**ВВЕДЕНИЕ**

Моделирование - наиболее мощный универсальный метод исследования и оценки эффективности систем, поведение которых зависит от воздействия случайных факторов. Области применения методов имитации чрезвычайно широки и разнообразны. Однако по опыту работы и материалам диссертационных советов можно сделать вывод о том, что исследователи пока довольно редко используют в качестве инструментальных средств исследования системы моделирования, преимущества которых вполне очевидны. Системы моделирования имеют специализированные средства, реализующие дополнительные возможности по организации модельных экспериментов на компьютере. Они также предоставляют возможность учитывать в моделях фактор времени, то есть строить динамические имитационные модели, что особенно важно для многих систем .

Применение универсальных языков программирования при реализации имитационных моделей позволяет исследователю достигнуть гибкости при разработке, отладке и испытании модели. Однако языки моделирования, ориентированные на определённую предметную область, являются языками более высокого уровня, поэтому дают возможность с меньшими затратами создавать программы моделей для исследования сложных систем.

Специализированные языки моделирования делят на три группы, соответствующие видам имитации: для непрерывных, дискретных и комбинированных процессов. Научно-технический прогресс в наше время невозможен без исследования, построения и использования сложных систем и процессов, разнообразных по своей физической природе, функциональному назначению, путям реализации. Примерами таких систем являются системы обеспечения АЭС и сама АЭС, компьютеризованные информационно-измерительные и информационно-управляющие системы радиационного и экологического контроля, технологические потоки, телекоммуникационные системы и т. д. Исследование поведения таких систем при их эксплуатации путём натурного эксперимента чрезвычайно дорого, сложно, а при проектировании- невозможно. Именно поэтому основным методом исследования сложных систем является метод математического моделирования, то есть метод описания поведения физических систем при помощи математических соотношений или уравнений.

Построение математической модели и экспериментирование на ней доступно каждому, знакомому с принципами и методами современного математического вычислительного эксперимента. С учебной точки зрения, каждый студент, освоивший основные методы математического моделирования, получает в свои руки универсальный инструмент выполнения курсовых, бакалаврских и дипломных работ по своей специальности и тем самым значительно упрощает себе задачу.

**1 ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ**

* 1. **Основные понятия и определения**

Моделирование применяется практически во всех сферах человеческой деятельности.

Научно-техническое развитие обычно идёт по пути: наблюдение - теоретические исследования – эксперимент - организация производственных процессов. В научных исследованиях большую роль играют *гипотезы -* определённые предсказания, основывающиеся на небольшом количестве опытных данных, наблюдений, догадок. *Аналогия* - суждение о каком-либо частном сходстве двух объектов. Гипотезы и аналогии, сведённые в удобные для исследования логические схемы, позволяющие проводить эксперименты, называются *моделями*. Другими словами, *модель (*лат.modulus-мера*)* - это объект-заместитель объекта-оригинала, обеспечивающий изучение некоторых свойств оригинала. А замещение одного объекта другим в целях получения информации о важнейших свойствах объекта-оригинала с помощью объекта-модели называется моделированием. Моделирование можно рассматривать как один из мощнейших инструментов, используемых при исследовании больших и сложных систем и процессов.

*Система* – совокупность взаимосвязанных элементов, объединенных в одно целое, где под целью понимается совокупность результатов, определяемых назначением системы.

*Сложная система* характеризуется множеством взаимосвязанных и взаимодействующих элементов и подсистем различной физической природы, составляющих нераздельное целое, обеспечивающих выполнение некоторой сложной функции. Разбиение сложных систем на подсистемы называют декомпозицией.

*Функция системы* - правило получения результатов, предписанных назначением.

Процесс моделирования предполагает наличие объекта исследования, исследователя, имеющего перед собой конкретную задачу, и модели. Если результаты моделирования могут служить для прогнозирования процессов, протекающих в исследуемых объектах, то модель адекватна объекту.

* 1. **Виды и уровни моделирования**

В фундаменте моделирования лежит *теория подобия*, которая утверждает, что абсолютное подобие имеет место лишь при замещении одного объекта точно таким же. При реальном моделировании абсолютное подобие не имеет места. Поэтому в качестве первого признака классификации видов моделирования можно выбрать степень полноты подобия. В основе полного и неполного моделирования лежит соответственно полное и неполное подобие, проявляющееся во времени и пространстве. В основе приближённого моделирования лежит приближённое подобие.

Вторым признаком классификации будем считать характер изучаемых процессов. В соответствии с ним виды моделирования могут быть разделены на детерминированные и стохастические, статические и динамические, дискретные, непрерывные и дискретно-непрерывные. *Детерминированное* моделирование отображает детерминированные процессы, то есть лишённые случайных воздействий. *Стохастическое* моделирование отображает вероятностные процессы и события. *Статическое* моделирование служит для описания поведения объекта в какой-либо момент времени, а динамическое отражает поведение объекта во времени.

*Дискретное* моделирование служит для описания процессов, которые предполагаются дискретными, соответственно *непрерывное* или *аналоговое* моделирование позволяет отразить непрерывные процессы в системах.

Третий признак классификации-формы представления моделируемого объекта. В соответствии с ним различают абстрактные и реальные модели. При *реальном* моделировании используется возможность исследования различных характеристик либо на реальном объекте целиком, либо на его части. Разновидностью реального моделирования является так называемое *натурное* моделирование, при котором осуществляется проведение исследований на реальном объекте с последующей обработкой результатов эксперимента на основе теории подобия.

Другим видом реального моделирования является *физическое*, отличающееся от натурного тем, что исследование проводится на установках, которые сохраняют природу явления и обладают физическим подобием, например, модель корабля, самолёта, космического аппарата.

*Абстрактная* или *мысленная* модель-это описание объекта исследована на каком-то языке либо текстуально, либо формализовано.

*Гносеологические* модели направлены на изучение объективных законов природы.

*Информационные* или *кибернетические* модели описывают поведение объекта-оригинала, но не копируют его.

*Сенсуальные* модели-модели чувств, эмоций, либо модели, оказывающие влияние на органы чувств человека.

*Концептуальная* модель-это абстрактная модель, определяющая структуру моделируемой системы, свойства её элементов и причинно-следственные связи, присущие исследуемому объекту и существенные в рамках данного конкретного исследования.

*Математическая* модель представляется на языке математических отношений. Она имеет форму функциональных зависимостей между параметрами, учитываемыми соответствующей концептуальной моделью. Под *математическим* моделированием будем понимать процесс установления соответствия данному реальному объекту некоторого математического объекта(математической модели) и исследование этой модели, позволяющее получить характеристики рассматриваемого реального объекта. Математическое моделирование подразделяют на аналитическое, имитационное и комбинированное.

Для *аналитического* моделирования характерно то, что процессы функционирования элементов системы записываются в виде некоторых функциональных соотношений или логических условий .Недостаток: для получения аналитических зависимостей приходится упрощать модель.

При *имитационном* моделировании реализующий модель алгоритм воспроизводит процесс функционирования системы во времени. Существуют три представления времени:

* *Реальное время* - это время, в котором происходит функционирование моделируемой системы в реальной жизни.
* *Модельное (системное) время* - это время, в котором происходит функционирование моделируемой системы при проведении имитационного моделирования на ПК.
* *Машинное время* - это время, отражающее затраты компьютерного времени на проведение имитационного моделирования.

Основным преимуществом имитационного моделирования перед аналитическим является возможность решение более сложных и универсальных задач. Имитационные модели позволяют достаточно просто учитывать такие факторы, как наличие дискретных и непрерывных элементов, нелинейные характеристики элементов системы, многочисленные случайные воздействия. Сегодня имитационное моделирование - наиболее эффективный метод исследования сложных систем.

В зависимости от степени детализации описания сложных систем можно выделить три основных уровня моделирования:

* Уровень структурного моделирования с использованием алгоритмических моделей и применением специальных языков моделирования;
* Уровень логического моделирования функциональных схем в виде уравнений булевой алгебры;
* Уровень количественного моделирования или анализа.
  1. **Основы построения и реализации имитационных моделей**

На этапе исследования и проектирования систем при построении и реализации машинных моделей широко используется метод статистических испытаний – метод Монте-Карло, который базируется на использовании случайных чисел, то есть возможных значений некоторой случайной величины с заданным законом распределения вероятностей. *Сущность метода Монте-Карло* – составляется программа для осуществления одного случайного испытания, затем это испытание повторяется N раз, каждый независимо от остальных, и результаты усредняются.

*Сущность метода статистического моделирования* сводится к построению для процесса функционирования исследуемой системы некоторого моделирующего алгоритма, имитирующего поведение и взаимодействие элементов системы с учётом случайных входных воздействий внешней среды, и реализации возможности появления этого события при осуществлении рассматриваемого процесса. Событие, которое должно обязательно произойти, называют *достоверным.* Вероятность невозможного события равна нулю. *Независимые* случайные события – такие, что появление какого-либо одного из них не изменяет степени объективной возможности появления другого. *Случайные величины* (СВ) – это меры различных характеристик случайного явления и определяющих его факторов. Совокупность возможных значений случайной величины и вероятностей того, что она примет эти значения, образует *закон распределения* СВ.

*Дискретной* называют СВ, которая может принимать дискретное множество значений. *Непрерывная* СВ принимает любое значение из некоторого интервала. *Математическое ожидание* СВ – это среднее значение величины. *Дисперсия* – математическое ожидание квадрата отклонения СВ от её среднего значения.

* 1. **Алгоритмизация и машинная реализация модели**

Для стохастической системы функция состояний процесса z и соотношения модели определяют лишь распределение вероятностей для zi(t0+Δt) в момент времени t+Δt. В общем случае и начальные условия z0 могут быть случайными, задаваемыми соответствующим распределением вероятностей. При этом структура моделирующего алгоритма для стохастических систем дополняется распределением вероятностей для возможных состояний. Пусть счетчик системного времени показывает время t0. В соответствии с заданным распределением вероятностей выбирается z0i. Далее, исходя из распределения, получается состояние zi(t0+Δt) и так далее, пока не будет построена одна из возможных реализаций случайного многомерного процесса zi(t) в заданном интервале времени.

Рассмотренный принцип построения моделирующих алгоритмов называется принципом Δt. Это наиболее универсальный принцип, позволяющий определить последовательные состояния процесса функционирования системы через заданные интервалы времени Δt. Но с точки зрения затрат машинного времени он иногда оказывается неэкономичным.

При рассмотрении процессов функционирования некоторых систем можно обнаружить, что для них характерны два типа состояния:

* особые, присущие процессу функционирования системы только в некоторые моменты времени (моменты поступления входных или управляющих воздействий, возмущений внешней среды и т.п.);
* неособые, в которых процесс находится все остальное время.

Особые состояния характерны еще и тем обстоятельством, что функции состояний zi(t) в эти моменты времени изменяются скачком, а между особыми состояниями изменение координат zi(t) происходит плавно и непрерывно или не происходит вообще. Таким образом, следя при моделировании системы только за ее особыми состояниями в те моменты времени, когда эти состояния имеют место, можно получить информацию, необходимую для построения функций zi(t). Очевидно, что для описанного типа систем могут быть построены моделирующие алгоритмы по принципу «особых состояний», что и было проделано в ходе выполнения данной работы; разработанный по этому принципу алгоритм приведен в приложении Е.

Принцип особых состояний называется принципом ∆z. Для системы массового обслуживания в качестве особых состояний выбраны состояния в моменты поступления заявок на обслуживание и в моменты окончания обслуживания заявок, когда состояние системы, оцениваемое количеством находящихся в ней заявок, меняется скачком, что и представлено в алгоритме.

