Univerzitet Crne Gore
Elektrotehnički fakultet

Predmet:
Automatizovano projektovanje elektronskih kola i sistema (bivši SEK)

Tema:
Prosti kalkulator na FPGA DE2-70 ploči
Simple calculator on FPGA DE2-70 board

Autori:

Dejan Brajović 15/18
Božidar Anđrović 9/18

Datum i mjesto:
24.11.2018.g Podgorica

Sadržaj:

Sažetak:

Prosti kalkulator na FPGA DE2-70 ploči

Abstract:

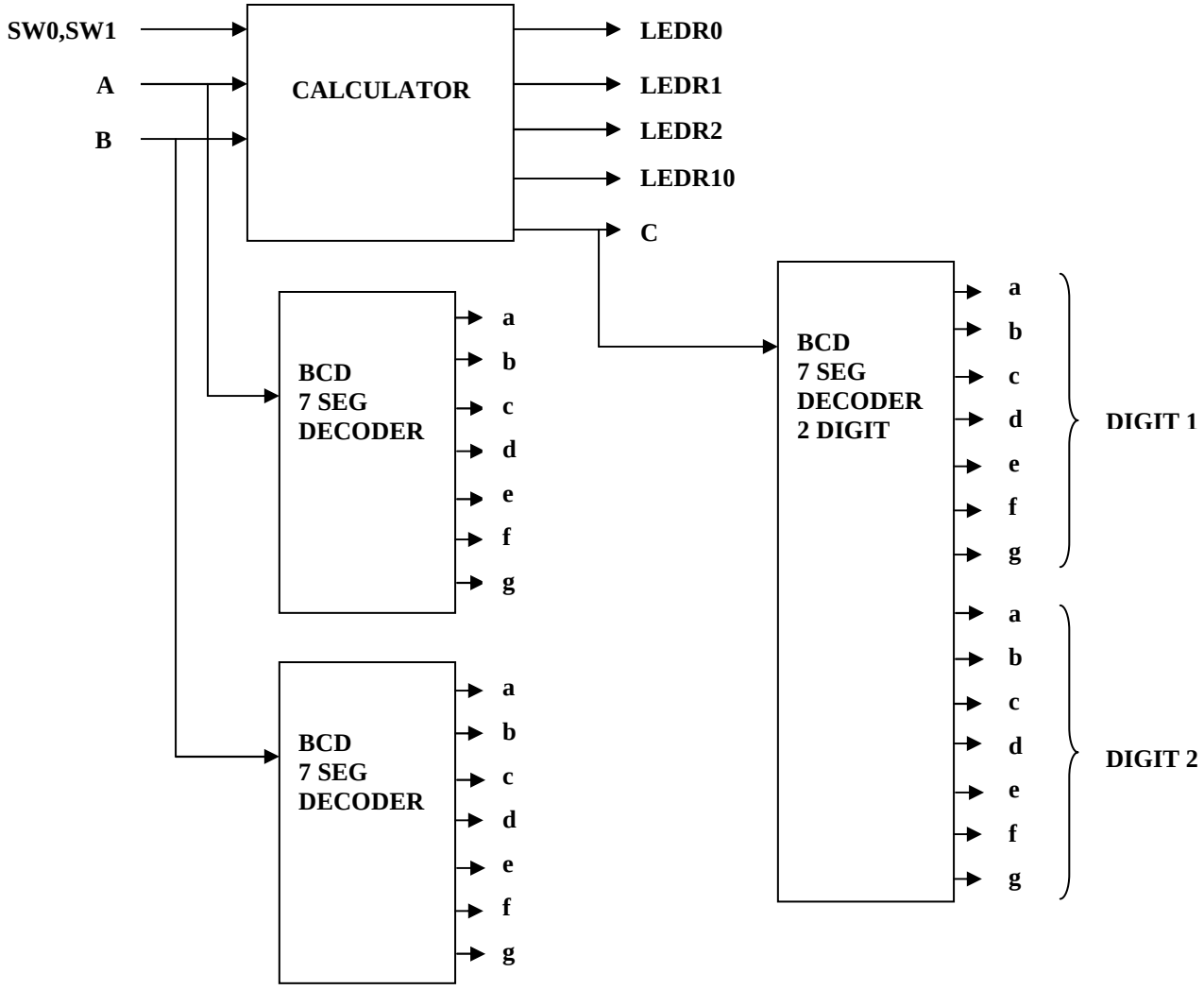
Simple calculator that adds, subtracts and multiplies two 3 bit numbers A and B that are assigned to the appropriate switches. Using two switches, the operation of additioning, subtracting or multiplying is determined. The numbers A, B, and the result C are displayed in the display. LEDR0 indicates that the addition, LEDR1 subtraction and LEDR2 multiplication is selected. LEDR10 indicates that the result of the operation is a negative number.

