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

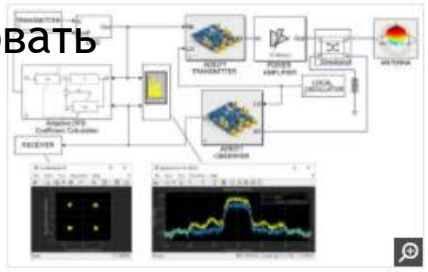
https://www.mssoft.ru/Makers/MathWorks/MATLAB/

- **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.

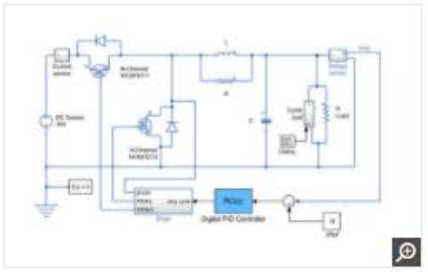
Matlab - Simulink

Кубики можно только использовать

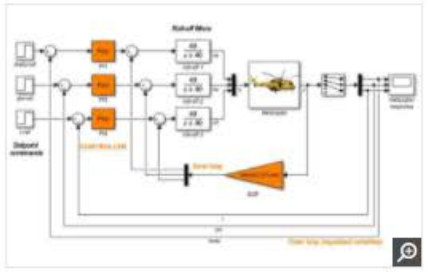
Simulink is for Every Project



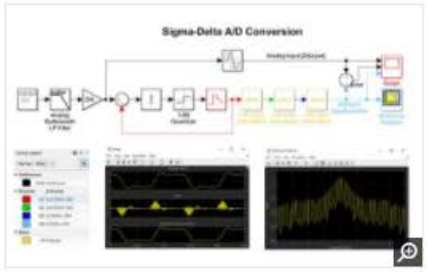
Wireless Communications



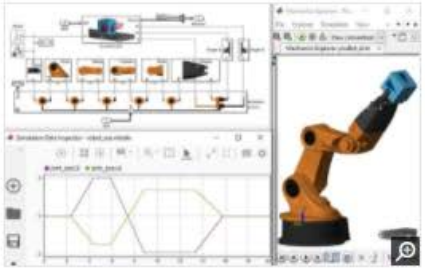
Power Electronics



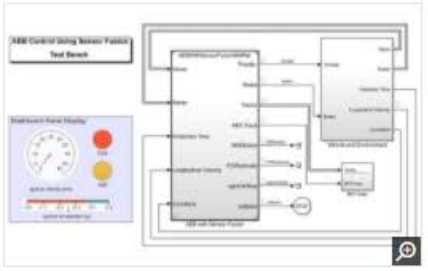
Control Systems



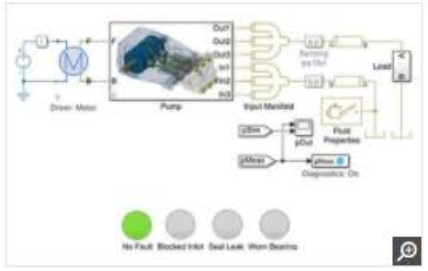
Signal Processing



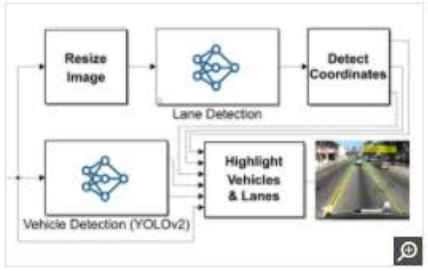
Autonomous Systems and Robotics



Advanced Driver Assistance Systems



Digital Twins

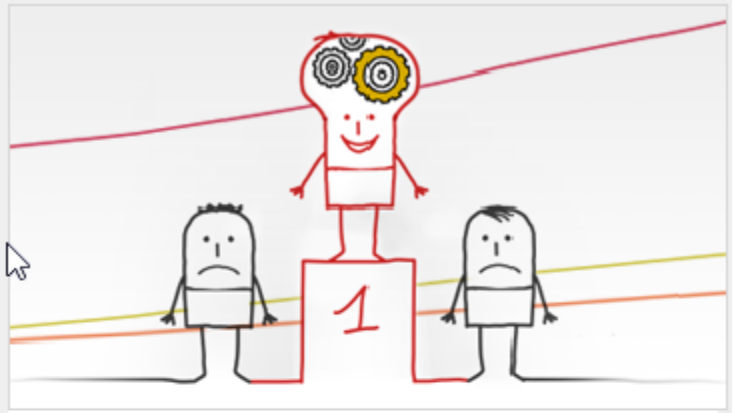


Artificial Intelligence

WOLFRAM
COMPUTATION MEETS KNOWLEDGE

WOLFRAM SYSTEM MODELER

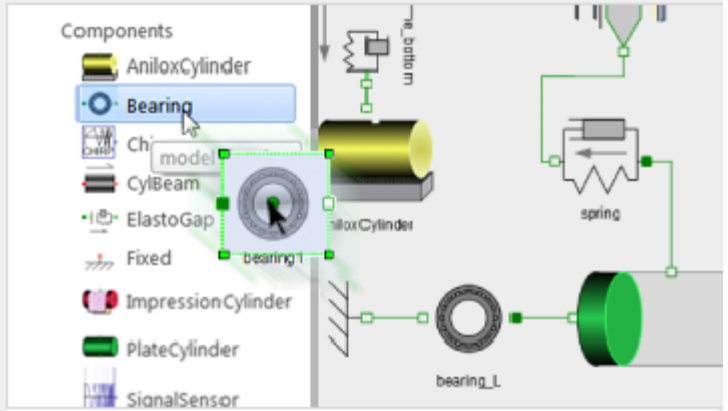
Движущая сила к пониманию, инновациям и результатам



Сравнение SystemModeler

Узнайте о преимуществах SystemModeler по сравнению с конкурентами.

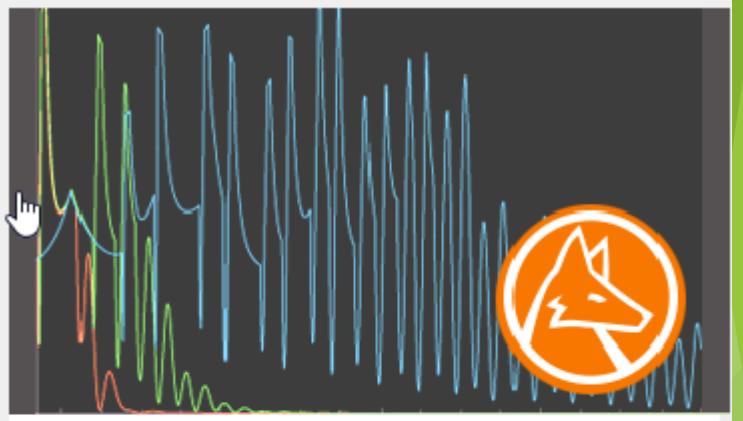
Wolfram SystemModeler является наиболее полной средой для моделирования физических систем. В отличие от других сред моделирования, SystemModeler не требует для работы установки никаких дополнительных модулей и полностью поддерживает язык Modelica. SystemModeler, также, поддерживает работу с языком Wolfram 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

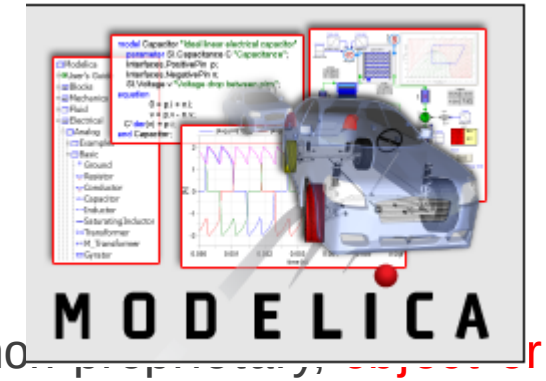
- Промышленность – что хотите, лишь бы Вам было хорошо.
- Образование – то, что позволит научить студента осознанно выбирать нужный инструмент для решения конкретной задачи.

