

Задания

1. Дана матрица A, квадратная, размер NxN. Привести матрицу к треугольному виду с помощью алгоритма Гаусса (тип данных double).
 - a. Теория - <https://web.archive.org/web/20100415172454/http://iproc.ru/parallel-programming/lection-4/>
 - i. Посмотрите рекомендацию по программной реализации
 - b. Практика - <http://hardfire.ru/gauss>
 - c. N при моделировании, оптимизации и Co-simulation – 8.
 - d. N для исследования производительности – 32, 64, 128, 256, 512.
2. Дана матрица A, квадратная, размер NxN. Привести матрицу к треугольному виду с помощью алгоритма Гаусса (тип данных float).
 - a. Теория - <https://web.archive.org/web/20100415172454/http://iproc.ru/parallel-programming/lection-4/>
 - i. Посмотрите рекомендацию по программной реализации
 - b. Практика - <http://hardfire.ru/gauss>
 - c. N при моделировании, оптимизации и Co-simulation – 8.
 - d. N для исследования производительности – 32, 64, 128, 256.
3. Дана матрица A, квадратная, размер NxN. Привести матрицу к треугольному виду с помощью алгоритма Барейса (тип данных int256).
 - a. Теория - <https://web.archive.org/web/20100415172454/http://iproc.ru/parallel-programming/lection-4/>
 - i. Посмотрите рекомендацию по программной реализации.
 - b. N при моделировании, оптимизации и Co-simulation – 8.
 - c. N для исследования производительности – 32, 64, 128, 256.
4. Дана матрица A, квадратная, размер NxN. Привести матрицу к треугольному виду с помощью алгоритма Барейса (тип данных int128).
 - a. Теория - <https://web.archive.org/web/20100415172454/http://iproc.ru/parallel-programming/lection-4/>
 - i. Посмотрите рекомендацию по программной реализации.
 - b. N при моделировании, оптимизации и Co-simulation – 8.
 - c. N для исследования производительности – 32, 64, 128, 256.
5. Дана матрица A, квадратная, размер NxN. Привести матрицу к треугольному виду с помощью алгоритма Барейса (провести исследования оптимизированного решения для типов данных int, long long).
 - a. Теория - <https://web.archive.org/web/20100415172454/http://iproc.ru/parallel-programming/lection-4/>
 - i. Посмотрите рекомендацию по программной реализации.
 - b. N при моделировании, оптимизации и Co-simulation – 8.
 - c. N для исследования производительности – 32, 64, 128, 256.
6. Даны вектора A и B, каждый из N элементов. Надо найти вектор X, из N элементов, каждый элемент которого равен: $X[i] = -B[i]/A[i]$. Т.е. каждый элемент является решением линейного уравнения $a*x+b=0$
 - a. Подсказка: http://hardfire.ru/line_equation
 - b. N при моделировании, оптимизации и Co-simulation – 8.
 - c. N для исследования производительности – 8192, 16384, 32768, 65 536.
 - d. Надо: провести исследования оптимизированного решения для типов данных int, long long, float, double
7. Даны вектора A, B и C, каждый из N элементов, каждый i элемент векторов определяет квадратное уравнение
$$a[i] * x[i]^2 + b[i] * x[i] + c[i] = 0$$
 - a. Надо найти вектора X1 и X2, из N элементов, каждый i элемент которых равен:
$$D[i] = b[i]^2 - 4 * a[i] * c[i]$$
$$X1[i] = (-b[i] - \sqrt{D[i]}) / (2 * a[i])$$
$$X2[i] = (-b[i] + \sqrt{D[i]}) / (2 * a[i])$$
 - b. Подсказка: http://hardfire.ru/sqr_equation
 - c. N при моделировании, оптимизации и Co-simulation – 8.
 - d. N для исследования производительности – 8192, 16384, 32768, 65 536.
 - e. Надо: провести исследования оптимизированного решения для типов данных int, long long, float, double

