**ВВЕДЕНИЕ**

С тех пор как человек в своей повседневной деятельности перестал довольствоваться информацией, доставляемой ему лишь его органами чувств, и привлек им в помощь инструментальные средства, проблема создания и совершенствования этих средств стала одним из важнейших направлений, определяющих прогресс естественных наук и промышленного производства. Созданием и использованием средств измерений в тех или иных приложениях занимаются многие научные и производственные коллективы. К радикальным изменениям характера эксперимента привело использование вычислительной техники, сделавшей возможной оперативную обработку больших массивов информации.

Информационно- измерительные системы относятся к информационной технике, которые содержат основные ветви:

* вычислительная техника
* техника передачи информации
* техника хранения
* техника поиска информации
* информационно-измерительная техника

Каждая из этих ветвей имеет свои особенности, принципы и методы построения технических устройств. В тоже время они объединяются общими теоретическими основами, так как оперируют с понятием информация.

Получение всего объема информации в современных автоматизированных системах и ее обработка должны выполняться за ограниченное время. Если эти функции возложить на человека, вооруженного лишь простейшими вычислительными устройствами, то в силу физиологических ограничений он не сможет их выполнить.

Таким образом перед измерительной техникой были поставлены проблемы создания новых средств и усовершенствования уже имеющихся, способных разгрузить человека от необходимости сбора и обработки информации.

Важнейшими составляющими эффективности ИИС, являются показатели живучести, надежности, точности воспроизведения исходной информации, способности функционировать в реальном масштабе времени, затрат оборудования, стоимости разработки, производства и эксплуатации.

Хорошо знакомым на сегодняшний день является представление информации в виде двоичных позиционных параллельных или последовательных кодов.

Гораздо менее известной является дискретная форма представления информации в виде вероятностных отображений. Данная форма позволяет значительно уменьшить аппаратный объем схем для выполнения арифметических операций и значительно увеличить быстродействие данных систем. Однако недостатки этой формы представления данных вынуждают научное сообщество искать усовершенствование этого метода.

Наиболее эффективными являются методы ускорения арифметических операций, позволяющие повысить быстродействие систем на уровне обработки информации, представленной в вероятностной форме.

**1 ПОСТАНОВКА ЗАДАЧИ**

Тема работы: Применение методов ускоренного умножения и сложения в вероятностной форме представления данных.

Целью проводимых исследований является повышение быстродействия выполнения арифметических операций при представлении информации в вероятностной форме.

Задачи для разрешения поставленной цели

1. Исследование вероятностной формы представления данных.
2. Исследование методов ускорения операции сложения.
3. Исследование методов ускорения операции умножения.
4. Разработка модели преобразователя “число-вероятность”.
5. Разработка модели ускоренного выполнения операции сложения.
6. Разработка модели ускоренного выполнения операции умножения
7. Оценка качества вероятностного сумматора и вероятностного множителя с ускоренным выполнением арифметических операций
8. Анализ полученных результатов.

Объектом исследования является проблема потери быстродействия преобразователя «число-вероятность» за счет большого количества статистических испытаний.

Предметом исследования в данной работе являются увеличение быстродействия выполнения арифметических операций при представлении данных в вероятностной форме.

### Для достижения поставленной цели используется [прототипно-ориентированный](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D1%82%D0%B8%D0%BF%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) [сценарный](https://ru.wikipedia.org/wiki/%D0%A1%D1%86%D0%B5%D0%BD%D0%B0%D1%80%D0%BD%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) язык программирования JavaScript. Данный язык имеет высокое быстродействие при обработке входных данных большого объема.

### 2 ОБЗОР ЛИТЕРАТУРНЫХ ИСТОЧНИКОВ ПО ТЕМАТИКЕ ИССЛЕДОВАНИЯ

При написании данной работы были использованы научная и учебно-методическая литература, статьи в периодических изданиях Государства Украины и Российской Федерации, публикации в сети Интернет.

**2.1 Вероятностная форма представления данных**

Для исследования вероятностной формы представления данных была рассмотрена учебно-методическая литератураСапожникова Н.Е. Дискретная схемотехника. Севастополь: СНИЯЭиП, 2005.(2)

В данном пособии речь идет о том, что суть стохастического или вероятностного преобразования заключается в том, что любому значению параметра преобразуемой величины можно привести в соответствие некоторую вероят­ность.

В зависимости от правила, в соответствии с которым это происходит, методы преобразования делятся на однолинейное однополярное, однолинейное двухполярное и двухлинейное двухполярное представления.

В первом, наиболее простом случае, значение параметра преобразуемой величины либо всегда положительно, либо всегда отрицательно, а сам процесс преобразования выполняется в соответствии с правилом

, (2.1)

где -e значение параметра преобразуемого сигнала *Х(t)*;

*j-*е значение параметра вспомогательного случайного сигнала *R(t),* изменяющегося в интервале изменения *Х(t)*;

*i=*  - число циклов преобразования сигнала *Х(t)*;

*j=*  - количество статистических испытаний каждого значения  внутри временного интервала ;

 - значение вероятностного отображения параметра сигнала  из ряда .

Вероятностное отображение обладает свойствами синхронности (тактируемости) и независимости каждого члена отображения от любого другого.

Первое свойство заключается в том, что формирование членов вероятностного отображения производится через постоянный интервал времени , определяемый частотой выполнения правила (1).

Свойство независимости каждого члена вероятностного  
отображения от любого другого следует из того факта, что получение вероятностного отображения соответствует схеме испытаний Бернулли.

Схема испытаний Бернулли. Повторные независимые испытания называются испытаниями Бернулли, если каждое испытание имеет только два возможных исхода и вероятности исходов остаются неизменными для всех испытаний.

Для случайной последовательности, полученной в соответствии с данной схемой, автокорреляционная функ­ция представляет собой -функцию при . Для доказательства этого следует показать, что повторные испытания в соответствии с (2.1) также являются независимыми. Значения вспомогатель­ной случайной функции *R(t)* формируются в дискретные моменты времени. В любой момент времени функция может находиться только в одном из своих состояний с вероятностью . Очевид­но, что для любого *t*

(2.2)



и при заданных вероятностях  распределение  может быть задано плотностью вероятности

(2.3)



где

(2.4)



есть распределение фиксированной величины , определяемое функцией Дирака.

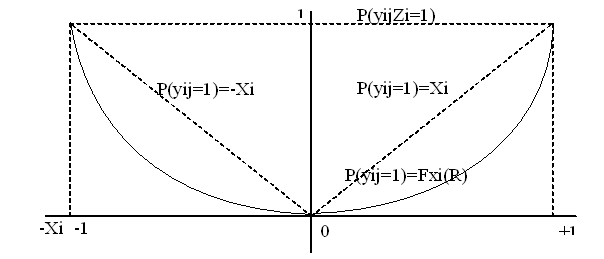


Рисунок 2.1 – Графики, поясняющие однолинейное однополярное вероятностное представление информации

Использование этих свойств и применение вероятностно представленных дискретных сигналов позволяет упростить функциональные узлы для выполнения арифметических и логических операций, в частности, сложения, вычитания, умножения, возведения в целую степень, деления, компарации и т.д. и, тем самым, резко уменьшить их аппаратурный объём.

Если принять диапазон изменения *Х(t)* единичным, то максимальное значение , будет представлено в вероятностном отображении значениями с вероятностью, равной единице в каждом шаге, а минимальное значение теми же значениями с нулевой вероятностью. График, поясняющий однолинейное однополярное вероятностное представление дискретной информации, показан на рис.2.1 в правой части рисунка.

С учётом исходного правила преобразования, вероятности появления «1» и «0» в вероятностном отображении равняются

, (2.5)



. (2.6)

Математическое ожидание от вероятностного отображения определяется через ряд распределения, представленного в таблице 2.1, для дискретной случайной величины .

Таблица 2.1 – Ряд распределения

|  |  |  |
| --- | --- | --- |
|  | 1 | 0 |
|  |  |  |

Тогда

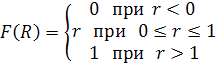
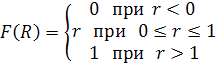
. (2.7)



Таким образом, вероятность появления «1» в вероятностном отображении есть математическое ожидание (МО) от отображения и численно равняется значению интегрального закона распределения вспомогательного сигнала *R(t)* при уровне сравнения .

Особый интерес представляет случай, когда вспомогательный случайный сигнал *R(t)* подчиняется равномерному закону распределения в соответствии с

. (2.8)



Для него последнее выражение для МО перепишется в виде

, (2.9)



т.е. имеем случай линейного вероятностного преобразования.

Важнейшим следствием из выражений для МО является тот факт, что значение параметра поддаётся восстановлению из вероятностного отображения, то есть возможно обратное преобразование «вероятность-значение параметра» (числа, амплитуды, частоты, фазы и т.д.). Действительно, априори зная закон распределения вспомогательного случайного сигнала *R(t)* и определяя МО от вероятностного отображения, то есть ординату интегрального закона распределения , путём функционального преобразования можно определить величину , являющуюся оценкой. В качестве такой оценки, удовлетворяющей требованиям несмещённости, состоятельности и эффективности, в соответствии с теоремой Чебышева, принимается

. (2.10)



Таким образом, при однолинейном однополярном преобразовании параметра сигнала в вероятность, для получения исходного значения следует подсчитать количество единиц в вероятностном отображении и отнести его к количеству статистических испытаний (количеству членов вероятностного отображения).

Однолинейное однополярное представление легко распространя­ется на двухлинейное двухполярное. Значения вероятностного отоб­ражения , как и в первом случае, будут получены в соответствии с пра­вилом (1.1), а затем для распределения по "положительной" или по "отрицательной" линии каждое значение должно быть логически умножено на переменную Z, которая может принять только два значения: +1 либо -1. Тогда выражение для вероятностного отображения примет вид

(2.11)



С точки зрения аппаратурной реализации это означает только то, что при значения будут поданы на "положитель­ную" шину, в противном случае - на "отрицательную". В этом слу­чае максимальная положительная величина будет представлена в последнем выражении с единичной вероятностью значениями на "положительной" шине и значениями на "отрицательной"; максимальная отрицательная величина - наобо­рот. Зависимость промежуточных значений от вели­чины параметра преобразуемого сигнала также показана на рис.2.1, при­чем пунктиром показан частный случай линейного преобразования.

Отсюда следует, что выражение для математического ожидания от вероятностного отображения для двухлинейного двухполярного представления остаётся справедливым и оба свойства вероятностного отображения, рас­смотренные ранее, не изменятся.

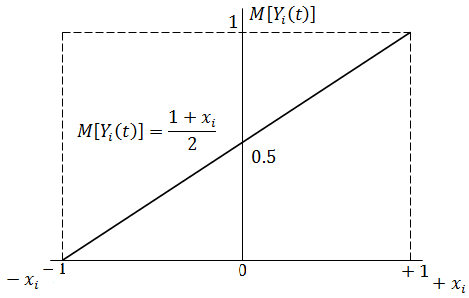


Рисунок 2.2 – График, поясняющий работу однолинейного двухполярного преобразователя

При однолинейном двухполярном представлении все значения параметра сигнала , изменяющегося в интер­вале от до могут быть переданы и по одной линии, если восполь­зоваться соотношением

 (2.12)

В этом случае число +1 представляется серией импульсов с вероятностью, равной единице в каждом такте (Р=1), число 0 - произво­льной серией, состоящей из единиц и нулей, следующих с равной вероятностью Р=0,5, а число -1 - серией импульсов с вероят­ностью Р=0. На рис.1.2 показан график, поясняющий работу однолинейного двухполярного преобразователя.

От двухлинейного симметричного преобразователя однолинейный двухполярный отличается способом кодирова­ния параметров исходного сигнала. Так, при представлении входной информации в виде позиционного двоичного кода, кодирование осу­ществляется по круговой системе, как это показано на рис.2.3. Максимальное положительное число соответствует коду 1.111...1, где перед точкой показан знак числа. Положи­тельные числа, в отличие от представления цифровых позиционных кодов, имеют в знаковом разряде 1, а отрицательные - 0.

Отрицательные числа представляются дополнительными кодами. Максимальному отрицательному числу соответствует код 0.000...0.

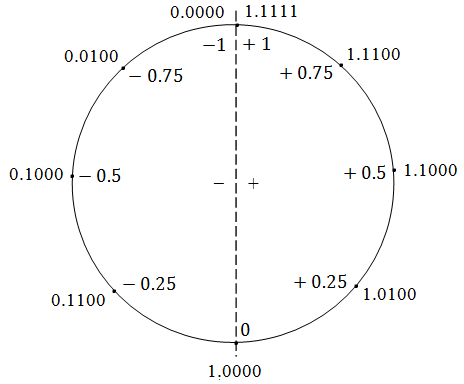


Рисунок 2.3 – Круговая схема кодирования чисел при однолинейном двухполярном вероятностном преобразовании

**2.2 Вероятностные сумматоры**

Основная информация по вероятностным сумматорам была взята в пособиях Гладкий В.С. Вероятностные вычислительные модели. – М.: Наука, 1973. и Федоров Р. Ф., Яковлев В. В., Добрис Г. В. Стохастические преобразова­тели информации. — Л.: Машиностроение. Ленингр. отделение, 1978.(3)

Для выполнения операции сложения двоичных шестнадцатиразрядных чисел могут быть использованы параллельные или последователь­ные сумматоры. Применение первых, при высоком быстродействии, приводит к значительным затратам оборудования. Так аппаратурный объем параллельного комбинационного сумматора, при использовании в качестве единицы измерения количества элементарных логических элементов, составит 181 элемент.

Если значение логической функции на выходе логического эле­мента, реализующего операцию дизъюнкции, в *j*-й момент времени обозначить через , то для *Q*  слагаемых, представленных в виде вероятностных отображений можно записать

 (2.13)

и дизъюнкция вероятностных отображений слагаемых примет вид

 (2.14)

Однако выполнять сложение в соответствии с последним выражением нельзя, так как

 (2.15)

Выход заключается в попеременной подаче значений на соответствующие входы схемы "ИЛИ" через время  где  - период следования тактовых импульсов, а *Q* – как и ранее, количество слагаемых. Тогда сумму вероятностных отображений можно рассматривать как частное вероятностное отображение суммы, представляющее собой последовательность  на периоде *Т,* разделённых . Группируя члены вероятностного отображения по одинаковой позиции на периоде *Т,* получим

вероятностное отображение дизъюнкции в виде

 (2.16)

Математическое ожидание случайной функции *S(t)* можно рассматривать как сумму математических ожиданий вероятностных отображений независимых и несовместных логических переменных, то есть



 (2.17)

Для случайных вспомогательных сигналов *R(t)* распределённых равномерно имеем

 (2.18)

и далее получим

 (2.19)

С другой стороны, поскольку при любом, в том числе и сколь угодно малом *ε*

 (2.20)

в качестве оценки для суммы входных слагаемых получим

, (2.21)

откуда следует, что вероятностный сумматор для суммирования *Q* слагаемых представляет собой (рис. 2.4) *Q*-входовую логическую схему "ИЛИ" и *Q-1* устройств временной задержки на время  каждое (1).

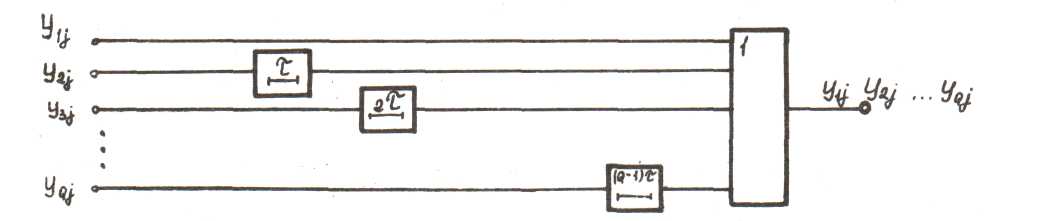


Рисунок 2.4. - Функциональная схема вероятностного сумматора при однолинейном однополярном представлении слагаемых

Таким образом, для двух слагаемых аппаратурный объем вероят­ностного сумматора сравнительно с детерминированным будет меньше в 90 раз. Для вычитающих устройств это число ещё более возрастёт за счёт необходимости представления вычитаемого в детерминирован­ном цифровом сумматоре в дополнительном коде.

Основной недостаток рассмотренного метода вероятностного суммирования заключается в том, что вероятностное отображение суммы образуется со сжатием во времени и для восстановления ис­ходного периода следования членов вероятностного отображения тре­буется введение дополнительных запоминающих элементов.

Этого не­достатка лишён метод вероятностного суммирования, при котором сумматор можно рассматривать как переключатель, который в такто­вые моменты времени случайным образом подключает к выходу одну из входных шин. Получить такой переключатель можно, подавая случайные числа, представленные в параллельном позиционном коде, на входы дешифратора. Тогда значение логической функции дизъюнкции на выходе схемы «ИЛИ» в *j*-й момент времени будет равно

, (2.22)

где - значение логической переменной (бита унитарного кода) на выходе дешифратора. В случае если случайный сигнал на входе дешифратора равномерен, имеем

 (2.23)

и предыдущее выражение перепишется в виде

 (2.24)

При линейном вероятностном преобразовании выражение упростится:

 (2.25)

Определим математическое ожидание случайной величины , способной принимать только два возможных значения – 0 или 1

 . (2.26)

В качестве оценки МО, с учётом следствия из теоремы Чебышева, примем

, (2.27)

откуда получим окончательное выражение для вычисления суммы вероятностных отображений

. (2.28)

Аппаратно в качестве вероятностного сумматора может быть использован мультиплексор на *Q* информационных и *m* управляющих входов, где , нагруженный на вход двоичного накопительного счётчика с числом разрядов  Синтезированная таким образом структура имеет значительно меньший аппаратурный объём, чем цифровой многоразрядный сумматор.