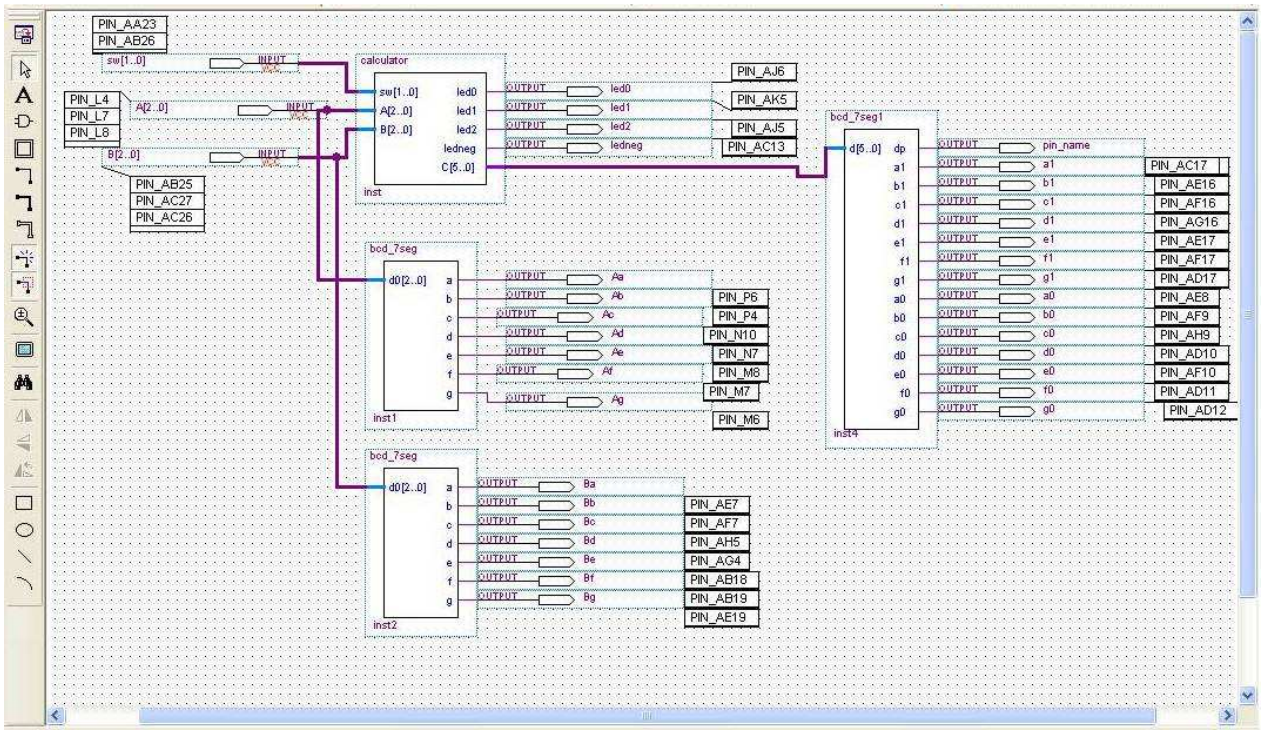
1. Opis Problema:

Potrebno je napraviti kod koji demonstrira rad prostog kalkulatora koji sabira, oduzima i množi dva 3 bitna broja A i B koji se zadaju odgovarajućim prekidačima . Pomoću dva prekidača određuje se operacija sabiranja ,oduzimanja ili množenja.Brojevi A , B i rezultat C prikazuju se na displeju. LEDR0 oznacava da je izabrano sabiranje, LEDR1 oduzimanje i LEDR2 množenje. LEDR10 označava da je rezultat operacije negativan broj.

2. Harversko-softversko resenje i simulacija:

2.1 Blok šema kalkulatora:





2.2 Objašnjenje kola:

Kalkulator sabira , oduzima i množi dva trobitna broja A i B koji se zadaju odgovarajućim prekidačima [SW15, SW16, SW17] i [SW2, SW3, SW4]. SW0 i SW1 se upotrebljavaju za selekciju operacije : 0,1 sabiranje ; 1,0 oduzimanje ; 1,1 mnozenje. Ulazni brojevi se prikazuju na HEX3 (A) i HEX2 (B) dok HEX1 i HEX0 prikazuju C (C=A+B ili A-B…). LEDR0 oznacava da je izabrano sabiranje, LEDR1 oduzimanje i LEDR2 mnozenje. LEDR10 oznacava da je rezultat operacije negativan broj.

