Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кафедра ЕОМ

Звіт

З лабораторної роботи № 3

З *дисципліни «Моделювання комп'юткрних систем»*

На тему: «Поведінковий опис цифрового автомата Перевірка роботи автомата за допомогою стенда»

Виконав: ст. гр. КІ-202

Довганюк О. С.

Прийняв: ст. в.

Козак Н. Б.

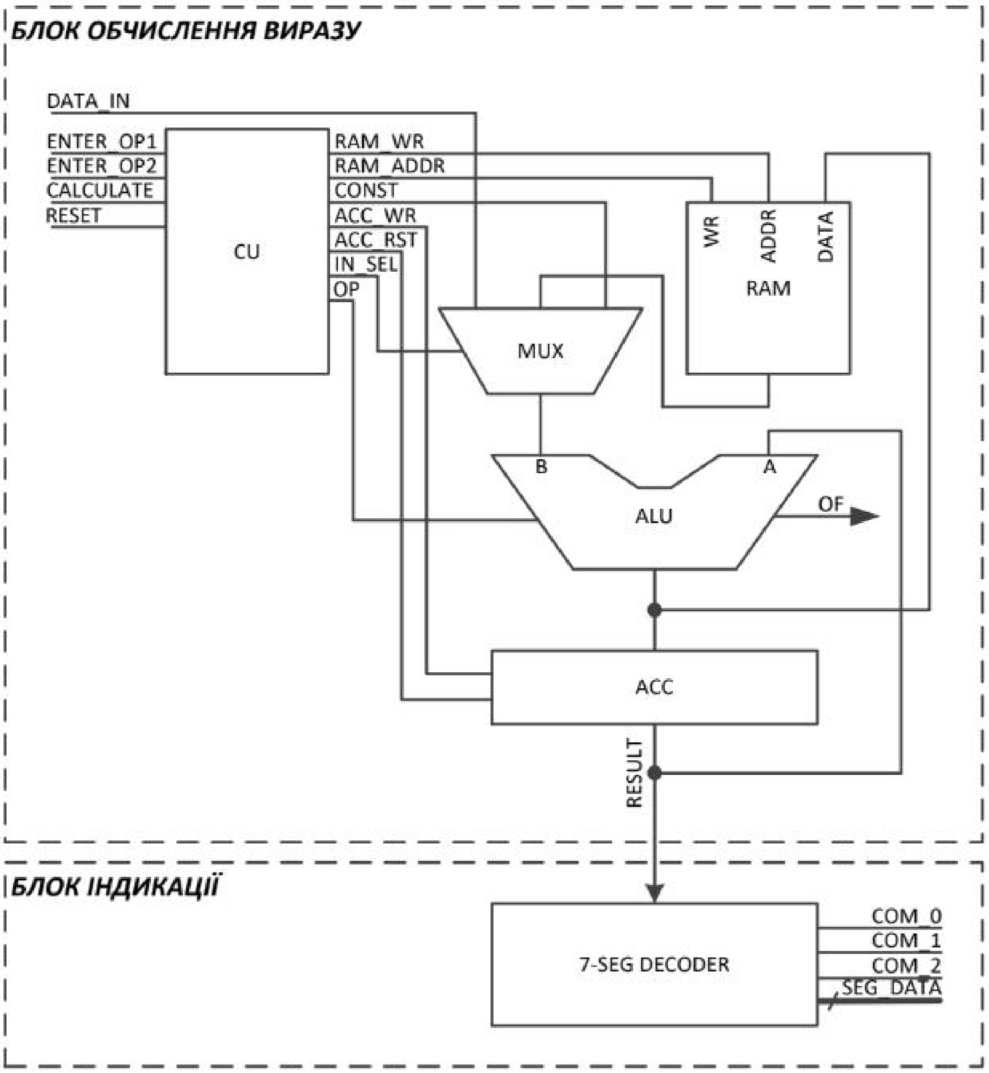
Львів – 2023

Мета роботи

На базі стенда реалізувати цифровий автомат для обчислення значення виразу.

Завдання

1. Функціонал пристрою повинен бути реалізований згідно отриманого варіанту завдання. Дивись розділ ЗАВДАННЯ:.
2. Пристрій повинен бути ітераційним (АЛП *(ALU)* повинен виконувати за один такт одну операцію), та реалізованим згідно наступної структурної схеми (*Малюнок 1*):

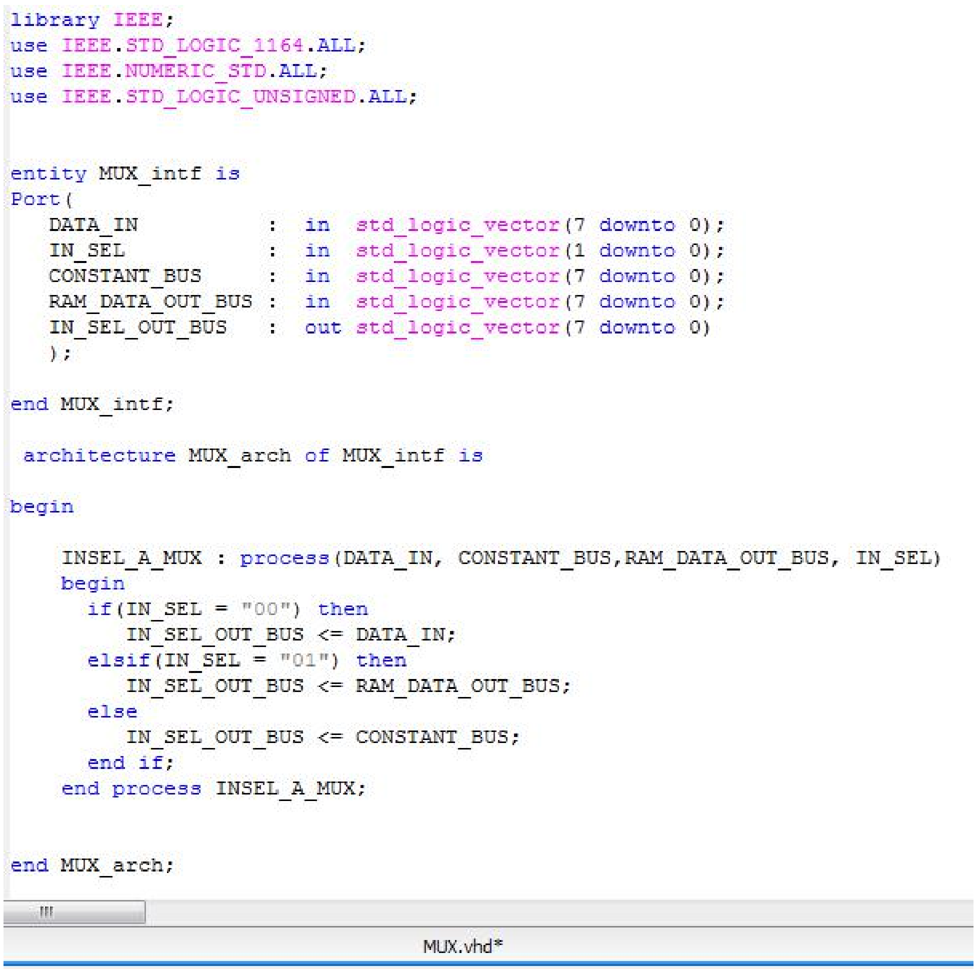


Варіант 7(7):

