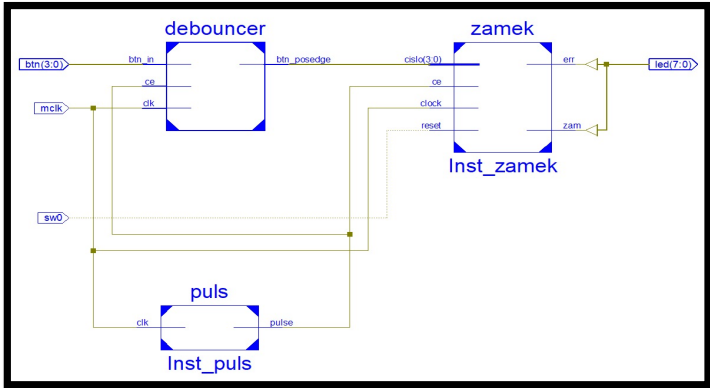


Střední průmyslová škola elektrotechnická Haviřov	Třída: 4.B
	Skupina: 1
	Zpráva číslo: 3
	Den: 22.3.2021
	Jméno učitele: Ing. Božena Ralbovská Jméno: Jakub Lengsfeld Známka:

1. Zadání - měření intenzity světla

Pomocí FPGA obvodu Spartan 3E firmy Xilinx realizujte automat pro řízení chodu **garáže pro 5 automobilů**.

Automat bude signalizovat stavy volno a obsazeno podle příjezdějících a odjíždějících aut. Příjezd a odjezd nastavujte tlačítky, nulování pomocí přepínače, počet aut zobrazujte na sedmi segmentovém displeji. Ošetřete zákmity tlačítek proti chybnému počítání aut.