  - 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.

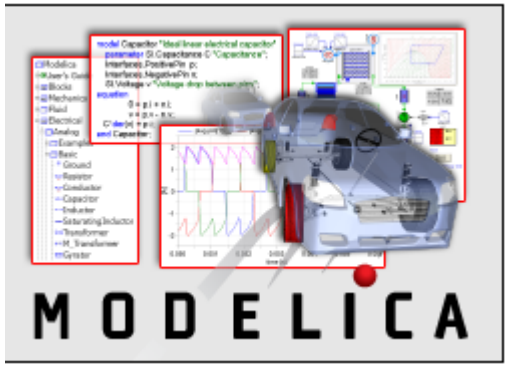
  - ООМ,
  - Математические пакеты и библиотеки,
  - Среды моделирования,
  - В рамках отведенных на это часов !!!!!!

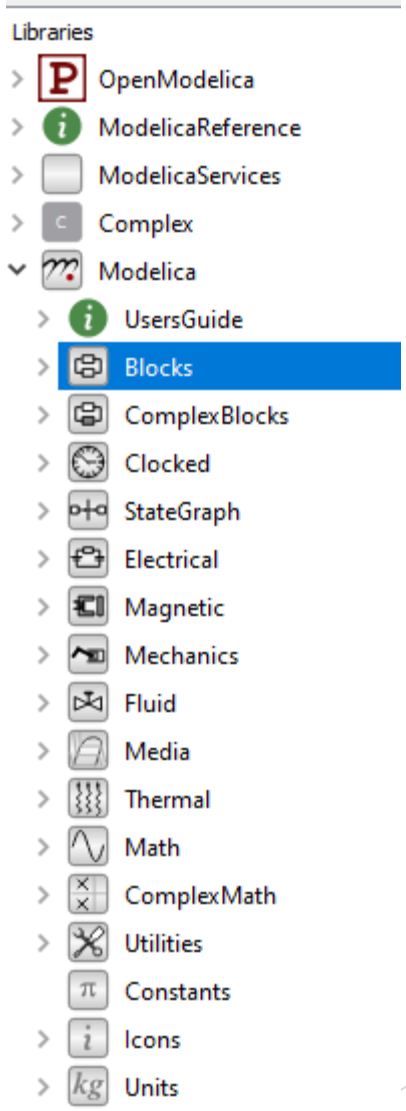


- The **Modelica Language** is a no[n-]p[roprietary, object-o]riented, equation based conveniently model complex physical systems containing, e.g., mechanical, electrical, electronic, hydraulic, thermal, control, electric power or process-oriented subcomponents. See also, overview in pdf, ppt format 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!



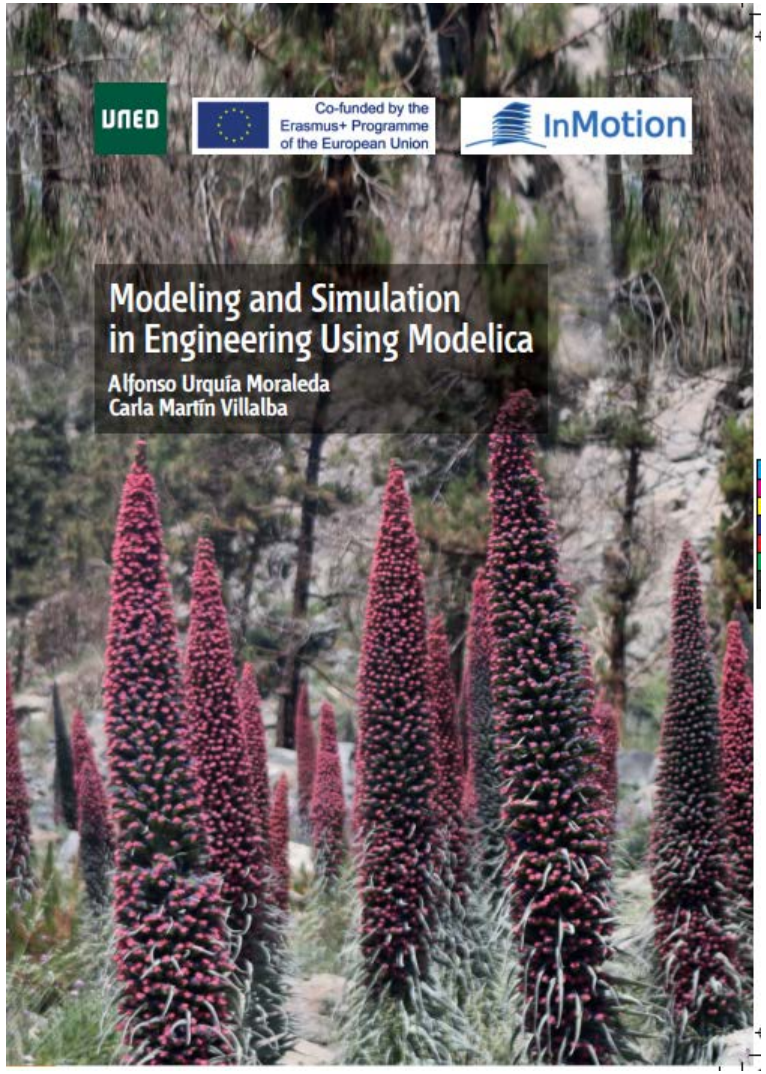


Инкапсуляция
Наследование
Полиморфизм
Пакеты

OpenModelica Matlab interface https://www.openmodelica.org/doc/OpenModelicaUsersGuide/latest/ommatlab.html

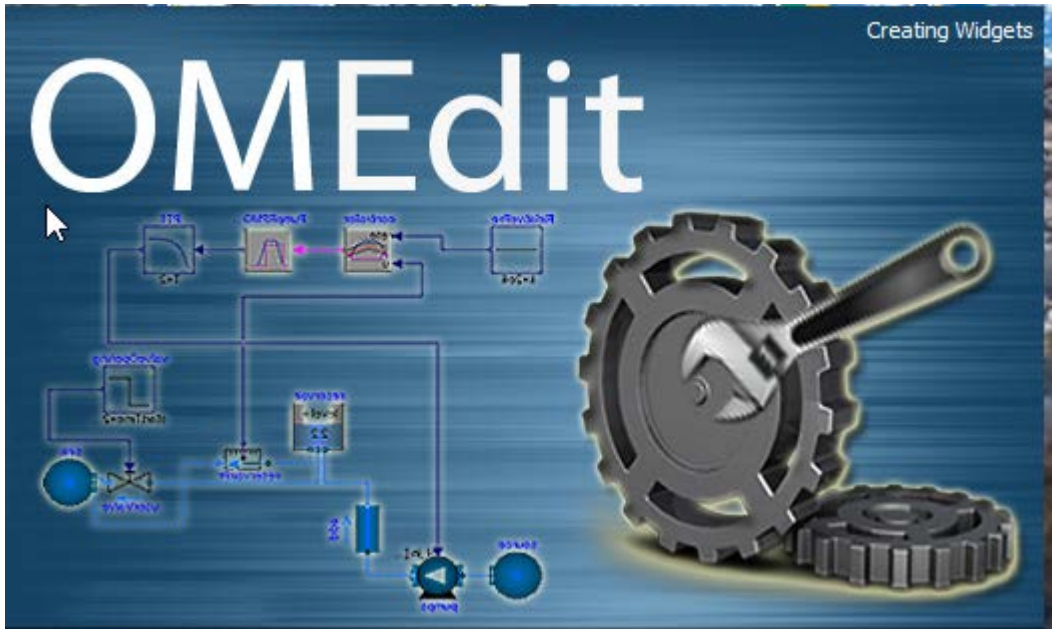
- OMMatlab is architectured to combine both the solving strategy and model building. So domain experts (people writing the models) and computational engineers (people writing the solver code) can work on one unified tool that is industrially viable for optimization of Modelica models, while offering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

