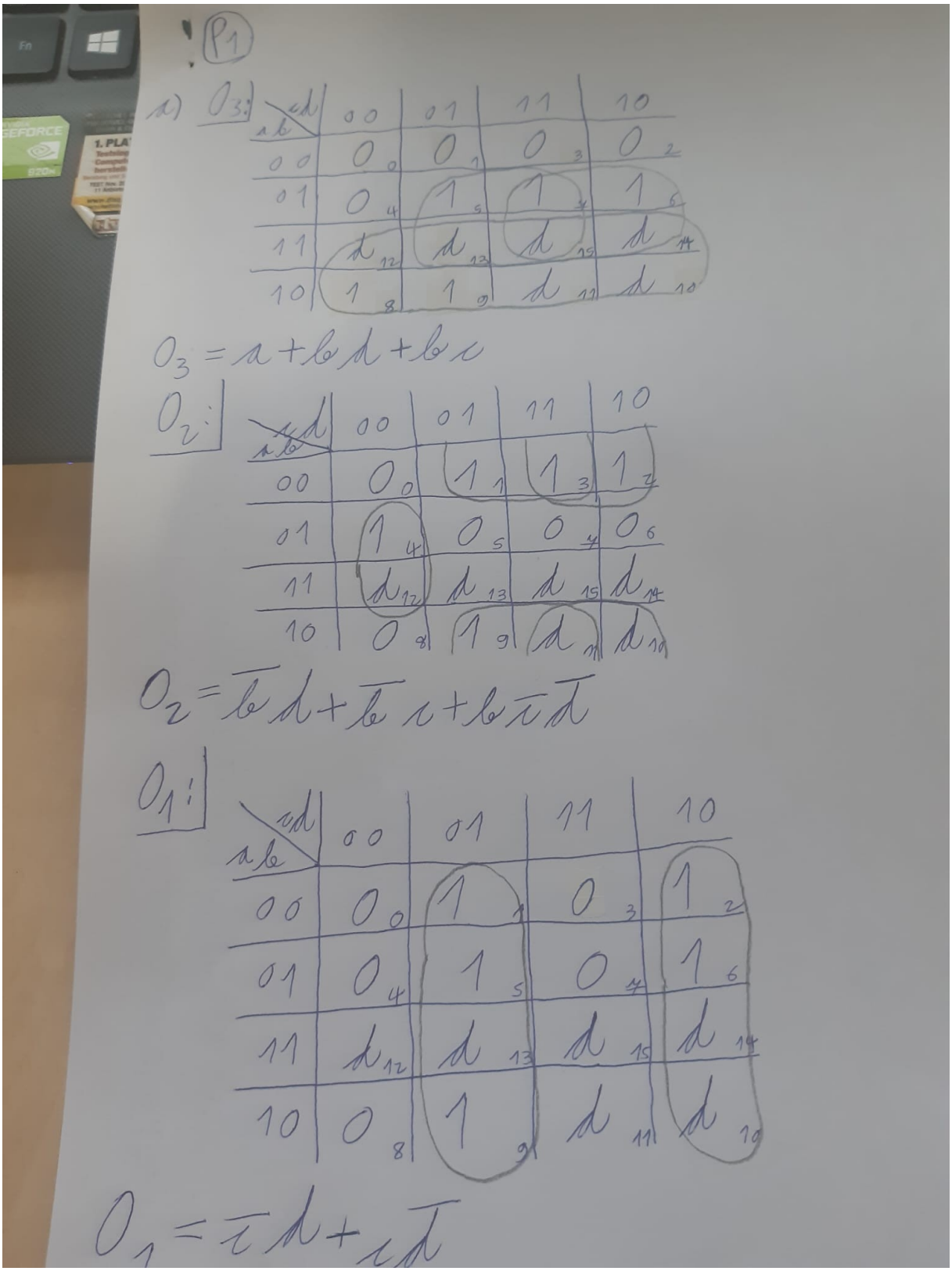
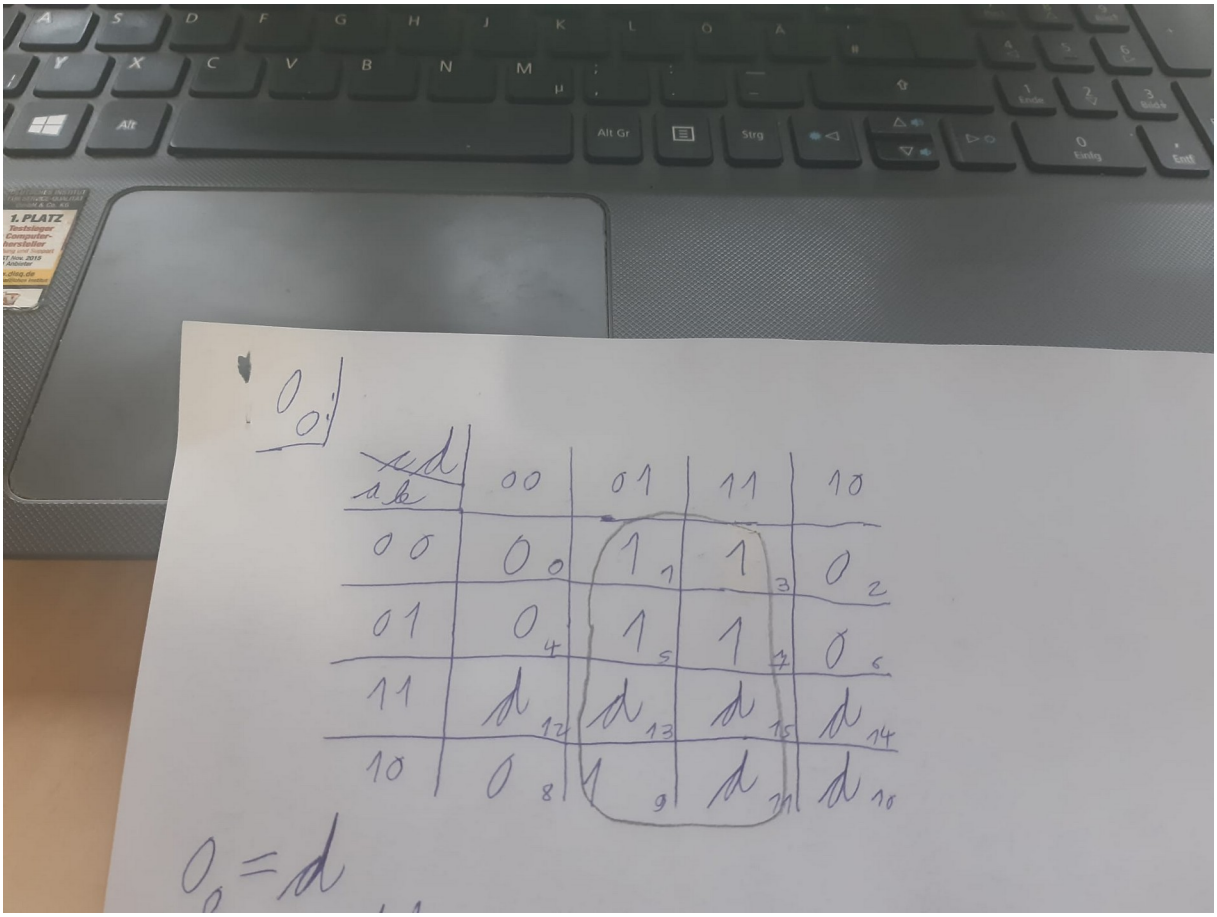
Test Recapitulativ

