

Содержание

Общие организационно-методические указания по подготовке и проведению лабораторных работ.....	4
Лабораторная работа 1	
Моделирование с использованием пакета визуального блочного имитационного моделирования Simulink.....	5
Лабораторная работа 2	
Моделирование с использованием системы имитационного моделирования AnyLogic.....	18

Общие организационно-методические указания по подготовке и проведению лабораторных работ

При подготовке к лабораторной работе студенты **должны:**

- уяснить цель и порядок проведения лабораторной работы;
- изучить учебные материалы, изложенные в рекомендуемой к лабораторной работе литературе.

В результате самостоятельной подготовки студенты должны уметь ответить на вопросы. На занятии каждый студент должен иметь конспект лекций, данные организационно-методических указаний и отдельную тетрадь для оформления отчетов по лабораторным работам.

Лабораторная работа начинается с контроля готовности студентов к занятию, который, в свою очередь, начинается с проверки присутствия студентов на занятии, наличия у каждого студента организационно-методических указаний, тетради для оформления отчетов по лабораторным работам, конспекта лекций и заканчивается контрольным опросом студентов по знанию основных теоретических положений занятия.

Затем студенты под руководством преподавателя осваивают особенности использования изучаемой на занятии системы (пакета) имитационного моделирования. Далее самостоятельно в соответствии с индивидуальным заданием формируют имитационную модель заданной системы и осуществляют ее исследование.

Сформированная модель и результаты исследований оформляются студентами в отчетах.

Заканчивается занятие подведением итогов с оценкой работы студентов, оформлением и защитой отчета лабораторной работы.

Лабораторная работа №1

Моделирование с использованием пакета визуального блочного имитационного моделирования Simulink

Цель занятия – закрепление теоретических знаний по моделированию систем и практическое освоение пакета визуального блочного имитационного моделирования Simulink.


Время - 6 часов.

1. Основные теоретические положения лабораторной работы

Пакет визуального блочного имитационного моделирования Simulink является пакетом расширения системы MATLAB. Simulink предназначен для моделирования линейных и нелинейных динамических систем и устройств, представленных своей функциональной схемой, именуемой S-моделью или просто моделью. Пакет Simulink обеспечивает различные варианты моделирования: во временной области, в частотной области, с событийным управлением, на основе спектральных преобразований Фурье, с использованием метода Монте-Карло и т.д. [1].

Для построения функциональной блок-схемы моделируемых систем и устройств Simulink имеет обширную библиотеку блочных компонентов и удобный редактор блок-схем. Он основан на графическом интерфейсе пользователя и по существу является типичным средством визуально-ориентированного программирования. Используя наборы блоков, пользователь с помощью мыши переносит блоки на рабочий стол пакета Simulink и соединяет линиями входы и выходы блоков. Таким образом создается блок-схема (диаграмма) моделируемой системы или устройства, то есть модель. Файлы моделей, сформированных в пакете Simulink, имеют расширение *.mdl.

Ценность пакета Simulink заключается в обширной, открытой для изучения и модификации библиотеке компонентов (блоков). Она включает в себя источники воздействий (сигналов) с практически любыми временными зависимостями, масштабирующие, линейные и нелинейные преобразователи с разнообразными формами передаточных характеристик, квантующее устройство, интегрирующие и дифференцирующие блоки и т.д. В библиотеке имеет набор виртуальных регистрирующих устройств – от простых измерителей (вольтметр, амперметр) до универсальных осциллографов, графопостроителей, анализаторов спектра и т.д. К достоинству пакета Simulink можно отнести и возможность задания в блоках произвольных математических выражений.

Запуск пакета Simulink обычно осуществляется с помощью кнопки , расположенной на панели инструментов системы MATLAB (см. рис.1).

Пакет Simulink можно запустить, выполнив команду `simulink` в командной строке системы MATLAB.

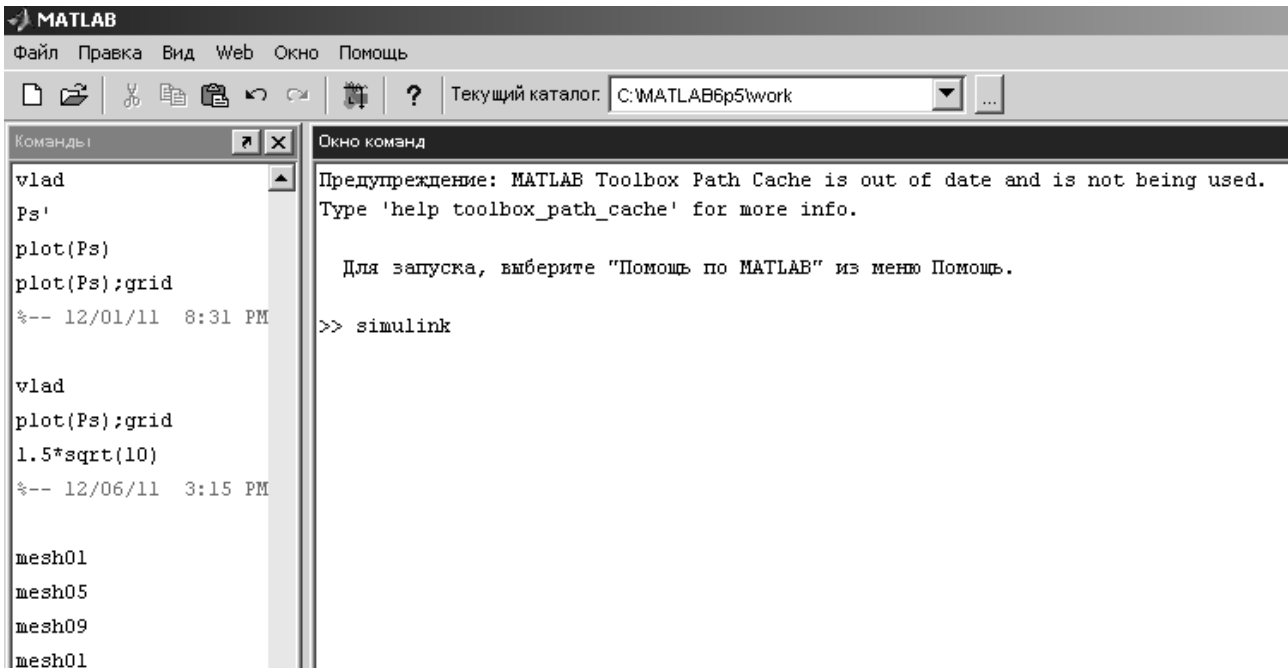


Рис. 1. Фрагмент главного окна системы MATLAB

При выполнении вышеуказанных действий открывается окно интегрированного браузера библиотек (см. рис.2).