**2.3 Вероятностные множители**

Основная информация по вероятностным множителям была также взята в пособиях Гладкий В.С. Вероятностные вычислительные модели. – М.: Наука, 1973. и Федоров Р. Ф., Яковлев В. В., Добрис Г. В. Стохастические преобразова­тели информации. — Л.: Машиностроение. Ленингр. отделение, 1978.(3)

Была проанализирована информация о том, что представление синхронного вероятностного отображения в виде последовательно следующих логических переменных формально позволяет распространить на вероятностно представленный сигнал законы алгебры логики. Рассмотрим  вероятностно представленных сигналов, для каждого из которых выполняется условие :



****

……………………………….

****

……………………………….

****

Их конъюнкция, соответственно, будет иметь вид:

 (2.29)

Обозначим , тогда предыдущее выражение можно переписать в виде:

. (2.30)

Определим математическое ожидание случайной функции как

, (2.31)

где  - вероятность того, что логическая функция примет одно из двух возможных значений 0 или 1. В соответствии с таблицей истинности для конъюнкции *Q* независимых логических переменных

. (2.32)

Учитывая правило вероятностного преобразования, определим вероятность появления «1» в вероятностном отображении как

 (2.33)

то есть численно равное интегральному закону распределения вспомогательного случайного сигнала  при уровне сравнения . Таким образом, выражение для математического ожидания конъюнкции логической функции представляет собой произведение интегральных законов распределения , то есть

 (2.34)

В случае если  есть равномерный закон распределения, последнее выражение можно переписать в виде

 (2.35)

Таким образом, для вычисления произведения *Q* вероятностно преобразованных сигналов следует определить математическое ожидание логической функции конъюнкции. Как и раннее, принимая в качестве оценки математического ожидания среднее значение, окончательно получим

 (2.36)

что и требовалось доказать. При вычислении произведения двух вероятностно преобразованных сомножителей последнее выражение перепишется в виде

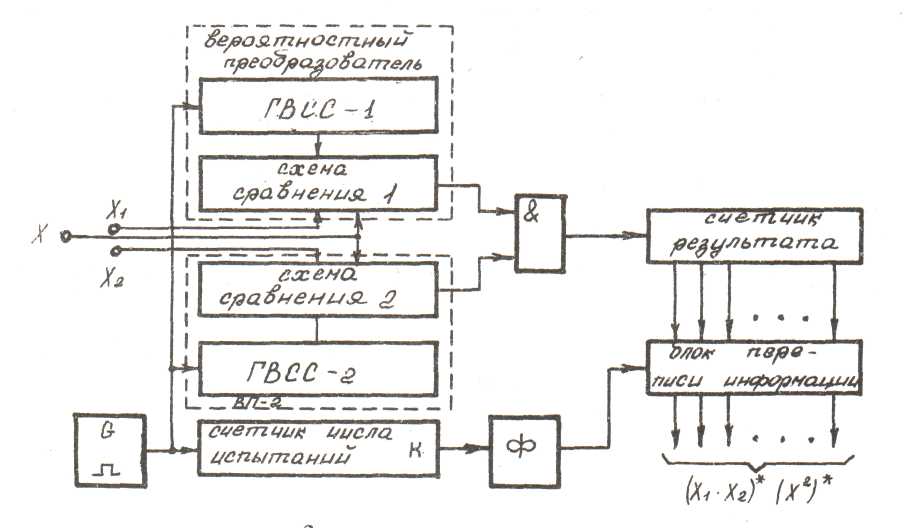
  (2.37)

Рисунок 2.5. - Структурно-функциональная схема вероятностного множительного устройства

Анализируя два последних выражения, приходим к выводу, что для вычисления *Q* вероятностно представленных сомножителей необходим один конъюнктор на *Q* входов, а все остальные блоки изображённые на рис 2.5, носят вспомогательный характер. Сравнивая полученный результат со схемой матричного умножения для перемножения цифровых кодов, приходим к выводу, что с точки зрения затрат оборудования вероятностный умножитель эффективнее в сотнираз.

### 2.4 Язык программирования JavaScript

### JavaScript -  прототипно-ориентированный сценарный язык программирования. Является реализацией языка [ECMAScript](https://ru.wikipedia.org/wiki/ECMAScript" \o "ECMAScript).

### JavaScript может выполняться не только в браузере, а где угодно, нужна лишь специальная программа –интерпретатор. Процесс выполнения скрипта называют «интерпретацией».

Для выполнения программ, не важно на каком языке, существуют два способа: «компиляция» и «интерпретация».

* Компиляция – это когда исходный код программы, при помощи специального инструмента, другой программы, которая называется «компилятор», преобразуется в другой язык, как правило – в машинный код. Этот машинный код затем распространяется и запускается. При этом исходный код программы остаётся у разработчика.
* Интерпретация – это когда исходный код программы получает другой инструмент, который называют «интерпретатор», и выполняет его «как есть». При этом распространяется именно сам исходный код (скрипт). Этот подход применяется в браузерах для JavaScript.

Современные интерпретаторы перед выполнением преобразуют JavaScript в машинный код или близко к нему, оптимизируют, а уже затем выполняют. И даже во время выполнения стараются оптимизировать. Поэтому JavaScript работает очень быстро.

**2.5** **Генераторы псевдослучайных чисел**

При использовании метода статистических испытаний, при моделировании работы вероятностных преобразователей и вероят­ностных операционных узлов на ЭВМ для получения случайных пер­вичных потоков часто используются рекуррентные преобразования. Сравнение различных методов формирования псевдослучайных чисел с равномерным законом распределения мгно­венных значений позволили выбрать метод, предложенный Д.Х. Лемером и заключающийся в решении рекуррентного соотношения (5):

(2.38)



где - некоторое начальное значение;

 - множитель;

 - приращение;

- модуль.

При выборе необходимо соблюдать опреде­ленные правила:

- значение должно вычисляться точно без ошибок в округ-­  
лении ;

- если - степень двойки, то следует выбирать из соот-­  
ношения =5, если m - степень десятки, то из соотношения = = 21, причем , в то же время:

(2.39)



- приращение *с* не должно быть чётно и кратно 5.

Алгоритм, основанный на вычислениях по формуле Лемера, дос­таточно просто реализуется на ПК, однако он мало пригоден для реализации автономного ГПСЧ, так как требует наличия АЛУ. Кроме того, выпол­нение операции умножения многоразрядных чисел снижает быстродей­ствие. Поэтому для построения ГПСЧ вероятностных преобразовате­лей используется метод формирования псевдослучайных чисел с рав­номерным распределением, основанный на использовании линейных последовательностей максимальной длины, в соответствии с которым последовательности максимальной длины фор­мируются *N*-разрядным регистром сдвига с цепью обратной свя­зи через сумматоры по mod.2, при этом период последовательности составляет . Основные свойства последователь­ности:

- число символов "1" в последовательности всегда на единицу  
больше, чем число символов "0";

- серии следующих друг за другом одинаковых символов появляют-­  
ся в последовательности с такой же частотой, как и в случайной  
последовательности равновероятных двоичных символов;

- любой двоичный набор из l<=N смежных элементов встречается  
в последовательности с равной вероятностью   
за исключением набора из l нулей, для которого  так как нулевых наборов в последовательности на единицу меньше;

- нормированная автокорреляционная функция последовательности качественно подобна автокорреляционной функции белого шума, то есть при больших *М* для всех , не кратных *М,* имеем .

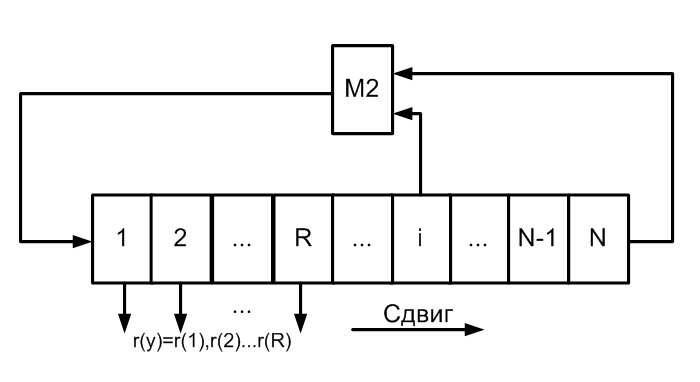


Рисунок 2.6 – Структурная схема последовательного ГПСЧ

**2.6 Основные законы распределения непрерывных случайных величин**

Во многих задачах полезной характеристикой случайной величины является её характеристическая функция. Характеристической функцией случайной величины называется математическое ожидание комплексной случайной величины , рассматриваемое как функции параметра s (здесь и далее в этой части i – мнимая единица). Таким образом, характеристическая функция непрерывной случайной величины задаётся формулой



(2.40)



где – плотность вероятности.



Отметим следующие свойства характеристической функции:

1) при любом действительном значении s характеристическая функция по модулю не превосходит единицы, то есть

(2.41)



2) характеристическая функция равна единицы при , то s=0 есть g(0)=1. Плотность вероятности случайной величины X можно выразить через её характеристическую функцию:



(2.42)



Таким образом, характеристическая функция случайной величины является её полной вероятностной характеристикой. Зная характеристическую функцию случайной величины, можно найти её плотность вероятности, а следовательно, и функцию распределения, то есть полностью определить закон распределения случайной величины. Через характеристическую функцию можно выразить также числовые характеристики случайной величины, в частности её математическое ожидание и дисперсию:

(2.43)



### 2.6.1 Нормальный закон распределения (закон Гаусса)

Плотность вероятности нормально распределённой случайной величины x выражается формулой



. (2.44)

Кривая распределения изображена на рисунке (2.7.а) Она симметрична относительно точки (точка максимума). При уменьшении ордината точки максимума неограниченно возрастает, при этом кривая пропорционально сплющивается вдоль оси абсцисс, так что площадь под её графиком остаётся равной единицы рисунок (2.7.б).

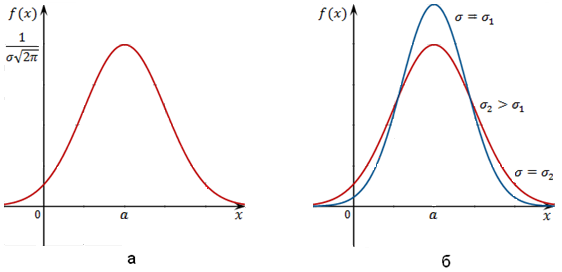


Рисунок 2.7 – Кривая распределения – а, с уменьшением СКО – б

Нормальный закон распределения широко применяется в задачах практики. Объяснить причины этого впервые удалось Ляпунову. Он показал, что если случайная величина может рассматриваться как сумма большого числа малых слагаемых, то при достаточно общих условиях закон распределения этой случайной величины близок к нормальному независимо от того, каковы законы распределения отдельных слагаемых. А так как практически случайные величины в большинстве случаев бывают результатом действия множества причин, то нормальный закон оказывается наиболее распространённым законом распределения. Укажем числовые характеристики нормально распределённой случайной величины (математическое ожидание и дисперсия):

(2.45)



Таким образом, параметры и в выражении (2.45) нормального закона распределения представляют собой математическое ожидание и среднее квадратическое отклонение случайной величины. Принимая это во внимание, формулу (2.44) можно представить следующим образом:



 (2.46)

Эта формула показывает, что нормальный закон распределения полностью определяется математическим ожидание и дисперсией случайной величины. Таким образом, математическое ожидание и дисперсия полностью характеризуют нормально распределённую случайную величину. Разумеется, что в общем случае, когда характер закона распределения неизвестен, знание математического ожидания и дисперсии недостаточно для определения этого закона распределения.

Характеристическая функция нормального распределения случайной величины задаётся формулой:

(2.47)



### 2.6.2 Логарифмически нормальное распределение

Говорят, что случайная величина имеет логарифмически нормальное распределение (сокращённо логнормальное распределение), если её логарифм распределён нормально, то есть если



(2.48)



где величина имеет нормальное распределение с параметрами



Плотность логнормального распределения задаётся формулой

(2.49)



Математическое ожидание и дисперсию логнормального распределения определяют по формулам

(2.50)



Логарифмически нормальное распределение встречается в ряде технических задач. Оно даёт распределение размеров частиц при дроблении, содержаний элементов в минералах в извержённых горных пародах, численности рыб в море и т.д. Встречается такое распределение во всех задачах, где логарифм рассматриваемой величины можно представить в виде суммы большого количества независимых равномерно малых величин:

, (2.51)



где независимы.

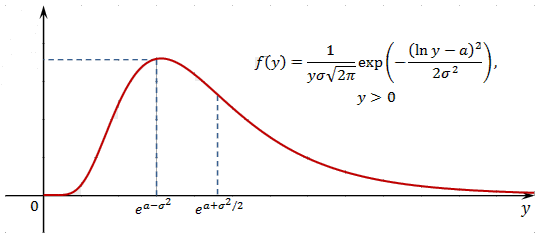


Рисунок 2.8 – Кривая логарифмического нормального распредиления.

2.6.3 Гамма-распределение

Говорят, что случайная величина имеет гамма-распределение с параметрами a>0 и , b>0 если её плотность распределения вероятностей имеет вид



, (2.52)



где .



Математическое ожидание и дисперсия, подчинённые гамма-распределению, задаются формулами

(2.53)



Отметим, что при гамма-распределение имеет моду:



(2.54)



### 2.6.4 Экспоненциальный закон распределения

Экспоненциальным распределением называется частный случай гамма-распределения с параметрами , то есть плотность вероятности в этом случае:



. (2.55)



Используя свойства два плотности распределения, можно найти функцию распределения экспоненциального закона:



. (2.56)



Основные характеристики (математическое ожидание и дисперсия) случайной величины , распределённой по экспоненциальному, имеют вид:



. (2.57)



Характеристическая функция экспоненциального распределения задаётся формулой:

(2.58)



Статистический смысл параметра λ состоит в следующем: λ есть среднее число событий на единицу времени, то есть 1/λ есть средний промежуток времени между двумя последовательными событиями.



Экспоненциальное (показательное) распределение часто встречается в теории массового обслуживания (например, x- время ожидания при техническом обслуживании или x - продолжительность телефонных разговоров, ежедневно регистрируемых на телефонной станции) и теории надёжности (например, x- срок службы радиоэлектронной аппаратуры).

2.6.5 Распределение Вейбула

Случайная величина подчиняется закону распределения Вейбула с параметрами , если её плотность распределения вероятностей записывается в виде:



. (2.59)



Математическое ожидание и мода случайной величины, распределённые по закону Вейбула, имеют следующий вид:

(2.60)



Распределение Вейбула в ряде случаев характеризует срок службы радиоэлектронной аппаратуры и, кроме того, применяется для аппроксимации различных несимметричных распределений в математической статистике.

### 2.6.6 Равномерный закон распределения

Случайная величина x называется распределённой равномерно на отрезке , [a,b] если её плотность распределения вероятностей постоянна на данном отрезке:



(2.61)



Все возможные значения равномерно распределённой случайной величины лежат в пределах некоторого интервала; кроме того.в пределах этого интервала все значения случайной величины одинаково вероятны (обладаю одной и той же плотностью вероятности). Равномерно распределение реализуется в экспериментах, где наудачу ставиться точка на отрезке [a,b] (x- абсцисса поставленной точки). Равномерно распределённая случайная величина встречается также в измерительной практике при округлении отчётов измерительных приборов до целых делений шкал. Ошибка при округлении отчёте до ближайшего целого деления является случайной величиной x, которая может принимать с постоянной плотностью вероятности любое значение между двумя соседними целыми делениями.

Математическое ожидание и дисперсия равномерно распределённой случайной величины:

(2.62)



Характеристическая функция равномерного распределения задаётся формулой:

(2.63)

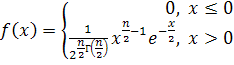
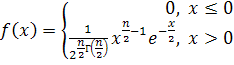


2.6.7 Распределение хи-квадрат

Частный случай гамма-распределения с параметрами и   
 называется распределением хи-квадрат с n степенями свободы (пишут ). Если случайная величина подчиняется закону , то её плотность распределения вероятностей есть:



(2.64)



Основные характеристики распределение хи квадрат (математическое ожидание и дисперсия):

(2.65)



Случайная величина , подчиняющаяся хи-квадрат распределению, равна сумме квадратов n независимых случайных величин   
, каждая из которых имеет стандартизированное нормальное распределение, то есть:



(2.66)



Пусть и - независимые случайные величины, имеющие хи-квадрат распределение со степенью свободы соответственно и . Сумма этих случайных величин имеет также хи-квадрат распределение с +. степенями свободы:



(2.67)



Заметим, что распределение формулы 1.39 при больших значениях n(n>30) с достаточной для практических расчётов точностью аппроксимируется нормальным распределением с математическим ожиданием n и дисперсией 2n. Поэтому при больших значениях n вероятности рассчитываются по нормальному закону.



Распределение играет большую роль в математической статистике.