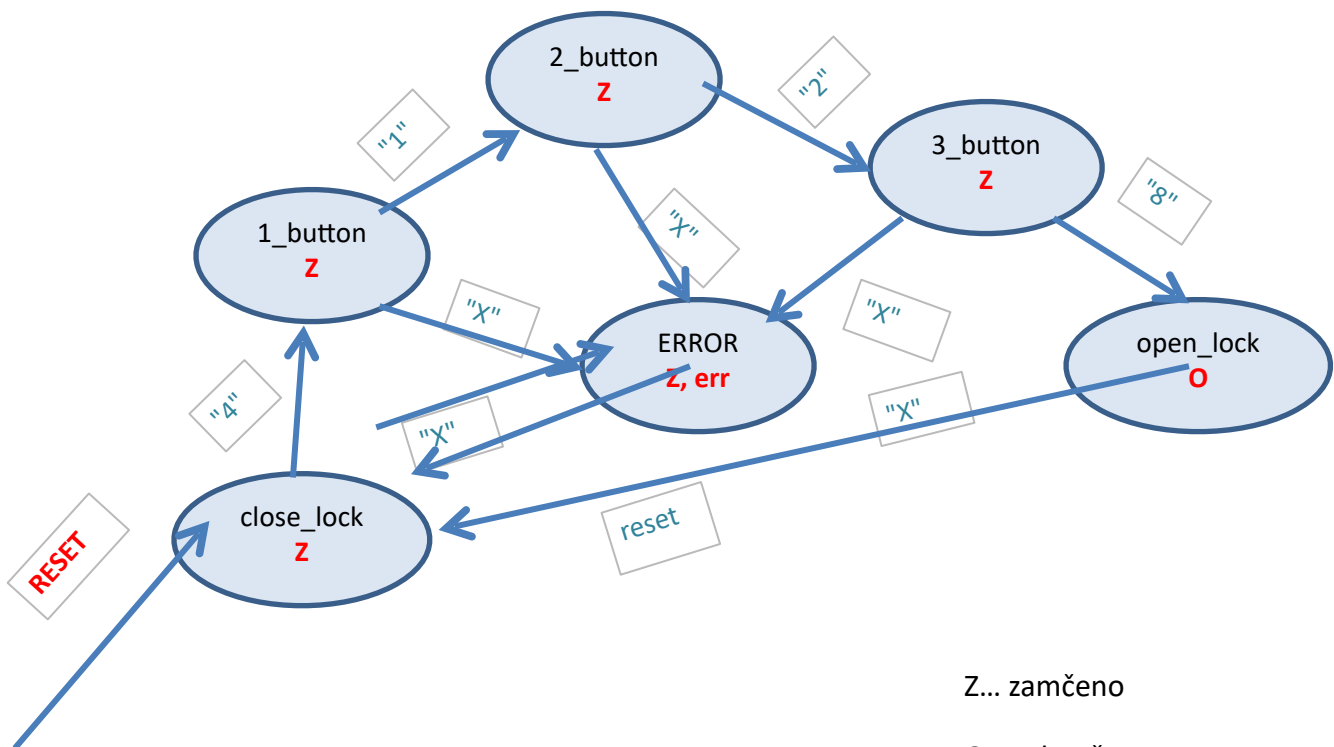
2. Teoretický rozbor

Spartan:

- Je programovatelné hradlové pole, jedná se o typ integrovaného obvodu).
- Jedná se účelově programovatelné zařízení, pracuje pomocí instrukcí uložených v paměti. Přijatá data zpracovává a posílá na výstupy.

VHDL:

- Jedná se o programovací jazyk. Používá se pro návrh simulací digitálních integrovaných obvodů.
- Například programovatelných hradlových polí nebo různých zákaznických obvodů.
- Umožňuje návrh jak logických, tak i sekvenčních struktur.



Z... zamčeno

O... odemčeno

Err... alarm

X... jakákoliv hodnota

Přechodová tabulka:

Stav	4	1	2	8	X
st_close	st1	st_err	st_err	st_err	st_err
st1	st_err	st2	st_err	st_err	st_err
st2	st_err	st_err	st3	st_err	st_err
st3	st_err	st_err	st_err	st_open	st_err
st_open	st_close	st_close	st_close	st_close	st_close
st_err	st_close	st_close	st_close	st_close	st_close

Tabulka vnitřních stavů:

Stav	Kód
st_close	000
st1	001
st2	010
st3	011
st_open	100
st_err	101

Výstupní tabulka:

Stav	Zámek	Error
st_close	0	0
st1	0	0
st2	0	0
st3	0	0
st_open	1	0
st_err	0	1

3.1 Program

```

top_zamek.vhd x zamek_1.vhd x puls.vhd x
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.ALL;
3  use IEEE.STD_LOGIC_unsigned.ALL;
4
5  entity top_zamek is
6  Port ( mclk : in  STD_LOGIC;
7        sw0 : in  STD_LOGIC;
8        btn : in  STD_LOGIC_VECTOR (3 downto 0);
9        led : out STD_LOGIC_VECTOR (7 downto 0));
10 end top_zamek;
11
12 architecture struct of top_zamek is
13 signal cel,err1,reset1 : STD_LOGIC;
14 signal btn_posedge:STD_LOGIC_VECTOR (3 downto 0);
15 signal alarmy:STD_LOGIC_VECTOR (1 downto 0):="00";
16
17 begin
18   Inst_puls: entity work.puls PORT MAP(
19     clk => mclk,
20     pulse => cel
21   );
22
23   Inst_debouncer0: entity work.debouncer PORT MAP(
24     clk => mclk,
25     ce => cel,
26     btn_in => btn(0) ,
27     btn_posedge => btn_posedge(0)
28   );

```

```

23 Inst_debouncer0: entity work.debouncer PORT MAP(
24     clk => mclk,
25     ce => cel,
26     btn_in => btn(0) ,
27     btn_posedge => btn_posedge(0)
28 );
29
30 Inst_debouncer1: entity work.debouncer PORT MAP(
31     clk => mclk,
32     ce => cel,
33     btn_in => btn(1) ,
34     btn_posedge => btn_posedge(1)
35 );
36
37 Inst_debouncer2: entity work.debouncer PORT MAP(
38     clk => mclk,
39     ce => cel,
40     btn_in => btn(2) ,
41     btn_posedge => btn_posedge(2)
42 );
43
44 Inst_debouncer3: entity work.debouncer PORT MAP(
45     clk => mclk,
46     ce => cel,
47     btn_in => btn(3) ,
48     btn_posedge => btn_posedge(3)
49 );
50
51 Inst_zamek_1: entity work.zamek_1 PORT MAP(
52     clock => mclk ,
53     reset => sw0,
54     ce => cel,
55     cislo => btn_posedge(3 downto 0) ,
56     err => err1,
57     zam => led(0)
58 );
59
60 led(7 downto 3) <= "00000";
61 led(1) <= err1;
62 reset1 <= sw0;
63
64 count: process (err1, reset1, alarmy)
65 begin
66     if reset1='1' then led(2) <= '0'; alarmy <= "00";
67     elsif rising_edge(err1) then if alarmy = "10" then led(2) <= '1';
68                                     else alarmy <= alarmy + 1; led(2) <= '0';
69                                     end if;
70     end if;
71 end process count;
72 end struct;