8. Даны вектора A и B, каждый из N элементов, каждый i элемент A[i] больше соответствующего элемента B[i].
Надо найти вектор X, из N элементов, каждый i элемент которого – является наибольшим общим делителем A[i] и B[i]
- теория: https://foxford.ru/wiki/matematika/algorithm-evklida?utm_referrer=https%3A%2F%2Fyandex.ru%2F
 - практика: <http://hardfire.ru/gcd>
 - N при моделировании, оптимизации и Co-simulation – 8.
 - N для исследования производительности – 8192, 16384, 32768, 65 536.
 - Надо: провести исследования оптимизированного решения для типов данных int, long long, float, double
9. Даны 5 векторов A, B, C, D1, D2 каждый размером N элементов, задающие N пар параллельных плоскостей ($A[i]+B[i]+C[i]+D1[i] = 0$ и $A[i]+B[i]+C[i]+D2[i] = 0$) в пространстве. Надо найти все элементы вектора d: d[i] – расстояние между i-ми парами параллельных плоскостей.
- Теория

Расстояние между плоскостями — равно длине перпендикуляра, опущенного с одной плоскости на другую.

- Если заданы уравнения параллельных плоскостей $Ax + By + Cz + D_1 = 0$ и $Ax + By + Cz + D_2 = 0$, то расстояние между плоскостями можно найти, используя следующую формулу

$$d = \frac{|D_2 - D_1|}{\sqrt{A^2 + B^2 + C^2}}$$


- N при моделировании, оптимизации и Co-simulation – 8.
 - N для исследования производительности – 8192, 16384, 32768, 65 536.
 - Надо: провести исследования оптимизированного решения для типов данных int, long long, float, double
10. Даны 5 векторов A, B, C, x₀, y₀ каждый размером N элементов, задающие N пар: прямая $A[i]x+B[i]y+C[i]=0$ и точка $M_0(x[i]_0, y[i]_0)$ в пространстве. Надо найти все элементы вектора d: d[i] – расстояние между [i] прямой и [i] точкой.
- Теория

Расстояние от точки до прямой

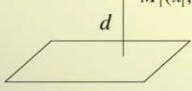
Расстояние от точки $M_0(x_0, y_0)$ до прямой $Ax + By + C = 0$ находят по формуле

$$d = \frac{|Ax_0 + By_0 + C|}{\sqrt{A^2 + B^2}}$$

- N при моделировании, оптимизации и Co-simulation – 8.
 - N для исследования производительности – 8192, 16384, 32768, 65 536.
 - Надо: провести исследования оптимизированного решения для типов данных int, long long, float, double
11. Даны 7 векторов A, B, C, D, x1, y1, z1 каждый размером N элементов, задающие N пар: плоскость $A[i]+B[i]+C[i]+D[i] = 0$ и $M_1(x[i]_1, y[i]_1, z[i]_1)$ в пространстве. Надо найти все [i]-е элементы вектора d: d[i] – расстояние между [i] плоскостью и [i] точкой.
- Теория

Расстояние от точки до плоскости

Расстояние от точки $M_1(x_1; y_1; z_1)$ до плоскости $Ax + By + Cz + D = 0$ находится по формуле

$$d = \frac{|Ax_1 + By_1 + Cz_1 + D|}{\sqrt{A^2 + B^2 + C^2}}$$


Расстояние – это длина перпендикуляра, опущенного из точки на плоскость

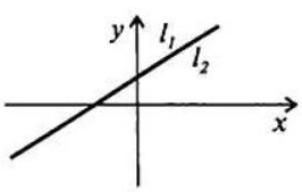
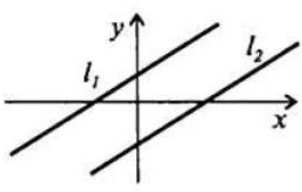
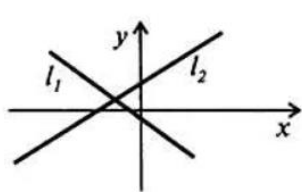
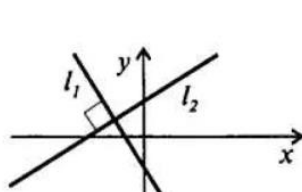
Правило: для нахождения расстояния от точки до плоскости нужно координаты точки подставить в левую часть уравнения плоскости, разделить на длину вектора нормали плоскости и полученное значение взять по абсолютной величине.

