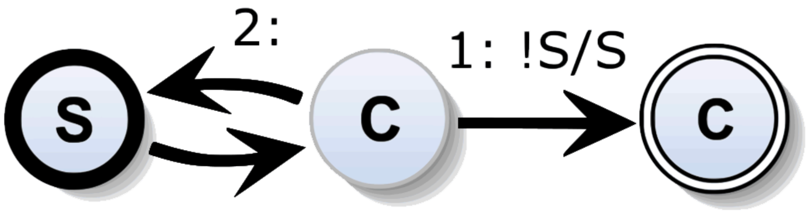


SCCharts Tutorial

SYNCHRON'16, Bamberg

KIELER SCCharts

C | A | U
Christian-Albrechts-Universität zu Kiel
Technische Fakultät

SEIT 350 JAHREN GANZ WEIT OBEN

1 Tutorial Preparations

Welcome to the SCCharts Tutorial! To start, please copy the Eclipse-based KIELER SCCharts application from the provided USB thumb drive to your hard drive.

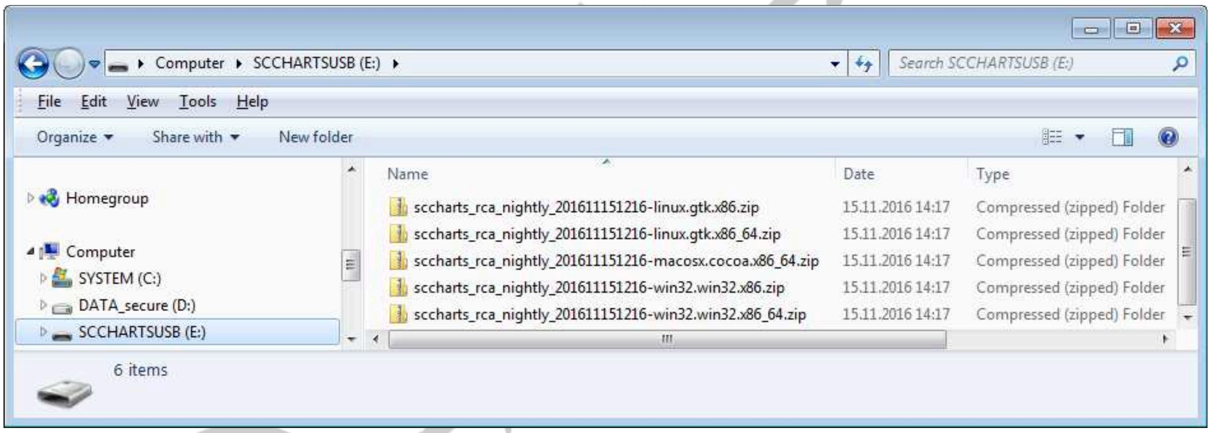
Additionally, you will need a **Java 8 SDK** on your machine. For Windows and the Lego tutorial part you will need the **32bit** version of Java and KIELER SCCharts. You will also find Java 8 installation files on the USB thumb drive in case you need to install it.

The following section, Section 1.1a, describes how to copy the appropriate files using a concrete example build. Note that the USB stick will contain the most recent build instead.
Alternatively, you read this tutorial documentation after the workshop and thus do not have a USB drive with the installation, then you can download the proper release candidate or the nightly RCA as explained in Section 1.1b.

1.1a Copy from USB Stick

(a) Plugin the provided USB stick: **/USBStick/KIELERSCCharts/**

(b) Choose a version suited for your OS and copy it to your computer.



Name	Date	Type
sccharts_rca_nightly_201611151216-linux.gtk.x86.zip	15.11.2016 14:17	Compressed (zipped) Folder
sccharts_rca_nightly_201611151216-linux.gtk.x86_64.zip	15.11.2016 14:17	Compressed (zipped) Folder
sccharts_rca_nightly_201611151216-macosx.cocoa.x86_64.zip	15.11.2016 14:17	Compressed (zipped) Folder
sccharts_rca_nightly_201611151216-win32.win32.x86.zip	15.11.2016 14:17	Compressed (zipped) Folder
sccharts_rca_nightly_201611151216-win32.win32.x86_64.zip	15.11.2016 14:17	Compressed (zipped) Folder

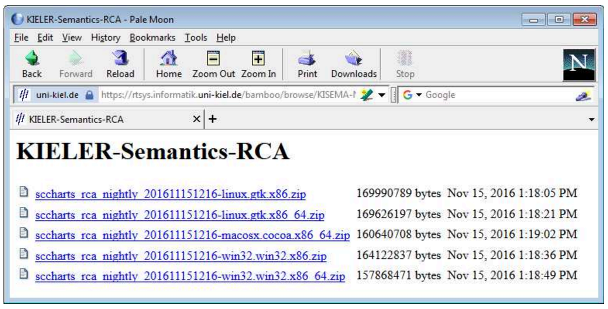
1.1b Download

(a) Use the following URL:

http://rtsys.informatik.uni-kiel.de/~kieler/files/release_sccharts_0.12.0/

or http://tinyurl.com/scchartsrelease12

(b) Choose a version suited for your OS and download it to your computer.


KIELER-Semantics-RCA - Pale Moon
File Edit View History Bookmarks Tools Help
Back Forward Reload Home Zoom Out Zoom In Print Downloads Stop
uni-kiel.de https://rtsys.informatik.uni-kiel.de/bamboo/browse/KISEMA-
Google
KIELER-Semantics-RCA

KIELER-Semantics-RCA

sccharts_rca_nightly_201611151216-linux.gtk.x86.zip	169990789 bytes	Nov 15, 2016 1:18:05 PM
sccharts_rca_nightly_201611151216-linux.gtk.x86_64.zip	169626197 bytes	Nov 15, 2016 1:18:21 PM
sccharts_rca_nightly_201611151216-macosx.cocoa.x86_64.zip	160640708 bytes	Nov 15, 2016 1:19:02 PM
sccharts_rca_nightly_201611151216-win32.win32.x86.zip	164122837 bytes	Nov 15, 2016 1:18:36 PM
sccharts_rca_nightly_201611151216-win32.win32.x86_64.zip	157868471 bytes	Nov 15, 2016 1:18:49 PM

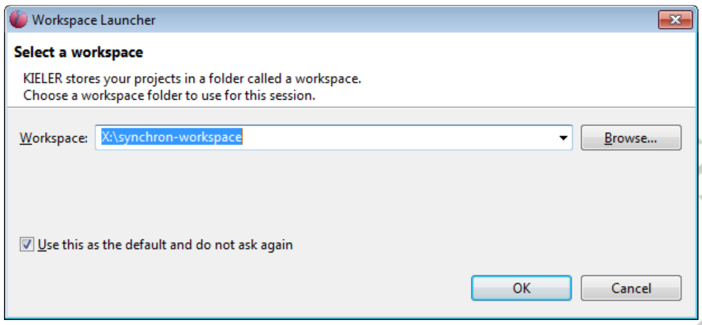
1.2 Extract / Unzip

Untar or unzip the KIELER SCCharts Tools in a folder of your harddrive.

1.3 Start KIELER SCCharts Tools

After you have copied and extracted the KIELER SCCharts Tools, you can start the KIELER.exe or KIELER.app. For Mac, you may need to set your security settings such that non-appstore applications are allowed to be started.

When you start the KIELER SCCharts Tools for the first time, choose an appropriate workspace path on your harddrive. You might later want to copy files there, so remember the path. For convenience, you may check the "[x] Use this as the default and do not ask again" option.



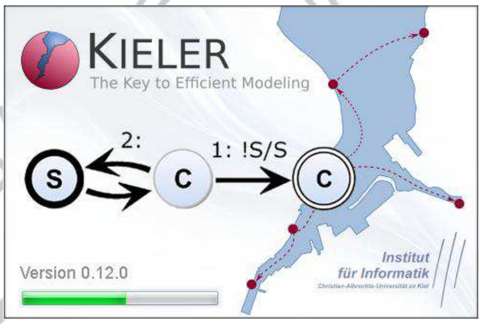
Workspace Launcher

Select a workspace

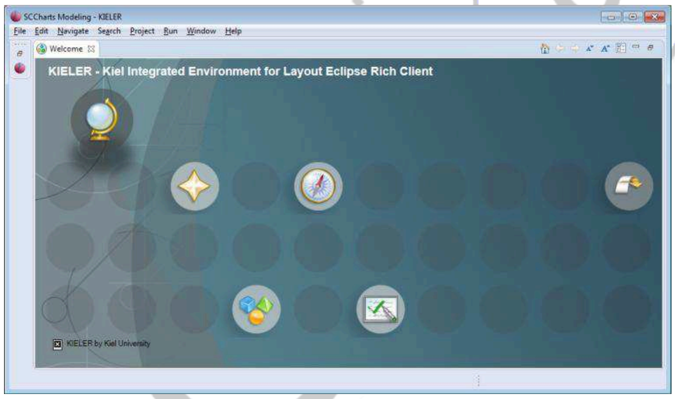
KIELER stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

Workspace: X:\synchron-workspace

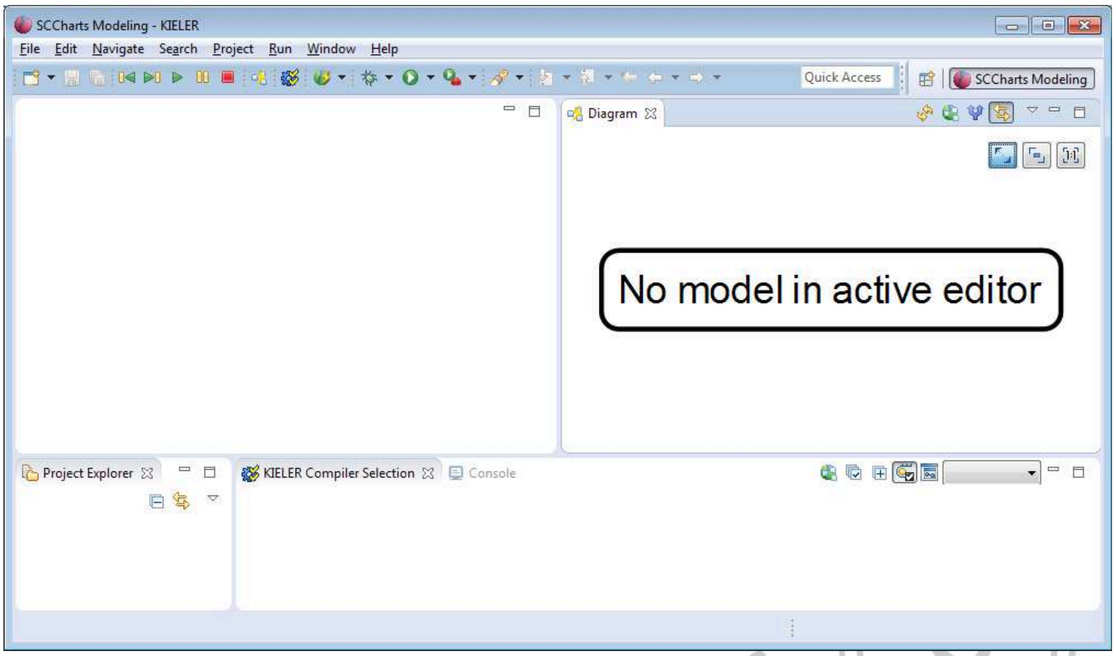
[✓] Use this as the default and do not ask again



After that, KIELER should start and you should see the splash screen (see above). Then you should see the following welcome screen:


SCCharts Modeling - KIELER
File Edit Navigate Search Project Run Window Help
Welcome
KIELER - Kiel Integrated Environment for Layout Eclipse Rich Client
KIELER by Kiel University

Close this tab. This will bring you to the KIELER SCCharts modeling perspective that should look similar to the following screenshot:


SCCharts Modeling - KIELER
File Edit Navigate Search Project Run Window Help
Quick Access
SCCharts Modeling
Diagram
No model in active editor
Project Explorer
KIELER Compiler Selection
Console

If your screen is missing the displayed *Diagram* and *KIELER Compiler Selection* view then you might not have the Java 8 SDK installed or Eclipse may use the wrong Java version if you have more than one installed on you system. Please use the provided Java 8 installation files on the USB stick.