Характеристики процесса функционирования таких систем с особыми состояниями оцениваются по информации об особых состояниях, а неособые состояния при моделировании не рассматриваются. «Принцип ∆z» дает возможность для ряда систем существенно уменьшить затраты машинного времени на реализацию моделирующих алгоритмов по сравнению с «принципом Δt». Логика построения моделирующего алгоритма, реализующего «принцип ∆z», отличается от рассмотренной для «принципа Δt» только тем, что включает в себя процедуру определения момента времени tδ, соответствующего следующему особому состоянию системы.

* 1. **Системы массового обслуживания**

Системы массового обслуживания (СМО) – класс математических схем, разработанных в теории массового обслуживания и различных приложениях для формализации процессов функционирования систем, которые по своей сути являются процессами обслуживания.

Характерным для работы таких систем является случайное появление заявок на обслуживание и завершение обслуживания случайные моменты времени, то есть стохастический характер их функционирования.

В любом элементарном акте обслуживания можно выделить две основные составляющие: ожидание обслуживания заявкой и само обслуживание заявки.

Основными элементами СМО являются входной поток заявок, входной поток обслуживаний, очереди заявок, ожидающих обслуживания, каналы обслуживания и выходной поток обслуженных заявок и заявок, которым по тем или иным причинам в обслуживании отказано.

СМО классифицируют по следующим признакам:

* по количеству каналов обслуживания они делятся на одноканальные и многоканальные СМО;
* по организации ожидания заявки – на системы с отказами и системы с ожиданием или с очередями (накопителями);
* СМО с накопителями, в свою очередь, делятся на системы с приоритетами и без приоритетов;
* по количеству фаз обслуживания – на однофазные и многофазные;
* по взаимосвязи с потоками заявок – на разомкнутые и замкнутые.

Эффективность работы СМО характеризуется следующими основными показателями:

* абсолютная пропускная способность – среднее количество заявок, которое может обслужить система в единицу времени;
* относительная пропускная способность – отношение среднего числа заявок, обслуженных СМО в единицу времени, к среднему числу всех заявок, поступивших в СМО за это же время;
* коэффициент занятости – отношение среднего числа занятых каналов к общему числу каналов;
* коэффициент простоя – отношение среднего числа свободных каналов к общему числу каналов.

Задача моделирования СМО ставится следующим образом: пусть имеется СМО, на вход которой поступает поток заявок; необходимо осуществить: моделирование входного потока заявок; моделирование алгоритма функционирования СМО; ввод некоторых операторов по организации очереди.

Математические модели входных потоков заявок на обслуживание представляют собой потоки событий, которыми называется последовательность событий, происходящих одно за другим в какие-то случайные моменты времени. Поток событий называется однородным, если он характеризуется только моментами поступления этих событий (вызывающими моментами) и задается последовательностью {tn}={0≤t1≤t2…≤tn≤…}, где tn – момент наступления (поступления) n-го события – неотрицательное вещественное число. Потоком неоднородных событий называется последовательность {tn, fn}, где fn – набор признаков события (например, наличие приоритета, принадлежность заявки к тому или иному источнику заявок).

Поток называется ординарным, если вероятность того, что на малый интервал времени Δt, примыкающий к моменту времени t, попадает больше одного события, пренебрежимо мала по сравнению с вероятностью того, что на этот же интервал времени Δt попадает ровно одно событие, то есть P1(t, Δt)>>P>1(t, Δt). Стационарным потоком событий называется поток, для которого вероятность появления того или иного числа событий на интервале времени τ зависит лишь от длины этого участка и не зависит от того, где на оси времени 0...t взят этот участок.

При моделировании систем применительно к элементарному каналу обслуживания Кi можно считать, что поток заявок wi, то есть интервалы времени между моментами появления заявок на входе Кi , образуют подмножество неуправляемых переменных, а поток обслуживания ui, то есть интервалы времени между началом и окончанием обслуживания заявки, образуют под множество управляемых переменных.

Заявки, обслуженные каналом Кi , и заявки, покинувшие прибор обслуживания заявок по различным причинам необслуженными (например, из-за переполнения накопителя), образуют выходной поток yi , то есть интервалы времени между моментами выхода заявок образуют подмножество выходных переменных. Процесс функционирования прибора обслуживания заявок можно представить как процесс изменения состояний его элементов во времени zi(t). Переход в новое состояние для прибора означает изменение количества заявок, которые в нем находятся (в канале Кi и в накопителе Нi ).

В практике моделирования систем, имеющих более сложные структурные связи и алгоритмы поведения, для формализации используются не отдельные приборы обслуживания, а Q-схемы, образуемые композицией многих элементарных приборов обслуживания. Если каналы Кi различных приборов обслуживания соединены параллельно, то имеет место многоканальное обслуживание (многоканальная Q-схема). Если приборы Пi и их параллельные композиции соединены последовательно, то имеет место многофазное обслуживание (многофазная Q-схема). Таким образом, для задания Q-схемы необходимо использовать оператор сопряжения R, отражающий взаимосвязь элементов структуры (каналов и накопителей) между собой.

При рассмотрении алгоритмов функционирования приборов обслуживания необходимо также задать набор правил, по которым заявки покидают накопители и каналы: для накопителей – либо правила переполнения, по которым заявки в зависимости от наполнения накопителя покидают систему, либо правила ухода, связанные с истечением времени ожидания заявки в накопителе; для каналов – правила выбора маршрутов или направлений ухода. Кроме того, для заявок необходимо задать правила, по которым они остаются в канале или не допускаются до обслуживания каналом, то есть правила блокировок канала.

Весь набор возможных алгоритмов поведения заявок в системе представляется в виде оператора А.

Таким образом, Q-схема, описывающая процесс функционирования СМО любой сложности, однозначно задается в виде: Q=(W,U,Z,R,H,A), где W – подмножество заявок; U – подмножество обслуживаний; Z – подмножество состояний элементов Q-схемы; R – подмножество взаимосвязей элементов структуры СМО; H – совокупность собственных (внутренних) параметров Q-схемы; A – совокупность алгоритмов функционирования.

Без ограничений позволяют исследовать Q-схему только имитационные модели.

* 1. **Проведение экспериментов с математическими моделями**

Эффективность использования экспериментальных ресурсов существенным образом зависит от выбора плана эксперимента. Причины этого две:

* План эксперимента в значительной степени определяет порядок статистического анализа его результатов;
* Успешность разрешения с помощью эксперимента поставленных экспериментатором задач существенно зависит от его плана.

Итак, основная задача планирования машинных экспериментов – получение необходимой информации об исследуемой системе при ограничении на ресурсы. Наиболее распространенные задачи планирования: уменьшение затрат машинного времени на моделирование, увеличение точности и достоверности результатов моделирования, проверка адекватности модели и т. д.

Важное значение при планировании эксперимента имеют следующие обстоятельства:

* простота повторения условий эксперимента с моделью на ПК;
* возможность управления экспериментом, включая его прерывание и возобновление;
* лёгкость варьирования условий;
* наличие корреляции между последовательностью точек в моделировании.
  + 1. **Основные понятия в теории планирования**

Различают входные и выходные переменные. Переменная может быть фактором или реакцией в зависимости от её роли в эксперименте. *Фактор* является *эндогенной* (управляемой, входной) переменной, а *реакция* – *эндогенной* (выходной). *Эксперимент* – некоторая процедура организации и наблюдения каких-то явлений, которые осуществляются в условиях, близких к естественным, либо имитируют их. Различают пассивный эксперимент, когда исследователь наблюдает протекающий процесс, и активный, когда наблюдатель вмешивается и организует протекание.

Каждый фактор в эксперименте может принимать одно из нескольких значений, называемых *уровнями.* Фиксированный набор уровней определяет одно из возможных состояний рассматриваемой системы. Этот набор представляет собой условие проведения одного из возможных экспериментов. Каждому фиксированному набору уровней факторов соответствует точка в многомерном пространстве, называемом факторным пространством.

Эксперимент необходимо поставить так, чтобы при минимальных затратах ресурсов, варьируя значения факторов, можно было построить адекватную математическую модель системы и оценить её характеристики. При планировании нужно определить основные свойства факторов, которые при проведении эксперимента могут быть управляемыми или неуправляемыми, наблюдаемыми или ненаблюдаемыми, изучаемыми или неизучаемыми.

*Управляемый фактор* – его уровни направленно выбираются в процессе эксперимента. *Наблюдаемый фактор* – если его значения наблюдаются и регистрируются, а экспериментатор заинтересован в исследовании влияния этого фактора. Наблюдаемые факторы должны совпадать с управляемыми. *Наблюдаемые неуправляемые факторы* – сопутствующие или вторичные.

Основные требования, предъявляемые к факторам: управляемость и требование непосредственного воздействия на модель. Если при моделировании изменяется несколько факторов, то появляются дополнительные требования совместимости и независимости. Совместимость – все возможные комбинации факторов осуществимы, независимость соответствует возможности установления фактора на уровень, независимо от уровней других факторов.

* + 1. **Виды планов модельного эксперимента**

При проведении эксперимента для оценки характеристик экспериментатор старается создать такие условия, которые способствуют выявлению влияния факторов, находящихся в функциональной зависимости с искомой характеристикой. Для этого необходимо:

* Отобрать факторы, влияющие на характеристику и описать функциональную зависимость;
* Установить диапазон изменения каждого фактора;
* Определить координаты точек факторного пространства, в которых следует проводить эксперимент;
* Оценить необходимое число реализаций и их порядок.

Эксперимент, в котором реализуются все сочетания уровней факторов, называется *полным факторным экспериментом*. Для уменьшения числа комбинаций используется *дробный факторный эксперимент*, позволяющий вдвое сократить количество экспериментов. Если при эксперименте не исследуется взаимное влияние всех факторов, то проводят *частичный факторный эксперимент.* К частичным относятся планы «латинский квадрат», рандомизированный план и другие.

* + 1. **Тактическое и стратегическое планирование**

При стратегическом планировании эксперимента ставится задача построения оптимального плана эксперимента для достижения цели моделирования. Тактическое планирование преследует частные цели оптимальной реализации каждого конкретного эксперимента из множества необходимых, заданных при стратегическом планировании.

Проблемы стратегического планирования:

* Построение плана машинного эксперимента;
* Наличие большого количества факторов;
* Многокомпонентная функция реакции;
* Стохастическая сходимость результатов машинного эксперимента;
* Ограниченность машинных ресурсов на проведение эксперимента.

Проблемы тактического планирования:

* Определение начальных условий и их влияния на достижение установившегося результата при моделировании;
* Обеспечение точности и достоверности результатов моделирования;
* Уменьшение дисперсии оценок характеристик процесса функционирования моделируемых систем;
* Выбор правил автоматической остановки имитационного эксперимента.
  1. **Анализ результатов моделирования**

Различают корреляционный и дисперсионный анализ. Основной задачей *корреляционного анализа* является оценка силы связи и проверка статистических гипотез о наличии и силе связи. Не все факторы, влияющие на процессы, являются случайными, поэтому при анализе обычно рассматриваются связи между случайными и неслучайными величинами. Такие связи называются регрессионными, а метод статистики, изучающий их – регрессионный анализ. *Корреляционный анализ* – оценка уравнения регрессии. *Корреляция* – статистическая зависимость между СВ, не имеющими строгого функционального характера, при которой изменение одной СВ приводит к изменению математического ожидания другой. Различают парную, частную и множественную корреляцию. *Парная регрессия*: связь между двумя признаками. *Частная регрессия*: зависимость между результатом и одним факторным признаком при фиксированном значении других факторных признаков. *Множественная регрессия*: зависимость результатов от двух и более факторных признаков.

Регрессия исследует форму зависимости. Преимущество регрессионного анализа: возможность построения модели, в которой зависимость отклика аналитически определяется как функция от нескольких независимых переменных.

