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 ”

Виконала: ст. гр. КІ-32

Селебинка Д.В.

Перевірив: викладач

Козак Н.Б.

Львів – 2020

**МЕТА 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.

4. Оформити звіт.

**Виконання:**

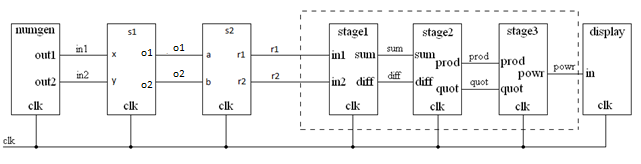


Рис. 1 Модель арифметичного конвеєра з внесеними змінами

**Лістинг програми:**

**Numgen.cpp**

#include "systemc.h"

#include "numgen.h"

// definition of the `generate' method

void numgen::generate()

{

static double a = 20;

static double b = 30;

a -= 1;

b -= 1.5;

out1.write(a);

out2.write(b);

} // end of `generate' method

**Main.cpp**

#include "systemc.h"

#include "stage1.h"

#include "stage2.h"

#include "stage3.h"

#include "display.h"

#include "numgen.h"

#include "s1.h"

#include "s2.h"

#define NS \* 1e-9

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

{

//Signals

sc\_signal<double> in1;

sc\_signal<double> in2;

sc\_signal<double> sum;

sc\_signal<double> diff;

sc\_signal<double> prod;

sc\_signal<double> quot;

sc\_signal<double> powr;

sc\_signal<double> o1;

sc\_signal<double> r1;

sc\_signal<double> r2;

sc\_signal<double> o2;

//Clock

//<TRACE>

sc\_clock clk("CLOCK", 10, 0.5, 0);

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

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

S1 s\_1("s1");

s\_1(in1, in2, o1, o2, clk);

S2 s\_2("s2");

s\_2(o1, o2, r1, r2, clk);

stage1 S\_1("stage1"); //instance of `stage1' module

//Named port binding

S\_1.in1(r1);

S\_1.in2(r2);

S\_1.sum(sum);

S\_1.diff(diff);

S\_1.clk(clk);

stage2 S\_2("stage2"); //instance of `stage2' module

S\_2(sum, diff, prod, quot, clk); //Positional port binding

stage3 S\_3("stage3"); //instance of `stage3' module

S\_3(prod, quot, powr, clk); //Positional port binding

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

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

//<TRACE>

sc\_trace\_file \*tf = sc\_create\_vcd\_trace\_file("pipe");

sc\_trace(tf, clk, "CLK");

sc\_trace(tf, o1, "o1");

sc\_trace(tf, o2, "o2");

sc\_trace(tf, r1, "r1");

sc\_trace(tf, r2, "r2");

sc\_trace(tf, in1, "IN1");

sc\_trace(tf, in2, "IN2");

sc\_trace(tf, powr, "powr");

//</TRACE>

//<TRACE>

sc\_start(clk, 500);

//</TRACE>

return 0;

}

**S1.h**

#ifndef TEST\_SYSTEMC\_S1\_H

#define TEST\_SYSTEMC\_S1\_H

#include "systemc.h"

#include "cmath"

#include "s2.h"

SC\_MODULE(S1) {

sc\_in<double> x;

sc\_in<double> y;

sc\_out<double> o1;

sc\_out<double> o2;

sc\_in<bool> clk;

SC\_CTOR(S1) {

SC\_METHOD(calc\_o2);

sensitive << x << y;

sensitive << clk;

SC\_METHOD(calc\_o1);

sensitive << x << y;

sensitive << clk;

}

void calc\_o2() {

double x\_val = x.read();

double y\_val = y.read();

double acc = 0;

for (double i = 1; i <= y\_val; i++) {

acc += x\_val\*i;

}

o2.write(acc);

}

void calc\_o1() {

double x\_val = x.read();

double y\_val = y.read();

o1.write(pow(x\_val, y\_val));

}

};

#endif //TEST\_SYSTEMC\_S1\_H

#pragma once

**S2.h**

#ifndef TEST\_SYSTEMC\_S2\_H

#define TEST\_SYSTEMC\_S2\_H

#include "systemc.h"