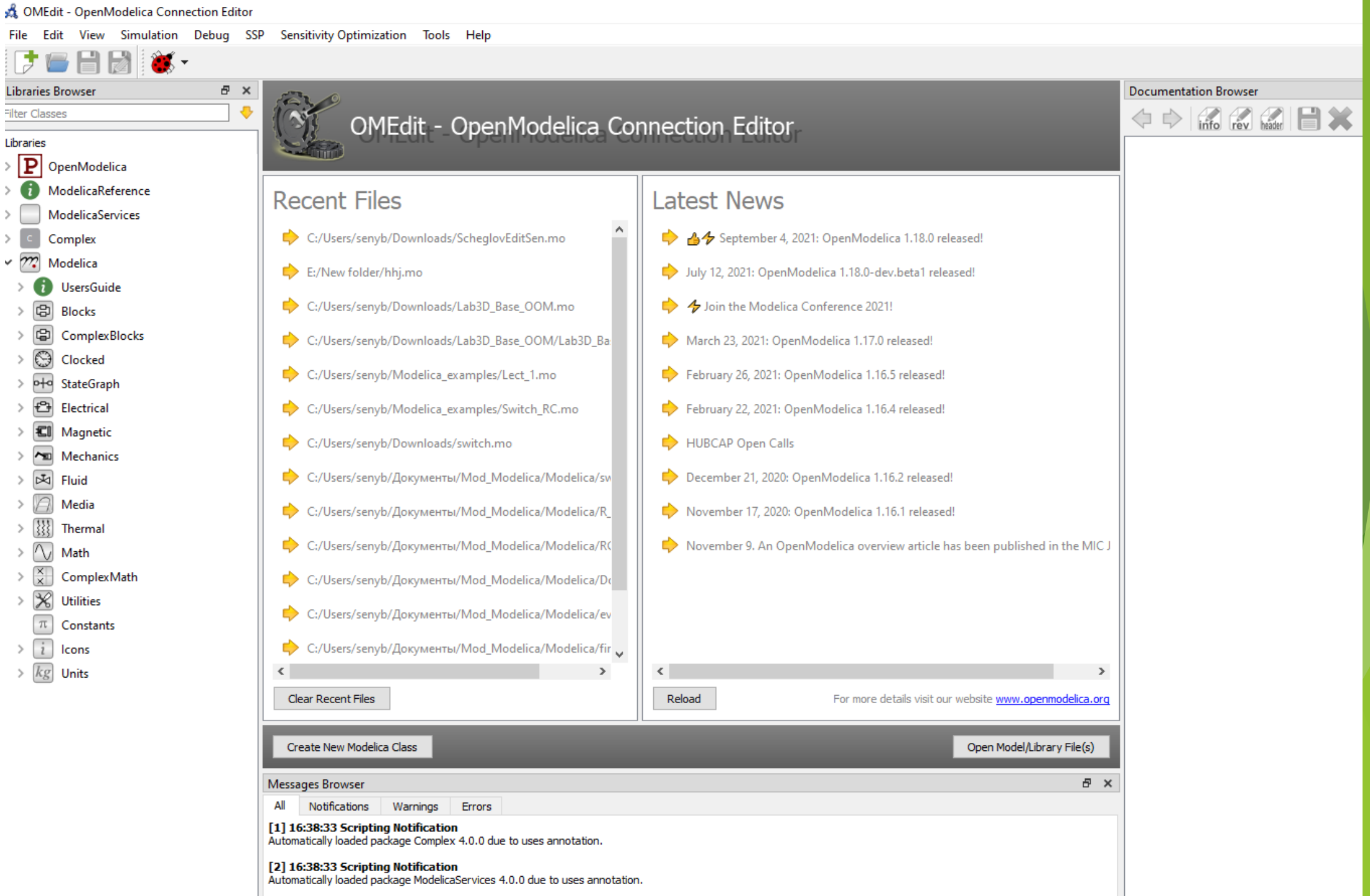
Книги проекта InMotion

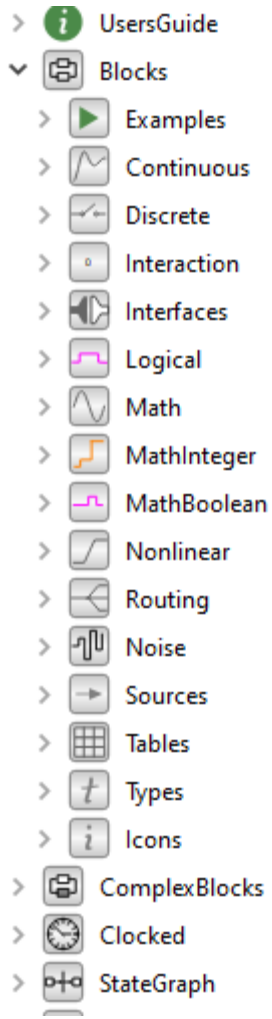


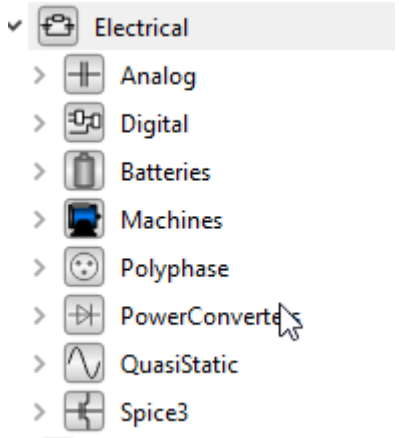
Contents

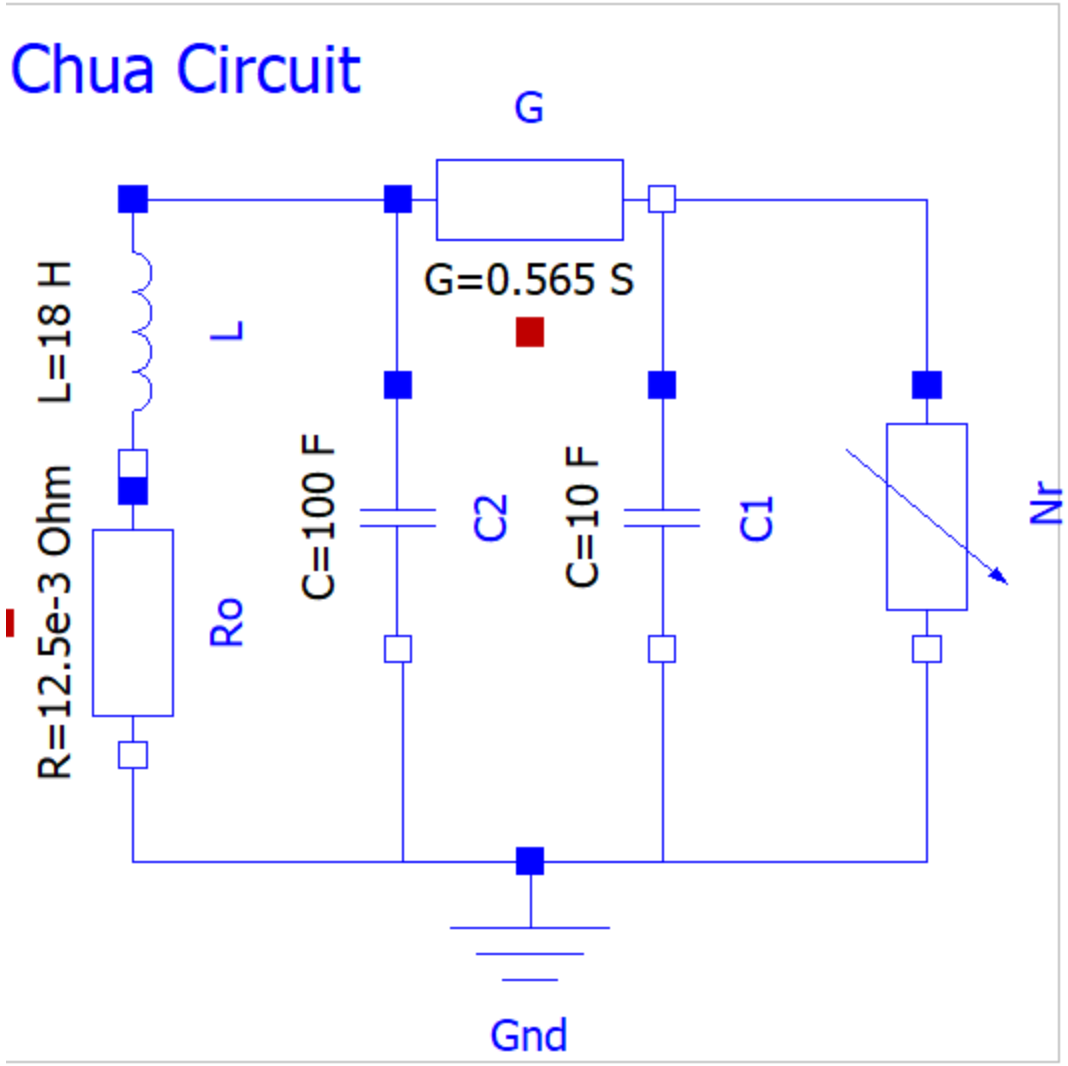
https://www.openmodelica.org/

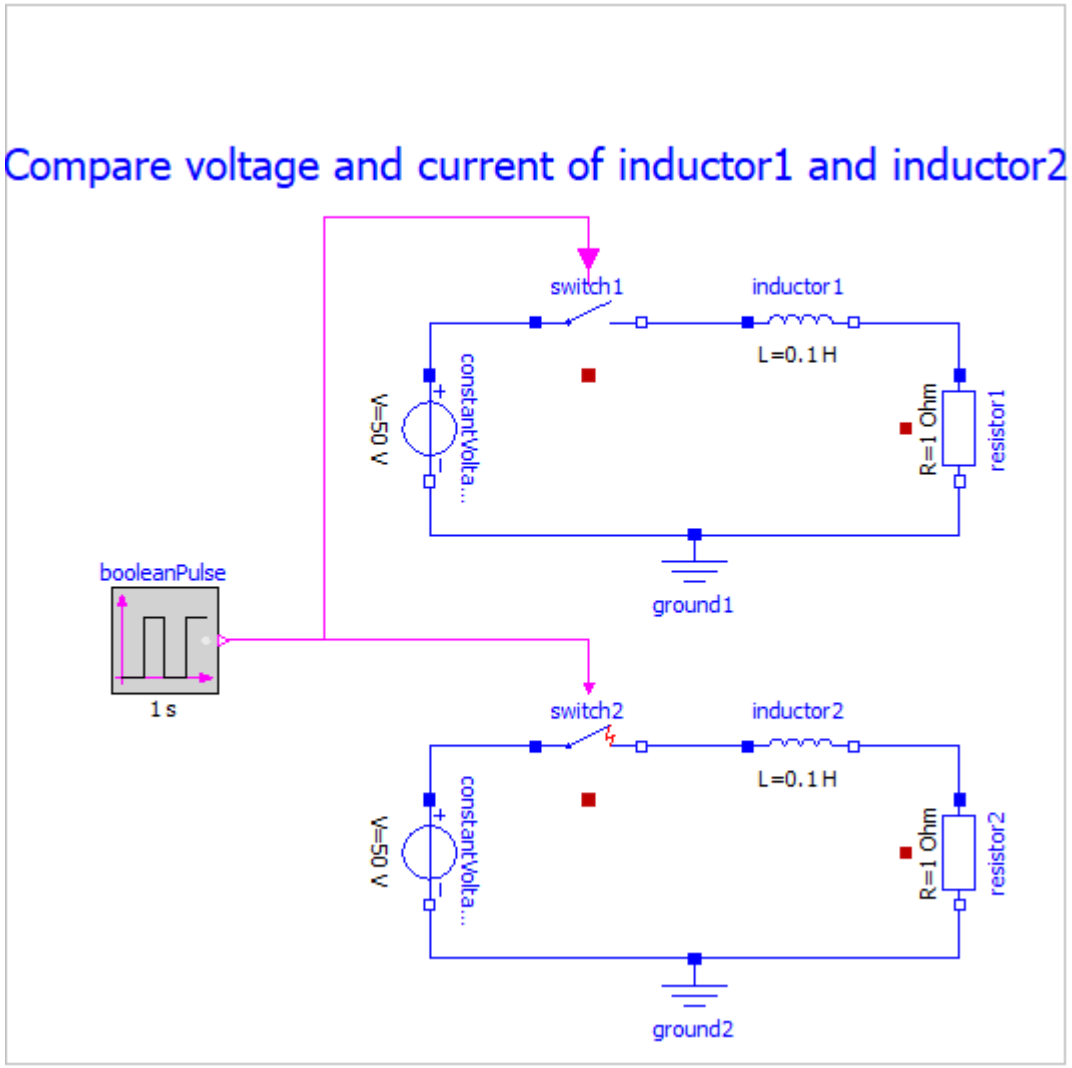


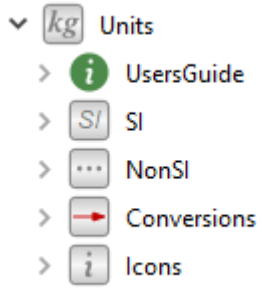

OMEdit - OpenModelica Connection Editor
File Edit View Simulation Debug SSP Sensitivity Optimization Tools Help
Libraries Browser
Filter Classes
Libraries
OpenModelica