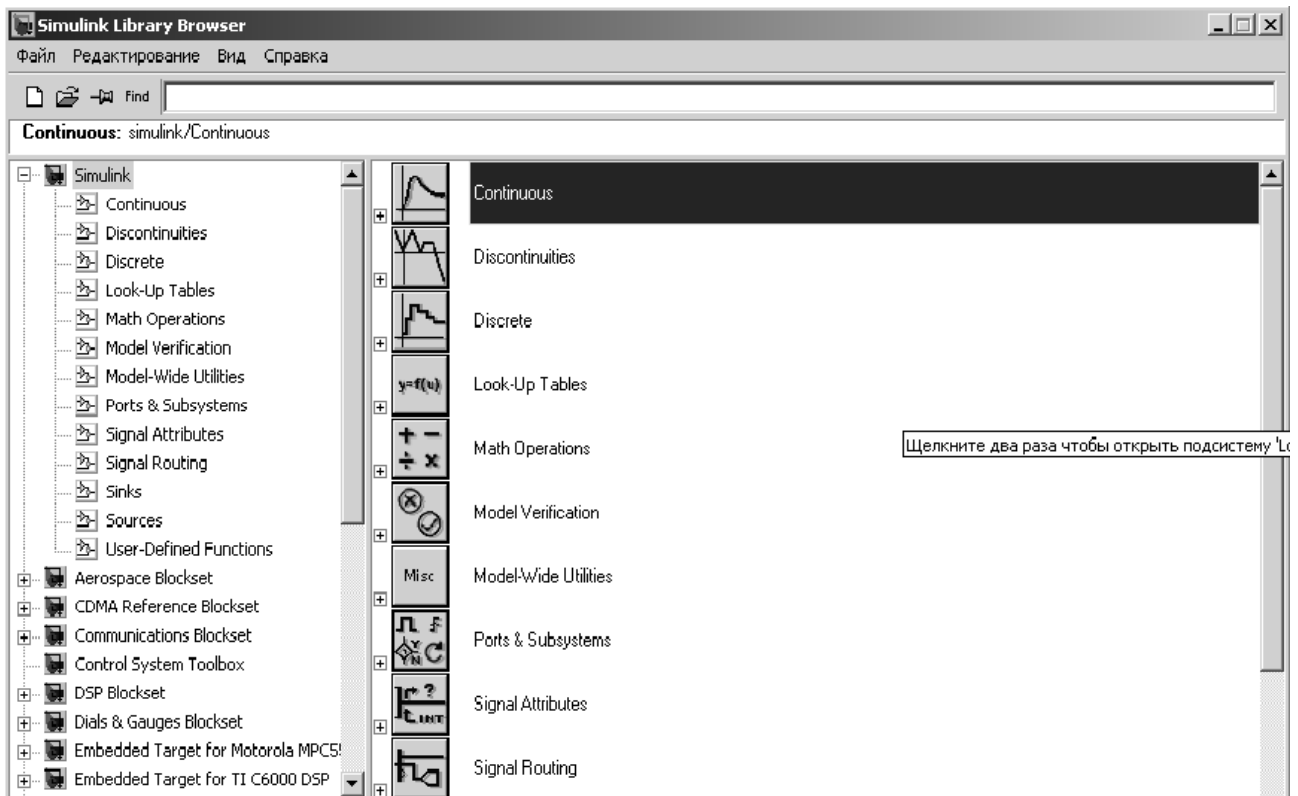


Рис. 2. Окно браузера библиотек Simulink

В окне браузера содержится дерево компонентов библиотек Simulink. Для создания S-модели открывается пустое окно, соответствующей кнопкой в панели инструментов браузера библиотек Simulink (или командой **File→New→Model** в меню). Меню Simulink содержит следующие позиции (см. рис. 3):

File – работа с файлами моделей и библиотек (создание, считывание, сохранение, печать);

Edit – операции редактирования, работа с буфером обмена и создание подсистем;

View – управление отображением панели инструментов и строки состояния;

Simulation – управление процессом моделирования (старт, пауза, вывод окна настройки параметров моделирования);

Format – операции форматирования модели;

Tools – управление видом анализа.

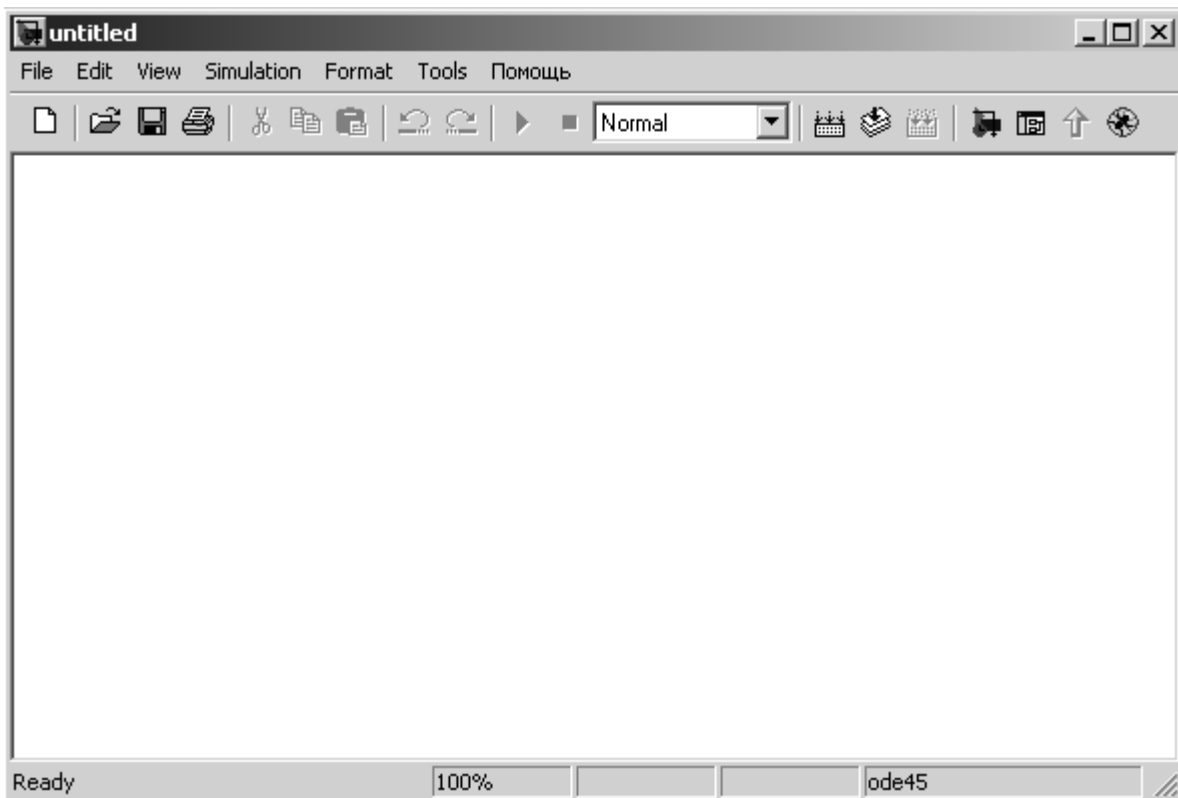



Рис. 3. Окно модели

Вся S-модель строится из блоков, имеющих входы и выходы. Существует библиотека стандартных блоков, кроме того, можно создавать свои собственные блоки любой сложности. Существует две основные группы специальных устройств (блоков) – источники сигналов (**Sources**) и устройства вывода (**Sinks**). Блоки имеют названия. Для изменения названия надо щелкнуть

по нему левой кнопкой мыши и отредактировать текст. Каждый блок имеет свои настраиваемые свойства. Для их изменения надо дважды щелкнуть на блоке мышью и изменить нужные значения в диалоговом окне. Для того чтобы повернуть блок на 90 градусов, надо выделить его и нажать клавиши **Ctrl+R**. Комбинация **Ctrl+I** позволяет выполнить зеркальное отражение входов и выходов. Верхнее меню **Format** предназначено для изменения оформления выделенного блока. Для выделенного блока можно изменить цвет текста и линий (**Foreground color**), цвет фона (**Background color**), вывести тень (**Show drop shadow**), переместить название на другую сторону (**Flip name**). Для выделения одного блока или соединительной линии надо щелкнуть левой кнопкой по нужному элементу. Для выделения нескольких блоков надо «обвести» их при нажатой левой кнопке мыши. Клавиша **Delete** удаляет выделенную часть. Чтобы скопировать блок (или выделенную часть), надо перетащить его при нажатой правой кнопке мыши.

Блоки соединяются линиями связи, по которым распространяются сигналы. Для соединения блоков надо щелкнуть левой кнопкой мыши по источнику сигнала и затем, при нажатой клавише **Ctrl**, по блоку-приемнику. Можно также протянуть мышкой линию связи между нужными выходом и входом. Чтобы подать один сигнал на два блока (сделать «развилку»), надо сначала создать одну линию обычным способом. Чтобы провести вторую линию, следует нажать правую кнопку мыши на линии в точке развилки и протащить линию ко второму блоку.

Модель можно скопировать в буфер обмена в виде растрового рисунка. Для этого в окне модели надо выбрать в верхнем меню пункт **Edit – Copy model to clipboard**. Предварительно лучше уменьшить размеры окна до минимальных, чтобы не было белых полей.

Для запуска моделирования надо щелкнуть левой кнопкой мыши по кнопке  на панели инструментов. Эта же кнопка позволяет остановить моделирование при необходимости.

Параметры моделирования (метод интегрирования, обработка ошибок) устанавливаются с помощью окна **Simulation – Parameters**. Самые важные параметры – это время моделирования (**Stop time**) и метод численного интегрирования уравнений (**Solver options**).

Основные источники сигналов (**Sources**)



Constant – сигнал постоянной величины.



Step – ступенчатый сигнал, меняется время скачка (**Step Time**), начальное (**Initial Value**) и конечное значение (**Final Value**).



Ramp – линейно возрастающий сигнал с заданным наклоном (**Slope**). Можно задать также время начала изменения сигнала (**Start Time**) и начальное значение (**Initial Value**).



Pulse Generator – генератор прямоугольных импульсов, задаются амплитуда (*Amplitude*), период (*Period*), ширина (*Pulse Width*, в процентах от периода), фаза (*Phase Delay*).



Repeating Sequence – последовательность импульсов, их форма задается в виде пар чисел (время; величина сигнала).



Sine Wave – синусоидальный сигнал, задается амплитуда (*Amplitude*), частота (*Frequency*), фаза (*Phase*) и среднее значение (*Bias*).



Signal Builder – построитель сигналов, позволяющий задавать форму сигнала, перетаскивая мышью опорные точки.



Random Number – случайные числа с нормальным (гауссовым) распределением. Можно задать среднее значение (*Mean Value*), дисперсию (*Variance*), период изменения сигнала (*Sample Time*).

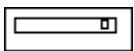


Uniform Random Number – случайные числа с равномерным распределением в заданном интервале от *Minimum* до *Maximum*.



