Laborator 4

Olaru Gabriel Iulian
-324CC-

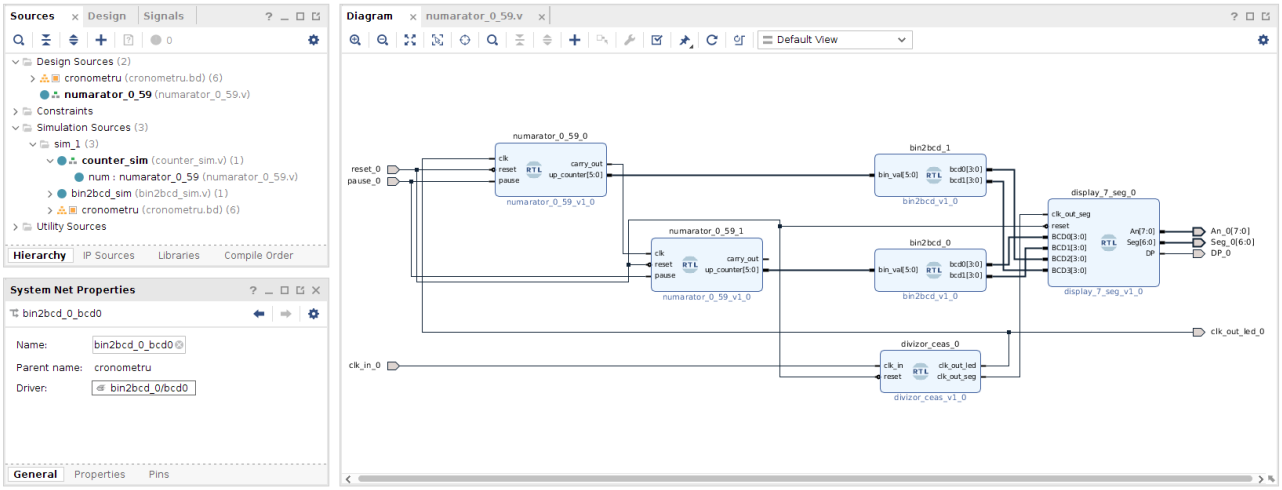
I. Cronometru

Sa se realizeze schema bloc din platforma de laborator conform figurii 3.3 pentru intregul circuit, adaugandu-se modulele pentru afisare 7 segmente (figura 3.8) si divizorul de ceas (figura 3.2). Circuitul NU va fi simulat in intregime pentru ca divizorul de ceas va altera semnalele. El este necesar doar in vederea download-arii in placa FPGA.

·circuitul obtinut in Block Diagram

· fisierul de constrangeri in care sunt asociati pinii potriviti (in lucrare se explica modalitatea de modificare a pinilor conform segmentelor si anozilor la afisarea pe 7 segmente).

