UNIVERSITATEA TEHNICĂ DIN CLUJ-NAPOCA

Facultatea de Electronică, Telecomunicații și Tehnologia Informației

SISTEME CU CIRCUITE INTEGRATE DIGITALE

Proiect automat secvential

Proiect realizat de: Achiriloaei Ioana-Isabela

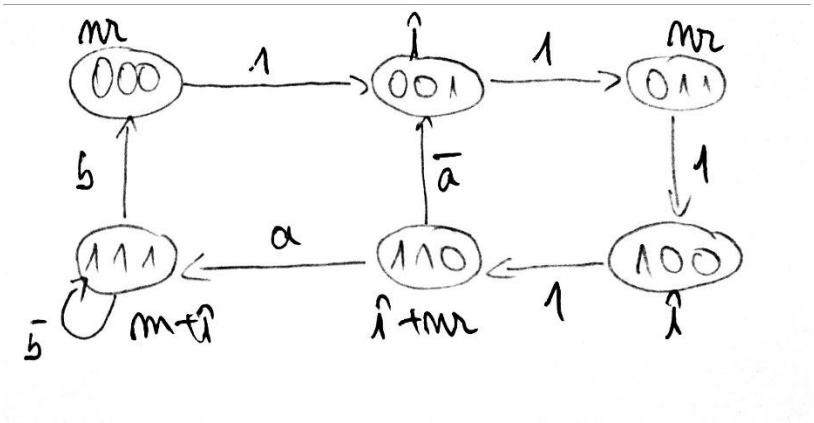
Seria: A

Grupa: 2123

Indrumator: Asistent universitar Ing.Bianca Bota

CERINTE:

Sa se proiecteze un automat secvențial corespunzător cu diagrama de stări din figura de mai jos.



1. Proiectarea circuitului utilizând multiplexoare 2:1 și numărător 74163.

2. Descrierea structurală al automatului.

3. Descrierea comportamentală al automatului.

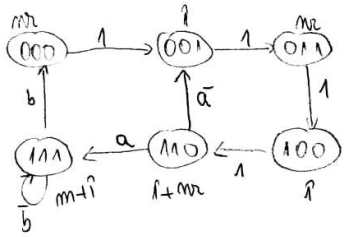
4. Modul de verificare, in care sunt generate semnale de control astfel încât automatul sa parcurgă toate stările posibile din diagramă.

Proiectarea circuitului:

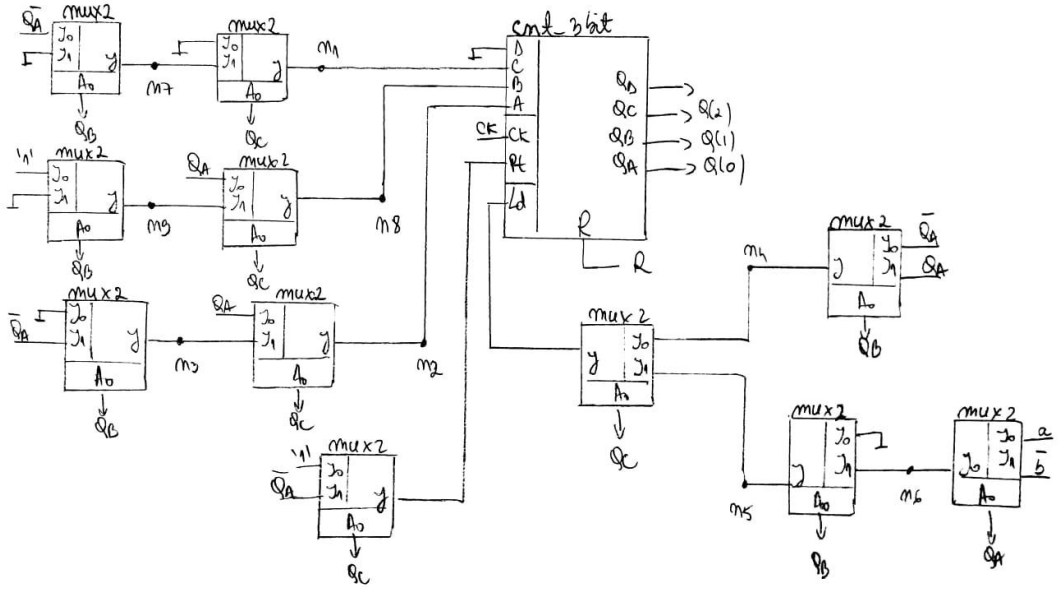
Pentru a proiecta circuitul se va completa un tabel de adevar corespunzator automatului unde trecem valorile starilor urmatoare in funcție de starea curenta și de semnalele de control. Rezultatele sunt trecute in tabelul de mai jos.

