**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»**

Інститут комп’ютерних технологій, автоматики та метрології

Кафедра ЕОМ



**Звіт**

Лабораторна робота №3

З дисципліни: “Моделювання комп’ютерних систем”

Тема: "Поведінковий опис цифрового автомата. Перевірка роботи автомата за допомогою стенда Elbert V2 – Spartan 3A FPGA"

**Виконав:**

ст. гр. КІ-24

Дроздовський М.Б

**Перевірив:**

Кузьо М. М.

ЛЬВІВ 2020

**Мета роботи:**

На базі стенда реалізувати цифровий автомат для обчислення

значення виразу.

**MUX.vhd**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity MUX is

port(

InputA,InputB,InputC: in std\_logic\_vector(7 downto 0);

IN\_SEL: in std\_logic\_vector(1 downto 0);

MuxOut: out std\_logic\_vector(7 downto 0)

);

end MUX;

architecture MUX of MUX is

begin

MUX\_pr: process(InputA,InputB,InputC,IN\_SEL)

begin

case IN\_SEL is

when "00" =>

MuxOut<= InputA;

when "01" =>

MuxOut<= InputB;

when "10" =>

MuxOut<= InputC;

when others =>

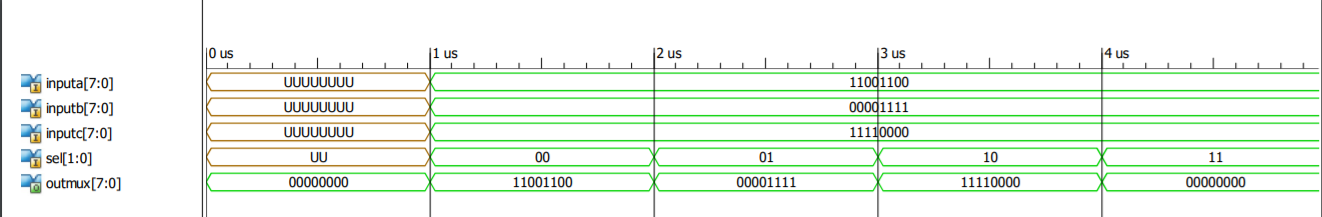
MuxOut<="00000000";

end case;

end process MUX\_pr;

end MUX;

**Симуляція MUX**



**ACC.vhd**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity ACC is

port(

CLK: in std\_logic;

WR,RST: in std\_logic;

InData: in std\_logic\_vector(7 downto 0);

OutData: out std\_logic\_vector(7 downto 0)

);

end ACC;

architecture ACC of ACC is

signal tempData: std\_logic\_vector(7 downto 0):="00000000";

begin

ACC\_proc : process(CLK, InData, WR, RST)

begin

if (rising\_edge(CLK)) then

if (WR = '1') then

tempData <= InData;

OutData<=InData;

elsif(RST = '1') then

tempData <= "00000000";

OutData<="00000000";

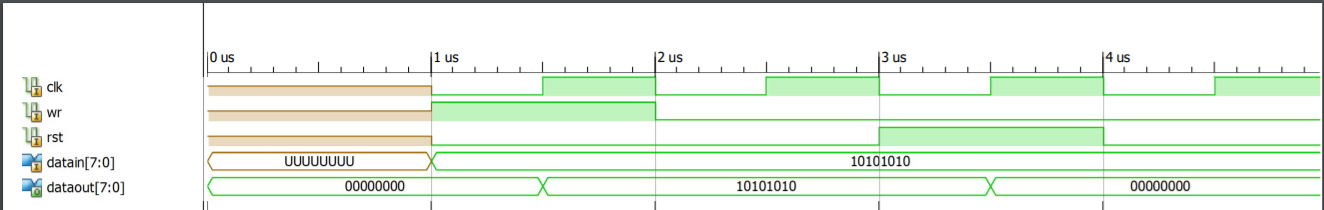
end if;

end if;

end process ACC\_proc;

end ACC;