```

```

top_zamek.vhd x zamek_1.vhd x puls.vhd x
1 library IEEE;
2 use IEEE.STD_LOGIC_1164.ALL;
3
4 entity zamek_1 is
5     Port ( clock : in  STD_LOGIC;
6           reset : in  STD_LOGIC;
7           ce : in  STD_LOGIC;
8           cislo : in  STD_LOGIC_VECTOR (3 downto 0);
9           err : out  STD_LOGIC;
10          zam : out  STD_LOGIC);
11 end zamek_1;
12
13 architecture Behavioral of zamek_1 is
14     type state_type is (st_close,st1, st2,st3,st_open,st_err);
15     signal state, next_state : state_type;
16 begin
17     SYNC_PROC: process (clock,reset)
18     begin
19         if (clock'event and clock = '1') then
20             if (reset = '1') then
21                 state <= st_close;
22             else if ce='1' then
23                 state <= next_state;
24             end if;
25         end if;
26     end process;
27
28     OUTPUT_DECODE: process (state)
29     begin
30         case (state) is
31             when st_close => zam<='0';err<='0';
32             when st1 => zam<='0';err<='0';
33             when st2 => zam<='0';err<='0';
34             when st3 => zam<='0';err<='0';
35             when st_open => zam<='1';err<='0';
36             when st_err => zam<='0';err<='1';
37             when others=>null;
38         end case;
39     end process;
40
41 end architecture Behavioral of zamek_1;
42

```

```

NEXT_STATE_DECODE: process (state, cislo)
begin
    next_state <= state;
    case (state) is
        when st_close => if cislo="0100" then next_state <= st1;
                           elsif cislo="0000" then next_state <= st_close;
                           else next_state <= st_err;
                           end if;
        when st1 => if cislo="0001" then next_state <= st2;
                    elsif cislo="0000" then next_state <= st1;
                    else next_state <= st_err;
                    end if;
        when st2 => if cislo="0010" then next_state <= st3;
                    elsif cislo="0000" then next_state <= st2;
                    else next_state <= st_err;
                    end if;
        when st3 => if cislo="1000" then next_state <= st_open;
                    elsif cislo="0000" then next_state <= st3;
                    else next_state <= st_err;
                    end if;
        when st_open => if cislo="0000" then next_state <= st_close;
                        else next_state <= st_err;
                        end if;
        when st_err => if cislo="0000" then next_state <= st_err;
                       else next_state <= st_close;
                       end if;
        when others => null;
    end case;
end process;

end Behavioral;

```

```

top_zamek.vhd x zamek_1.vhd x puls.vhd x
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.ALL;
3  use IEEE.STD_LOGIC_ARITH.ALL;
4
5  entity puls is
6  generic (
7      div_factor : integer := 50000); -- 1 ms
8  port(
9      clk : in std_logic;
10     pulse : out std_logic
11 );
12 end puls;
13
14 Architecture Behavioral of puls is
15     signal counter : integer := 0;
16
17 begin
18     process(clk)
19     begin
20         if(rising_edge(clk)) then
21             if counter = div_factor - 1 then
22                 pulse <= '1';
23                 counter <= 0;
24             else
25                 counter <= counter + 1;
26                 pulse <= '0';
27             end if;
28         end if;
29     end process;
30 end Behavioral;
31