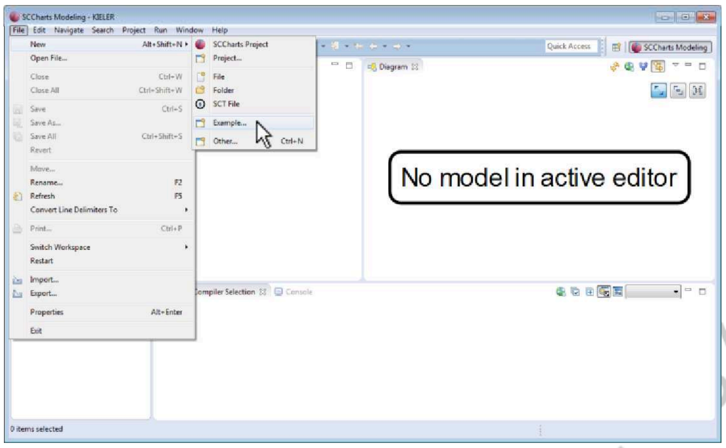
Congratulations, you are now ready to model SCCharts and to get used to the interactive and incremental model-based KIELER SCCharts compiler. :-)

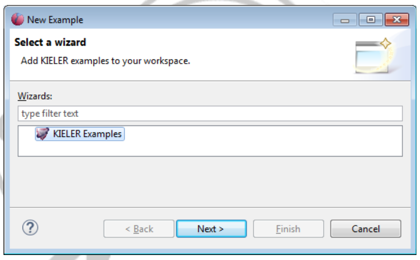
2. Exercise I: Textical Modeling

Learning Objective: Familiarize yourself with getting examples and textical SCCharts modeling.

2.1 Getting SCCharts Examples

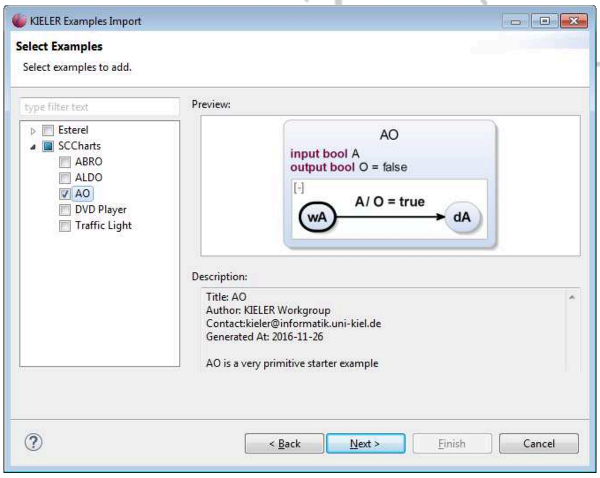
Click on "File" -> "New" -> "Example..." in the main menu. Then select "KIELER Examples" and continue with "Next >".





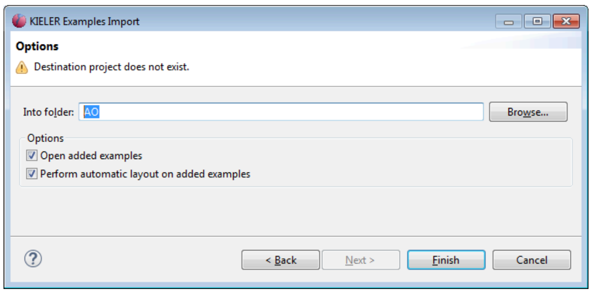
Now you may choose from various examples.

Choose the AO example by checking its checkbox and clicking "Next >". Then enter a proper name that will become the name of the Eclipse project, e.g., call it "AO".



Title: AO
Author: KIELER Workgroup
Contact:kieler@informatik.uni-kiel.de
Generated At: 2016-11-26

AO is a very primitive starter example

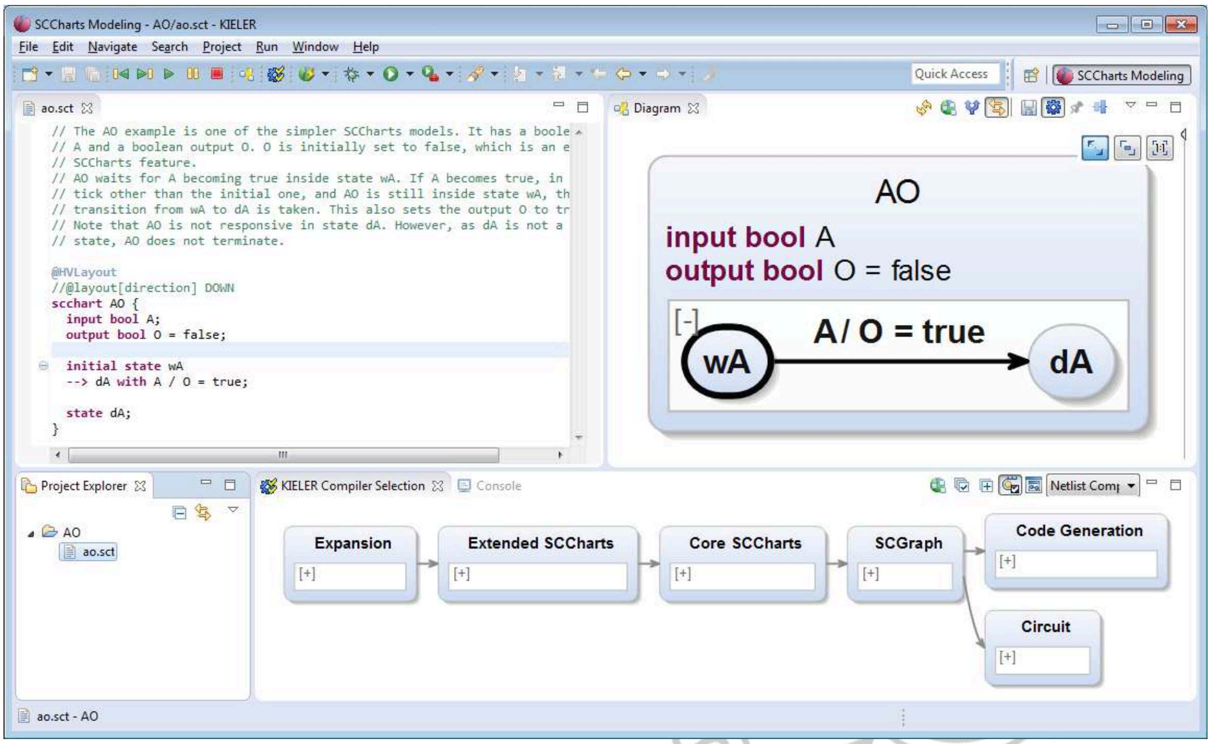


Destination project does not exist.

Into folder: AO

Options
- [x] Open added examples
- [x] Perform automatic layout on added examples

After clicking "Finish", your screen should look similar to the following one. The textual SCCharts editor is opened on the left side, the automatically created diagram is shown on the right side, the Project Explorer is found on the lower left side and the KIELER Compiler Selection is visible in the lower right part of the window.



```
// The AO example is one of the simpler SCCharts models. It has a boole
// A and a boolean output O. O is initially set to false, which is an e
// SCCharts feature.
// AO waits for A becoming true inside state wA. If A becomes true, in
// tick other than the initial one, and AO is still inside state wA, th
// transition from wA to dA is taken. This also sets the output O to tr
// Note that AO is not responsive in state dA. However, as dA is not a
// state, AO does not terminate.

@HVLayout
//@layout[direction] DOWN
scchart AO {
  input bool A;
  output bool O = false;

  initial state wA
  --> dA with A / O = true;

  state dA;
}
```

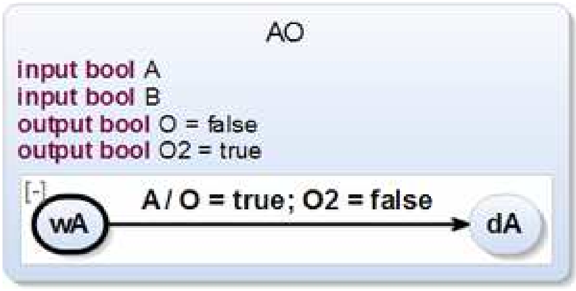
2.2 Textical Modeling Basics

You may now edit the AO example. Expect the diagram to be updated whenever you save your model (<Ctrl>+<S> or <Cmd>+<S>).

First, let's add another output variable O2 and initialize it to true. Just modify the code as follows:

```
scchart AO {
  input bool A;
  input bool B;
  output bool O = false;
  output bool O2 = true;

  initial state wA
  --> dA with A / O = true; O2 = false;

  state dA;

}
```

After saving the model, the diagram will update accordingly and show the following graphical SCChart:



2.3 Textual SCCharts Editor

Code Formatter

The editor is equipped with an automated code formatter that can be called using <Ctrl>+<Shift>+<F>. This can be useful any time when editing an SCChart. For example if you remove some line breaks, your code may look like as follows:

```
scchart AO {
  input bool A;
  output bool O = false; output bool O2 = true;
  initial state wA  --> dA with A / O = true; O2 = false;
  state dA;
}
```

Now, after pressing <Ctrl>+<Shift>+<F>, the code formatter re-arranges the SCT code and makes it better readable. It arranges the interface part of a state at the top and separates it and all child states by a blank line. Outgoing transitions of a state are put at the bottom of a state declaration, after the internal behavior which is declared inside curly brackets "{ ... }", each transition in a new line.

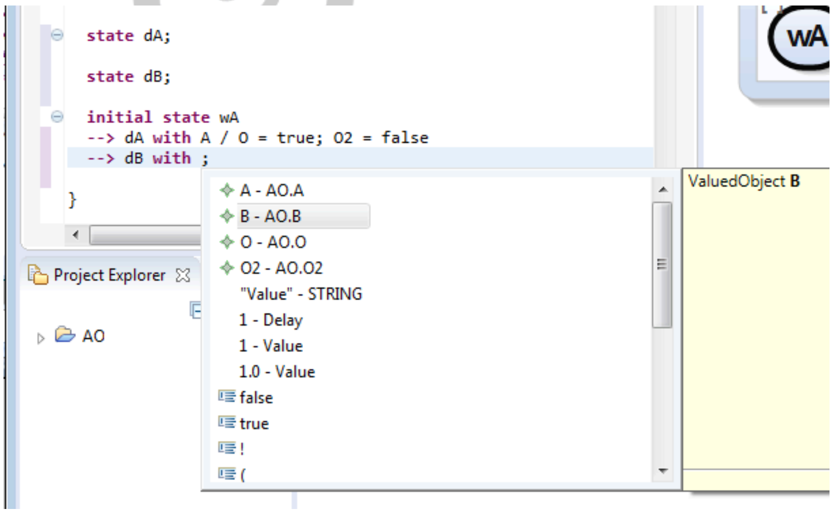
```
scchart AO {
  input bool A;
  output bool O = false;
  output bool O2 = true;

  initial state wA
  --> dA with A / O = true; O2 = false;

  state dA;
}
```

Content Assist

The editor is equipped with a content assist feature which helps you writing SCT code more quickly. It also guides you towards syntactically correct SCT code if, for example you are not sure about the concrete order of specific keywords.

By pressing <Ctrl>+<Space> you activate the content assist.

Try to define a new bool input "B", a new state "dB", and add another transition from "wA" to "dB" in case of "B" becomes true. Set "O2" to false only in this transition.



```
  state dA;

  state dB;

  initial state wA
  --> dA with A / O = true; O2 = false
  --> dB with ;

}
```

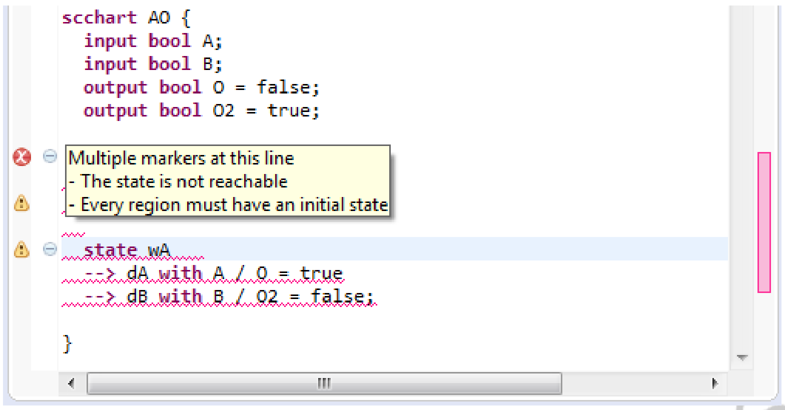
A - AO.A
B - AO.B
O - AO.O
O2 - AO.O2
"Value" - STRING
1 - Delay
1 - Value
1.0 - Value
false
true
!
(

ValuedObject **B**

Syntax Validation

The editor is equipped with a syntax validator which checks the SCT model for syntactical problems or errors. It is called automatically while typing. You will notice possible error or warning markers beside the concerning lines.