QC	QB	QA	ACTIUNE	Ld	Pt	C	B	A
0	0	0	numara	1	1	x	x	x
0	0	1	incarca	0	x	0	1	1
0	1	0	incarca	0	x	0	0	0
0	1	1	numara	1	x	x	x	x
1	0	0	incarca	0	x	1	1	0
1	0	1	incarca	0	x	0	0	0
1	1	0	incarca+numara	a	1	0	0	1
1	1	1	mentine+incarca	$\bar{b}$	0	0	0	0

Cu ajutorul valorilor obtinute in tabelul de mai sus, putem completa schema logica realizata din numaratorul 74163 si cele 11 muxuri 2:1



Q_C	Q_B	Q_A	Ld	Pt	C	B	A
0	0	0	1	1	X	X	X
0	0	1	0	X	0	1	1
0	1	0	0	X	0	0	0
0	1	1	1	1	X	X	X
1	0	0	0	X	1	1	0
1	0	1	0	X	0	0	0
1	1	0	a	1	0	0	1
1	1	1	$\bar{b}$	0	0	0	0



Descrierea structurala al automatului:

cnt_3bit:

```
3   library IEEE;
4   use IEEE.STD_LOGIC_1164.ALL;
5   use IEEE.std_logic_signed.all;
6
7   entity cnt3bit is
8       Port ( ck : in STD_LOGIC;
9              Pt : in STD_LOGIC;
10             Ld : in STD_LOGIC;
11             R : in STD_LOGIC;
12             D : in STD_LOGIC_VECTOR( 2 downto 0);
13             Q : out STD_LOGIC_VECTOR(2 downto 0)
14             );
15
16  end cnt3bit;
17
18  architecture Behavioral of cnt3bit is
19  signal stare : STD_LOGIC_VECTOR(2 downto 0);
20  begin
21  Q <= stare;
22  cnt3bit: process (ck)
23  begin
24     if R = '0' then stare <= "000";
25     elsif rising_edge(ck) then
26         if Ld='0' then
```

```
27              stare <= D;
28          else
29              if Pt='0' then
30                 stare <= stare;
31              else
32                 stare <= stare+1;
33              end if;
34          end if;
35     --end if;
36     end if;
37  end process;
38  end Behavioral;
```

mux 2:1

```
2  library IEEE;
3  use IEEE.STD_LOGIC_1164.ALL;
4
5  entity mux_2 is
6      Port ( A : in STD_LOGIC;
7             I : in STD_LOGIC_VECTOR(0 to 1);
8             Y : out STD_LOGIC);
9  end mux_2;
10
11 architecture Behavioral of mux_2 is
12
13 begin
14 mux2: process (A,I)
15 begin
16     case A is
17         when '0' =>  Y <= I(0);
18         when '1' =>  Y <= I(1);
19         when others => Y <= 'X';
20    end case;
21 end process;
22
23 end Behavioral;
```

inversor:

```
22 library IEEE;
23 use IEEE.STD_LOGIC_1164.ALL;
24
25 entity invs is
26     Port ( x : in STD_LOGIC;
27            f : out STD_LOGIC);
28 end invs;
29
30 architecture Behavioral of invs is
31
32 begin
33 f<= not x;
34
35 end Behavioral;
```