2 **ЭТАПЫ СОЗДАНИЯ МОДЕЛИ**

Процесс создание модели можно разделить на следующие этапы:

* Постановка задачи
* Построение концептуальной схемы модели
* Формализация поставленной задачи, сбор данных статистики и их обработка
* Выбор модели
* Построение модели
* Проверка адекватности модели
* Планирование эксперимента
* Проведение эксперимента, сбор результатов
* Анализ результатов
  1. **Постановка задачи (Задание №3)**

Система обработки информации содержит мультиплексный канал и три миниЭВМ. Сигналы от датчиков поступают на вход канала через интервалы времени 5-15 мкс. В канале они буферизуются и предварительно обрабатываются в течении 7-13 мкс. Затем они поступают на обработку в ту миниЭВМ, где имеется наименьшая по длине входная очередь. Емкости входных накопителей во всех миниЭВМ рассчитаны на хранение величин 10 сигналов. Время обработки сигнала в любой миниЭВМ равно 33 мкс.

*Цель исследования*: Смоделировать процесс обработки 500 сигналов, поступающих с датчиков. Определить средние времена задержки сигналов в канале и миниЭВМ и вероятности переполнения входных накопителей. Обеспечить ускорение обработки сигнала в ЭВМ до 25 мкс при достижении суммарной очереди сигналов значения 25 единиц.

* 1. **Формализация задачи, структурная и концептуальная схемы**

На первом этапе машинного моделирования формулируется модель и строится ее формальная схема, то есть основным назначением этого этапа является переход от содержательного описания объекта к его математической модели, другими словами, процесс формализации.

На приведенном ниже рисунке 1 представлена структурная схема системы.

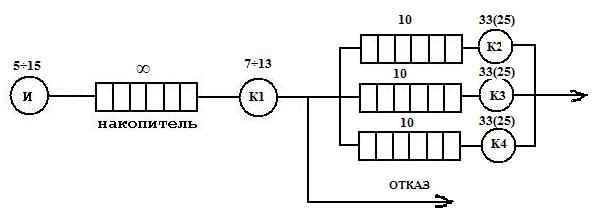


Рисунок 1 – Структурная схема системы

Структурная схема описывает составные части, может содержать некоторые параметры, но при этом не поясняет функционирование реальной системы. Функционирование системы поясняется на основе концептуальной схемы (рисунок 2), которая отображает причинно-следственные связи между компонентами системы.

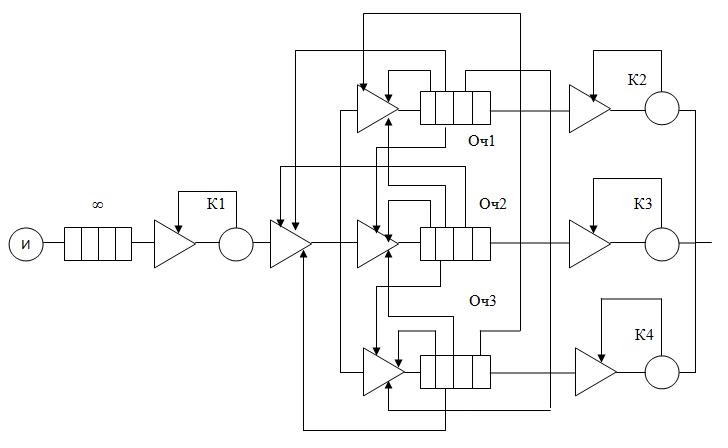


Рисунок 2 – Концептуальная схема моделируемой системы

В отличие от структурной схемы концептуальная схема сдержит логические вентили, которые указывают на условия продвижения заявок в системе.

После перехода от описания моделируемой системы к ее модели необходимо построить математические модели процессов, происходящие в различных блоках. Математическая модель представляет собой совокупность соотношений, определяющих характеристики процесса функционирования системы в зависимости от структуры системы, алгоритмов поведения, параметров системы, воздействий внешней среды, начальных условий и времени. Математическая модель является результатом формализации процесса функционирования исследуемой системы, то есть построения формального (математического) описания процесса с необходимой, в рамках проводимого исследования, степенью приближения к действительности.

Однако на практике получение модели достаточно простого вида для сложных систем чаще всего невозможно, поэтому обычно процесс функционирования системы разбивается на ряд элементарных подпроцессов. При этом необходимо так производить разбиение, чтобы построение моделей отдельных подпроцессов было элементарно и не вызывало трудностей при формализации. Таким образом, на этой стадии сущность формализации подпроцессов будет состоять в подборе типовых математических схем. Для стохастических процессов ими являются схемы массового обслуживания(СМО).

Основные требования, предъявляемые к модели процесса функционирования системы:

* Полнота модели, то есть пользователь может получать набор оценок исследуемых характеристик системы с заданной точностью и достоверностью;
* Гибкость модели должна давать возможность варьирования структуры, алгоритмов и параметров системы;
* Длительность разработки и реализации модели сложной системы должна быть по возможности минимальной;
* Структура модели должна быть блочной и допускать удаление, добавление и замену блоков без переделки всей модели;
* Программные и технические средства должны обеспечивать эффективную машинную реализацию модели и удобное обращение с ней пользователя;
* Должно быть организовано проведение целенаправленных экспериментов с моделью системы с использованием аналитико-имитационного подхода.
  1. **Выбор и построение модели**

В данной курсовой работе была выбрана имитационная модель, так как состав средней по сложности имитационной модели полностью реализуется на современных персональных компьютерах и позволяет исследовать имитационную модель, задаваемую в виде определенной совокупности отдельных блочных модулей и связей между ними в их взаимодействии в пространстве и времени при реализации какого-либо процесса.

Имитационная модель характеризуется набором переменных, с помощью которых удается управлять изучаемым процессом, и набором начальных условий, когда можно изменять условия проведения машинного эксперимента. Для полного анализа характеристик процесса функционирования систем приходится многократно воспроизводить имитационный эксперимент, варьируя исходные данные задачи. При этом, как следствие, возникает увеличение затрат машинного времени.

После всех выше проведённых действий выполняется построение блок-схемы модели.

Общий алгоритм моделирования системы представлен на рисунке 3.

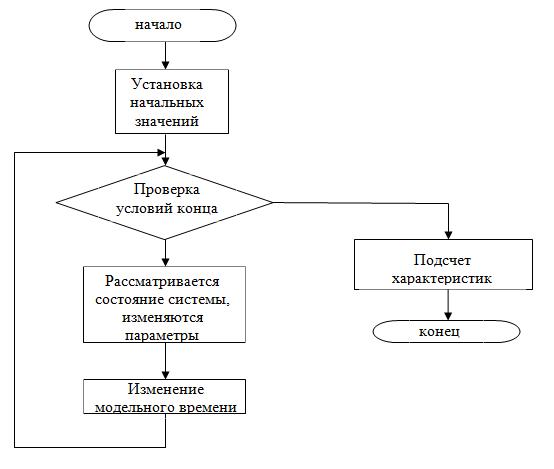


Рисунок 3 – Общий алгоритм моделирования системы

**3 МОДЕЛИРОВАНИЕ В СРЕДЕ MATLAB**

**3.1 Основные положения**

Система MatLAB предназначена для автоматизации инженерных и научных расчетов. MatLAB ориентирована на проведение вычислений с массивами вещественных и комплексных чисел (вектора и матрицы), содержит большое количество процедур и функций, реализующих сложные алгоритмы численной математики. Программа Simulink является приложением к пакету MatLAB. При моделировании с использованием Simulink реализуется принцип визуального программирования, в соответствии с которым, пользователь на экране из библиотеки стандартных блоков создает модель устройства и осуществляет расчеты.

В ходе моделирования имеется возможность следить за процессами, происходящими в системе. Для этого используются специальные устройства наблюдения, входящие в состав библиотеки Simulink. Результаты моделирования могут быть представлены в виде графиков или таблиц.

MatLAB интерфирует технические компоненты среды, которые сочетают в себе цифровые вычисления, графику, визуализацию и язык программирования. StateFlow позволяет моделировать поведение системы. Stateflow представляет собой графический инструментарий для проектирования сложных систем управления и является самым значительным дополнением к среде Simulink. Подсистема Stateflow дает возможность моделировать поведение сложных событийно-управляемых систем. В основе Stateflow лежит специальная форма представления гибридного поведения - карта состояний. Основные элементы карты состояний - это «состояния» (state) и «переходы» (transitions).

StateChart – основная диаграмма состояний и их переходов. Основные графические компоненты диаграммы: состояние, переход, событие, действие. *Событие* – нечто, происходящее вне рассматриваемой системы, возможно требующее при этом некоторых ответных действий. События считаются мгновенными для выбранного уровня абстрагирования. *Действие* – реакции моделируемой системы на события. Также мгновенны. *Состояние* – условие, в котором пребывает моделируемая система некоторое время, в течение которого система ведёт себя одинаковым образом. *Переход* – смена состояний, вызванная событием. Суперсостояние (гиперсостояние) – объединяет несколько состояний, имеющих идентичную реакцию на одно и то же событие.

С точки зрения StateFlow состояние описывает режим управляемой событиями системы. состояний - это «состояния» (state) и «переходы» (transitions).

Каждое состояние описывает один режим в работе событийно-управляемой системы. Состояние становиться активным, если срабатывает условие перехода, ведущего к этому состоянию или если на это состояние установлен переход по умолчанию. Каждое состояние на диаграмме Stateflow имеет «родителя», которым, по умолчанию, является сама диаграмма Stateflow.

Подсостояние – это состояние, которое может быть активно, только если активно состояние, называемое его родителем. Для того, чтобы создать подсостояние, необходимо, выбрав кнопку State, щелкнуть мышкой в поле того состояния, которое предполагается иметь родительским, причем вложенность подсостояний может быть произвольной. Состояние с его подсостояниями можно сгруппировать. Это удобно использовать, если необходимо переместить такое состояние с одного места на другое, но при этом не нарушить расположение его подсостояний относительно друг друга. Для этого необходимо, щелкнув правой кнопкой мышки, выбрать в появившемся меню пункт Make Contents/Group, или просто дважды щелкнуть левой кнопкой мышки в поле состояния. Сгруппированные состояния Simulink выделяет тем, что обводит их утолщенной линией. Разгруппировать состояния можно опять же дважды щелкнув в поле состояния мышкой.

Система с параллелизмом позволяет иметь два или более состояний, которые могут быть активны в одно и тоже время. Каждое из них работает независимо от других таких же состояний.

Переход представляет собой объект, связывающий между собой два состояния. Переход на диаграмме Stateflow представляется стрелкой, начало которой относиться к состоянию-источнику, а конец – к состоянию-адресату. Переход на диаграмме задается неявно. Для этого достаточно, установив мышку у состояния-источника, нажать левую кнопку мышки и, удерживая ее, протянуть появившуюся стрелку до состояния-адресата.

Переход имеет метку, которая описывает условия срабатывания перехода и выполняемые при этом действия.

Переменные – это неграфические объекты на диаграмме Stateflow, предназначенные для хранения числовых данных. Переменные могут использоваться на любом уровне иерархии.

Различают следующие виды переменных:

* входные;
* выходные;
* локальные;
* константы;
* существующие только в течение определенного временного интервала;
* определенные в рабочем пространстве Matlab;
* импортируемые из источника, внешнего к диаграмме Stateflow и диаграмме Simulink;
* экспортируемые адресату, внешнему к диаграмме Stateflow и диаграмме Simulink.

Для того чтобы создать входные (выходные и т.д.) переменные в блоке Stateflow, необходимо в окне соответствующей диаграммы Stateflow в меню выбрать соответствующий пункт (Add/Data…/Input from Simulink для создания входной, Add/Data…/Output to Simulink для создания выходной и Add/Data…/Local для создания локальной переменной) и в открывшемся окне ввести имя соответствующей переменной и другие ее характеристики.