ModelicaReference
ModelicaServices
Complex
Modelica
UsersGuide
Blocks
ComplexBlocks
Clocked
StateGraph
Electrical
Magnetic
Mechanics
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
info rev header
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.

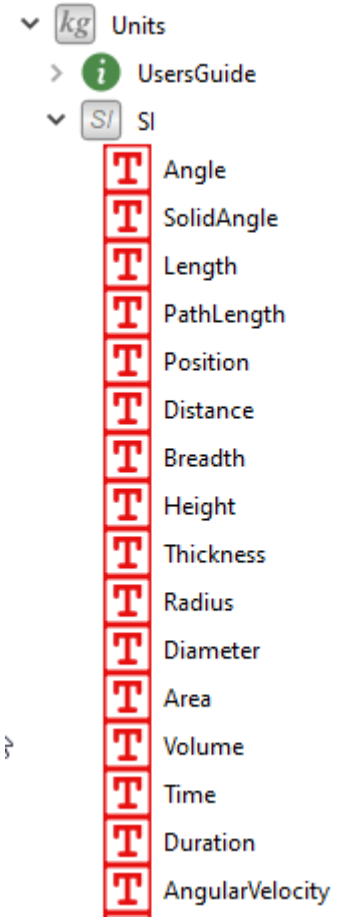

UsersGuide
Blocks
Examples
Continuous
Discrete
Interaction
Interfaces
Logical
Math
MathInteger
MathBoolean
Nonlinear
Routing
Noise
Sources
Tables
Types
Icons
ComplexBlocks
Clocked
StateGraph