! Расстояние – величина всегда положительная

- a. N при моделировании, оптимизации и Co-simulation – 8.
 - b. N для исследования производительности – 8192, 16384, 32768, 65 536.
 - c. Надо: провести исследования оптимизированного решения для типов данных int, long long, float, double
12. Даны 6 векторов A1, B1, C1, A2, B2, C2, каждый размером N элементов, задающие N пар прямых в пространстве: $A1[i]x + B1[i]y + C1[i]z = 0$ и $A2[i]x + B2[i]y + C2[i]z = 0$ в пространстве.
- a. Надо определить взаимное расположение пар прямых:
 - i. Совпадающие прямые. Возвращаемое значение = 4
 - ii. Параллельные прямые. Возвращаемое значение = 3
 - iii. Пересекающиеся прямые. Возвращаемое значение = 2
 - iv. Перпендикулярные прямые. Возвращаемое значение = 1
 - b. Тест должен запускать функцию не менее 3-х раз. Исходные вектора A1, B1, C1, A2, B2, C2 должны содержать N/4 совпадающих прямых, N/4 параллельных прямых, N/4 пересекающихся прямых, N/4 перпендикулярных прямых.
 - c. Теория

ВЗАИМНОЕ РАСПОЛОЖЕНИЕ ДВУХ ПРЯМЫХ, ЗАДАННЫХ ОБЩИМ УРАВНЕНИЕМ ПРЯМОЙ

$$l_1: a_1x + b_1y + c_1 = 0; \quad l_2: a_2x + b_2y + c_2 = 0$$

СОВПАДАЮЩИЕ ПРЯМЫЕ		
	<p>Для коэффициентов a_i, b_i, c_i, отличных от нуля:</p> $\frac{a_1}{a_2} = \frac{b_1}{b_2} = \frac{c_1}{c_2}$	<p>Для произвольных коэффициентов a_i, b_i, c_i:</p> $\begin{cases} a_1b_2 = a_2b_1 \\ a_1c_2 = a_2c_1 \end{cases}$
ПАРАЛЛЕЛЬНЫЕ ПРЯМЫЕ		
	$\frac{a_1}{a_2} = \frac{b_1}{b_2} \neq \frac{c_1}{c_2}$	$\begin{cases} a_1b_2 = a_2b_1 \\ a_1c_2 \neq a_2c_1 \end{cases}$
ПЕРЕСЕКАЮЩИЕСЯ ПРЯМЫЕ		
	$\frac{a_1}{a_2} \neq \frac{b_1}{b_2}$	$a_1b_2 \neq a_2b_1$
ПЕРПЕНДИКУЛЯРНЫЕ ПРЯМЫЕ		
	$\frac{a_1}{b_1} = -\frac{b_2}{a_2}$	$a_1a_2 + b_1b_2 = 0$

- a. N при моделировании, оптимизации и Co-simulation – 8.
- b. N для исследования производительности – 8192, 16384, 32768, 65 536.
- c. Надо: провести исследования оптимизированного решения для типов данных int, long long, float, double

13. Дана матрица A, размер NxN.

- a. Создать функцию проверки матрицы на симметричность.

Определение. Если $A^T = A$, то матрица A называется **симметричной**.

Создаваемая функция должна быть иерархической - содержать две подфункции: транспонирования матрицы и сравнения матриц.

Тест, который должен запускать создаваемую функцию не менее 4-х раз, должен порождать две симметричные матрицы, например, как-то так:

```
int matrix[size][size];
for (int i = 0; i < size; i++) {
    for (int j = 0; j < i + 1; j++) {
        int num = rand();
        matrix[i][j] = num;
        matrix[j][i] = num;
    }
}
```

и две несимметричные матрицы. Заполнение элементов матриц случайными числами.

- b. N при моделировании, оптимизации и Co-simulation – 8.
- c. N для исследования производительности – 32, 64, 128, 256.
- d. Надо: провести исследования оптимизированного решения для типов данных int, long long, float, double

14. Даны матрицы A и B, квадратные, размер NxN.

- a. Создать функцию, которая определяет являются ли матрицы перестановочными

Определение. Матрицы, для которых $AB = BA$, называются **перестановочными**.

Тест, который должен запускать создаваемую функцию не менее 4-х раз, должен два раза порождать перестановочные матрицы, и два раза не перестановочные. Простейшим примером матрицы, перестановочной со всеми квадратными, является единичная матрица соответствующего порядка.