Для переходов состояний из неактивного в активное необходимо событие или условие. События бывают: вход, выход, в течение и создание собственного события. Метки переходов описывают обстоятельства, под действием которых система переходит от одного состояния к другому. Если событие не указано, переход будет осуществляться при любом событии. Переходы, которые заканчиваются в соединениях, могут иметь только действия условий. *Условие* – это заключённое в скобки логическое выражение, определяющее произойдёт переход или нет.

**3.2 Разработка и описание модели**

В ходе разработки модели были использованы следующие переменные:

* Tmew- время между поступлением заявок;
* Tobs- время обслуживания заявки в канале предварительной обработки;
* Tobs1- время обслуживания заявки в ЭВМ 1;
* Tobs2- время обслуживания заявки в ЭВМ 2;
* Tobs3- время обслуживания заявки в ЭВМ 3;
* Tobssum- суммарное время, затраченное на обслуживание заявок каналом предварительной обработки;
* Tobs1sum- суммарное время, затраченное на обслуживание заявок ЭВМ\_1;
* Tobs2sum- суммарное время, затраченное на обслуживание заявок ЭВМ\_2;
* Tobs3sum- суммарное время, затраченное на обслуживание заявок ЭВМ\_3;
* Bocher- бесконечная очередь;
* Kolzajv- количество сгенерированных заявок;
* Osv\_kan0- переменная флаг, показывающая состояние канала предварительной обработки (занят-1/свободен-0);
* Prihod- переменная флаг, показывающая приход заявки (есть-1 /нет-0);
* Raspr- переменная, служащая для запоминания распределения;
* Ver\_per- вероятность переполнения входных накопителей;
* Total\_time- общее время работы системы;
* N1- переменная, показывающая количество заявок в накопителе 1;
* N2- переменная, показывающая количество заявок в накопителе 2;
* N3- переменная, показывающая количество заявок в накопителе 3;
* Kol\_otkaz- количество отказов;
* kol\_z1- количество заявок обслуженных ЭВМ 1;
* kol\_z2- количество заявок обслуженных ЭВМ 2;
* kol\_z3- количество заявок обслуженных ЭВМ 3;
* svzsevm1- среднее время задержки сигналов в ЭВМ 1;
* svzsevm2- среднее время задержки сигналов в ЭВМ 2;
* svzsevm3- среднее время задержки сигналов в ЭВМ 3;
* svzskan- среднее время задержки сигналов в канале.

Каждому блоку структурной схемы модели соответствует блок в Stateflow модели. Добавлены также два блока, которые обеспечивают подсчет статистических данных.

Перечень блоков, входящих в Stateflow модель, и их обозначения:

* 2- блок генерации заявок;
* 4- блок – бесконечная очередь;
* 3- канал предварительной обработки заявок;
* 5- блок распределения заявок по 3-м накопителям;
* 1,6,9- мини-ЭВМ 1,2,3 обработки заявок;
* 8,7- блоки, в которых подсчитываются статистические данные.

Общий вид Stateflow- модели представлен в приложении А. Внутренняя структура, представленная в виде совокупности блоков, представлена в приложении Б.

Рассмотрим подробно каждый из блоков:

* Блок генерации заявок Istochnik изображен на рисунке 4.

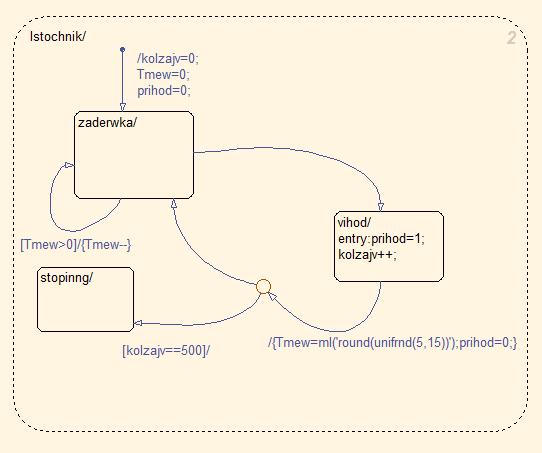


Рисунок 4 – Блок генерации заявок

В данном блоке осуществляется генерация заявок для модели и остановка их генерации. Для реализации этого блок содержит 3 подблока: zaderwka, vihod и stopinng.

В блок zaderwka система переходит автоматически по безусловному переходу. Далее происходит переход в блок vihod, так как время задержки еще не сгенерировано. В блоке vihod происходит наращивание счетчика количества заявок и при следующем переходе генерируется время задержки между генерациями заявок. Этот переход осуществляется в соединение, которое организует разветвление: при достижении количества заявок равного 500 осуществляется переход в блок stopinng для завершения генерации, в ином случае переход осуществляется в блок zaderwka, где реализуется задержка между генерациями заявок с помощью декрементирования соответствующей переменной.

* Блок «бесконечная очередь» представлен на рисунке 5.

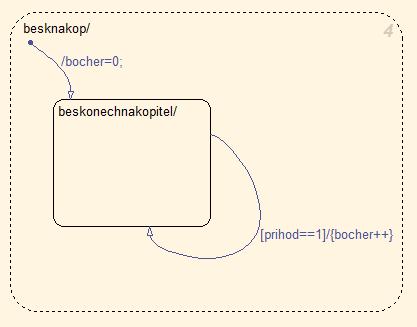


Рисунок 5 - Блок «бесконечная очередь»

Данный блок содержит лишь один блок. Суть работы заключается в инкрементировании переменной «длина очереди» по приходу заявки.

* Канал предварительной обработки заявок изображен на рисунке 6.

Данный блок включает в себя два подблока (vhod и obrabotka) и два соединения. Блок vhod организует при переходе в соединение поступление заявок из очереди и генерацию времени обработки заявки в канале. Переход из соединения в соединение вычисляет суммарное время обслуживания. Переход из соединения в блок obrabotka уменьшает длину бесконечной очереди перед данным каналом. Затем в этом блоке осуществляется циклический переход, обеспечивающий задержку обработки заявки в канале. Далее происходит переход из блока obrabotka в блок vhod, оповещающий об освобождении канала.

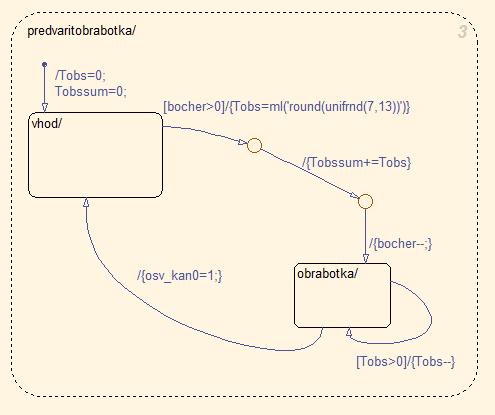


Рисунок 6 - Канал предварительной обработки заявок

* Блок распределения заявок по трем накопителям представлен в приложении В.

Данный блок анализирует состояния накопителей перед ЭВМ и направляет заявку в накопитель с наименьшей очередью, а также фиксирует наличие отказа в случае, когда все накопители заняты.

* Мини-ЭВМ 1,2,3 обработки заявок имеют одинаковую структуру, которая представлена в приложении Г.

Структура и принцип работы данных блоков схожи с блоком предварительной обработки заявок. Отличие заключается в том, что время обработки заявок в канале не генерируется, а зависит от суммарной длины очередей перед каналами.

* Блоки, в которых подсчитываются статистические данные приведены в приложении Д.

Данные блоки ведут статистику необходимых характеристик процесса моделирования и обеспечивают их вывод.

1. **МОДЕЛИРОВАНИЕ В СРЕДЕ GPSSW**

* 1. **Общие сведения о языке GPSS World**

Система GPSS World - это мощная среда компьютерного моделирования общего назначения, разработанная для профессионалов в области моделирования. Это комплексный моделирующий инструмент, охватывающий области как дискретного, так и непрерывного компьютерного моделирования, обладающий высочайшим уровнем интерактивности и визуального представления информации. Использование GPSS World дает возможность оценить эффект конструкторских решений в чрезвычайно сложных системах реального мира.

Для описания имитационной модели на языке GPSS полезно представить её в виде схемы, на которой отображаются элементы СМО - каналы, накопители, узлы и источники. Описание на языке GPSS есть совокупность операторов (блоков), характеризующих процессы обработки заявок. Имеются операторы и для отображения возникновения заявок, задержки их в каналах, занятия памяти, выхода из СМО, изменения параметров заявок (например, приоритетов), вывода на печать накопленной информации, характеризующей загрузку устройств, заполненность очередей и т.п. Каждый транзакт, присутствующий в модели, может иметь до 12 параметров. Существуют операторы, с помощью которых можно изменять значения любых параметров транзактов и операторы, характер исполнения которых зависит от значений того или иного параметра обслуживаемого транзакта. Для моделирования используется событийный метод. Соблюдение правильной временной последовательности имитации событий в СМО обеспечивается интерпретатором GPSS- программной системой, реализующей алгоритмы имитационного моделирования.

* 1. **Операторы языка GPSS используемые в работе**

В записи оператора выделяют такие части: метку, название, поле переменных и комментарий.

GENERATE - вводит транзакты в модель.

Формат: GENERATE А, В, С, D, E, F, G

А - среднее значение интервала времени;

В - разброс или модификатор среднего значения (по умолчанию ноль);

С - время появления первого транзакта;

D - общее число генерируемых транзактов;

Е - уровень приоритета каждого транзакта; (от 0 до 127, значение по умолчанию 0);

F - число параметров (по умолчанию 12);

G - тип параметра (F - полнословный, Н - полусловный - по умолчанию).

Вводит транзакты в модель, посылая их в следующий по порядку блок. Если в поле В не указана функция, то интервал между поступлением транзактов определяется случайным числом, равномерно распределенным в диапазоне от (А - В) до (А + В). Если поле В является функцией (FN$), то этот интервал определяется произведением поля А на значение функции, заданной в поле В.

QUEUE - помещает транзакт в конец бесконечной очереди.

Формат: QUEUE А, В

А - номер очереди (числовое или символьное имя очереди);

В - число добавляемых к очереди элементов (по умолчанию 1).

Увеличивает текущее содержимое очереди, указанной в поле А, на значение в поле В. Если поле В не определено, увеличивает содержимое очереди на единицу. Транзакт может находиться в двух различных очередях одновременно.

DEPART - удаляет транзакт из бесконечной очереди.

Формат: DEPART А, В

А - номер (имя) очереди;

В - число удаляемых из очереди элементов.

Удаляет текущий транзакт из очереди, указанной в поле А, и уменьшает содержимое очереди на значение поля В. Транзакт может находиться одновременно в двух различных очередях.

SEIZE - занимает устройство.

Формат: SEIZE A

А - номер устройства.

Транзакт пытается занять устройство, определенное полем А. Если прервано, транзакт задерживается в предыдущем блоке.

RELEASE - освобождает устройство.

Формат: RELEASE A

А - номер устройства (числовое или символьное имя освобождаемого устройства).

Устройство, указанное в поле А, освобождается и становится доступным для других транзактов. Освобождать устройство должен тот же транзакт, который его занимал.

ADVANCE – задерживает транзакт.

Формат: ADVANCE А, В

А – среднее время задержки (константа, если В не задано);

В – разброс относительно среднего значения, должен быть меньше или равен А.

Блок ADVANCE моделирует временную задержку транзакта в течение определенного интервала. Значение задержки по умолчанию равно нулю. Если поле В не является FN$, то время задержки является случайным числом, распределенным равномерно на интервале от (А + В) до (А - В). Если поле В является функцией FN$, то время задержки определяете произведением поля А на значение функции в поле В.

Функция может быть использована для задания времени задержки с определенным средним значением и отклонением, зависящим от А.

Блок STORAGE – память.

