# **Лабораторна робота №2.2**

**Тема:** Розробка програмного проекту криптографічного процесора для захисту текстової інформації в середовищі Active HDL 5.1

**Мета роботи :**

1. Вивчити прийоми проектування програмних засобів захисту інформації на мові VHDL.
2. Вивчити методологію створення VHDL проектів в текстовому редакторові САПР Active HDL 5.1.
3. Вивчити методологію моделювання роботи VHDL проекту за допомогою редактора тимчасових діаграм.

**Короткі теоретичні відомості**

Середовище **Active HDL** має широкий спектр інструментальних засобів що підтримують програмні методи моделювання проектів електронних пристроїв. У ній підтримується безліч способів для формування вхідних тестуючих дій для проведення випробувань і моделювання роботи проектованих пристроїв. Є також можливість реалізувати процес моделювання шляхом створення спеціальних випробувальних стендів. Середовище **Active HDL** забезпечене також зручними засобами для відладки VHDL проектів цифрових пристроїв. Вона містить також повчальні програми по основах її використання в процесах проектування і моделювання проектів.

Процес створення комплексного проекту електронного пристрою умовно можна розділити на два етапи:

* Етап створення проекту;
* Етап моделювання і відладки проекту.

Маючи у своєму розпорядженні проекти вище вказаних модулів, вимагається перевірити роботу проекту на предмет виявлення в нім помилок функціонального плану. Така перевірка виконується зазвичай шляхом моделювання роботи проекту.

**Завдання на виконання роботи :** розробити програмний проект криптографічного процесора для захисту текстової інформації в середовищі Active HDL 5.1

Криптографічний процесор повинен мати такі характеристики:

* Об'єм шифрованої інформації : 24 байти.
* Форма представлення шифрованої інформації : символьна;
* Форма представлення закодованої інформації : десяткові числа.
* Форма представлення декодованої інформації : ASCII код;
* Форма представлення шифруючої інформації(ключів шифрування / дешифрування) : символьна;
* Шифруюча операція: ділення;
* Дешифруюча операція: зворотна до тієї, що шифрує.
* Шифрований текст: *аналогічно з 1-м процесором*.
* Кількість шифруючих / дешифруючих ключів : 8.
* Початкова шифруюча інформація: прізвище студента відповідно до заданої форми представлення.
* Період  тактового сигналу CLK в процесі моделювання роботи процесора : Група 431/1: 30 нс;
* Часовий інтервал моделювання : .
* Спосіб подання стимулюючих дій при моделюванні: послідовний(побайтний) відповідно до періоду  тактового сигналу CLK.
* Спосіб формування стимулюючих дій : «ручний» з допомогою ДО Stimulators.
* Діалог для формування стимулюючих дій по їх видах:
* Тактовий сигнал CLK : діалог CLK;
* Кодів шифрованої інформації і кодів ключів : формульный  діалог.

**Хід роботи**

1. Запишіть інформацію, яка шифруватиметься і яка використовуватиметься як ключ шифрування.
2. Створіть проект за прикладом, описаному в лабораторній роботі №2.1.
3. Запишіть код програми, яка шифруватиме текстову інформацію. Відкомпілюйте код, при наличи помилок - виправіть їх. Приклад коду :

 library IEEE;

use IEEE.STD LOGIC 1164.all;

entity lr22 is

 port( clk : in STD \_ LOGIC;

 text : in CHARACTER;

 kluch : in CHARACTER;

 kod : out INTEGER;

 decod : out INTEGER );

end lr22;

architecture lr22 \_ arch of lr22 is

begin

 process(clk)

 variable text \_ int, kluch \_ int, tmp, tmp2, tmp3 : integer ;

 variable conv \_ text \_ vec, conv \_ kod \_ vec : std \_ logic \_ vector(7 downto 0);

 variable deshifr \_ int : integer;

 variable shifrkod : integer;

function CONV \_ ch \_ int(X: character) return std \_ logic \_ vector is

 variable RESULT : std \_ logic \_ vector(7 downto 0);

 variable TEMP : integer := CHARACTER'POS(X);

begin

 for i in RESULT'REVERSE \_ RANGE loop

 case TEMP mod 2 is

 when 0 => RESULT(i) := '0';

 when 1 => RESULT(i) := '1';

 when others => null;

 end case;

 TEMP := TEMP/2;

 end loop;

return RESULT;

end CONV \_ ch \_ int;

------------------------------------------------------------------------------

-- From STD \_ LOGIC \_ VECTOR to unsigned INTEGER converter

------------------------------------------------------------------------------

function CONV \_ vec \_ int(X :STD \_ LOGIC \_ VECTOR) return INTEGER is

 constant XMAP :INTEGER :=0;

 variable RESULT :INTEGER :=0;

 begin

 for i in X'RANGE loop

 RESULT:=RESULT\*2;

 case X(i) is

 when '0' | 'L' => null;

 when '1' | 'H' => RESULT :=RESULT+1;

 when others => RESULT :=RESULT+XMAP;

 end case;

 end loop;

 return RESULT;

 end CONV \_ vec \_ int;

---------------------------------------------------------------------

begin

 if clk'event and clk = '1' then

 conv \_ text \_ vec := CONV \_ ch \_ int( text); -- Char to vec

 conv \_ kod \_ vec := CONV \_ ch \_ int(kluch);

 text \_ int := CONV \_ vec \_ int( conv \_ text \_ vec); --vec to init

 kluch \_ int := CONV \_ vec \_ int( conv \_ kod \_ vec);

 tmp := text \_ int;

 shifrkod := tmp / kluch \_ int; -- coding

 tmp2 := shifrkod;

 kod <= tmp2; --init

 deshifr \_ int := shifrkod \* kluch \_ int; --decoding

 tmp3:=deshifr \_ int;

 decod <= tmp3;

 end if;

end process;

end lr22 \_ arch;

1. Створіть нову часову діаграму. Перемістіть в неї усі необхідні порти. Просимулируйте проект. Назначте для портів потрібні значення. Запустіть проект. Приклад часової діаграми вказаний на рисунку 1.



Рис.1