Т.е. тест два раза создает произвольную матрицу и проверяет ее на «перестановочность» с единичной матрицей, и два раза создает две произвольные матрицы и проверяет их на «перестановочность». Заполнение элементов матриц – с помощью rand().

- b. N при моделировании, оптимизации и Co-simulation – 8.
- c. N для исследования производительности – 32, 64, 128, 256.
- d. Надо: провести исследования оптимизированного решения для типов данных int, long long, float, double

15. Даны вектора A, B, C, X, размер N.

- a. Надо найти все элементы вектора r, размером N: $r[i]$ равно значению многочлена $A[i]x[i]^2 + B[i]x[i] + C[i]$.
- b. N при моделировании, оптимизации и Co-simulation – 8.
- c. N для исследования производительности – 8192, 16384, 32768, 65 536.
- d. Надо: провести исследования оптимизированного решения для типов данных int, long long, float, double

16. Дана матрица A, квадратная, размер NxN.

- a. Создать функцию, которая находит S - след матрицы (сумму элементов главной диагонали).
Создаваемая функция должна быть сделана иерархической (содержать подфункции): нахождения S; нахождения D
- b. N при моделировании, оптимизации и Co-simulation – 8.
- c. N для исследования производительности – 32, 64, 128, 256.
- d. Надо: провести исследования оптимизированного решения для типов данных int, long long, float, double, int256

17. Даны матрица A и матрица B, размером NxN.

- a. Надо найти матрицу C, размером NxN:
 - i. $C[i,j] = (A[i,j] + B[i,j])^2$
- b. N при моделировании, оптимизации и Co-simulation – 8.
- c. N для исследования производительности – 32, 64, 128, 256.
- d. Надо: провести исследования оптимизированного решения для типов данных int, long long, float, double

18. Даны матрица A и матрица B, размером NxN.

- a. Надо найти матрицу C, размером NxN:
 - i. $C[i,j] = (A[i,j] - B[i,j])^2$
- b. N при моделировании, оптимизации и Co-simulation – 8.
- c. N для исследования производительности – 8192, 16384, 32768, 65 536.
- d. Надо: провести исследования оптимизированного решения для типов данных int, long long, float, double

19. Даны матрица A и матрица B, размером NxN.

- a. Надо найти матрицу C, размером NxN:
 - i. $C[i,j] = (A[i,j] + B[i,j]) * (A[i,j] - B[i,j])$
- b. N при моделировании, оптимизации и Co-simulation – 8.
- c. N для исследования производительности – 8192, 16384, 32768, 65 536.
- d. Надо: провести исследования оптимизированного решения для типов данных int, long long, float, double