Формат: имя STORAGE A

имя —имя многоканальных устройств (МКУ), используемое для ссылок на него;

A —емкость (количество каналов обслуживания) МКУ, задаваемая константой.

ENTER - для занятия обслуживания МКУ используется пара блоков.

Формат: ENTER   A,B

A - указывается номер или имя МКУ;

B *-* число каналов МКУ, занимаемых при входе в блок ENTER.

При входе транзакта в блок ENTER текущее содержимое МКУ увеличивается на число единиц, указанное в поле B*.* Если свободная емкость МКУ меньше значения поля B, то транзакт не может войти в блок ENTER и остается в предыдущем блоке, образуя очередь в списке текущих событий.

LEAVE - для освобождения памяти.

Формат: LEAVE A,B

A указывается номер или имя МКУ;

B *-* число каналов МКУ освобождаемых при входе в блок LEAVE.

При входе транзакта в блок LEAVE текущее содержимое МКУ уменьшается на число единиц, указанное в поле B. Не обязательно освобождается такое же число каналов МКУ, какое занималось при входе данного транзакта в блок ENTER, однако текущее содержимое МКУ не должно становиться отрицательным.

TRANSFER - изменяет движение транзакта в модели.

Формат: TRANSFER А, В, С, D

А - режим передачи (" ",., ALL, BOTH, FN, P, PICK, SBR,SIM);

В - номер блока;

С – номер блока;

D — значение индекса, используемое в режиме ALL.

Транзакт направляется в блок, определяемый в соответствии с режимом передачи, указанным в поле А.

Режимы передачи поля А:

* " " - транзакт передается в блок, определяемый полем В;
* "." - статистический режим; в поле А указано десятичное число, выражающее вероятность перехода в блок С; его дополнение до единицы указывает вероятность перехода в блок В.
* ALL - транзакт последовательно пытается перейти в блоки, определяемые значениями В, B+D, B+2D С.
* BOTH - транзакт последовательно пытается войти в блок В, затем в блок С, до тех  
  пор, пока один из них станет доступным.
* FN - функциональный режим: поле В является номером функции; следующий блок определяется суммой значения этой функции поля С.
* Р - параметрический режим: поле В является номером параметра; следующий блок определяется суммой значения этого параметра и поля С.
* PICK - выборочный режим: блок выбирается с равной вероятностью из блоков с  
  номерами: В, В+1,..., С.
* SBR - режим перехода к подпрограмме, номер текущего блока помещается в параметр, указанный в поле С, а транзакт передается в блок, номер которого указан  
  в поле В.
* SIM - одновременный режим: проверяется одновременное выполнение условий беспрепятственного движения транзактов в задерживающих блоках. Если условие  
  выполняется, транзакт передается в следующий блок, в противном случае транзакт переходит на блок С.

START - управление процессом моделирования.

Формат: STAR[T] А, В, С, D

А - счетчик числа завершений;

В - подавление вывода на печать (В = NP);

С - промежуточный вывод статистики; 0 - распечатка списков.

GATE - служит для задержки или изменения маршрута транзактов в зависимости от состояния аппаратных объектов модели.

Формат: имя GATE X A,B

X- вспомогательный операнд, содержит код состояния проверяемого аппаратного объекта. Операнд X может принимать следующие значения: U (устройство занято); NU (устройство свободно); I (устройство захвачено); NI (устройство не захвачено); SE (МКУ пусто); SNE (МКУ не пусто); SF (МКУ заполнено); SNF (МКУ не заполнено); LS ( ЛП включен), LR (ЛП выключен).

A, В - указывают имя или номер этого объекта. Если проверяемый объект находится в заданном состоянии, то блок GATE пропускает транзакт к следующему блоку. Если же заданное в блоке условие не выполняется, то транзакт переходит к блоку, указанному в поле B, а если это поле пусто, то задерживается перед блоком GATE.

TERMINATE - служит для удаления транзактов из модели.

Формат: TERMINATE A

А - указывает, на сколько единиц уменьшается содержимое счётчика завершений при входе транзакта в данный блок TERMINATE. Если поле А не определено, то оно считается равным 0 и транзакты, проходящие через такой блок, не уменьшают содержимого счётчика завершений.

TEST – перемещае транзакт согласно сравнению.

Формат: TEST Х А, В, С

Х – условный оператор может принимать значения:

* G – больше;
* GE – больше или равно;
* L – меньше;
* LE – меньше или равно;
* E – равно;
* NE – не равно.

А и В – сравниваемые величины.

С – метка, по которой переходит транзакт, если условие не выполняется.

Также в программу ведены переменные, для отслеживания занятости очередей:

aa VARIABLE Q$och1

ba VARIABLE Q$och2

caa VARIABLE Q$och3

daa VARIABLE Q$och3+Q$och2+Q$och1

Текст программы приведен в приложении Е.

Отчет по результатам моделирования представлен в приложении Ж.

1. **МОДЕЛИРОВАНИЕ В СРЕДЕ DELPHI**
   1. **Общие сведения о языке высокого уровня Borland Delphi**

Delphi - это визуальная среда разработки программ. Язык программирования высокого уровня Borland Delphi достаточно прост для освоения, но очень эффективен и достаточно мощный, он обеспечивает самый удобный и быстрый способ создания прикладных программ (в частности их интерфейсов) по сравнению с другими предложенными языками программирования высокого уровня. В Delphi собрана большая коллекция компонентов, позволяющая визуально создать любой интерфейс программы.

Язык программирования высокого уровня Borland Delphi поддерживает объектно-ориентированное программирование (ООП), а используя средства ООП можно легко создать имитационную модель СМО, представляя каждую подсистему СМО в виде отдельного класса. В свойствах класса можно хранить параметры подсистемы, а в методах - алгоритм её работы.

**5.2 Принцип разработки программы**

При моделировании системы на языке высокого уровня целесообразно предварительно разработать подробный алгоритм функционирования системы. Такой алгоритм соответствует приведенному на рисунке 3 общему алгоритму моделирования, но каждый блок в нем подробно описан. Данный алгоритм приведен в приложении З.

По данному алгоритму составлена программа, код которой содержится в приложении И.

А в приложении К изображен интерфейс разработанной программы.

В проект включены следующие переменные:

* + - * Tpos – время поступления заявки;
      * Tobr1 – время обработки заявки на канале предварительного обслуживания;
      * Tosv1 – время освобождения канала предварительного обслуживания;
      * T – такт модельного времени;
      * L1 – длина бесконечной очереди;
      * N – количество обработанных заявок;
      * K1 – занятость канале предварительного обслуживания;
      * K2 – занятость ЭВМ 1;
      * Tobr – время обработки на ЭВМ 1,2,3;
      * Tosv2 – время освобождения ЭВМ 1;
      * Otk – количество отказов;
      * K3 – занятость ЭВМ 2;
      * K4 – занятость ЭВМ 3;
      * L2 – длина очереди 1;
      * L3 – длина очереди 2;
      * L4 – длина очереди 3;
      * Tosv3 - время освобождения ЭВМ 2;
      * Tosv4 - время освобождения ЭВМ 3;
      * V – вероятность отказов.

При моделировании системы в среде Borland Delphi применен принцип Δz для продвижения модельного времени. Таким образом рассматриваются лишь те моменты времени, когда наступает в системе некоторое событие, которое влечет за собой изменение состояния всей системы. Таким событием могут быть:

* Поступление заявки в систему;
* Освобождение канала предварительной обработки;
* Освобождение одной из ЭВМ.

Поэтому при определении следующего рассматриваемого такта выбирается событие, которое наступит раньше всех, то есть с минимальным временем наступления. Возможны такие случаи, что данные события наступают одновременно. Это связано с тем, что в качестве модельного времени используются дискретные такты времени с целочисленными значениями.

1. **АНАЛИЗ ПОЛУЧЕННЫХ РЕЗУЛЬТАТОВ**

**6.1 Сравнение результатов моделирование системы, полученных в трех моделях**

С помощью разработанных моделей системы были проведены эксперименты. В результате получены следующие значения для заданных характеристик (среднего времени пребывания заявки в системе и вероятности отказов):

* В среде MatLAB:

- вероятность отказов равна нулю;

* В среде GPSSW:

- вероятность отказов равна нулю;

- среднего времени пребывания заявки - 266.835;

* В среде Delphi:

- вероятность отказов равна нулю;

- среднего времени пребывания заявки - 266,126.

По этим результатам можно сделать вывод, что все разработанные модели моделируют реальную систему приблизительно одинаково и выдают близкие по значению результаты.

**6.2 Планирование и проведение эксперимента**

Имитационное моделирование является по сути своей машинным экспериментом с моделью исследуемой или проектируемой системы. План имитационного эксперимента на ЭВМ представляет собой метод получения с помощью эксперимента необходимой пользователю информации. Эффективность использования экспериментальных ресурсов существенным образом зависит от выбора плана эксперимента, т.к. именно план определяет порядок и объем проведения вычислений на ЭВМ, приемы накопления и статистической обработки результатов моделирования системы S. Поэтому задача планирования машинных экспериментов с моделью Мм формулируется следующим образом: необходимо получить информацию об объекте моделирования, заданном в виде моделирующего алгоритма (программы), при минимальных затратах машинных ресурсов на реализацию процесса моделирования.

При кибернетическом подходе к планированию эксперимента различают входные и выходные переменные: х1, х2,…,хк,у1,у2,…,уi. В зависимости от того, какую роль играет каждая переменная в проводимом эксперименте, она может являться либо фактором или реакцией. В экспериментах с машинными моделями Мм системы S фактор является экзогенной или управляемой ( входной) переменной, а реакция - эндогенной ( выходной) переменной.

Каждый фактор xi (i=) может принимать одно из нескольких значений, называемых уровнями. Фиксированный набор уровней факторов определяет одно из возможных состояний рассматриваемой системы. Одновременно этот набор представляет собой условия проведения одного из возможных экспериментов.

Каждому фиксированному набору уровней факторов соответствует определенная точка в многомерном пространстве, называемом факторным пространством. Эксперименты не могут бать реализованы во всех точках фаторного пространства, а лишь в принадлежащих допустимой области.

Существует вполне определенная связь между уровнями факторов и реакцией (откликом) системы, которую можно представить в виде соотношения: 

Функцию, связывающую реакцию с фактором в выше приведенном выражении называют функцией реакции, а геометрический образ, соответствующий функции реакции,- поверхностью реакции.

При проведении машинного эксперимента с моделью для оценки ха­рактеристик процесса функционирования исследуемой системы необхо­димо создать такие условия, которые способствовали бы выявлению влияния факторов, находящихся в функциональной связи с искомой характеристикой. Для этого необходимо: отобрать факторы  влияющие на искомую характеристику, и описать функциональную зависимость; установить диапазон изменения факторов  определить координаты точек факторного пространства  в котором следует проводить эксперимент; оценить необходимое число реализаций и их порядок в эксперименте.

Свойства объекта исследования, т.е. процесса машинного моделиро­вания системы S, можно описывать с помощью различных методов (моделей планирования).

Получение модели, описывающей реакции изучаемой системы S на многофакторное возмущение, — одна из задач математического плани­рования эксперимента. Наиболее распространенными и полно отвечаю­щими задачам статистического моделирования являются полиномиальные модели. Задача нахождения полиномиальной модели, описывающей систему или отдельные ее характеристики, состоит в оценке вида и пара­метров некоторой функции 

В данной работе необходимо провести машинный эксперимент по исследованию характеристик системы, построить план эксперимента, описать модель планирования машинного эксперимента, получить оценки коэффициентов модели и провести планируемый имитационный эксперимент с моделью.

Необходимо оценить зависимость загрузки канала предварительной обработки от времени поступления заявок, времени обслуживания заявки на данном канале и на канале ЭВМ 1.