Band Limited White Noise – случайный сигнал, ограниченный по полосе белый шум (имеющий равномерный спектр до некоторой частоты). Блок используется как источник белого шума для моделей непрерывных систем. Задается интенсивность (*Noise Power*) и интервал дискретизации (*Sample Time*), в течение которого удерживается постоянное значение сигнала. Чем меньше интервал, тем точнее моделирование, однако больше вычислительные затраты.

Основные устройства вывода (*Sinks*)

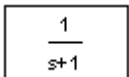


Display – цифровой дисплей; показывает изменение входного сигнала в цифровом виде.

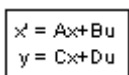


Scope – осциллограф; показывает изменение сигнала в виде графика, позволяет передавать данные в рабочую область MATLAB для последующей обработки и оформления.

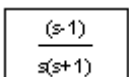
Линейные системы (*Continuous*)



Transfer Fcn – передаточная функция, в параметрах задаются числитель (*Numerator*) и знаменатель (*Denominator*) в виде полиномов.



State Space – модель в пространстве состояний, в параметрах задается четверка матриц, определяющих модель, и начальные условия для вектора состояния (*Initial conditions*).



Zero-Pole – модель в форме «нули-полюса», в параметрах задаются массивы нулей (*Zeros*), полюсов (*Poles*), а также коэффициент усиления (*Gain*).



Integrator – интегратор с возможностью установки начальных условий (**Initial condition**), а также пределов насыщения (**Lower saturation limit** и **Upper saturation limit**). Когда сигнал выхода выходит за границы, определяемые этими пределами, интегрирование прекращается.

Нелинейные звенья (*Discontinuities*)



Saturation – насыщение, в параметрах задаются верхний и нижний пределы (**Upper limit** и **Lower limit**).



Dead zone – нечувствительность, «мертвая зона». В параметрах задаются пределы нечувствительности (**Start of dead zone** и **End of dead zone**).



Rate Limiter – ограничитель скорости изменения сигнала, в параметрах задаются пределы на скорость увеличения (**Rising slew rate**) и на скорость уменьшения (**Falling slew rate**).



Relay – реле, в параметрах задаются точки переключения (**Switch on point** и **Switch off point**), а также величины сигналов в режимах «включено» (**Output when on**) и «выключено» (**Output when off**).



Backlash – люфт, «мертвый ход». В параметрах задаются величина мертвого хода (**Deadband width**) и начальное значение выхода (**Initial output**).



Coulomb and Viscous Friction – кулоновское и вязкое трение.

Другие часто используемые блоки

Math Operations



Gain – усилитель; задается коэффициент усиления (**Gain**).



Sum – сумматор; используется для сложения и вычитания входов. Параметр **List of signs** задает количество входов, их знаки («+» для сложения и «-» для вычитания). Промежутки между входами (обозначаются знаком |).



Trigonometric Function – тригонометрическая функция.

Signal Routing



Manual Switch – ручной переключатель; позволяет двойным щелчком переключать выход на один из двух входных сигналов.



Mux – мультиплексор; объединяет несколько сигналов в один «жгут» (векторный сигнал), в параметрах задается число входов (**Number of Inputs**).



Demux – демультимплексор; позволяет «разбить» векторный сигнал на несколько скалярных, в параметрах задается число выходов (*Number of Outputs*).

Если на схеме много блоков, она становится громоздкой и работать с такой схемой неудобно. Чтобы не перегружать схемы, можно объединять блоки в подсистемы. Проще всего выделить нужные блоки мышкой («обвести» при нажатой левой кнопке мыши) и нажать клавиши **Ctrl+G** (или выбрать пункт меню **Edit – Create subsystem**). На основной схеме подсистема изображается как блок типа **Subsystem** (из группы **Ports and Subsystems**). Этот блок можно добавить и вручную, перетащив из окна **Library Browser**.

С помощью двойного щелчка левой кнопки мыши можно «открыть» подсистему. Входы обозначаются блоками **In**, а выходы – блоками **Out** (также из группы **Ports and Subsystems**). Можно добавлять новые входы и выходы, удалять ненужные, менять названия, работая с ними так же, как с остальными блоками.

Внутри подсистем можно создавать вложенные подсистемы. В подсистеме не может быть блоков с одинаковыми названиями, однако в разных подсистемах – могут быть. Сохранить модель можно из окна любой подсистемы, но закрытие окна подсистемы не приводит к закрытию модели.

Пользователь, всерьез занимающийся моделированием в Simulink, рано или поздно сталкивается с необходимостью подготовки собственных блоков, обладающих свойствами стандартных блоков. Необходимо заметить, что главное отличие подсистем от блоков состоит в том, что подсистемы не имеют своей уникальной пиктограммы и окна параметров и не связаны с разделом библиотеки. Для построения пользовательских блоков Simulink предлагает специальный механизм маскирования подсистем. Маскированные подсистемы – это такие подсистемы, которые имеют специальный признак (маску), скрывающий их внутреннюю, порой достаточно сложную структуру.

Маскированные подсистемы обладают рядом достоинств:

- они имеют свои пиктограммы с уникальным изображением;
- их можно использовать как библиотечные блоки;
- у них есть свое окно установки параметров;
- есть возможность в любой момент сбросить маску и наблюдать структуру блока;
- применение масок расширяет возможности построения сложных моделей;
- имеется возможность легко отредактировать подсистему, превращенную в маскированную;
- повышается наглядность моделей-диаграмм;
- повышается защищенность подсистем от модификации, в том числе преднамеренной.

Для создания маскированных подсистем требуется выполнить ряд действий:

- разработать и протестировать модель с соответствующими блоками;
- выделить часть блоков и оформить их в виде подсистемы;

- задать подсистеме статус маскированной подсистемы;
- с помощью специального редактора масок создать окно установки параметров, определить средства инициализации подсистемы, создать документацию под маскированную подсистему и ее справочную систему;
- выполнить тестирование основной модели с ее маскированными подсистемами и прочими блоками;
- выполнить редактирование созданной маскированной подсистемы;
- демаскировать (если необходимо) подсистему.

Для создания маски достаточно выделить нужную подсистему, установив на ней курсор мыши и щелкнув ее левой кнопкой, после чего выбрать команду **Mask Sysystem...** меню **Edit**. Данная команда запускает редактор маски - появляется его окно с открытой незаполненной вкладкой **Icon**.

Редактор маски имеет четыре вкладки. Отметим их назначение:

Icon - подготовка пиктограммы (значка) блока;

Parameters - подготовка окна параметров блока (появилась в Simulink 5);

Initialization - подготовка средства инициализации;

Documentation - подготовка документации по блоку.

Сверху окна имеется общее для всех вкладок поле **Mask type**, в которое заносится имя блока. Внизу имеются также общие для всех вкладок пять кнопок:

OK - завершить установки и создать маску;

Cancel - отказаться от создания маски и закрыть окно редактора;

Unmask - снять маску с выделенного блока;

Help - открыть окно справки по редактору маски;

Apply - применить заданные установки, не выходя из редактора маски.

Важнейшим для маскируемой подсистемы является создание окна параметров. Для этого служит вкладка **Parameters**.

При первом открытии вкладка **Parameters** содержит пустой список **Dialog parameters**. Этот список состоит из пяти столбцов:

Prompt - имена используемых в маске блоков;

Type – типы параметров (выбираются из раскрывающегося списка: **Edit** – редактируемое поле ввода; **Checkbox** – поле для флажка; **Popup** – раскрывающийся список);

Variable – имена переменных, несущих значение соответствующих параметров;

Evaluate – признак исполнения;

Tunable – признак возможности изменения.

Для создания этого списка служат четыре кнопки вкладки: **Add**, **Delete**, **Up**, **Down**.

Модели, не содержащие текстовых комментариев, не наглядны. Зачастую без комментариев даже трудно понять, что, собственно говоря, моделируется. Поэтому подготовка текстовых комментариев - важный момент в культуре моделирования, кстати, как в программировании.

Для создания надписи в окне модели требуется указать мышью место надписи и дважды щелкнуть левой кнопкой мыши. Появится блок надписи с курсором. Введенный текст можно отредактировать, для чего его требуется выделить и, установив курс, нажать правую кнопку мыши. В появившемся контекстном меню выбрать нужную команду.

Пакет Simulink позволяет осуществлять форматирование модели. Для этого требуется воспользоваться меню форматирования Format.

В меню Format (и в контекстном меню) находится ряд команд форматирования блоков. Их можно разделить на несколько характерных групп.

Управление отображением надписей и видом блоков:

Font - установка шрифта для текстовых надписей;

Textalignment - выравнивание текста в текстовом блоке;

Flipname - помещение подписи блока сверху или снизу блока;

Show/Hidename - отображение или скрытие подписи выделенного блока;

Flipblock - отражение блока относительно вертикальной оси;

Rotateblock - вращение блока на 90°;

Showdropshadow - показ тени от блока.