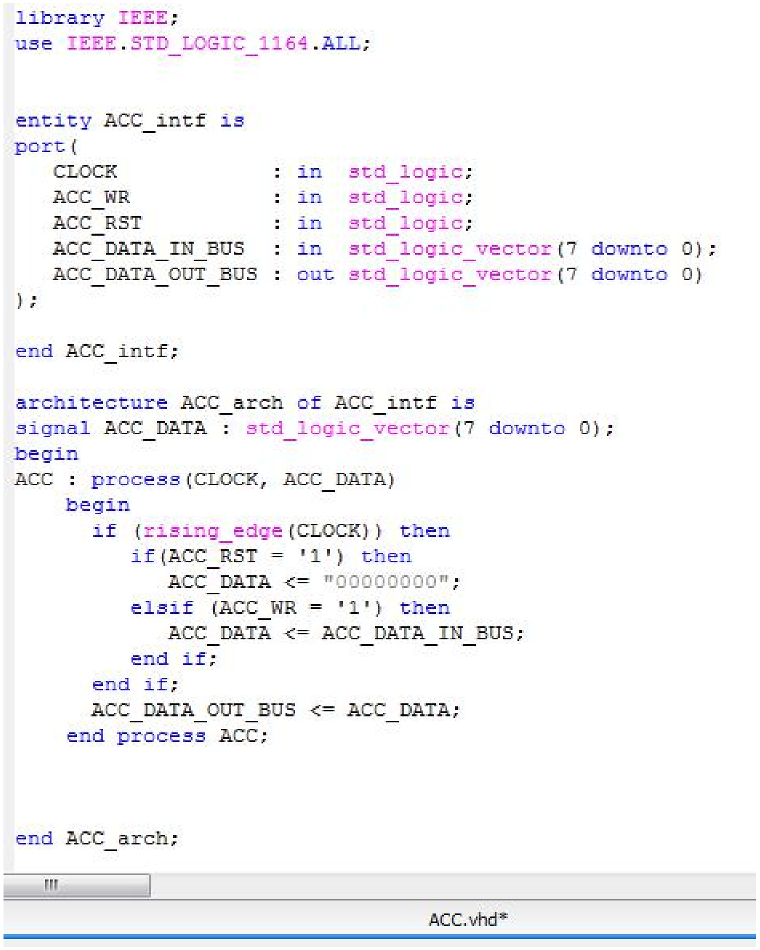
7	((OP1 << 2) – OP2) + 4

Виконання роботи

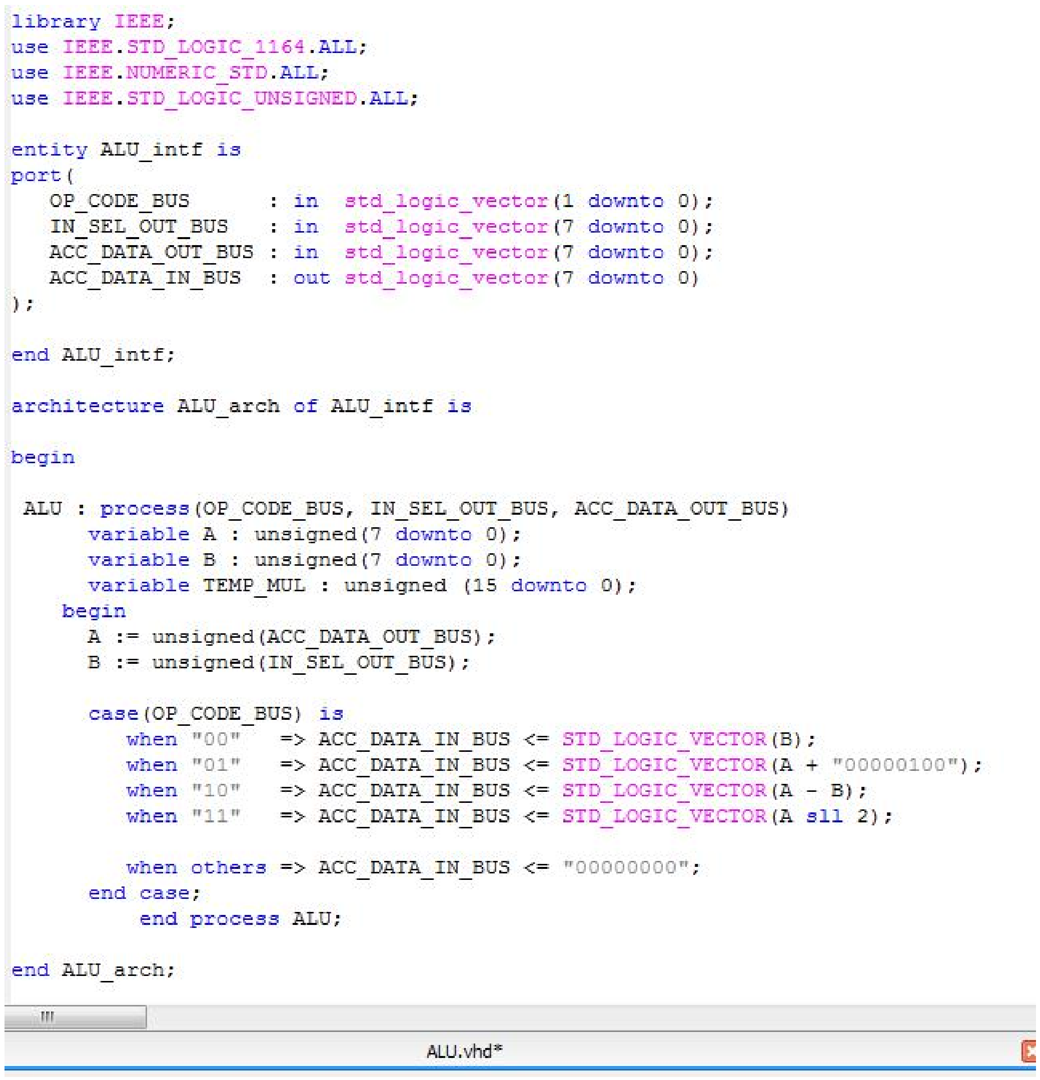
1) Спочатку створюю новий проект користуючись методичними вказівками до лабораторної роботи No1.
2) Додаю до проекту нові *.vhd файли в яких реалізовую логіку: мультиплексора MUX, регістра ACC, арифметико-логічного пристрою ALU, пристрою керування CU, пам’яті пристрою RAM, перетворювача блоку 7-ми сегментних індикаторів DEC:



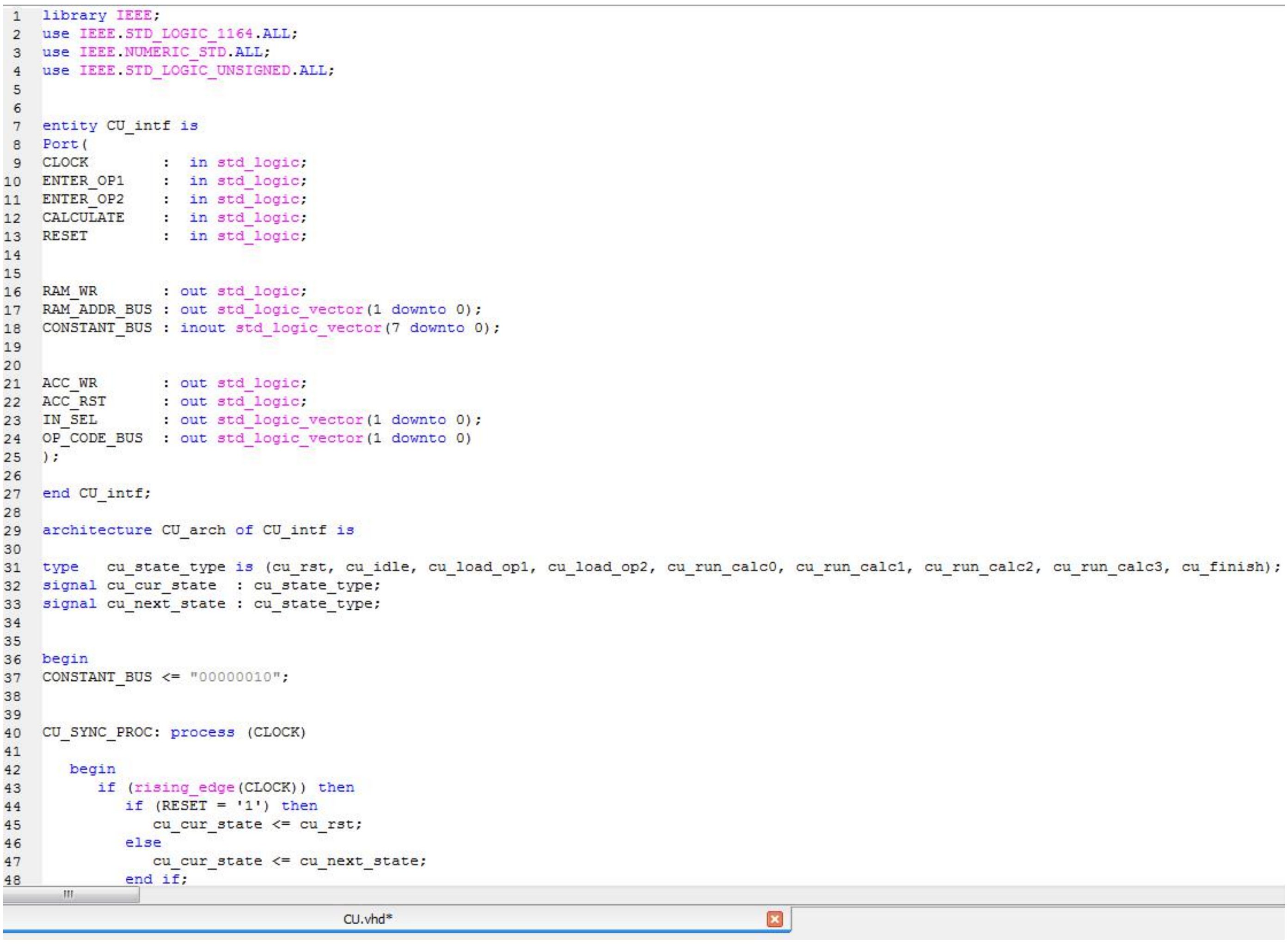
```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;


entity MUX_intf is
Port(
   DATA_IN          :  in  std_logic_vector(7 downto 0);
   IN_SEL           :  in  std_logic_vector(1 downto 0);
   CONSTANT_BUS     :  in  std_logic_vector(7 downto 0);
   RAM_DATA_OUT_BUS :  in  std_logic_vector(7 downto 0);
   IN_SEL_OUT_BUS   :  out std_logic_vector(7 downto 0)
   );

end MUX_intf;

 architecture MUX_arch of MUX_intf is

begin

    INSEL_A_MUX : process(DATA_IN, CONSTANT_BUS,RAM_DATA_OUT_BUS, IN_SEL)
    begin
      if(IN_SEL = "00") then
         IN_SEL_OUT_BUS <= DATA_IN;
      elsif(IN_SEL = "01") then
         IN_SEL_OUT_BUS <= RAM_DATA_OUT_BUS;
      else
         IN_SEL_OUT_BUS <= CONSTANT_BUS;
      end if;
    end process INSEL_A_MUX;


end MUX_arch;
```