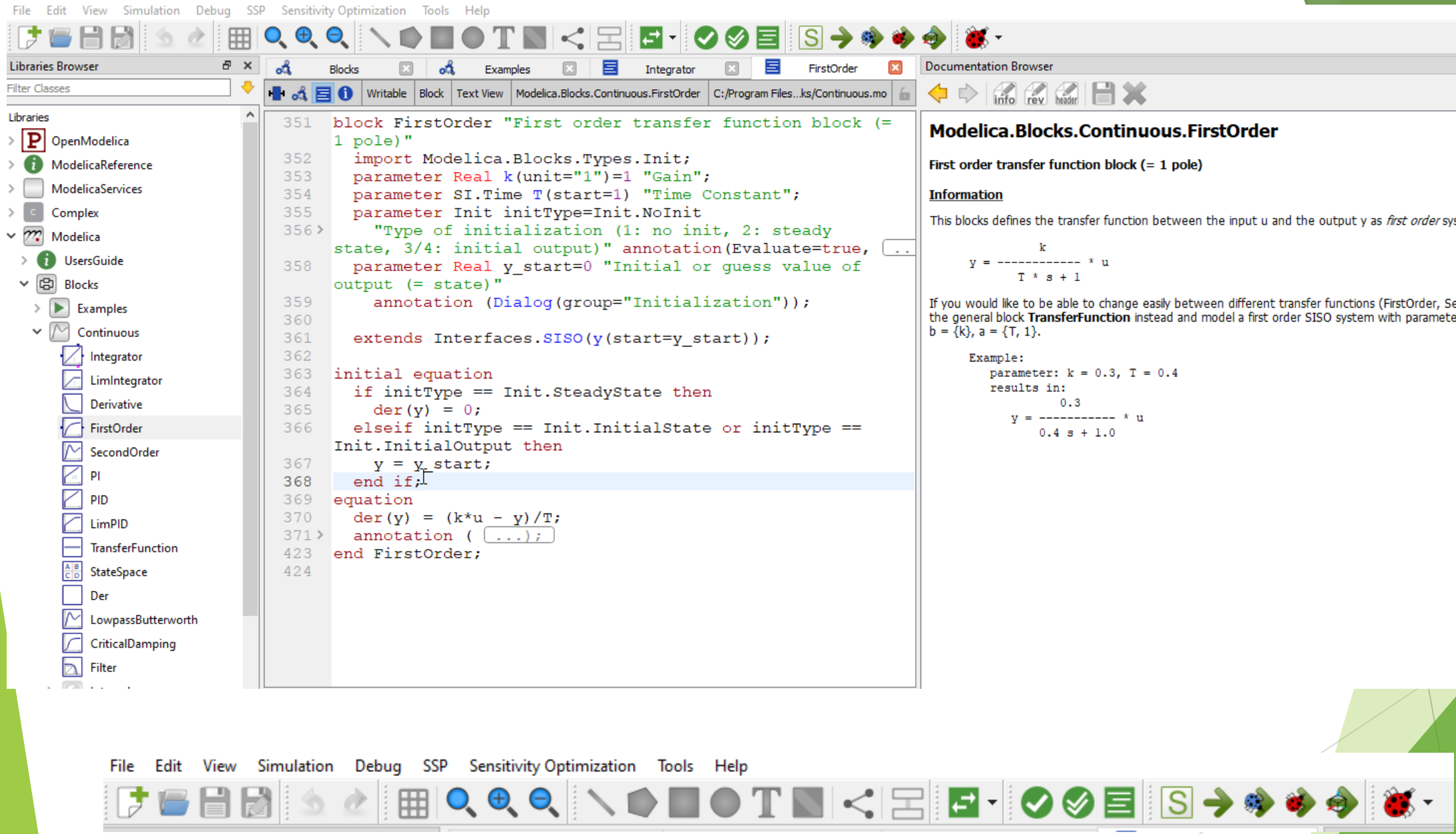

Electrical
Analog
Digital
Batteries
Machines
Polyphase
PowerConverters
QuasiStatic
Spice3


Chua Circuit
G
G=0.565 S
L
L=18 H
R=12.5e-3 Ohm
Ro
C=100 F
C2
C=10 F
C1
Nr
Gnd

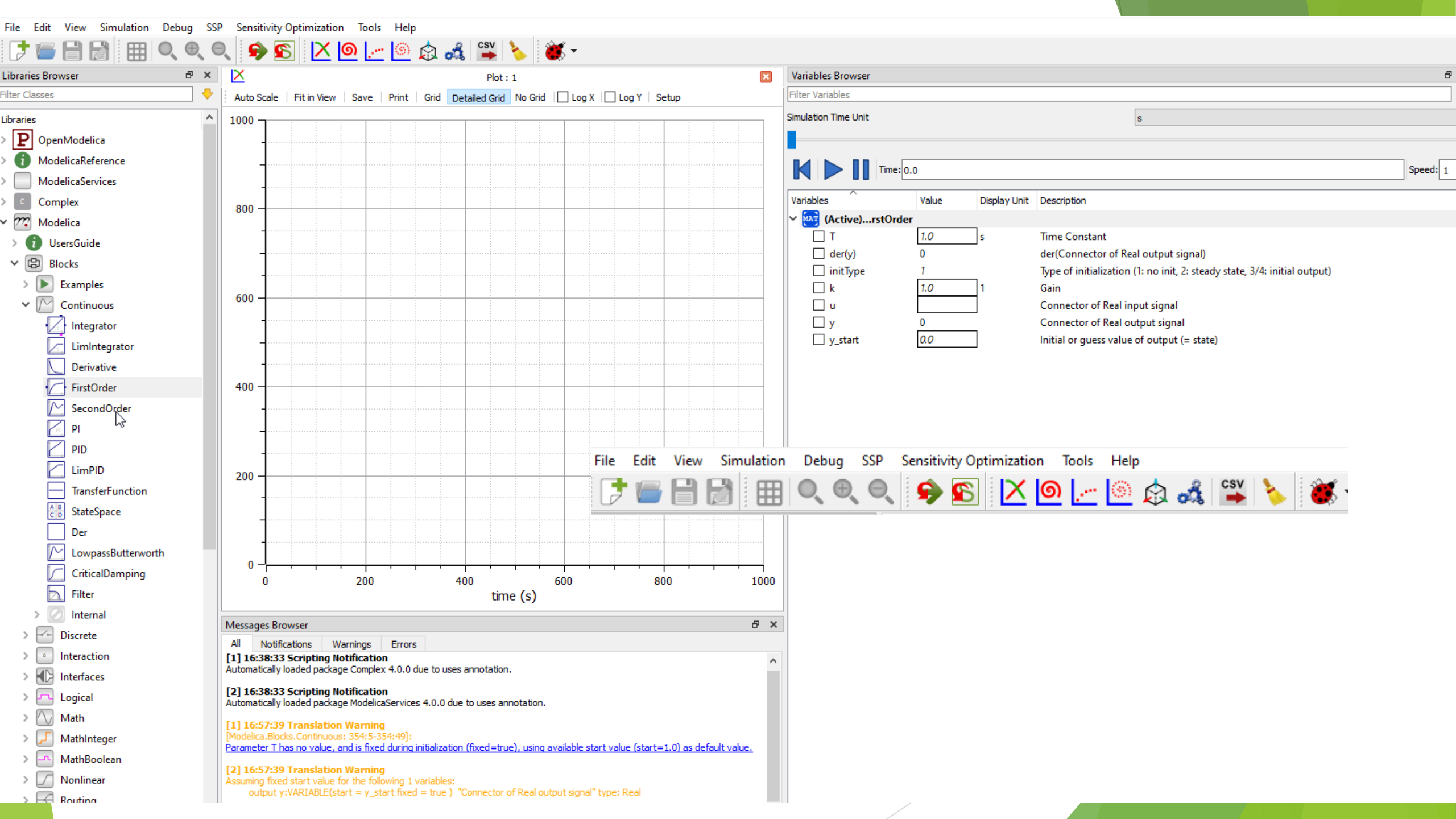

Compare voltage and current of inductor1 and inductor2
switch1
inductor1
L=0.1 H
V=50 V
constantVolta...
R=1 Ohm
resistor1
booleanPulse
1 s
ground1
switch2
inductor2
L=0.1 H
V=50 V
constantVolta...
R=1 Ohm
resistor2
ground2