Farcaș Adrian Tiberiu





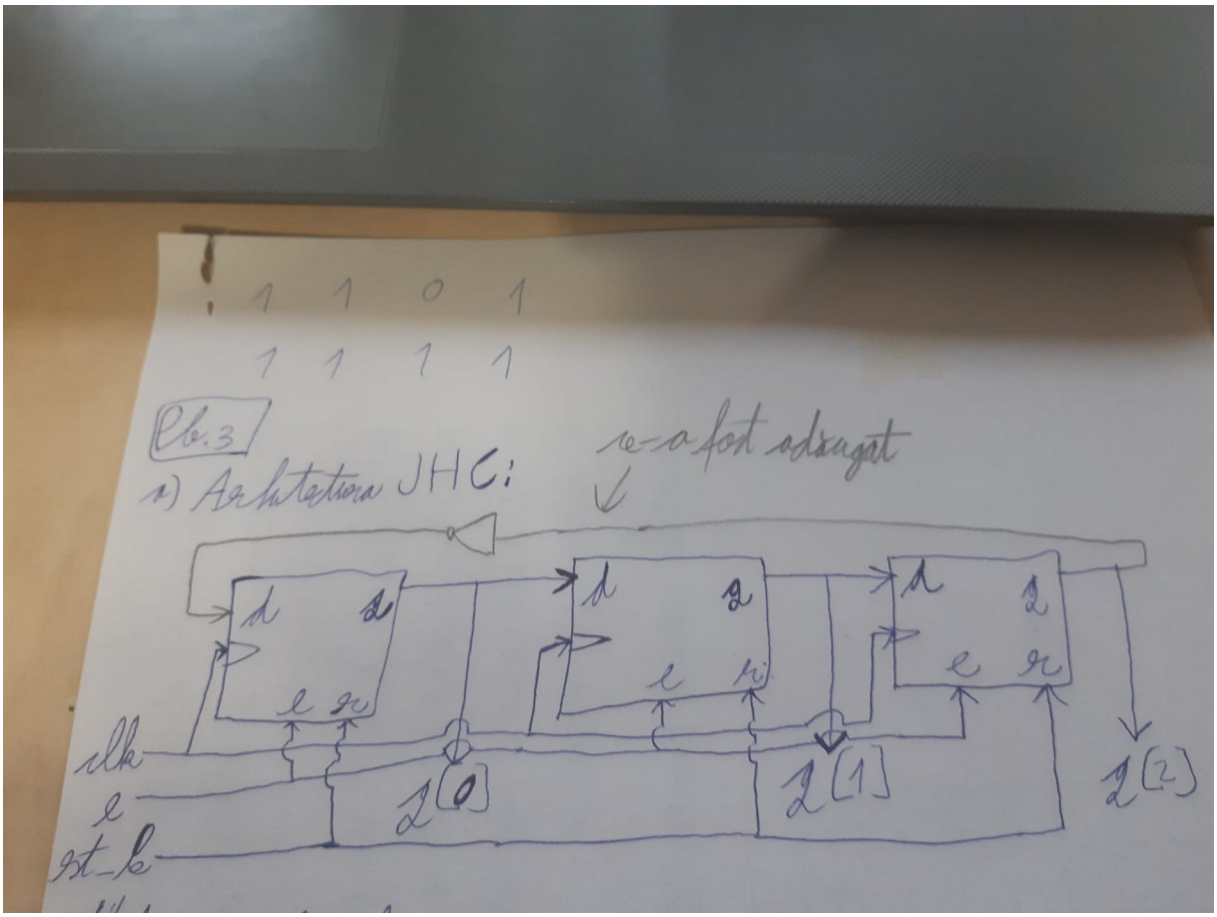
Programele de la problema 1:

```
module minimization
(
      input [3:0] i,
      output [3:0] o
);

      assign o[0] = i[3];
      assign o[1] = (~i[2] & i[3]) | (i[2] & ~i[3]);
      assign o[2] = (~i[1] & i[3]) | (~i[1] & i[2]) | (i[1] & i[2] & ~i[3]);
      assign o[3] = i[0] | (i[1] & i[3]) | (i[1] & i[2]);

endmodule
```

```
module minimization_tb;

  reg [3:0] i_tb;
  wire [3:0] o_tb;

  minimization S11
  (
        .i(i_tb),
        .o(o_tb)
  );

  initial begin
        i_tb = 0;
        #10;
  end

  always begin
        #10 i_tb = i_tb + 1;
  end

endmodule
```

