       Министерство науки и высшего образования Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

“НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО”

Факультет Программной инженерии и компьютерной техники



Отчет

По лабораторной работе № 2

По предмету:  Системы на кристалле

Вариант 1

Студенты:

Андрейченко Леонид Вадимович

Степанов Михаил Алексеевич

Группа P34301

Преподаватель:

Быковский Сергей Вячеславович

Санкт-Петербург

2023

Цель работы

Получить базовые навыки использования средств высокоуровневого синтеза в процессе проектирования СнК.

Задание

1. Спроектировать и описать функциональность аппаратного ускорителя для алгоритма из лабораторной работы №1 на языках C/С++, пригодную для синтеза в аппаратный блок.

2. Провести синтез аппаратного ускорителя.

3. Разработать тестовое окружение для проверки функциональности синтезированного аппаратного ускорителя.

4. Оценить следующие характеристики:

4.1. Время выполнения алгоритма при частоте тактового сигнала в 100 МГц.

4.2. Число занимаемых ресурсов ПЛИС (XC7A100T-1CSG324C).

4.3. Время и занимаемые ресурсы ПЛИС с использованием следующих оптимизаций: раскрутка циклов, конвейеризация циклов.

Выполнение

Разработанный алгоритм был перенесен в Vivado HLS, и произведен его синтез. Получившийся код:

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

#include "fir.h"

int permutations[7];

int get\_index(int raw, int val) {

return !raw ? 0 : (raw & 1 ? val : get\_index(raw / 2, val + 1));

}

void swap(int i, int j) {

int temp = permutations[i];

permutations[i] = permutations[j];

permutations[j] = temp;

}

int next\_set(int n) {

int j = n - 2;

while (j != -1 && permutations[j] >= permutations[j + 1]) j--;

if (j == -1)

return 0;

int k = n - 1;

while (permutations[j] >= permutations[k]) k--;

swap(j, k);

int l = j + 1, r = n - 1;

while (l<r)

swap(l++, r--);

return 1;

}

int get\_element(int i, int j, int exc, int matrix[7][7], int y[7]) {

return j == exc ? y[i] : matrix[i][j];

}

int count\_inversion(int n) {

int sum = 0;

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

for (int j = i + 1; j < n; j++) {

if (permutations[i] > permutations[j]) sum++;

}

}

return sum;

}

void clear\_perm(){

for(int i = 0; i < 7; i++)

permutations[i] = i;

}

int eval\_elem(int param, int n, int matrix[7][7], int y[7]) {

int res = count\_inversion(n) % 2 ? -1 : 1;

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

res \*= get\_element(i, permutations[i], param, matrix, y);

}

return res;

}

int calc\_det(int param, int n, int matrix[7][7], int y[7]){

clear\_perm();

int res = eval\_elem(param, n, matrix, y);

while(next\_set(n)){

res += eval\_elem(param, n, matrix, y);

}

return res;

}

void calc\_result(int n, int matrix[7][7], int y[7], int result[7]){

int main\_delta = calc\_det(n + 1, n, matrix, y);

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

int delta = calc\_det(i, n, matrix, y);

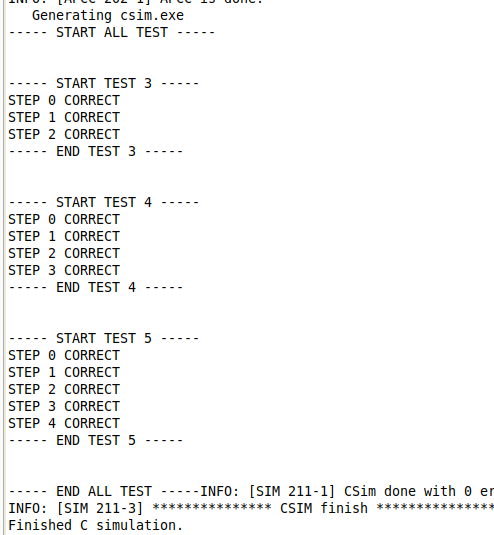
result[i] = delta / main\_delta;

}

}

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

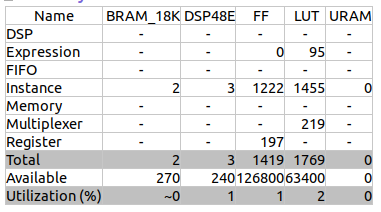
Далее были написаны тесты для проверки корректности работы получившегося аппаратного блока. Результаты тестирования:



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

Стандартная реализация

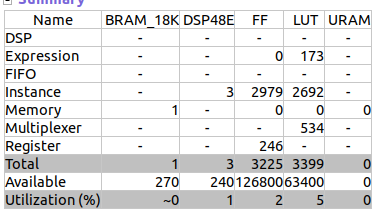




Время выполнения алгоритма 3 =12.3 us, 4 = 74 us, 5 = 554 us

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

pipeline

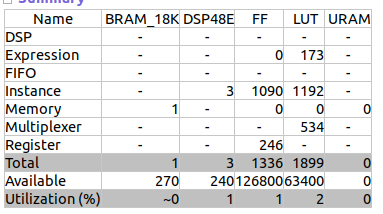




Время выполнения алгоритма 3 = 5.5 us, 4 = 24 us, 5 = 172 us

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

unroll





Время выполнения алгоритма 3 = 8us, 4 = 24 us, 5 = 163 us

Выводы

Самая быстрая реализация – разворачивание цикла.