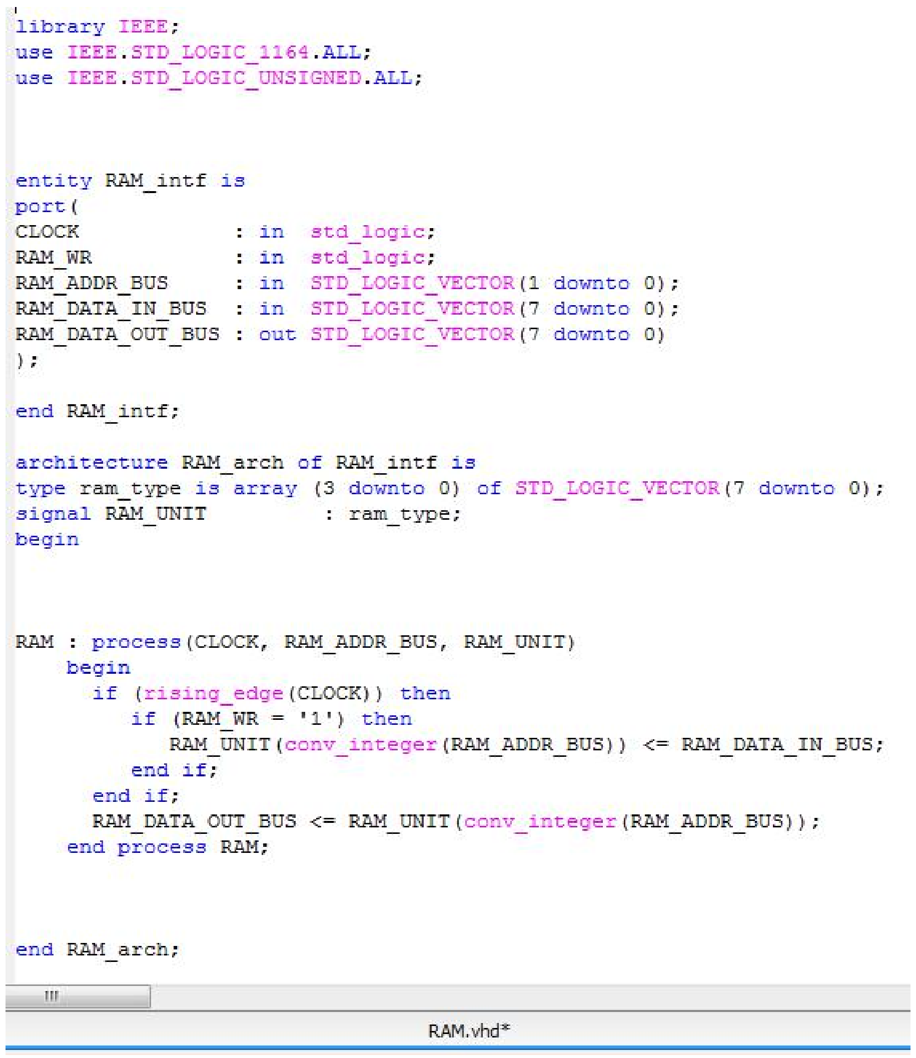
```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;


entity ACC_intf is
port(
   CLOCK            : in  std_logic;
   ACC_WR           : in  std_logic;
   ACC_RST          : in  std_logic;
   ACC_DATA_IN_BUS  : in  std_logic_vector(7 downto 0);
   ACC_DATA_OUT_BUS : out std_logic_vector(7 downto 0)
);

end ACC_intf;

architecture ACC_arch of ACC_intf is
signal ACC_DATA : std_logic_vector(7 downto 0);
begin
ACC : process(CLOCK, ACC_DATA)
    begin
      if (rising_edge(CLOCK)) then
         if(ACC_RST = '1') then
            ACC_DATA <= "00000000";
         elsif (ACC_WR = '1') then
            ACC_DATA <= ACC_DATA_IN_BUS;
         end if;
      end if;
      ACC_DATA_OUT_BUS <= ACC_DATA;
    end process ACC;



end ACC_arch;
```

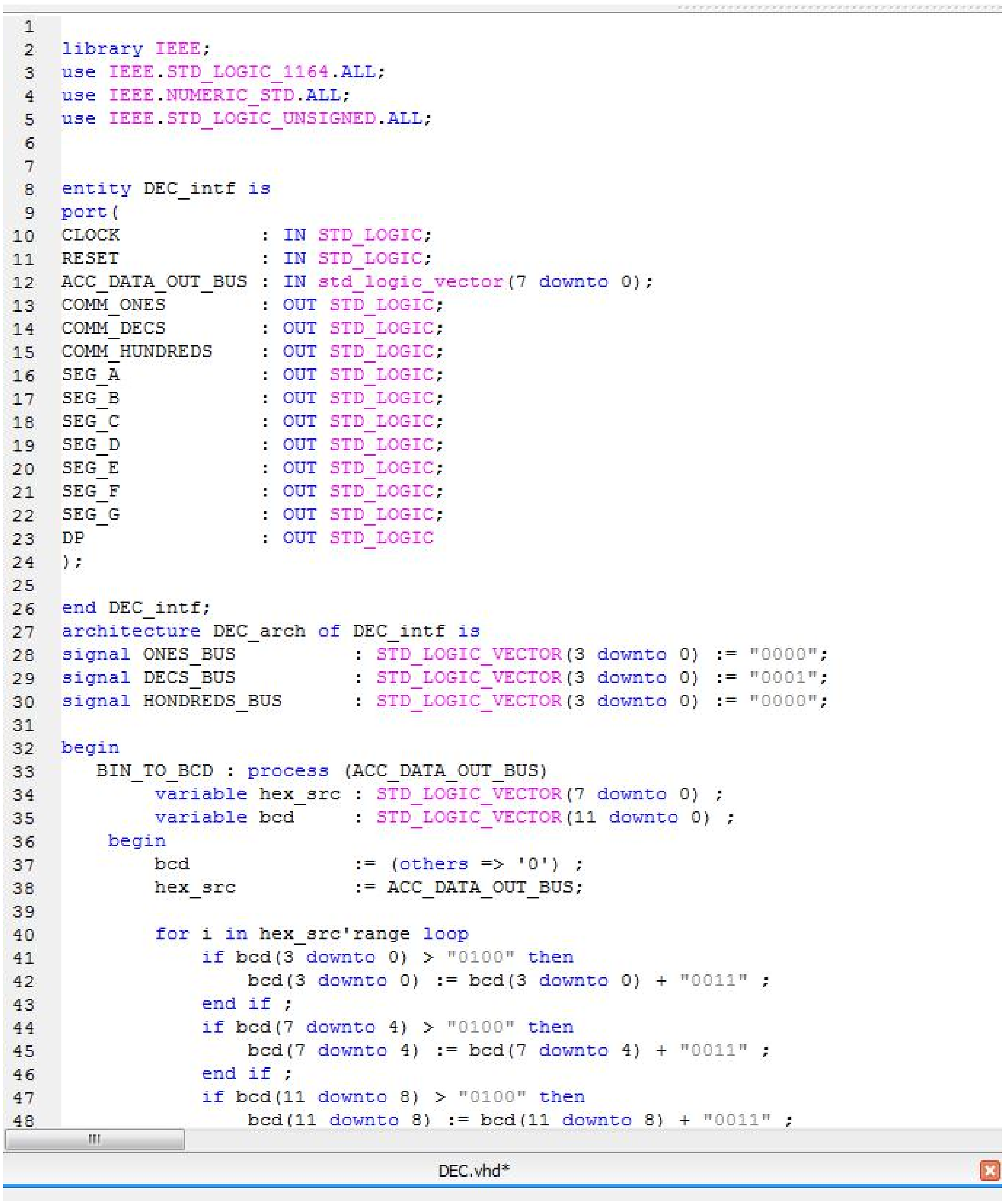


```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity ALU_intf is
port(
   OP_CODE_BUS       : in  std_logic_vector(1 downto 0);
   IN_SEL_OUT_BUS    : in  std_logic_vector(7 downto 0);
   ACC_DATA_OUT_BUS  : in  std_logic_vector(7 downto 0);
   ACC_DATA_IN_BUS   : out std_logic_vector(7 downto 0)
);

end ALU_intf;

architecture ALU_arch of ALU_intf is

begin

 ALU : process(OP_CODE_BUS, IN_SEL_OUT_BUS, ACC_DATA_OUT_BUS)
      variable A : unsigned(7 downto 0);
      variable B : unsigned(7 downto 0);
      variable TEMP_MUL : unsigned (15 downto 0);
    begin
      A := unsigned(ACC_DATA_OUT_BUS);
      B := unsigned(IN_SEL_OUT_BUS);

      case(OP_CODE_BUS) is
         when "00"   => ACC_DATA_IN_BUS <= STD_LOGIC_VECTOR(B);
         when "01"   => ACC_DATA_IN_BUS <= STD_LOGIC_VECTOR(A + "00000100");
         when "10"   => ACC_DATA_IN_BUS <= STD_LOGIC_VECTOR(A - B);
         when "11"   => ACC_DATA_IN_BUS <= STD_LOGIC_VECTOR(A sll 2);

         when others => ACC_DATA_IN_BUS <= "00000000";
      end case;
          end process ALU;

end ALU_arch;
```



```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;


entity CU_intf is
Port(
CLOCK        :  in std_logic;
ENTER_OP1    :  in std_logic;
ENTER_OP2    :  in std_logic;
CALCULATE    :  in std_logic;
RESET        :  in std_logic;


RAM_WR       : out std_logic;
RAM_ADDR_BUS : out std_logic_vector(1 downto 0);
CONSTANT_BUS : inout std_logic_vector(7 downto 0);


ACC_WR       : out std_logic;
ACC_RST      : out std_logic;
IN_SEL       : out std_logic_vector(1 downto 0);
OP_CODE_BUS  : out std_logic_vector(1 downto 0)
);

end CU_intf;

architecture CU_arch of CU_intf is

type   cu_state_type is (cu_rst, cu_idle, cu_load_op1, cu_load_op2, cu_run_calc0, cu_run_calc1, cu_run_calc2, cu_run_calc3, cu_finish);
signal cu_cur_state  : cu_state_type;
signal cu_next_state : cu_state_type;


begin
CONSTANT_BUS <= "00000010";


CU_SYNC_PROC: process (CLOCK)

   begin
      if (rising_edge(CLOCK)) then
         if (RESET = '1') then
            cu_cur_state <= cu_rst;
         else
            cu_cur_state <= cu_next_state;
         end if;
```