Показ меток портов:

Showportlabels - показ меток портов.

Установка цветов:

ForegroundColor - установка цвета линий выделенных блоков;

BackgroundColor - установка цвета фона для выделенных блоков;

Screencolor - установка цвета фона для всего окна модели.

Прочие установки:

Librarylinkdisplay - отображение связей с библиотеками;

Simplytimecolors - установка цвета блока индикации времени;

Widenonscalarlines - увеличение/уменьшение ширины линий;

Signaldimensions - отображение размерности сигналов;

Portdatatypes - вывод данных о типе портов;

Executionorder - вывод порядкового номера блока в последовательности исполнения.

Полный перечень возможностей пакета Simulink приведен в [1,3,5].

2. Порядок выполнения работы

2.1. На этапе подготовки к лабораторной работе студенты должны, используя литературу [1-5], конспект лекций и учебные материалы данного пособия, углубить свои знания по:

- библиотекам пакета Simulink;

- методике подготовки диаграмм моделей, их редактированию и настройке.

2.2. Преподаватель перед проведением занятия проводит контрольный опрос студентов и определяет степень их готовности к лабораторной работе. Затем преподаватель разбивает группу студентов из расчета не более 10 подгрупп.

Каждая подгруппа получает от преподавателя индивидуальный вариант задания на лабораторную работу.

Лабораторная работа состоит из двух частей. В первой части работы студенты, используя пакет Simulink, в соответствии с индивидуальным заданием формируют заданную модель и настраивают ее параметры. Вторая часть работы посвящена исследованию сформированной модели.

Полученные результаты студенты оформляют индивидуально в виде отчета и представляют его преподавателю.

3. Содержание отчета

Отчет должен включать в себя следующие пункты:

1. Задание на выполнение лабораторной работы.
2. Имитационную модель системы, сформированную с использованием пакета Simulink.
3. Результаты моделирования.

4. Пример задания на лабораторную работу №1

Задача лабораторной работы - моделирование фильтра Винера в Simulink и оценка результатов моделирования.

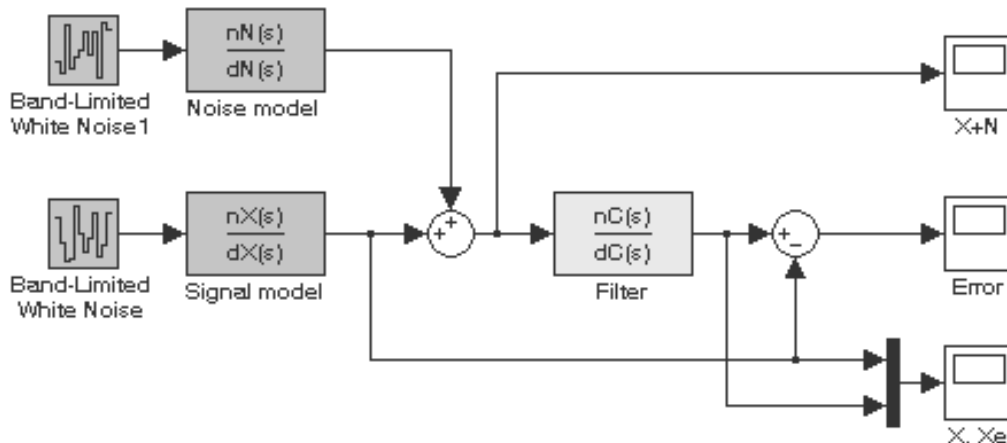
Инструкция по выполнению работы

<p>1. Создайте новый m-файл (скрипт) с командами, которые очищают рабочую область и задают формирующие фильтры и спектральные плотности полезного сигнала $S_x(\omega) = \frac{1}{\omega^2 + 1}$ и помехи $S_N(\omega) = 0,01$ (белый шум):</p>
<pre>clear all; Fx = tf(1, [1 1]); Fn = tf(0.1); Sx = Fx*Fx'; Sn = Fn*Fn';</pre>
<p>2. Скопируйте в рабочую папку файлы wiener.m и h2opt.m, добавьте в скрипт команду [C,errOpt] = wiener(Sx, Sn) и запустите скрипт клавишей F5. Запишите в отчет передаточную функцию оптимального фильтра и СКВО ошибки фильтрации (значение errOpt).</p>

3. Добавьте в скрипт команды для подготовки данных к моделированию

```
[nX,dX] = tfdata(Fx, 'v');
[nN,dN] = tfdata(Fn, 'v');
[nC,dC] = tfdata(C, 'v');
b = bandwidth(Fx*C); % полоса пропускания Fx(s)*W(s)
tau = 2*pi/100/b; % интервал корреляции для источника шума
```

4. Постройте Simulink-модель задачи фильтрации:

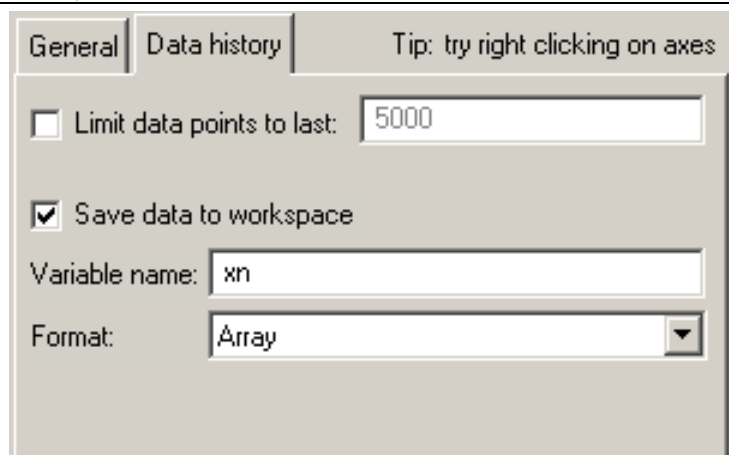


Чтобы вывести на нижний осциллограф два сигнала, используйте мультиплексор (блок Mux из группы Signal Routing). Сохраните модель с именем lab1sim.mdl.

5. В параметрах источников белого шума установите единичную интенсивность (Noise Power), интервал корреляции tau (Sample Time) и разное начальное значение последовательности случайных чисел (Seed).

6. В параметрах блока Noise model (блок Transfer Fcn из группы Continuous) задайте числитель nN и знаменатель dN (из рабочей области). Для модели полезного сигнала (Signal model) задайте числитель nX и знаменатель dX, а для фильтра (Filter) – nW и dW.

7. В параметрах первого осциллографа (обозначенного X+N) сбросьте флажок ограничивающий количество запоминаемых данных (Limit data points), и задайте вывод результатов в массив xn. Аналогично установите вывод второго осциллографа (Error) в массив err, а третьего (X, Xe) – в массив xxe.



<p>8. Установите время моделирования 50 с (меню Simulation – Simulation parameters – Stop Time). Добавьте в скрипт команды для запуска моделирования и расчета СКВО ошибки по результатам моделирования:</p> <pre>sim('lab1sim') errSim = std(err(:,2))</pre> <p>Сравните этот результат с полученным ранее теоретическим значением.</p>
<p>9. Добавьте команды для построения графика смеси «сигнал+шум», а также исходного и восстановленного сигналов (на отдельном поле).</p> <pre>figure(1) plot (xn(:,1), xn(:,2)); title('Signal + noise'); figure(2) title('Original and reconstructed signals'); plot(xxe(:,1),xxe(:,2),xxe(:,1),xxe(:,3)); legend('Original', 'Reconstructed');</pre> <p>Запустите скрипт и скопируйте графики в отчет.</p>
<p>10. Повторите расчеты для $F_N(s) = 0,01$ и $F_N(s) = 1$. Приведите в отчете графики, теоретическое СКВО ошибки и СКВО, полученное при моделировании. Сделайте выводы о влиянии интенсивности шума измерений.</p>
<p>11. В цикле постройте серию оптимальных регуляторов при изменении дисперсии помехи от 0,01 до 1 и постройте график. Для этого в скрипт нужно добавить такой код:</p> <pre>varN = [0.01:0.02:0.1 0.2:0.1:1]; % массив дисперсий помехи sigmaE = []; % пустой массив СКВО ошибки for i=1:length(varN) % цикл по всем вариантам Sn = tf(varN(i)); % спектр шума [C,errOpt] = wiener (Sx, Sn); % синтез регулятора C % показать регулятор sigmaE(i) = errOpt; % запомнить СКВО ошибки end; figure(3); % график в новом окне plot (varN, sigmaE);</pre> <p>Сделайте выводы по графику.</p>
<p>12. Добавьте в массив varN большие (от 2 до 100) значения дисперсии помехи</p> <pre>varN = [0.01:0.02:0.1 0.2:0.1:1 2:10 20:10:100];</pre> <p>и повторите расчеты.</p> <p>Посмотрите, к какому значению стремится СКВО ошибки в оптимальной системе при увеличении дисперсии помехи. Попробуйте обосновать эти результаты теоретически.</p>

5. Контрольные вопросы

1. Как запустить пакет Simulink?
2. Как устанавливаются параметры S-модели?
3. Как добавить в модель графопостроитель?
4. Как осуществить отладку S-модели?
5. Что такое маскирование подсистем?
6. Как осуществляется работа с масками?
7. Как осуществляется подготовка отчетов по моделированию?