Programele de la problema 2:

```
1   module divide_by_v (
2     input [3:0] i,
3     output reg [3:0] o
4     );
5
6     always @(i) begin
7         o = i >> 2;
8     end
9   endmodule
```



Programele de la problema 4:

```
module bistabilD(

    input d, clk, rst, set,

    output reg q
        );



always @ (negedge rst, negedge set, posedge clk) begin

  if(!set)

    q<=1;

  if(!rst)

    q<=0;

  else

    q<=d;

  end

endmodule
```

```
module JHC(
  input clk, e, r,
  output [2:0] q
);

bistabilD b1 (.d(~q[2]), .clk(clk), .rst(r), .set(e), .q(q[0]));
bistabilD b2 (.d(q[0]), .clk(clk), .rst(r), .set(e), .q(q[1]));
bistabilD b3 (.d(q[1]), .clk(clk), .rst(r), .set(e), .q(q[2]));

endmodule
```

Secvential:

```
5       output reg [2:0] q
6     );
7
8   reg [2:0] aux = 3'b100;
9
10  initial begin
11      q <= 3'b000;
12  end
13
14  always @(posedge clk, negedge rst) begin
15    if (!rst)
16      q <= 0;
17    else (en) begin
18      q <= q^aux;
19      aux <= aux >> 1;
20      if (aux == 0)
21        aux <= 3'b100;
22    end
23  end
24  endmodule
```

Programe de la problema 4:

```
module dff_ar(
        input d, clk, rst, set,
        output reg q
        );

        always @ (negedge rst, negedge set, posedge clk) begin
        if(!set)
        q<=1;
        if(!rst)
        q<=0;
        else
        q<=d;
        end

endmodule
```

```
module LSFR(
        input clk, rst_b,
        output [4:0]q
           );

         dff_ar d1(.d(q[4]), .clk(clk), .rst(1), .set(rst_b), .q(q[0]));
         dff_ar d2(.d(q[0]^q[4]), .clk(clk), .rst(1), .set(rst_b), .q(q[1]));
         dff_ar d3(.d(q[1]), .clk(clk), .rst(1), .set(rst_b), .q(q[2]));
         dff_ar d4(.d(q[2]), .clk(clk), .rst(1), .set(rst_b), .q(q[3]));
         dff_ar d5(.d(q[3]^q[4]), .clk(clk), .rst(1), .set(rst_b), .q(q[4]));

endmodule
```

```
module LFSR_tb;
reg clk_tb;
reg rst_b_tb;
wire [4:0]q_tb;

LFSR uut(.clk(clk_tb), .rst_b(rst_b_tb), ,q(q_tb));

initial begin
        rst_b_tb=1'b1;
        clk_tb=1'b0;
end

always begin
        clk_tb=clk_tb+1'b1;
end
```