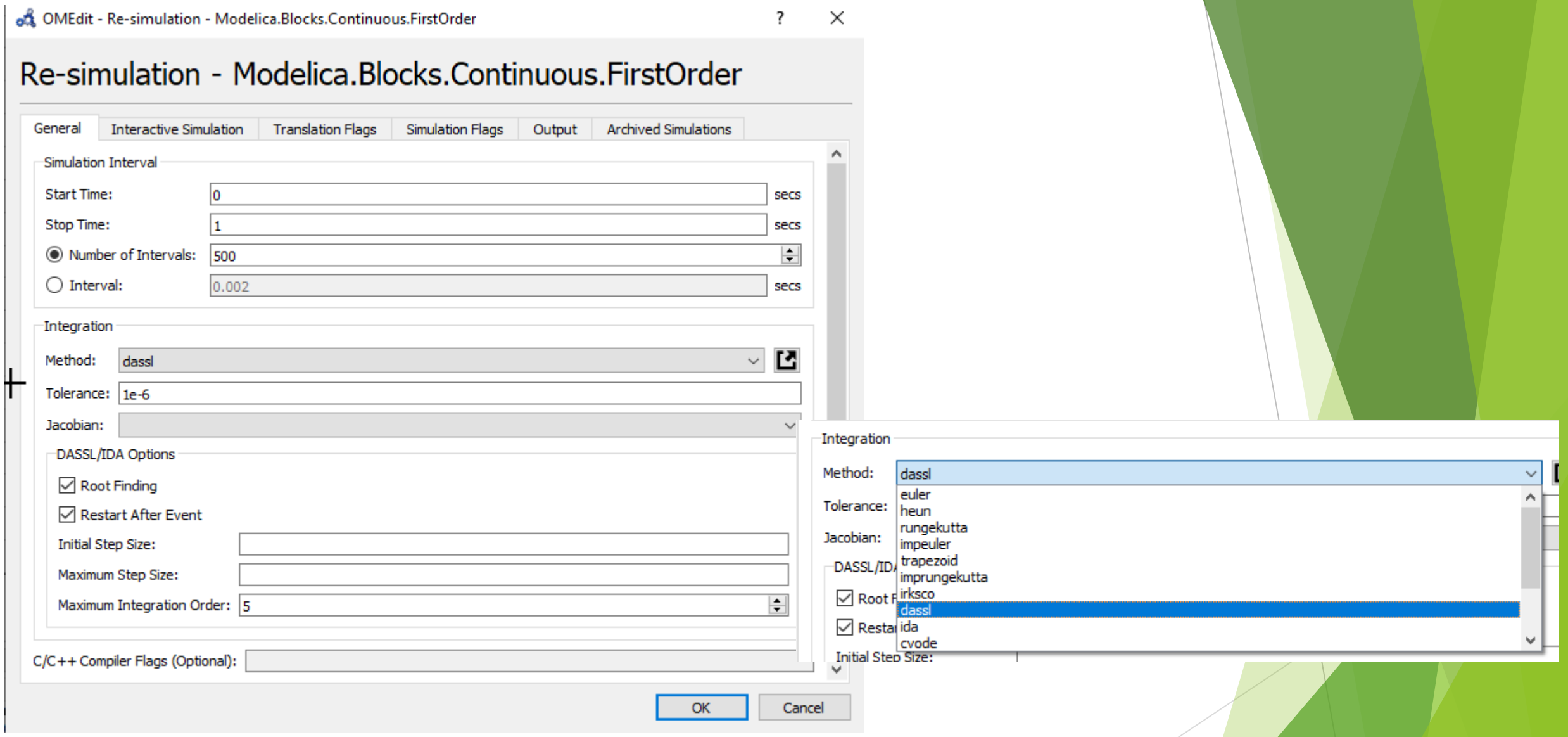

Units
UsersGuide
SI
NonSI
Conversions
Icons


Units
UsersGuide
SI
Angle
SolidAngle
Length
PathLength
Position
Distance
Breadth
Height
Thickness
Radius
Diameter
Area
Volume
Time
Duration
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
Blocks
Examples
Integrator
FirstOrder
Writable
Block
Text View
Modelica.Blocks.Continuous.FirstOrder
C:/Program Files...ks/Continuous.mo
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
info
rev
header
Modelica.Blocks.Continuous.FirstOrder
First order transfer function block (= 1 pole)
Information
This blocks defines the transfer function between the input u and the output y as first order system:
k
y = ------------ * u
T * s + 1
If you would like to be able to change easily between different transfer functions (FirstOrder, SecondOrder, ...) by changing parameters, us
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:
0.3
y = ----------- * u
0.4 s + 1.0

File Edit View Simulation Debug SSP Sensitivity Optimization Tools Help


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
1000
800
600
400
200
0
0 200 400 600 800 1000
time (s)
Variables Browser
Filter Variables
Simulation Time Unit
s
Time: 0.0
Speed: 1
Variables Value Display Unit Description
(Active)...rstOrder
T 1.0 s Time Constant
der(y) 0 der(Connector of Real output signal)
initType 1 Type of initialization (1: no init, 2: steady state, 3/4: initial output)
k 1.0 1 Gain
u Connector of Real input signal
y 0 Connector of Real output signal
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
File Edit View Simulation Debug SSP Sensitivity Optimization Tools Help


OMEdit - Re-simulation - Modelica.Blocks.Continuous.FirstOrder
Re-simulation - Modelica.Blocks.Continuous.FirstOrder
General
Interactive Simulation
Translation Flags
Simulation Flags
Output
Archived Simulations
Simulation Interval
Start Time: 0 secs
Stop Time: 1 secs
Number of Intervals: 500
Interval: 0.002 secs
Integration
Method: dassl
Tolerance: 1e-6
Jacobian:
DASSL/IDA Options
Root Finding
Restart After Event
Initial Step Size:
Maximum Step Size:
Maximum Integration Order: 5
C/C++ Compiler Flags (Optional):
OK
Cancel
Integration
Method: dassl
Tolerance:
Jacobian:
Initial Step Size:
euler
heun
rungekutta
impeuler
trapezoid
imprungekutta
irksco
dassl
ida
cvode

II Simulation of continuous-time models

III Hybrid system modeling and simulation

- StateGraph
  - UsersGuide
  - Examples
  - Interfaces
  - InitialStep
  - InitialStepWithSignal
  - Step
  - StepWithSignal
  - Transition
  - TransitionWithSignal
  - Alternative
  - Parallel
  - PartialCompositeStep
  - StateGraphRoot

Modelica.StateGraph

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

Information

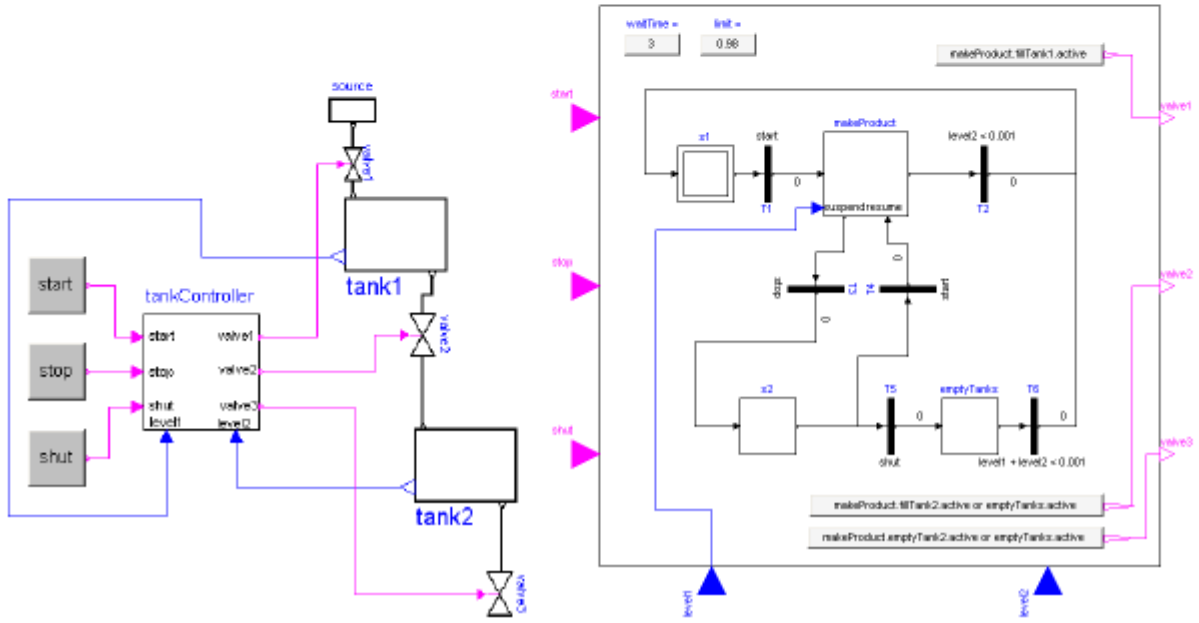
Note, there is a much improved version of this library called "Modelica_StateGraph2". If this library is not yet distributed with your Modelica tool, you can download it from https://github.com/modelica/Modelica_StateGraph2. In the Users Guide a detailed comparison is given. It is highly recommended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

