Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет информационных технологий и управления

Кафедра интеллектуальных информационных технологий

ОТЧЁТ

по лабораторной работе №1

по дисциплине

Модели решения задач в интеллектуальных системах

“Реализация модели решения задач на конвейерной архитектуре”

Вариант 1

|  |  |
| --- | --- |
| Выполнил | ст. гр. 221701 Робилко Т. М. |
| Проверил | Ивашенко В. П. Н.В |
|  |  |

Минск, 2025

**Тема:** Программирование операций обработки данных и знаний с конвейеризированной обработкой потока данных.

**Цель:** Реализовать и исследовать модель решения на конвейерной архитектуре задачи вычисления попарного произведения (деления (обращения)) компонентов двух векторов чисел.

**Вариант 1:** алгоритм вычисления произведения пары 4-разрядных чисел умножением с младших разрядов со сдвигом множимого (частичного произведения) влево.

**Дано:** сгенерированные два вектора А и В заданной длинной m каждый, элементы которых являются положительными числами заданной разрядности р (2р и р).

**Получить:** вектор значений операции целочисленного умножения (деления) для каждой пары чисел, имеющих длину m и разрядность компонентов 2р (р и р)

**Описание реализации**

Реализация выполнена на языке программирования C#.

Для представления чисел создан класс *Number*, инкапсулирующий беззнаковое число (32-битное число типа uint) и переменную, показывающую количество бит в числе. За счёт этого возможно обрабатывать и отображать согласно требованиям к работе числа произвольной разрядности от 1 до 32 бит.

Настройка количества бит в числе позволяет изменять параметр *r* конвейера для тестирования системы и ответа на контрольные вопросы.

Числа считываются парами из файла либо генерируются на основании последовательности чисел (в зависимости от конфигурации программы). За считывание пар чисел из файла отвечает класс *PairReader*. За генерацию пар чисел отвечает класс *MockPairReader.* Оба класса реализуют интерфейс *IPairReader*.

Наличие класса *MockPairReader* позволяет генерировать пары чисел в заданном количестве, что позволяет изменять параметр *n* конвейера для ответа на контрольные вопросы.

Для осуществления умножения чисел реализован класс *MultiplicationTriple*, инкапсулирующий три числа (переменные типа *Number*): множимое, множитель и частичная сумма.

Для представления конвейера разработан класс *Pipeline*. Также определен вложенный в него класс *PipelineStage* для хранения операции этапа конвейера и индекса этапа (для пошагового отображения вычислений), а также тип-делегат *PipelineStageFunction* для типизации функций, которые могут быть заданы как операция этапа конвейера.

Класс *Pipeline* содержит поля:

* + - 1. Input – очередь объектов классов *MultiplicationTriple*, подающихся на вход конвейеру.
      2. Output – очередь объектов классов *MultiplicationTriple*, которые получаются в результате вычислений на конвейере.
      3. Stages – массив этапов конвейера (объекты типа *PipelineStage*)
      4. CurrentTick – целочисленная переменная для отображения текущего такта конвейера.

Для перехода на следующий такт вызывается метод *Tick* класса *Pipeline*

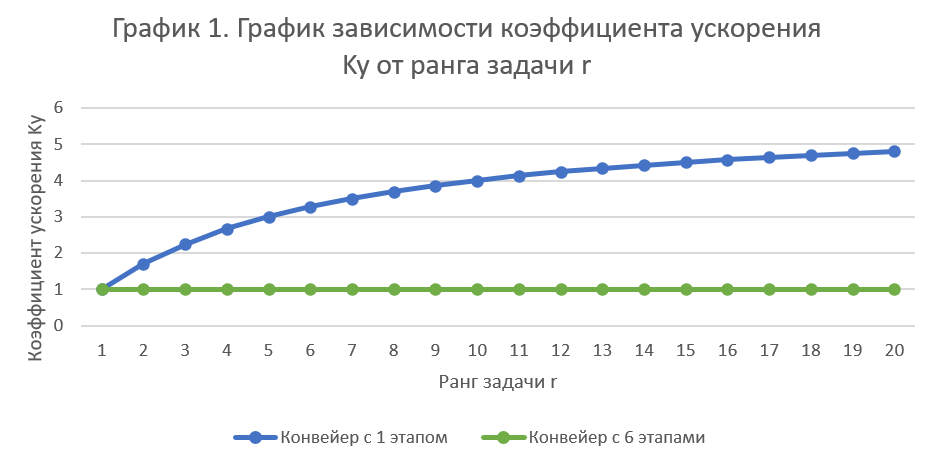
Этапы конвейера инициализируется статической функцией *PipelineStageFunction* статического класса *Arithmetics.* Функция соответствует сигнатуре *PipelineStageFunction*, отчего может использоваться в качестве операции этапа конвейера. В классе *Arithmetics* также содержится метод *Multiply*, принимающий список пар объектов класса *Number* и возвращающий пару из количества затраченных тактов и списка чисел-результат умножения. Метод реализует покомпонентное умножение векторов чисел с использованием конвейерной архитектуры.