**Симуляція ACC**

****

**ALU.vhd**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.NUMERIC\_STD.ALL;

entity ALU is

port(

A,B: in std\_logic\_vector(7 downto 0);

OP: in std\_logic\_vector(1 downto 0);

AluOut: out std\_logic\_vector(7 downto 0);

OverFlow: out std\_logic

);

end ALU;

architecture ALU of ALU is

signal tempOverFlow: std\_logic:='0';

begin

ALU\_pr: process(OP,A,B)

variable intA,intB,intC: integer;

variable Result: std\_logic\_vector(9 downto 0);

begin

intA:= to\_integer(unsigned(A));

intB:= to\_integer(unsigned(B));

if(tempOverFlow = '1') then

intC:= 1;

else

intC:= 0;

end if;

Result:= "0000000000";

case OP is

when "00" =>

Result:= std\_logic\_vector(to\_signed(intA+intB, 10));

tempOverFlow<=Result(8);

AluOut<= Result(7 downto 0);

when "01" =>

Result:= std\_logic\_vector(to\_signed(intA-intB+(intC\*256), 10));

tempOverFlow<=Result(8);

AluOut<= Result(7 downto 0);

when "10" =>

AluOut<= std\_logic\_vector(unsigned(A) sll intB);

when "11" =>

AluOut<=B;

when others =>

AluOut<="00000000";

tempOverFlow<='0';

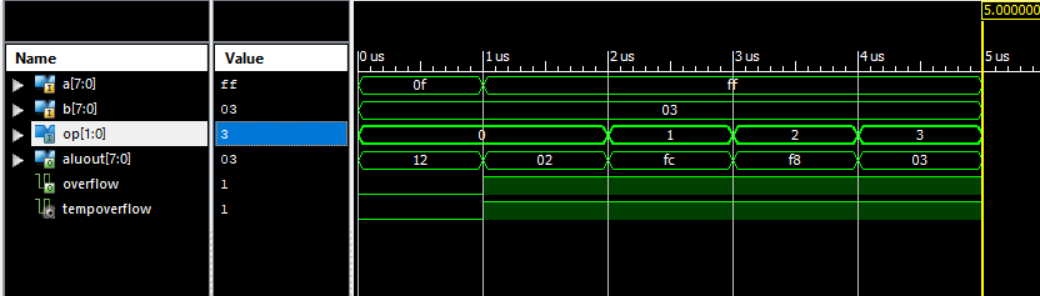
end case;

end process ALU\_pr;

OverFlow<=tempOverFlow;

end ALU;

**Симуляція ALU**

****

**RAM.vhd**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity RAM is

port(

CLK: in std\_logic;

InData: in std\_logic\_vector(7 downto 0);

ADDR, WR: in std\_logic;

RamOut: out std\_logic\_vector(7 downto 0)

);

end RAM;

architecture RAM of RAM is

signal OP1,OP2: std\_logic\_vector(7 downto 0):= "00000000";

begin

RAM\_pr: process(CLK,InData,ADDR,WR)

begin

if(rising\_edge(CLK)) then

if(ADDR = '0') then

if(WR = '1') then

OP1<= InData;

RamOut<= InData;

else

RamOut<= OP1;

end if;

elsif(ADDR = '1') then

if(WR = '1') then

OP2<= InData;

RamOut<= InData;

else

RamOut<= OP2;

end if;

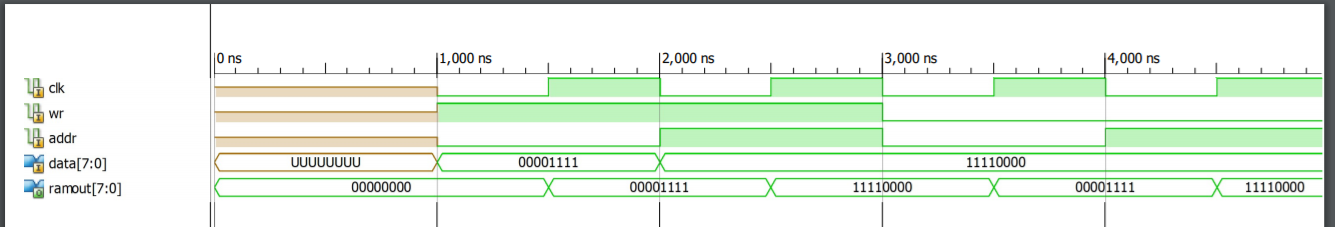
end if;

end if;

end process RAM\_pr;

end RAM;