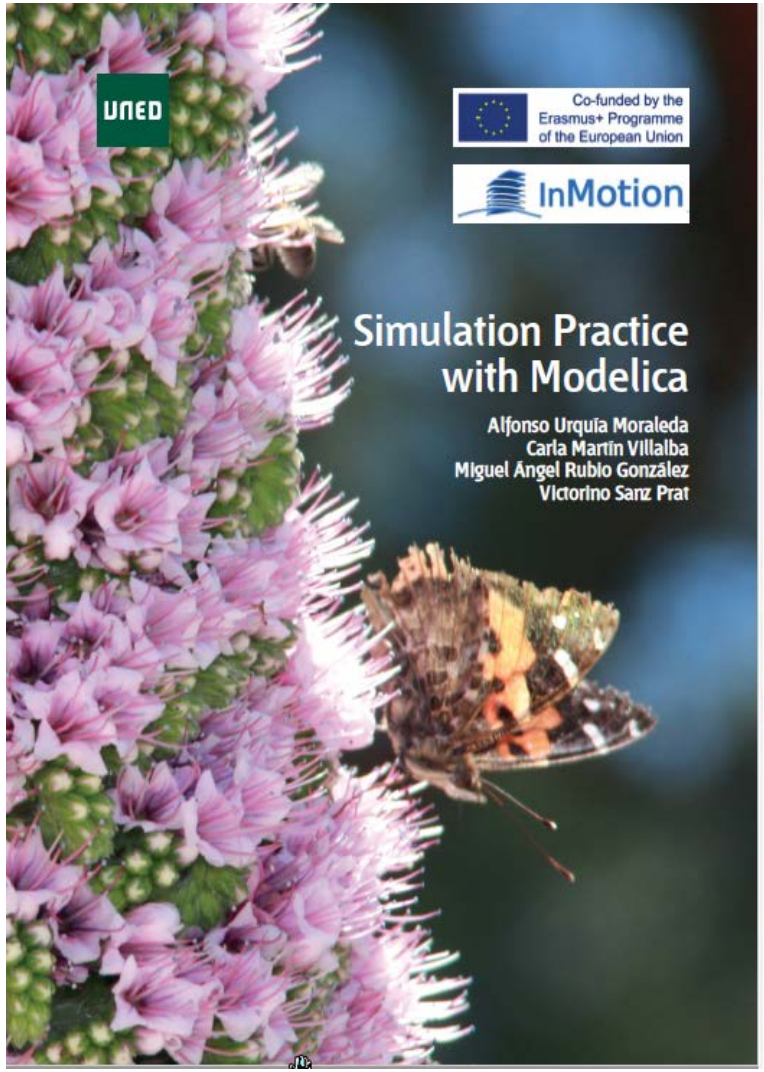
For an introduction, have especially a look at:

- 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:

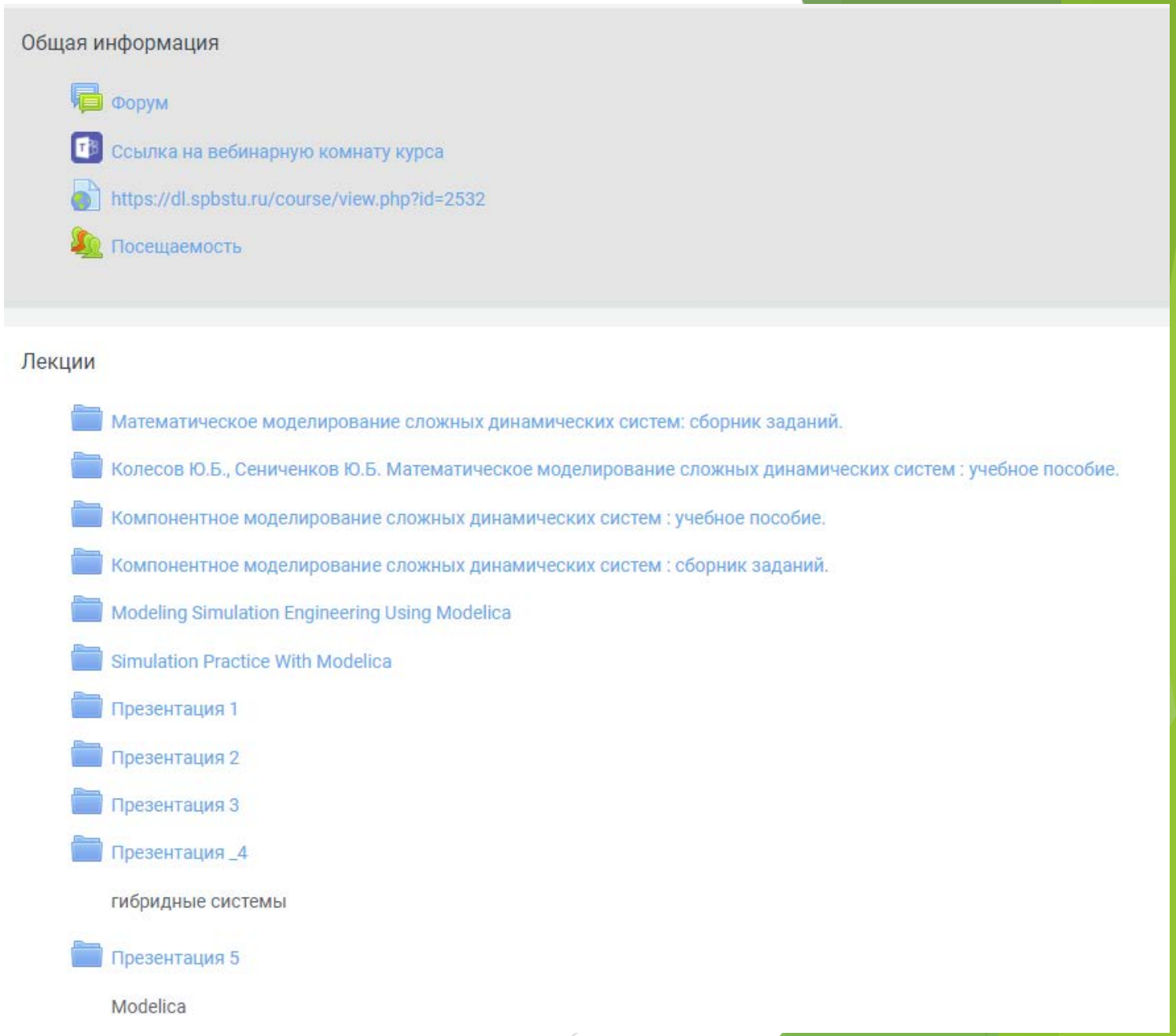


Книги проекта InMotion



Contents

Курс ООМ для инженеров



Основы объектно-ориентированного моделирования Сениченков Юрий Борисович

Презентация 5

Modelica

- 01 - Introduction Modelica.pdf
- Modelica.pptx
- ModelicaTutorial14.pdf
- modelicatutorialfritzson.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

Лаб4_Основы_ООМ
Lab4_Base_ООМ
Лаб4_основы_ООМ

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

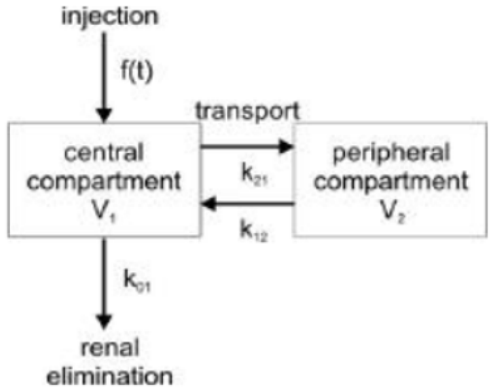
по дисциплине «Основы объектно-ориентированного моделирования»

Студент гр. *3540202/10201* *Ю.А. Пелёвина*

Руководитель Ю.Б. Сениченков

Постановка задачи

Дана следующая структура модели



Которую можно выразить следующей системой уравнений

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

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

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

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

Задание а

Смоделировать систему со следующими параметрами k01=0.0041, k12=0.0585, k21=0.0498, and V1=7.3; x1(0) = x2(0) = 0. Сделать прогон системы на 240 единиц модельного времени для трёх случаев параметров для функции инъекции:

- D1 = 2500, τ1 = 0.5 min
- D2 = 2500, τ2 = 3 min,
- D3 = 2500, τ3 = 240 min.

