

Опыт использования OpenModelica

Сениченков Ю.Б.

Универсальность

- ▶ ООМ
- ▶ ООМ + математические модели
- ▶ ООМ + математические модели+ имитационные модели

Универсальные среды

- ▶ Matlab-Simulink, [Matlab \(для студентов\)](#),
- ▶ Maple-Maplesim ([Try MapleSim free for 15 days with no obligation](#))
- ▶ Mathematica-SystemModeler ([Try SystemModeler for Free, free 30-day trial of SystemModeler via download](#))
- ▶ Dymola, [OpenModelica](#)
- ▶ AnyDynamics (professional), [AnyDynamics](#)

Объектно-ориентированное
моделирование

Matlab

<https://www.mssoft.ru/Makers/MathWorks/MATLAB/>

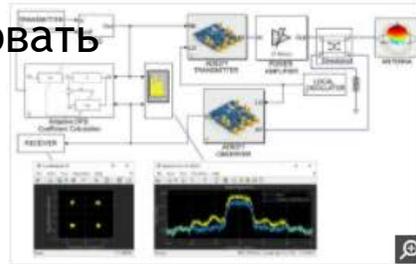
- ▶ MATLAB – это высокоуровневый язык технических расчетов, интерактивная среда разработки алгоритмов и современный инструмент анализа данных.
- ▶ MATLAB по сравнению с традиционными языками программирования (C/C++, Java, Pascal, FORTRAN) позволяет на порядок сократить время решения типовых задач и значительно упрощает разработку новых алгоритмов.
- ▶ Ядро MATLAB позволяет максимально просто работать с матрицами реальных, комплексных и аналитических типов данных и со структурами данных и таблицами поиска.
- ▶ MATLAB содержит встроенные функции линейной алгебры (LAPACK, BLAS), быстрого преобразования Фурье (FFTW), функции для работы с полиномами, функции базовой статистики и численного решения дифференциальных уравнений; расширенные математические библиотеки для Intel MKL.

Matlab - Simulink

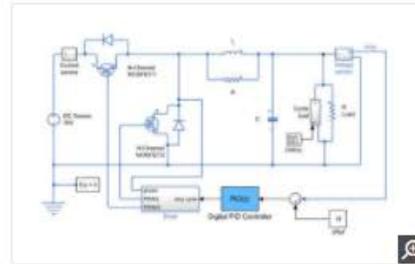


Simulink is for Every Project

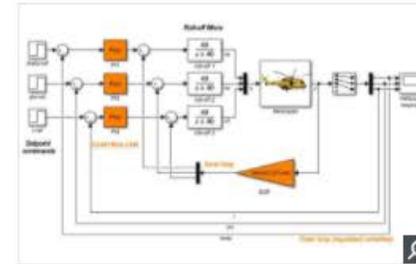
Кубики можно
ТОЛЬКО
ИСПОЛЬЗОВАТЬ



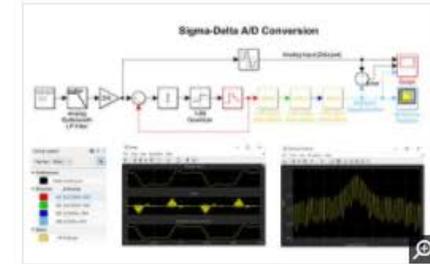
Wireless Communications



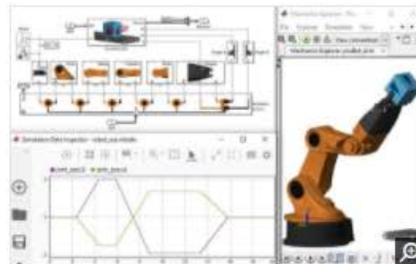
Power Electronics



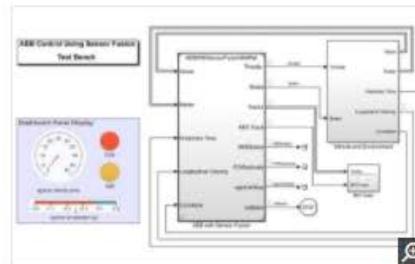
Control Systems



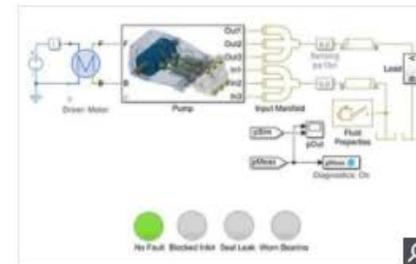
Signal Processing



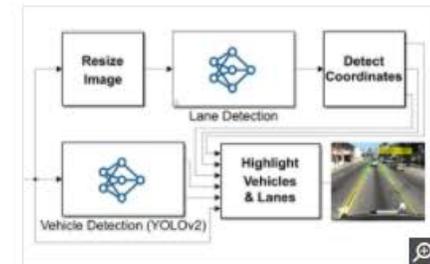
Autonomous Systems and Robotics



Advanced Driver Assistance Systems



Digital Twins

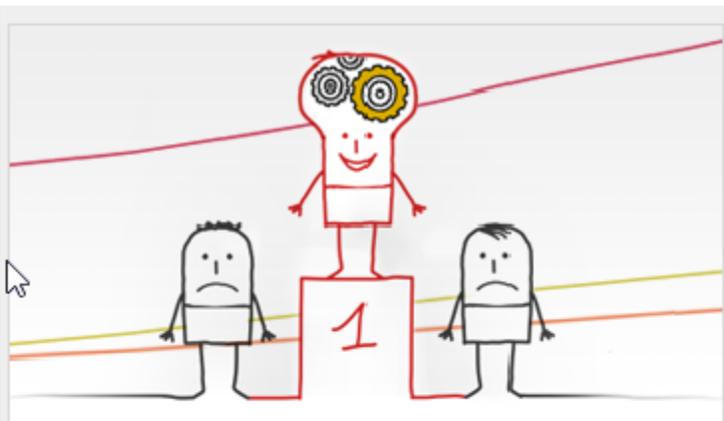


Artificial Intelligence



WOLFRAM SYSTEM MODELER

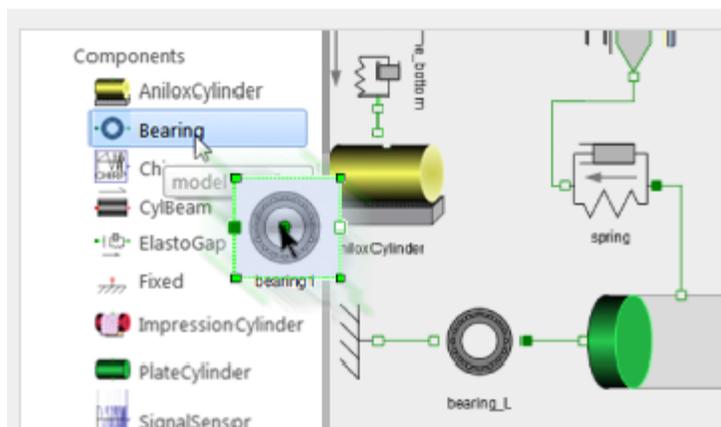
Движущая сила к пониманию, инновациям и результатам



Сравнение SystemModeler

Узнайте о преимуществах SystemModeler по сравнению с конкурентами.