Для запуска системы и ее конфигурации созданы дополнительные классы *Debug* и *Configurations*, отвечающие за логгирование и запуск вычислений либо в одиночном режиме, либо в режиме осуществления нескольких прогонов с разными параметрами системы (параметры *n*, *r* конвейера).

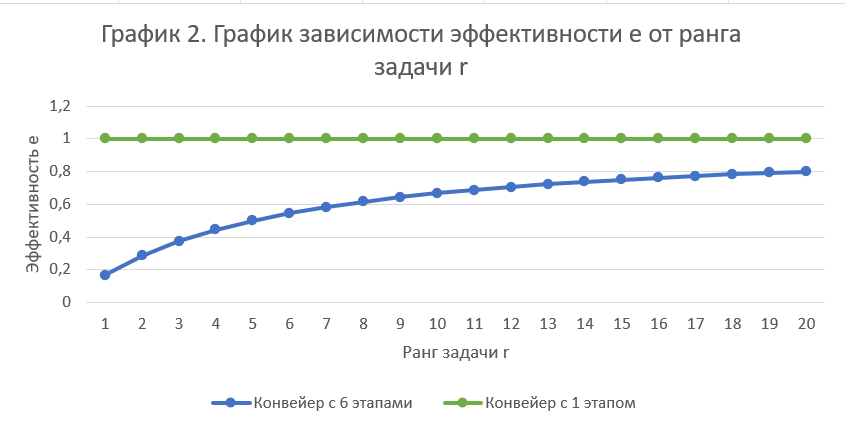
**Построение графиков**

Введём обозначения:

1. n – количество этапов конвейера;
2. r – ранг задачи (количество обрабатываемых пар);
3. T1(n, r) = r \* n – время, затрачиваемое на вычисления в однопроцессорной вычислительной системе;
4. Tn(n, r) = n + r – 1, при условии, что r > 0 – время, затрачиваемое на вычисления в параллельной вычислительной системе;
5. Ky(T1, Tn) = – коэффициент ускорения;
6. e(Ky, n) = – эффективность.



*Рис. 1 – График зависимости коэффициента ускорения Ky от ранга задачи для конвейеров с 1 и с 6 этапами соответственно.*



*Рис. 2 – График зависимости эффективности е от ранга задачи r для конвейеров с 1 и с 6 этапами соответственно.*

**Контрольные вопросы**

1. **Проверить, что модель создана верно: программа работает правильно (на всех этапах конвейера).**

Для проверки введём 2 вектора чисел длиной 4:

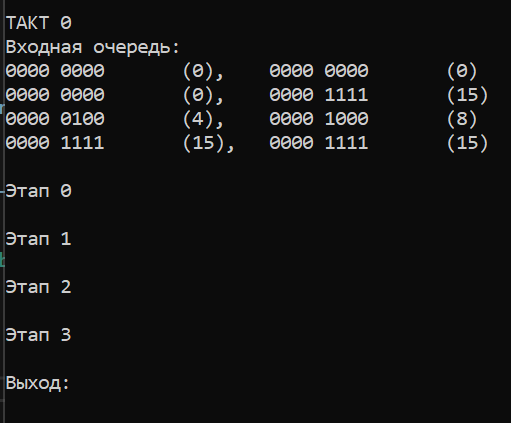
Вектор 1 – 0, 0, 4, 15;

Вектор 2 – 0, 15, 8, 15.