**Симуляція RAM**



**CU.vhd**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.NUMERIC\_STD.ALL;

entity ALU is

port(

A,B: in std\_logic\_vector(7 downto 0);

OP: in std\_logic\_vector(1 downto 0);

AluOut: out std\_logic\_vector(7 downto 0);

OverFlow: out std\_logic

);

end ALU;

architecture ALU of ALU is

signal tempOverFlow: std\_logic:='0';

begin

ALU\_pr: process(OP,A,B)

variable intA,intB,intC: integer;

variable Result: std\_logic\_vector(9 downto 0);

begin

intA:= to\_integer(unsigned(A));

intB:= to\_integer(unsigned(B));

if(tempOverFlow = '1') then

intC:= 1;

else

intC:= 0;

end if;

Result:= "0000000000";

case OP is

when "00" =>

Result:= std\_logic\_vector(to\_signed(intA+intB, 10));

tempOverFlow<=Result(8);

AluOut<= Result(7 downto 0);

when "01" =>

Result:= std\_logic\_vector(to\_signed(intA-intB+(intC\*256), 10));

tempOverFlow<=Result(8);

AluOut<= Result(7 downto 0);

when "10" =>

AluOut<= std\_logic\_vector(unsigned(A) sll intB);

when "11" =>

AluOut<=B;

when others =>

AluOut<="00000000";

tempOverFlow<='0';

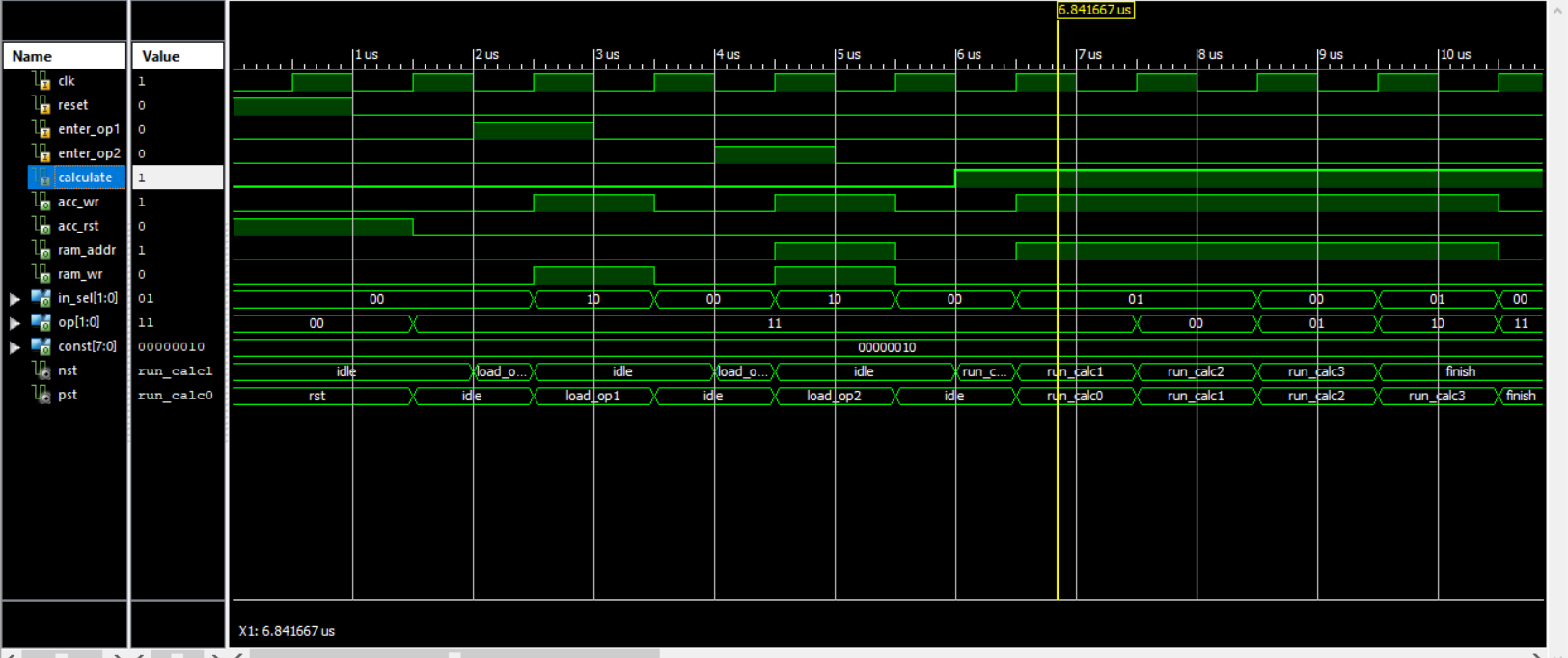
end case;