Try to remove the "initial" keyword from the state "wB".



```
scchart AO {
  input bool A;
  input bool B;
  output bool O = false;
  output bool O2 = true;

  state wA
  --> dA with A / O = true
  --> dB with B / O2 = false;

}
```

```
Multiple markers at this line
- The state is not reachable
- Every region must have an initial state
```

If you hover the mouse over these markers you get a hint what the problem is. In this case now all states become not reachable (which is a warning because this does not harm but in the end is dead code that will be eliminated). Additionally, as every region must have an initial state and the region in "AO" doesn't have one any more now, this will result in an error marker telling that there is no initial state found.

3. Exercise II: Interactive Compilation

Learning Objective: Familiarize yourself with the interactive incremental compilation tool chain for SCCharts.

3.1 Interactive Compilation

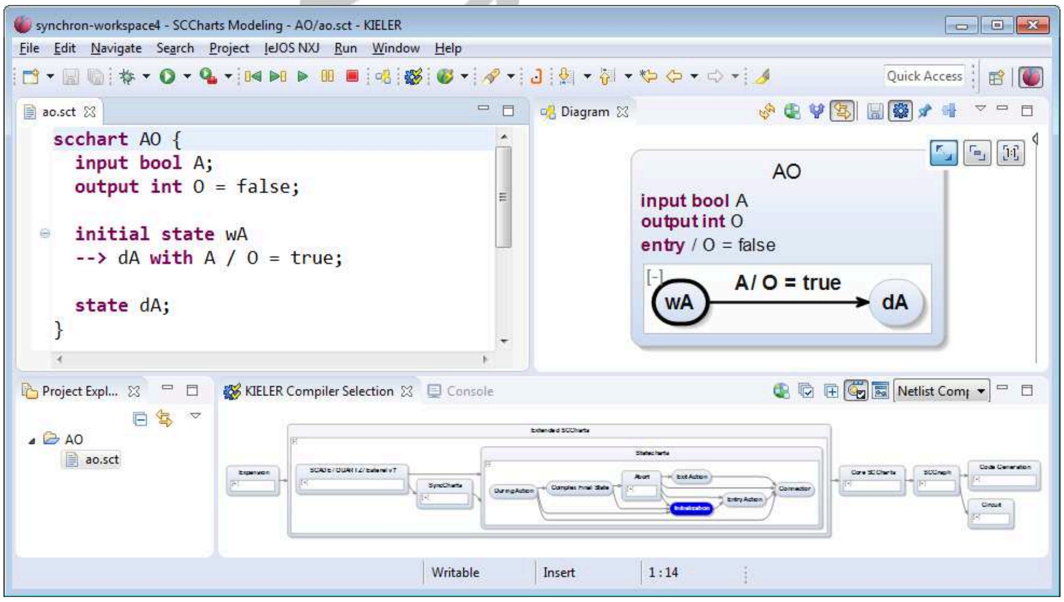
Start again with the AO example:

```
scchart AO {
  input bool A;
  output bool O = false;

  initial state wA
  --> dA with A / O = true;

  state dA;

}
```

Note that the "O = false" is the use of the *initialization* feature. While compiling AO, initializations are replaced by entry actions which will perform the task of initializing a variable.

To inspect what this concretely means for your model, use the KIELER Compiler Selection view and select the "Initialization" feature transformation there. The model diagram will update and show the transformed model with entry actions as follows. Note that you need to first expand the "Extended SCCharts" feature group and then the "Statecharts" feature group in order to see the "Initialization" feature transformation.

Note that the diagram in the KIELER Compiler Selection re-uses the syntax of SCCharts but actually is not an SCChart. Nodes in this diagram represent feature transformations or feature groups and transitions represent dependencies. A dependency between two feature transformations simply means a certain order for applying these transformations.



synchron-workspace4 - SCCharts Modeling - AO/ao.sct - KIELER

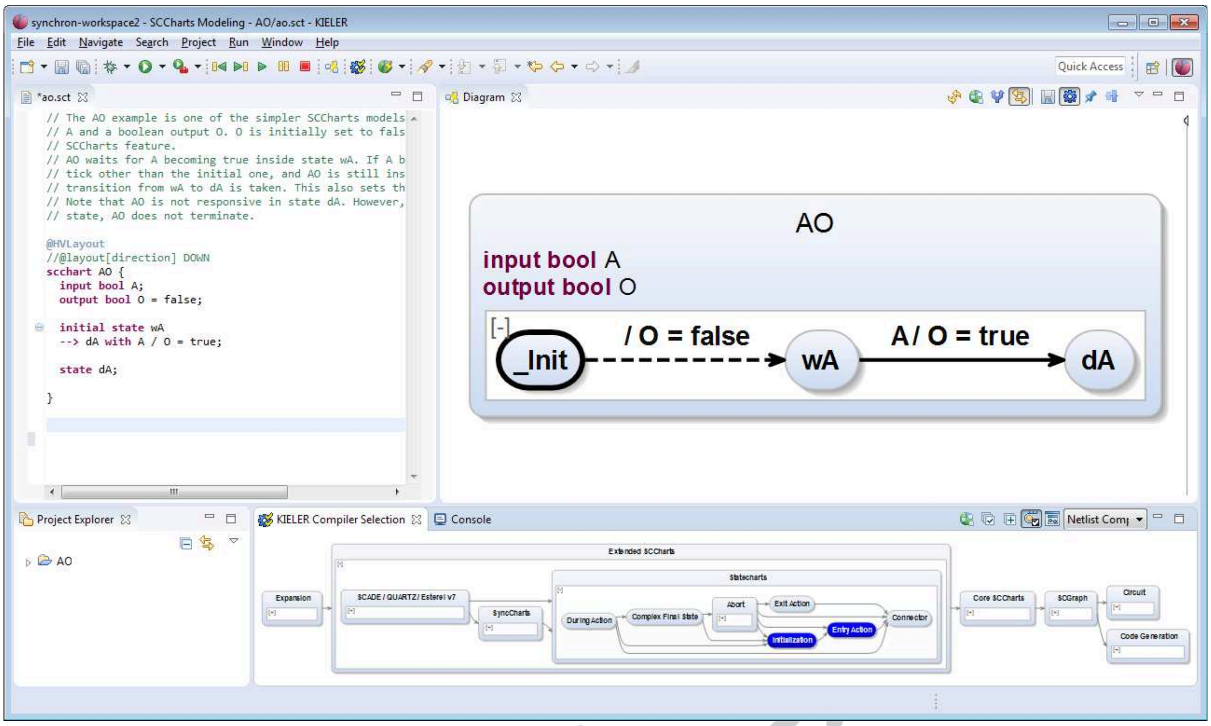
```
scchart AO {
  input bool A;
  output int O = false;

  initial state wA
  --> dA with A / O = true;

  state dA;
}
```

AO
input bool A
output int O
entry / O = false

Writable | Insert | 1 : 14

For example, because the "Initialization" feature transformation produced new entry actions, it makes sense to transform the "Entry Action" feature only after the "Initialization" feature.

After also selecting the "Entry Action" feature transformation the transformed SCCharts looks like follows:

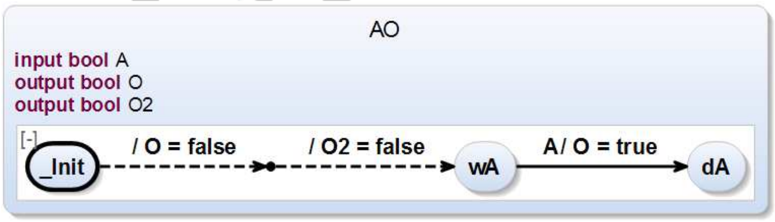


```
// The AO example is one of the simpler SCCharts models
// A and a boolean output O. O is initially set to fals
// SCCharts feature.
// AO waits for A becoming true inside state wA. If A b
// tick other than the initial one, and AO is still ins
// transition from wA to dA is taken. This also sets th
// Note that AO is not responsive in state dA. However,
// state, AO does not terminate.

@HVLayout
//@layout[direction] DOWN
scchart AO {
  input bool A;
  output bool O = false;

  initial state wA
  --> dA with A / O = true;

  state dA;

}
```

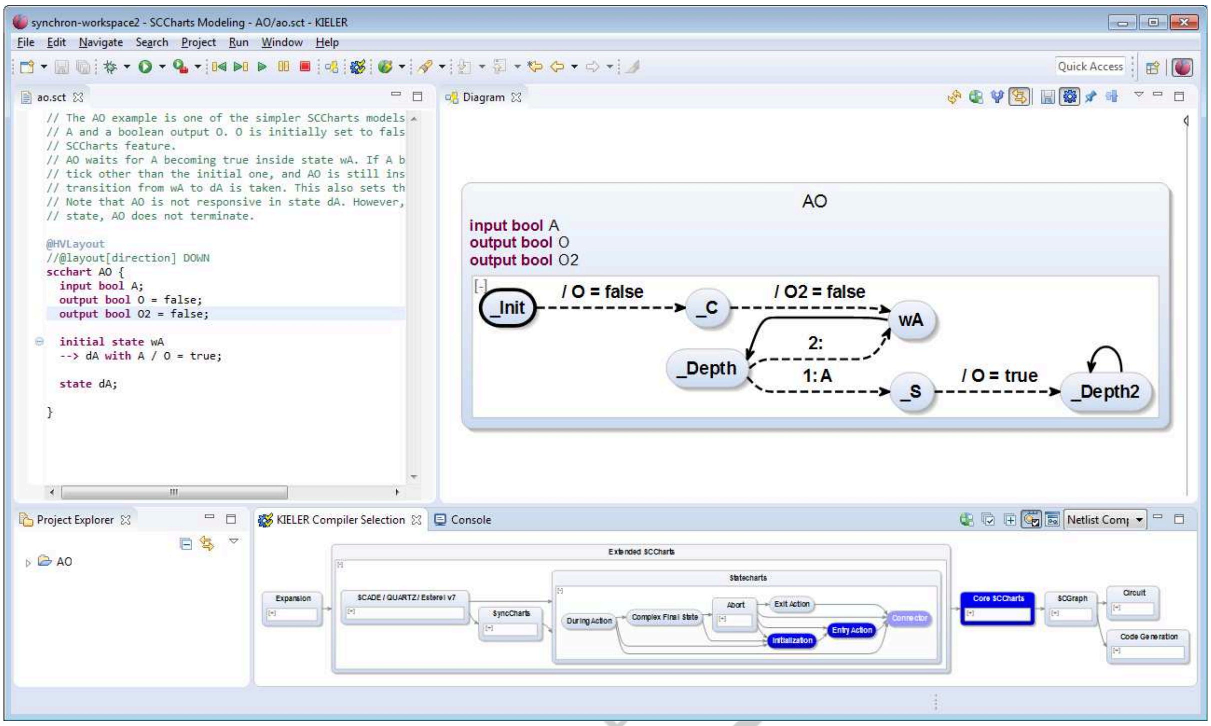
3.2 Interactive Compilation Modeling

Even if you select certain transformations to apply in the KIELER Compiler Selection view, you can still edit the original SCChart. The output will then be the updated original model with the selected feature transformations applied such that the selected features are not contained in the (intermediate) model any more. These feature are replaced by more basic features. For example "Entry" is a more basic feature compared to "Initialization".