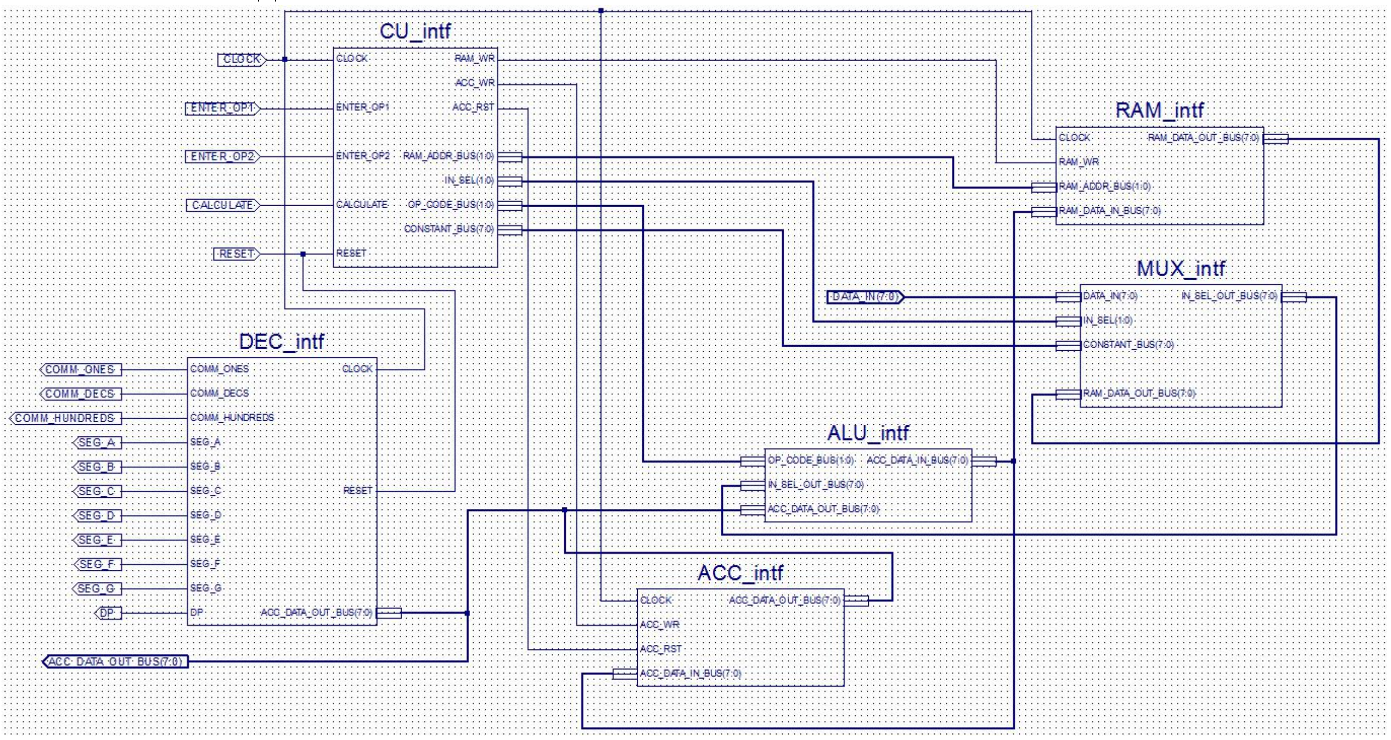

```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;



entity RAM_intf is
port(
CLOCK            : in  std_logic;
RAM_WR           : in  std_logic;
RAM_ADDR_BUS     : in  STD_LOGIC_VECTOR(1 downto 0);
RAM_DATA_IN_BUS  : in  STD_LOGIC_VECTOR(7 downto 0);
RAM_DATA_OUT_BUS : out STD_LOGIC_VECTOR(7 downto 0)
);

end RAM_intf;

architecture RAM_arch of RAM_intf is
type ram_type is array (3 downto 0) of STD_LOGIC_VECTOR(7 downto 0);
signal RAM_UNIT          : ram_type;
begin



RAM : process(CLOCK, RAM_ADDR_BUS, RAM_UNIT)
    begin
      if (rising_edge(CLOCK)) then
         if (RAM_WR = '1') then
            RAM_UNIT(conv_integer(RAM_ADDR_BUS)) <= RAM_DATA_IN_BUS;
         end if;
      end if;
      RAM_DATA_OUT_BUS <= RAM_UNIT(conv_integer(RAM_ADDR_BUS));
    end process RAM;



end RAM_arch;
```



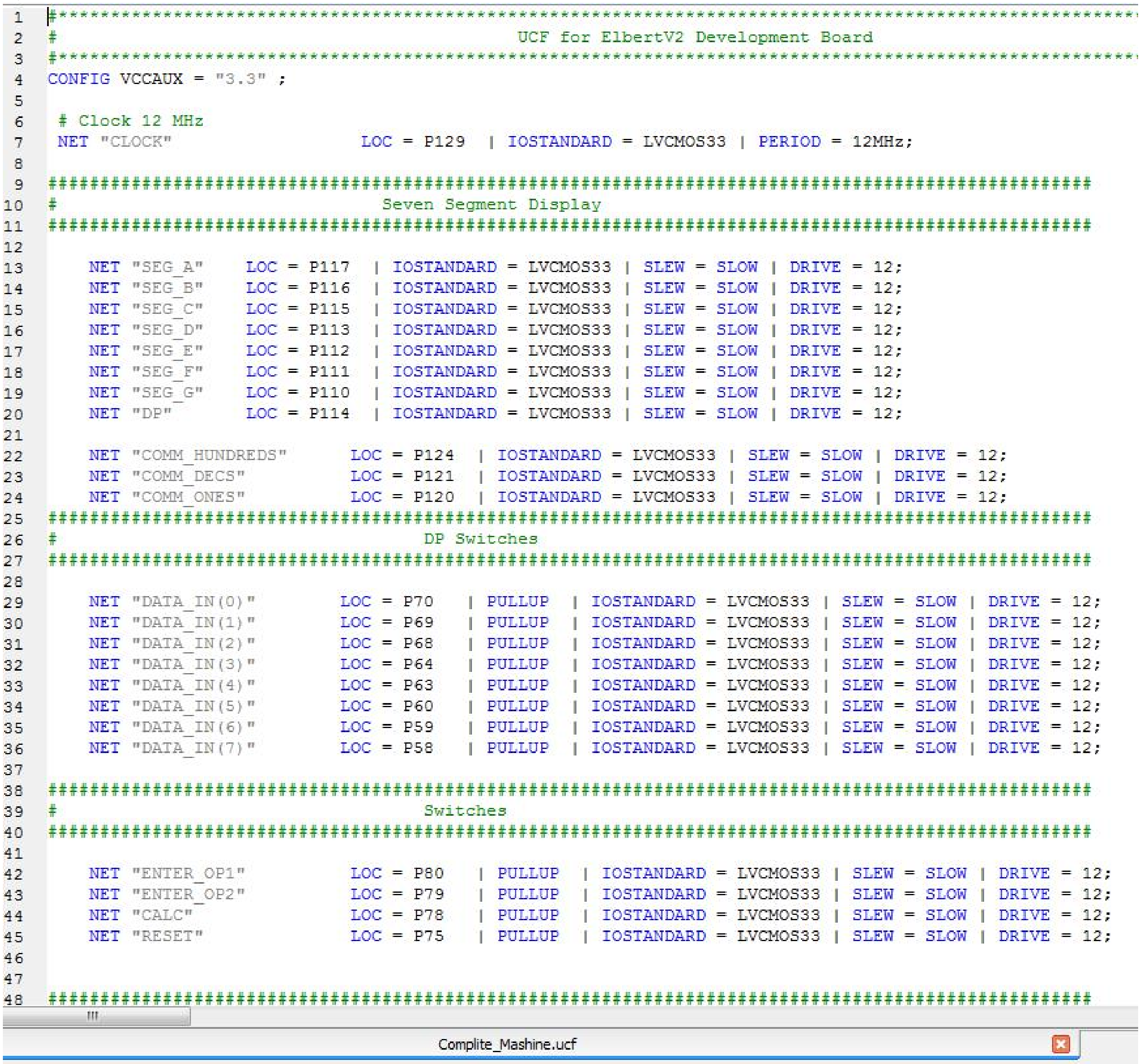
```vhdl

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;


entity DEC_intf is
port(
CLOCK            : IN STD_LOGIC;
RESET            : IN STD_LOGIC;
ACC_DATA_OUT_BUS : IN std_logic_vector(7 downto 0);
COMM_ONES        : OUT STD_LOGIC;
COMM_DECS        : OUT STD_LOGIC;
COMM_HUNDREDS    : OUT 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;
DP               : OUT STD_LOGIC
);

end DEC_intf;
architecture DEC_arch of DEC_intf is
signal ONES_BUS         : STD_LOGIC_VECTOR(3 downto 0) := "0000";
signal DECS_BUS         : STD_LOGIC_VECTOR(3 downto 0) := "0001";
signal HONDREDS_BUS     : STD_LOGIC_VECTOR(3 downto 0) := "0000";

begin
   BIN_TO_BCD : process (ACC_DATA_OUT_BUS)
        variable hex_src : STD_LOGIC_VECTOR(7 downto 0) ;
        variable bcd     : STD_LOGIC_VECTOR(11 downto 0) ;
    begin
        bcd              := (others => '0') ;
        hex_src          := ACC_DATA_OUT_BUS;

        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" ;
```

3) Генерую Schematic символи для створених файлів.

4) Створюю файл TopLevel.sch, та виконую інтеграцію компонентів системи між собою та зі стендом:

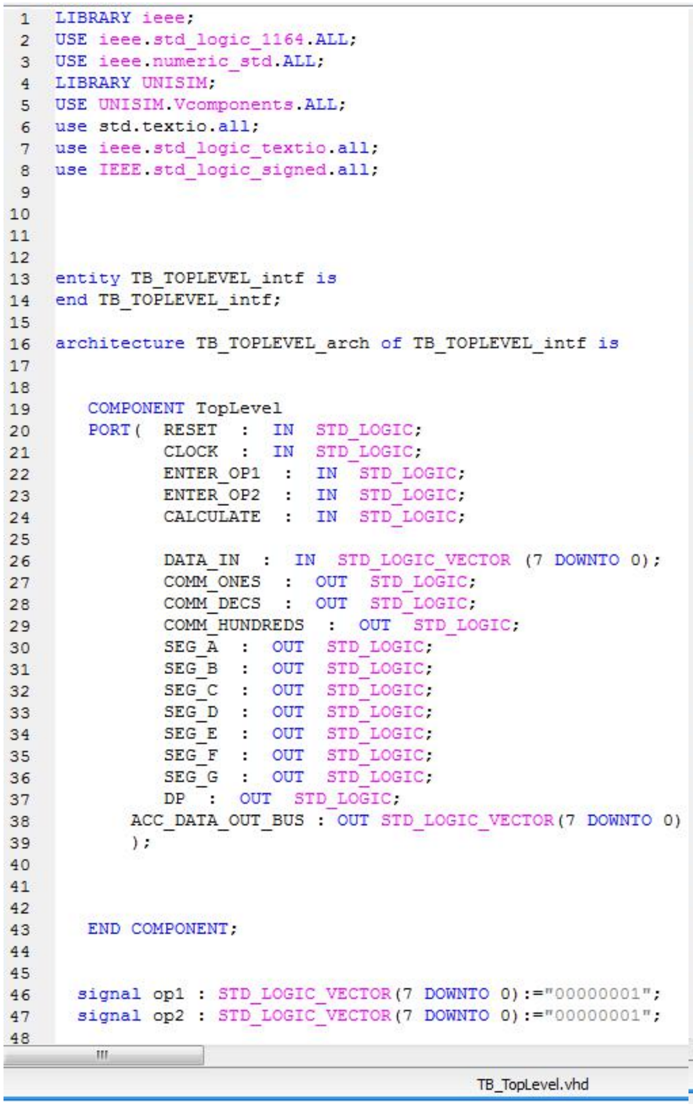


5) Створюю файл Constraints.ucf та призначаю виводам схеми фізичні виводи цільової FPGA:



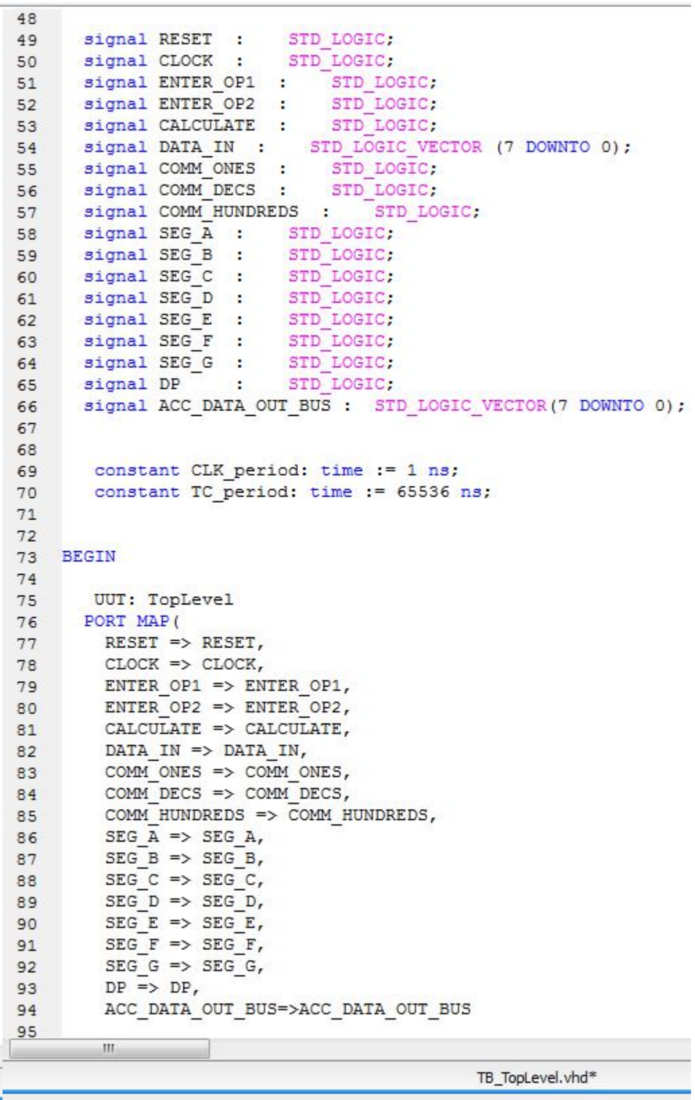
```
#********************************************************************************************************
#                                              UCF for ElbertV2 Development Board
#********************************************************************************************************
CONFIG VCCAUX = "3.3" ;

 # Clock 12 MHz
 NET "CLOCK"                  LOC = P129  | IOSTANDARD = LVCMOS33 | PERIOD = 12MHz;

####################################################################################################
#                                Seven Segment Display
####################################################################################################

    NET "SEG_A"    LOC = P117  | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
    NET "SEG_B"    LOC = P116  | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
    NET "SEG_C"    LOC = P115  | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
    NET "SEG_D"    LOC = P113  | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
    NET "SEG_E"    LOC = P112  | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
    NET "SEG_F"    LOC = P111  | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
    NET "SEG_G"    LOC = P110  | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
    NET "DP"       LOC = P114  | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;

    NET "COMM_HUNDREDS"      LOC = P124  | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
    NET "COMM_DECS"          LOC = P121  | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
    NET "COMM_ONES"          LOC = P120  | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
####################################################################################################
#                                    DP Switches
####################################################################################################

    NET "DATA_IN(0)"        LOC = P70   | PULLUP   | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
    NET "DATA_IN(1)"        LOC = P69   | PULLUP   | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
    NET "DATA_IN(2)"        LOC = P68   | PULLUP   | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
    NET "DATA_IN(3)"        LOC = P64   | PULLUP   | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
    NET "DATA_IN(4)"        LOC = P63   | PULLUP   | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
    NET "DATA_IN(5)"        LOC = P60   | PULLUP   | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
    NET "DATA_IN(6)"        LOC = P59   | PULLUP   | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
    NET "DATA_IN(7)"        LOC = P58   | PULLUP   | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;

####################################################################################################
#                                    Switches
####################################################################################################

    NET "ENTER_OP1"          LOC = P80   | PULLUP   | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
    NET "ENTER_OP2"          LOC = P79   | PULLUP   | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
    NET "CALC"               LOC = P78   | PULLUP   | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;
    NET "RESET"              LOC = P75   | PULLUP   | IOSTANDARD = LVCMOS33 | SLEW = SLOW | DRIVE = 12;


####################################################################################################
```

6) Створюю файл TestBenchTopLevel.vhd та прописую в ньому поведінку вхідних сигналів для тестування схеми за допомогою симулятора:



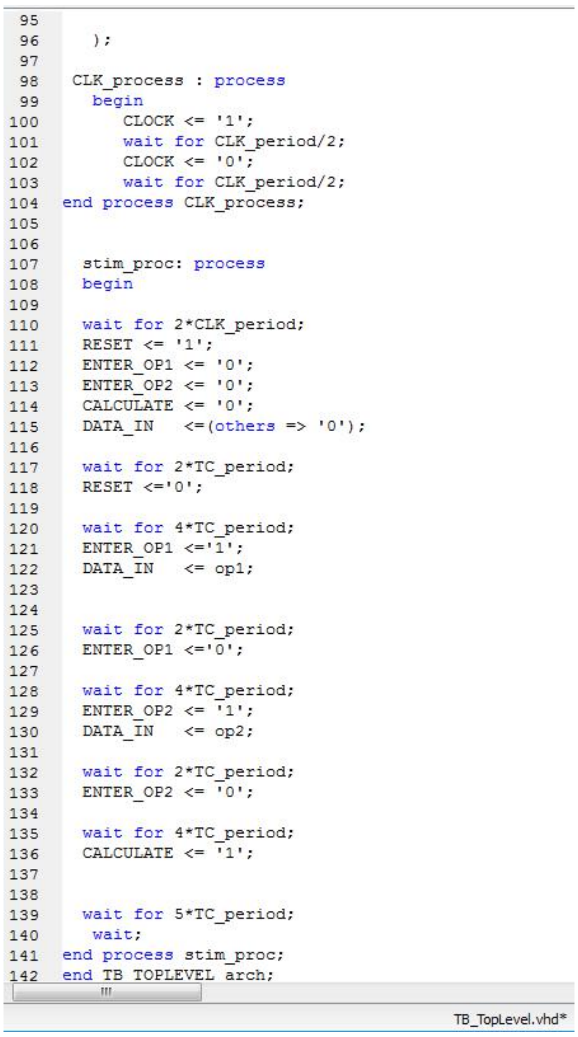
```vhdl
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
LIBRARY UNISIM;
USE UNISIM.Vcomponents.ALL;
use std.textio.all;
use ieee.std_logic_textio.all;
use IEEE.std_logic_signed.all;




entity TB_TOPLEVEL_intf is
end TB_TOPLEVEL_intf;

architecture TB_TOPLEVEL_arch of TB_TOPLEVEL_intf is


   COMPONENT TopLevel
   PORT(  RESET   :  IN  STD_LOGIC;
          CLOCK   :  IN  STD_LOGIC;
          ENTER_OP1   :  IN  STD_LOGIC;
          ENTER_OP2   :  IN  STD_LOGIC;
          CALCULATE   :  IN  STD_LOGIC;

          DATA_IN   :  IN  STD_LOGIC_VECTOR (7 DOWNTO 0);
          COMM_ONES   :  OUT  STD_LOGIC;
          COMM_DECS   :  OUT  STD_LOGIC;
          COMM_HUNDREDS   :  OUT  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;
          DP   :  OUT  STD_LOGIC;
       ACC_DATA_OUT_BUS  : OUT STD_LOGIC_VECTOR(7 DOWNTO 0)
       );



   END COMPONENT;


  signal op1 : STD_LOGIC_VECTOR(7 DOWNTO 0):="00000001";
  signal op2 : STD_LOGIC_VECTOR(7 DOWNTO 0):="00000001";
```

TB_TopLevel.vhd



```vhdl

  signal RESET   :     STD_LOGIC;
  signal CLOCK   :     STD_LOGIC;
  signal ENTER_OP1   :     STD_LOGIC;
  signal ENTER_OP2   :     STD_LOGIC;
  signal CALCULATE   :     STD_LOGIC;
  signal DATA_IN   :    STD_LOGIC_VECTOR (7 DOWNTO 0);
  signal COMM_ONES   :     STD_LOGIC;
  signal COMM_DECS   :     STD_LOGIC;
  signal COMM_HUNDREDS   :     STD_LOGIC;
  signal SEG_A   :     STD_LOGIC;
  signal SEG_B   :     STD_LOGIC;
  signal SEG_C   :     STD_LOGIC;
  signal SEG_D   :     STD_LOGIC;
  signal SEG_E   :     STD_LOGIC;
  signal SEG_F   :     STD_LOGIC;
  signal SEG_G   :     STD_LOGIC;
  signal DP      :     STD_LOGIC;
  signal ACC_DATA_OUT_BUS :  STD_LOGIC_VECTOR(7 DOWNTO 0);


   constant CLK_period: time := 1 ns;
   constant TC_period: time := 65536 ns;


BEGIN

   UUT: TopLevel
  PORT MAP(
    RESET => RESET,
    CLOCK => CLOCK,
    ENTER_OP1 => ENTER_OP1,
    ENTER_OP2 => ENTER_OP2,
    CALCULATE => CALCULATE,
    DATA_IN => DATA_IN,
    COMM_ONES => COMM_ONES,
    COMM_DECS => COMM_DECS,
    COMM_HUNDREDS => COMM_HUNDREDS,
    SEG_A => SEG_A,
    SEG_B => SEG_B,
    SEG_C => SEG_C,
    SEG_D => SEG_D,
    SEG_E => SEG_E,
    SEG_F => SEG_F,
    SEG_G => SEG_G,
    DP => DP,
    ACC_DATA_OUT_BUS=>ACC_DATA_OUT_BUS
```