Литература

1. Дьяконов В.П. Simulink 5/6/7. Самоучитель. - М.: ДМК-Пресс, 2008.
2. Леоненков А.В. Нечеткое моделирование в среде MATLAB и fuzzyTECH. – СПб.: БХВ-Петербург, 2005.
3. Черных И.В. Simulink: среда создания инженерных приложений /под общ. ред. В. Г. Потемкина. - М.: Диалог-МИФИ, 2003.
4. Лазарев Ю. Моделирование процессов и систем в MATLAB. Учебный курс. – СПб.: Питер, 2005.
5. Долбенков, В. И. Simulink в задачах систем автоматического управления: учеб. пособие / В.И. Долбенков ; М-во образования и науки Рос. Федерации, Федер. агентство по образованию, Юж.-Урал. гос. ун-т, каф. "Системы упр.". - Челябинск : Изд-во ЮУрГУ, 2005.

Лабораторная работа №2

Моделирование с использованием системы имитационного моделирования AnyLogic

Цель занятия – закрепление теоретических знаний по моделированию систем и практическое освоение системы имитационного моделирования AnyLogic.

Время - 6 часов.

1. Основные теоретические положения лабораторной работы

AnyLogic - программное обеспечение для имитационного моделирования, разработанное российской компанией «Экс Джей Текнолоджис». Инструмент обладает современным графическим интерфейсом и позволяет использовать язык Java для разработки моделей.

Основными строительными блоками модели в AnyLogic являются активные объекты, которые позволяют моделировать любые объекты реального мира. Активный объект является экземпляром класса активного объекта. Чтобы создать модель AnyLogic, необходимо создать классы активных объектов (или использовать объекты библиотек AnyLogic) и задать их взаимосвязи. AnyLogic интерпретирует создаваемые графически классы активных объектов в классы Java, поэтому можно пользоваться всеми преимуществами объектно-ориентированного моделирования. Активные объекты могут содержать вложенные объекты, причем уровень вложенности не ограничен. Это позволяет производить декомпозицию модели на любое количество уровней детализации. Создав класс активного объекта, можно создать любое количество объектов – экземпляров данного класса. Каждый активный объект имеет структуру (совокупность включенных в него активных объектов и их связи), а также поведение, определяемое совокупностью переменных, параметров, стейтчартов и т.п. Каждый экземпляр активного объекта в работающей модели имеет свое собственное поведение, он может иметь свои значения параметров, функционирует независимо от других объектов, взаимодействуя с ними и с внешней средой.

Основной технологией программирования в AnyLogic является визуальное программирование – построение с помощью графических объектов и пиктограмм иерархий структуры и поведения активных объектов.

Основными средствами описания поведения объектов являются переменные, события и диаграммы состояний. **Переменные** отражают изменяющиеся характеристики объекта. **События** могут наступать с заданным интервалом времени и выполнять заданное действие. **Диаграммы состояний** (или **стейтчарты**) позволяют визуально представить поведение объекта во времени под воздействием событий или условий, они состоят из графического

изображения состояний и переходов между ними (т.е. по сути это конечный автомат). Любая сложная логика поведения объектов модели может быть выражена с помощью комбинации стейтчартов, дифференциальных и алгебраических уравнений, переменных, таймеров и программного кода на Java. Алгебраические и дифференциальные уравнения записываются аналитически.

AnyLogic имеет удобные средства представления функционирования моделируемой системы в живой форме динамической анимации, что позволяет «увидеть» поведение сложной системы. Визуализация процесса функционирования моделируемой системы позволяет проверить адекватность модели, выявить ошибки при задании логики. С помощью совершенной технологии визуализации работающих моделей AnyLogic можно создавать интерактивные анимации произвольной сложности, связывая графические объекты (в т.ч. импортированные чертежи) во встроенном редакторе с объектами модели. Как и модель, анимация имеет иерархическую структуру, которая может динамически изменяться. Возможно создание нескольких точек зрения или нескольких уровней детальности в пределах одной анимации. В AnyLogic поддерживается как двумерная, так и трёхмерная анимация.

Многие системы моделирования позволяют менять параметры модели только до запуска модели на выполнение. AnyLogic позволяет пользователю вмешиваться в работу модели, изменяя параметры модели в процессе ее функционирования. Примером таких средств являются *слайдеры*, которые могут быть введены в окно анимации.

После запуска AnyLogic открывается рабочее окно, в котором для продолжения работы надо создать новый проект или открыть уже существующий. Чтобы создать новый проект, щелкните по соответствующей кнопке на панели инструментов или выберите пункт меню **Файл-Создать проект** и затем из меню – **Модель**. Откроется диалоговое окно **Новая модель**, где задается имя и местоположение нового проекта. Далее следуйте указаниям **Мастера создания модели**. Можно создавать новую модель «с нуля» или использовать шаблон. При открытии проекта (нового или существующего) AnyLogic всегда открывает среду разработки проекта – графический редактор модели (рис. 1). Рассмотрим основные составляющие этого редактора.

Окно проекта обеспечивает легкую навигацию по элементам проекта, таким как пакеты, классы и т.д. Поскольку проект организован иерархически, то он отображается в виде дерева: сам проект образует верхний уровень дерева рабочего проекта, пакеты – следующий уровень, классы активных объектов и сообщений – следующий и т.д. Можно копировать, перемещать и удалять любые элементы дерева объектов, легко управляя рабочим проектом.

Одна из ветвей в дереве проекта имеет название **Simulation:Main** (эксперимент). Эксперимент хранит набор настроек, с помощью которых конфигурируют работу модели. Один эксперимент создается автоматически при создании проекта.

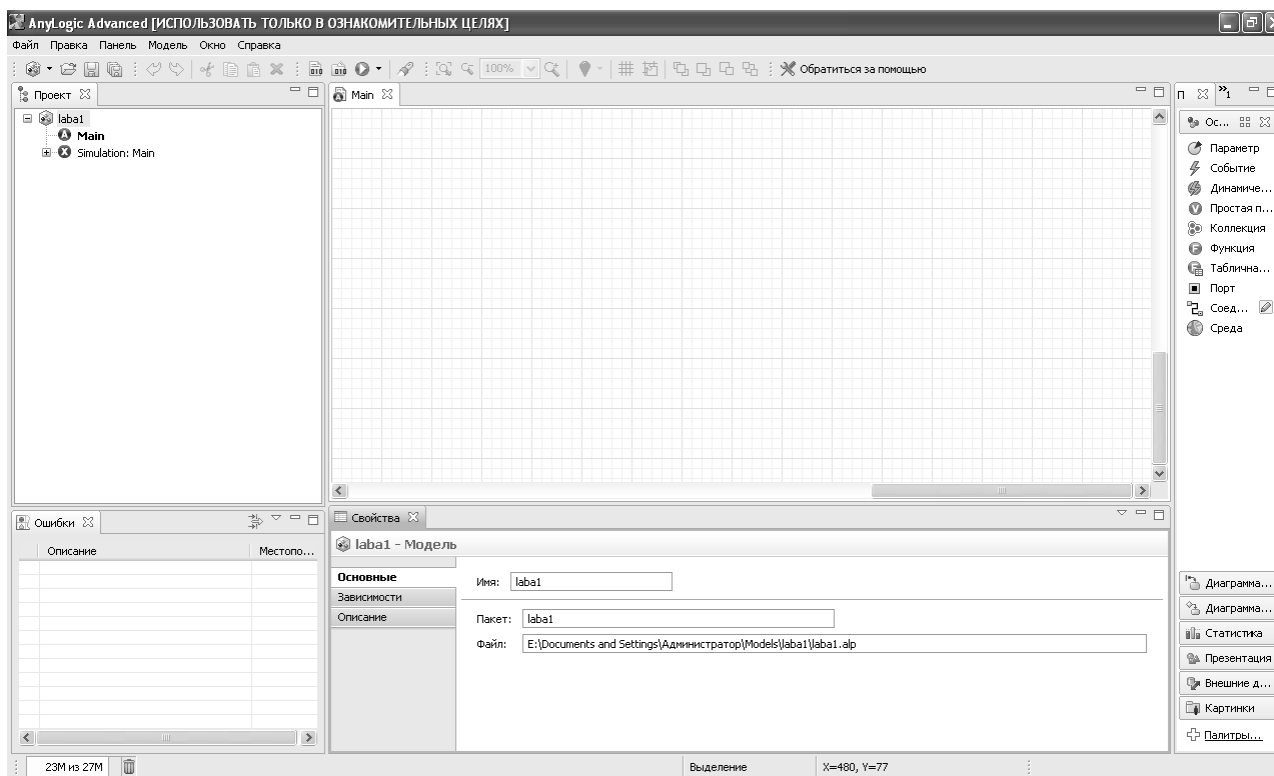


Рис. 1. Графический редактор модели AnyLogic

Это простой эксперимент с именем *Simulation*, позволяющий визуализировать модель с помощью анимации и поддерживающий инструменты для отладки модели.