You can add another interface declaration line, introducing another output variable O2 that is initialized to false. The intermediate transformed SCChart diagram should look like this:



You could further experiment with transforming the intermediate result down to a Normalized Core SCChart by selecting all "Core SCCharts" feature transformations. The result should look similar to the following diagram:



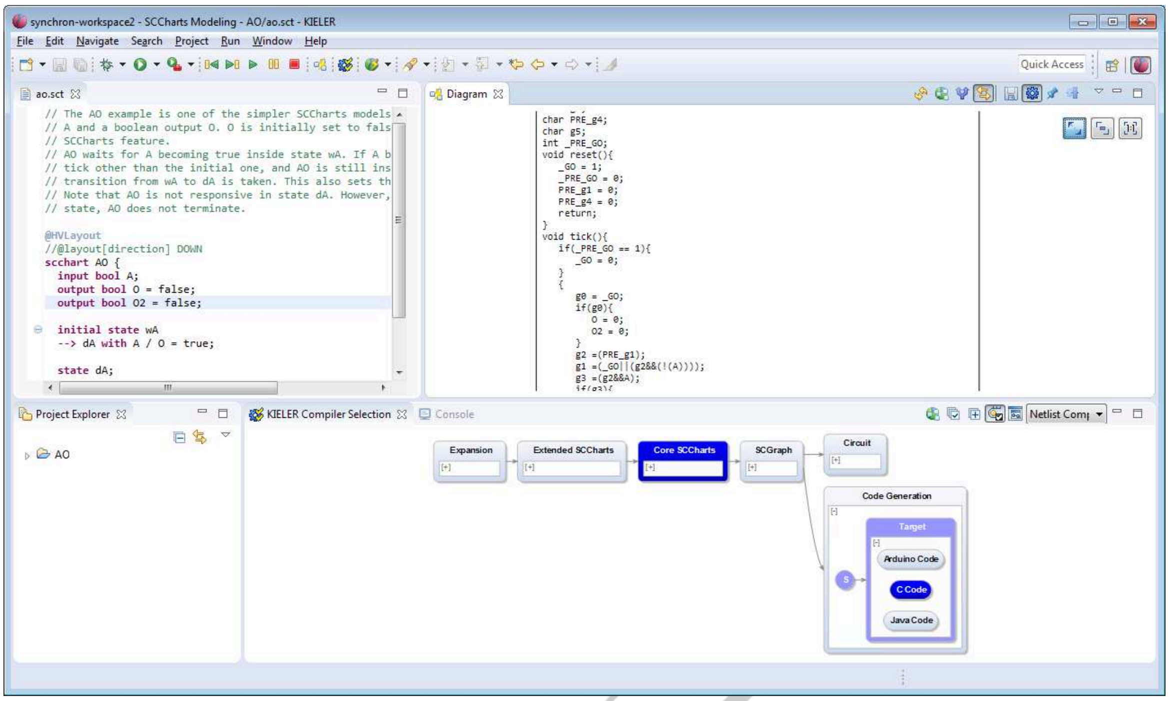
```
// The AO example is one of the simpler SCCharts models
// A and a boolean output O. O is initially set to fals
// SCCharts feature.
// AO waits for A becoming true inside state wA. If A b
// tick other than the initial one, and AO is still ins
// transition from wA to dA is taken. This also sets th
// Note that AO is not responsive in state dA. However,
// state, AO does not terminate.

@HVLayout
//@layout[direction] DOWN
scchart AO {
  input bool A;
  output bool O = false;
  output bool O2 = false;

  initial state wA
  --> dA with A / O = true;

  state dA;

}
```

In the normalized form, the SCChart will contain only 5 primitive patterns which are immediate Actions, immediate Conditionals, delayed Pause, hierarchy with Fork/Join, and Threads starting with an initial node and optionally final states.

For AO, the normalized form for example reveals that the model never terminates after setting "O" to true and reaching "_Depth2".

3.3 Generating Code

Code can be created easily by selecting a transformation from the KIELER Compiler Selection in the "Code Generation" feature group.



```
// The AO example is one of the simpler SCCharts models
// A and a boolean output O. O is initially set to fals
// SCCharts feature.
// AO waits for A becoming true inside state wA. If A b
// tick other than the initial one, and AO is still ins
// transition from wA to dA is taken. This also sets th
// Note that AO is not responsive in state dA. However,
// state, AO does not terminate.

@HVLayout
//@layout[direction] DOWN
scchart AO {
  input bool A;
  output bool O = false;
  output bool O2 = false;

  initial state wA
  --> dA with A / O = true;

  state dA;
```

```
char PRE_g4;
char g5;
int _PRE_GO;
void reset(){
   _GO = 1;
   _PRE_GO = 0;
   PRE_g1 = 0;
   PRE_g4 = 0;
   return;
}
void tick(){
   if(_PRE_GO == 1){
      _GO = 0;
   }
   {
      g0 = _GO;
      if(g0){
         O = 0;
         O2 = 0;
      }
      g2 =(PRE_g1);
      g1 =(_GO||(g2&&(!(A))));
      g3 =(g2&&A);
```

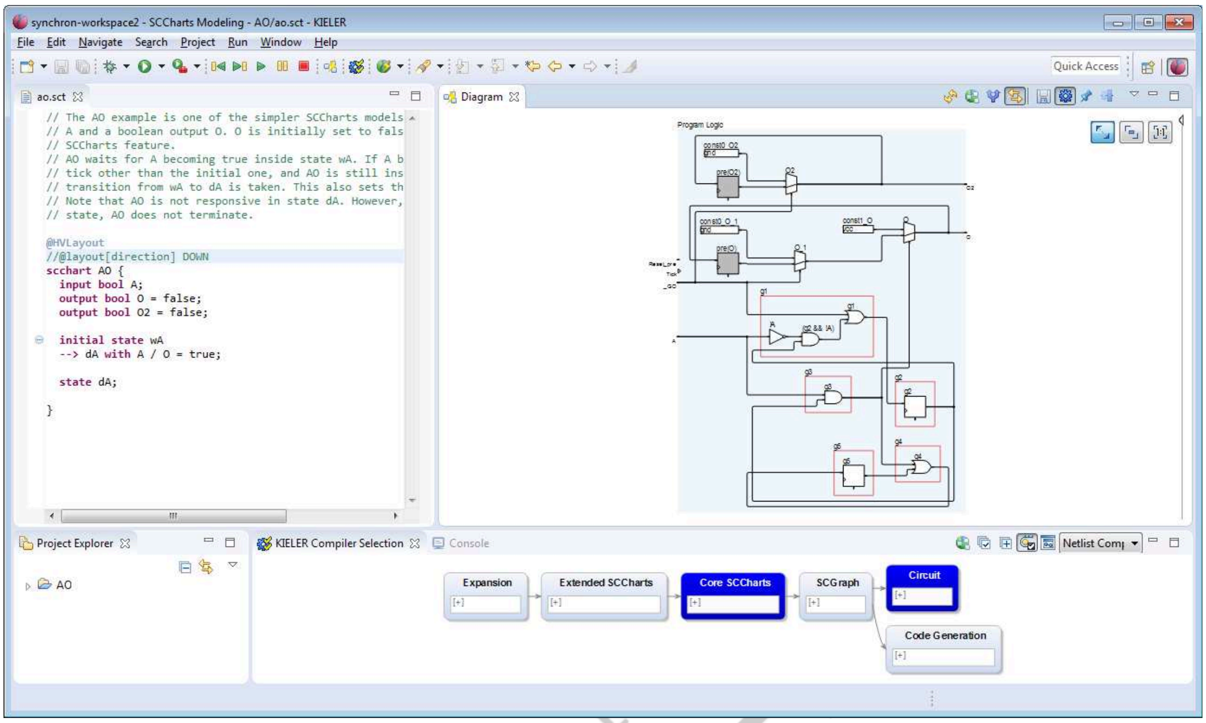
If you doubleclick the code preview of the Diagram view, the code will open in a textual editor.

Note that the code generation will produce a reset() and a tick() function for every SCChart. The original interface is located in the top part of the variable declaration followed by internal guards for the circuit-based code generation.

Note that if you have opened the textual editor then the KIELER Compiler Selection view is disabled and will not react to mouse clicks

3.4 Generating Hardware Circuits

You can easily generate HW circuits by just clicking on the "Circuit" feature group.



```
// The AO example is one of the simpler SCCharts models
// A and a boolean output O. O is initially set to fals
// SCCharts feature.
// AO waits for A becoming true inside state wA. If A b
// tick other than the initial one, and AO is still ins
// transition from wA to dA is taken. This also sets th
// Note that AO is not responsive in state dA. However,
// state, AO does not terminate.

@HVLayout
//@layout[direction] DOWN
scchart AO {
  input bool A;
  output bool O = false;
  output bool O2 = false;

  initial state wA
  --> dA with A / O = true;

  state dA;

}
```

Note that also the HW circuit synthesis only works for bool values and not for int or other kind of values.

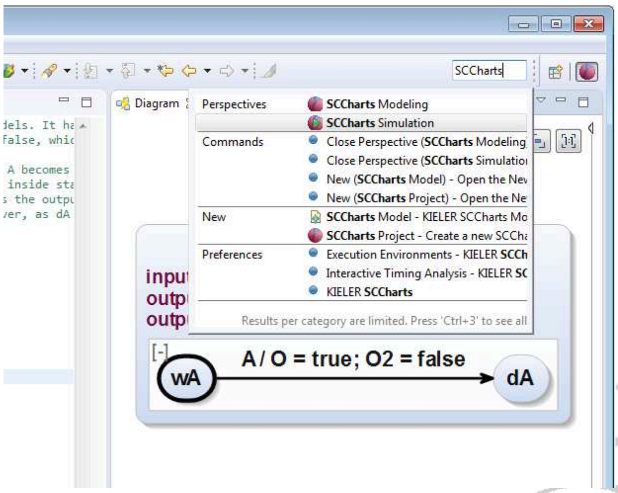
4. Exercise III: Simulation

Learning Objective: Familiarize yourself with the simulation feature of KIELER SCCharts.

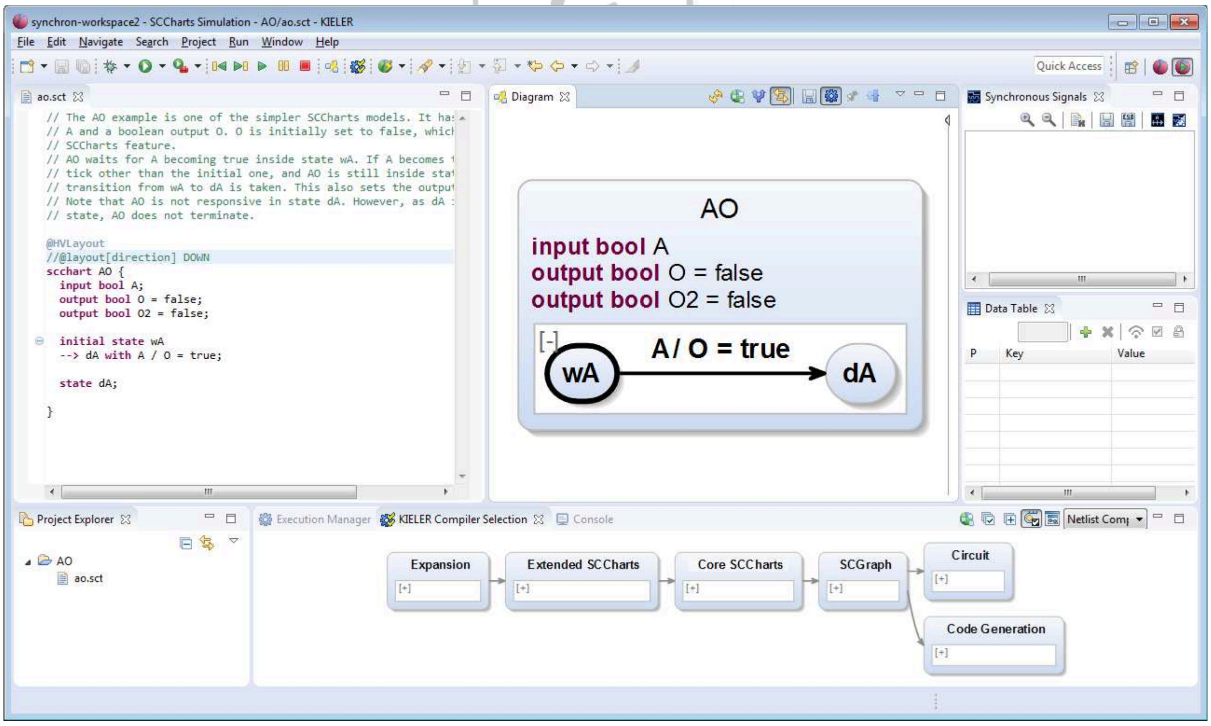
The simulation feature helps to understand the dynamics of an SCChart.

4.1 Simulation Perspective

There is a dedicated perspective for simulating SCCharts. You can reach it by starting to type "SCCharts" into the Quick Access textbox at the upper right corner of the window.