I.I. Circuitul Block Diagram



I.II. Constrangeri

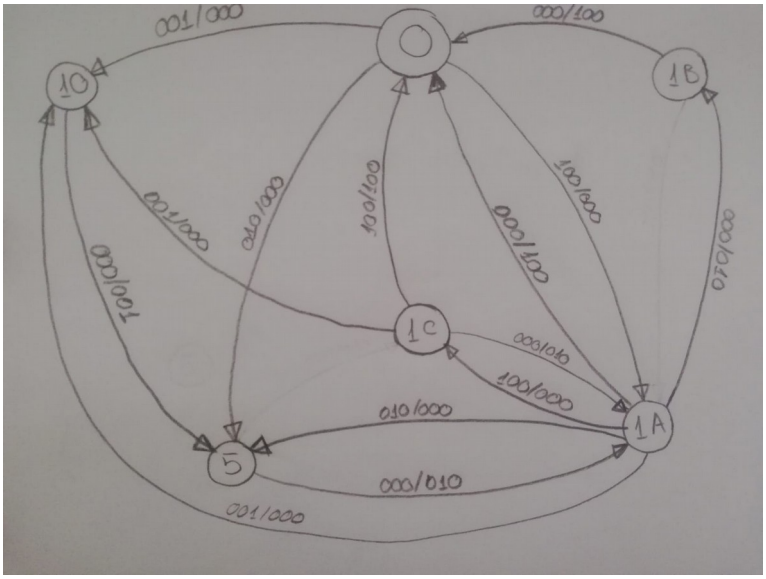
```
## Clock signal
set_property -dict { PACKAGE_PIN E3    IOSTANDARD LVCMOS33 } [get_ports { clk_in_0 }]; #IO_L12P_T1_MRCC_35 Sch=clk100mhz
create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports {clk_in_0}];


##Switches

set_property -dict { PACKAGE_PIN J15   IOSTANDARD LVCMOS33 } [get_ports { pause_0 }]; #IO_L24N_T3_RS0_15 Sch=sw[0]
set_property -dict { PACKAGE_PIN L16   IOSTANDARD LVCMOS33 } [get_ports { reset_0 }]; #IO_L3N_T0_DQS_EMCCLK_14 Sch=sw[1]
set_property -dict { PACKAGE_PIN M13   IOSTANDARD LVCMOS33 } [get_ports { clk_out_led+0 }]; #IO_L6N_T0_D08_VREF_14 Sch=sw[2]
#set_property -dict { PACKAGE_PIN R15   IOSTANDARD LVCMOS33 } [get_ports { SW[3] }]; #IO_L13N_T2_MRCC_14 Sch=sw[3]
#set_property -dict { PACKAGE_PIN R17   IOSTANDARD LVCMOS33 } [get_ports { SW[4] }]; #IO_L12N_T1_MRCC_14 Sch=sw[4]
#set_property -dict { PACKAGE_PIN T18   IOSTANDARD LVCMOS33 } [get_ports { SW[5] }]; #IO_L7N_T1_D10_14 Sch=sw[5]
#set_property -dict { PACKAGE_PIN U18   IOSTANDARD LVCMOS33 } [get_ports { SW[6] }]; #IO_L17N_T2_A13_D29_14 Sch=sw[6]
#set_property -dict { PACKAGE_PIN R13   IOSTANDARD LVCMOS33 } [get_ports { SW[7] }]; #IO_L5N_T0_D07_14 Sch=sw[7]
#set_property -dict { PACKAGE_PIN T8    IOSTANDARD LVCMOS18 } [get_ports { SW[8] }]; #IO_L24N_T3_34 Sch=sw[8]
#set_property -dict { PACKAGE_PIN U8    IOSTANDARD LVCMOS18 } [get_ports { SW[9] }]; #IO_25_34 Sch=sw[9]
#set_property -dict { PACKAGE_PIN R16   IOSTANDARD LVCMOS33 } [get_ports { SW[10] }]; #IO_L15P_T2_DQS_RDWR_B_14 Sch=sw[10]
#set_property -dict { PACKAGE_PIN T13   IOSTANDARD LVCMOS33 } [get_ports { SW[11] }]; #IO_L23P_T3_A03_D19_14 Sch=sw[11]
#set_property -dict { PACKAGE_PIN H6    IOSTANDARD LVCMOS33 } [get_ports { SW[12] }]; #IO_L24P_T3_35 Sch=sw[12]
#set_property -dict { PACKAGE_PIN U12   IOSTANDARD LVCMOS33 } [get_ports { SW[13] }]; #IO_L20P_T3_A08_D24_14 Sch=sw[13]
#set_property -dict { PACKAGE_PIN U11   IOSTANDARD LVCMOS33 } [get_ports { SW[14] }]; #IO_L19N_T3_A09_D25_VREF_14 Sch=sw[14]
#set_property -dict { PACKAGE_PIN V10   IOSTANDARD LVCMOS33 } [get_ports { SW[15] }]; #IO_L21P_T3_DQS_14 Sch=sw[15]