2.6.8 Распределение Стьюдента

Распределение хи-квадрат Случайная величина есть отношение двух независимых случайных величин и , то есть:



(2.68)



Распределение случайной величины называется распределением Стьюдента с степенями свободы. Его плотность задаётся формулой:



(2.69)



Математическое ожидание и дисперсия случайной величины, подчинённой распределению Стьюдента , есть:



(2.70)



Как и в случае и хи-квадрат распределением, при увеличении распределение Стьюдента стремиться к нормальному, более того, стандартизованному нормальному (то есть с нулевым математическим ожиданием и единичной дисперсией). Распределение Стьюдента, как хи-квадрат распределение, широко применяется в задачах математической обработки измерений.



2.6.9 Распределение Фишера

Пусть случайная величина , равна отношению двух независимых случайных величин и , то есть:

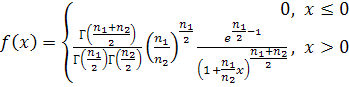
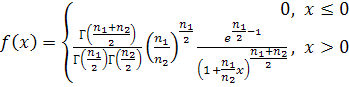


(2.71)



Распределение случайной величины называется распределением Фишера с и степенями свободы. Оно имеет следующую плотность вероятности:

(2.72)



Математическое ожидание случайной величины, подчинённой распределению Фишера, , определяется по формуле:

(2.73)



Между случайными величинами, имеющими нормальное распределение: хи-квадрат, Стьюдента и Фишера, имеют место соотношения:

(2.74)



2.6.10 Бета распределение

Бета распределение в теории вероятностей и статистике - двухпараметрическое семейство абсолютно непрерывных распределений. Используется для описания случайных величин, значения которых ограничены конечным интервалом.

Пусть распределение случайной величины задаётся плотностью вероятности , имеющей вид:



, (2.75)



где произвольные фиксированные параметры и бета-функция.



 (2.76)



Математическое ожидание (1.50) и дисперсия (1.51) случайной величины , имеющей бета-распределение, имеют вид:



(2.77)



(2.78)



**3. ВАРИАНТНЫЙ АНАЛИЗ**

Процедура выбора оптимального варианта системы соответствует общей процедуре проектирования сложных систем управления на основе системотехнического подхода и сводится к решению таких типовых задач, как определение общей структуры системы, организация взаимодействия между подсистемами, учет влияния внешних воздействий, выбор оптимальных структур подсистем и оптимальных алгоритмов функционирования системы. Проектирование ведется исходя из целей создания системы и решаемых ею задач. Оценка соответствия системы поставленным целям и задачам производится по критериям ее качества.

Метод анализа иерархий (МАИ) является систематической процедурой для иерархического представления элементов, определяющих суть любой проблемы. Метод состоит в декомпозиции проблемы на всё более простые составные части и дальнейшей обработки последовательных суждений лица принимающего решение (ЛПР) по парным сравнениям. В результате может быть выражена относительная степень (интенсивность) взаимодействия элементов в иерархии. В результате получаются численные выражения этих суждений. МАИ включает в себя процедуры синтеза множественных суждений, получение приоритетных критериев и нахождение альтернативных решений. Полученные знания являются оценками в шкале отношений и соответствуют жёстким оценкам.

Решение проблемы - процесс поэтапного установления приоритетов.

**3.1 Вариантный анализ вероятностных сумматоров**

Необходимо сделать выбор одного из трех рассматриваемых вариантов. Предварительный просмотр критериев привел к выбору трех альтернатив:

* Система А. Последовательный вероятностный сумматор
* Система Б. Параллельный вероятностный сумматор со случайным переключением входных шин данных
* Система В. Параллельный вероятностный сумматор с проверкой входных данных

Для решения задачи с помощью метода анализа иерархий следует определить критерии, по которым далее будем производить сравнение предложенных систем. Количество критериев должно удовлетворять условию: n=7http://www.techstages.ru/images/books/841/image009.png2.

В результате получаем следующие критерии:

A1 – количество компонентов

А2 – количество итераций

А3 – скорость обработки информации

А4 – точность результата

А5 – обработка одновременно более 2-х операндов

А6 – область применения

Количество критериев равно шести, что говорит о подходящем наборе критериев, для вывода объективного конечного результата.

Представим проблему выбора в виде трехуровневой иерархии, показанной на рисунке 3.1.



Рисунок 3.1 – Трехуровневая схема представления иерархии

Далее расставим веса критериев, основываясь на информации, полученной в ходе испытания моделей. Полученные данные критериев представлены в таблице 3.1.

Таблица 3.1 – Относительные веса критериев

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | А1 | А2 | А3 | А4 | А5 | А6 | Произведение | Вес |
| А1 | 1,00 | 5,00 | 0,17 | 6,00 | 0,20 | 4,00 | 4,00 | 0,026 |
| А2 | 0,20 | 1,00 | 0,13 | 0,17 | 3,00 | 7,00 | 0,09 | 0,001 |
| А3 | 6,00 | 8,00 | 1,00 | 7,00 | 0,11 | 5,00 | 186,67 | 2,926 |
| А4 | 8,00 | 6,00 | 0,14 | 1,00 | 6,00 | 4,00 | 161,28 | 2,528 |
| А5 | 5,00 | 0,33 | 9,00 | 0,16 | 1,00 | 0,17 | 0,40 | 0,005 |
| А6 | 5,00 | 0,14 | 0,20 | 0,25 | 6,00 | 1,00 | 0,21 | 0,002 |
| Сумма | 25,20 | 20,48 | 10,63 | 14,58 | 16,31 | 21,17 |  |  |

Вес критериев получаются как отношение произведения относительных весов критерия по горизонтали, возведенного в степень (где 6 – количество критериев), к сумме относительных весов критерия по вертикали. Числа в таблицах округлены до сотых долей, кроме столбца с весами критериев.

Следующим шагом выполним сравнение альтернатив по каждому критерию отдельно. В таблице 3.2-3.7 представлены данные по системам.

Таблица 3.2 – Оценки по первому критерию

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Система А | Система Б | Система В | K1 |
| Система А | 1 | 5 | 4 | 2,298851 |
| Система Б | 0,2 | 1 | 2 | 0,010256 |
| Система В | 0,25 | 0,5 | 1 | 0,002976 |

Таблица 3.3 – Оценки по второму критерию

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Система А | Система Б | Система В | K1 |
| Система А | 1 | 0,111111 | 0,111111 | 0,000108 |
| Система Б | 9 | 1 | 1 | 0,710526 |
| Система В | 9 | 1 | 1 | 0,710526 |

Таблица 3.4 – Оценки по третьему критерию

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Система А | Система Б | Система В | K3 |
| Система А | 1 | 0,111111 | 0,111111 | 0,000108 |
| Система Б | 9 | 1 | 1 | 0,710526 |
| Система В | 9 | 1 | 1 | 0,710526 |

Таблица 3.5 – Оценки по четвертому критерию

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Система А | Система Б | Система В | K4 |
| Система А | 1 | 4 | 1 | 0,296296296 |
| Система Б | 0,25 | 1 | 0,166667 | 0,000631313 |
| Система В | 1 | 6 | 1 | 0,461538462 |

Таблица 3.6 – Оценки по пятому критерию

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Система А | Система Б | Система В | K5 |
| Система А | 1 | 0,166667 | 0,25 | 0,000631 |
| Система Б | 6 | 1 | 4 | 2,823529 |
| Система В | 4 | 0,25 | 1 | 0,031746 |

Таблица 3.7 – Оценки по пятому критерию

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Система А | Система Б | Система В | K6 |
| Система А | 1 | 0,5 | 0,33333333 | 0,00462963 |
| Система Б | 2 | 1 | 0,5 | 0,047619048 |
| Система В | 3 | 2 | 1 | 0,545454545 |

Для получения результатов необходимо для каждой из рассматриваемых систем просуммировать нормализованные критерии, умноженные на свои веса. В таблице 3.8 представлен результат.

Таблица 3.8 – Результаты

|  |  |
| --- | --- |
|  | Результат |
| Система А | 0,810270888 |
| Система Б | 2,094582021 |
| Система В | 3,248020878 |

В итоге, после проведения вариантного анализа, можно сделать вывод, что система В наилучшим образом подходит в качестве сумматора, использующего данные в вероятностной форме.

**3.2 Вариантный анализ вероятностных множителей**

Необходимо сделать выбор одного из двух рассматриваемых вариантов. Предварительный просмотр критериев привел к выбору двух альтернатив:

* Система А. Параллельный вероятностный умножитель
* Система Б. Параллельный вероятностный умножитель с групповой структурой

Для решения задачи с помощью метода анализа иерархий следует определить критерии, по которым далее будем производить сравнение предложенных систем. Количество критериев должно удовлетворять условию: n=7http://www.techstages.ru/images/books/841/image009.png2.

В результате получаем следующие критерии:

A1 – количество компонентов

А2 – количество итераций

А3 – скорость обработки информации

А4 – точность результата

А5 – обработка одновременно более 2-х операндов

А6 – область применения

Количество критериев равно шести, что говорит о подходящем наборе критериев, для вывода объективного конечного результата.

Представим проблему выбора в виде трехуровневой иерархии, показанной на рисунке 3.2.



Рисунок 3.2 – Трехуровневая схема представления иерархии

Далее основываясь на таблице альтернатив 3.9, выполним сравнение альтернатив по каждому критерию отдельно. В таблице 3.10-3.14 представлены данные по системам.

Таблица 3.9 – Оценки по первому критерию

|  |  |  |  |
| --- | --- | --- | --- |
|  | Система А | Система Б | K1 |
| Система А | 1 | 4 | 0,533333333 |
| Система Б | 0,25 | 1 | 0,008333333 |

Таблица 3.10 – Оценки по второму критерию

|  |  |  |  |
| --- | --- | --- | --- |
|  | Система А | Система Б | K2 |
| Система А | 1 | 0,111111111 | 0,001851852 |
| Система Б | 9 | 1 | 1,35 |

Таблица 3.11 – Оценки по третьему критерию

|  |  |  |  |
| --- | --- | --- | --- |
|  | Система А | Система Б | K3 |
| Система А | 1 | 0,142857143 | 0,00297619 |
| Система Б | 7 | 1 | 1,020833333 |

Таблица 3.12 – Оценки по четвертому критерию

|  |  |  |  |
| --- | --- | --- | --- |
|  | Система А | Система Б | K4 |
| Система А | 1 | 0,5 | 0,027777778 |
| Система Б | 2 | 1 | 0,222222222 |

Таблица 3.13 – Оценки по пятому критерию

|  |  |  |  |
| --- | --- | --- | --- |
|  | Система А | Система Б | K5 |
| Система А | 1 | 2 | 0,222222222 |
| Система Б | 0,50 | 1 | 0,027777778 |

Таблица 3.14 – Оценки по шестому критерию

|  |  |  |  |
| --- | --- | --- | --- |
|  | Система А | Система Б | K6 |
| Система А | 1 | 0,5 | 0,027777778 |
| Система Б | 2 | 1 | 0,222222222 |

Для получения результатов необходимо для каждой из рассматриваемых систем просуммировать нормализованные критерии, умноженные на свои веса. В таблице 3.15 представлен результат.

Таблица 3.15 – Результаты

|  |  |
| --- | --- |
|  | Результат |
| Система А | 0,094127401 |
| Система Б | 3,550871515 |

В итоге, после проведения вариантного анализа, можно сделать вывод, что система Б наиболее эффективна и лучше всего подходит в качестве вероятностного умножителя.

**4 РАЗРАБОТКА МОДЕЛЕЙ УСКОРЕННОГО ВЫПОЛНЕНИЯ ОПЕРАЦИИ СЛОЖЕНИЯ**

**4.1 Вероятностный последовательный сумматор**

Самой простой схемой вероятностного сумматора – является схема вероятностного последовательного сумматора, обрабатывающего вероятностное представление операндов в порядке очереди.На рисунке 4.1 представлена схема данного сумматора (1).



Рисунок 4.1 – вероятностный последовательный сумматор

Представленный выше сумматор состоит из следующих частей:

* Регистр хранения операндов
* Генератор тактовых импульсов
* Преобразователь «число- вероятностное отображение»
* Счетчик числа испытаний
* Счетчик результата
* Блок переписи результата

Регистр хранения операндов – является параллельным регистром и необходим для хранения операндов арифметической операции.

Генератор тактовых импульсов - генерирует электрические импульсы заданной частоты (обычно прямоугольной формы) для синхронизации различных процессов в устройствах.

Счетчик числа испытаний необходим для корректировки результата из-за большого количества статистических испытаний.

Счетчик результата соответственно ведет подсчет результата.

Блок переписи результата корректирует результат в зависимости от значения счетчика числа испытаний.

Стоит более детально рассмотреть преобразователь «число- вероятностное отображение». Данный блок состоит из схемы сравнения и генератора случайных чисел.

Анализ структурных схем вероятностных преобразователей дает основание сделать вывод, что схемы сравнения, используемые в них, подразделяются на:

- аналоговые схемы сравнения по амплитуде и фазе;

- цифровые схемы сравнения двоичных чисел, представленных в ви-  
де позиционных кодов (3).

На рисунке 4.2 представлена функциональная схема одного разряда цифровой схемы сравнения вероятностного преобразователя.



Рисунок 4.2 - Функциональная схема одного разряда цифровой схемы сравнения вероятностного преобразователя.

В зависимости от того, аналоговый или цифровой сигнал преоб­разуется в вероятностный, в состав вероятностного преобразовате­ля должен входить аналоговый либо цифровой ГВСС. В подавляющем большинстве случаев для получения аналогового вспомогательного сигнала используются цифровые ГВСС, нагруженные на соответствую­щие преобразователи (код-напряжение, код-фаза). Поэтому генера­торы случайных сигналов, как правило, строятся на базе цифровых элементов и называются генераторами случайных последовательнос­тей (ГСП), причем элементами последовательностей могут быть двоичные символы или многоразрядные числа. В послед­нем случае ГСП называют генераторами случайных чисел (ГСЧ).

В настоящее время перспективны два основных направления в построении источников (генераторов) случайных чисел (ГСЧ). Первое связано с использованием физических флуктуационных шумов в резисторах, транзисторах и других физических приборах. Второе использует математические методы получения так называемых псевдослучайных чисел, свойства которых близки к случайным. В этом случае ГСЧ называют генераторами псевдослучайных чисел (ГПСЧ). Рассмотрим последовательно оба этих направления, учитывая, что от качества и быстродействия генераторов вспомогательных случайных сигналов зависят основные характеристики вероятностных процессоров.

Большинство средств моделирования, а также языки программирования в настоящее время имеют встроенные генераторы случайных чисел. В нашей работе для ускорения обработки информации и проведения более детального анализа, будем использовать генератор псевдовероятностных чисел. Это позволит повысить точность полученных результатов.

**4.2 Параллельный вероятностный сумматор со случайным переключением входных шин данных**

Суть предложенного метода параллельного вероятностного суммирования заключается в выполнении параллельной операции арифметического сложения вероятностно представленных операндов, при котором сумматор можно рассматривать как «переключатель», который в тактовые моменты времени случайным образом подключает к выходу одну из входных шин, что позволяет значительно ускорить получение результата.

На рисунке 4.3 представлена функциональная схема параллельного вероятностного сумматора со случайным переключением шин (1).



Рисунок 4.3 –Функциональная схема параллельного вероятностного сумматора со случайным переключением шин.

Представленный выше сумматор состоит из следующих частей:

* Генератор псевдослучайных чисел
* Генератор тактовых импульсов
* Счетчик числа испытаний
* Счетчик результата
* Блок переписи результата
* Мультиплексор

Генератор тактовых импульсов - генерирует электрические импульсы заданной частоты (обычно прямоугольной формы) для синхронизации различных процессов в устройствах.

Счетчик числа испытаний необходим для корректировки результата из-за большого количества статистических испытаний.

Счетчик результата соответственно ведет подсчет результата.

Блок переписи результата корректирует результат в зависимости от значения счетчика числа испытаний.

На информационные входы мультиплексора подаются шины с вероятностным отображение операндов. Генератор псевдослучайных чисел передает значение на управляющие входы, которые подключают одну из шин к счетчику результата.

**4.3 Параллельный вероятностный сумматор с проверкой входных данных**

Суть предложенного метода заключается в проверке значения элементов вероятностного отображения. Если значение операции «ИЛИ» равно нулю, на счетчик результата сигнал не поступает. Если на вход элемента или поступает одно истинное значение, значит на счетчик поступит сигнал и запишет результат. Если поступает два истинных значения – в счетчик необходимо записать удвоенные результат. На рисунке 4.4 представлена функциональная схема данного сумматора.



Рисунок 4.4 – Функциональная схемапараллельного вероятностного сумматора с проверкой входных данных

Представленный выше сумматор состоит из следующих частей:

* Генератор тактовых импульсов с задержкой такта
* Счетчик числа испытаний
* Счетчик результата
* Блок переписи результата
* Дешифратор
* Блок задержки сигнала

Генератор тактовых импульсов - генерирует электрические импульсы заданной частоты (обычно прямоугольной формы) для синхронизации различных процессов в устройствах. Данный генератор имеет вход пуска такта.

Счетчик числа испытаний необходим для корректировки результата из-за большого количества статистических испытаний.

Счетчик результата соответственно ведет подсчет результата.

Блок переписи результата корректирует результат в зависимости от значения счетчика числа испытаний.

На вход дешифратора подаются шины вероятностных отображений операндов. По сигналу на выходах дешифратора вырабатывается позиционный код.

Элемент задержки – задерживает сигнал на один такт.

Приведенная выше схема работоспособна для двух операндов. Для большего числа операндов необходимо увеличивать дешифратор и добавлять соответствующие схемы задержки.