Структурная диаграмма. При построении модели нужно задать ее структуру (т.е. описать, из каких частей состоит модель системы) и поведение отдельных объектов системы. В AnyLogic структурными элементами модели являются активные объекты. Структура активного объекта задается графически на структурной диаграмме. Поведение задается с помощью стейтчарта и определяет реакции активного объекта на внешние события – логику его действий во времени.

Диаграмма состояний (или стейтчарт) – это модифицированные графы переходов конечного автомата. Стейтчарт позволяет графически задать пространство состояний алгоритма поведения объекта, а также события, которые являются причинами срабатывания переходов из одних состояний в другие, и действия, происходящие при смене состояний. Стейтчарты в AnyLogic поддерживают следующие типы событий:

сигнал – объект может послать сигнал другому объекту, чтобы уведомить его о чем-то;

таймаут – в течение заданного промежутка времени в стейтчарте ничего не происходит;

событие – событие, при котором значение булево выражения становится «истина».

Окно свойств. В редакторе AnyLogic для каждого выделенного элемента модели существует свое окно свойств, в котором указываются свойства (параметры) этого элемента. При выделении какого-либо элемента в окне редактора снизу появляется окно свойств, показывающее параметры данного выделенного элемента. Окно свойств содержит несколько вкладок. Каждая вкладка содержит элементы управления, такие как поля ввода, флажки, переключатели, кнопки и т.д., с помощью которых можно просматривать и изменять свойства элементов модели. Число вкладок и их внешний вид зависит от типа выбранного элемента.

Окно палитры. Содержит элементы (графические объекты), которые могут быть добавлены на структурную диаграмму. Элементы разбиты по группам, отображаемым на разных вкладках. Чтобы добавить объект палитры на диаграмму, щелкните сначала по элементу в палитре, а затем щелкните по диаграмме.

Активный объект может иметь параметры. Параметры обычно используются для задания характеристик объекта. Можно задать различные значения параметров для разных объектов одного и того же класса, что требуется в тех случаях, когда объекты имеют одинаковое поведение, но их характеристики разные. Возможно описание параметров любых Java-классов. Чтобы создать параметр класса активного объекта (см. рис. 2), в окне *Проект* щелкните мышью по классу активного объекта. В окне *Свойства* щелкните по кнопке *Новый параметр*. Задайте свойства параметра в открывшемся диалоговом окне *Параметр*.

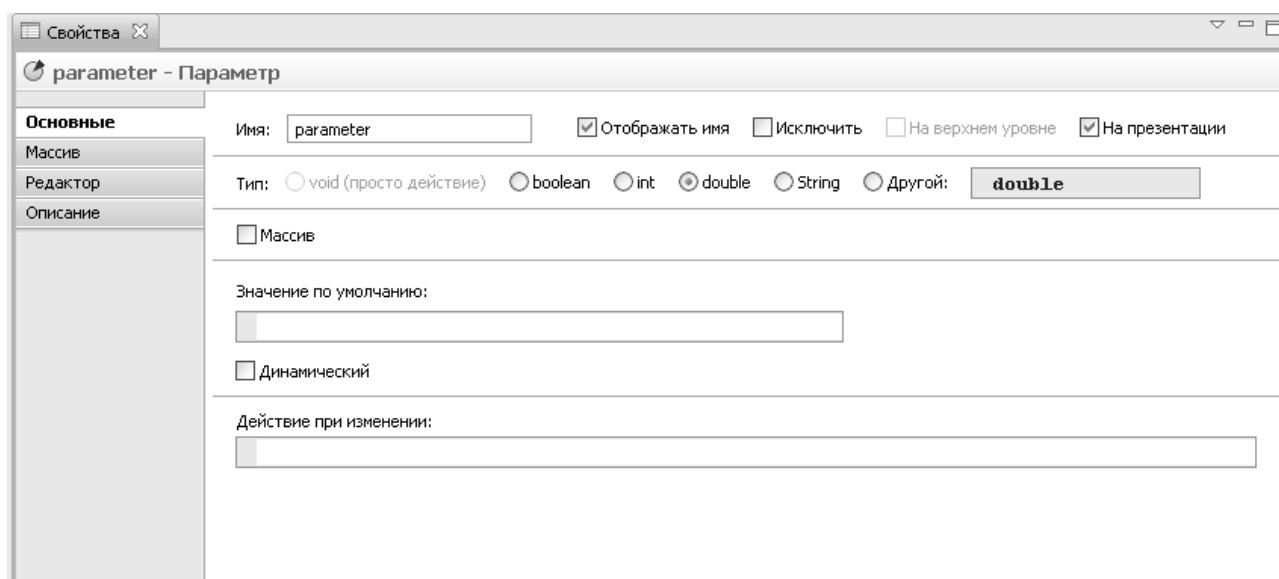


Рис. 2. Диалоговое окно *Параметр*

Активный объект может содержать переменные. Переменные могут быть либо внутренними, либо интерфейсными. Активный объект может иметь

переменные, моделирующие, меняющиеся во времени величины. Переменные могут быть вынесены в интерфейс активного объекта и связаны с переменными других активных объектов. Тогда при изменении значения одной переменной будет немедленно меняться и значение связанной с ней зависимой переменной другого объекта. Этот механизм обеспечивает непрерывное и/или дискретное взаимодействие объектов.

Запускать и отлаживать модель можно с помощью меню *Модель* и панели инструментов (см. рис. 3).

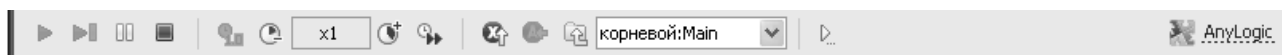


Рис. 3. Панель инструментов

При запуске модели запускается компилятор, который построит исполняемый код проекта на языке Java, оттранслирует его и затем запустит модель на исполнение. При этом открывается окно наблюдения, в котором, в свою очередь, открываются окна:

- окно структуры модели, в котором в дереве с корнем *root* можно наблюдать мгновенные значения переменных и параметров;
- окно анимации;
- окно поведения (стрейтчарт) с подсвеченным красным цветом тем состоянием, которое активно в данный момент;
- окна графиков, где иллюстрируются изменения переменных в модельном времени.

Полный перечень возможностей и описание системы AnyLogic приведены в [1].

2. Порядок выполнения работы

2.1. На этапе подготовки к лабораторной работе студенты должны, используя литературу [1-3], конспект лекций и учебные материалы данного пособия, углубить свои знания по системе имитационного моделирования AnyLogic.

2.2. Преподаватель перед проведением занятия проводит контрольный опрос студентов и определяет степень их готовности к лабораторной работе. Затем преподаватель разбивает группу студентов из расчета не более 10 подгрупп.

Каждая подгруппа получает от преподавателя индивидуальный вариант задания на лабораторную работу.

Лабораторная работа состоит из двух частей. В первой части работы студенты, используя систему AnyLogic, в соответствии с индивидуальным заданием формируют заданную модель и настраивают ее параметры. Вторая часть работы посвящена исследованию сформированной модели.

Полученные результаты, студенты оформляют индивидуально в виде отчета и представляют его преподавателю.

3. Содержание отчета

Отчет должен включать в себя следующие пункты:

1. Задание на выполнение лабораторной работы.
2. Имитационную модель системы, сформированную с использованием системы AnyLogic.
3. Результаты моделирования.

4. Пример задания на лабораторную работу №2

Задача лабораторной работы - создать и изучить типичную системно-динамическую модель.

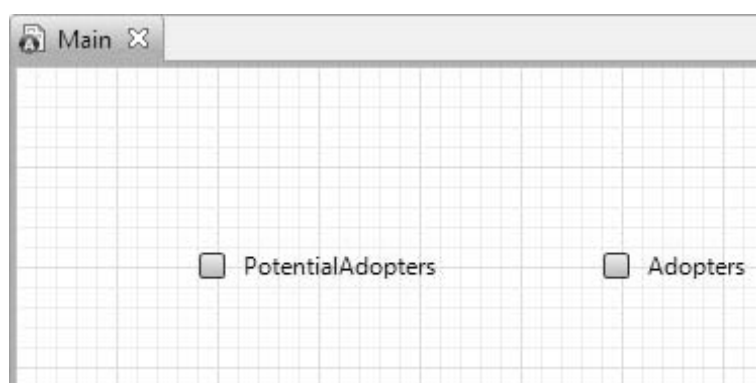
Инструкция по выполнению работы

1. Создайте новый проект для будущей модели и сохраните его в своей папке. Откройте структурную диаграмму двойным щелчком мыши по элементу дерева Main в окне Проект.