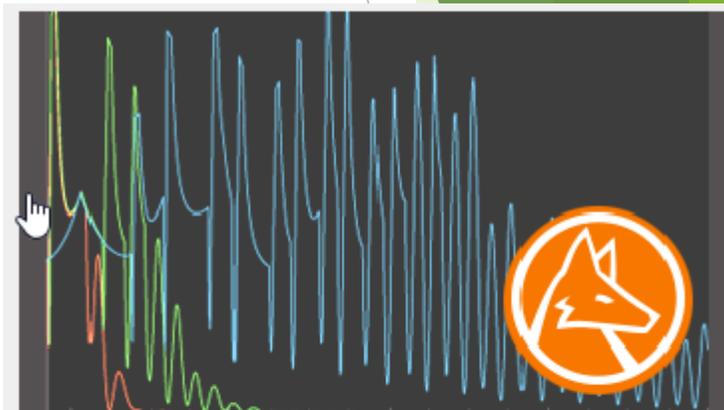
Wolfram SystemModeler является наиболее полной средой для моделирования физических систем. В отличие от других сред моделирования, SystemModeler не требует для работы установки никаких дополнительных модулей и полностью поддерживает язык Modelica. SystemModeler, также, поддерживает работу с языком Wolfram Language для создания единого рабочего цикла моделирования, симуляций и анализа. [Less](#)



Создание моделей

Интуитивно и быстро создавайте модели в SystemModeler, используя drag-and-drop подход. Выбирайте нужные компоненты, например, транзисторы или пружины, и размещайте их внутри рабочей области.

[More](#)



Широкие возможности для анализа

Выполняйте символьные и численные расчеты, работая с уравнениями моделей и результатами симуляций. Используйте всю мощь языка Wolfram Language для анализа моделей. [More](#)

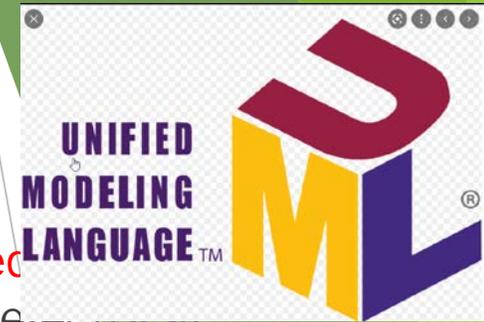
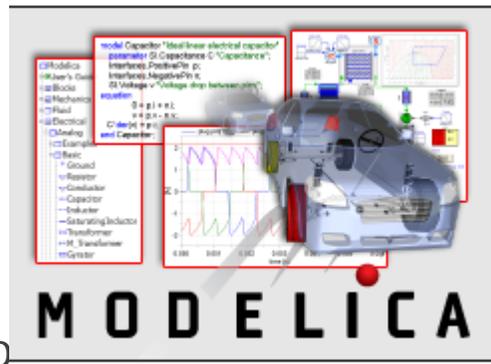
 [Wolfram SystemModeler: начало работы с Mathematica](#)

Что использовать

- ▶ Промышленность - что хотите, лишь бы Вам было хорошо.
- ▶ Образование - то, что позволит научить студента осознанно выбирать нужный инструмент для решения конкретной задачи.
 - ООМ,
 - Математические пакеты и библиотеки,
 - Среды моделирования,
 - В рамках отведенных на это часов
!!!!!!

Что использовать

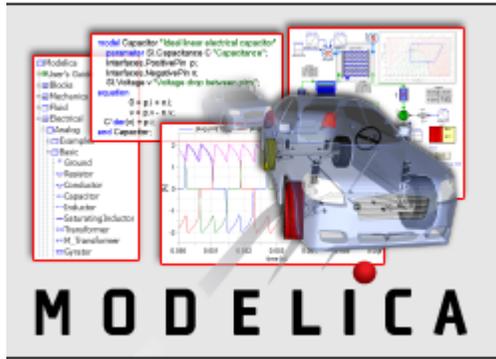
- ▶ Промышленность - что хотите, лишь бы Вам было хорошо.
- ▶ Образование - то, что позволит научить студента осознанно выбирать нужный инструмент для решения конкретной задачи.
 - ООМ,
 - Математические пакеты и библиотеки,
 - Среды моделирования,
 - В рамках отведенных на это часов
!!!!!!



- ▶ The Modelica Language is a non-proprietary, oriented, equation based **conveniently model complex physical systems containing**, e.g., mechanical, electrical, electronic, hydraulic, thermal, control, electric power or process-oriented subcomponents. See also, overview in [pdf](#), [ppt](#) format and [Modelica Language Specification 3.5](#).
- ▶ ~~Modelica Simulation Environments are available [commercially and free of charge](#)~~, such as CATIA Systems, Dymola, JModelica.org, LMS AMESim, MapleSim, Modelon Impact, MWorks, OpenModelica, SimulationX, and Wolfram SystemModeler. Modelica models can be imported conveniently into Simulink using export features of Dymola, MapleSim, and SimulationX.

OpenModelica

Кубики можно не только использовать, но и модифицировать!



Инкапсуляция
Наследование
Полиморфизм
Пакеты

Libraries

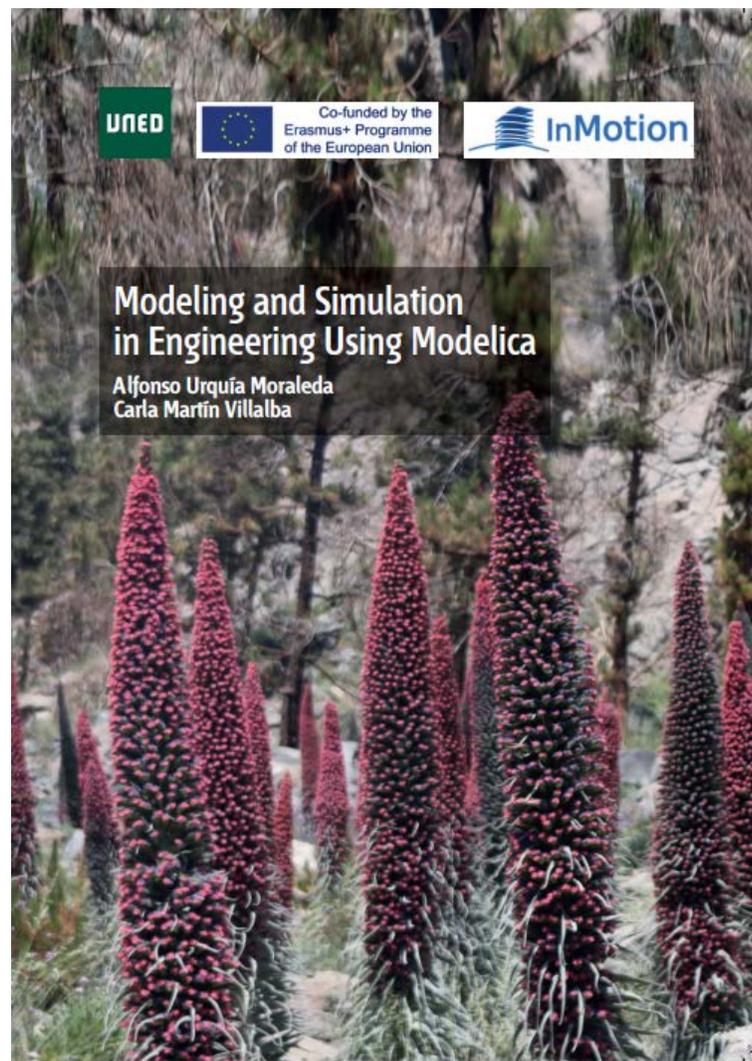
- >  OpenModelica
- >  ModelicaReference
- >  ModelicaServices
- >  Complex
- ✓  Modelica
 - >  UsersGuide
 - >  **Blocks**
 - >  ComplexBlocks
 - >  Clocked
 - >  StateGraph
 - >  Electrical
 - >  Magnetic
 - >  Mechanics
 - >  Fluid
 - >  Media
 - >  Thermal
 - >  Math
 - >  ComplexMath
 - >  Utilities
 - >  Constants
 - >  Icons
 - >  Units

OpenModelica Matlab interface

<https://www.openmodelica.org/doc/OpenModelicaUsersGuide/latest/ommatlab.html>

- ▶ OMMatlab is architected to combine both the solving strategy and model building. So domain experts (people writing the models) and computational engineers (people writing the solver code) can work on one unified tool that is industrially viable for optimization of Modelica models, while offering a flexible platform for algorithm development and research.