interconectare:

```
4      use IEEE.STD_LOGIC_1164.ALL;
5
6
7      entity interconectare is
8          Port ( ck : in STD_LOGIC;
9                 R : in STD_LOGIC;
10                a : in STD_LOGIC;
11                b : in STD_LOGIC;
12                Q : out STD_LOGIC_VECTOR (2 downto 0));
13     end interconectare;
14
15     architecture Behavioral of interconectare is
16     component cnt3bit is
17         Port ( ck : in STD_LOGIC;
18                Pt : in STD_LOGIC;
19                Ld : in STD_LOGIC;
20                R : in STD_LOGIC;
21                D : in STD_LOGIC_VECTOR( 2 downto 0);
22                Q : out STD_LOGIC_VECTOR(2 downto 0)
23                );
24
25     end component;
26     component mux_2 is
27         Port ( A : in STD_LOGIC;
28                I : in STD_LOGIC_VECTOR(0 to 1);
29                Y : out STD_LOGIC);
```

```
30      end component;
31
32      component invs is
33          Port ( x : in STD_LOGIC;
34                 f : out STD_LOGIC);
35      end component;
36
37      signal Qa,Qb,Qc,Pt,Ld,Qan,Qbn,bn,an,n1,n2,n3,n4,n5,n6,n7,n8,n9 : std_logic;
38      begin
39      Q<=Qc&Qb&Qa;
40      Qan<=not Qa;
41      Qbn<=not Qb;
42      an<=not a;
43      bn<=not b;
44
45      u1: cnt3bit port map(
46                           ck=>ck,
47                           R=>R,
48                           Ld=>Ld,
49                           Pt=>Pt,
50                           D(0)=>n2,
51                           D(1)=>Qbn,
52                           D(2)=>n1,
53                           Q(0)=>Qa,
54                              Q(1)=>Qb,
55                              Q(2)=>Qc
56
57      );
58      u2: mux_2 port map(
59            I(0)=>'0',
60            I(1)=>n7,
61            A=>Qc,
62            Y=>n1
63      );
64      u3: mux_2 port map(
65            I(0)=>Qan,
66            I(1)=>'0',
67            A=>Qb,
68            Y=>n7
69      );
70      u4: mux_2 port map(
71            I(0)=>Qa,
72            I(1)=>n9,
73            A=>Qc,
74            Y=>n8
75      );
76      u5: mux_2 port map(
77            I(0)=>'1',
```

```
        I(1)=>'0',
        A=>Qb,
        Y=>n9
    );

    u6: mux_2 port map(
        I(0)=>Qa,
        I(1)=>n3,
        A=>Qc,
        Y=>n2

    );
    u7 :mux_2 port map(
        I(0)=>'0',
        I(1)=>Qan,
        A=>Qb,
        Y=>n3
    );

    u8:mux_2 port map(
        I(0)=>'1',
        I(1)=>Qan,
        A=>Qc,
        Y=>Pt
    );
    u9:mux_2 port map(
        I(0)=>n4,
        I(1)=>n5,
        A=>Qc,
        Y=>Ld
    );
    u10:mux_2 port map(
        I(0)=>Qan,
        I(1)=>Qa,
        A=>Qb,
        Y=>n4
    );
    u11:mux_2 port map(
        I(0)=>'0',
        I(1)=>n6,
        A=>Qb,
        Y=>n5
    );
    u12:mux_2 port map(
        I(0)=>a,
        I(1)=>bn,
        A=>Qa,
        Y=>n6

    );
    end Behavioral;
```

Descrierea comportamentala al automatului:

Automat_ref are scopul de a descrie comportamentul circuitului, iar la final acest semnal trebuie sa fie identic cu semnalul automatului proiectat.

automat_ref:

```
4   use IEEE.STD_LOGIC_1164.ALL;
5
6
7
8   entity automat_ref is
9       Port ( ck : in STD_LOGIC;
10             R : in STD_LOGIC;
11             a : in STD_LOGIC;
12             b : in STD_LOGIC;
13             Q_ref : out std_logic_vector(2 downto 0));
14  end automat_ref;
15
16  architecture Behavioral of automat_ref is
17  signal current_state, next_state: std_logic_vector(2 downto 0);
18  begin
19  registrare:
20  process (ck)
21  begin
22      if R = '0' then
23          current_state <= "000";
24      elsif rising_edge(ck) then
25              current_state <= next_state;
26          end if;
27      end process;
28  selectie:
29  process (current_state, a, b)
```

```
30      begin
31      case current_state is
32      when "000" => next_state <= "001";
33      when "001" => next_state <= "011";
34      when "011" => next_state <= "100";
35      when "100" => next_state <= "110";
36      when "110" => if a ='1' then
37                    next_state <= "111";
38                    else
39                    next_state <="001";
40                    end if;
41      when "111" => if b='1' then
42                     next_state <= "000";
43                     else
44                     next_state <= "111";
45                     end if;
46
47      when "010" => next_state <= "000";
48      when "101" => next_state <= "000";
49      when others => next_state <= "001";
50
51       end case;
52       end process;
53       Q_ref <= current_state;
54      end Behavioral;
```