TB_TopLevel.vhd*



```

   );

 CLK_process : process
   begin
      CLOCK <= '1';
      wait for CLK_period/2;
      CLOCK <= '0';
      wait for CLK_period/2;
end process CLK_process;


  stim_proc: process
  begin

  wait for 2*CLK_period;
  RESET <= '1';
  ENTER_OP1 <= '0';
  ENTER_OP2 <= '0';
  CALCULATE <= '0';
  DATA_IN   <=(others => '0');

  wait for 2*TC_period;
  RESET <='0';

  wait for 4*TC_period;
  ENTER_OP1 <='1';
  DATA_IN   <= op1;


  wait for 2*TC_period;
  ENTER_OP1 <='0';

  wait for 4*TC_period;
  ENTER_OP2 <= '1';
  DATA_IN   <= op2;

  wait for 2*TC_period;
  ENTER_OP2 <= '0';

  wait for 4*TC_period;
  CALCULATE <= '1';


  wait for 5*TC_period;
   wait;
end process stim_proc;
end TB_TOPLEVEL_arch;
```

TB_TopLevel.vhd*

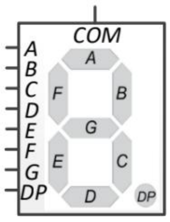
7) Запускаю симулятор для файла TestBenchTopLevel.vhd та перевіряю правильність роботи схеми:

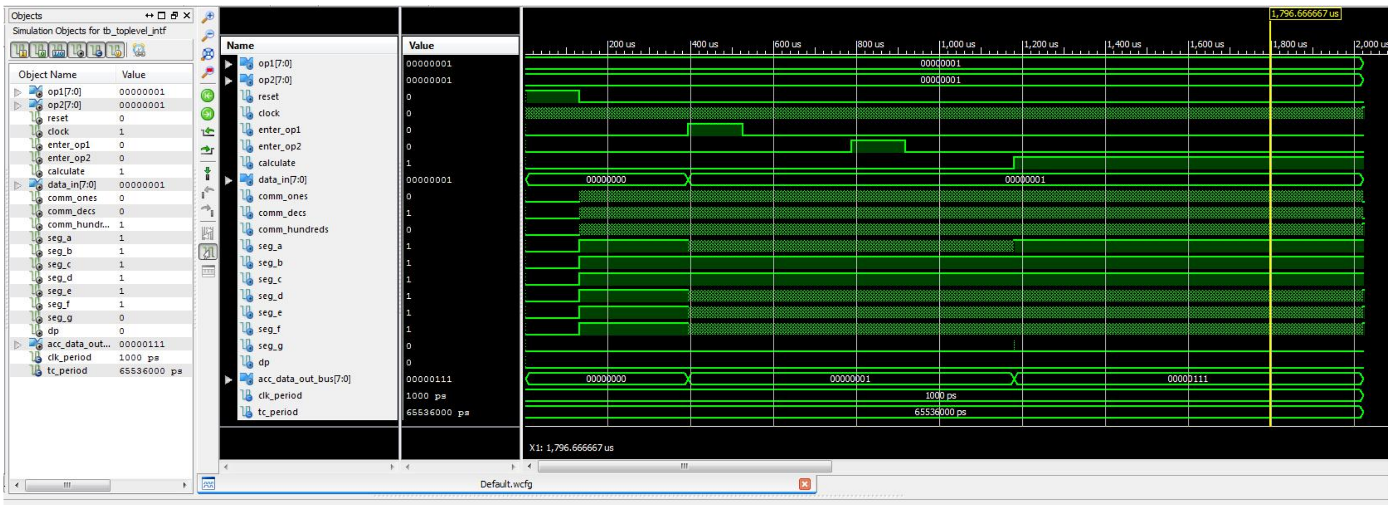
Моя формула:

7	$((OP1 << 2) - OP2) + 4$

OP1 = 1
OP2 = 1
1) 1 << 2 = 4
2) 4 - 1 = 3
3) 3 + 4 = 7

Hundred s= 0	A B C D E F $\overline{G}$
Decs = 0	A B C D E F $\overline{G}$
Ones = 7	A B C $\overline{D}$ $\overline{E}$ $\overline{F}$ $\overline{G}$





Objects

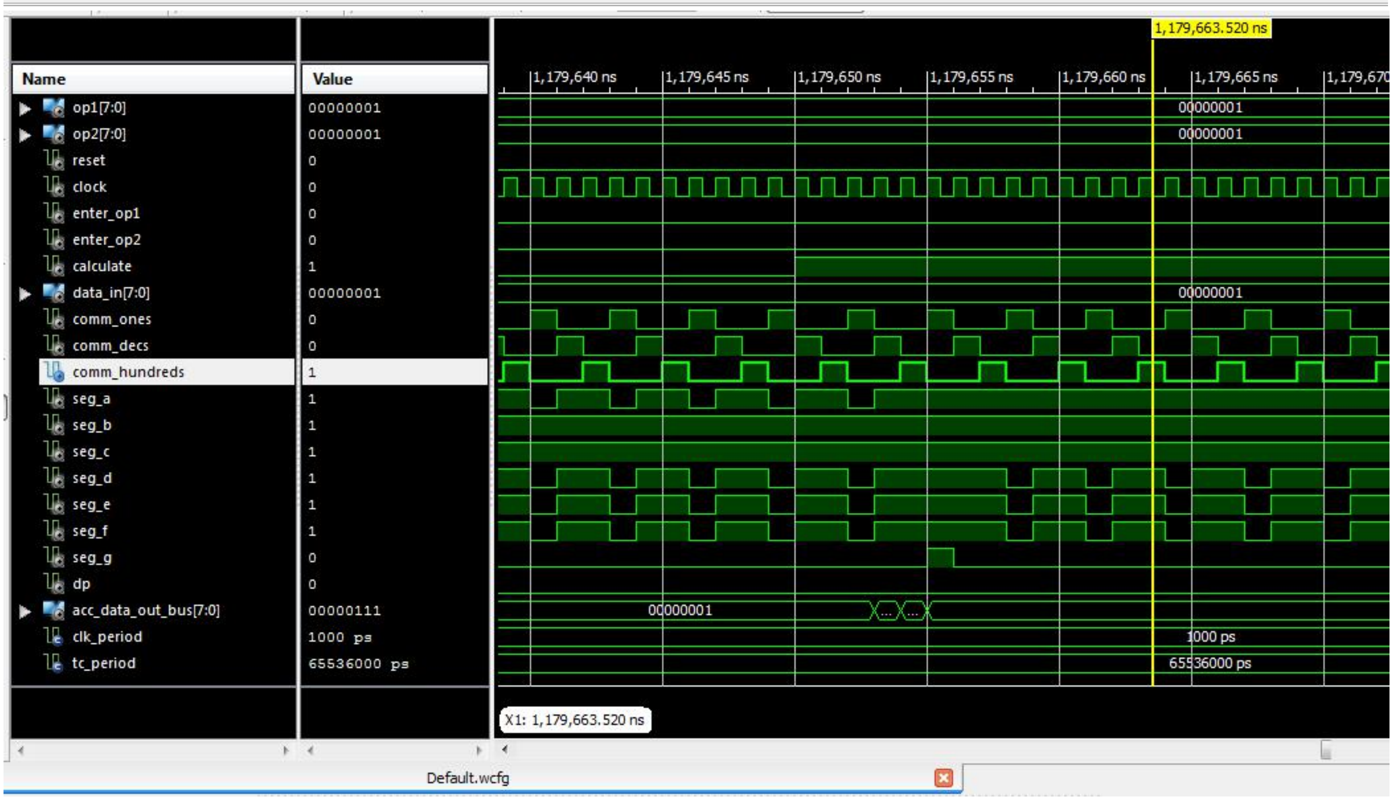
Simulation Objects for tb_toplevel_intf

Object Name	Value
op1[7:0]	00000001
op2[7:0]	00000001
reset	0
clock	1
enter_op1	0
enter_op2	0
calculate	1
data_in[7:0]	00000001
comm_ones	0
comm_decs	0
comm_hundr...	1
seg_a	1
seg_b	1
seg_c	1
seg_d	1
seg_e	1
seg_f	1
seg_g	0
dp	0
acc_data_out...	00000111
clk_period	1000 ps
tc_period	65536000 ps

Name	Value
op1[7:0]	00000001
op2[7:0]	00000001
reset	0
clock	0
enter_op1	0
enter_op2	0
calculate	1
data_in[7:0]	00000001
comm_ones	0
comm_decs	1
comm_hundreds	0
seg_a	1
seg_b	1
seg_c	1
seg_d	1
seg_e	1
seg_f	1
seg_g	0
dp	0
acc_data_out_bus[7:0]	00000111
clk_period	1000 ps
tc_period	65536000 ps