SCCharts
Diagram
Perspectives
SCCharts Modeling
SCCharts Simulation
Commands
Close Perspective (SCCharts Modeling
Close Perspective (SCCharts Simulatio
New (SCCharts Model) - Open the Nev
New (SCCharts Project) - Open the Ne
New
SCCharts Model - KIELER SCCharts Mo
SCCharts Project - Create a new SCCha
Preferences
Execution Environments - KIELER SCCh
Interactive Timing Analysis - KIELER SC
KIELER SCCharts
Results per category are limited. Press 'Ctrl+3' to see all
A / O = true; O2 = false
wA
dA

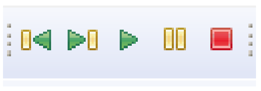
After choosing the simulation perspective the window should look like this:

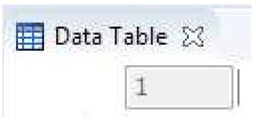


```
// The AO example is one of the simpler SCCharts models. It ha
// A and a boolean output O. O is initially set to false, whic
// SCCharts feature.
// AO waits for A becoming true inside state wA. If A becomes
// tick other than the initial one, and AO is still inside sta
// transition from wA to dA is taken. This also sets the output
// Note that AO is not responsive in state dA. However, as dA
// state, AO does not terminate.

@HVLayout
//@layout[direction] DOWN
scchart AO {
  input bool A;
  output bool O = false;
  output bool O2 = false;

  initial state wA
  --> dA with A / O = true;

  state dA;

}
```

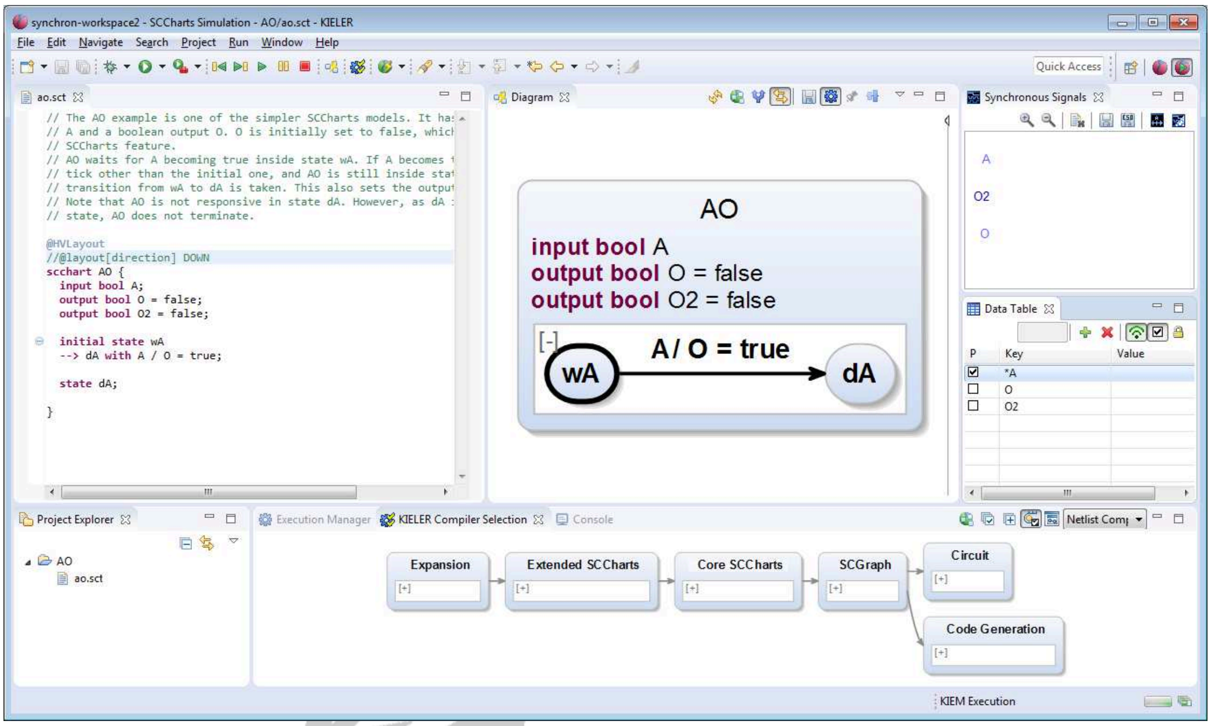
4.2 Simulation Control and Initialization

The synchronous step control buttons can be found in the toolbar. The Data Table has a text field which indicates the synchronous tick number.





After clicking on the step button (the second button from left), the simulation is initialized before the first/initial tick.



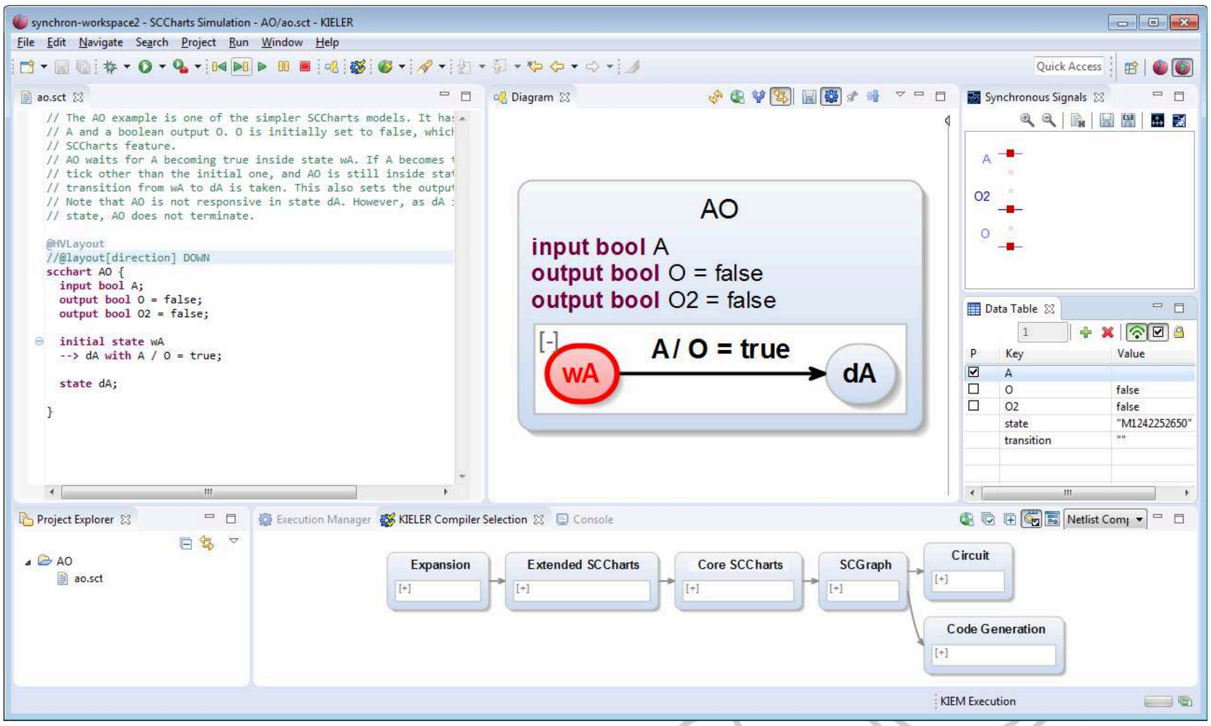
```
// The AO example is one of the simpler SCCharts models. It ha:
// A and a boolean output O. O is initially set to false, whicl
// SCCharts feature.
// AO waits for A becoming true inside state wA. If A becomes t
// tick other than the initial one, and AO is still inside stat
// transition from wA to dA is taken. This also sets the output
// Note that AO is not responsive in state dA. However, as dA :
// state, AO does not terminate.

@HVLayout
//@layout[direction] DOWN
scchart AO {
  input bool A;
  output bool O = false;
  output bool O2 = false;

  initial state wA
  --> dA with A / O = true;

  state dA;

}
```

P	Key	Value
☑	*A	
☐	O	
☐	O2	

The Data Table shows all interface variables and signals that are present in the model. The checkbox can be used for input signals to be set to present in the next tick or for input int/bool variables to set them to 1/true. The star (*) marks all modified entries that are considered the for the next synchronous tick reaction calculation of the SCChart.

4.3 Synchronous Ticks

For example, one could set the input "A" to be true in the initial tick by checking the check box (see above screen shot). Then when performing the first real tick computation by clicking on the step button again, the active state wA is high-lighted as follows:

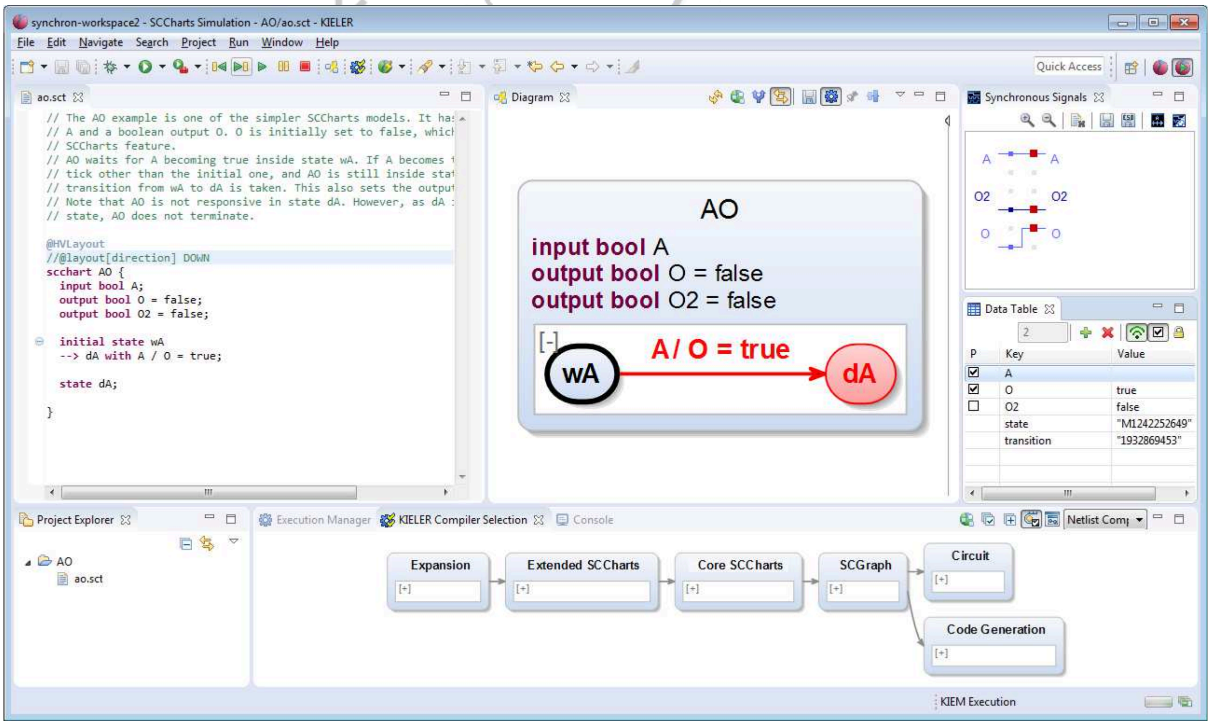


```
// The AO example is one of the simpler SCCharts models. It ha:
// A and a boolean output O. O is initially set to false, which
// SCCharts feature.
// AO waits for A becoming true inside state wA. If A becomes t
// tick other than the initial one, and AO is still inside stat
// transition from wA to dA is taken. This also sets the output
// Note that AO is not responsive in state dA. However, as dA :
// state, AO does not terminate.

@HVLayout
//@layout[direction] DOWN
scchart AO {
  input bool A;
  output bool O = false;
  output bool O2 = false;

  initial state wA
  --> dA with A / O = true;

  state dA;

}
```

P	Key	Value
☑	A	
☐	O	false
☐	O2	false
	state	"M1242252650"
	transition	""

Note that AO did not react to the input "A" in the initial tick because it is modeled with a delayed transition.