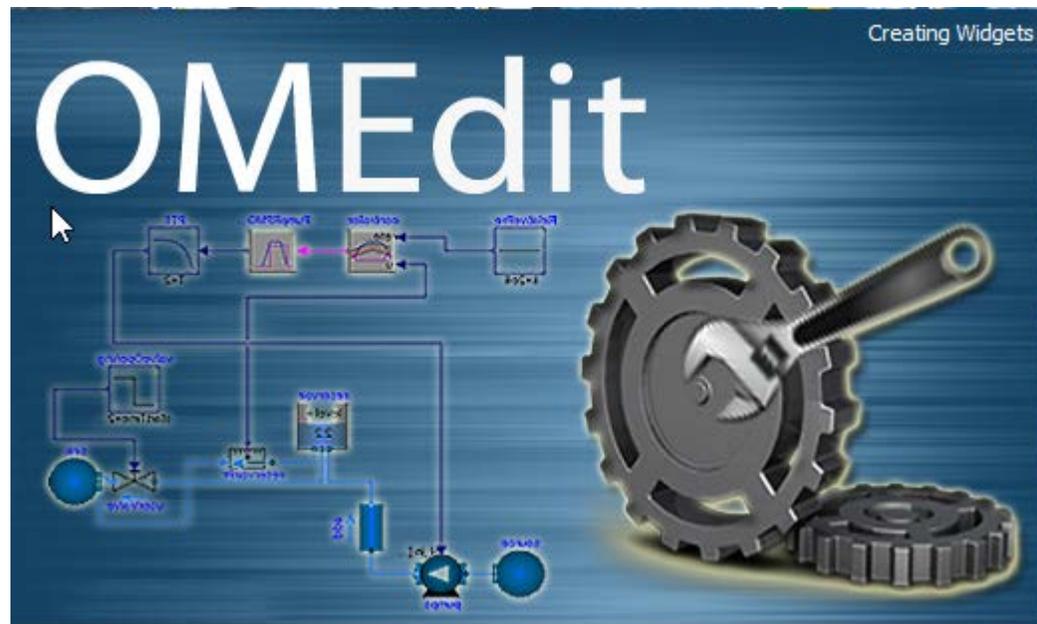
Книги проекта InMotion



Contents

Modelica code	11
Preface	13
I Continuous-time modeling	
Lesson 1 Modeling methodology and tools	19
1.1 Introduction	20
1.2 Physical modeling paradigm	20
1.3 Object-oriented modeling	23
1.4 Modeling environments	26
1.5 Getting started with Modelica	29
1.6 Further reading.	42
Lesson 2 Continuous-time atomic models	45
2.1 Introduction	46
2.2 Rectifier circuit.	46
2.3 Translation in one dimension	56
2.4 Translation in two dimensions	61
2.5 Radial heat transfer in a pipe.	69
2.6 Further reading.	75
Lesson 3 Model libraries	77
3.1 Introduction	78
3.2 Electrical library	78
3.3 Longitudinal vibrations of a bar	92
3.4 Longitudinal heat conduction in a bar	96
3.5 Control of level and temperature in a tank	101
3.6 Dissipation of heat generated in a circuit	114
3.7 Further reading.	117

<https://www.openmodelica.org/>





Libraries Browser

Filter Classes

- Libraries
- > OpenModelica
- > ModelicaReference
- > ModelicaServices
- > Complex
- ▼ Modelica
 - > UsersGuide
 - > Blocks
 - > ComplexBlocks
 - > Clocked
 - > StateGraph
 - > Electrical
 - > Magnetic
 - > Mechanics
 - > Fluid
 - > Media
 - > Thermal
 - > Math
 - > ComplexMath
 - > Utilities
 - > Constants
 - > Icons
 - > Units

OMEdit - OpenModelica Connection Editor

Recent Files

- C:/Users/senyb/Downloads/ScheglovEditSen.mo
- E:/New folder/hhj.mo
- C:/Users/senyb/Downloads/Lab3D_Base_OOM.mo
- C:/Users/senyb/Downloads/Lab3D_Base_OOM/Lab3D_Ba
- C:/Users/senyb/Modelica_examples/Lect_1.mo
- C:/Users/senyb/Modelica_examples/Switch_RC.mo
- C:/Users/senyb/Downloads/switch.mo
- C:/Users/senyb/Документы/Mod_Modelica/Modelica/sw
- C:/Users/senyb/Документы/Mod_Modelica/Modelica/R
- C:/Users/senyb/Документы/Mod_Modelica/Modelica/RC
- C:/Users/senyb/Документы/Mod_Modelica/Modelica/Dc
- C:/Users/senyb/Документы/Mod_Modelica/Modelica/ev
- C:/Users/senyb/Документы/Mod_Modelica/Modelica/fir

Clear Recent Files

Latest News

- September 4, 2021: OpenModelica 1.18.0 released!
- July 12, 2021: OpenModelica 1.18.0-dev.beta1 released!
- Join the Modelica Conference 2021!
- March 23, 2021: OpenModelica 1.17.0 released!
- February 26, 2021: OpenModelica 1.16.5 released!
- February 22, 2021: OpenModelica 1.16.4 released!
- HUBCAP Open Calls
- December 21, 2020: OpenModelica 1.16.2 released!
- November 17, 2020: OpenModelica 1.16.1 released!
- November 9. An OpenModelica overview article has been published in the MIC J

Reload

For more details visit our website www.openmodelica.org

Create New Modelica Class Open Model/Library File(s)