2. Создайте два накопителя.

Для этого перетащите элемент Накопитель из палитры Системная динамика на диаграмму класса активного объекта. На диаграмме появится маленький голубой прямоугольник, обозначающий переменную-накопитель.

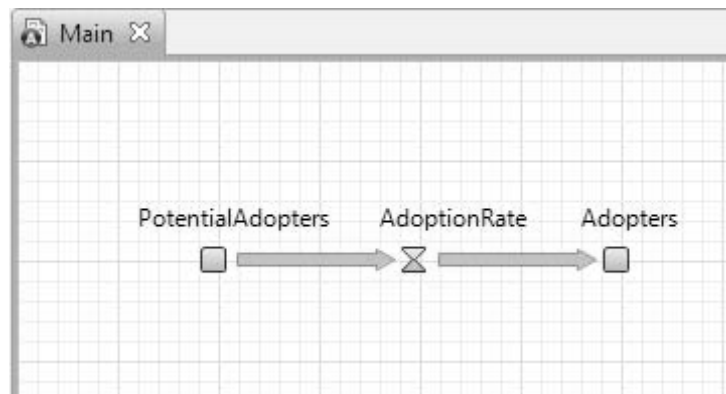
Измените имя накопителя – окно Свойства, вкладка Основные, введите PotentialAdopters в поле редактирования Имя. Таким же образом создайте еще один накопитель, назовите его Adopters.



3. Создайте поток.

Чтобы создать поток, сделайте двойной щелчок мыши по накопителю PotentialAdopters, а потом щелкните по накопителю Adopters. AnyLogic создаст новую переменную-поток и сделает ее исходящим потоком для накопителя PotentialAdopters и входящим – для Adopters. На диаграмме появятся стрелки,

которые будут обозначать образовавшиеся зависимости между потоком и этими накопителями. Выделите созданную переменную в графическом редакторе и измените имя этого потока на AdoptionRate.



4. Посмотрите свойства накопителей.

Свойства

PotentialAdopters - Накопитель

Основные

Имя: Отображать имя

Массив

Массив

Начальное значение:

$d(\text{PotentialAdopters})/dt =$

Свойства

Adopters - Накопитель

Основные

Имя: Отображать имя

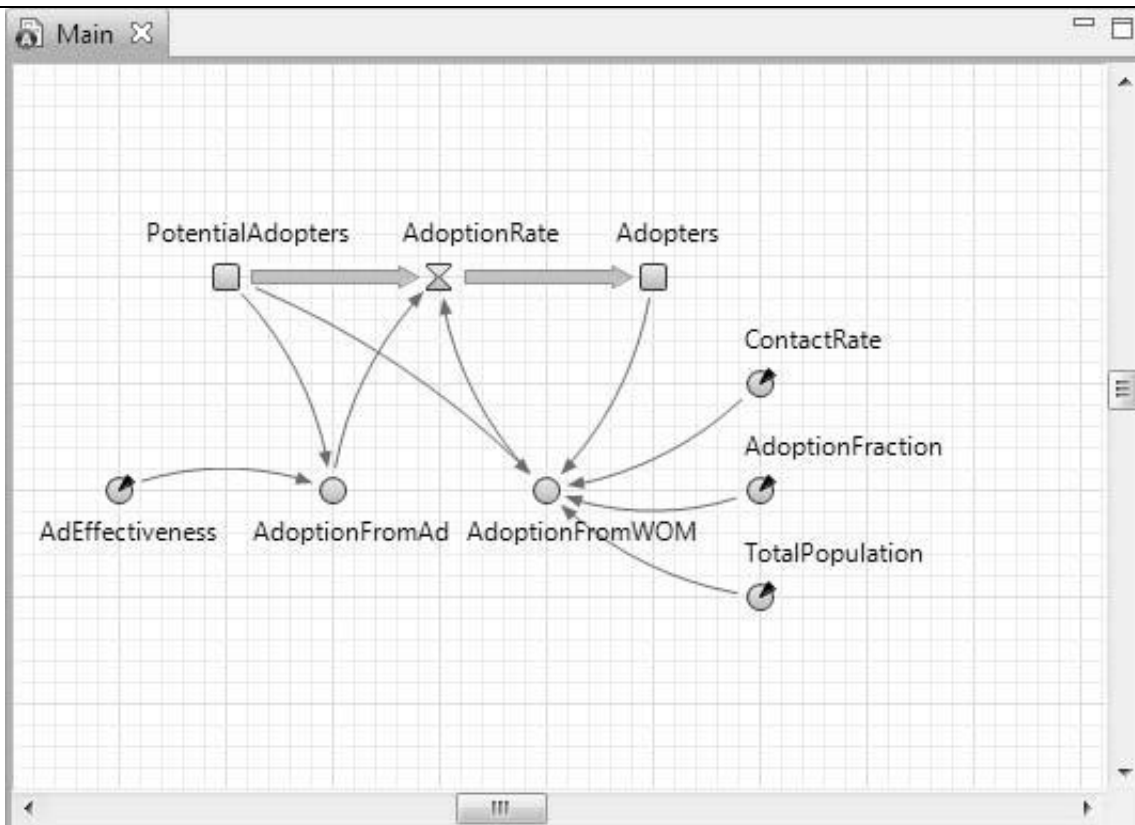
Массив

Массив

Начальное значение:

$d(\text{Adopters})/dt =$

<p>5. Создайте константы модели.</p> <p>Перетащите элемент Параметр из палитры Системная динамика на диаграмму класса активного объекта. Далее:</p> <ul style="list-style-type: none"> - в поле Имя введите TotalPopulation. В поле По умолчанию введите 100000. Можно задать также краткое описание константы в поле Описание; - создайте константу AdEffectiveness. Задайте значение по умолчанию 0.011; - задайте частоту и назовите ее ContactRate. Задайте значение по умолчанию 100; - задайте константу AdoptionFraction. Задайте значение 0.015.
<p>6. Задайте начальные значения накопителей.</p> <p>Начальное равно нулю, поэтому в окне свойств накопителя Adopters введите 0 в поле редактирования Начальное значение. В окне свойств накопителя PotentialAdopters введите TotalPopulation в поле редактирования Начальное значение. Вы можете сделать это с помощью Мастера (Ctrl + пробел).</p>
<p>7. Создайте две вспомогательные переменные:</p> <ul style="list-style-type: none"> - перетащите элемент Вспомогательная переменная из палитры Системная динамика на диаграмму класса активного объекта и назовите ее AdoptionFromAd. В поле <u>AdoptionFromAd=</u> введите: $AdEffectiveness * PotentialAdopters$. - создайте еще одну переменную и назовите ее AdoptionFromWOM. Задайте формулу: $ContactRate * AdoptionFraction * PotentialAdopters * Adopters / TotalPopulation$.
<p>8. Задайте формулу.</p> <p>Значение потока определяется суммой двух его независимых составляющих. В окне свойств переменной AdoptionRate на вкладке Основные введите формулу, по которой будет вычисляться значение потока, в поле <u>AdoptionRate=</u>: $AdoptionFromAd + AdoptionFromWOM$.</p> <p>Создание модели завершено.</p>



9. Настройте текущий эксперимент модели.

В окне свойств эксперимента Simulation:Main перейдите на вкладку Модельное время, выберите В заданное время из списка Остановить. В расположенном ниже поле введите 8. Модель остановится после того, как истекнут 8 единиц модельного времени.

Задайте выполнение модели в режиме реального времени (вкладка/презентация окна свойств эксперимента).

Задайте скорость выполнения – 2.

Если вы не укажете никакого конкретного метода, т. е. оставите выбранный по умолчанию метод Automatic, то во время работы модели AnyLogic будет автоматически выбирать численный метод в соответствии с поведением системы. На вкладке Дополнительные окна свойств эксперимента выберите метод RK4 из выпадающего списка Дифф. уравнения.