## LEDs

set_property -dict { PACKAGE_PIN H17   IOSTANDARD LVCMOS33 } [get_ports { clk_out_led[0] }]; #IO_L18P_T2_A24_15 Sch=led[0]
#set_property -dict { PACKAGE_PIN K15   IOSTANDARD LVCMOS33 } [get_ports { LED[1] }]; #IO_L24P_T3_RS1_15 Sch=led[1]
#set_property -dict { PACKAGE_PIN J13   IOSTANDARD LVCMOS33 } [get_ports { LED[2] }]; #IO_L17N_T2_A25_15 Sch=led[2]
#set_property -dict { PACKAGE_PIN N14   IOSTANDARD LVCMOS33 } [get_ports { LED[3] }]; #IO_L8P_T1_D11_14 Sch=led[3]
```

```
##7 segment display

set_property -dict { PACKAGE_PIN T10   IOSTANDARD LVCMOS33 } [get_ports { Seg[6] }]; #IO_L24N_T3_A00_D16_14 Sch=ca
set_property -dict { PACKAGE_PIN R10   IOSTANDARD LVCMOS33 } [get_ports { Seg[5] }]; #IO_25_14 Sch=cb
set_property -dict { PACKAGE_PIN K16   IOSTANDARD LVCMOS33 } [get_ports { Seg[4] }]; #IO_25_15 Sch=cc
set_property -dict { PACKAGE_PIN K13   IOSTANDARD LVCMOS33 } [get_ports { Seg[3] }]; #IO_L17P_T2_A26_15 Sch=cd
set_property -dict { PACKAGE_PIN P15   IOSTANDARD LVCMOS33 } [get_ports { Seg[2] }]; #IO_L13P_T2_MRCC_14 Sch=ce
set_property -dict { PACKAGE_PIN T11   IOSTANDARD LVCMOS33 } [get_ports { Seg[1] }]; #IO_L19P_T3_A10_D26_14 Sch=cf
set_property -dict { PACKAGE_PIN L18   IOSTANDARD LVCMOS33 } [get_ports { Seg[0] }]; #IO_L4P_T0_D04_14 Sch=cg

set_property -dict { PACKAGE_PIN H15   IOSTANDARD LVCMOS33 } [get_ports { DP }]; #IO_L19N_T3_A21_VREF_15 Sch=dp

set_property -dict { PACKAGE_PIN J17   IOSTANDARD LVCMOS33 } [get_ports { An[2] }]; #IO_L23P_T3_FOE_B_15 Sch=an[0]
set_property -dict { PACKAGE_PIN J18   IOSTANDARD LVCMOS33 } [get_ports { An[3] }]; #IO_L23N_T3_FWE_B_15 Sch=an[1]
set_property -dict { PACKAGE_PIN T9    IOSTANDARD LVCMOS33 } [get_ports { An[0] }]; #IO_L24P_T3_A01_D17_14 Sch=an[2]
set_property -dict { PACKAGE_PIN J14   IOSTANDARD LVCMOS33 } [get_ports { An[1] }]; #IO_L19P_T3_A22_15 Sch=an[3]
set_property -dict { PACKAGE_PIN P14   IOSTANDARD LVCMOS33 } [get_ports { An[4] }]; #IO_L8N_T1_D12_14 Sch=an[4]
set_property -dict { PACKAGE_PIN T14   IOSTANDARD LVCMOS33 } [get_ports { An[5] }]; #IO_L14P_T2_SRCC_14 Sch=an[5]
set_property -dict { PACKAGE_PIN K2    IOSTANDARD LVCMOS33 } [get_ports { An[6] }]; #IO_L23P_T3_35 Sch=an[6]
set_property -dict { PACKAGE_PIN U13   IOSTANDARD LVCMOS33 } [get_ports { An[7] }]; #IO_L23N_T3_A02_D18_14 Sch=an[7]
```