If you set the input "A" for the next (second) tick and do another tick computation then the transition to "dA" is taken. Note that you have to click the checkbox of "A" twice in this scenario because the first click will modify "A" and unselect the checkbox and the second click will select the checkbox again. The simulation is also visualizing taken transitions as follows:



```
// The AO example is one of the simpler SCCharts models. It ha:
// A and a boolean output O. O is initially set to false, which
// SCCharts feature.
// AO waits for A becoming true inside state wA. If A becomes t
// tick other than the initial one, and AO is still inside stat
// transition from wA to dA is taken. This also sets the output
// Note that AO is not responsive in state dA. However, as dA :
// state, AO does not terminate.

@HVLayout
//@layout[direction] DOWN
scchart AO {
  input bool A;
  output bool O = false;
  output bool O2 = false;

  initial state wA
  --> dA with A / O = true;

  state dA;

}
```

P	Key	Value
☑	A	
☑	O	true
☐	O2	false
	state	"M1242252649"
	transition	"1932869453"

Note that the Synchronous Signals view above the Data Table shows the history of the signals and (bool/in) variable values according to the presence or true/false or 1/0 value.

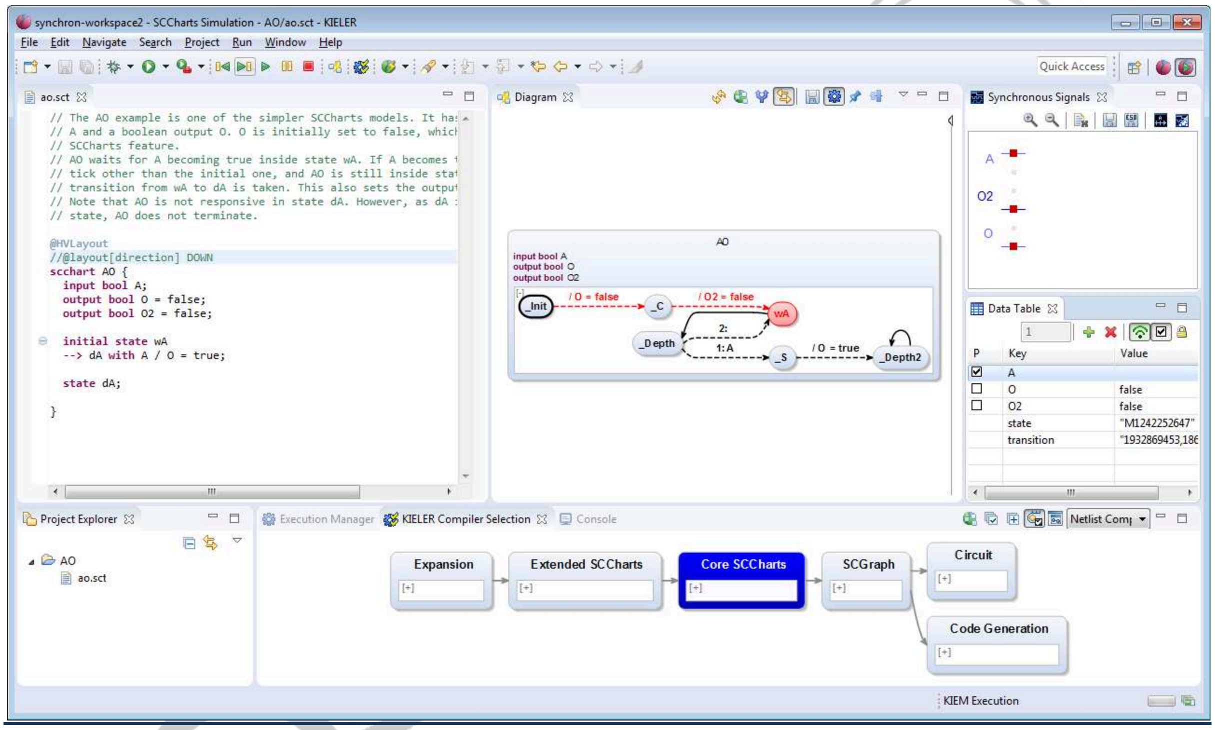
4.4 Simulating Intermediate Models

Usually, the input for the simulation is the originally modeled SCChart as it was in the example above.

If one would like to investigate the dynamics of intermediate models during compilation then one needs again the KIELER Compiler Selection view. Stop any running simulation execution before selecting feature transformations.

E.g., select the "Core SCCharts" feature transformation. Note that this will include all necessary feature transformations from the "Extended SCCharts" feature group as well.

If you now initialize the simulation and for the initial/first tick initialize "A" with true (checked checkbox) then the behavior why AO does not react to "A" in the initial tick becomes more clear:

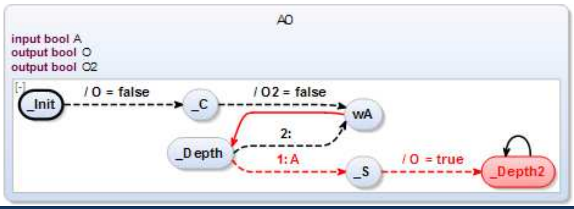


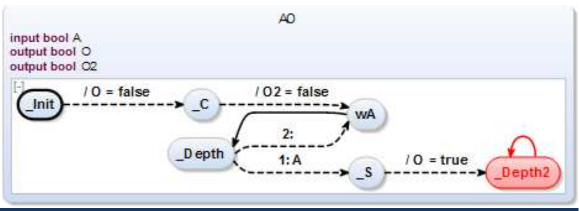
```
// The AO example is one of the simpler SCCharts models. It has
// A and a boolean output O. O is initially set to false, which
// SCCharts feature.
// AO waits for A becoming true inside state wA. If A becomes t
// tick other than the initial one, and AO is still inside stat
// transition from wA to dA is taken. This also sets the output
// Note that AO is not responsive in state dA. However, as dA i
// state, AO does not terminate.

@HVLayout
//@layout[direction] DOWN
scchart AO {
  input bool A;
  output bool O = false;
  output bool O2 = false;

  initial state wA
  --> dA with A / O = true;

  state dA;

}
```

P	Key	Value
☑	A	
☐	O	false
☐	O2	false
	state	"M1242252647"
	transition	"1932869453,186

All the initialization is done which sets the output "O" and "O2" both to false. But then the state "wA" is entered without checking for its outgoing transitions in the initial tick (because the only outgoing transition is a delayed and not an immediate transition).

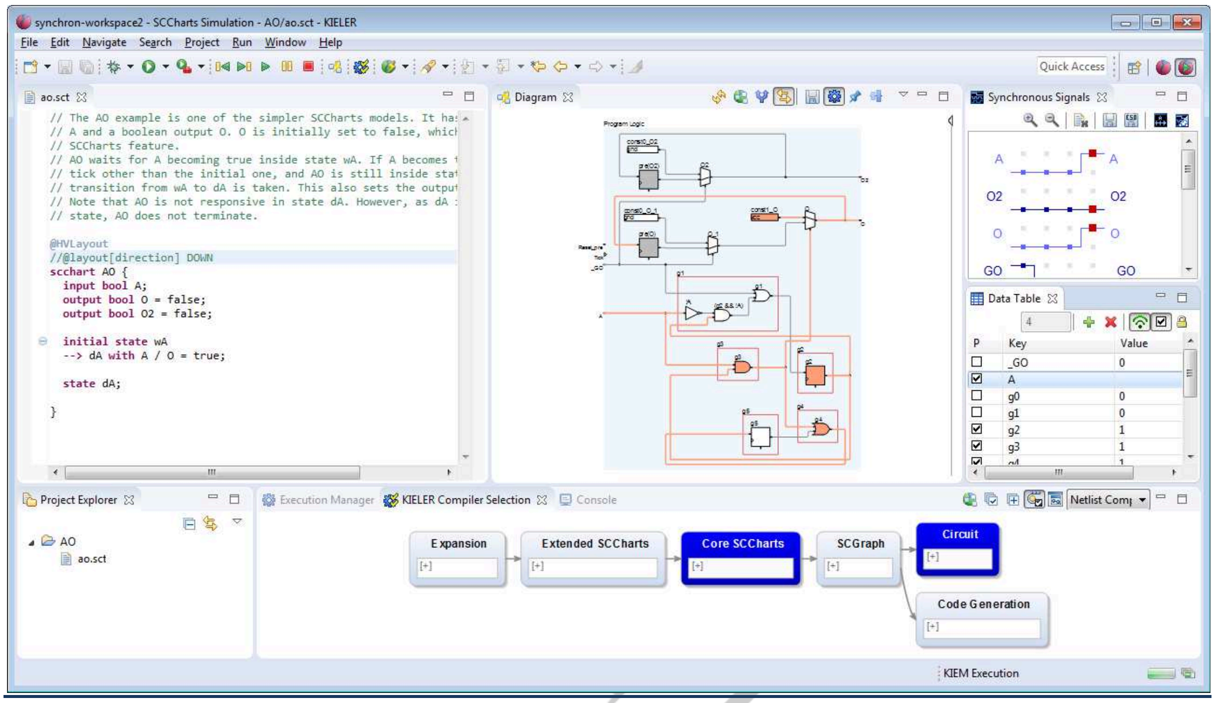
Setting "A" again for the second tick results in the following simulation visualization diagram (left). The right diagram shows all following ticks in which AO will never terminate.





4.5 Hardware Circuit Simulation

For simulating HW circuits simply click on "Circuit" in the KIELER Compiler Selection view and then on the step button to initialize the simulation execution.



```
// The AO example is one of the simpler SCCharts models. It ha
// A and a boolean output O. O is initially set to false, which
// SCCharts feature.
// AO waits for A becoming true inside state wA. If A becomes t
// tick other than the initial one, and AO is still inside stat
// transition from wA to dA is taken. This also sets the output
// Note that AO is not responsive in state dA. However, as dA i
// state, AO does not terminate.

@HVLayout
//@layout[direction] DOWN
scchart AO {
  input bool A;
  output bool O = false;
  output bool O2 = false;

  initial state wA
  --> dA with A / O = true;

  state dA;

}
```

P	Key	Value
☐	_GO	0
☑	A	
☐	g0	0
☐	g1	0
☑	g2	1
☑	g3	1

Note that additional guards will be shown as an output in the Data Table which are used to visualize the active circuit parts. Note that links that are red indicate that the values on these links are 1/true and gray links indicate that the values on these links are 0/false. Multiplexer with a 1/true selector input will forward their upper input, otherwise the lower input.

Note that also the HW circuit simulation only works for bool values and not for int or other kind of values.

5. Bonus Exercise IV: SCG

Learning Objective: Familiarize yourself with the control-flow graph representation of SCCharts, namely the Sequentially Constructive Graph (SCG).

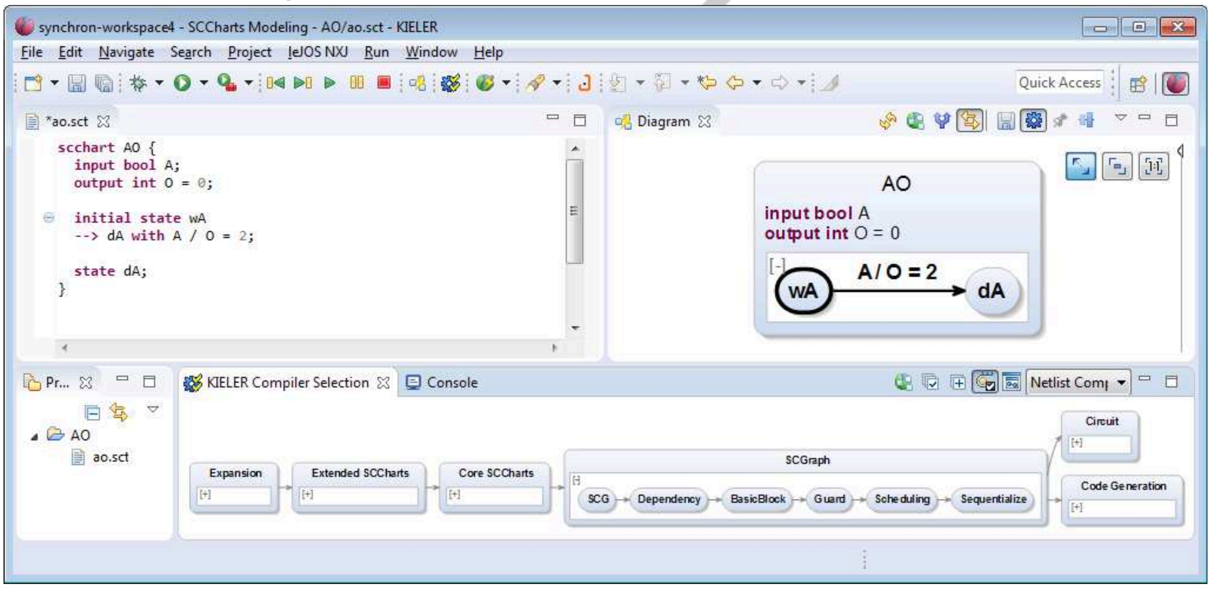
Note: **If time permits**, you are invited to do this bonus exercise. The objective in detail is to understand the scheduling of the SCCharts compiler regarding initializations, updates, and reads (iur) of variables. In particular, this exercise shows how scheduling difficulties can be understood and resolved.