**5 РАЗРАБОТКА МОДЕЛЕЙ УСКОРЕННОГО ВЫПОЛНЕНИЯ ОПЕРАЦИИ УМНОЖЕНИЯ**

**5.1 Вероятностный параллельный умножитель**

Суть предложенного метода заключается в параллельной подаче вероятностного отображения операндов на элемент «И». На рисунке 5.1 представлена функциональная схема вероятностного параллельного умножителя.



Рисунок 5.1 - Функциональная схема вероятностного параллельного умножителя

Представленный выше умножитель состоит из следующих частей:

* Генератор тактовых импульсов
* Счетчик числа испытаний
* Счетчик результата
* Блок переписи результата
* Преобразователь «число- вероятностное отображение»

Генератор тактовых импульсов - генерирует электрические импульсы заданной частоты (обычно прямоугольной формы) для синхронизации различных процессов в устройствах.

Счетчик числа испытаний необходим для корректировки результата из-за большого количества статистических испытаний.

Счетчик результата соответственно ведет подсчет результата.

Блок переписи результата корректирует результат в зависимости от значения счетчика числа испытаний.

Преобразователь «число – вероятностное отображение» содержит генератор случайных, равномерно распределенных чисел и схему сравнения.

**5.2 Параллельный вероятностный умножитель с групповой структурой**

Суть предложенного метода заключается в параллельной подаче значения операндов на одноразрядные множители. Данный множитель состоит из двух схем сравнения операндов и элемента «И». На рисунке 5.2 представлена функциональная схемавероятностного умножителя с групповой структурой.



Рисунок 5.2- Вероятностного умножитель с групповой структурой.

Представленный выше умножитель состоит из следующих частей:

* Генератор тактовых импульсов с задержкой сигнала
* Счетчик числа испытаний
* Счетчик результата
* Блок переписи результата
* Одноразрядный вероятностный умножитель
* Параллельный обратный счетчик

Генератор тактовых импульсов - генерирует электрические импульсы заданной частоты (обычно прямоугольной формы) для синхронизации различных процессов в устройствах. Данный генератор имеет вход пуска такта.

Счетчик числа испытаний необходим для корректировки результата из-за большого количества статистических испытаний.

Счетчик результата соответственно ведет подсчет результата.

Блок переписи результата корректирует результат в зависимости от значения счетчика числа испытаний.

Параллельный обратный счетчик имеет число входов, равное количеству одноразрядных множителей в системе. Счетчик имеет декрементный счет и выдает сигнал об окончании счета на специальный выход “0”.

**6 АНАЛИЗ ПОЛУЧЕННЫХ РЕЗУЛЬТАТОВ**

Для проведения анализа предложенных систем необходимо провести опыты на программных моделях. Для каждой системы выполним следующие шаги:

* Проведем пять испытаний с увеличением значения операндов
* Замерим аппаратный объем для каждого опыта
* Проверим точность полученных результатов, в зависимости от количества испытаний
* Запишем количество тактов, затраченных на выполнение операции
* По полученной сводной таблице начертим график
* Подведем итоги

**6.1 Анализ вероятностного последовательного сумматора**

Для исследования вероятностного последовательного сумматора рассмотрим таблицу 6.1 исходных данных для модели.

Таблица 6.1 – Исходные данные

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Опыт 1 | Опыт 2 | Опыт 3 | Опыт 4 | Опыт 5 |
| A | 4 | 12 | 39 | 76 | 91 |
| B | 5 | 15 | 54 | 45 | 89 |

Из таблицы видно, что с каждым опытом значение операндов растет. Это поможет проследить зависимость точности, быстродействия и аппаратного объема от значения операндов.

6.1.1 Опыт первый

В нашу модель занесем исходные данные из таблицы 6.1 и установим значение количества испытаний равное 100. На рисунке 6.1 представлен результат выполнения операции сложения в вероятностном последовательном сумматоре.

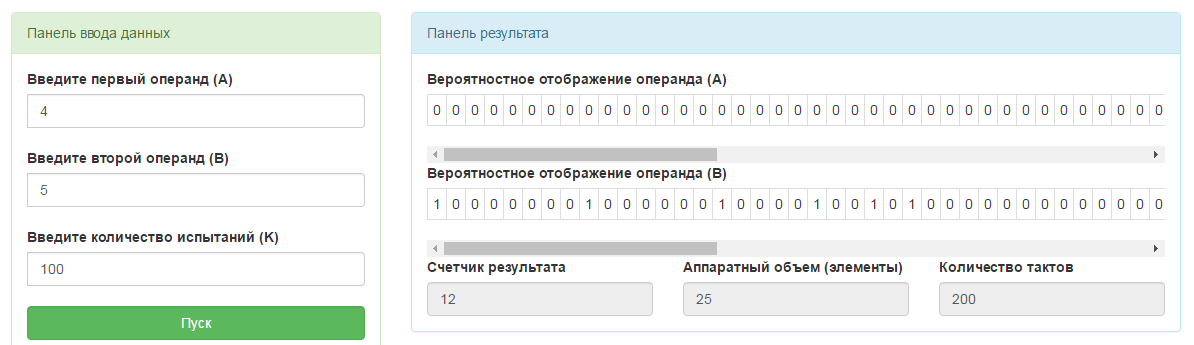


Рисунок 6.1 – Результат первого опыта вероятностного последовательного сумматора

При количестве испытаний равным 100 мы получили абсолютную погрешность в 3 ед. Данная точность нам не подходит, поэтому необходимо увеличить количество испытаний до 1000. На рисунке 6.2 представлен результат работы модели при 1000 статистических испытаний.

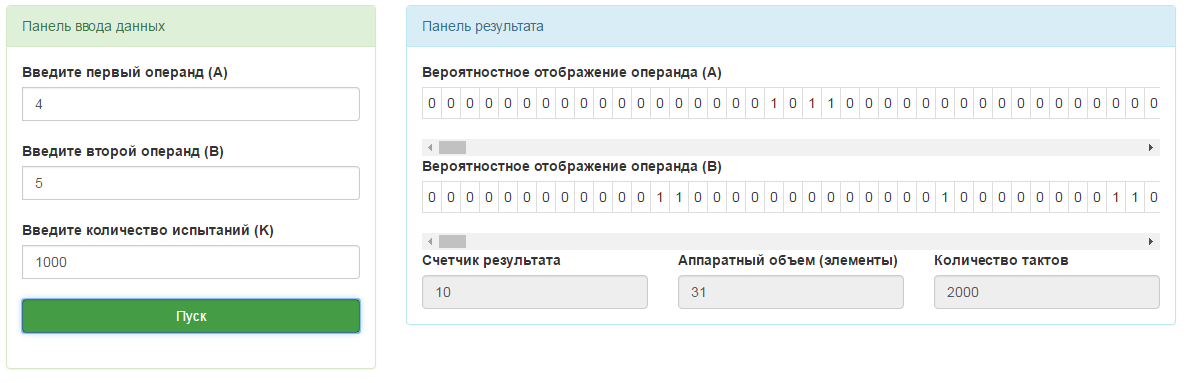


Рисунок 6.2 – Результат первого опыта вероятностного последовательного сумматора при количестве статистических испытаний в 1000

Как видно из рисунка 6.2 результат снова не точен, поэтому необходимо увеличить количество статистических испытаний до 10 000. На рисунке 6.3 представлен результат работа последовательного вероятностного сумматора для количества статистических испытаний равным 10 000.

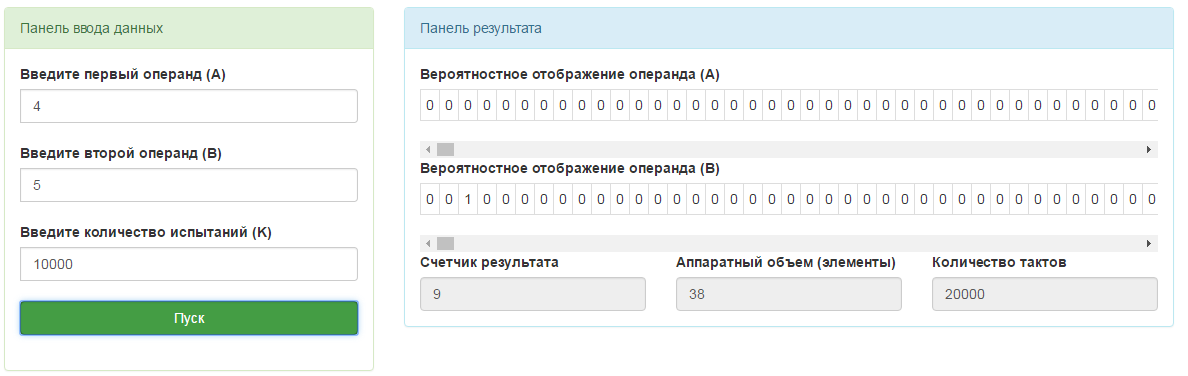


Рисунок 6.3 – Результат первого опыта вероятностного последовательного сумматора при количестве статистических испытаний в 10 000

В данном случае мы получили верный результат. В дальнейшем для всех моделей будем брать количество статистических испытаний равным 10 000, так как вероятностное отображение операндов позволяет получить правильный результат.

Стоит также отметить, что с увеличением статистических испытаний – увеличивается и аппаратный объем систем.

6.1.2 Опыт второй

Зададим новые значения операндов из таблицы 6.1. На рисунке 6.4 представлен результат работы модели.

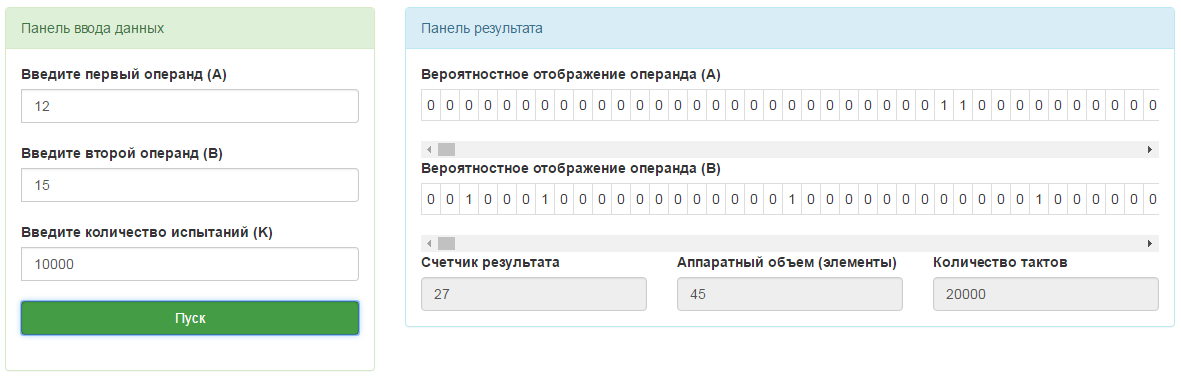


Рисунок 6.4 – Результат работы последовательного вероятностного сумматора во втором опыте

Как видно из результатов аппаратный объем вырос, а количество тактов осталось неизменным.

6.1.3 Опыты с третьего по пятый

Следующим шагом поочередно введем в модель значения всех оставшихся опытов. На рисунках 6.5-6.7 представлены результаты.

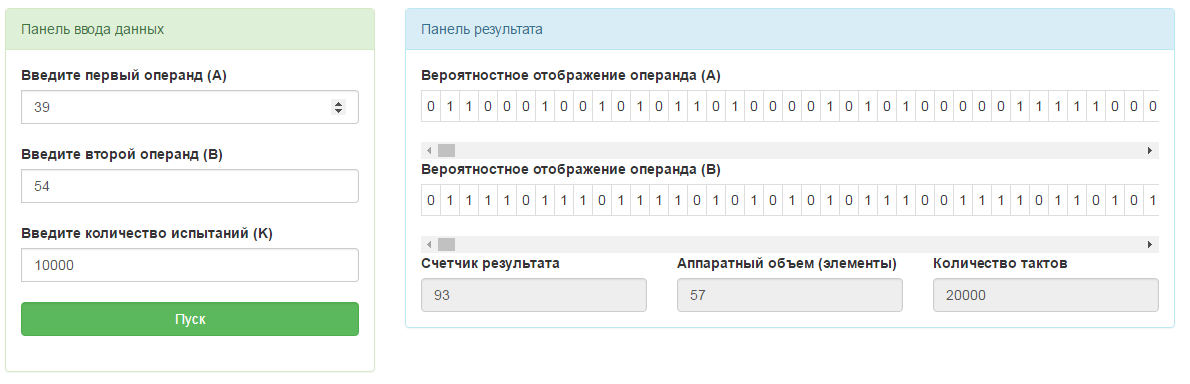


Рисунок 6.5 – Результат работы последовательного вероятностного сумматора в третьем опыте

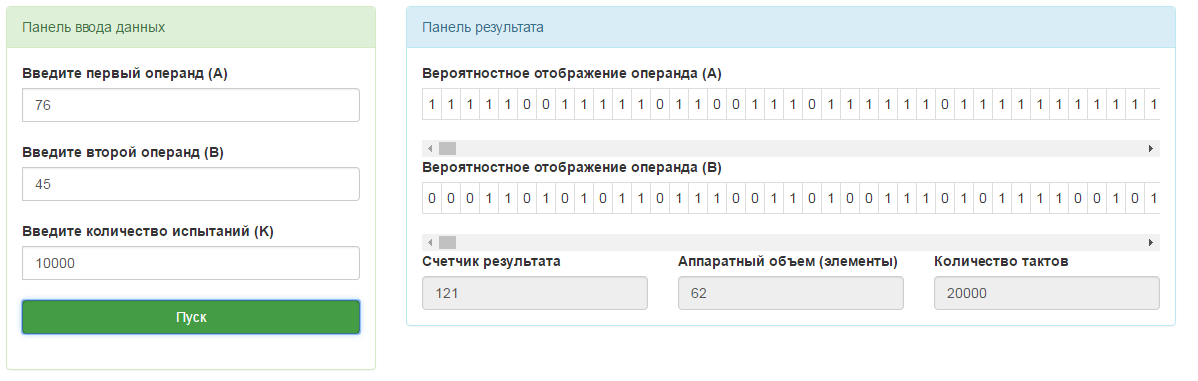


Рисунок 6.6 – Результат работы последовательного вероятностного сумматора в четвертом опыте

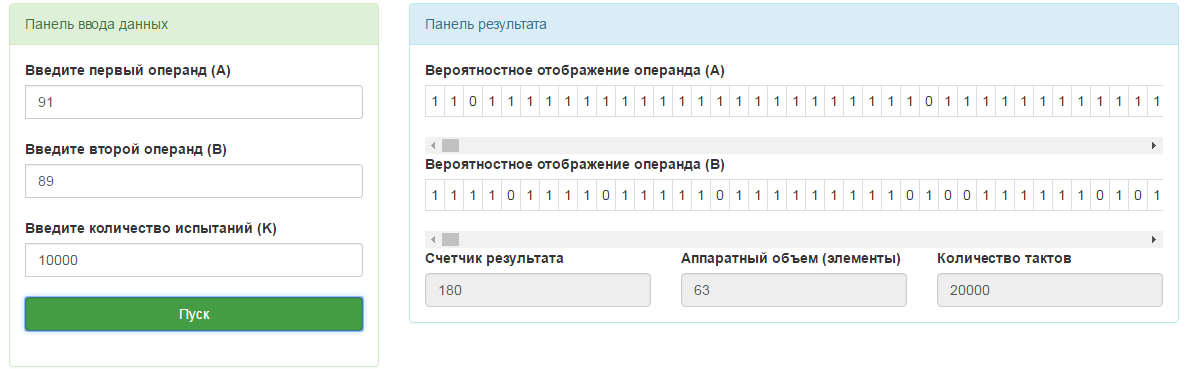


Рисунок 6.7 – Результат работы последовательного вероятностного сумматора в пятом опыте

6.1.4 Результат испытаний

Следующим шагом занесем всю полученную информацию в сводную таблицу 6.2.

Таблица 6.2 – Сводная таблица результата работы последовательного вероятностного сумматора

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Опыт 1 | Опыт 2 | Опыт 3 | Опыт 4 | Опыт 5 |
| A | 4 | 12 | 39 | 76 | 91 |
| B | 5 | 15 | 54 | 45 | 89 |
| Аппаратный объем | 38 | 45 | 57 | 62 | 63 |
| Количество тактов | 20000 | 20000 | 20000 | 20000 | 20000 |

Можно сделать вывод, что количество тактов напрямую зависит от количества статистических испытаний и не зависит от значений операнда.

Построим график зависимости аппаратного объема от значений операндов. На рисунке 6.8 представлен данный график.

Рисунок 6.8 – График зависимости аппаратного объема от значения операндов

Как видно из графика 6.8 аппаратный объем растет от значений операндов. Наибольший интерес представляют четвертый и пятый опыты, где количество элементов практически равно. Из этого можно сделать вывод, что рост аппаратного объема происходит за счет увеличения регистров хранения операндов и размера счетчика результата.

6.1.5 Вывод

В ходе анализа вероятностного последовательного сумматора была выявлена прямая зависимость аппаратного объема от значений операндов. Также было принято решение об использовании количества статистических испытаний равное 10 000, так как это дает приемлемую погрешность.

**6.2 Анализ вероятностного параллельного сумматора**

Для анализа вероятностного параллельного сумматора также воспользуемся исходными данными из таблицы 6.1. Количество статистических испытаний возьмем равным 10 000.