end process ALU\_pr;

OverFlow<=tempOverFlow;

end ALU;

**Симуляція CU**

****

**SevenSegDecoder.vhd**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.NUMERIC\_STD.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

entity SevenSegDecoder is

port(

InputData: in std\_logic\_vector(7 downto 0);

CLK: in std\_logic;

Seg\_A: out std\_logic;

Seg\_B: out std\_logic;

Seg\_C: out std\_logic;

Seg\_D: out std\_logic;

Seg\_E: out std\_logic;

Seg\_F: out std\_logic;

Seg\_G: out std\_logic;

Seg\_DP: out std\_logic;

Com\_Ones: out std\_logic;

Com\_Tens: out std\_logic;

Com\_Hundreds: out std\_logic

);

end SevenSegDecoder;

architecture SevenSegDecoder of SevenSegDecoder is

signal ONES\_BUS : STD\_LOGIC\_VECTOR(3 downto 0) := "0000";

signal DECS\_BUS : STD\_LOGIC\_VECTOR(3 downto 0) := "0000";

signal HONDREDS\_BUS : STD\_LOGIC\_VECTOR(3 downto 0) := "0000";

begin

BIN\_TO\_BCD : process (InputData)

variable hex\_src : STD\_LOGIC\_VECTOR(7 downto 0) ;

variable bcd : STD\_LOGIC\_VECTOR(11 downto 0) ;

begin

bcd := (others => '0') ;

hex\_src := InputData;

for i in hex\_src'range loop

if bcd(3 downto 0) > "0100" then

bcd(3 downto 0) := bcd(3 downto 0) + "0011" ;

end if ;

if bcd(7 downto 4) > "0100" then

bcd(7 downto 4) := bcd(7 downto 4) + "0011" ;

end if ;

if bcd(11 downto 8) > "0100" then

bcd(11 downto 8) := bcd(11 downto 8) + "0011" ;

end if ;

bcd := bcd(10 downto 0) & hex\_src(hex\_src'left) ; -- shift bcd + 1 new entry

hex\_src := hex\_src(hex\_src'left - 1 downto hex\_src'right) & '0' ; -- shift src + pad with 0

end loop ;

HONDREDS\_BUS <= bcd (11 downto 8);

DECS\_BUS <= bcd (7 downto 4);

ONES\_BUS <= bcd (3 downto 0);

end process BIN\_TO\_BCD;