Documentation Browser



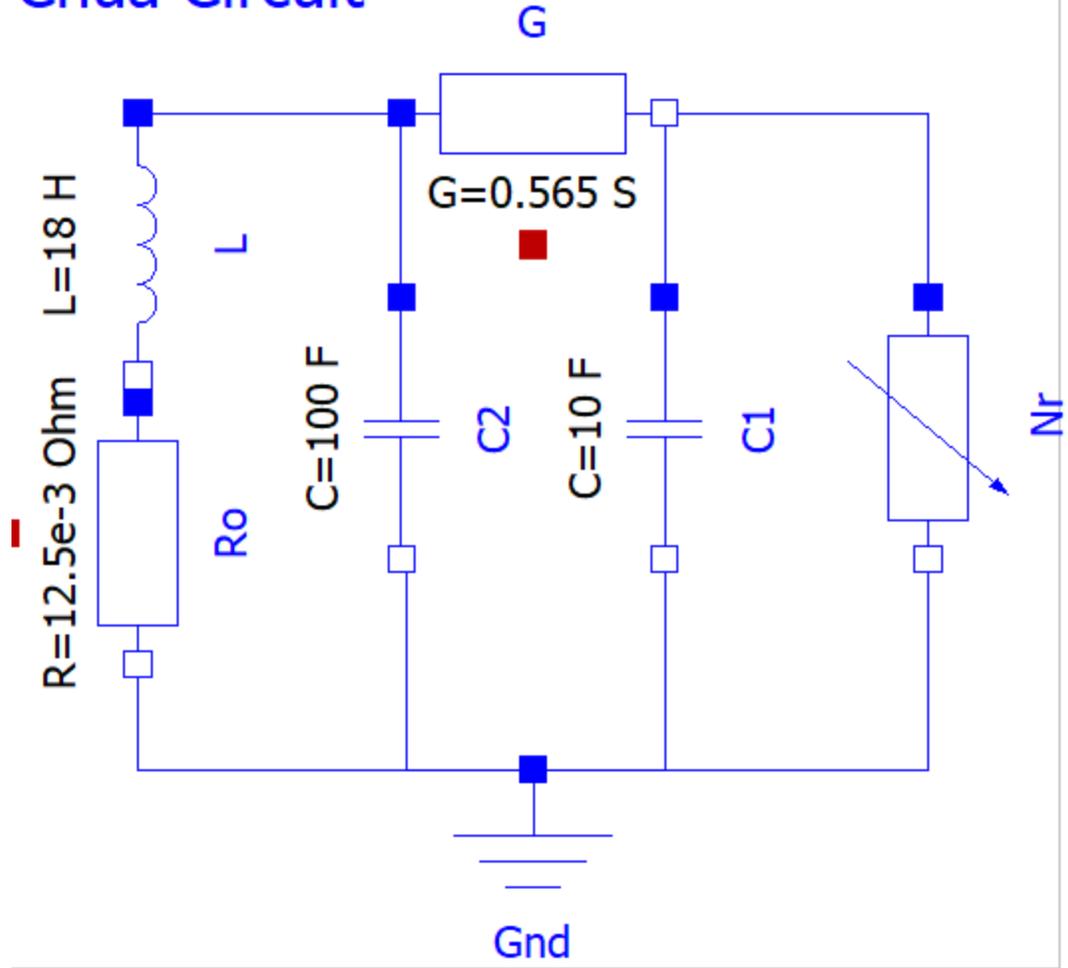
Messages Browser

- All Notifications Warnings Errors
- [1] 16:38:33 Scripting Notification
Automatically loaded package Complex 4.0.0 due to uses annotation.
- [2] 16:38:33 Scripting Notification
Automatically loaded package ModelicaServices 4.0.0 due to uses annotation.

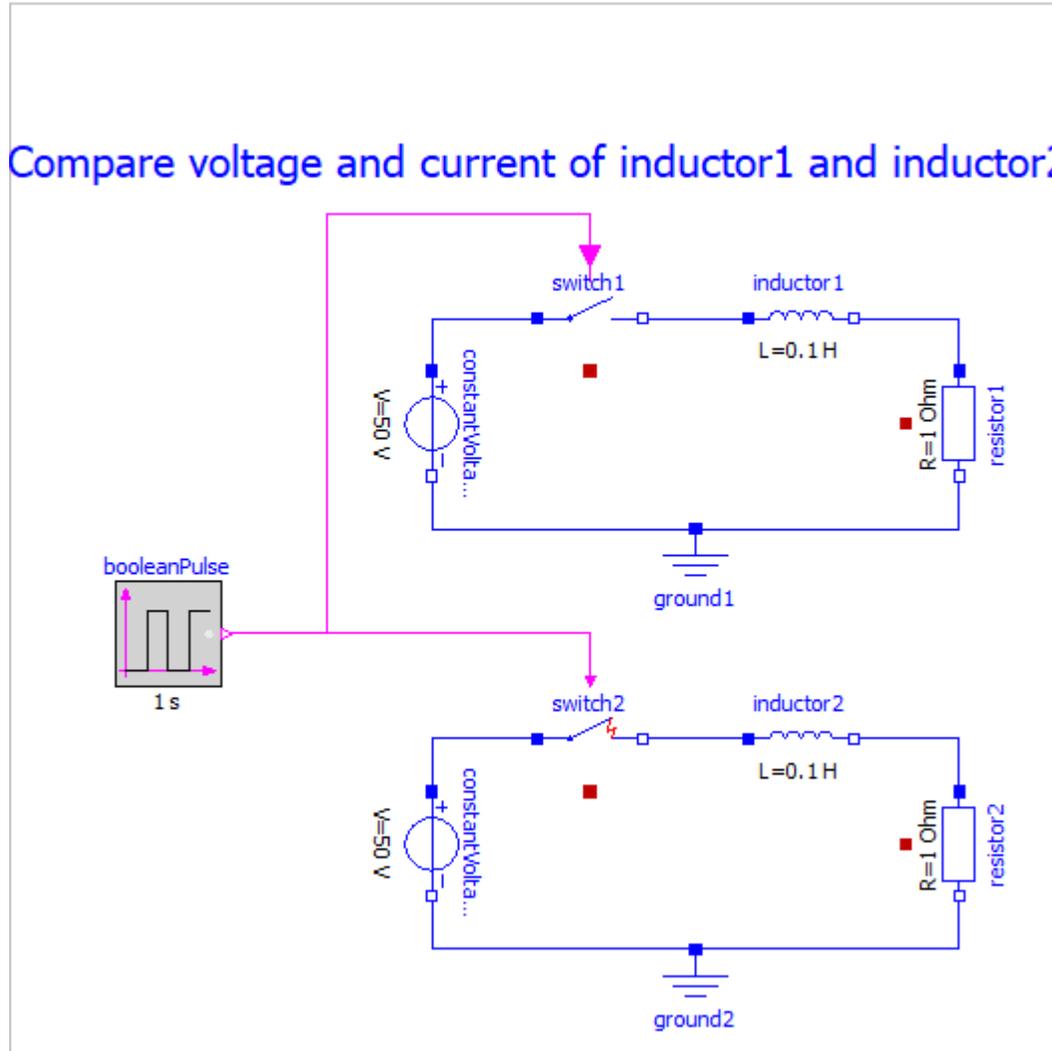
- >  UsersGuide
- ▼  Blocks
 - >  Examples
 - >  Continuous
 - >  Discrete
 - >  Interaction
 - >  Interfaces
 - >  Logical
 - >  Math
 - >  MathInteger
 - >  MathBoolean
 - >  Nonlinear
 - >  Routing
 - >  Noise
 - >  Sources
 - >  Tables
 - >  Types
 - >  Icons
- >  ComplexBlocks
- >  Clocked
- >  StateGraph

- Electrical
 - Analog
 - Digital
 - Batteries
 - Machines
 - Polyphase
 - PowerConverters
 - QuasiStatic
 - Spice3

Chua Circuit



Compare voltage and current of inductor1 and inductor2



- ▼ **kg** Units
 - > **i** UsersGuide
 - > **SI** SI
 - > **...** NonSI
 - > **→** Conversions
 - > **i** Icons

- ▼ **kg** Units
 - > **i** UsersGuide
 - ▼ **SI** SI
 - T** Angle
 - T** SolidAngle
 - T** Length
 - T** PathLength
 - T** Position
 - T** Distance
 - T** Breadth
 - T** Height
 - T** Thickness
 - T** Radius
 - T** Diameter
 - T** Area
 - T** Volume
 - T** Time
 - T** Duration
 - T** AngularVelocity

File Edit View Simulation Debug SSP Sensitivity Optimization Tools Help

Libraries Browser

Filter Classes

Libraries

- OpenModelica
- ModelicaReference
- ModelicaServices
- Complex
- Modelica
 - UsersGuide
 - Blocks
 - Examples
 - Continuous
 - Integrator
 - LimIntegrator
 - Derivative
 - FirstOrder
 - SecondOrder
 - PI
 - PID
 - LimPID
 - TransferFunction
 - StateSpace
 - Der
 - LowpassButterworth
 - CriticalDamping
 - Filter

```

351 block FirstOrder "First order transfer function block (=
    1 pole)"
352   import Modelica.Blocks.Types.Init;
353   parameter Real k(unit="1")=1 "Gain";
354   parameter SI.Time T(start=1) "Time Constant";
355   parameter Init initType=Init.NoInit
356 >   "Type of initialization (1: no init, 2: steady
state, 3/4: initial output)" annotation(Evaluate=true,
358   parameter Real y_start=0 "Initial or guess value of
output (= state)"
359   annotation (Dialog(group="Initialization"));
360
361   extends Interfaces.SISO(y(start=y_start));
362
363   initial equation
364     if initType == Init.SteadyState then
365       der(y) = 0;
366     elseif initType == Init.InitialState or initType ==
Init.InitialOutput then
367       y = y_start;
368     end if;
369   equation
370     der(y) = (k*u - y)/T;
371 >   annotation ( ... );
423 end FirstOrder;
424

```

Documentation Browser

Modelica.Blocks.Continuous.FirstOrder

First order transfer function block (= 1 pole)

Information

This block defines the transfer function between the input u and the output y as *first order* system:

$$y = \frac{k}{T \cdot s + 1} \cdot u$$

If you would like to be able to change easily between different transfer functions (FirstOrder, SecondOrder, ...) by changing parameters, use the general block **TransferFunction** instead and model a first order SISO system with parameters $b = \{k\}$, $a = \{T, 1\}$.

