Міністерство освіти і науки

Національний університет “Львівська політехніка”

Кафедра ЕОМ



**Звіт**

з лабораторної роботи №4

з дисципліни: “ Комп’ютерні системи ”

на тему:

“ Аналіз програмної моделі процесу роботи арифметичного конвеєра, ч.2.”

Виконала: ст. гр. КІ-32

Фоменко Г. Ю.

Перевірив: Козак Н.Б.

Львів 2020

**Мета лабораторної роботи:**

Навчитись здійснювати аналіз програмних моделей комп’ютерних систем, виконаних на мові System C.

**Завдання:**

1. Проаналізувати склад програмної моделі арифметичного конвеєра, (програма PIPE), яка виконана на мові System C.

2. Здійснити модернізацію функцій або параметрів арифметичного конвеєра (див. лабораторну роботу № 3), шляхом під’єднання розроблених модулів S1 та S2 (див. лабораторну роботу № 2). Порядок та тип з’єднання мають бути обгрунтовані, можливо розробка буферних або додаткових модулів з метою надавання нових властивостей тестувальній моделі.

3. Накреслити кінцеву структурну схему отриманої програмної моделі.

4. Навести стисло код та внесені нові зміни.

5. Навести результати тестування та використання програмної моделі.

**Варіант – 19**

**S1-** 2 вихідних порта power, f1

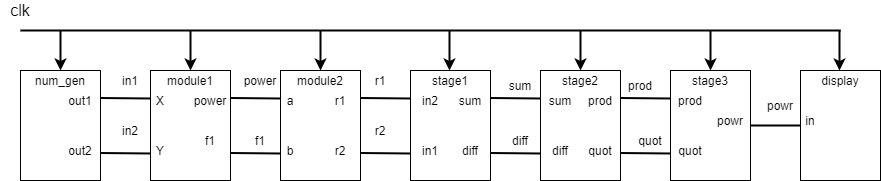
power = Pow(X, Y) ;

f1 = X \* 1 + X \* 2 + X \* 3 + …. + X \* Y;

**S2-** r1 = OR (a, b) , r2 = NOT( OR (a, b));

**Послідовність виконання роботи**

Аналіз програмної моделі:



Лістинг програми з додатковими функціональними блоками:

#include "systemc.h"

#include "stage1.h"

#include "stage2.h"

#include "stage3.h"

#include "display.h"

#include "numgen.h"

#define NS \* 1e-9

struct module1 : sc\_module {

sc\_in<double> in1, in2; // input port 1

sc\_in<bool> clk; // clock

sc\_out<double> power, f1;

void do\_add()

{

double X, Y;

double temp = 1;

X = in1.read();

Y = in2.read();

power.write(pow(X, Y));

for(double i = 1; i <= Y;i++)

{

temp += X\*i;

}

f1.write(temp);

};

//Constructor

SC\_CTOR(module1) {

SC\_METHOD(do\_add); // declare print as SC\_METHOD and

sensitive\_pos << clk; // make it sensitive to positive clock edge

}

public:

};

struct module2 : sc\_module {

sc\_in<double> power, f1; // input port 1

sc\_in<bool> clk; // clock

sc\_out<double> r1, r2;

void do\_next()

{

int a, b;

double temp = 1;

a = power.read();

b = f1.read();

r1.write(a|b);

r2.write(~(a|b));

};

//Constructor

SC\_CTOR(module2) {

SC\_METHOD(do\_next); // declare print as SC\_METHOD and

sensitive\_pos << clk; // make it sensitive to positive clock edge

}

public:

};

int sc\_main(int ac, char\* av[])

{

//Signals

sc\_signal<double> in1;

sc\_signal<double> in2;

sc\_signal<double> r1;

sc\_signal<double> r2;

sc\_signal<double> power;

sc\_signal<double> f1;

sc\_signal<double> sum;

sc\_signal<double> diff;

sc\_signal<double> prod;

sc\_signal<double> quot;

sc\_signal<double> powr;

//Clock

sc\_signal<bool> clk;

numgen N("numgen"); //instance of `numgen' module

N(in1, in2, clk); //Positional port binding

// in1 = 4;

//in2 = 2;

module1 M1("module1");

M1(in1, in2, clk, power, f1);

module2 M2("module2");

M2(power, f1, clk, r1, r2);

// display D("display"); //instance of `display' module

//D(mod2\_1,clk, mod2\_2);

stage1 S1("stage1"); //instance of `stage1' module

//Named port binding

S1.in1(r1);

S1.in2(r2);

S1.sum(sum);

S1.diff(diff);

S1.clk(clk);

stage2 S2("stage2"); //instance of `stage2' module

S2(sum, diff, prod, quot, clk); //Positional port binding

stage3 S3("stage3"); //instance of `stage3' module

S3(prod, quot, powr, clk); //Positional port binding

display D("display"); //instance of `display' module

D(powr, clk, diff); //Positional port binding

sc\_initialize(); //Initialize simulation

for (int i = 0; i < 20; i++) {

clk.write(1);

sc\_cycle(10 NS);

clk.write(0);

sc\_cycle(10 NS);

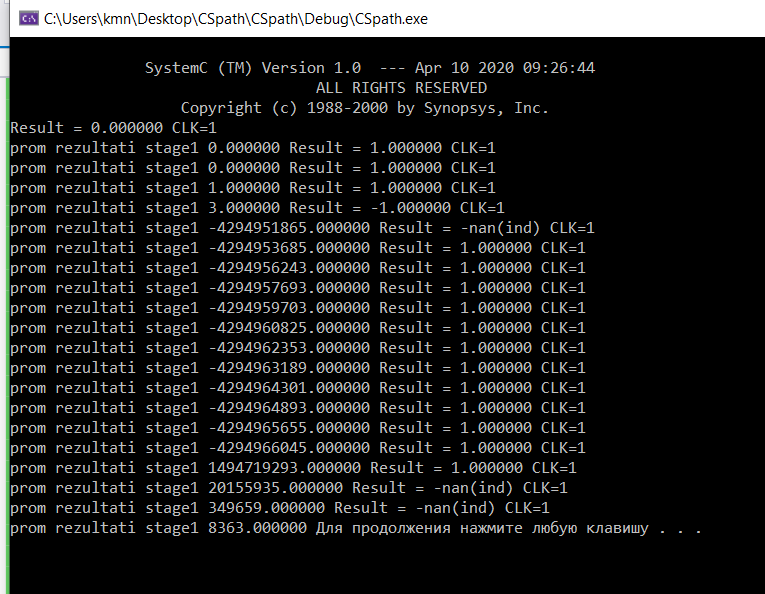
}

system("pause");

return 0;

}

Результат виконання програми



**Висновок:** виконуючи дану лабораторну роботу навчилася здійснювати аналіз програмних моделей комп’ютерних систем, виконаних на мові System C.