II. Automat de bauturi

Se doreste proiectarea unui automat pentru bauturi r acoritoare. Pentru a simplifica problema, se considera ca se poate elibera un singur tip de bautura racoritoare. Aceasta costa 3 RON. Automatul accepta bancnote de 1, 5 , respectiv 10 RON si elibereaza rest daca s-a introdus o suma mai mare decat pretul bauturii. Se presupune ca exista un mecanism pentru sortarea banilorsi care emite treisemnale, cate unul pentru fiecare tip de bancnota. Semnalele determina tranzit, ia automatului dintr-o stare în alta.

II.I. Schema Aparatului si Codificarea Starilor



S0=000 -initial
S1=001 -1ron A
S2=010 -1ron B
S3=011 -1ron C
S4=100 -5ron
S5=101 -10ron

II.II. Implementare

```
//S0=000 -initial
//S1=001 -1ron A
//S2=010 -1ron B
//S3=011 -1ron C
//S4=100 -5ron
//S5=101 -10ron
module cafea(
    input clk, reset, I, V, X,
    output B, R1, R5
    );

    reg B, R1, R5;
    reg[1:0] state, nextState;

    always @ (posedge clk)
        if (reset)
            state <= 0;
        else
            state <= nextState;

        always @ (state & I & V & X)
            case({state, I, V, X})
                6'b001000: begin
                            if (state == 3'b000)
                                nextState = 3'b101;B=0;R1=0;R5=0;
                            if (state == 3'b011)
                                nextState = 3'b101;B=0;R1=0;R5=0;
                            if (state == 3'b011)
                                nextState = 3'b101;B=0;R1=0;R5=0;
                end
                6'b010000: begin
                            if (state == 3'b000)
                                nextState = 3'b100;B=0;R1=0;R5=0;
                            if (state == 3'b001)
                                nextState = 3'b100;B=0;R1=0;R5=0;
                           end
                6'b100000: begin
                            if (state == 3'b000)
                                nextState = 3'b001;B=0;R1=0;R5=0;
                            if (state == 3'b001)
                                nextState = 3'b011;B=0;R1=0;R5=0;
                           end
                6'b00100: begin
                            if (state == 3'b010)
                                nextState = 3'b000;B=1;R1=0;R5=0;
                            if (state == 3'b001)
                                nextState = 3'b000;B=1;R1=0;R5=0;
                           end
                6'b00001: begin
                                nextState = 3'b100;B=0;R1=0;R5=1;
                           end
                6'b00010: begin
                             if (state == 3'b100)
                                nextState = 3'b001;B=0;R1=1;R5=0;
                            if (state == 3'b011)
                                nextState = 3'b001;B=0;R1=1;R5=0;
                            if (state == 3'b001)
                                nextState = 3'b010;B=0;R1=1;R5=0;
                            if (state == 3'b010)
                                nextState = 3'b001;B=0;R1=1;R5=0;
                           end
                default:   begin
                           state=3'b000;B=0;R1=0;R5=0;
                           end
            endcase
endmodule
```