```


3.2 Piny

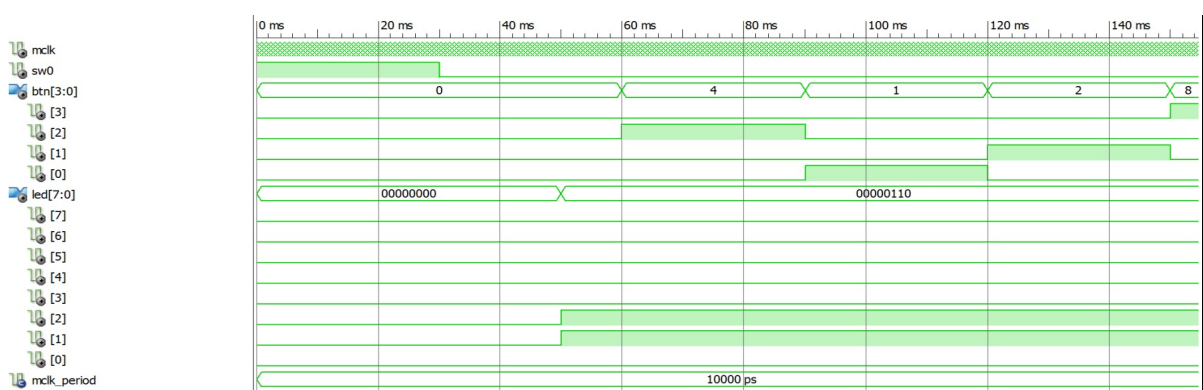
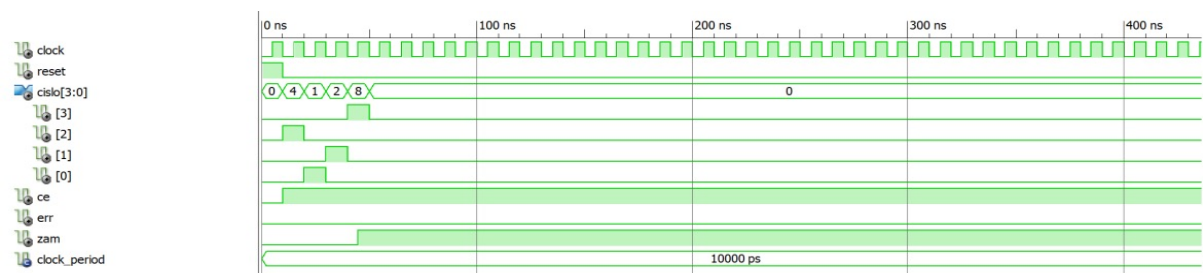
```
# clock pins for Basys2 Board
NET "mclk" LOC = "B8"; # Bank = 0, Signal name = MCLK
NET "mclk" PERIOD = 20ns ;

NET "led<7>" LOC = "G1" ; # Bank = 3, Signal name = LD7
NET "led<6>" LOC = "P4" ; # Bank = 2, Signal name = LD6
NET "led<5>" LOC = "N4" ; # Bank = 2, Signal name = LD5
NET "led<4>" LOC = "N5" ; # Bank = 2, Signal name = LD4
NET "led<3>" LOC = "P6" ; # Bank = 2, Signal name = LD3
NET "led<2>" LOC = "P7" ; # Bank = 3, Signal name = LD2
NET "led<1>" LOC = "M11" ; # Bank = 2, Signal name = LD1
NET "led<0>" LOC = "M5" ; # Bank = 2, Signal name = LD0

NET "sw0" LOC = "P11"; # Bank = 2, Signal name = SW0

NET "btn<3>" LOC = "A7"; # Bank = 1, Signal name = BTN3
NET "btn<2>" LOC = "M4"; # Bank = 0, Signal name = BTN2
NET "btn<1>" LOC = "C11"; # Bank = 2, Signal name = BTN1
NET "btn<0>" LOC = "G12"; # Bank = 0, Signal name = BTN0
```

4. Simulace



5. Zhodnocení

Program funguje podle zadání. Testování proběhlo pomocí simulace.

Všechno hodnoty jsou nastavené permanentně a nelze je měnit.

Jediný problém mi dělalo testování funkčnosti, z důvodu nedostupnosti spartana.