INDICATE : process(CLK)

type DIGIT\_TYPE is (ONES, DECS, HUNDREDS);

variable CUR\_DIGIT : DIGIT\_TYPE := ONES;

variable DIGIT\_VAL : STD\_LOGIC\_VECTOR(3 downto 0) := "0000";

variable DIGIT\_CTRL : STD\_LOGIC\_VECTOR(6 downto 0) := "0000000";

variable COMMONS\_CTRL : STD\_LOGIC\_VECTOR(2 downto 0) := "000";

begin

if (rising\_edge(CLK)) then

case CUR\_DIGIT is

when ONES =>

DIGIT\_VAL := ONES\_BUS;

CUR\_DIGIT := DECS;

COMMONS\_CTRL := "001";

when DECS =>

DIGIT\_VAL := DECS\_BUS;

CUR\_DIGIT := HUNDREDS;

COMMONS\_CTRL := "010";

when HUNDREDS =>

DIGIT\_VAL := HONDREDS\_BUS;

CUR\_DIGIT := ONES;

COMMONS\_CTRL := "100";

when others =>

DIGIT\_VAL := ONES\_BUS;

CUR\_DIGIT := ONES;

COMMONS\_CTRL := "000";

end case;

case DIGIT\_VAL is --abcdefg

when "0000" => DIGIT\_CTRL := "1111110";

when "0001" => DIGIT\_CTRL := "0110000";

when "0010" => DIGIT\_CTRL := "1101101";

when "0011" => DIGIT\_CTRL := "1111001";

when "0100" => DIGIT\_CTRL := "0110011";

when "0101" => DIGIT\_CTRL := "1011011";

when "0110" => DIGIT\_CTRL := "1011111";

when "0111" => DIGIT\_CTRL := "1110000";

when "1000" => DIGIT\_CTRL := "1111111";

when "1001" => DIGIT\_CTRL := "1111011";

when others => DIGIT\_CTRL := "0000000";

end case;

COM\_ONES <= COMMONS\_CTRL(0);

COM\_TENS <= COMMONS\_CTRL(1);

COM\_HUNDREDS <= COMMONS\_CTRL(2);

SEG\_A <= DIGIT\_CTRL(6);

SEG\_B <= DIGIT\_CTRL(5);

SEG\_C <= DIGIT\_CTRL(4);

SEG\_D <= DIGIT\_CTRL(3);

SEG\_E <= DIGIT\_CTRL(2);

SEG\_F <= DIGIT\_CTRL(1);

SEG\_G <= DIGIT\_CTRL(0);

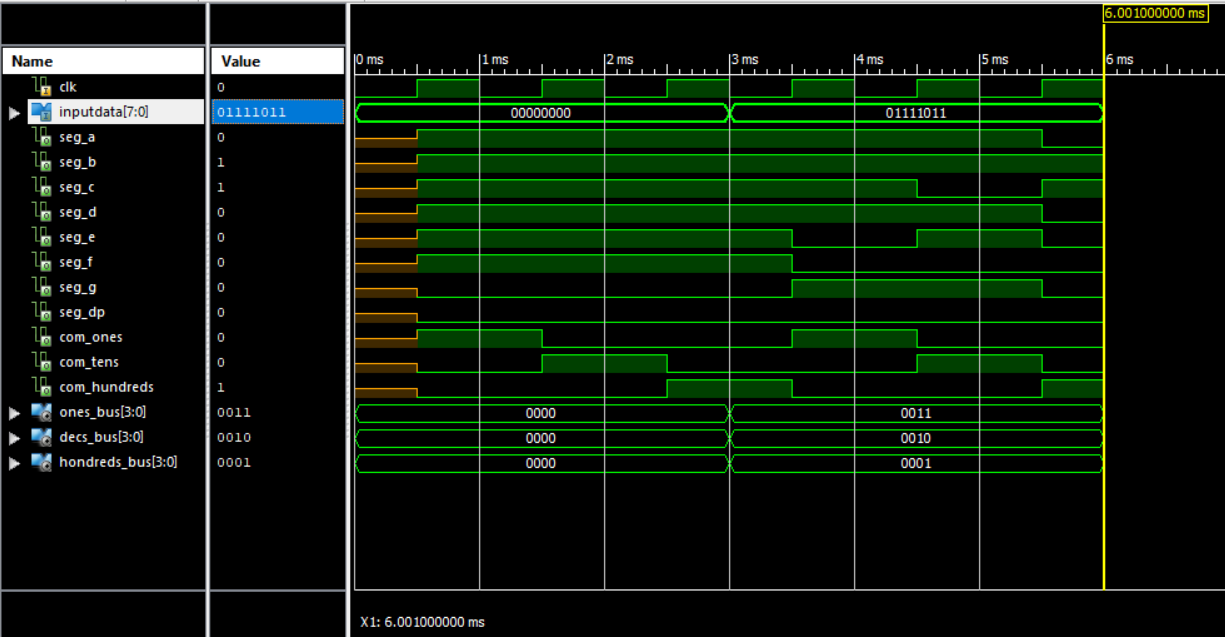
SEG\_DP <= '0';