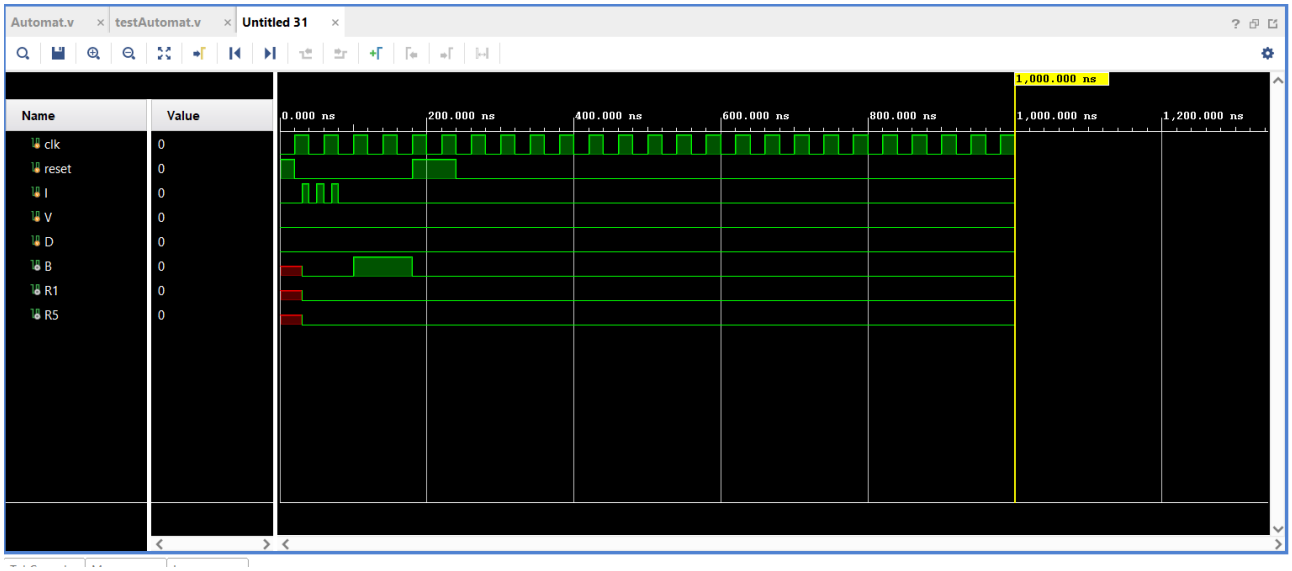
II.III. Simulare

Simulam, pe rand, pentru 1leu, 5lei si 10lei.

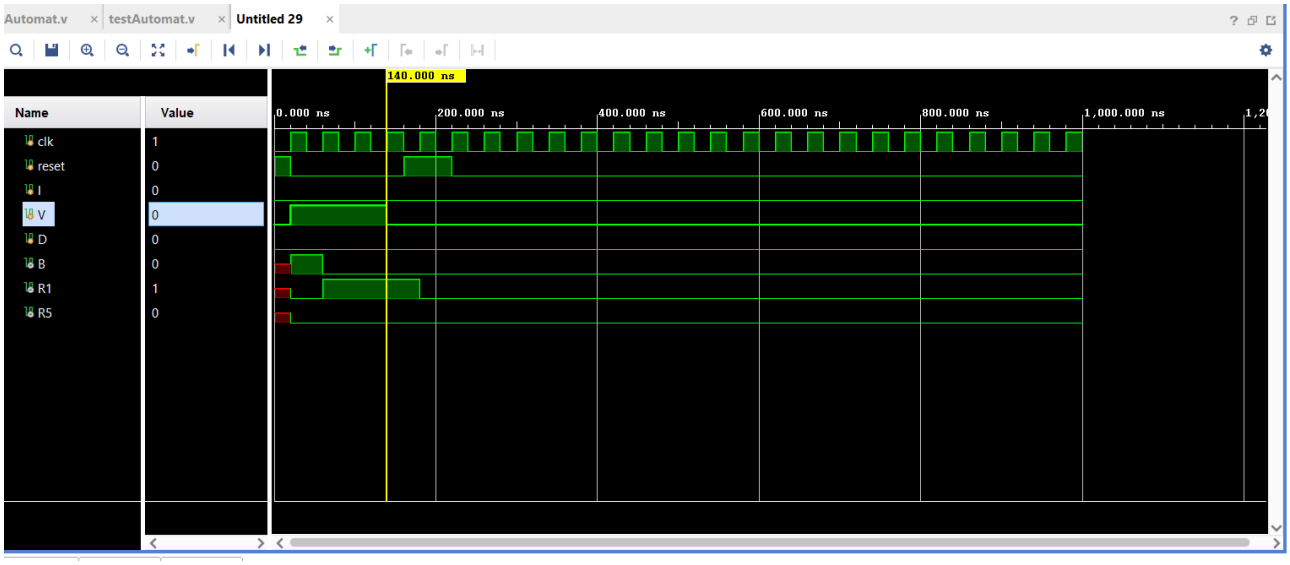
```
module cafea_sim(
    );
    reg clk, reset, I, V, X;
    wire B, R1, R5;
    cafea a(clk, reset, I, V , X, R1, R5, B);
    initial
    begin
        clk = 0;
        I = 0; V = 0; X = 0;
        reset = 1;
        #20 reset = 0;
        #20 I = 1; V = 0; X = 0;
        #20 I = 0; V = 1; X = 0;
        #20 I = 0; V = 0; X = 1;
    end
    initial
    begin
        forever
        #20 clk = !clk;
    end;
```

II.IV. Rezultatele Simularii

Cazul cu 1 leu.



Cazul cu 5 lei.



Cazul cu 10 lei.