SC\_MODULE(S2) {

sc\_in<double> a;

sc\_in<double> b;

sc\_out<double> r1;

sc\_out<double> r2;

sc\_in<bool> clk;

SC\_CTOR(S2) {

SC\_METHOD(calc\_r1);

sensitive << a << b;

sensitive << clk;

SC\_METHOD(calc\_r2)

sensitive << a << b;

sensitive << clk;

}

void calc\_r1() {

double a\_val = a.read();

r1.write(a\_val \* a\_val);

}

void calc\_r2() {

double b\_val = b.read();

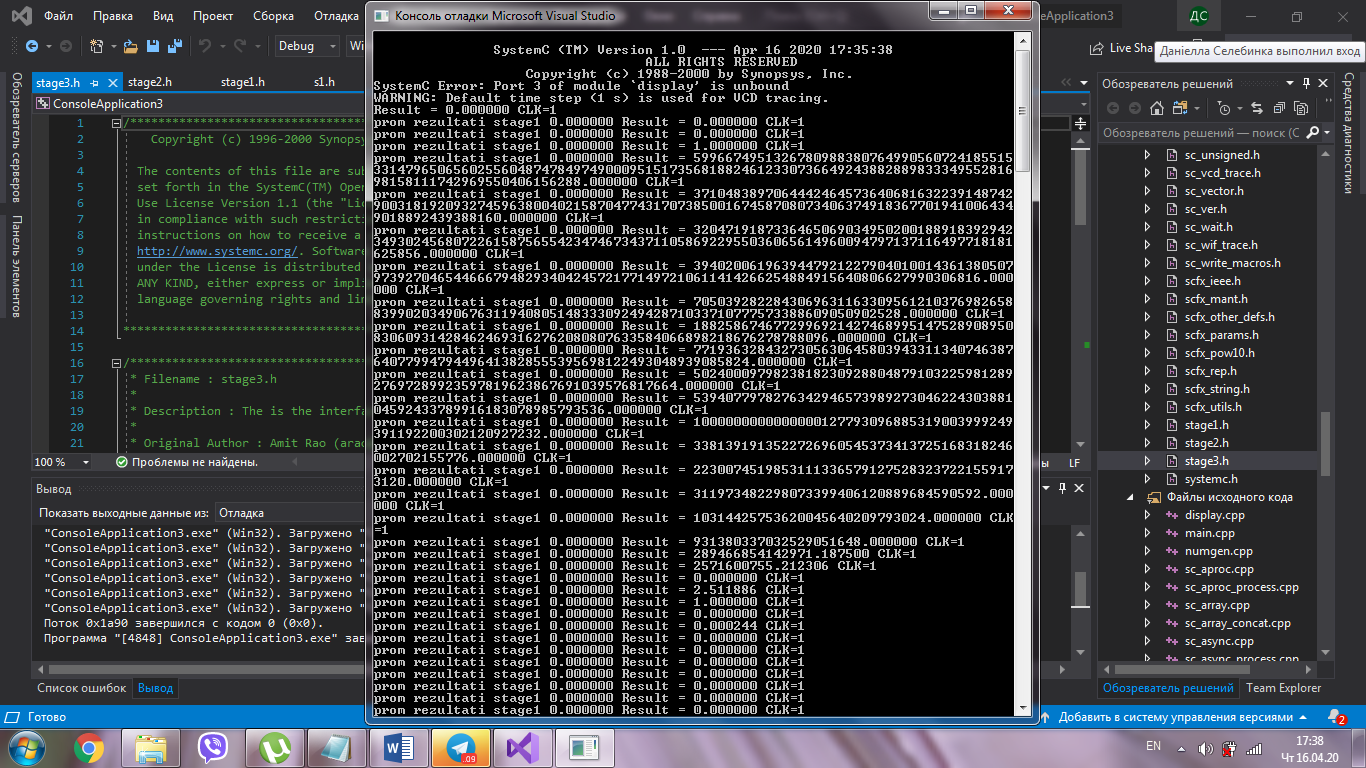
r2.write(b\_val \* b\_val);

}

};

#endif //TEST\_SYSTEMC\_S2\_H

Скрін виконання :



**Висновок:** на даній лабораторній роботі я навчилася здійснювати аналіз програмних моделей комп’ютерних систем, виконаних на мові System C.