6.2.1 Опыт первый

В модель сумматора внесем исходные данные из таблицы 6.1 и установим значение количества испытаний равное 10 000. На рисунке 6.9 представлен результат выполнения операции сложения в вероятностном параллельном сумматоре.

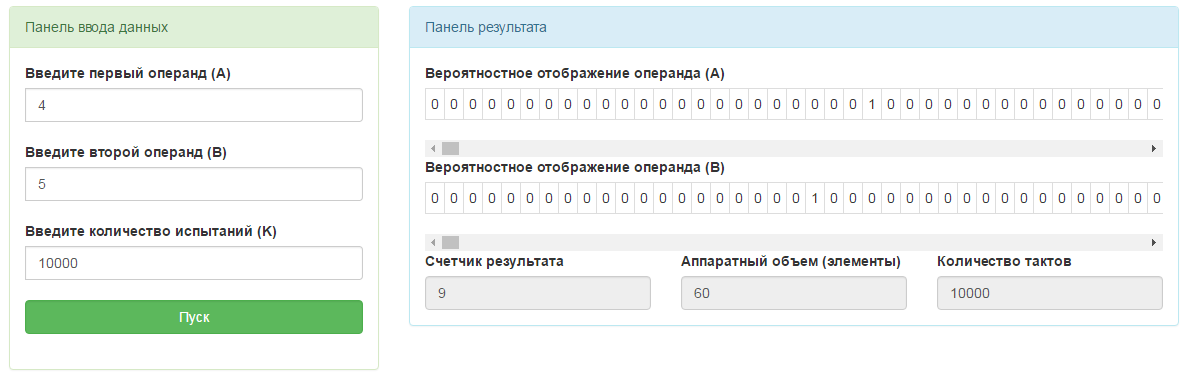


Рисунок 6.9 – Результат первого испытания вероятностного параллельного сумматора

После испытания модели мы получили верный результат. Аппаратный объем в отличии он последовательного сумматора вырос, однако быстродействие выросло в два раза.

6.2.2 Опыт второй

Зададим новые значения операндов из таблицы 6.1. На рисунке 6.10 представлен результат работы модели.

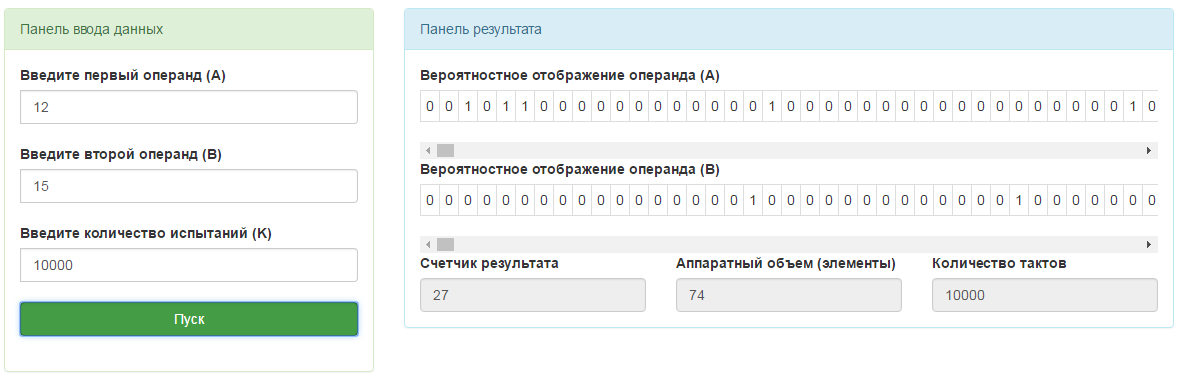


Рисунок 6.10 – Результат работы параллельного вероятностного сумматора во втором опыте

Как и в случае с последовательным сумматором, количество тактов для выполнение арифметической операции напрямую зависит от количества статистических испытаний. Аппаратный объем также растет от значения операндов.

6.2.3 Опыты с третьего по пятый

Следующим шагом поочередно введем в модель значения всех оставшихся опытов. На рисунках 6.11-6.13 представлены результаты.

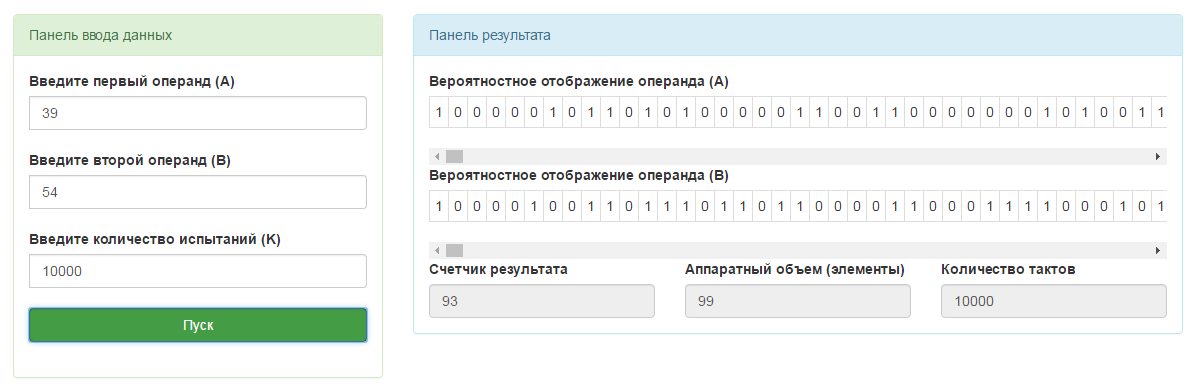


Рисунок 6.11 – Результат работы параллельного вероятностного сумматора в третьем опыте

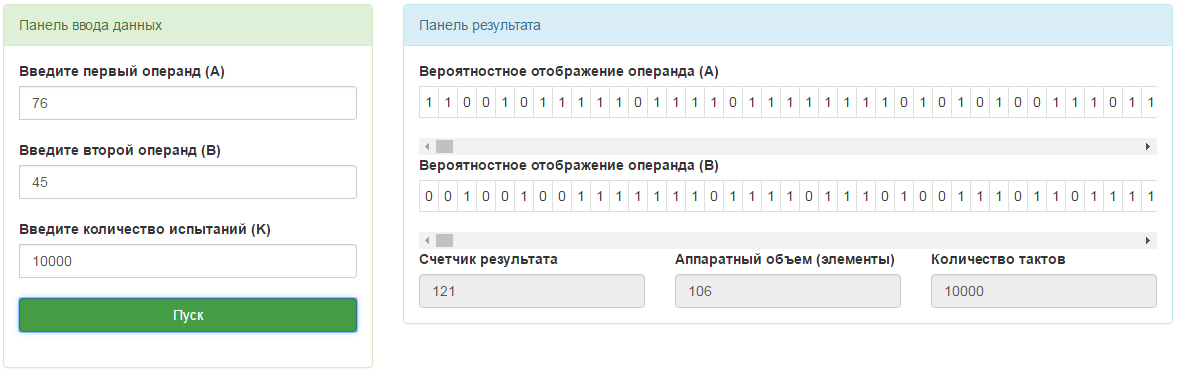


Рисунок 6.12 – Результат работы параллельного вероятностного сумматора в четвертом опыте

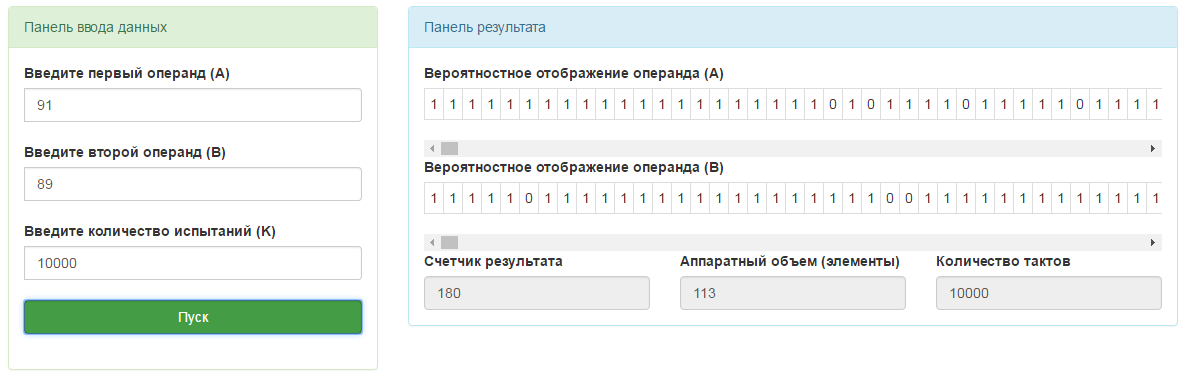


Рисунок 6.13 – Результат работы параллельного вероятностного сумматора в пятом опыте

6.2.4 Результат испытаний

Следующим шагом занесем всю полученную информацию в сводную таблицу 6.3.

Таблица 6.3 – Сводная таблица результата работы параллельного вероятностного сумматора

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Опыт 1 | Опыт 2 | Опыт 3 | Опыт 4 | Опыт 5 |
| A | 4 | 12 | 39 | 76 | 91 |
| B | 5 | 15 | 54 | 45 | 89 |
| Аппаратный объем | 60 | 74 | 99 | 106 | 113 |
| Количество тактов | 10000 | 10000 | 10000 | 10000 | 10000 |

Можно сделать вывод, что количество тактов напрямую зависит от количества статистических испытаний и не зависит от значений операнда.

Построим график зависимости аппаратного объема от значений операндов. На рисунке 6.14 представлен данный график.

Рисунок 6.14 – График зависимости аппаратного объема от значения операндов

6.2.5 Вывод

В ходе анализа вероятностного параллельного сумматора была выявлена прямая зависимость аппаратного объема от значений операндов. В отличии от последовательного сумматора, количество тактов необходимых для выполнения операции сложения уменьшилось в два раза, что дает огромное преимущество в быстродействии всей системы.

**6.3 Вероятностный параллельный сумматор с проверкой входных значений**

Следующим шагом стоит рассмотреть еще один метод, в котором входные последовательности вероятностного отображения операндов просматриваются дешифратором.

Для анализа вероятностного параллельного сумматора с проверкой входных значений также воспользуемся исходными данными из таблицы 6.1. Количество статистических испытаний возьмем равным 10 000.

6.3.1 Опыт первый

В модель сумматора внесем исходные данные из таблицы 6.1 и установим значение количества испытаний равное 10 000. На рисунке 6.15 представлен результат выполнения операции сложения в вероятностном параллельном сумматоре с проверкой входных значений.

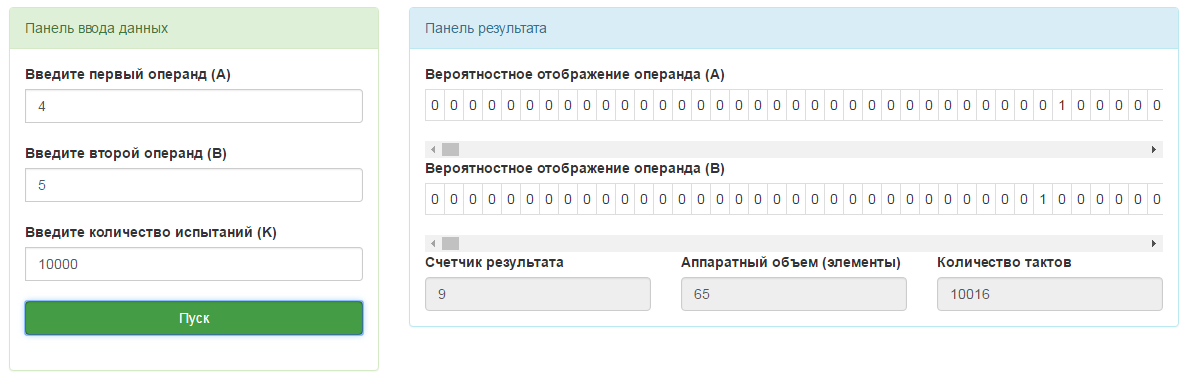


Рисунок 6.15 – Результат первого испытания вероятностного параллельного сумматора с проверкой входных значений

После испытания модели мы получили верный результат. Данный сумматор имеет самый большой аппаратный объем, среди выбранных моделей.

6.3.2 Опыт второй

Зададим новые значения операндов из таблицы 6.1. На рисунке 6.16 представлен результат работы модели.

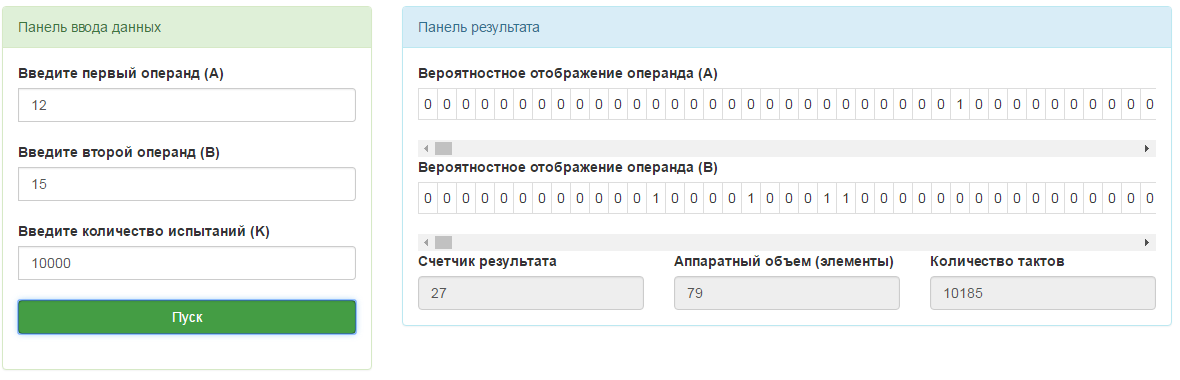


Рисунок 6.16 – Результат работы параллельного вероятностного сумматора с проверкой входных значений во втором опыте

Заметим, что у данного сумматора количество необходимых для выполнения операции сложения тактов не зависит только от количества статистических испытаний.

6.3.3 Опыты с третьего по пятый

Следующим шагом поочередно введем в модель значения всех оставшихся опытов. На рисунках 6.17-6.19 представлены результаты.

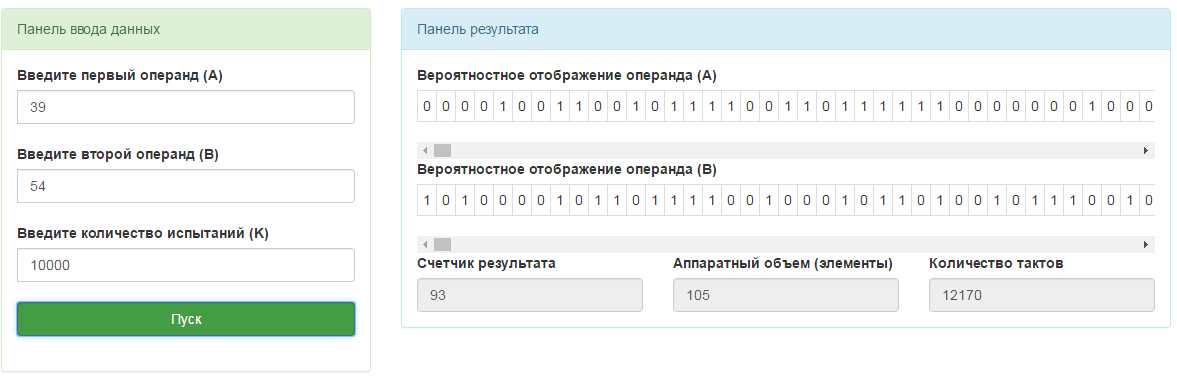


Рисунок 6.17 – Результат работы параллельного вероятностного сумматора с проверкой входных значений в третьем опыте

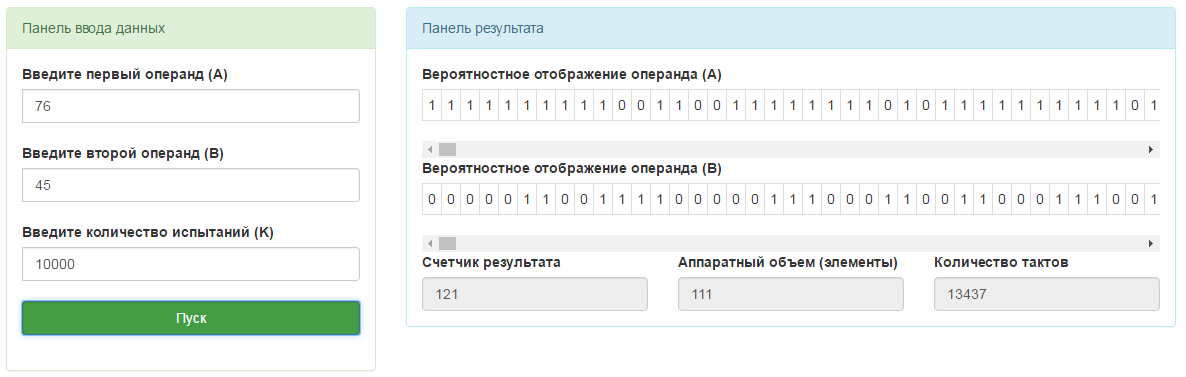


Рисунок 6.18 – Результат работы параллельного вероятностного сумматора с проверкой входных значений в четвертом опыте

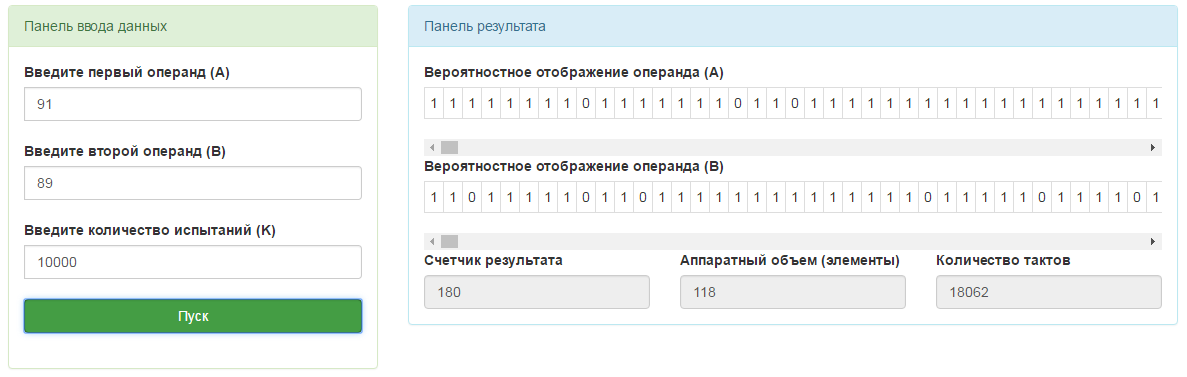


Рисунок 6.19 – Результат работы параллельного вероятностного сумматора с проверкой входных значений в пятом опыте

6.3.4 Результат испытаний

Следующим шагом занесем всю полученную информацию в сводную таблицу 6.4.