Для составления плана эксперимента выделим следующие факторы:

* время поступления заявок - х1;
* времени обслуживания заявки на канале предварительной обработки - х2;
* время обслуживания в канале ЭВМ 1 – х3.

Определяем локальную подобласть планирования эксперимента пу­тем выбора основного (нулевого) уровня  и интервалов варьирова­ния для каждого выбранного фактора. Сведем полученные данные в табл.1.

Таблица 1 – Локальная область планирования эксперимента

|  |  |  |  |
| --- | --- | --- | --- |
| Факторы | Уровни факторов | | |
| -1 | 0 | +1 |
|  | 8 | 10 | 12 |
|  | 8 | 10 | 12 |
|  | 25  20 | 33  25 | 40  30 |

Существует вполне определенная зависимость между уровнями фак­торов и реакцией системы, которую представим в виде соотношения .

Для определения зависимости  строим математическую (аналити­ческую) модель планирования в виде полинома первого порядка (уравнение 1).

 (1)

Выбранная модель включает в себя линейные члены полинома и их произведения. Для оценки коэффициентов модели используем план эксперимента типа 23­ , то есть полнофакторный эксперимент. Составим матрицу планирования, представленную в табл.2.

Таблица 2 – Матрица планирования

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| х1 | х2 | х3 | х1х2 | х1х3 | х2х3 | х1х2х3 | уi |
| -1 | -1 | -1 | 1 | 1 | 1 | -1 | у1 |
| -1 | -1 | 1 | 1 | -1 | -1 | 1 | у2 |
| -1 | 1 | -1 | -1 | 1 | -1 | 1 | у3 |
| -1 | 1 | 1 | -1 | -1 | 1 | -1 | у4 |
| 1 | -1 | -1 | -1 | -1 | 1 | 1 | у5 |
| 1 | -1 | 1 | -1 | 1 | -1 | -1 | у6 |
| 1 | 1 | -1 | 1 | -1 | -1 | -1 | у7 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | у8 |

В соответствии с матрицей планирования проведем эксперимент и получим значения уi, представленные в таблице 3.

Таблица 3 – Результаты экспериментов

|  |  |
| --- | --- |
| у1 | 0,994 |
| у2 | 0,971 |
| у3 | 0,998 |
| у4 | 0,998 |
| у5 | 0,669 |
| у6 | 0,675 |
| у7 | 0,993 |
| у8 | 0,996 |

Вычислим значения коэффициентов bi с помощью пакета Excel по уравнениям (2).

|  |  |  |
| --- | --- | --- |
| A\*X=Y | A(-1)\*A\*X=A(-1)\*Y | X=A(-1)\*Y |

(2)

Матрица А представлена таблицей 4.

Таблица 4 – Матрица А

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| b0 | х1 | х2 | х3 | х1х2 | х1х3 | х2х3 | х1х2х3 |
| 1 | -1 | -1 | -1 | 1 | 1 | 1 | -1 |
| 1 | -1 | -1 | 1 | 1 | -1 | -1 | 1 |
| 1 | -1 | 1 | -1 | -1 | 1 | -1 | 1 |
| 1 | -1 | 1 | 1 | -1 | -1 | 1 | -1 |
| 1 | 1 | -1 | -1 | -1 | -1 | 1 | 1 |
| 1 | 1 | -1 | 1 | -1 | 1 | -1 | -1 |
| 1 | 1 | 1 | -1 | 1 | -1 | -1 | -1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

Матрица У представлена таблицей 5.

Таблица 5 – Матрица У

|  |
| --- |
| Y |
| 0,994 |
| 0,971 |
| 0,998 |
| 0,998 |
| 0,669 |
| 0,675 |
| 0,993 |
| 0,996 |

Обратная матрице А представлена таблицей 6.

Таблица 6 – Обратная матрица А

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0,125 | 0,125 | 0,125 | 0,125 | 0,125 | 0,125 | 0,125 | 0,125 |
| -0,125 | -0,125 | -0,125 | -0,125 | 0,125 | 0,125 | 0,125 | 0,125 |
| -0,125 | -0,125 | 0,125 | 0,125 | -0,125 | -0,125 | 0,125 | 0,125 |
| -0,125 | 0,125 | -0,125 | 0,125 | -0,125 | 0,125 | -0,125 | 0,125 |
| 0,125 | 0,125 | -0,125 | -0,125 | -0,125 | -0,125 | 0,125 | 0,125 |
| 0,125 | -0,125 | 0,125 | -0,125 | -0,125 | 0,125 | -0,125 | 0,125 |
| 0,125 | -0,125 | -0,125 | 0,125 | 0,125 | -0,125 | -0,125 | 0,125 |
| -0,125 | 0,125 | 0,125 | -0,125 | 0,125 | -0,125 | -0,125 | 0,125 |

Матрица Х искомых коэффициентов представлена таблицей 7.

Таблица 7 – Матрица искомых коэффициентов

|  |
| --- |
| bi |
| 0,91175 |
| -0,0785 |
| 0,0845 |
| -0,00175 |
| 0,07675 |
| 0,004 |
| 0,0025 |
| -0,00325 |

По полученным результатам можно сделать выводы о влиянии факторов на заданную характеристику:

* + при увеличении времени между поступлениями заявок загрузка канала предварительной обработки снижается;
  + при увеличении время обработки заявки на данном канале загрузка канала предварительной обработки увеличивается;
  + коэффициент при третем факторе приблизительно равен нулю, а это свидетельствует о том, что данный фактор не влияет на заданную характеристику.

Подставим в уравнение (1) найденные коэффициенты из таблицы 7 и значения переменных основного уровня из таблицы 1. Получим теоретическое значение характеристики и сравним его с практическим.

Теоретическое значение: у=0,9.

Экспериментальное значение: у= 0.985.

Значения приблизительно равны. Отклонение объясняется наличием вероятностного фактора при моделировании и погрешностью расчетов.

**ЗАКЛЮЧЕНИЕ**

В процессе выполнения курсового проекта были разработаны три имитационные модели системы обработки информации (в среде GPSS World, на языке высокого уровня Borland Delphi, а также с использованием программного пакета Simulink (Stateflow) среды MatLab).

По полученным в результате экспериментов данным можно сделать вывод, что модели работаю одинаково, а результаты их моделирования схожи между собой.

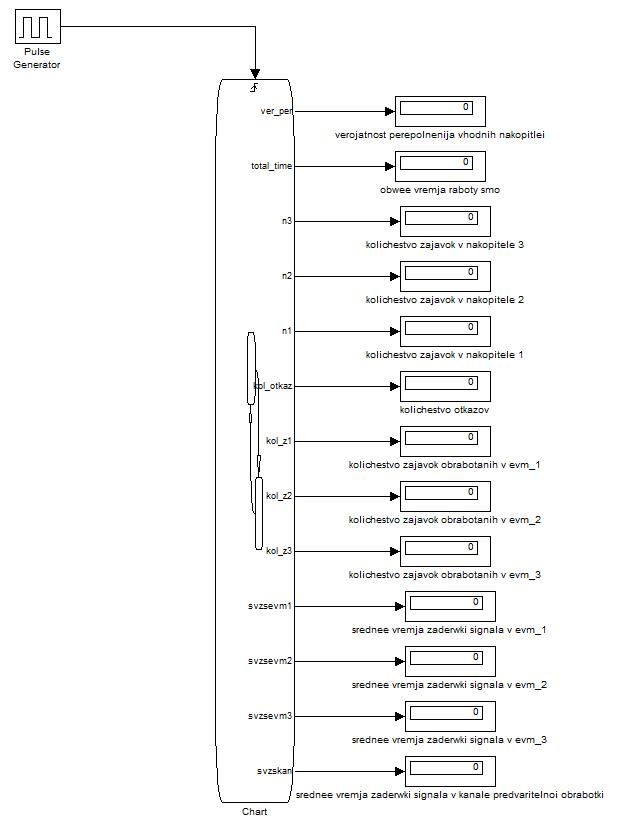
Как вывод по моделированию можно заключить, что вероятность отказа в системе равна нулю, что показали все спроектированные модели. Загрузка каналов при этом стремится к единице, что свидетельствует о рациональном использовании аппаратных средств данной системы обработки информации. Недостатком системы можно считать постепенное увеличение времени пребывания заявок в системе. Оно изменяется от 40 до 370 тактовых единиц времени и в среднем равно около 266.

Также было проведено планирование эксперимента и осуществлен анализ полученных данных. Выявлена зависимость некоторых характеристик от заданных факторов.

Основательно закреплены теоретические знания по дисциплине «Моделирование сложных систем».