Example:

```

parameter: k = 0.3, T = 0.4
results in:

```

$$y = \frac{0.3}{0.4 \cdot s + 1.0} \cdot u$$

File Edit View Simulation Debug SSP Sensitivity Optimization Tools Help



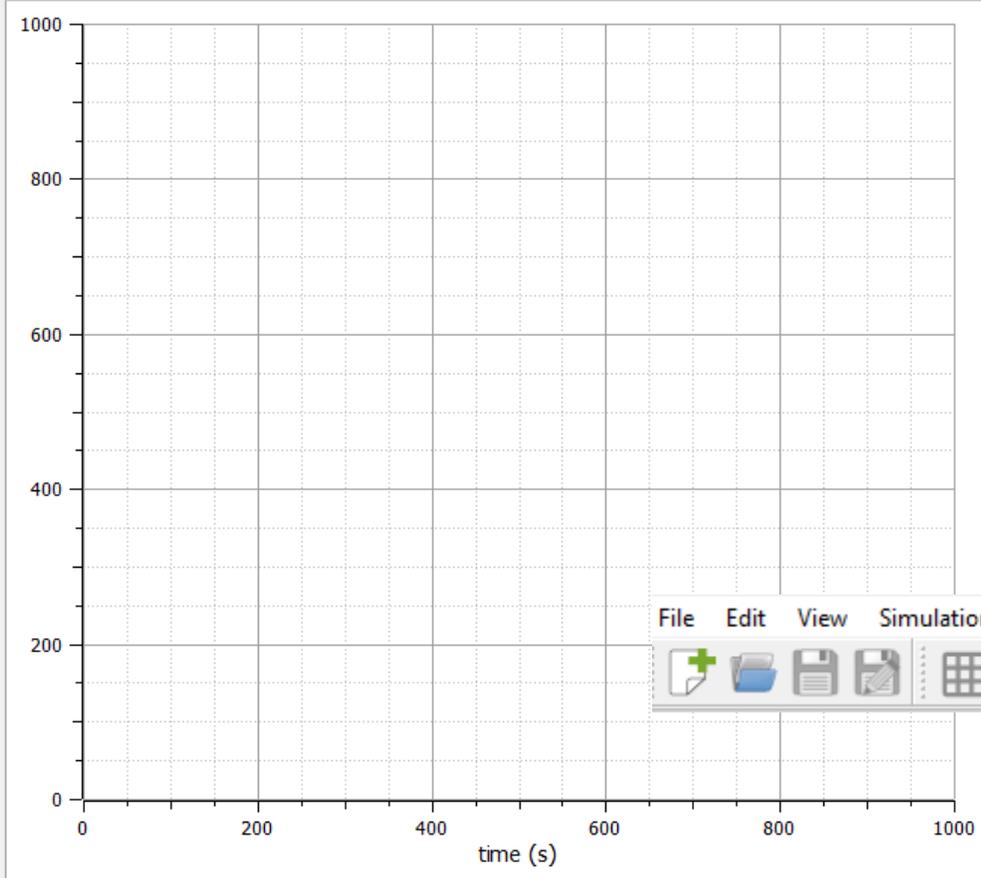


Libraries Browser

Filter Classes

- Libraries
- > OpenModelica
- > ModelicaReference
- > ModelicaServices
- > Complex
- > Modelica
- > UsersGuide
- > Blocks
- > Examples
- > Continuous
 - Integrator
 - LimIntegrator
 - Derivative
 - FirstOrder
 - SecondOrder
 - PI
 - PID
 - LimPID
 - TransferFunction
 - StateSpace
 - Der
 - LowpassButterworth
 - CriticalDamping
 - Filter
- > Internal
- > Discrete
- > Interaction
- > Interfaces
- > Logical
- > Math
- > MathInteger
- > MathBoolean
- > Nonlinear
- > Routing

Plot : 1
 Auto Scale | Fit in View | Save | Print | Grid | Detailed Grid | No Grid | Log X | Log Y | Setup



Variables Browser

Filter Variables

Simulation Time Unit: s

Time: 0.0 Speed: 1

Variables	Value	Display Unit	Description
(Active)...rstOrder			
<input type="checkbox"/> T	1.0	s	Time Constant
<input type="checkbox"/> der(y)	0		der(Connector of Real output signal)
<input type="checkbox"/> initType	1		Type of initialization (1: no init, 2: steady state, 3/4: initial output)
<input type="checkbox"/> k	1.0	1	Gain
<input type="checkbox"/> u			Connector of Real input signal
<input type="checkbox"/> y	0		Connector of Real output signal
<input type="checkbox"/> y_start	0.0		Initial or guess value of output (= state)



Messages Browser

All Notifications Warnings Errors

[1] 16:38:33 Scripting Notification
 Automatically loaded package Complex 4.0.0 due to uses annotation.

[2] 16:38:33 Scripting Notification
 Automatically loaded package ModelicaServices 4.0.0 due to uses annotation.

[1] 16:57:39 Translation Warning
 [Modelica.Blocks.Continuous: 354; 5-354:49]:
[Parameter T has no value, and is fixed during initialization \(fixed=true\), using available start value \(start=1.0\) as default value.](#)

[2] 16:57:39 Translation Warning
 Assuming fixed start value for the following 1 variables:
 output y:VARIABLE(start = y_start fixed = true) "Connector of Real output signal" type: Real

Re-simulation - Modelica.Blocks.Continuous.FirstOrder

General Interactive Simulation Translation Flags Simulation Flags Output Archived Simulations

Simulation Interval

Start Time: secs

Stop Time: secs

Number of Intervals:

Interval: secs

Integration

Method:

Tolerance:

Jacobian:

DASSL/IDA Options

Root Finding

Restart After Event

Initial Step Size:

Maximum Step Size:

Maximum Integration Order:

C/C++ Compiler Flags (Optional):

Integration

Method:

Tolerance:

Jacobian:

DASSL/IDA Options

Root Finding

Restart After Event

Initial Step Size:

- euler
- heun
- rungekutta
- impeuler
- trapezoid
- imprungekutta
- irksco
- dassl**
- ida
- cvoid