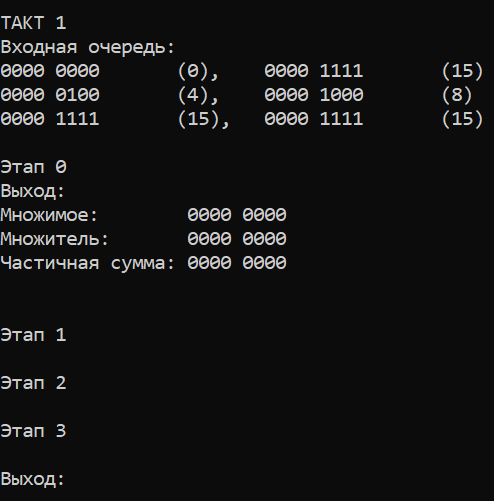
Таким образом, получим 4 пар – (0, 0), (0, 15), (4, 8), (15, 15)

Ожидаемые ответы в результате работы программы: 0, 0, 32, 225.

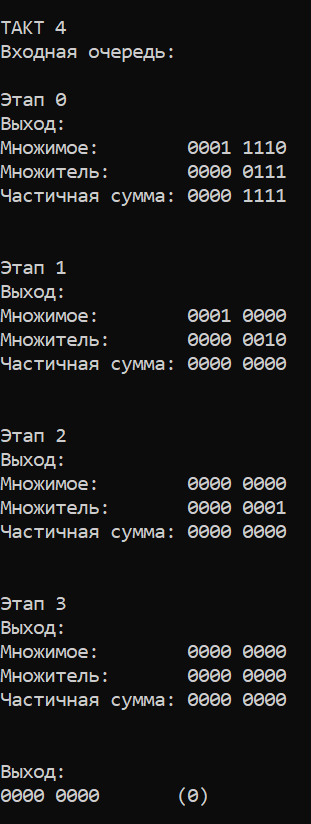
Такт 0 (момент запуска программы):



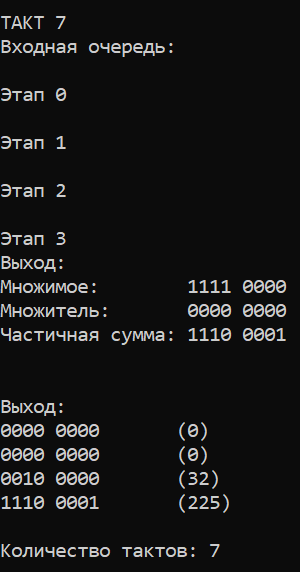
Такт 1:



Такт 4:



Такт 7:



Остальные этапы опущены за аналогичностью промежуточных результатов.

Результаты совпадают с ожидаемыми. Таким образом, программа работает правильно на всех этапах конвейера, модель построена верно.

1. **Объясните на графиках точки перегиба и асимптоты:**

Чтобы определить асимптоты для графиков коэффициента ускорения, необходимо рассмотреть случаи при и при .

Для первого случая коэффициент ускорения Ky ограничен значением n, так как

В свою очередь

Подставив в (1), получим

Возьмём предел:

Предел даёт неопределенность вида . Преобразуем:

Подставив в (3), получим:

Таким образом, для варианта работы коэффициент ускорения Ky ограничен 4.

Для второго случая коэффициент ускорения Ky ограничен значением r. Возьмем предел для (2):

Как и в первом случае, предел (4) даёт неопределенность вида .

Подставив в (4), получим:

Асимптоты и перегибы на графиках эффективности объясняются похожим образом. Достаточно рассмотреть случаи и .

В первом случае, пользуясь формулой

возьмём предел:

Эффективность показывает долю работы одного процессорного элемента, а при ограниченном количестве блоков каждый блок будет задействован в вычислениях. Следовательно, эффективность будет максимальной.

Для второго случая возьмём предел:

Количество вычислительных блоков гораздо больше ранга задачи, следовательно, некоторые блоки будут простаивать. Следовательно, эффективность будет минимальной.

1. **Спрогнозируйте, как изменится вид графиков при изменении параметров модели.**

Возьмём графики для коэффициента ускорения Ку. При увеличении ранга задачи r увеличивается и значение коэффициента ускорения. При увеличении количества процессорных элементов n также увеличивается значение коэффициента ускорения. Следовательно, график будет приближаться к асимптоте.

Возьмём графики для эффективности е. При увеличении ранга задачи r значение эффективности увеличивается. График приближается к асимптоте. Однако при увеличении количества процессорных элементов n значение эффективности снижается.