end if;

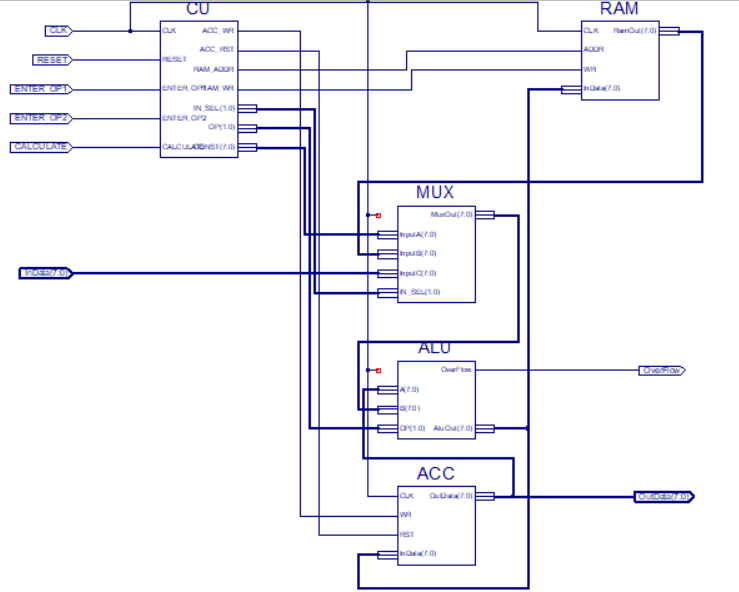
end process INDICATE;

end SevenSegDecoder;