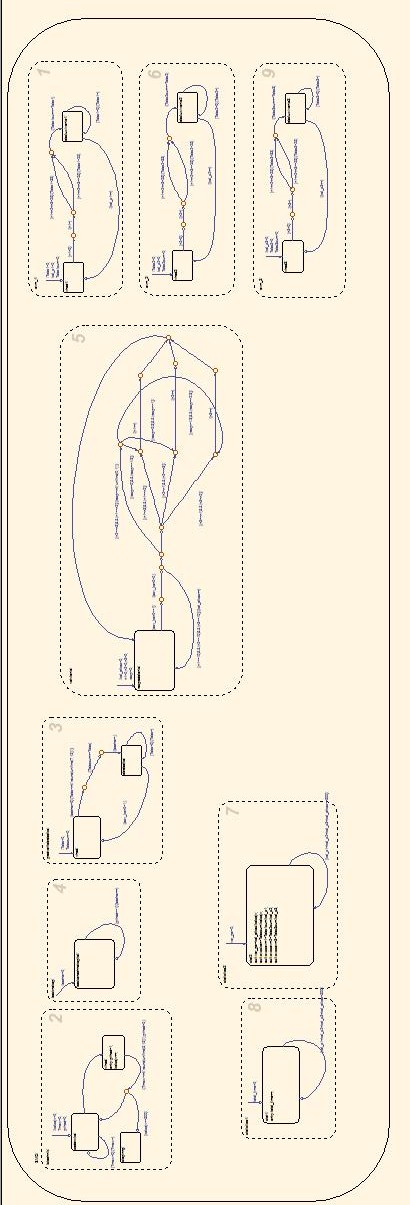
ПРИЛОЖЕНИЕ А

**Общий вид Stateflow- модели**

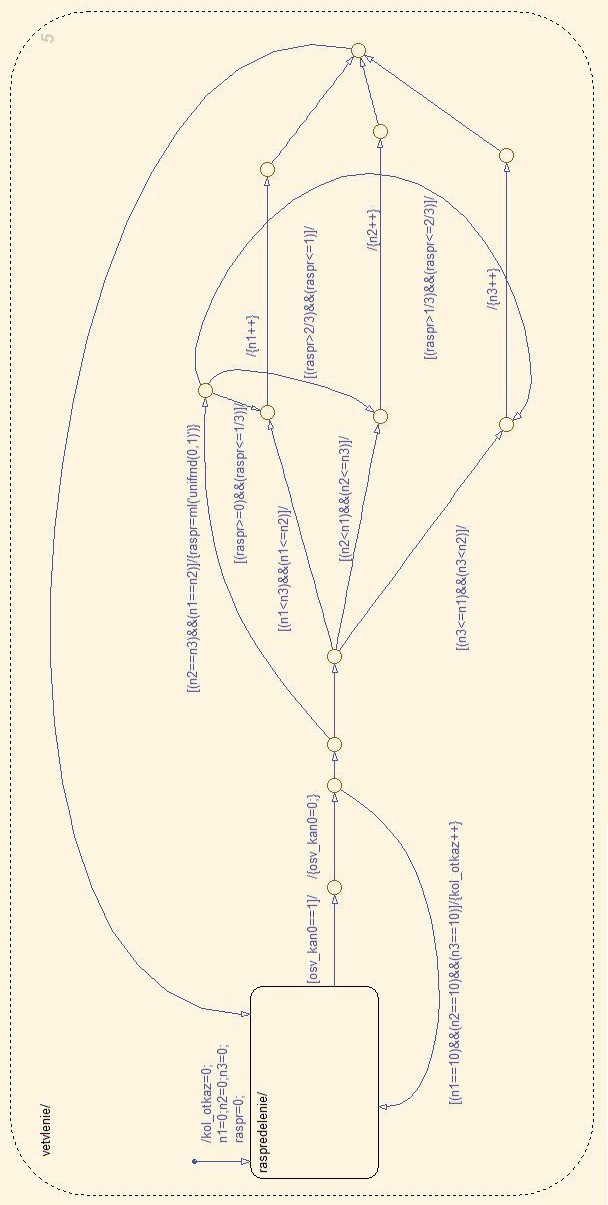
****

ПРИЛОЖЕНИЕ Б

**Внутренняя структура Stateflow- модели**

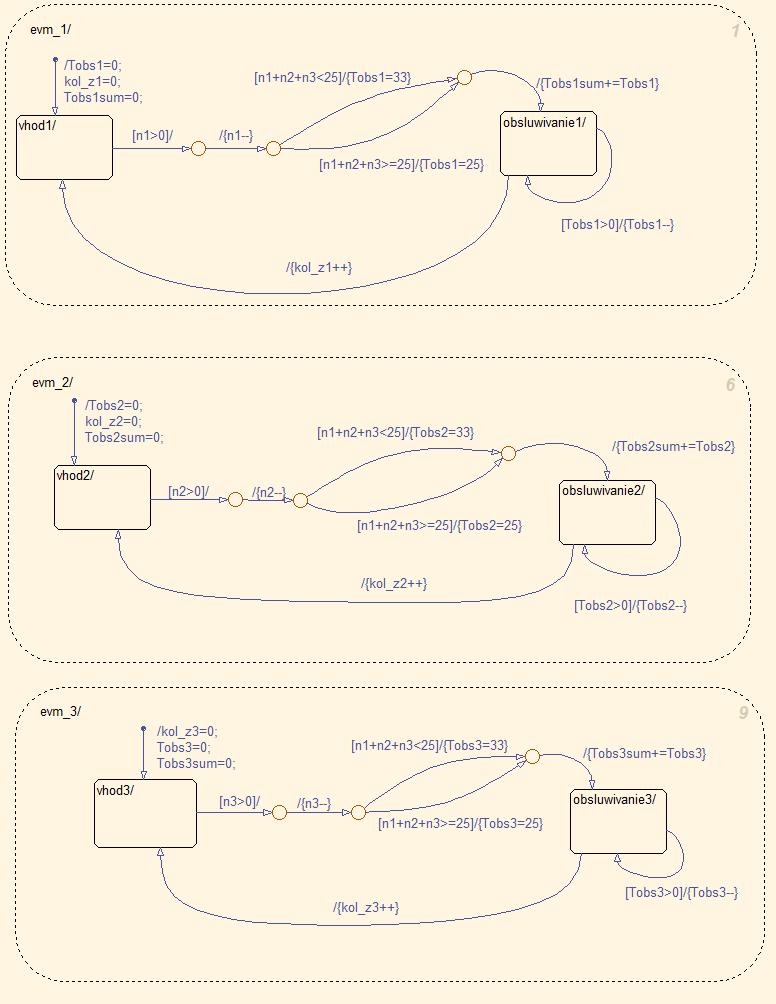
****

ПРИЛОЖЕНИЕ В

**Блок распределения заявок по трем накопителям представлен**

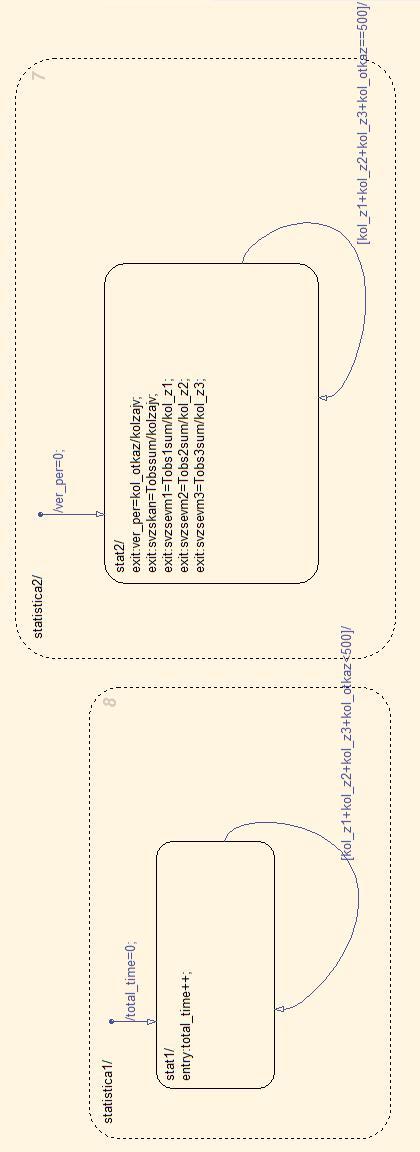
ПРИЛОЖЕНИЕ Г

**Мини-ЭВМ 1,2,3 обработки заявок**

****

ПРИЛОЖЕНИЕ Д

**Блоки статистических данных**

****

ПРИЛОЖЕНИЕ Е

**Текст программы GPSSW**

och1 STORAGE 10 //объявление очереди емкостью 10

och2 STORAGE 10 //объявление очереди емкостью 10

och3 STORAGE 10 //объявление очереди емкостью 10

aa VARIABLE S$och1 //переменная – длина очереди 1

ba VARIABLE S$och2 //переменная – длина очереди 2

caa VARIABLE S$och3 //переменная – длина очереди 3

daa VARIABLE S$och3+S$och2+S$och1 //переменная – общая длина очередей

GENERATE 10,5 // генерация заявок

QUEUE vrem // вход в очередь сбора статистики

QUEUE och // вход в бесконечную очередь канала предварительной обработки

SEIZE KAN1 //занятие канала

DEPART och //освобождение очереди перед каналом

ADVANCE 10,3 //задержка в канале

RELEASE KAN1 // освобождение канала

TEST L V$daa,30,metk1 // проверка - суммарная очередь меньше 30

mt3 TEST LE V$aa,V$ba,mt1 //проверка – поиск самой короткой очереди

TEST LE V$aa,V$caa,mt1 //проверка – поиск самой короткой очереди

ENTER och1 //вход в очередь 1

SEIZE KAN2 // занятие канала ЭВМ 1

LEAVE och1 // освобождение очереди 1

TEST L V$daa,25,metk45 //проверка суммарной длины очередей

ADVANCE 33,0 //задержка в канале

TRANSFER ,met25 //безусловный переход

metk45 ADVANCE 25,0 //задержка в канале

met25 RELEASE KAN2 //освобождение канала ЭВМ 1

TRANSFER ,met2 //безусловный переход

mt1 TEST LE V$ba,V$aa,mt2 //проверка – поиск самой короткой очереди

TEST LE V$ba,V$caa,mt2 //проверка – поиск самой короткой очереди

ENTER och2 //вход в очередь 2

SEIZE KAN3 // занятие канала ЭВМ 2

LEAVE och2 // освобождение очереди 2

TEST L V$daa,25,metk2 //проверка суммарной длины очередей

ADVANCE 33,0 //задержка в канале

TRANSFER ,met20 //безусловный переход

metk2 ADVANCE 25,0 //задержка в канале

met20 RELEASE KAN3 //освобождение канала ЭВМ 2

TRANSFER ,met2 //безусловный переход

mt2 TEST LE V$caa,V$ba,mt3 //проверка – поиск самой короткой очереди

TEST LE V$caa,V$aa,mt3 //проверка – поиск самой короткой очереди

ENTER och3 //вход в очередь 3

SEIZE KAN4 // занятие канала ЭВМ 3

LEAVE och3 // освобождение очереди 3

TEST L V$daa,25,metk36 //проверка суммарной длины очередей

ADVANCE 33,0 //задержка в канале

TRANSFER ,met29 //безусловный переход

metk36 ADVANCE 25,0 //задержка в канале

met29 RELEASE KAN4 //освобождение канала ЭВМ 3

met2 DEPART vrem //освобождение очереди сбора статистики

TERMINATE 1 //вывод заявки из системы

metk1 QUEUE otk //занятие бесконечной очереди отказов

DEPART otk //освобождение бесконечной очереди отказов

TERMINATE 1 //вывод отказа из системы

START 500 //моделировать 500 раз

ПРИЛОЖЕНИЕ Ж

**Отчет моделирования в GPSSW**

START TIME END TIME BLOCKS FACILITIES STORAGES

0.000 5376.281 45 4 3

NAME VALUE

AA 10003.000

BA 10004.000

CAA 10005.000

DAA 10006.000

KAN1 10009.000

KAN2 10010.000

KAN3 10011.000

KAN4 10012.000

MET2 41.000

MET20 29.000

MET25 18.000

MET29 40.000

METK1 43.000

METK2 28.000

METK36 39.000

METK45 17.000

MT1 20.000

MT2 31.000

MT3 9.000

OCH 10008.000

OCH1 10000.000

OCH2 10001.000

OCH3 10002.000

OTK UNSPECIFIED

VREM 10007.000

LABEL LOC BLOCK TYPE ENTRY COUNT CURRENT COUNT RETRY

1 GENERATE 537 0 0

2 QUEUE 537 0 0

3 QUEUE 537 8 0

4 SEIZE 529 0 0

5 DEPART 529 0 0

6 ADVANCE 529 1 0

7 RELEASE 528 0 0

8 TEST 528 0 0

MT3 9 TEST 528 0 0

10 TEST 299 0 0

11 ENTER 177 9 0

12 SEIZE 168 0 0

13 LEAVE 168 0 0

14 TEST 168 0 0

15 ADVANCE 145 1 0

16 TRANSFER 144 0 0

METK45 17 ADVANCE 23 0 0

MET25 18 RELEASE 167 0 0

19 TRANSFER 167 0 0

MT1 20 TEST 351 0 0

ПРИЛОЖЕНИЕ Ж

(продолжение)

21 TEST 317 0 0

22 ENTER 177 9 0

23 SEIZE 168 0 0

24 LEAVE 168 0 0

25 TEST 168 0 0

26 ADVANCE 143 0 0

27 TRANSFER 143 0 0

METK2 28 ADVANCE 25 1 0

MET20 29 RELEASE 167 0 0

30 TRANSFER 167 0 0

MT2 31 TEST 174 0 0

32 TEST 174 0 0

33 ENTER 174 7 0

34 SEIZE 167 1 0

35 LEAVE 166 0 0

36 TEST 166 0 0

37 ADVANCE 142 0 0

38 TRANSFER 142 0 0

METK36 39 ADVANCE 24 0 0

MET29 40 RELEASE 166 0 0

MET2 41 DEPART 500 0 0

42 TERMINATE 500 0 0

METK1 43 QUEUE 0 0 0

44 DEPART 0 0 0

ОТКАЗЫ 45 TERMINATE 0 0 0

FACILITY ENTRIES UTIL. AVE. TIME AVAIL. OWNER PEND INTER RETRY DELAY

KAN1 529 0.985 10.015 1 529 0 0 0 8

KAN2 168 0.995 31.849 1 500 0 0 0 9

KAN3 168 0.991 31.713 1 501 0 0 0 9

KAN4 167 0.983 31.653 1 504 0 0 0 7

QUEUE MAX CONT. ENTRY ENTRY(0) AVE.CONT. AVE.TIME AVE.(-0) RETRY

VREM(среднее время) 41 37 537 0 26.652 266.835 266.835 0

OCH 12 8 537 26 5.156 51.621 54.247 0

STORAGE CAP. REM. MIN. MAX. ENTRIES AVL. AVE.C. UTIL. RETRY DELAY

OCH1 10 1 0 9 177 1 6.224 0.622 0 0

OCH2 10 1 0 9 177 1 5.853 0.585 0 0

OCH3 10 2 0 8 174 1 5.464 0.546 0 0

CEC XN PRI M1 ASSEM CURRENT NEXT PARAMETER VALUE

504 0 5057.917 504 34 35

FEC XN PRI BDT ASSEM CURRENT NEXT PARAMETER VALUE

529 0 5379.984 529 6 7

538 0 5381.388 538 0 1

500 0 5385.710 500 15 16

501 0 5392.528 501 28 29

ПРИЛОЖЕНИЕ З

**Алгоритм работы программы в Delphi**

**IMAGE0017.TIF**

ПРИЛОЖЕНИЕ И

**Текст программы в Delphi**

unit Unit1;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, jpeg, ExtCtrls, Grids, Math;

type

TForm1 = class(TForm)