X1: 1,796.666667 us

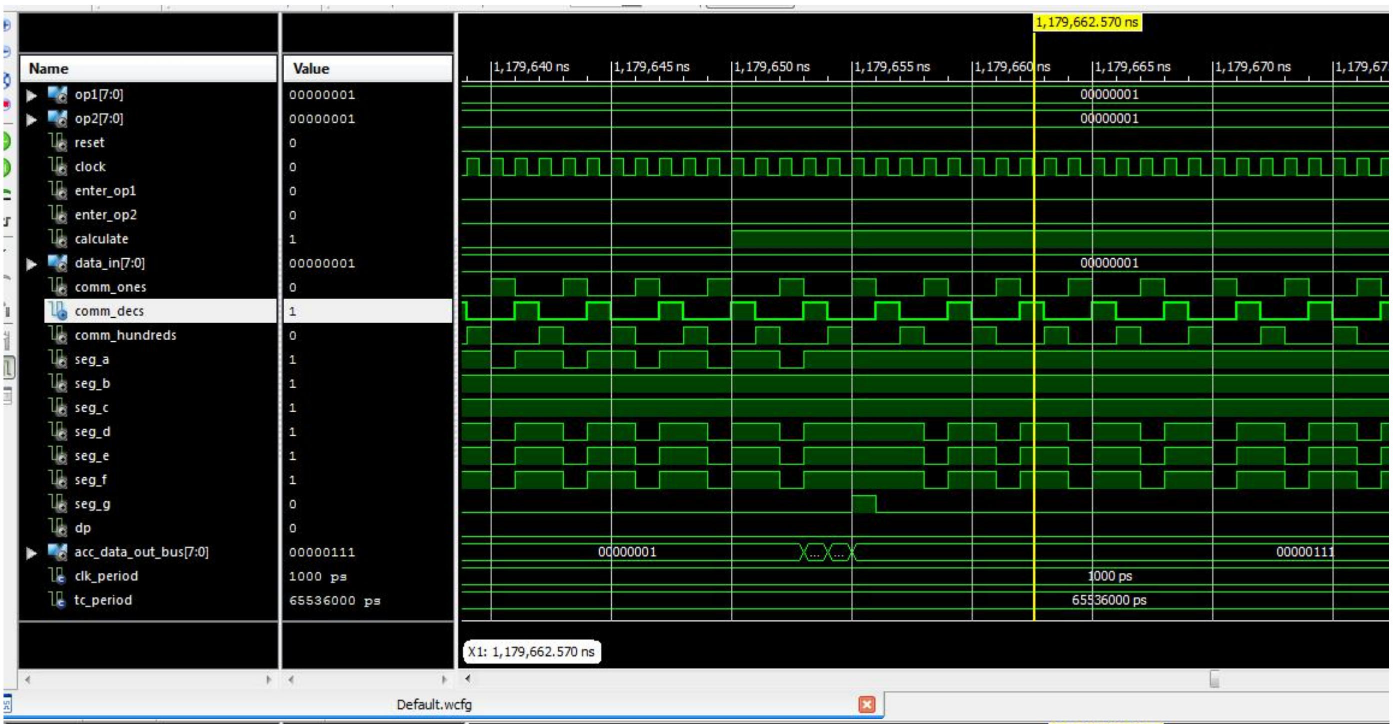
Default.wcfg



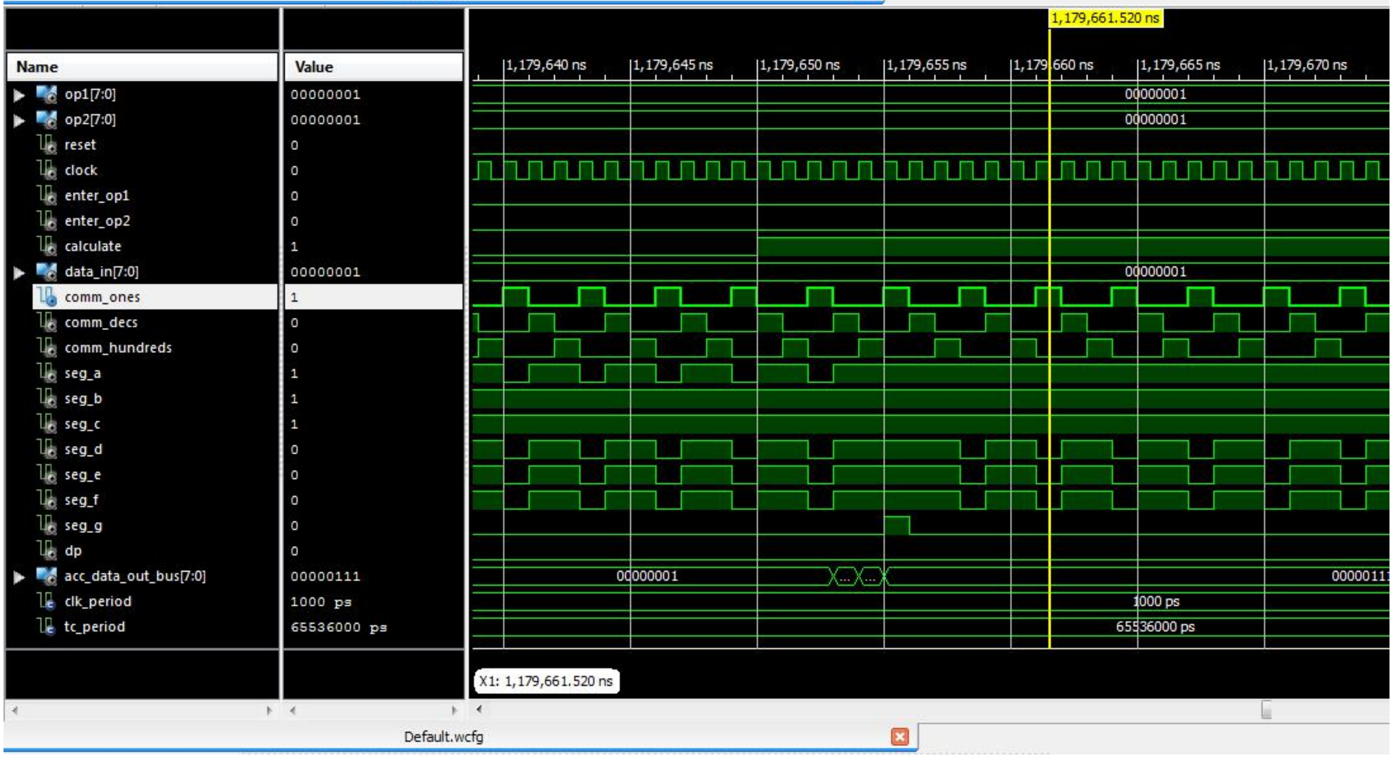
Name	Value
op1[7:0]	00000001
op2[7:0]	00000001
reset	0
clock	0
enter_op1	0
enter_op2	0
calculate	1
data_in[7:0]	00000001
comm_ones	0
comm_decs	0
comm_hundreds	1
seg_a	1
seg_b	1
seg_c	1
seg_d	1
seg_e	1
seg_f	1
seg_g	0
dp	0
acc_data_out_bus[7:0]	00000111
clk_period	1000 ps
tc_period	65536000 ps

X1: 1,179,663.520 ns

Default.wcfg

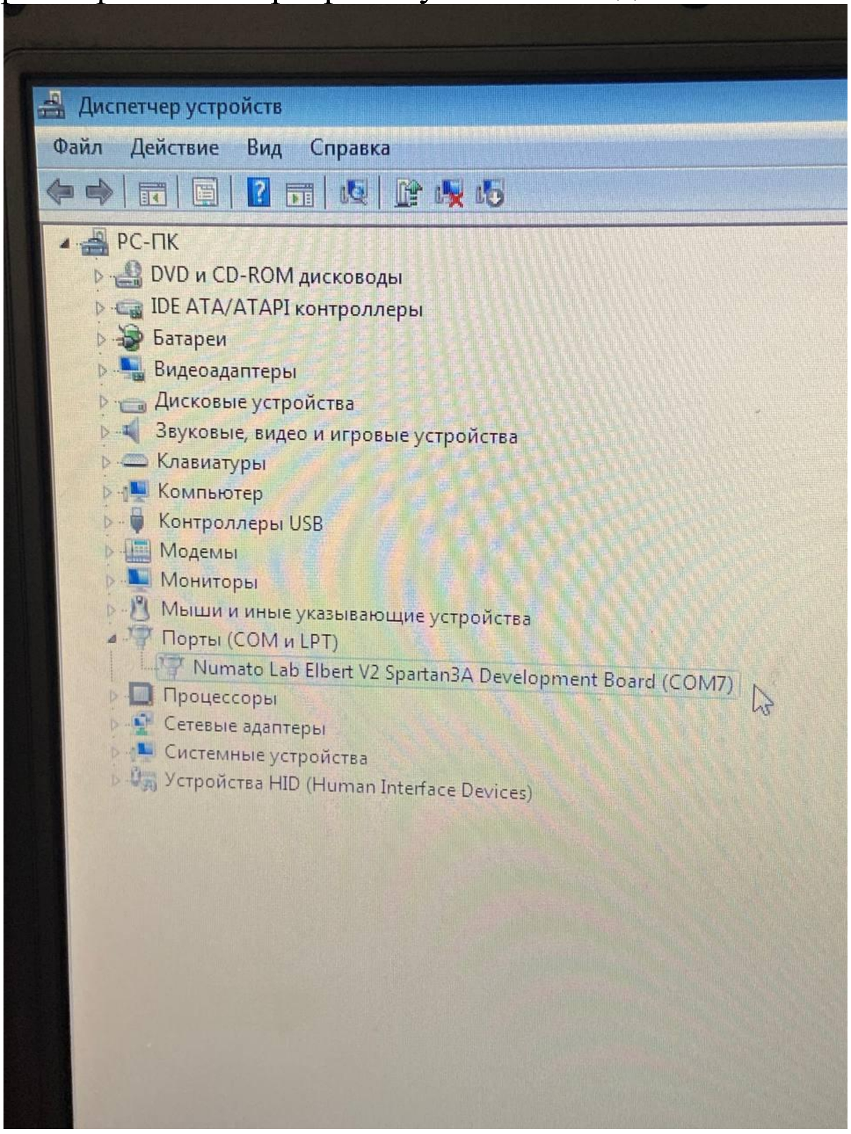


Name	Value
op1[7:0]	00000001
op2[7:0]	00000001
reset	0
clock	0
enter_op1	0
enter_op2	0
calculate	1
data_in[7:0]	00000001
comm_ones	0
comm_decs	1
comm_hundreds	0
seg_a	1
seg_b	1
seg_c	1
seg_d	1
seg_e	1
seg_f	1
seg_g	0
dp	0
acc_data_out_bus[7:0]	00000111
clk_period	1000 ps
tc_period	65536000 ps