1. **Каково соотношение между параметрами n, r, m, p модели сбалансированного конвейера?**

n – количество процессорных элементов;

r – максимальное количество пар, которые могут быть обработаны;

m – количество пар;

p – разрядность умножаемых попарно чисел;

r = m;

n = p = 4 (для варианта лабораторной работы).

1. **Допустим: имеется некоторая характеристика h (эффективность e или ускорение Ky) и для неё выполняется: h(n1, r1) = h(n2, r2) и n1 > n2. Каким будет соотношение между r1 и r2?**

В качестве характеристики h возьмём эффективность е. Подставив формулу

в заданное выражениеполучим

откуда:

Для выполнения необходимо . Отсюда .

1. **Дано: несбалансированный конвейер (заданы конкретные значения n, {ti} – времена выполнения обработки на этапах конвейера); е0 – некоторое фиксированное значение эффективности. Определить значение r0**, **при котором выполняется е(n, r0) > e0?**

Подставим полученную формулу коэффициента ускорения в формулу эффективности е:

Подставим полученную формулу эффективности в требуемое неравенство:

Выразим :

При условии

что удовлетворяет наложенным условиям с учётом неотрицательного числителя, так как для любого конвейера верно:

Если же

результат выражения отрицательный, так как знаменатель отрицательный. Так как по условию , данный случай в ответ не входит.

Если же

Что противоречит наложенным условиям .

Таким образом, ответ:

1. **Для несбалансированного конвейера (использовать исходные данные предыдущего вопроса) определить .**

Так как , то по правилу Лопиталя:

Следовательно, ответ:  *.*

1. **Дан несбалансированный конвейер (использовать исходные данные предыдущего вопроса). Каким образом можно перестроить данный конвейер, чтобы для заданного r0 выполнялось е(n, r0) > e0?**

Воспользуемся неравенством из задания 6 и преобразуем его:

Учитывая, что

получаем следовательно n < n0. Для выполнения заданного условия необходимо, чтобы n в перестроенном конвейере было меньше, чем n0 в конвейере до перестроения. При этом необходимо, чтобы при объединении этапов максимальное время этапа на перестроенном конвейере было равно максимальному времени этапа на конвейере до перестроения.

Необходимо объединять этапы конвейера таким образом, чтобы выполнялось неравенство

. Приведем примеры решения такой задачи на конкретных данных:

Пусть , , , значение примем за рассчитанное для данного конвейера значение эффективности .

Перестроим конвейер следующим образом: , , , – в данном случае 1 этап объединен со 2-ым, 3-й с 4-ым, 5-й с 6-ым, 7-ой с 8-ым. Тогда . В результате , что соответствует условию задачи.

1. **Дан несбалансированный конвейер (использовать исходные данные предыдущего вопроса) и значение минимального кванта времени (условной временной единицы). Каким образом нужно перестроить данный конвейер, чтобы получить максимально быстрый конвейер? Получить для него формулы ?**

Для того, чтобы получить максимально быстрый конвейер, нужно перестроить так, чтобы он стал сбалансированным, и каждый этап выполнялся за минимальное время , следовательно, = . Добиться этого можно путем разделения этапов конвейера.

Ответ: изменить количество этапов так, чтобы они выполнялись за одинаковое минимальное время (разделить этапы)*,* , .

**Личный вклад:**

Работа была выполнена в паре студентами группы 221701 Абушкевичем Алексеем Александровичем и Робилко Тимуром Марковичем. Реализация модели выполнена индивидуально согласно выбранным вариантам. Отчёт составлен совместно.

**Вывод:**

В ходе выполнения лабораторной работы была реализована модель решения задачи вычисления произведения пары 4-разрядных чисел умножением с младших разрядов со сдвигом множимого влево на конвейерной архитектуре. Модель реализована и протестирована. Построены и проанализированы семейства графиков и даны ответы на контрольные вопросы.

**Использованные источники:**

1. Модели решения задач в интеллектуальных системах. В 2 ч. Ч.1: Формальные модели обработки информации и параллельные модели решения задач: учеб.-метод. пособие/ В. П. Ивашенко. – Минск: БГУИР, 2020. – 79 с.
2. John L. Hennessy, David A. Patterson Computer Architecture: A Quantitative Approach Sixth Edition / Morgan Kaufmann, 2019. – 1527 с.
3. Методы и алгоритмы обработки данных [Электронный ресурс]. Режим доступа: https://studref.com/636037/ekonomika/vychislitelnyy\_konveyer (дата обращения 05.03.2024).