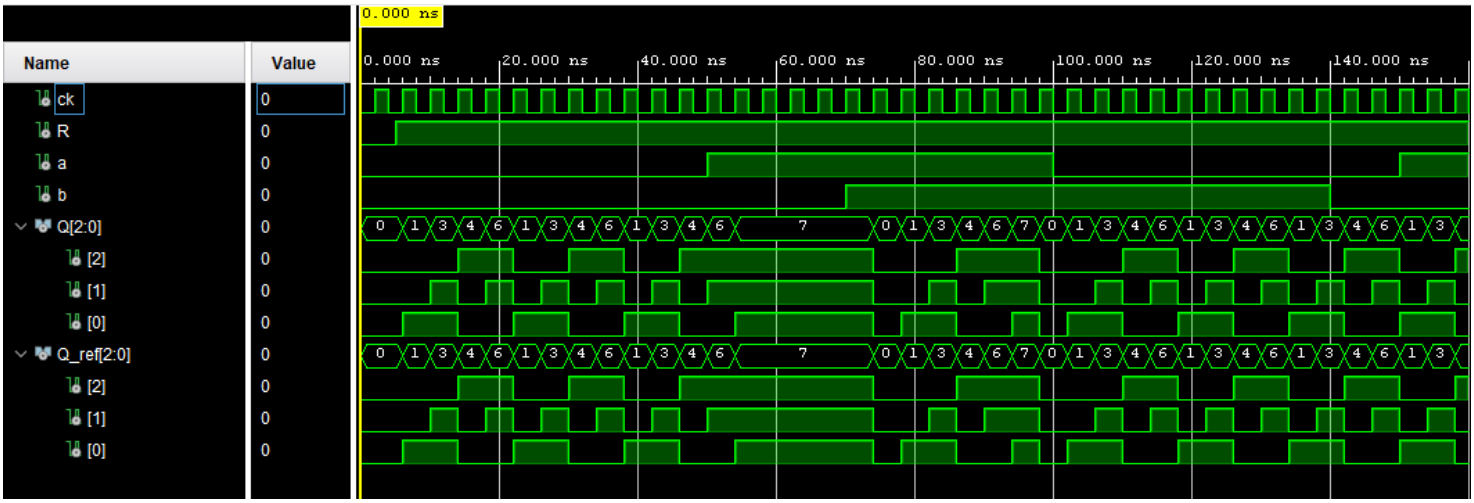
testbench:

```
3       library IEEE;
4       use IEEE.STD_LOGIC_1164.ALL;
5
6
7       entity test_bench is
8       --  Port ( );
9       end test_bench;
10
11      architecture Behavioral of test_bench is
12
13      component interconectare is
14          Port ( ck : in STD_LOGIC;
15                 R : in STD_LOGIC;
16                 a : in STD_LOGIC;
17                 b : in STD_LOGIC;
18                 Q : out STD_LOGIC_VECTOR (2 downto 0));
19      end component;
20
21      component automat_ref is
22          Port ( ck : in STD_LOGIC;
23                 R : in STD_LOGIC;
24                 a : in STD_LOGIC;
25                 b : in STD_LOGIC;
26                 Q_ref : out std_logic_vector(2 downto 0));
```

```
27      end component;
28
29
30      signal ck, R, a, b : std_logic;
31      signal Q, Q_ref : std_logic_vector(2 downto 0);
32
33
34      begin
35
36      gen_R: process
37      begin
38         R <= '0';
39         wait for 5 ns;
40         R <= '1';
41         wait for 10000 ns;
42      end process;
43
44      gen_a: process
45      begin
46
47          a <= '0';
48           wait for 50 ns;
49           a <= '1';
50         wait for 50 ns;
51
52      end process;
53
54      gen_b: process
55      begin
56
57         b <= '0';
58         wait for 70 ns;
59         b <= '1';
60         wait for 70 ns;
61
62      end process;
63
64      gen_ck: process
65      begin
66         ck <= '0';
67         wait for 2ns;
68         ck <= '1';
69         wait for 2ns;
70      end process;
```

```
71
72      automat1: interconectare port map(ck=>ck, a=>a, b=>b, Q=>Q, R => R);
73      automat2: automat_ref port map(ck=>ck, a=>a, b=>b, Q_ref=>Q_ref, R => R);
74
75
76      verificare: process (Q)
77      begin
78      if Q /= q_ref then
79      report "Rezultat diferit";
80      end if;
81      end process;
82
83      end Behavioral;
```

Modul de verificare:



Dupa cum reiese si din diagrama ilustrata mai sus, cele doua automate sunt identice fiind parcurse toate starile.

Cand a=0 si b=0, pornesc din starea “000” dupa care numar pana in “001”, apoi automatul incarca “011” care ulterior va numara starea “100” si va incarca “110”, pe acest front a este in continuare 0, deci va merge in starea ”001” care se va repeta pana cand a devine ‘1’ si va trece in starea “111” unde aceasta stare va fi mentinuta cat timp b este ‘0’, in momentul in care b va deveni ‘1’, atunci automatul va trece in starea “000” unde automatul va parcurge starile din nou.