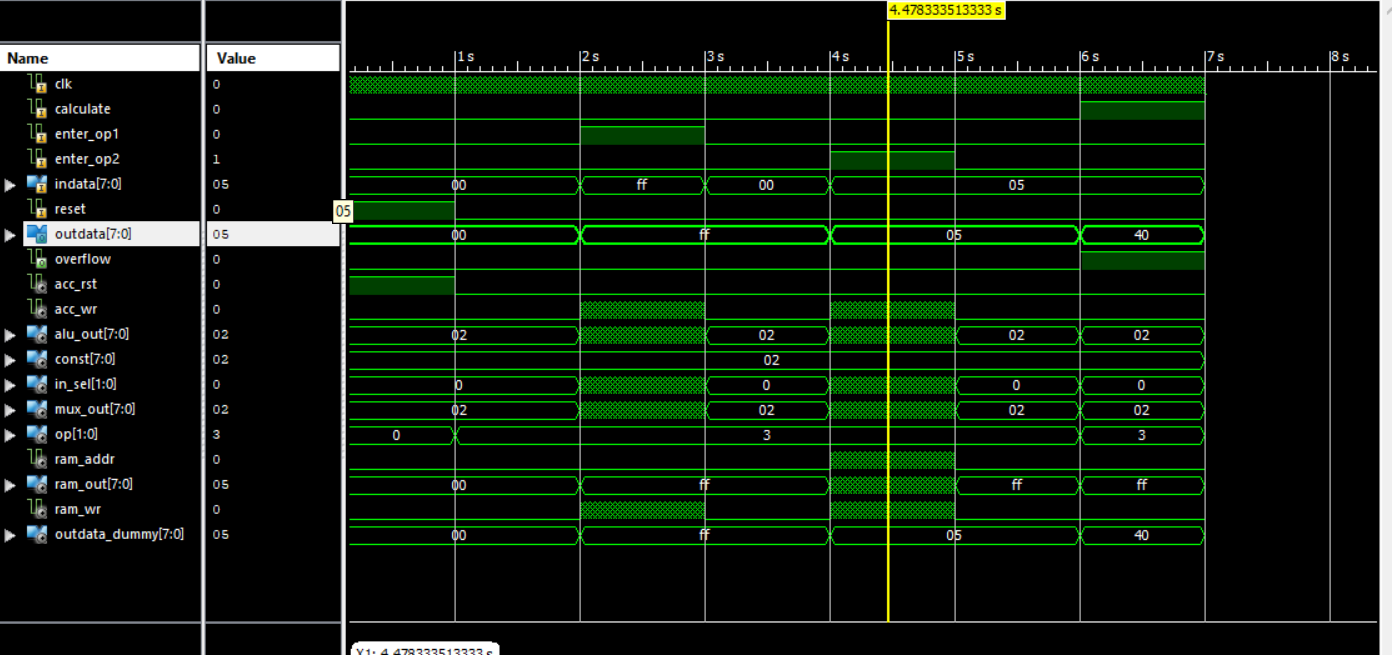
**Симуляція SevenSegDecoder**

****

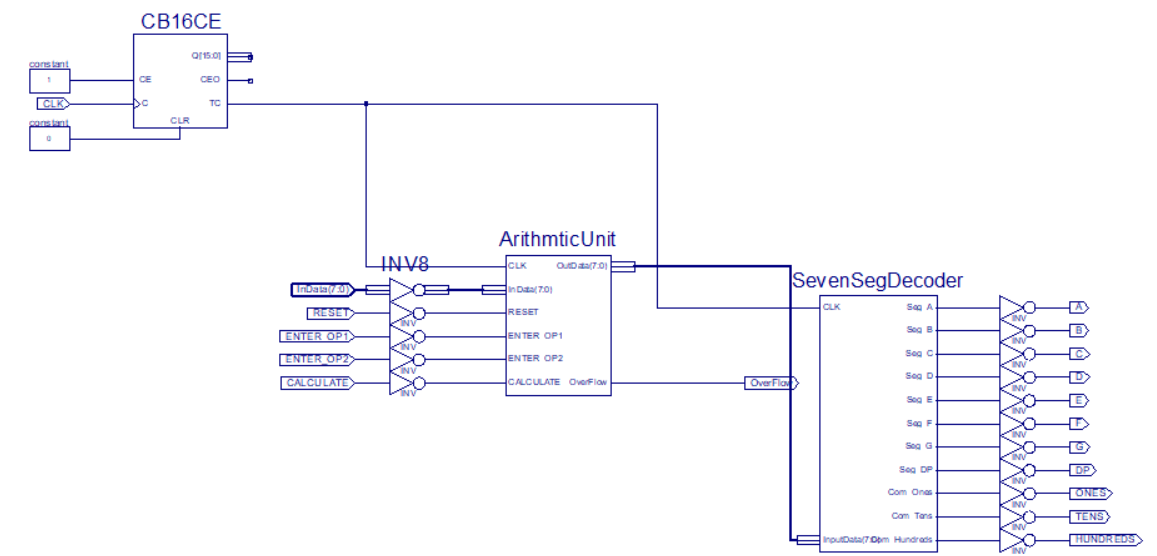
**ArithmeticUnit.sch**

****

**Симуляція ArithmeticUnit**

****

**TopLevel.sch**

****

**Висновок:**

На лабораторній роботі я синтезував цифровий обчислювач та перевірив його роботу.