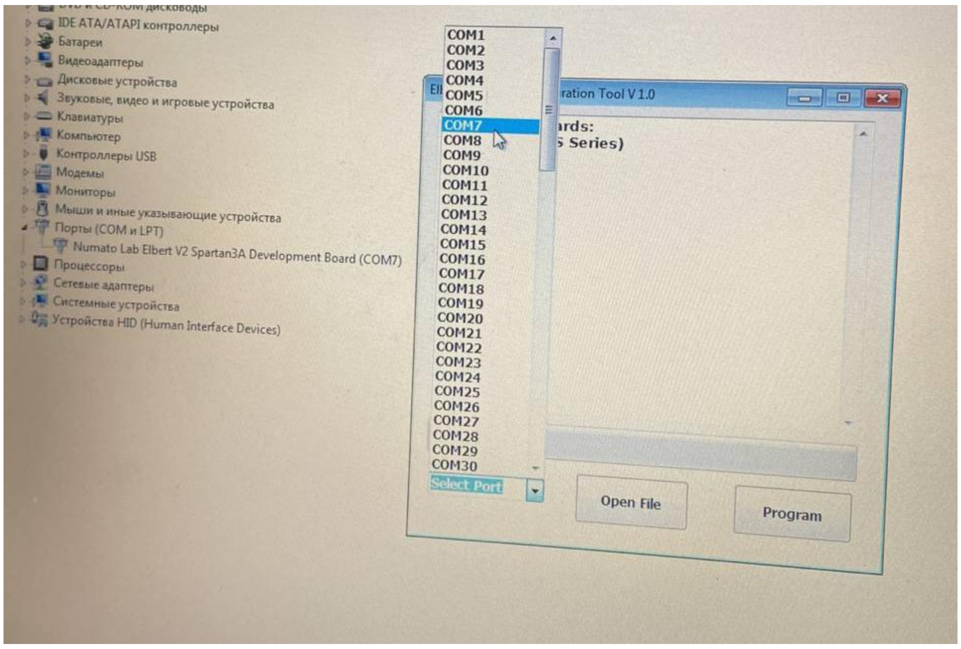
Name	Value
op1[7:0]	00000001
op2[7:0]	00000001
reset	0
clock	0
enter_op1	0
enter_op2	0
calculate	1
data_in[7:0]	00000001
comm_ones	1
comm_decs	0
comm_hundreds	0
seg_a	1
seg_b	1
seg_c	1
seg_d	0
seg_e	0
seg_f	0
seg_g	0
dp	0
acc_data_out_bus[7:0]	00000111
clk_period	1000 ps
tc_period	65536000 ps

8) Генерую бінарний файл та запрограмовую ним стенд:


Диспетчер устройств

Файл Действие Вид Справка

- PC-ПК
  - DVD и CD-ROM дисководы
  - IDE ATA/ATAPI контроллеры
  - Батареи
  - Видеоадаптеры
  - Дисковые устройства
  - Звуковые, видео и игровые устройства
  - Клавиатуры
  - Компьютер
  - Контроллеры USB
  - Модемы
  - Мониторы
  - Мыши и иные указывающие устройства
  - Порты (COM и LPT)
    - Numato Lab Elbert V2 Spartan3A Development Board (COM7)
  - Процессоры
  - Сетевые адаптеры
  - Системные устройства
  - Устройства HID (Human Interface Devices)


- IDE ATA/ATAPI контроллеры
- Батареи
- Видеоадаптеры
- Дисковые устройства
- Звуковые, видео и игровые устройства
- Клавиатуры
- Компьютер
- Контроллеры USB
- Модемы
- Мониторы
- Мыши и иные указывающие устройства
- Порты (COM и LPT)
  - Numato Lab Elbert V2 Spartan3A Development Board (COM7)
- Процессоры
- Сетевые адаптеры
- Системные устройства
- Устройства HID (Human Interface Devices)

El ration Tool V 1.0

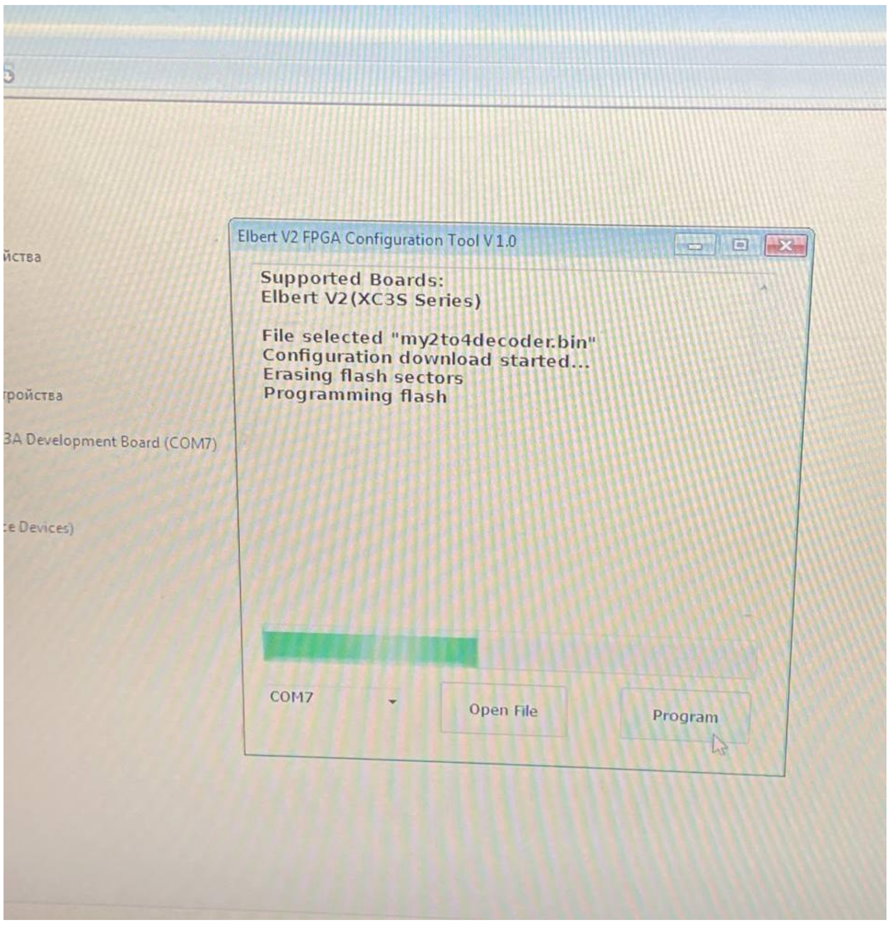
rds:
Series)

COM1
COM2
COM3
COM4
COM5
COM6
COM7
COM8
COM9
COM10
COM11
COM12
COM13
COM14
COM15
COM16
COM17
COM18
COM19
COM20
COM21
COM22
COM23
COM24
COM25
COM26
COM27
COM28
COM29
COM30

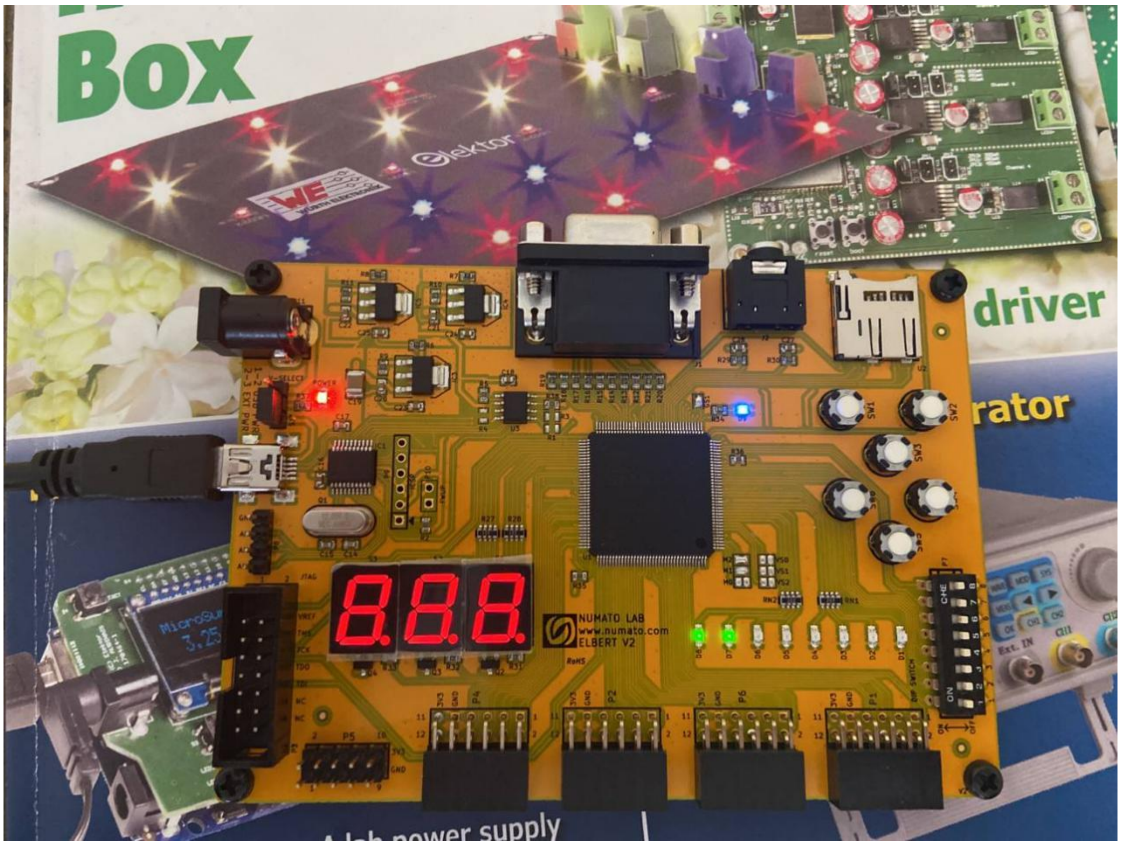
Select Port

Open File

Program



```
Supported Boards:
Elbert V2(XC3S Series)

File selected "my2to4decoder.bin"
Configuration download started...
Erasing flash sectors
Programming flash
```



Висновок: у цій лабораторній роботі я реалізував цифровий автомат, що обчислює вираз згідно заданого варіанту.