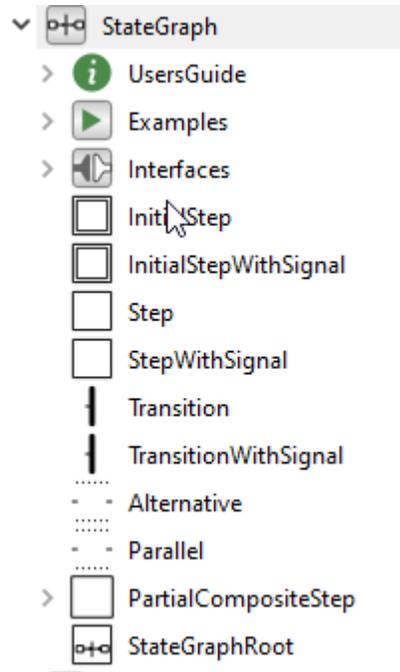
II Simulation of continuous-time models

Lesson 4	Computational causality	121
4.1	Introduction122
4.2	Classification of the model variables122
4.3	Structural singularity125
4.4	Partition algorithm126
4.5	Overdetermined and underdetermined systems128
4.6	Example: simulation of an electrical circuit130
4.7	Further reading.136
Lesson 5	Index and initialization of DAE systems	137
5.1	Introduction138
5.2	Structurally singular DAE systems138
5.3	Index of DAE systems.146
5.4	Initialization of DAE systems.157
5.5	Selection of the state variables167
5.6	Further reading.179
Lesson 6	Numerical methods	181
6.1	Introduction182
6.2	Systems of simultaneous equations182
6.3	Numerical solution of ODE.187
6.4	Numerical solution of DAE.192
6.5	Further reading.194

III Hybrid system modeling and simulation

Lesson 7	Hybrid system specification	197
7.1	Introduction198
7.2	The OHM formalism198
7.3	Model specification and simulation algorithm200
7.4	Model specification and Modelica description202
7.5	Models with a variable structure213
7.6	Model initialization223
7.7	Further reading.234
Lesson 8	Event detection and handling	235
8.1	Introduction236
8.2	Simultaneous events.236
8.3	Crossing function.242
8.4	Determination of the event instant248
8.5	Chattering.250
8.6	Further reading.254

Lesson 9	Hybrid modeling practice	255
9.1	Introduction255
9.2	Ideal electric switch263
9.3	Ideal diode265
9.4	Two-tank and valve system269
9.5	Bouncing ball272
9.6	Dry friction.274
9.7	Heat conduction in a wall282
9.8	Further reading.290
	Subject Index	291
	References	295



Modelica.StateGraph

Library of hierarchical state machine components to model discrete event and reactive systems

Information

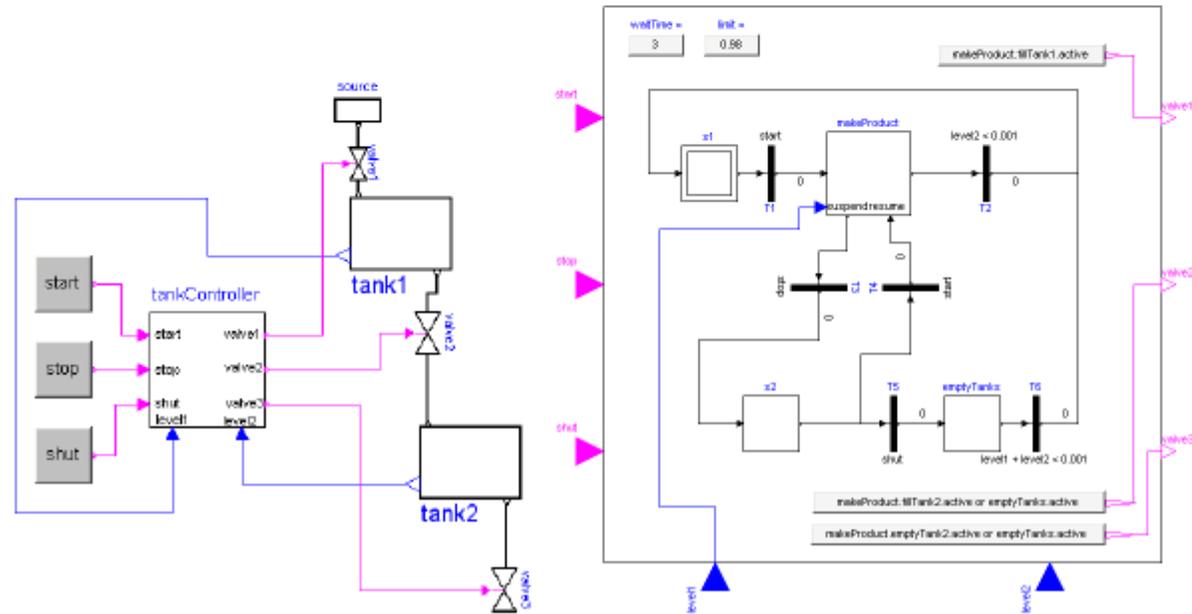
Note, there is a much improved version of this library called "Modelica_StateGraph2". If this library is not yet distributed with your Modelica tool, you can download it from https://github.com/modelica/Modelica_StateGraph2. In the [Users Guide](#) a detailed comparison is given. It is highly recommended to use Modelica_StateGraph2 instead of Modelica.StateGraph.

Library **StateGraph** is a **free** Modelica package providing components to model **discrete event** and **reactive** systems in a convenient way. It is based on the JGrafchart method and takes advantage of Modelica features for the "action" language. JGrafchart is a further development of Grafcet to include elements of StateCharts that are not present in Grafcet/Sequential Function Charts. Therefore, the StateGraph library has a similar modeling power as StateCharts but avoids some deficiencies of StateCharts.

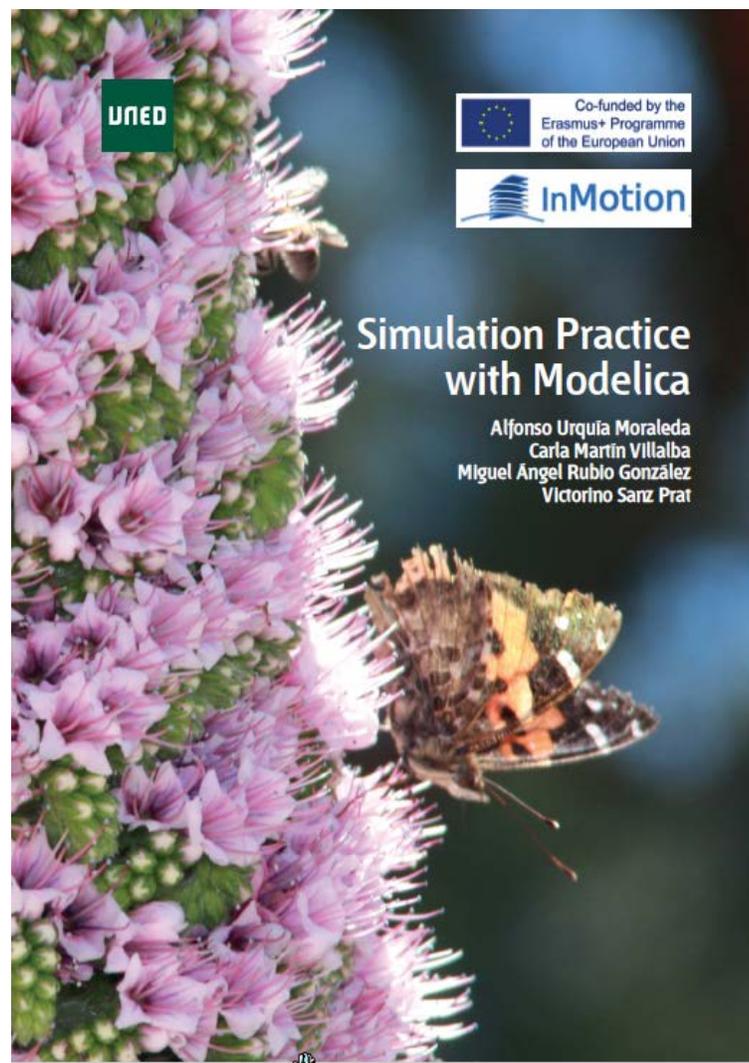
For an introduction, have especially a look at:

- [StateGraph.UsersGuide](#) discusses the most important aspects how to use this library.
- [StateGraph.Examples](#) contains examples that demonstrate the usage of this library.