20. Дан вектор A, размером N элементов. Дано произвольное число D. Надо определить имеется ли в векторе A комбинация из 6 подряд идущих элементов, равных D и если имеется, то сколько раз встречается такая комбинация элементов.
- Тест
 - должен запускать функцию не менее 3-х раз.
 - Для исследования N= 8192, 16384, 32768, 65 536 элемента
 - N при моделировании, оптимизации и Co-simulation N=128
 - Число D должно генерироваться случайным образом.
 - Исходный вектор A для каждого запуска должен генерироваться заново.
 - Он должен содержать N/8192 комбинаций из четырех чисел D для этапа исследования
 - 2 комбинаций для этапа оптимизации
 - Надо: провести исследования оптимизированного решения для типов данных int, long long
21. Дан вектор A, размером N элементов. Дано произвольное число D. Надо определить имеется ли в векторе A комбинация из 4 подряд идущих элементов, равных D и если имеется, то сколько раз встречается такая комбинация элементов.
- Тест
 - должен запускать функцию не менее 3-х раз.
 - Для исследования N= 8192, 16384, 32768, 65 536 элемента
 - N при моделировании, оптимизации и Co-simulation N=128
 - Число D должно генерироваться случайным образом.
 - Исходный вектор A для каждого запуска должен генерироваться заново.
 - Он должен содержать N/8192 комбинаций из четырех чисел D для этапа исследования
 - 2 комбинаций для этапа оптимизации
 - Надо: провести исследования оптимизированного решения для типов данных int256
22. Дан вектор A, размером N элементов. Даны производное число D. Надо определить имеется ли в векторе A комбинация из 4 подряд идущих элементов, равных D и если имеется, то сколько раз встречается такая комбинация элементов.
- Тест
 - должен запускать функцию не менее 3-х раз.
 - Для исследования N= 8192, 16384, 32768, 65 536 элемента
 - N при моделировании, оптимизации и Co-simulation N=128
 - Число D должно генерироваться случайным образом.
 - Исходный вектор A для каждого запуска должен генерироваться заново.
 - Он должен содержать N/8192 комбинаций из четырех чисел D для этапа исследования
 - 2 комбинаций для этапа оптимизации
 - Надо: провести исследования оптимизированного решения для типов данных int, long long
23. Дана матрица A, квадратная, размер NxN и вектор B, размер N. Тип элементов int.
- Создать функцию, которая осуществляет умножение матрицы на вектор при разделении данных по столбцам (раздел 7.7 <http://www.hpcc.unn.ru/mskurs/RUS/DOC/ppr07.pdf>). Следует реализовать 8 параллельно работающих потоков (каждый поток обрабатывает соответствующие $0 \dots N/4-1$; $N/4 \dots 2N/4-1$; $2N/4 \dots 3N/4-1$; $3N/4 \dots N-1$; столбцы матрицы и элементы вектора).
- Создаваемая функция должна быть сделана иерархической (содержать подфункции): нахождения частичного произведения (должно быть 4 экземпляра данной подфункции); суммирования частичных результатов, полученных в 4-х подфункциях.*
24. Дана матрица A, квадратная, размер NxN и вектор B, размер N. Тип элементов int.
- Создать функцию, которая осуществляет умножение матрицы на вектор при разделении данных по строкам (раздел 7.6 <http://www.hpcc.unn.ru/mskurs/RUS/DOC/ppr07.pdf>). Следует реализовать 4 параллельно работающих потоков (каждый поток обрабатывает соответствующие $0 \dots N/4-1$; $N/4 \dots 2N/4-1$; $2N/4 \dots 3N/4-1$; $3N/4 \dots N-1$; строки матрицы).
- Создаваемая функция должна быть сделана иерархической (содержать подфункции): нахождения частичного произведения (должно быть 4 экземпляра данной подфункции); суммирования частичных результатов, полученных в 4-х подфункциях.*
25. Дана матрица A, квадратная, размер NxN. Тип элементов int.
- Создать функцию, которая формирует
 - L, вектор, размер N – содержит максимальные элементы каждой из N строк матрицы A Следует реализовать 4 параллельно работающих потоков (каждый поток обрабатывает соответствующие $0 \dots N/4-1$; $N/4 \dots 2N/4-1$; $2N/4 \dots 3N/4-1$; $3N/4 \dots N-1$; строки матрицы)
- Создаваемая функция должна быть сделана иерархической (содержать подфункции): нахождения максимумов для каждой строки из заданного количества строк (должно быть 4 экземпляра данной подфункции); объединения данных, полученных в 4-х подфункциях.*

Общие замечания.

Рабочая папка для проекта: C:\Xilinx_trn\KP_NG_NZ

Если рабочий диск отличается от C, то об этом д.б. написано в приложении к отчету.

Имя проекта KP_NG_NZ

- NG – номер группы (502 или 001 или 501)
- NZ – номер задания.

Микросхема для реализации - xcku115_CIV-flvd1517-2-e

Этапы работы

Этап 1 (предварительные результаты следует представить до 28 ноября)

1. Разработку описания создаваемой функции на языке C++.
2. Разработку теста на языке C++.
3. Моделирование созданной функции.
4. Исследование вариантов аппаратной реализации функции для периода тактового сигнала 6, 10, 14 нс (uncertainty = 1нс). Построение в xls таблицы и графиков. Выбор оптимального (мин аппаратных затрат, max производительности) решения (периода тактового сигнала) для дальнейших исследований.
 - При данном исследовании следует выключить конвейеризацию всех циклов.
5. Использование известных Вам директив, актуальных для оптимизации вашей функции: цель - max производительность, ограничения – доступные аппаратные ресурсы микросхемы xcku115_CIV-flvd1517-2-e.
 - FIFO vs RAM для реализации памяти массивов
 - UNROLL, Array_Partition, Array_Reshape, Pipeline, DataFlow (с выбором FIFO vs PingPong), Inline, Loop Merge. Следует привести соображения по выбору тех или иных директив для оптимизации.
 - Все, даже неудачные, попытки оптимизации желательно оформить отдельными решениями и представить в пояснительной записке.
 - Для сравнения результатов следует заполнить таблицу и привести графики (решение ⇔ аппаратные затраты и быстродействие).