Решим задачу а

AnyDynamics

tau=0.5

Однокомпонентный случай

Многокомпонентный случай

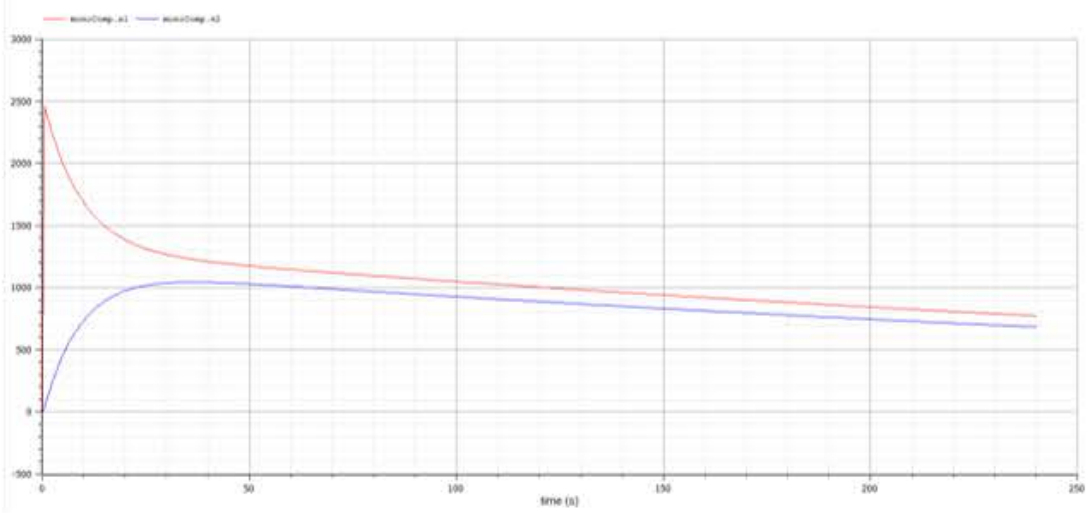
Однокомпонентный случай

Многокомпонентный случай

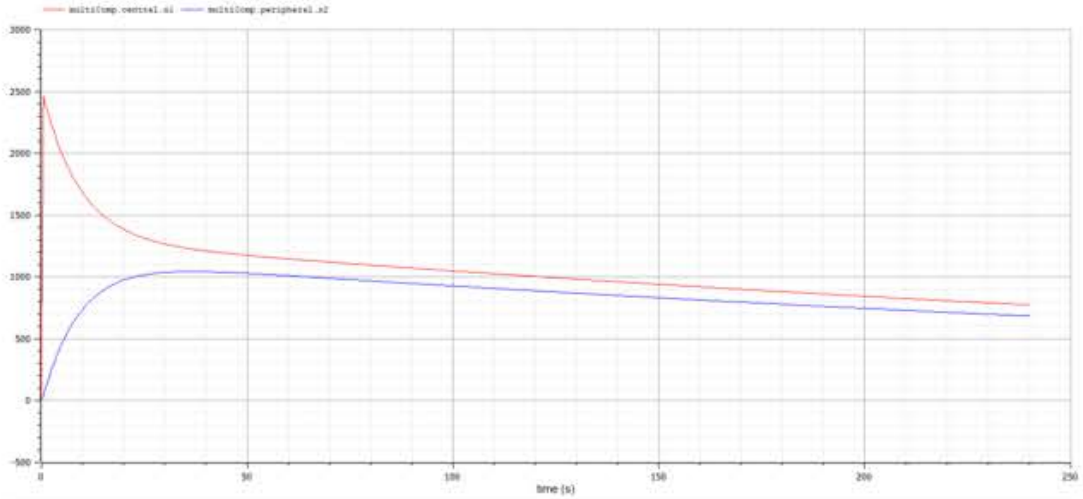
tau=240

Однокомпонентный случай

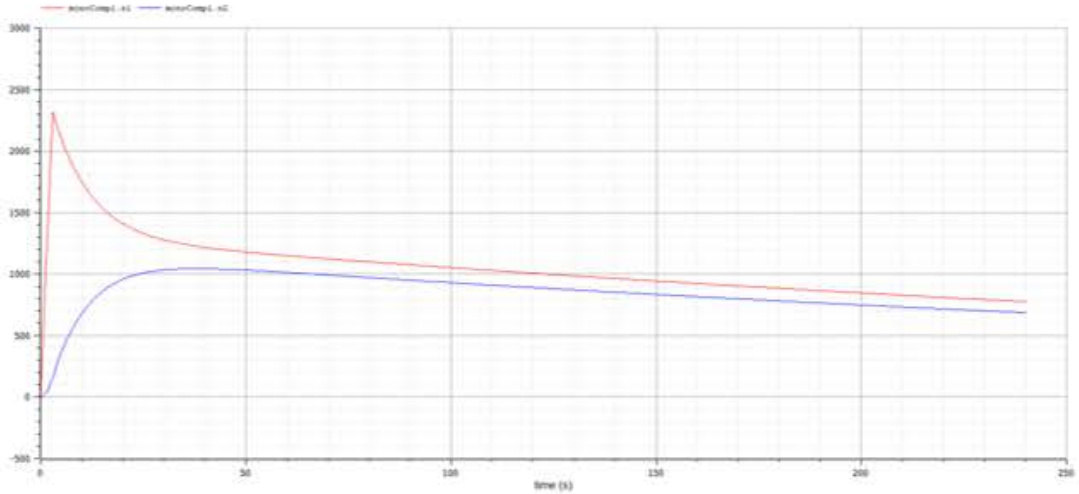
Однокомпонентный случай



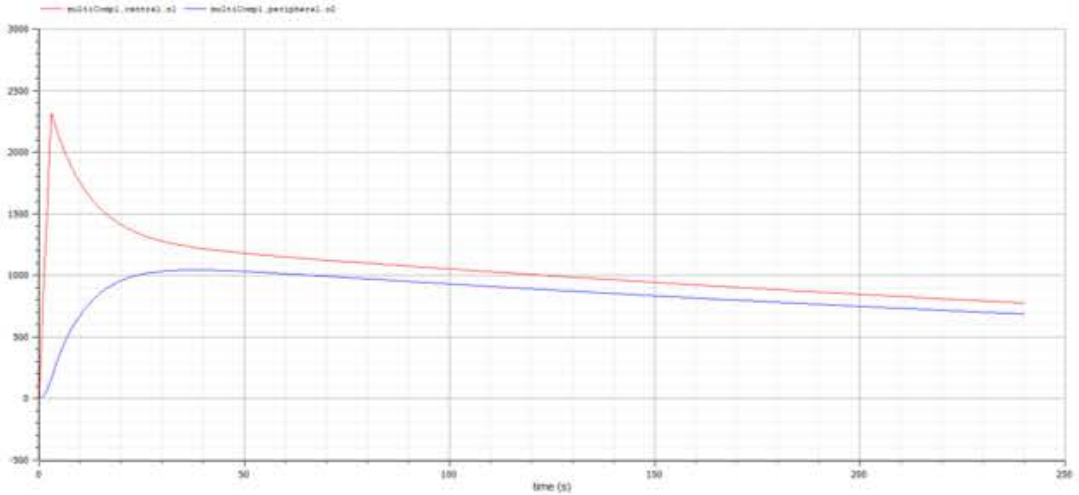
Многокомпонентный случай



tau=3



Многокомпонентный случай



tau=240

Однокомпонентный случай