A typical model generated with this library is shown in the next figure where on the left hand side a two-tank system with a tank controller and on the right hand side the top-level part of the tank controller as a StateGraph is shown:



Книги проекта InMotion



Contents

Modelica code	11
Preface	15
Assignment 1 Springs, damper and lever	21
1.1 System description	21
1.2 Tasks	22
1.3 Solution to Task 1	23
1.4 Solution to Task 2	24
1.5 Solution to Task 3	30
Assignment 2 Springs, pulley and load	33
2.1 System description	33
2.2 Tasks	34
2.3 Solution to Task 1	35
2.4 Solution to Task 2	36
2.5 Solution to Task 3	39
2.6 Solution to Task 4	42
Assignment 3 Bond graph library	45
3.1 System description	45
3.2 Tasks	45
3.3 Solution to Task 1	46
3.4 Solution to Task 2	50
Assignment 4 Source of liquid	75
4.1 System description	75
4.2 Tasks	76
4.3 Solution to Task 1	77
4.4 Solution to Task 2	77

Assignment 5	Ideal gas in a heated container	81
5.1	System description	81
5.2	Task	81
5.3	Solution	82
Assignment 6	Hysteresis controller	85
6.1	System description	85
6.2	Task	86
6.3	Solution	87
Assignment 7	Draining of a benzene storage tank	91
7.1	System description	91
7.2	Task	92
7.3	Solution	93
Assignment 8	Heating a liquid mixture	99
8.1	System description	99
8.2	Task	101
8.3	Solution	101
Assignment 9	Double-pipe heat exchanger	113
9.1	System description	113
9.2	Tasks	115
9.3	Solution to Task 1	117
9.4	Solution to Task 2	119
Assignment 10	Cellular Automata – The Game of Life	125
10.1	System description	125
10.2	Tasks	127
10.3	Solution to Task 1	127
10.4	Solution to Task 2	129
10.5	Solution to Task 3	130
10.6	Solution to Task 4	132
10.7	Solution to Task 5	148
Assignment 11	Air pollution	149
11.1	System description	149
11.2	Task	152
11.3	Solution	153
Assignment 12	Simplified Tennessee Eastman model	157
12.1	System description	157
12.2	Task 1	159
12.3	Solution to Task 1	160

12.4	Task 2	162
12.5	Solution to Task 2	162
12.6	Task 3	170
12.7	Solution to Task 3	172

Assignment 13	PEM fuel cell	175
13.1	System description	175
13.2	Outline of the assignment	178
13.3	Task 1	180
13.4	Solution to Task 1	184
13.5	Task 2	188
13.6	Solution to Task 2	199
13.7	Task 3	206
13.8	Solution to Task 3	219

Bibliography	247
---------------------	------------

Курс ООМ для инженеров

Общая информация



Форум



Ссылка на вебинарную комнату курса



<https://dl.spbstu.ru/course/view.php?id=2532>



Посещаемость

Лекции



Математическое моделирование сложных динамических систем: сборник заданий.



Колесов Ю.Б., Сениченков Ю.Б. Математическое моделирование сложных динамических систем : учебное пособие.



Компонентное моделирование сложных динамических систем : учебное пособие.



Компонентное моделирование сложных динамических систем : сборник заданий.



Modeling Simulation Engineering Using Modelica



Simulation Practice With Modelica



Презентация 1



Презентация 2



Презентация 3



Презентация _4

гибридные системы



Презентация 5

Modelica

Основы объектно-ориентированного моделирования Сениченков Юрий Борисович

[Личный кабинет](#) / [Мои курсы](#) / [cvd19-34226578152304310475](#) / [Лекции](#) / [Презентация 5](#)

Презентация 5

Modelica



-  [01 - Introduction Modelica.pdf](#)
-  [Modelica.pptx](#)
-  [ModelicaTutorial14.pdf](#)
-  [modelicatutorialfritzon.pdf](#)

Скачать папку

Редактировать

Лабораторные

 Лабораторные работы_1

 Задание_1

Тексты заданий в папке Лабораторная работа 1

Выполняется в OpenModelica и Anydynamics

 Лабораторная работа 3

AnyDynamics,OpenModelica

 Лабораторная работа 2

качественная теория на плоскости

 Задание 2

Дискретные системы

 Задание 3

задачи из журнала SNE

 Задание 4

Согласовать с преподавателем

 Лабораторная №5

 Задание 5

Отчет

Ла64_Основы_OOM

Lab4_Base_OOM

Ла64_основы_OOM

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО»

Институт компьютерных наук и технологий

Высшая школа программной инженерии



ПОЛИТЕХ

Санкт-Петербургский
политехнический университет
Петра Великого

Лабораторная работа №4

по дисциплине «Основы объектно-ориентированного моделирования»

I

Студент гр. 3540202/10201

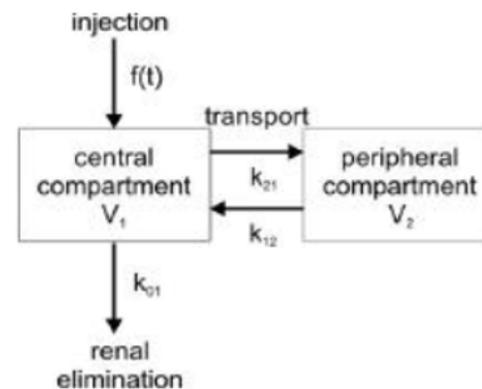
Ю.А. Пелёвина

Руководитель

Ю.Б. Сениченков

Постановка задачи

Дана следующая структура модели



Которую можно выразить следующей системой уравнений

$$dx_1/dt = f(t) - (k_{01} + k_{21})x_1 + k_{12}x_2 \quad (1)$$

$$dx_2/dt = k_{21}x_1 - k_{12}x_2 \quad (2)$$

$$f(t) = D/\tau, \quad 0 \leq t < \tau \quad (3)$$

$$c(t) = x_1(t) / V_1 \quad (4)$$

Задание а

Смоделировать систему со следующими параметрами $k_{01}=0.0041$, $k_{12}=0.0585$, $k_{21}=0.0498$, and $V_1=7.3$; $x_1(0) = x_2(0) = 0$. Сделать прогон системы на 240 единиц модельного времени для трёх случаев параметров для функции инъекции:

- $D_1 = 2500$, $\tau_1 = 0.5$ min
- $D_2 = 2500$, $\tau_2 = 3$ min
- $D_3 = 2500$, $\tau_3 = 240$ min