Таблица 6.4 – Сводная таблица результата работы параллельного вероятностного сумматора с проверкой входных значений

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Опыт 1 | Опыт 2 | Опыт 3 | Опыт 4 | Опыт 5 |
| A | 4 | 12 | 39 | 76 | 91 |
| B | 5 | 15 | 54 | 45 | 89 |
| Аппаратный объем | 65 | 79 | 105 | 111 | 118 |
| Количество тактов | 10016 | 10185 | 12170 | 13473 | 18062 |

Построим график зависимости аппаратного объема от значений операндов. На рисунке 6.20 представлен данный график.

Рисунок 6.20 – График зависимости аппаратного объема от значения операндов

Так как количество тактов в каждом опыте имеет разное количество, стоит также построить график зависимости количества необходимых для выполнения операции сложения тактов от значений операндов, представленный на рисунке 6.21.

Рисунок 6.21 – График зависимости количество необходимых для выполнения операции сложения тактов от значений операндов

6.3.5 Вывод

В ходе анализа вероятностного параллельного сумматора с проверкой входных значений были сделаны следующие выводы:

* Аппаратный объем растет с увеличением веса операндов.
* Количество необходимых для выполнения операции сложения тактов растет с приближением веса операндов к максимально допустимому весу.

**6.4 Сравнение предложенных моделей сумматоров**

Последним пунктом необходимо провести сравнительный анализ всех трех моделей по аппаратному объему и количеству тактов для выполнения операции сложения. На рисунке 22 представлен график сравнения аппаратного объема в интервале от 0 до 100 по всем трем моделям сумматоров.

Рисунок 22 - График сравнения аппаратного объема по трем моделям сумматоров.

Для полной картины построим график сравнения количества необходимых для выполнения операции сложения тактов по всем трем моделям в интервале значений операндов от 0 до 100. На рисунке 23 представлен данный график.

Рисунок 23 - График сравнения количества необходимых для выполнения операции сложения тактов по трем моделям

Можно сделать выводы, что последовательный вероятностный сумматор имеет самый малый аппаратный объем, при наименьшем показателе быстродействия системы.

Параллельный вероятностный сумматор имеет гораздо больший аппаратный объем, однако количество тактов, затраченных на выполнение арифметической операции в два раза меньше. Минусом данной системы является его зависимость не только от точности закона распределения при преобразовании операндов в вероятностное отображение, но и от точности ГПСЧ, который используется в операции сложения.

Параллельный вероятностный сумматор с проверкой входных значений имеет наибольший аппаратный объем и эффективен только при весе операндов намного меньшем максимально возможного веса.

**6.5 Анализ вероятностного параллельного умножителя**

Для исследования вероятностного параллельного умножителя рассмотрим таблицу 6.5 исходных данных для модели.

Таблица 6.5 – Исходные данные

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Опыт 1 | Опыт 2 | Опыт 3 | Опыт 4 | Опыт 5 |
| A | 10 | 12 | 39 | 65 | 87 |
| B | 6 | 15 | 47 | 59 | 92 |

Из таблицы видно, что с каждым опытом значение операндов растет. Это поможет проследить зависимость точности, быстродействия и аппаратного объема от значения операндов.

6.5.1 Опыт первый

В нашу модель занесем исходные данные из таблицы 6.5 и установим значение количества испытаний равное 100. На рисунке 6.24 представлен результат выполнения операции умножения в вероятностном параллельном умножителе.

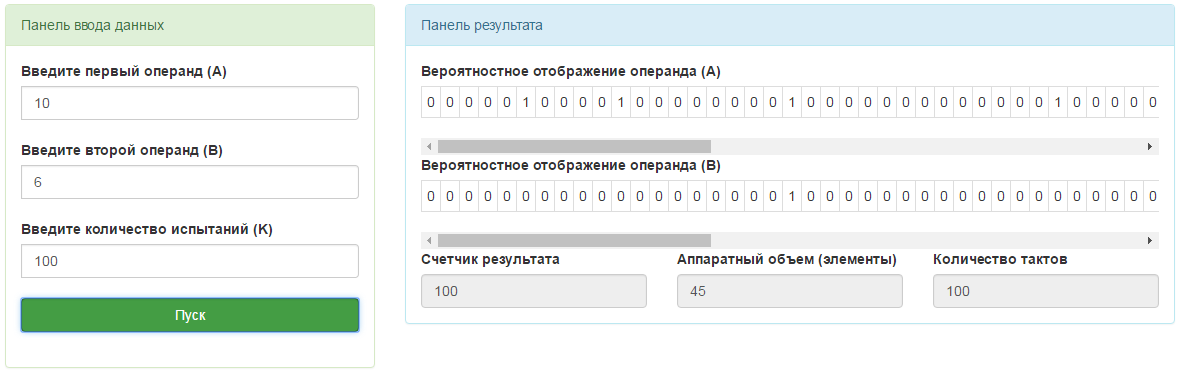


Рисунок 6.24 – Результат первого опыта вероятностного параллельного умножителя

При количестве испытаний равным 100 мы получили абсолютную погрешность в 60 ед. Данная точность нам не подходит, поэтому необходимо увеличить количество испытаний до 10 000. На рисунке 6.25 представлен результат работы модели при 10 000 статистических испытаний.

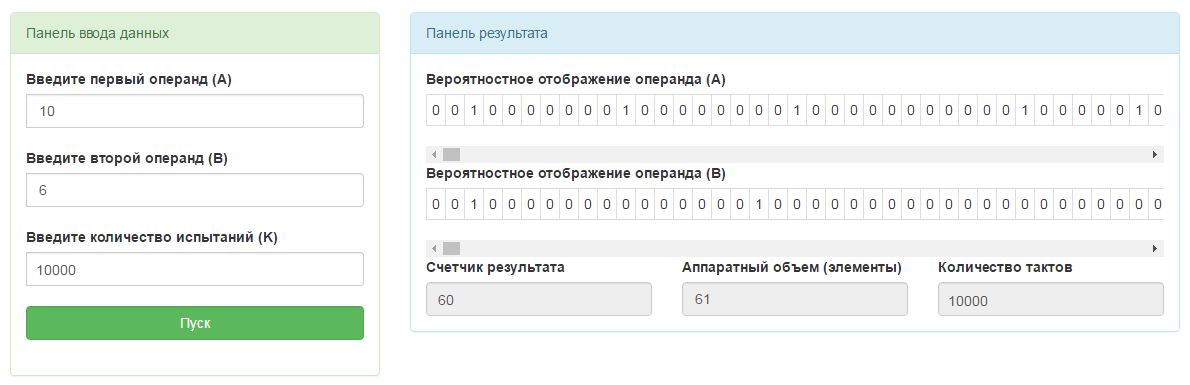


Рисунок 6.25 – Результат первого опыта вероятностного параллельного умножителя при количестве статистических испытаний в 10 000

В данном случае мы получили верный результат. В дальнейшем для всех моделей будем брать количество статистических испытаний равным 10 000, так как вероятностное отображение операндов позволяет получить правильный результат.

Стоит также отметить, что с увеличением статистических испытаний – увеличивается и аппаратный объем систем.

6.5.2 Опыт второй

Зададим новые значения операндов из таблицы 6.5. На рисунке 6.26 представлен результат работы модели.

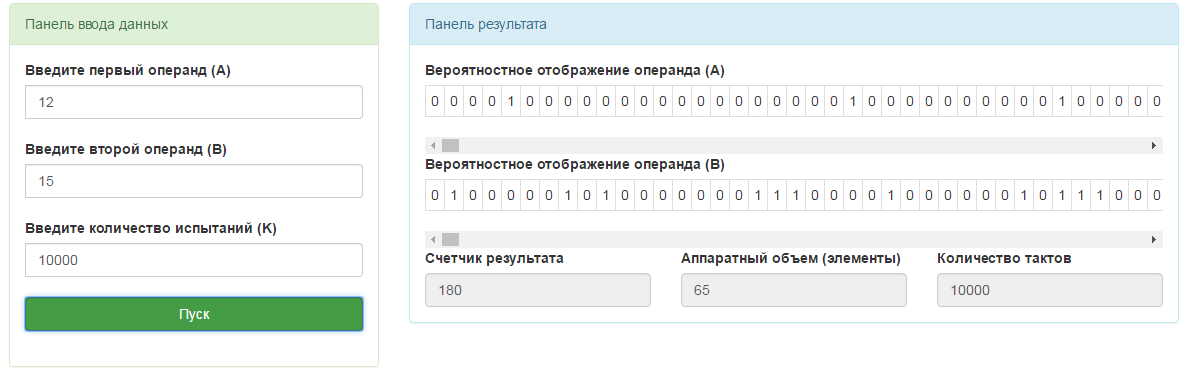


Рисунок 6.26 – Результат работы параллельного вероятностного умножителя во втором опыте

Как видно из результатов аппаратный объем вырос, а количество тактов осталось неизменным.

6.5.3 Опыты с третьего по пятый

Следующим шагом поочередно введем в модель значения всех оставшихся опытов. На рисунках 6.27-6.29 представлены результаты.

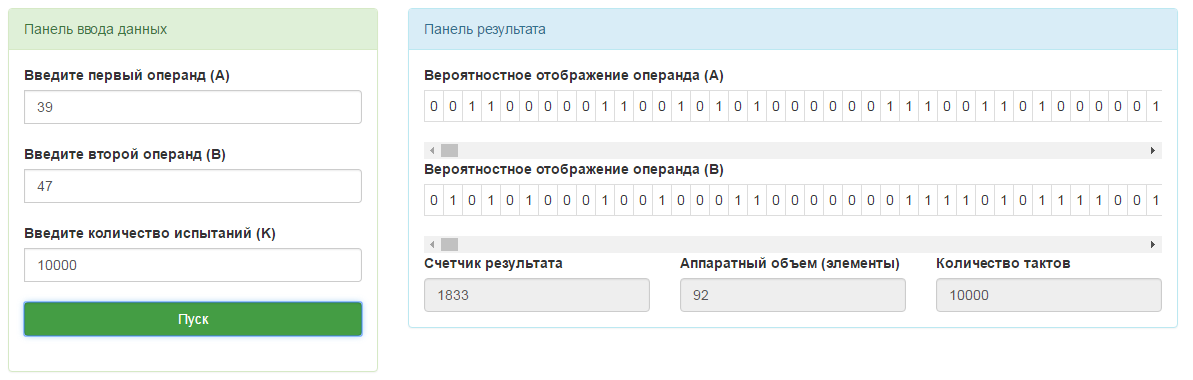


Рисунок 6.27 – Результат работы параллельного вероятностного умножителя в третьем опыте

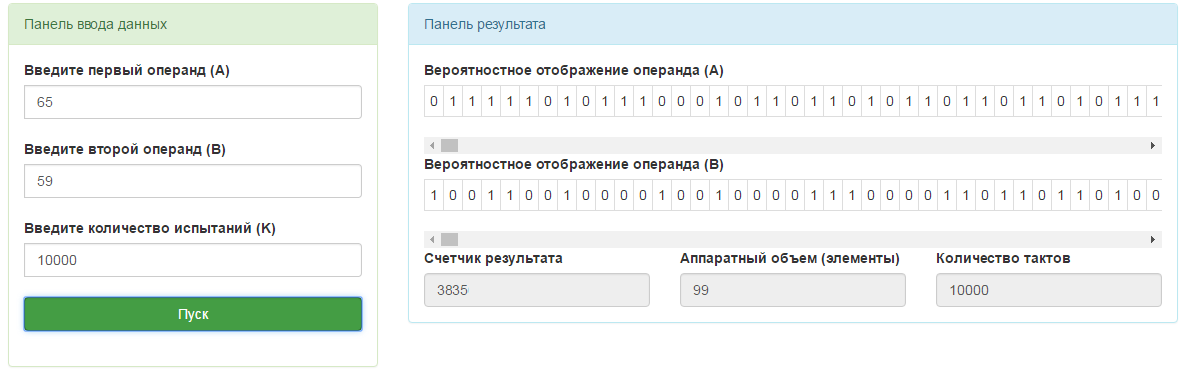


Рисунок 6.28 – Результат работы параллельного вероятностного умножителя в четвертом опыте

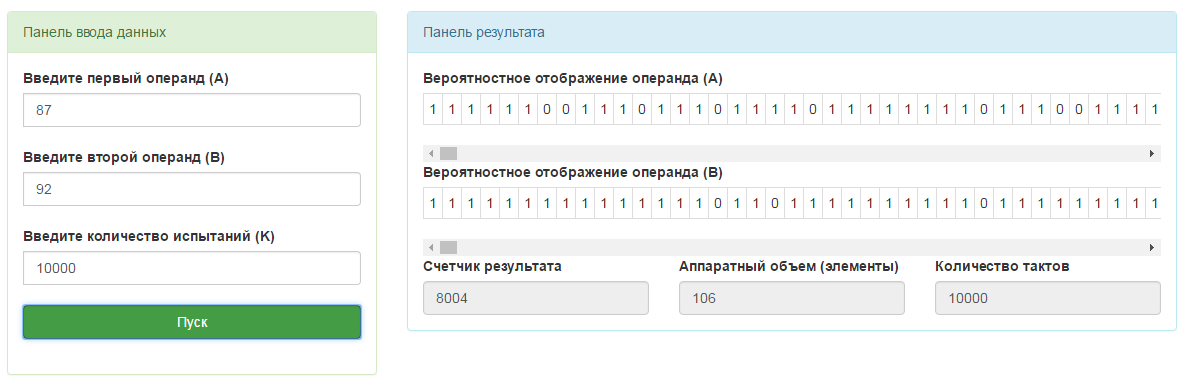


Рисунок 6.29 – Результат работы параллельного вероятностного умножителя в пятом опыте

6.5.4 Результат испытаний

Следующим шагом занесем всю полученную информацию в сводную таблицу 6.6.

Таблица 6.6 – Сводная таблица результата работы параллельного вероятностного умножителя

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Опыт 1 | Опыт 2 | Опыт 3 | Опыт 4 | Опыт 5 |
| A | 10 | 12 | 39 | 65 | 87 |
| B | 6 | 15 | 47 | 59 | 92 |
| Аппаратный объем | 61 | 65 | 92 | 99 | 106 |
| Количество тактов | 10000 | 10000 | 10000 | 10000 | 10000 |

Можно сделать вывод, что количество тактов напрямую зависит от количества статистических испытаний и не зависит от значений операндов.

Построим график зависимости аппаратного объема от значений операндов. На рисунке 6.30 представлен данный график.

Рисунок 6.30 – График зависимости аппаратного объема от значения операндов

Как видно из графика 6.30 аппаратный объем растет от значений операндов.

6.2.5 Вывод

В ходе анализа вероятностного параллельного умножителя была выявлена прямая зависимость аппаратного объема от значений операндов. Также было принято решение об использовании количества статистических испытаний равное 10 000, так как это дает приемлемую погрешность.

**6.6 Анализ вероятностного группового умножителя**

Для исследования вероятностного группового умножителя будем использовать таблицу 6.6 с исходными данными.

6.6.1 Опыт первый

В нашу модель занесем исходные данные из таблицы 6.5 и установим значение количества испытаний равное 10 000. На рисунке 6.31 представлен результат выполнения операции умножения в вероятностном групповом умножителе.