Этап 2 (предварительные результаты следует представить до 5 декабря)

1. Создание теста для анализа временных затрат при программной реализации функции (на Вашем ПК). Тест должен содержать не менее 32 запусков исследуемой функции.
2. Оценка производительности аппаратного решения
3. Оценка производительности программной реализации на ПК
Для оценки выбираем медиану среди полученных при каждом из запусков функции 32 значений временных затрат.
 - Надо реализовать
 - одноподольное выполнение вашей функции на вашем ПК
 - многоподольное выполнение
 - При исследовании производительности решения для типов данных int, long long, float, double, int256 исходные данные входных массивов должны быть псевдослучайными из всего диапазона соответствующих типов данных.
4. Сравнительный анализ программных и аппаратной реализаций (составить таблицу и привести графики).

Этап 3 (предварительные результаты следует представить до 11 декабря, 13 декабря обсуждение/вопросы)

5. Написание черновика отчета.

Требования

1. Параметр N и тип данных должны быть определены в h файле
2. Оптимизация аппаратной реализации и оценка ее производительности осуществляется в пакете Vitis HLS.
3. Способ оценки производительности: Π (число тактов) * (Estimated time period + Uncertainty) = периоду тактовой частоты работы устройства
4. При моделировании, в рамках пакета Vitis HLS, исследуемую функцию следует запускать не менее трех раз (если в задании не сказано иное).
5. Заполнение матриц в тестах осуществлять случайными числами (если в задании не указано иное).

6. Тест должен быть с самопроверкой результата (обязательно предусмотреть и отразить в записке проверку теста – при неправильных данных).
7. Алгоритм вычисления ожидаемых значений в тесте, по возможности, не должен повторять алгоритм вычисления проверяемой функции.

Оформить отчет, который должен включать

- Задание
- Раздел с описанием исходного кода функции
- Раздел с описанием теста
- Раздел с описанием оптимизации аппаратной реализации и оценкой ее производительности (привести описание всех созданных решений)
- Раздел с описанием теста для оценки производительности ПК (для двух вариантов – одноядерного и многоядерного).
- Результаты и анализ результатов производительности программной реализации на ПК
- Раздел со сравнительным анализ программной и аппаратной реализаций
- Выводы
- Приложение, в котором должны быть описаны папки и основные файлы в архиве проекта + должен быть указан рабочий диск, если он отличается от C.

Загружаемый на DL результат:

Архив, который должен включать всю рабочую папку проекта (включая, исходные коды, модернизированный тест для оценки программной реализации функции, xls таблицы....), отчет в редактируемом формате.

Распределение заданий

[illegible]

Группа 502				
	Номер задания			
1 Бальжиева Виктория Буянтуевна	22			
2 Белоногов Николай Иванович	21			
3 Джеус Андрей Сергеевич	20			
4 Дягай Александра Руслановна	19			
5 Дубовицкий Ян Вячеславович	18			
6 Загороднов Дмитрий Сергеевич	17			
7 Куксов Григорий Валерьевич	16			
8 Ляшенко Валерия Владимировна	15			
9 Марашов Александр Сергеевич	14			
10 Овсянников Егор Александрович	13			
11 Орлова Полина Алексеевна	12			
12 Птицын Алексей Алексеевич	11			
13 Селецкая Алиса Андреевна	10			
14 Скопецкий Анатолий Григорьевич	9			
15 Тяпуев Дмитрий Артемьевич	8			
16 Шеметов Степан Андреевич	7			
17 Волохов Леонид	6			

Группа 001				
	Номер задания			
Величко Вадим Олегович	1			
Волкова Елена Константиновна	5			
Воронов Владимир Александрович	10			
Кузина Мария Александровна	15			
Шакола Александр Александрович	20			