Решим задачу а

AnyDynamics

tau=0.5

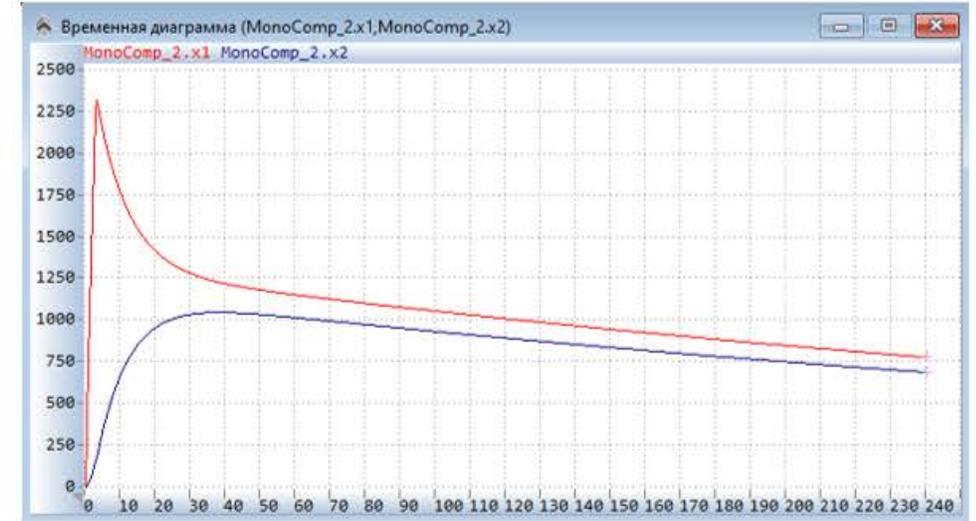
Однокомпонентный случай



Многокомпонентный случай



Однокомпонентный случай



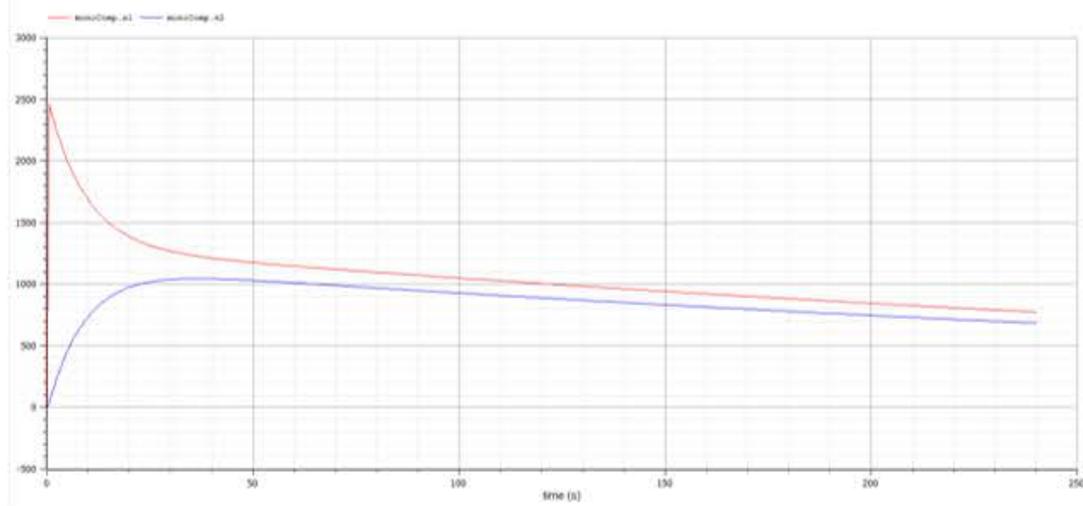
Многокомпонентный случай



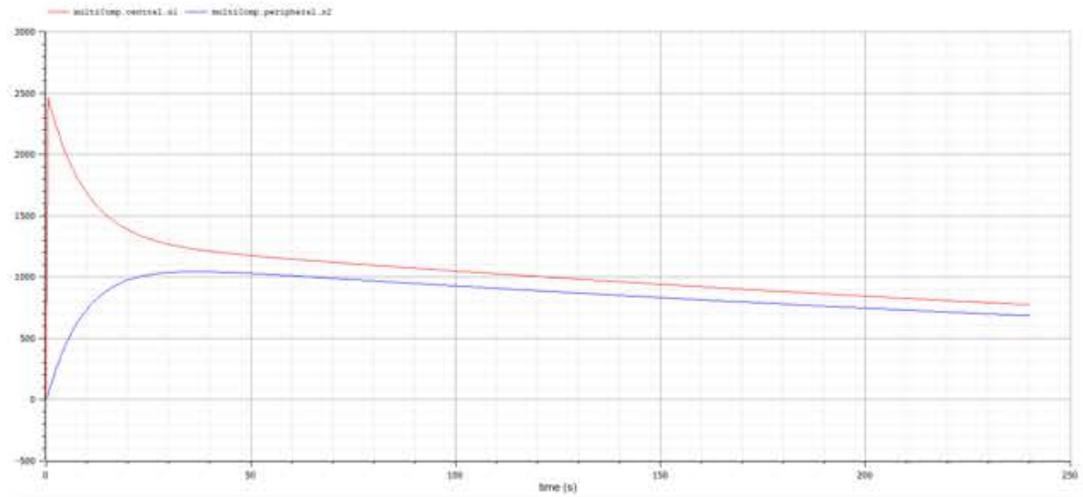
tau=240

Однокомпонентный случай

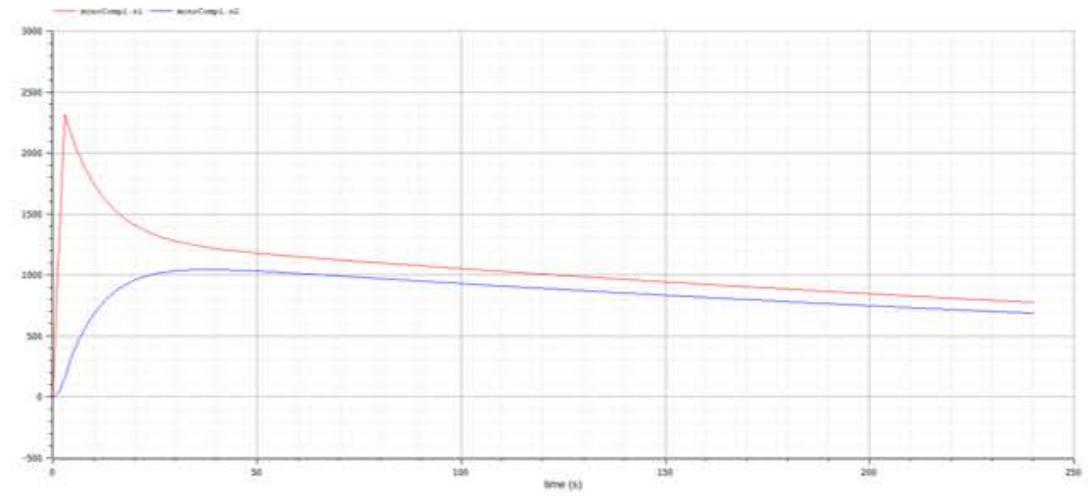
Однокомпонентный случай



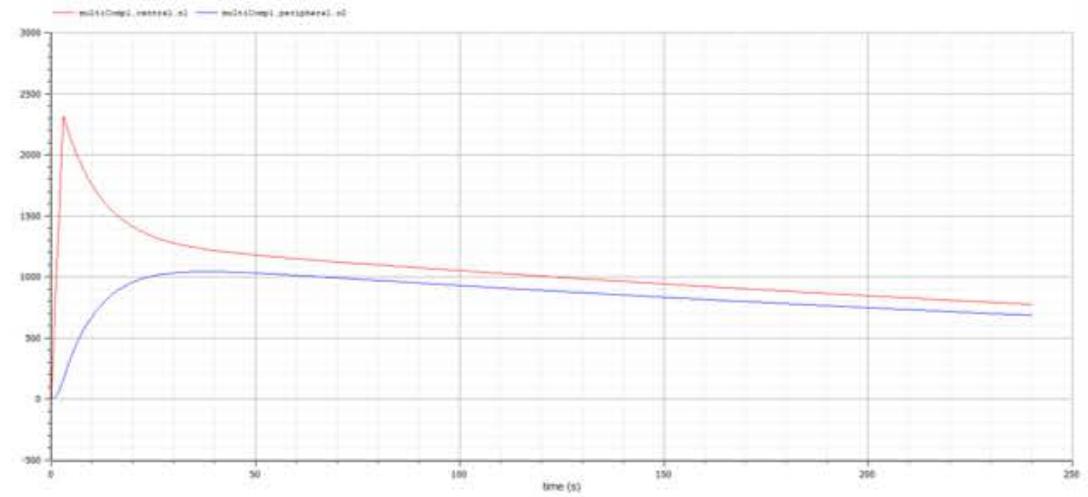
Многокомпонентный случай



$\tau=3$



Многокомпонентный случай



$\tau=240$

Однокомпонентный случай