2.3 Kod:

Calculator:

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

entity calculator is
port(sw:in std_logic_vector (1 downto 0);
A,B:in std_logic_vector(2 downto 0);
led0,led1,led2,ledneg : out std_logic;
C : out std_logic_vector(5 downto 0));
end calculator;
architecture arch of calculator is
begin
process(A,B,sw)
begin
        case sw is
                when "01"=>
                C<="000000"+A+B;
                ledneg<='0';
                led0<='1';
                led1<='0';
```

```
                    led2<='0';
                    when "10"=>
                    if (A<B)then
                    C<="000000"+B-A;
                    ledneg<='1';
                    led0<='0';
                    led1<='1';
                    led2<='0';
                    else
                    C<="000000"+A-B;
                    ledneg<='0';
                    led0<='0';
                    led1<='1';
                    led2<='0';
                    end if;
                    when "11"=>
                    C<="000000"+(A*B);
                    ledneg<='0';
                    led0<='0';
                    led1<='0';
                    led2<='1';
                    when "00"=>
                    C<="000000";
                    ledneg<='0';
                    led0<='0';
                    led1<='0';
                    led2<='0';
          end case;
end process;



end arch;
```

BCD_7SEG:

```
--bcd_7seg.vhd
--Common Anode BCD-to-seven-segment decoder
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
ENTITY bcd_7seg IS
PORT(
d0 : IN STD_LOGIC_vector (2 downto 0);
a, b, c, d, e, f, g : OUT STD_LOGIC);
END bcd_7seg;
ARCHITECTURE seven_segment OF bcd_7seg IS
SIGNAL input : STD_LOGIC_VECTOR (2 downto 0);
SIGNAL output : STD_LOGIC_VECTOR (6 downto 0);
BEGIN
input <= d0;
WITH input SELECT
output <= "0000001" WHEN "000",--display 0
"1001111" WHEN "001",--display 1
"0010010" WHEN "010",--display 2
"0000110" WHEN "011",--display 3
"1001100" WHEN "100",--display 4
"0100100" WHEN "101",--display 5
"0100000" WHEN "110",--display 6
"0001111" WHEN "111",--display 7


"1111111" WHEN others;
-- Separate the output vector to make individual pin outputs.
```

```
a <= output(6);
b <= output(5);
c <= output(4);
d <= output(3);
e <= output(2);
f <= output(1);
g <= output(0);
End seven_segment;
```

BCD_7SEG1:

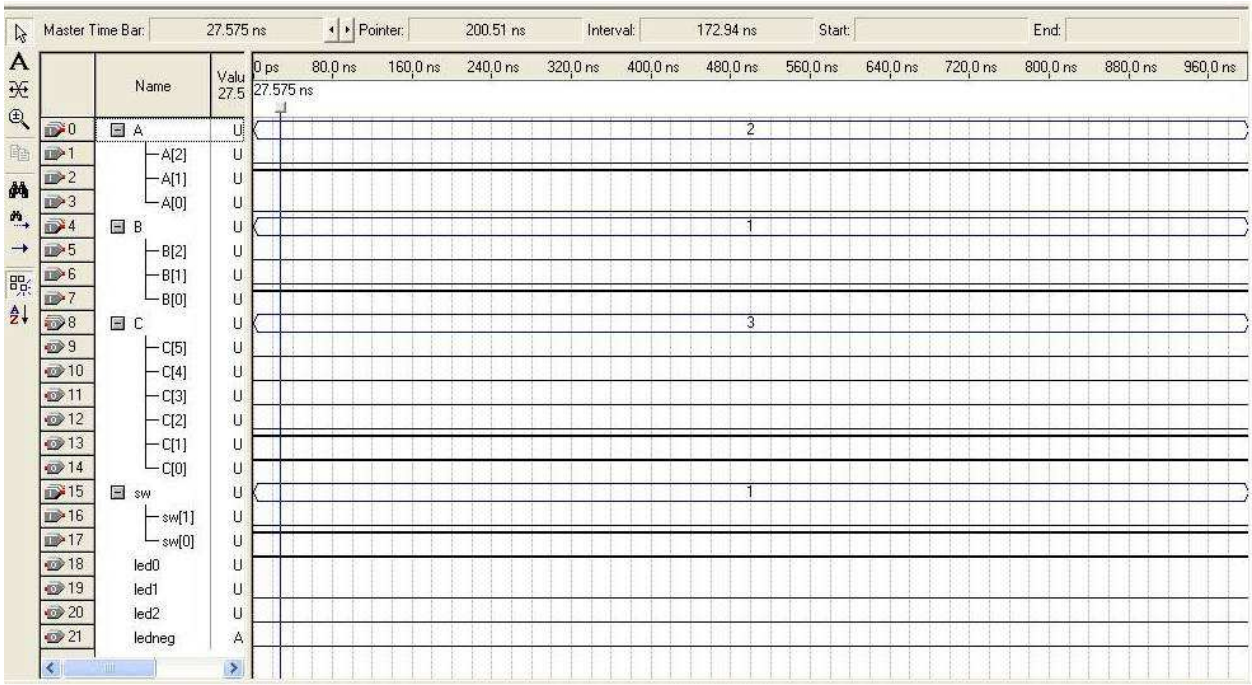
```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
ENTITY bcd_7seg1 IS
PORT(dp :out std_logic;
d : IN STD_LOGIC_vector (5 downto 0);
a1, b1, c1, d1, e1, f1, g1 ,a0, b0, c0, d0, e0, f0, g0: OUT STD_LOGIC);
END bcd_7seg1;
ARCHITECTURE seven_segment OF bcd_7seg1 IS
SIGNAL input : STD_LOGIC_VECTOR (5 downto 0);
SIGNAL output : STD_LOGIC_VECTOR (13 downto 0);