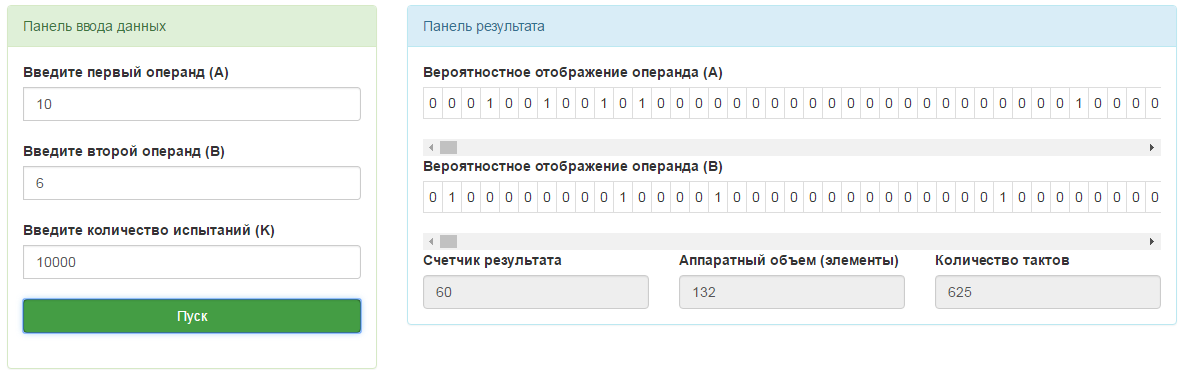


Рисунок 6.31 – Результат первого опыта вероятностного группового умножителя

Стоит отметить, что аппаратный объем намного выше, чем у параллельного умножителя, однако быстродействие значительно выросло.

6.6.2 Опыт второй

Зададим новые значения операндов из таблицы 6.5. На рисунке 6.32 представлен результат работы модели.

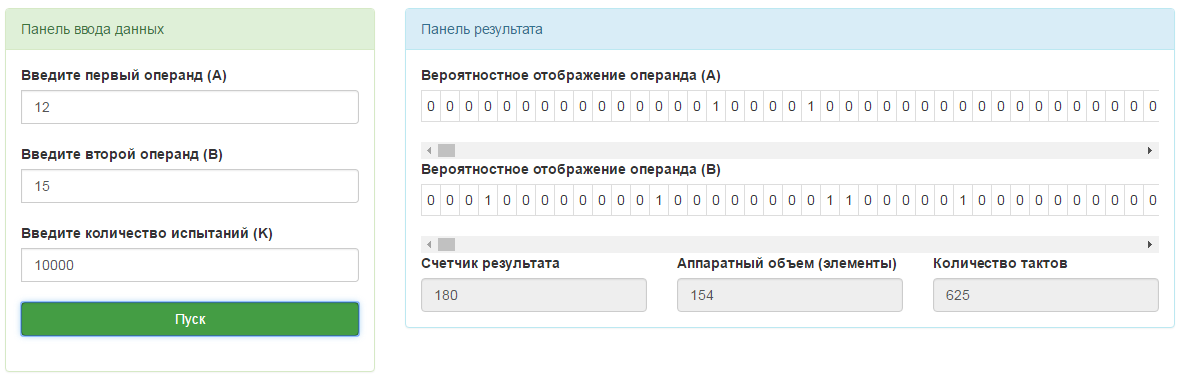


Рисунок 6.32 – Результат работы группового вероятностного умножителя во втором опыте

Как видно из результатов аппаратный объем вырос, а количество тактов осталось неизменным.

6.5.3 Опыты с третьего по пятый

Следующим шагом поочередно введем в модель значения всех оставшихся опытов. На рисунках 6.33-6.35 представлены результаты.

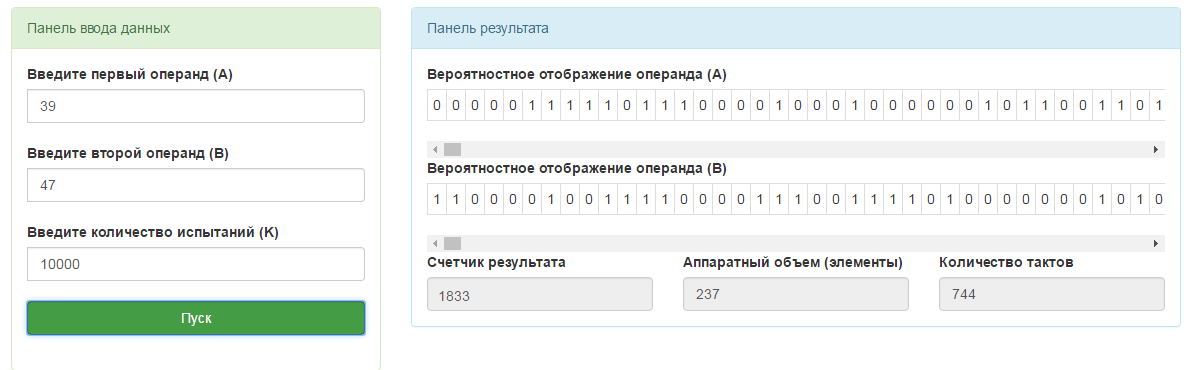


Рисунок 6.33 – Результат работы группового вероятностного умножителя в третьем опыте

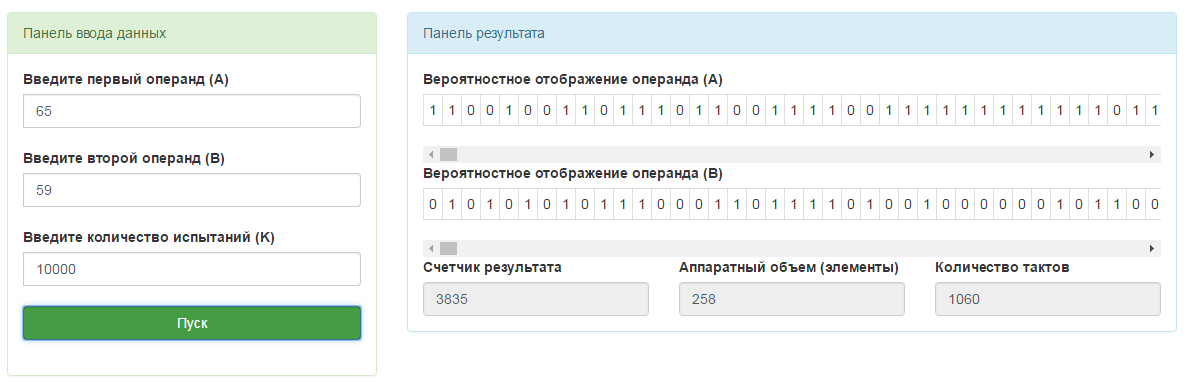


Рисунок 6.34 – Результат работы группового вероятностного умножителя в четвертом опыте

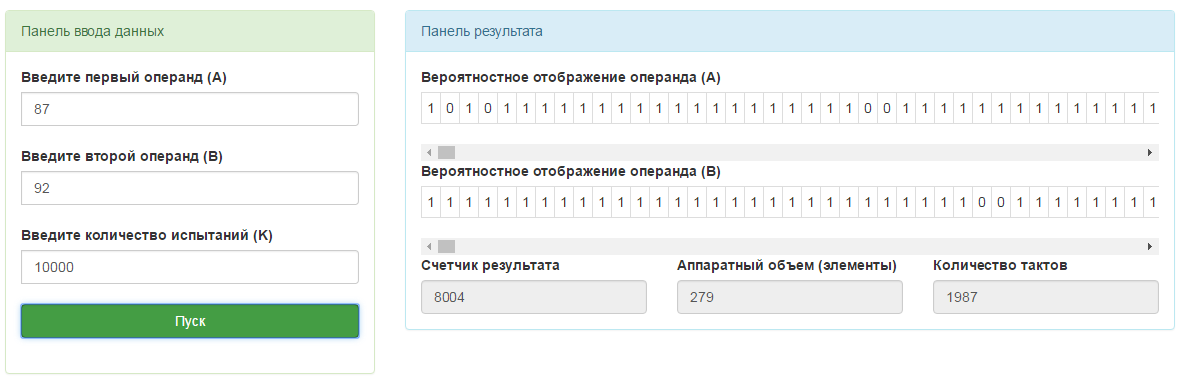


Рисунок 6.35 – Результат работы группового вероятностного умножителя в пятом опыте

6.6.4 Результат испытаний

Следующим шагом занесем всю полученную информацию в сводную таблицу 6.7.

Таблица 6.7 – Сводная таблица результата работы гурппового вероятностного умножителя

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Опыт 1 | Опыт 2 | Опыт 3 | Опыт 4 | Опыт 5 |
| A | 10 | 12 | 39 | 65 | 87 |
| B | 6 | 15 | 47 | 59 | 92 |
| Аппаратный объем | 135 | 154 | 237 | 258 | 279 |
| Количество тактов | 625 | 625 | 744 | 1060 | 1987 |

Можно сделать вывод, что количество тактов зависит не только от количества статистических испытаний, но и от значения операндов.

Построим график зависимости аппаратного объема от значений операндов. На рисунке 6.36 представлен данный график.

Рисунок 6.36 – График зависимости аппаратного объема от значения операндов

Как видно из графика 6.36 аппаратный объем растет от значений операндов.

Далее построим график зависимости количества тактов для выполнения операции умножения от значения операндов. Данный график представлен на рисунке 37.

Рисунок 6.37 – График зависимости количество необходимых для выполнения операции умножения тактов от значений операндов

6.6.5 Вывод

В ходе анализа вероятностного группового умножителя были сделаны следующие выводы:

* Аппаратный объем растет с увеличением веса операндов.
* Количество необходимых для выполнения операции сложения тактов растет с приближением веса операндов к максимально допустимому весу.

**6.7 Сравнение предложенных моделей умножителей**

Последним пунктом необходимо провести сравнительный анализ всех моделей по аппаратному объему и количеству тактов для выполнения операции умножения. На рисунке 38 представлен график сравнения аппаратного объема в интервале от 0 до 100 по всем моделям сумматоров.

Рисунок 38 - График сравнения аппаратного объема вероятностных умножителей

Для полной картины построим график сравнения количества необходимых для выполнения операции умножения тактов по всем моделям в интервале значений операндов от 0 до 100. На рисунке 39 представлен данный график.

Рисунок 39 - График сравнения количества необходимых для выполнения операции умножения тактов

Можно сделать выводы, что параллельный вероятностный умножитель имеет самый малый аппаратный объем. Однако групповой умножитель имеет преимущество перед параллельным в быстродействии, так как позволяет уменьшить в 4 раза количество необходимых для выполнения операции умножения тактов.

После проведения анализа, можно сделать вывод, что количество одноразрядных умножителей в групповом должно быть кратно четырем.

**ЗАКЛЮЧЕНИЕ**

В результате проделанной работы были проанализированы основные литературные источники по вопросам вероятностного и псевдовероятностного представления данных, реализации арифметических операций в вероятностном представлении, методы ускорения арифметических операций.

При помощи среды программировании JavaScript была реализована программа, листинг которой представлен в приложении А, позволяющая оценить преимущества моделей вероятностных сумматоров и умножителей с ускоренным выполнением арифметических операций.

Был проведен вариантный анализ вероятностных сумматоров и вероятностных умножителей. Были выбраны следующие критерии:

* количество компонентов
* количество итераций
* скорость обработки информации
* точность результата
* обработка одновременно более 2-х операндов
* область применения

Вариантный анализ показал, что вероятностный групповой умножитель и вероятностный сумматор с проверкой входных значений имеет преимущества перед другими вероятностными моделями при определенных условиях.

Также был проведен анализ результатов работы моделей вероятностных сумматоров и умножителей. Можно сделать вывод, что последовательный вероятностный сумматор имеет самый малый аппаратный объем, при наименьшем показателе быстродействия системы.

Параллельный вероятностный сумматор имеет гораздо больший аппаратный объем, однако количество тактов, затраченных на выполнение арифметической операции в два раза меньше. Минусом данной системы является его зависимость не только от точности закона распределения при преобразовании операндов в вероятностное отображение, но и от точности ГПСЧ, который используется в операции сложения.

Параллельный вероятностный сумматор с проверкой входных значений имеет наибольший аппаратный объем и эффективен только при весе операндов намного меньшем максимально возможного веса. Стоит также отметить, что с увеличением количества входных операндов – аппаратный объем также будет расти из-за добавления элементов задержки.

Параллельный вероятностный умножитель имеет самый малый аппаратный объем. Однако групповой умножитель имеет преимущество перед параллельным в быстродействии, так как позволяет уменьшить в 4 раза количество необходимых для выполнения операции умножения тактов.

После проведения анализа, можно сделать вывод, что количество одноразрядных умножителей в групповом должно быть кратно четырем.

Результаты выпускной квалификационной работы показали, что методы ускорения операции сложения и умножения эффективны в вероятностной форме представления данных. Они позволяют значительно увеличить быстродействие систем. Данные методы совместимы с методами повышения точности вероятностного преобразования, что делает их еще более привлекательными для дальнейшего изучения и применения.

**ПЕРЕЧЕНЬ ПРИНЯТЫХ СОКРАЩЕНИЙ**

ИИС- информационно-измерительные системы

МО - математическое ожидание;

СКС - специализированные компьютерные системы;

ГСЧ - генератор случайных чисел;

ГПСРЧ - генератор псевдослучайно распределённых чисел ;

ГПСЧ- генератор псевдослучайных чисел;

СКО - среднеквадратическое отклонение;

ПСС - псевдослучайный сигнал;

ПСЧ - псевдослучайное число;

СВ - случайная величина;

БСВ - базовая случайная величина;

# ЧС - Черезвичайная ситуация;

ГОСТ - государственный стандарт;

ИС - интегральная схема;

ОП - оперативная память;

ООС - отрицательная обратная связь;

ПАВ - преобразователь «амплитуда – вероятность»;

ПФВ - преобразователь «фаза – вероятность»;

ПЧаВ - преобразователь «частота – вероятность»;

ПЧВ - преобразователь «число – вероятность»;

ПЗУ- постоянное запоминающее устройство;

СОТ - служба охраны труда;

ЦП - центральный процессор;

**БИБЛИОГРАФИЧЕСКИЙ СПИСОК**

1. Вероятностный параллельный сумматор /Н.Е. Сапожников, д.т.н., проф., Д.В. Моисеев, к.т.н. - Севастопольский национальный университет ядерной энергии и промышленности

2. Сапожников Н.Е. Дискретная схемотехника. Ч.1. Севастополь: СНИЯЭиП, 2005.- 250 с.

3. Гладкий В.С. Вероятностные вычислительные модели. – М.: Наука, 1973.- 299 с.

4. Сапожников Н.Е. Дискретная схемотехника. Ч.2. Севастополь: СНУЯЭиП, 2005.- 304 с.

5. Сапожников Н.Е., Пряшников Ф.Д., Столярчук Ю.Ю.: К вопросу о представлении дискретных сигналов в псевдовероятностной двоичной форме. 2011г.

6. Сапожников Н.Е. Математическое моделирование на ПК. Севастополь: СНУЯЭиП, 2006.- 380 с.

7. Разевиг В.Д. Система схемотехнического проектирования Micro-Cap 6. -М.: Горячая линия - Телеком, 2001. - 344 с.

8. Гурский Е.И. Теория вероятностей с элементами математической статистики: Учеб.пособие для вузов. – М.: Высшая школа, 1971. – 328 с.

9. Федоров Р. Ф., Яковлев В. В., Добрис Г. В. Стохастические преобразователи информации. — Л.: Машиностроение. Ленингр. отделе-ние, 1978. — 304 с.

10. Н.Е. Сапожников, Д.В. Моисеев, Ю.Ю. Столярчук Оценка точности и быстродействия при вероятностной форме представления информации. Зб. наук.пр.СНУЯЕтаП. – Севастополь: СНУЯЕтаП, 2011. – Вип. 3 (94). − С. 134 – 140.

11. Методы цифрового моделирования и идентификации стационарных случайных процессов в информационно-измерительных системах/ А.Н.Лебедев, Д.Д.Недосекин, Г.А.Стеклова, Е.А.Чернявский. – Л.: Энергоатомиздат, 1988. – 64 с.