StringGrid1: TStringGrid;

Image1: TImage;

Label1: TLabel;

Button1: TButton;

LabeledEdit2: TLabeledEdit;

LabeledEdit1: TLabeledEdit;

StringGrid2: TStringGrid;

procedure Button1Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form1: TForm1;

implementation

{$R \*.dfm}

procedure TForm1.Button1Click(Sender: TObject);

var

Tpos,Tobr1,Tosv1,T,L1,N,K1,K2,Tobr,Tosv2,Otk:integer;

K3,K4,L2,L3,L4,Tosv3,Tosv4:integer;

j,S,m,u,x,sum:integer;

ПРИЛОЖЕНИЕ И

(продолжение)

V:real;

label met;

begin

randomize;

StringGrid1.Cells[0,0]:='Òàêò';

StringGrid1.Cells[1,0]:='Ò ïîñò';

StringGrid1.Cells[2,0]:='L î÷åð1';

StringGrid1.Cells[3,0]:='Êàíàë1';

StringGrid1.Cells[4,0]:='Ò îáð1';

StringGrid1.Cells[5,0]:='Ò îñâ1';

StringGrid1.Cells[6,0]:='L î÷åð2';

StringGrid1.Cells[7,0]:='Êàíàë2';

StringGrid1.Cells[8,0]:='Ò îñâ2';

StringGrid1.Cells[9,0]:='L î÷åð3';

StringGrid1.Cells[10,0]:='Êàíàë3';

StringGrid1.Cells[11,0]:='Ò îñâ3';

StringGrid1.Cells[12,0]:='L î÷åð4';

StringGrid1.Cells[13,0]:='Êàíàë4';

StringGrid1.Cells[14,0]:='Ò îñâ4';

StringGrid1.Cells[15,0]:='Ò îáð';

StringGrid1.Cells[16,0]:='N';

StringGrid1.Cells[17,0]:='Îòêàç';

Tpos:=0;

K1:=0;

K2:=0;

K3:=0;

K4:=0;

T:=0;

L1:=0;

ПРИЛОЖЕНИЕ И

(продолжение)

L2:=0;

L3:=0;

L4:=0;

N:=0;

j:=1;

Otk:=0;

Tosv2:=0;

Tobr:=33;

Tosv1:=0;

Tobr1:=0;

Tosv3:=0;

Tosv4:=0;

S:=0;

m:=1;

u:=0;

StringGrid1.RowCount:=1;

StringGrid2.Cells[0,0]:=IntToStr(0);

while N+Otk<500 do

begin

StringGrid1.RowCount:=StringGrid1.RowCount+1;

if (T=Tosv2)and (K2=1) then

begin

K2:=0;

N:=N+1;

StringGrid2.Cells[u,1]:=IntToStr(T);

u:=u+1;

if L2>0 then

begin

K2:=1;

ПРИЛОЖЕНИЕ И

(продолжение)

L2:=L2-1;

if L2+L3+L4>=25 then

Tobr:=25

else Tobr:=33;

Tosv2:=T+Tobr;

end; end;

if (T=Tosv3)and (K3=1) then

begin

K3:=0;

N:=N+1;

StringGrid2.Cells[u,1]:=IntToStr(T);

u:=u+1;

if L3>0 then

begin

K3:=1;

L3:=L3-1;

if L2+L3+L4>=25 then

Tobr:=25

else Tobr:=33;

Tosv3:=T+Tobr;

end;

end;

if (T=Tosv4)and (K4=1) then

begin

K4:=0;

N:=N+1;

StringGrid2.Cells[u,1]:=IntToStr(T);

ПРИЛОЖЕНИЕ И

(продолжение)

u:=u+1;

if L4>0 then

begin

K4:=1;

L4:=L4-1;

if L2+L3+L4>=25 then

Tobr:=25

else Tobr:=33;

Tosv4:=T+Tobr;

end;

end;

if (T=Tosv1)and (K1=1) then

begin

K1:=0;

if (L2=10) and (L3=10) and (L4=10) then

begin

Otk:=Otk+1;

StringGrid2.Cells[u,1]:=IntToStr(T);

u:=u+1;

goto met;

end;

if (L2<=L3) and (L2<=L4) then

begin

L2:=L2+1;

if K2=0 then

begin

K2:=1;

L2:=L2-1;

if L2+L3+L4>=25 then

ПРИЛОЖЕНИЕ И

(продолжение)

Tobr:=25

else Tobr:=33;

Tosv2:=T+Tobr

end;

goto met

end;

if (L3<=L2) and (L3<=L4) then

begin

L3:=L3+1;

if K3=0 then

begin

K3:=1;

L3:=L3-1;

if L2+L3+L4>=25 then

Tobr:=25

else Tobr:=33;

Tosv3:=T+Tobr;

end;

goto met

end;

if (L4<=L2) and (L4<=L3) then

begin

L4:=L4+1;

if K4=0 then

begin

K4:=1;

L4:=L4-1;

if L2+L3+L4>=25 then

Tobr:=25

else Tobr:=33;

ПРИЛОЖЕНИЕ И

(продолжение)

Tosv4:=T+Tobr;

end;

goto met

end;

met: if L1>0 then

begin

K1:=1;

L1:=L1-1;

Tobr1:=random(7)+7;

Tosv1:=T+Tobr1;

end;

end;

if T=Tpos then

begin

L1:=L1+1;

if K1=0 then

begin

K1:=1;

L1:=L1-1;

Tobr1:=random(7)+7;

Tosv1:=T+Tobr1

end;

S:=random(11)+5;

Tpos:=T+S;

if m<500 then

StringGrid2.Cells[m,0]:=IntToStr(Tpos);

m:=m+1;

end;

ПРИЛОЖЕНИЕ И

(продолжение)

StringGrid1.Cells[0,j]:=IntToStr(T);

StringGrid1.Cells[1,j]:=IntToStr(Tpos-S)+'+'+IntToStr(S);

StringGrid1.Cells[2,j]:=IntToStr(L1);

StringGrid1.Cells[3,j]:=IntToStr(K1);

StringGrid1.Cells[4,j]:=IntToStr(Tobr1);

StringGrid1.Cells[5,j]:=IntToStr(Tosv1);

StringGrid1.Cells[6,j]:=IntToStr(L2);

StringGrid1.Cells[7,j]:=IntToStr(K2);

StringGrid1.Cells[8,j]:=IntToStr(Tosv2);

StringGrid1.Cells[9,j]:=IntToStr(L3);

StringGrid1.Cells[10,j]:=IntToStr(K3);

StringGrid1.Cells[11,j]:=IntToStr(Tosv3);

StringGrid1.Cells[12,j]:=IntToStr(L4);

StringGrid1.Cells[13,j]:=IntToStr(K4);

StringGrid1.Cells[14,j]:=IntToStr(Tosv4);

StringGrid1.Cells[15,j]:=IntToStr(Tobr);

StringGrid1.Cells[16,j]:=IntToStr(N);

StringGrid1.Cells[17,j]:=IntToStr(Otk);

j:=j+1;

T:=Tpos;

if (Tosv1<T) and (K1=1) then

T:=Tosv1;

if (Tosv2<T) and (K2=1) then

T:=Tosv2;

if (Tosv3<T) and (K3=1) then

T:=Tosv3;

if (Tosv4<T) and (K4=1) then

T:=Tosv4;

ПРИЛОЖЕНИЕ И

(продолжение)

end;

V:=Otk/500;

LabeledEdit1.Text:=FloatToStr(V);

sum:=0;

for x:=0 to 499 do

begin

StringGrid2.Cells[x,2]:=IntToStr(StrToInt(StringGrid2.Cells[x,1])-StrToInt(StringGrid2.Cells[x,0]));

sum:=sum+StrToInt(StringGrid2.Cells[x,2]);

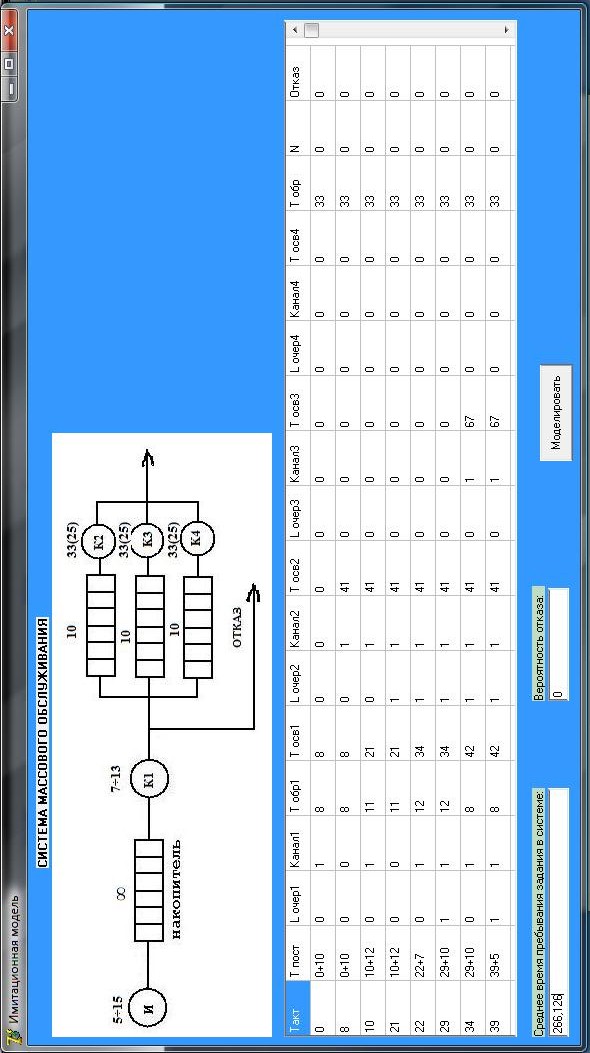
end;

LabeledEdit2.Text:=FloatToStr(sum/500);

end; end.

ПРИЛОЖЕНИЕ К

**Интерфейс программы**

****

**СПИСОК ЛИТЕРАТУРЫ**

1. Сапожников Н.Е. Математическое моделирование на ПК: Учеб. пособие. 2-е изд., перераб. и доп. – Севастополь: СНУЯЭиП, 2006. – 380 с.: ил.
2. Максимей И.В. Имитационное моделирование на ЭВМ. – М.: Радио и связь, 1988. – 232 с.: ил.
3. Сапожников Н.Е. Компьютерное моделирование в энергетике: Учеб. Пособие. – Севастополь: СИЯЭиП, 2001. – 284 с.: ил.
4. Фленов М.Е. Библия Delphi. – СПб.: БХВ-Петербург, 2004г.-880.: ил.
5. Л. А. Воробейчиков, Г. К. Сосновиков Основы моделирования на GPSS/PC.