BEGIN
dp<='1';
input <= d;
WITH input SELECT
output <= "11111110000001" WHEN "000000",--display 0
"11111111001111" WHEN "000001",--display 1
"11111110010010" WHEN "000010",--display 2
"11111110000110" WHEN "000011",--display 3
"11111111001100" WHEN "000100",--display 4
"11111110100100" WHEN "000101",--display 5
"11111110100000" WHEN "000110",--display 6
"11111110001111" WHEN "000111",--display 7
"11111110000000" WHEN "001000",--display 8
"11111110000100" WHEN "001001",--display 9
"10011110000001" WHEN "001010",--display 10
"10011111001111" WHEN "001011",--display 11
"10011110010010" WHEN "001100",--display 12
"10011110000110" WHEN "001101",--display 13
"10011111001100" WHEN "001110",--display 14
"10011110100100" WHEN "001111",--display 15
"10011110100000" WHEN "010000",--display 16
"10011110001111" WHEN "010001",--display 17
"10011110000000" WHEN "010010",--display 18
"10011110000100" WHEN "010011",--display 19
"00100100000001" WHEN "010100",--display 20
"00100101001111" WHEN "010101",--display 21
"00100100010010" WHEN "010110",--display 22
"00100100000110" WHEN "010111",--display 23
"00100101001100" WHEN "011000",--display 24
"00100100100100" WHEN "011001",--display 25
"00100100100000" WHEN "011010",--display 26
"00100100001111" WHEN "011011",--display 27
"00100100000000" WHEN "011100",--display 28
"00100100000100" WHEN "011101",--display 29
"00001100000001" WHEN "011110",--display 30
"00001101001111" WHEN "011111",--display 31
"00001100010010" WHEN "100000",--display 32
"00001100000110" WHEN "100001",--display 33
"00001101001100" WHEN "100010",--display 34
"00001100100100" WHEN "100011",--display 35
"00001100100000" WHEN "100100",--display 36
```

"00001100001111" WHEN "100101",--display 37
"00001100000000" WHEN "100110",--display 38
"00001100000100" WHEN "100111",--display 39
"10011000000001" WHEN "101000",--display 40
"10011001001111" WHEN "101001",--display 41
"10011000010010" WHEN "101010",--display 42
"10011000000110" WHEN "101011",--display 43
"10011001001100" WHEN "101100",--display 44
"10011000100100" WHEN "101101",--display 45
"10011000100000" WHEN "101110",--display 46
"10011000001111" WHEN "101111",--display 47
"10011000000000" WHEN "110000",--display 48
"10011000000100" WHEN "110001",--display 49

"11111111111111" WHEN others;
-- Separate the output vector to make individual pin outputs.
a0 <= output(6);
b0 <= output(5);
c0 <= output(4);
d0 <= output(3);
e0 <= output(2);
f0 <= output(1);
g0 <= output(0);

a1 <= output(13);
b1 <= output(12);
c1 <= output(11);
d1 <= output(10);
e1 <= output(9);
f1 <= output(8);
g1 <= output(7);
End seven_segment;

2.4 Simulacija:



3. Link na video:
https://youtu.be/SeOYUwBK9Zs

4. Literatura:
http://apeg.ac.me/nastava/Getting_Started_with_DE2-70_board.pdf
http://apeg.ac.me/nastava/DE2-70-pins.pdf
https://www.researchgate.net/profile/Radovan_Stojanovic/publication/279771239_AUTOMATIZOVANO_PROJEKTOVANJE_DIGITALNIH_SISTEMA_VHDL_i_FPGA/links/559aa3e508ae21086d2765f4/AUTOMATIZOVANO-PROJEKTOVANJE-DIGITALNIH-SISTEMA-VHDL-i-FPGA.pdf