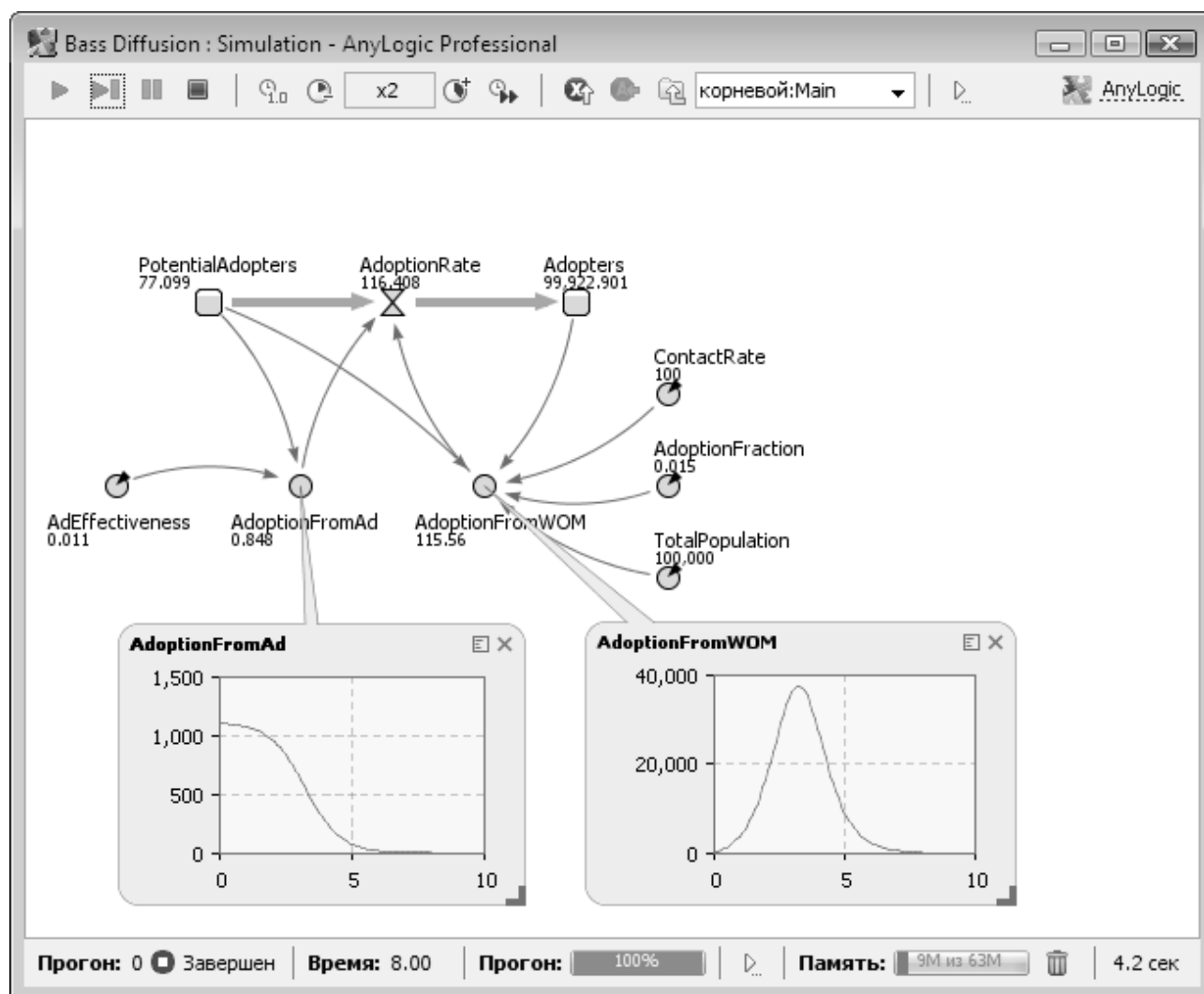
10. Проверьте ошибки и запустите модель.

Для проверки ошибок постройте проект с помощью кнопки панели инструментов Построить (или клавиша F7). В окне Ошибки появится список всех ошибок, обнаруженных в проекте, если таковые имеются. Двойным щелчком мыши по ошибке в этом списке вы можете перейти к предполагаемому месту ошибки, чтобы исправить ее. После построения проекта запустите модель.

11. Просмотрите значения переменных в окне работающей модели.

12. Исследуйте динамику обеих составляющих потока.

Для этого откройте окно инспекта для переменной AdoptionFromAd в окне презентации. Вы можете переключить окно инспекта в режим графика – оно будет отображать временной график изменений значения переменной в модельном времени. Текущее значение переменной будет отображаться рядом с началом координат графика. Окно инспекта автоматически масштабируется таким образом, чтобы полностью вместить кривые графиков от начала до конца периода моделирования. Откройте окно инспекта переменной AdoptionFromWOM и переключите его в режим графика.



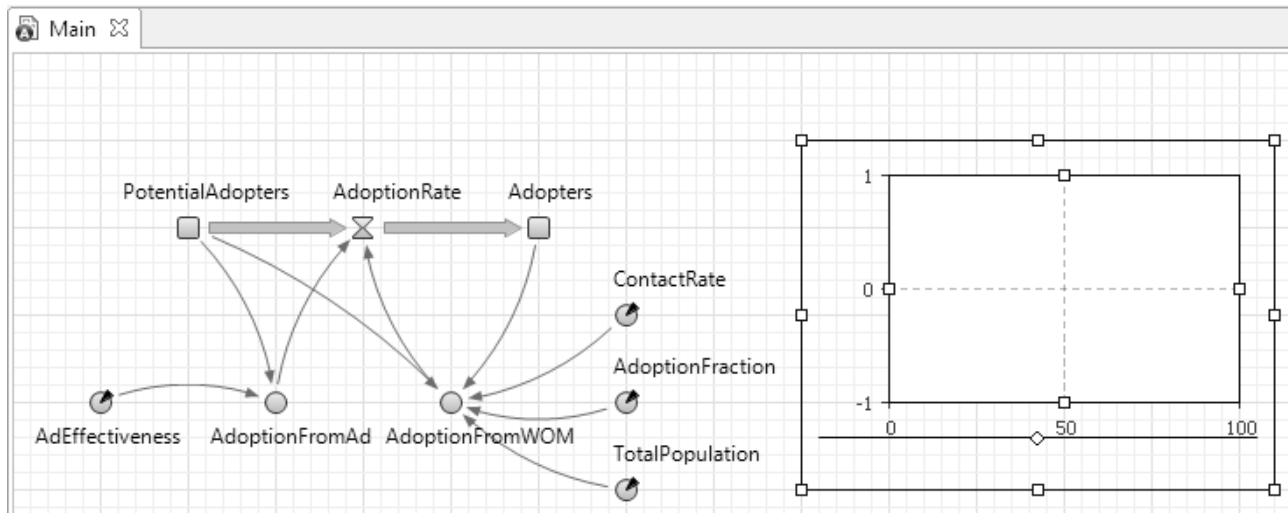
13. Изучите динамику модели с помощью диаграмм.

Для этого создайте диаграмму для отображения переменных Adopters и PotentialAdopters. Перетащите элемент Временной график из палитры Статистика на диаграмму класса Main и измените размер графика.

Перейдите на вкладку Основные панели Свойства. В поле Временной диапазон задайте диапазон временной оси диаграммы – 8. Диаграмма будет отображать график только для заданного временного интервала.

Добавьте элементы данных, историю изменения значений которых вы хотите

отображать на временном графике: щелкните мышью по кнопке **Добавить элемент данных**. Введите в поле **Выражение** имя соответствующего накопителя – **PotentialAdopters**. В поле **Заголовок** введите **Potential adopters**. Данная строка будет отображаться в легенде диаграммы для этого элемента данных. Выберите первую опцию из выпадающего списка **Стиль маркера**, чтобы наносимые на кривую графика точки не отображались дополнительными точками – маркерами. Добавьте на график еще один элемент данных, который будет отображать значение накопителя **Adopters**.



14. Добавьте график.

Для этого добавьте на диаграмму еще один временной график, поместите его под добавленным ранее графиком. Измените свойства графика. В качестве выражения должно быть задано имя потока **AdoptionRate**.

15. Запустите модель.

Первая диаграмма показывает, как изменяются переменные **PotentialAdopters** и **Adopters** во время «прогона» модели. Они представляют собой классические S-образные кривые.

16. Проанализируйте характеристики модели для своего варианта.

Поместите графики и диаграммы в отчет. Сделайте выводы.

5. Контрольные вопросы

1. В чем назначение стейтчартов?
2. Как запустить модель на выполнение?
3. В чем назначение слайдеров?
4. Что такое виртуальное время?
5. Как переключиться из режима виртуального времени в реальное?
6. Как изменить скорость выполнения модели?
7. Как изменяются режимы отрисовки изображения?
8. В чем смысл эксперимента в программе AnyLogic?

9. Какие типы экспериментов поддерживаются программой AnyLogic?
10. Как изменить текущие значения переменных и параметров модели при ее выполнении?
11. Как показать график изменения переменной модели?

Литература

1. Карпов Ю.Г. Имитационное моделирование систем. Введение в моделирование с AnyLogic 5. – СПб.: БХВ-Петербург, 2009.
2. Кисилева М.В. Имитационное моделирование систем в среде AnyLogic: учебно-методическое пособие /М.В. Киселёва. Екатеринбург: УГТУ – УПИ, 2009.
3. Осоргин А.Е. AnyLogic 6. Лабораторный практикум /А.Е. Осоргин. – Самара: ПГК, 2011.