СПИСОК ИЛЛЮСТРАЦИЙ

Рисунок 2.1 – Графики, поясняющие однолинейное однополярное вероятностное представление информации 8

Рисунок 2.2 – График, поясняющий работу однолинейного двухполярного преобразователя 11

Рисунок 2.3 – Круговая схема кодирования чисел при однолинейном двухполярном вероятностном преобразовании 12

Рисунок 2.4 – Функциональная схема вероятностного сумматора при однолинейном однополярном представлении слагаемых 14

Рисунок 2.5 – Структурно-функциональная схема вероятностного множительного устройства 18

Рисунок 2.6 – Структурная схема последовательного ГПСЧ 20

Рисунок 2.7 – Кривая распределения – а, с уменьшением СКО – б 22

Рисунок 2.8 – Кривая логарифмического нормального распредиления 24

Рисунок 3.1 – Трехуровневая схема представления иерархии 31

Рисунок 3.2 – Трехуровневая схема представления иерархии 34

Рисунок 4.1 – Вероятностный последовательный сумматор 36

Рисунок 4.2 – Функциональная схема одного разряда цифровой схемы сравнения вероятностного преобразователя 37

Рисунок 4.3 – Функциональная схема параллельного вероятностного сумматора со случайным переключением шин 38

Рисунок 4.4 – Функциональная схемапараллельного вероятностного сумматора с проверкой входных данных 39

Рисунок 5.1 – Функциональная схема вероятностного параллельного умножителя 41

Рисунок 4.6 – Вероятностного умножитель с групповой структурой.42

Рисунок 6.1 – Результат первого опыта вероятностного последовательного сумматора.44

Рисунок 6.2 – Результат первого опыта вероятностного последовательного сумматора при количестве статистических испытаний в 1000.45

Рисунок 6.3 – Результат первого опыта вероятностного последовательного сумматора при количестве статистических испытаний в 10 000.45

Рисунок 6.4 – Результат работы последовательного вероятностного сумматора во втором опыте.46

Рисунок 6.5 – Результат работы последовательного вероятностного сумматора в третьем опыте.46

Рисунок 6.6 – Результат работы последовательного вероятностного сумматора в четвертом опыте.46

Рисунок 6.7 – Результат работы последовательного вероятностного сумматора в пятом опыте.47

Рисунок 6.8 – График зависимости аппаратного объема от значения операндов.47

Рисунок 6.9 – Результат первого испытания вероятностного параллельного сумматора.48

Рисунок 6.10 – Результат работы параллельного вероятностного сумматора во втором опыте.49

Рисунок 6.11 – Результат работы параллельного вероятностного сумматора в третьем опыте.49

Рисунок 6.12 – Результат работы параллельного вероятностного сумматора в четвертом опыте.49

Рисунок 6.13 – Результат работы параллельного вероятностного сумматора в пятом опыте.50

Рисунок 6.14 – График зависимости аппаратного объема от значения операндов.50

Рисунок 6.15 – Результат первого испытания вероятностного параллельного сумматора с проверкой входных значений.51

Рисунок 6.16 – Результат работы параллельного вероятностного сумматора с проверкой входных значений во втором опыте 52

Рисунок 6.17 – Результат работы параллельного вероятностного сумматора с проверкой входных значений в третьем опыте 52

Рисунок 6.18 – Результат работы параллельного вероятностного сумматора с проверкой входных значений в четвертом опыте.52

Рисунок 6.19 – Результат работы параллельного вероятностного сумматора с проверкой входных значений в пятом опыте.53

Рисунок 6.20 – График зависимости аппаратного объема от значения операндов.53

Рисунок 6.21 – График зависимости количество необходимых для выполнения операции сложения тактов от значений операндов.54

Рисунок 6.22 – График сравнения аппаратного объема по трем моделям сумматоров..54

Рисунок 6.23 – График сравнения количества необходимых для выполнения операции сложения тактов по трем моделям.55

Рисунок 6.24 – Результат первого опыта вероятностного параллельного умножителя.56

Рисунок 6.25 – Результат первого опыта вероятностного параллельного умножителя при количестве статистических испытаний в 10 00056

Рисунок 6.26 – Результат работы параллельного вероятностного умножителя во втором опыте.57

Рисунок 6.27 – Результат работы параллельного вероятностного умножителя в третьем опыте 57

Рисунок 6.28 – Результат работы параллельного вероятностного умножителя в четвертом опыте 57

Рисунок 6.29 – Результат работы параллельного вероятностного умножителя в пятом опыте.58

Рисунок 6.30 – График зависимости аппаратного объема от значения операндов..58

Рисунок 6.31 – Результат первого опыта вероятностного группового умножителя.59

Рисунок 6.32 – Результат работы группового вероятностного умножителя во втором опыте.60

Рисунок 6.33 – Результат работы группового вероятностного умножителя в третьем опыте 60

Рисунок 6.34 – Результат работы группового вероятностного умножителя в четвертом опыте.60

Рисунок 6.35 – Результат работы группового вероятностного умножителя в пятом опыте 61

Рисунок 6.36 – График зависимости аппаратного объема от значения операндов 61

Рисунок 6.37 – График зависимости количество необходимых для выполнения операции умножения тактов от значений операндов.62

Рисунок 6.38 – График сравнения аппаратного объема вероятностных умножителей..63

Рисунок 6.39 – График сравнения количества необходимых для выполнения операции умножения тактов.63

**ПРИЛОЖЕНИЕ А – листинг программы**

*application.js*

function get\_probablistic (x,x\_dop,x\_max) {

var rand\_number = Math.random();

return rand\_number < x ? 1 : 0;

}

function get\_probablistic\_rand (x,x\_dop,x\_max) {

var rand\_number = randomInteger(1,x\_max)

return rand\_number < x ? 1 : 0;

}

function get\_probablistic\_pseydo (a,c,x,x\_dop,x\_max) {

var rand\_number = (a \* x\_dop + c) % x\_max;

return rand\_number <= x ? 1 : 0;

}

function get\_sum\_from\_array (probablistic\_array) {

var sum = 0;

probablistic\_array.forEach(function (item){

sum += item;

})

return sum;

}

function get\_number\_before\_probablistic (number,x\_max,k) {

return (number \* x\_max) / k;

}

function get\_number\_before\_multiplier (number,x\_max,k) {

return (number \* (x\_max \* x\_max)) / k;

}

function get\_modulus (x\_dop,modulus) {

return x\_dop % modulus;

}

function randomInteger (min, max) {

return Math.random() \* (max - min) + min;

}

function get\_log (x) {

return Math.ceil(Math.log(x+1) / Math.log(2));

}

*probablistic.coffee*

$(document).ready ->

console.log "скрипт загружен"

$('body').on 'click','#probabilistic-input-button', ->

tact = 0

main\_number\_one = $('#main\_number\_one').val() / 100

main\_number\_two = $('#main\_number\_two').val() / 100

count\_of\_test = parseInt $('#count-of-test').val()

upper\_bound = 100

a\_array = new Array

output\_table\_body\_one = $('#probabilistic-output-table-one tbody tr')

output\_table\_body\_two = $('#probabilistic-output-table-two tbody tr')

output\_table\_body\_one.html ''

output\_table\_body\_two.html ''

hardware = 1 #тактовый генератор

#get hardware

hardware += 3

if main\_number\_one > main\_number\_two

length\_of\_operand = get\_log(main\_number\_one \* 100)

else

length\_of\_operand = get\_log(main\_number\_two \* 100)

hardware += 5 \* (length\_of\_operand - 1)

hardware += get\_log count\_of\_test #счетчик испытаний

counter = 0

k\_iterator = 0

while k\_iterator < count\_of\_test

probabilistic\_elem = get\_probablistic(main\_number\_one,k\_iterator,upper\_bound)

a\_array.push probabilistic\_elem

if probabilistic\_elem == 1 then counter++

tact++

k\_iterator++

probabilistic\_array = a\_array

probabilistic\_array.map (value) ->

output\_table\_body\_one.append '<td>' + value + '</td>'

a\_array = new Array

k\_iterator = 0

while k\_iterator < count\_of\_test

probabilistic\_elem = get\_probablistic(main\_number\_two,k\_iterator,upper\_bound)

a\_array.push probabilistic\_elem

if probabilistic\_elem == 1 then counter++

tact++

k\_iterator++

probabilistic\_array = a\_array

probabilistic\_array.map (value) ->

output\_table\_body\_two.append '<td>' + value + '</td>'

hardware += get\_log counter #счетчик результата

$('#result\_counter').val Math.round(get\_number\_before\_probablistic(counter,upper\_bound,count\_of\_test))

$('#result\_screept\_speed').val hardware

$('#result\_tact\_count').val tact

*probablistic-parallel.coffee*

$(document).ready ->

console.log "скрипт загружен"

$('body').on 'click','#probabilistic-parallel-input-button', ->

tact = 0

main\_number\_one= $('#main\_number\_one').val() / 100

main\_number\_two= $('#main\_number\_two').val() / 100

count\_of\_test = parseInt $('#count-of-test').val()

upper\_bound = 100

a\_array\_one = new Array

a\_array\_two = new Array

output\_table\_body\_one = $('#probabilistic-output-table-one tbody tr')

output\_table\_body\_two = $('#probabilistic-output-table-two tbody tr')

output\_table\_body\_one.html ''

output\_table\_body\_two.html ''

hardware = 1 #тактовый генератор

#get hardware

hardware += 3

hardware += get\_log count\_of\_test #счетчик испытаний

#one

length\_of\_operand = get\_log(main\_number\_one \* 100)

hardware += 5 \* (length\_of\_operand - 1)#преобразователь

hardware += get\_log(main\_number\_one \* 100)#регистр хранения

#two

length\_of\_operand = get\_log(main\_number\_two \* 100)

hardware += 5 \* (length\_of\_operand - 1)#преобразователь

hardware += get\_log(main\_number\_two \* 100)#регистр хранения

#mux

hardware += 4

#ГПСЧ

hardware += 3

counter = 0

k\_iterator = 0

while k\_iterator < count\_of\_test

operand\_line = get\_modulus k\_iterator,2

probabilistic\_elem\_one = get\_probablistic(main\_number\_one,k\_iterator,upper\_bound)

a\_array\_one.push probabilistic\_elem\_one

probabilistic\_elem\_two = get\_probablistic(main\_number\_one,k\_iterator,upper\_bound)

a\_array\_two.push probabilistic\_elem\_two

if operand\_line == 0

probabilistic\_elem = probabilistic\_elem\_one

else

probabilistic\_elem = probabilistic\_elem\_two

if probabilistic\_elem == 1 then counter++

tact++

k\_iterator++

hardware += get\_log counter #счетчик результата

a\_array\_one.map (value) ->

output\_table\_body\_one.append '<td>' + value + '</td>'

a\_array\_two.map (value) ->

output\_table\_body\_two.append '<td>' + value + '</td>'

$('#result\_counter').val Math.round(get\_number\_before\_probablistic\_parallel(counter,upper\_bound,count\_of\_test,main\_number\_one,main\_number\_two))

$('#result\_screept\_speed').val hardware

$('#result\_tact\_count').val tact

*probablistic-parallel-new.coffee*

$(document).ready ->

console.log "скрипт загружен"

$('body').on 'click','#probabilistic-new-input-button', ->

tact = 0

main\_number\_one= $('#main\_number\_one').val() / 100

main\_number\_two= $('#main\_number\_two').val() / 100

count\_of\_test = parseInt $('#count-of-test').val()

upper\_bound = 100

a\_array\_one = new Array

a\_array\_two = new Array

output\_table\_body\_one = $('#probabilistic-output-table-one tbody tr')

output\_table\_body\_two = $('#probabilistic-output-table-two tbody tr')

output\_table\_body\_one.html ''

output\_table\_body\_two.html ''

hardware = 1 #тактовый генератор

#get hardware

hardware += 3

hardware += get\_log count\_of\_test #счетчик испытаний

hardware += 8 #дешифратор

#one

length\_of\_operand = get\_log(main\_number\_one \* 100)

hardware += 5 \* (length\_of\_operand - 1)#преобразователь

hardware += get\_log(main\_number\_one \* 100)#регистр хранения

#two

length\_of\_operand = get\_log(main\_number\_two \* 100)

hardware += 5 \* (length\_of\_operand - 1)#преобразователь

hardware += get\_log(main\_number\_two \* 100)#регистр хранения

hardware += 3 #Задержки сигналов

counter = 0

k\_iterator = 0

while k\_iterator < count\_of\_test

probabilistic\_elem\_one = get\_probablistic(main\_number\_one,k\_iterator,upper\_bound)

a\_array\_one.push probabilistic\_elem\_one

probabilistic\_elem\_two = get\_probablistic(main\_number\_two,k\_iterator,upper\_bound)

a\_array\_two.push probabilistic\_elem\_two

if probabilistic\_elem\_one == 0 && probabilistic\_elem\_two == 0

tact++

else if probabilistic\_elem\_one == 1 && probabilistic\_elem\_two == 1

counter+=2

tact+=2

else

counter++

tact++

k\_iterator++

hardware += get\_log counter #счетчик результата

a\_array\_one.map (value) ->

output\_table\_body\_one.append '<td>' + value + '</td>'

a\_array\_two.map (value) ->

output\_table\_body\_two.append '<td>' + value + '</td>'

$('#result\_counter').val Math.round(get\_number\_before\_probablistic(counter,upper\_bound,count\_of\_test))

$('#result\_screept\_speed').val hardware

$('#result\_tact\_count').val tact

*multiplier.coffee*

$(document).ready ->

console.log "скрипт загружен"

$('body').on 'click','#multiplier-input-button', ->

tact = 0

main\_number\_one= $('#main\_number\_one').val() / 100

main\_number\_two= $('#main\_number\_two').val() / 100

count\_of\_test = parseInt $('#count-of-test').val()

upper\_bound = 100

a\_array\_one = new Array

a\_array\_two = new Array

output\_table\_body\_one = $('#probabilistic-output-table-one tbody tr')

output\_table\_body\_two = $('#probabilistic-output-table-two tbody tr')

output\_table\_body\_one.html ''

output\_table\_body\_two.html ''

hardware = 1 #тактовый генератор

#get hardware

hardware += 3

hardware += get\_log count\_of\_test #счетчик испытаний

#one

length\_of\_operand = get\_log(main\_number\_one \* 100)

hardware += 5 \* (length\_of\_operand - 1)#преобразователь

hardware += get\_log(main\_number\_one \* 100)#регистр хранения

#two

length\_of\_operand = get\_log(main\_number\_two \* 100)

hardware += 5 \* (length\_of\_operand - 1)#преобразователь

hardware += get\_log(main\_number\_two \* 100)#регистр хранения

hardware += 1

counter = 0

k\_iterator = 0

while k\_iterator < count\_of\_test

probabilistic\_elem\_one = get\_probablistic(main\_number\_one,k\_iterator,upper\_bound)

a\_array\_one.push probabilistic\_elem\_one

probabilistic\_elem\_two = get\_probablistic(main\_number\_two,k\_iterator,upper\_bound)

a\_array\_two.push probabilistic\_elem\_two

if probabilistic\_elem\_one == 1 && probabilistic\_elem\_two == 1

counter++

tact++

k\_iterator++

hardware += get\_log counter #счетчик результата

a\_array\_one.map (value) ->

output\_table\_body\_one.append '<td>' + value + '</td>'

a\_array\_two.map (value) ->

output\_table\_body\_two.append '<td>' + value + '</td>'

$('#result\_counter').val get\_number\_before\_multiplier(counter,upper\_bound,count\_of\_test)

$('#result\_screept\_speed').val hardware

$('#result\_tact\_count').val tact

*multiplier\_new.coffee*

$(document).ready ->

console.log "скрипт загружен"

$('body').on 'click','#multiplier-new-input-button', ->

tact = 0

main\_number\_one= $('#main\_number\_one').val() /100

main\_number\_two= $('#main\_number\_two').val() / 100

count\_of\_test = parseInt $('#count-of-test').val()

count\_of\_multiplier = count\_of\_test / 4

upper\_bound = 100

a\_array\_one = new Array

a\_array\_two = new Array

output\_table\_body\_one = $('#probabilistic-output-table-one tbody tr')

output\_table\_body\_two = $('#probabilistic-output-table-two tbody tr')

output\_table\_body\_one.html ''

output\_table\_body\_two.html ''

hardware = 1 #тактовый генератор

#get hardware

hardware += 3

hardware += get\_log count\_of\_test #счетчик испытаний

#one

length\_of\_operand = get\_log(main\_number\_one \* 100)

hardware += (5 \* (length\_of\_operand - 1)) \* 4 #преобразователь

#two

length\_of\_operand = get\_log(main\_number\_two \* 100)

hardware += (5 \* (length\_of\_operand - 1)) \* 4 #преобразователь

hardware += 4 #коньюнктор

hardware += 4 #обратный счетчик

counter = 0

k\_iterator = 0

while k\_iterator < count\_of\_multiplier

counter\_null = false

probabilistic\_elem\_one = get\_probablistic(main\_number\_one,k\_iterator,upper\_bound)

a\_array\_one.push probabilistic\_elem\_one

probabilistic\_elem\_two = get\_probablistic(main\_number\_two,k\_iterator,upper\_bound)

a\_array\_two.push probabilistic\_elem\_two

if probabilistic\_elem\_one == 1 && probabilistic\_elem\_two == 1

counter++

counter\_null = true

tact++

probabilistic\_elem\_one = get\_probablistic(main\_number\_one,k\_iterator,upper\_bound)

a\_array\_one.push probabilistic\_elem\_one

probabilistic\_elem\_two = get\_probablistic(main\_number\_two,k\_iterator,upper\_bound)

a\_array\_two.push probabilistic\_elem\_two

if probabilistic\_elem\_one == 1 && probabilistic\_elem\_two == 1

counter++

counter\_null = true

tact++

probabilistic\_elem\_one = get\_probablistic(main\_number\_one,k\_iterator,upper\_bound)

a\_array\_one.push probabilistic\_elem\_one

probabilistic\_elem\_two = get\_probablistic(main\_number\_two,k\_iterator,upper\_bound)

a\_array\_two.push probabilistic\_elem\_two

if probabilistic\_elem\_one == 1 && probabilistic\_elem\_two == 1

counter++

counter\_null = true

tact++

probabilistic\_elem\_one = get\_probablistic(main\_number\_one,k\_iterator,upper\_bound)

a\_array\_one.push probabilistic\_elem\_one

probabilistic\_elem\_two = get\_probablistic(main\_number\_two,k\_iterator,upper\_bound)

a\_array\_two.push probabilistic\_elem\_two

if probabilistic\_elem\_one == 1 && probabilistic\_elem\_two == 1

counter++

counter\_null = true

tact++

if counter\_null == false

tact++

k\_iterator++

hardware += get\_log counter #счетчик результата

a\_array\_one.map (value) ->

output\_table\_body\_one.append '<td>' + value + '</td>'

a\_array\_two.map (value) ->

output\_table\_body\_two.append '<td>' + value + '</td>'

$('#result\_counter').val get\_number\_before\_multiplier(counter,upper\_bound,count\_of\_test)

$('#result\_screept\_speed').val hardware

$('#result\_tact\_count').val Math.round(tact / 4)