Recall that SCCharts distinguishes three kinds of variable accesses:

1. Absolute writes (also called "initializations"): These are always scheduled first. Example: X = 0
2. Relative writes (also called "updates"): These are scheduled after all absolute writes. Example: X = X + 1
3. Reads: These are always scheduled after all (absolute and relative) writes. Example: Y = X

Note, there concurrent absolute writes cannot be scheduled and are not allowed. However, sequential absolute writes are scheduled as modeled.
Concurrent relative writes can always be scheduled because of a commutative and associative combine function (e.g., ADD, OR, ...).

Consider the following SCChart:

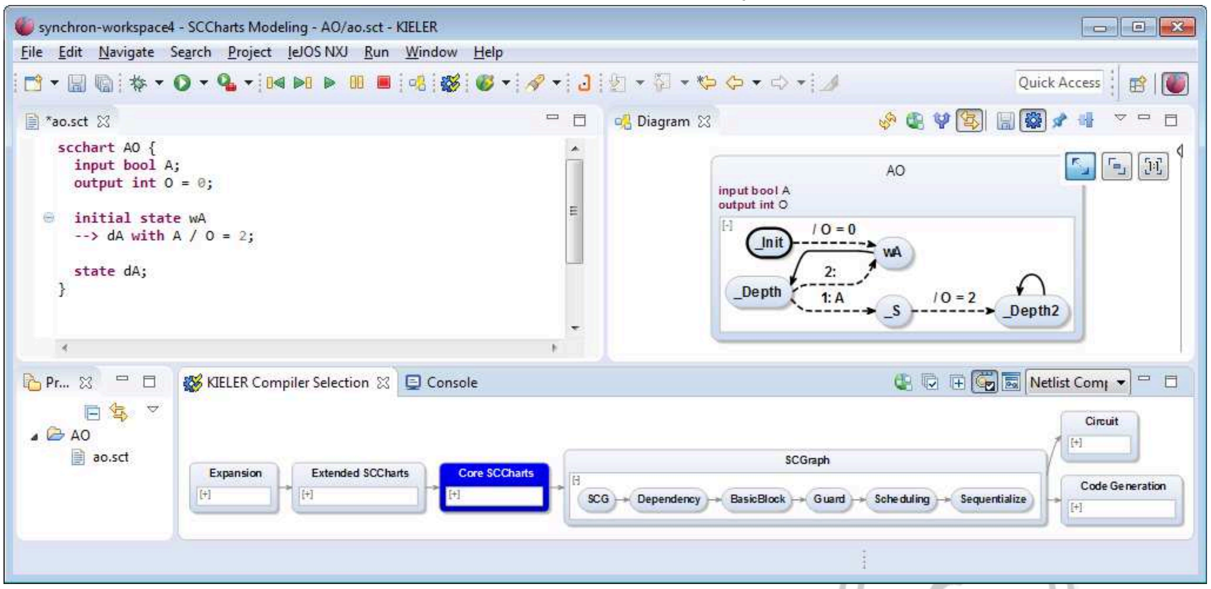


synchron-workspace4 - SCCharts Modeling - AO/ao.sct - KIELER

```
scchart AO {
  input bool A;
  output int O = 0;

  initial state wA
  --> dA with A / O = 2;

  state dA;
}
```

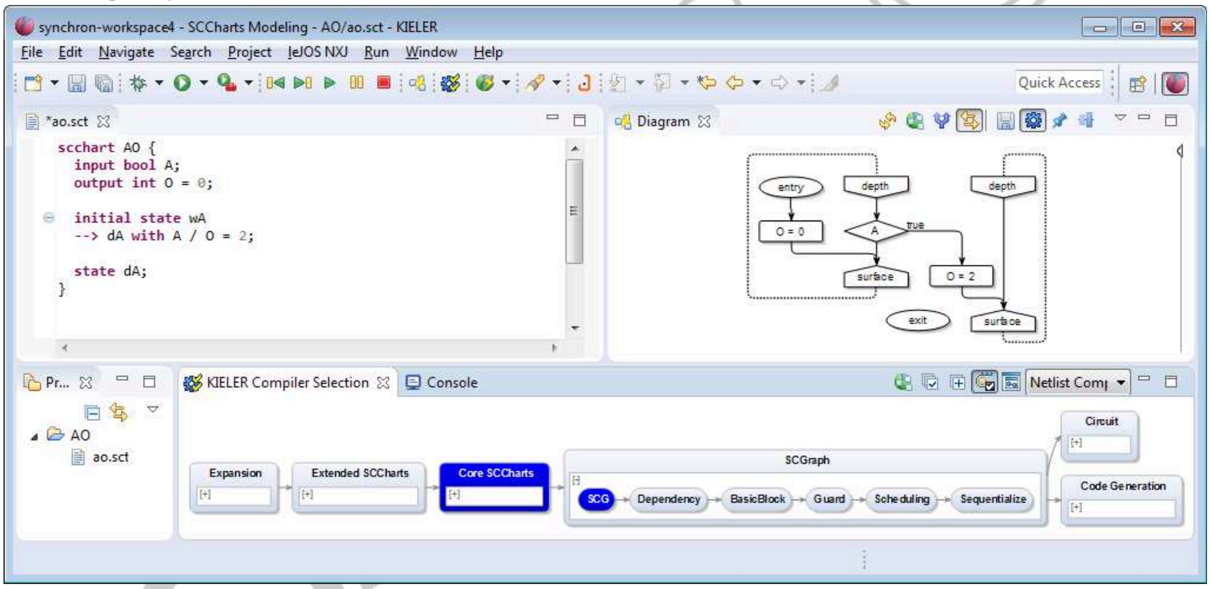
Note that this can be scheduled without any problems although there are two absolute writes. The first sets O=0 in the initialization and the second sets O=2 in the transition. The modeled sequence becomes clear if one has a look at the normalized SCChart.

Select all "Core SCCharts" transformations in the "Compiler Selection" view:



```
scchart AO {
  input bool A;
  output int O = 0;

  initial state wA
  --> dA with A / O = 2;

  state dA;
}
```

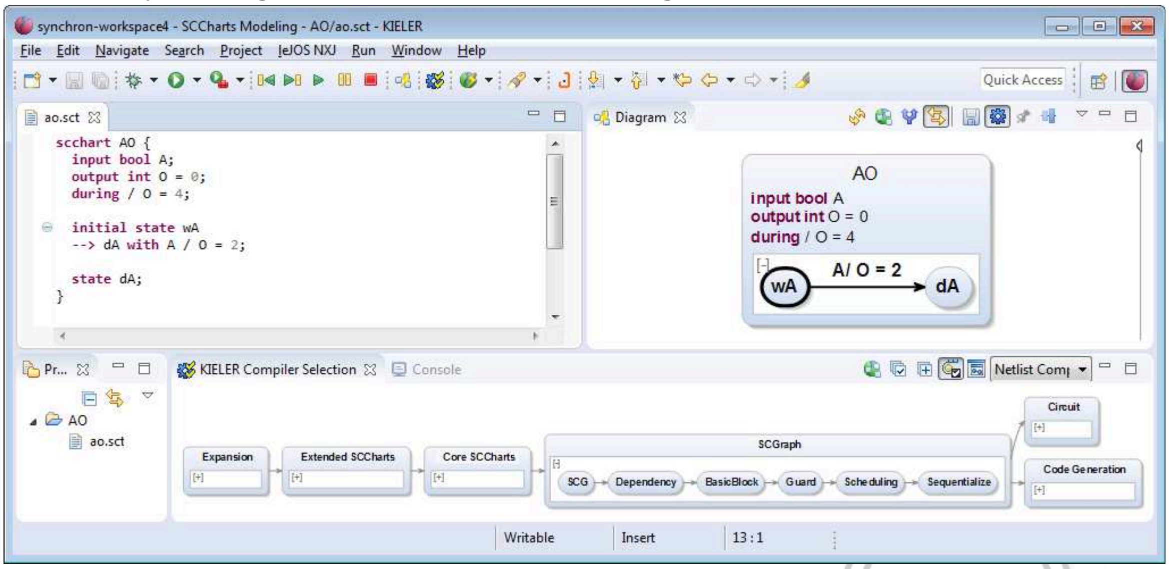
This can be directly mapped to the SCG representation. Select "SCG" in the "SCGraph" feature group:



```
scchart AO {
  input bool A;
  output int O = 0;

  initial state wA
  --> dA with A / O = 2;

  state dA;
}
```

The "entry" node is the start of the (single) thread here. If started, first O is initialized to 0 then a pause-construct ("surface" + "depth") is entered. In the next tick this pause-construct is left. Then, "A" is checked. If "A" is false (else branch), then the pause-construct is entered again. If "A" is true (then branch), then the O is set to 2 and another pause-construct (of the other former state "dA") is entered. As the program never terminates, the "exit" node is left unconnected.

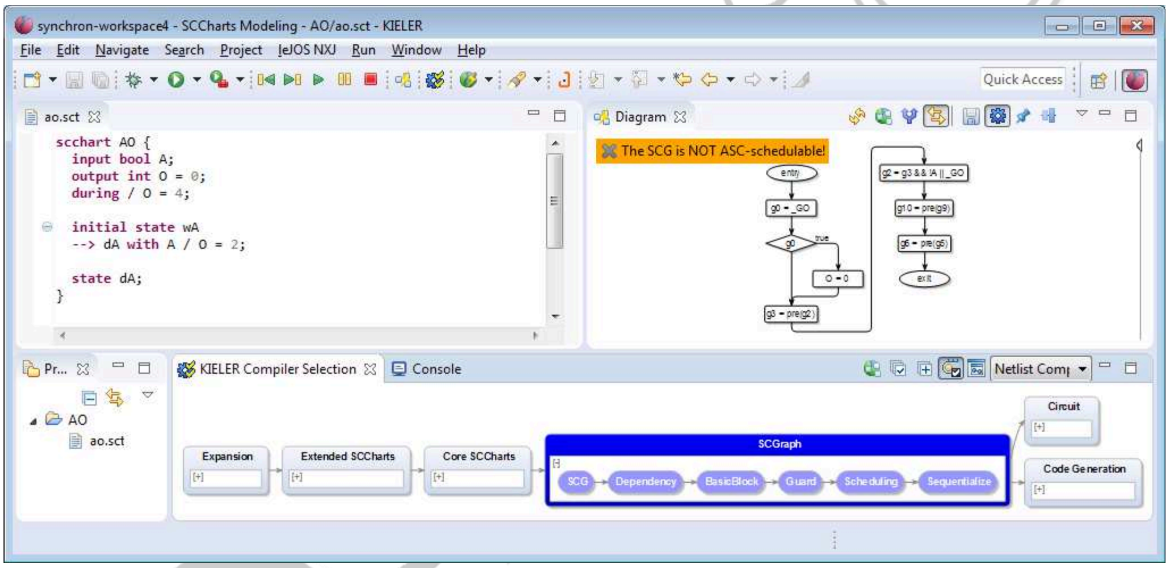
Now modify the original SCChart and add a during action that sets O to 4:



synchron-workspace4 - SCCharts Modeling - AO/ao.sct - KIELER

```
scchart AO {
  input bool A;
  output int O = 0;
  during / O = 4;

  initial state wA
  --> dA with A / O = 2;

  state dA;
}
```

If you now select all "SCGraph" transformations, then you notice that the SCG is not ASC-schedulable:

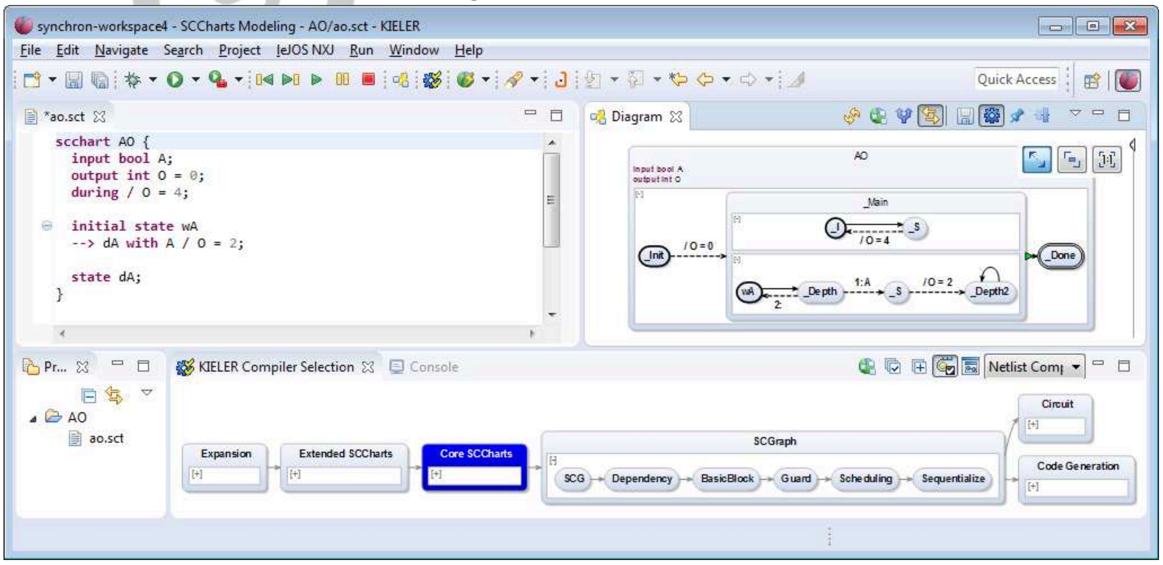


synchron-workspace4 - SCCharts Modeling - AO/ao.sct - KIELER

```
scchart AO {
  input bool A;
  output int O = 0;
  during / O = 4;

  initial state wA
  --> dA with A / O = 2;

  state dA;
}
```

The SCG is NOT ASC-schedulable!

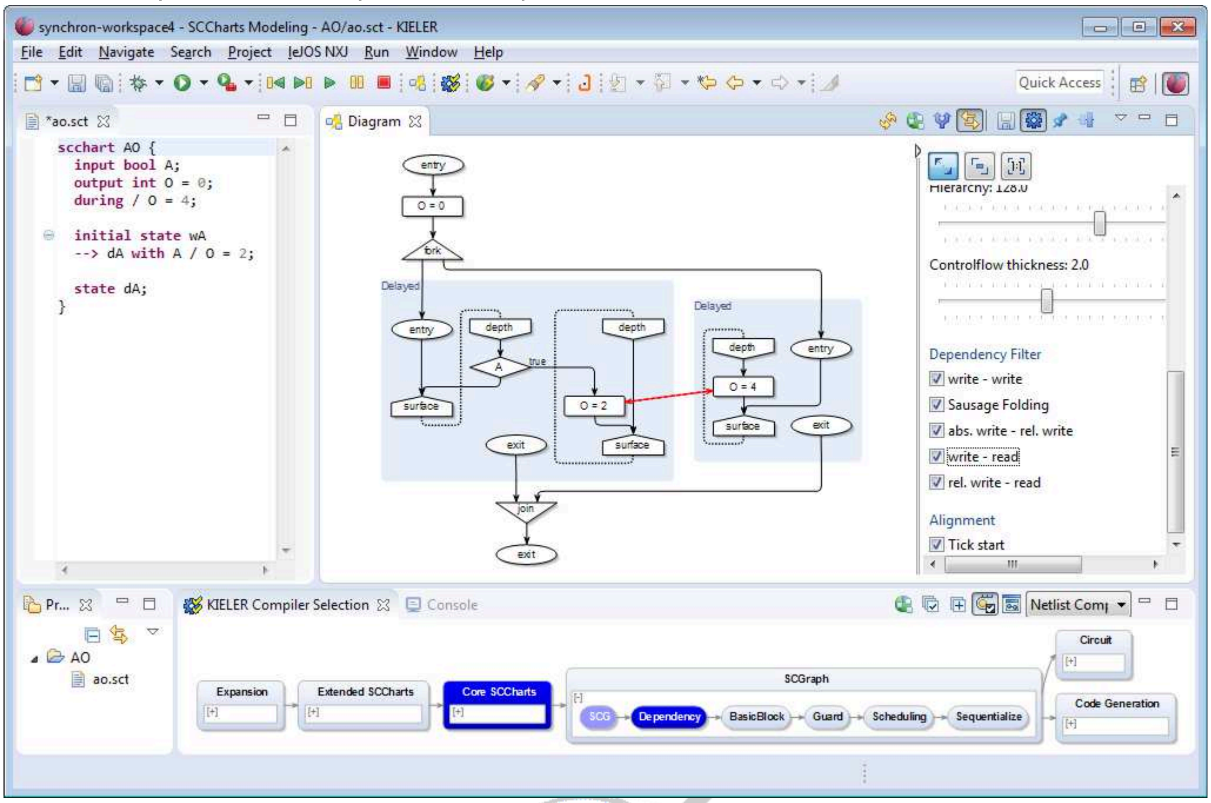
To investigate the problem, take a look at the normalized SCChat, selecting only all "Core SCCharts" transformations again:



synchron-workspace4 - SCCharts Modeling - AO/ao.sct - KIELER

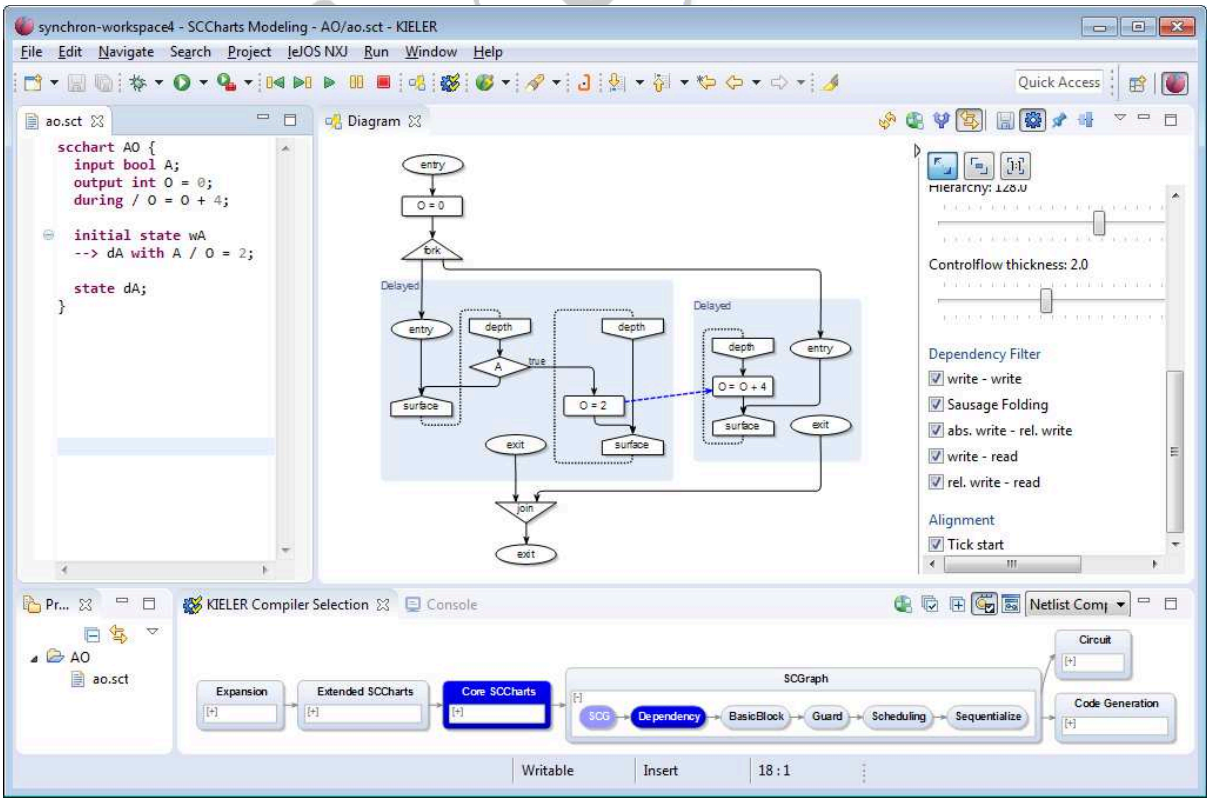
```
scchart AO {
  input bool A;
  output int O = 0;
  during / O = 4;

  initial state wA
  --> dA with A / O = 2;

  state dA;
}
```

The initialization of O=0 is still not a problem. But the during actions is executed concurrently such that this results in a concurrent O=4 and O=2 absolute write. Looking at the SCG you can visually see the dependencies and this so called write-write-conflict:



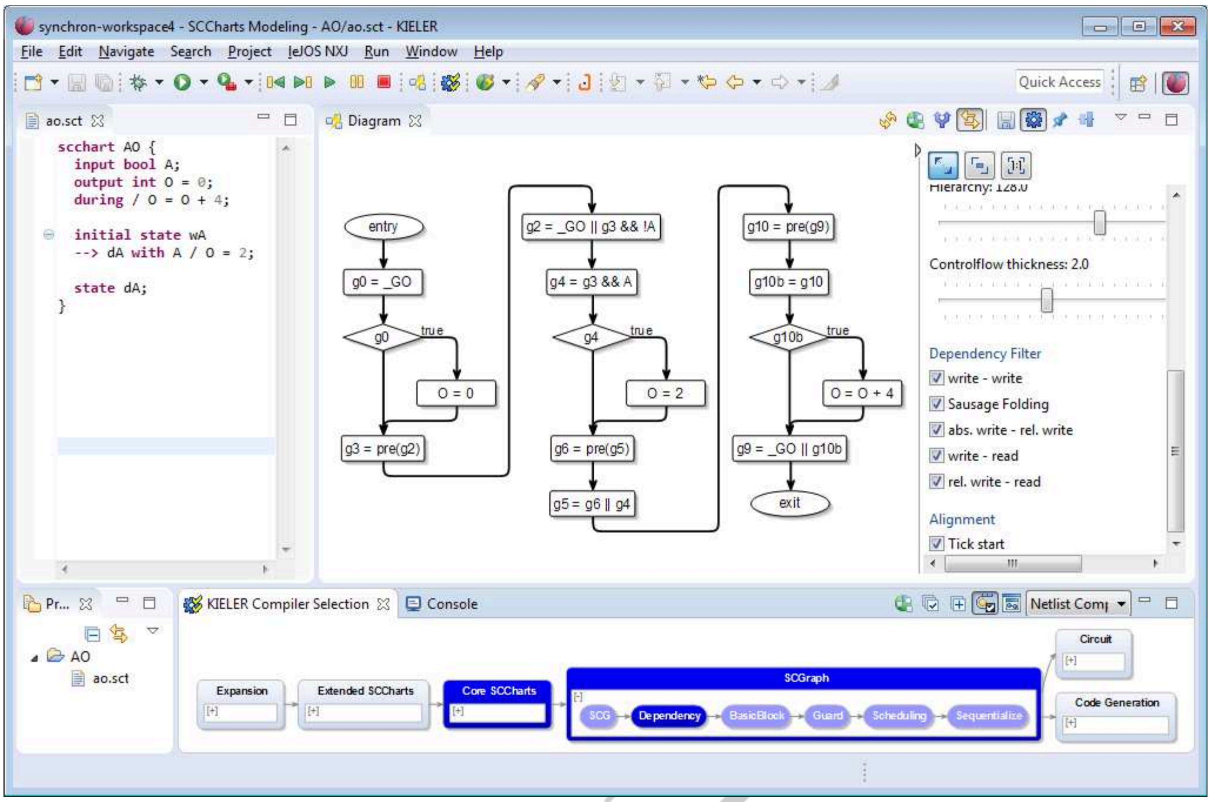
```
scchart AO {
  input bool A;
  output int O = 0;
  during / O = 4;

  initial state wA
  --> dA with A / O = 2;

  state dA;
}
```

A solution is to make one of these absolute writes a relative write, e.g., using addition as a proper combination function:



```
scchart AO {
  input bool A;
  output int O = 0;
  during / O = O + 4;

  initial state wA
  --> dA with A / O = 2;

  state dA;
}
```

As expected, the absolute write is now scheduled before the concurrent relative write.
And the SCG is now properly schedulable:



synchron-workspace4 - SCCharts Modeling - AO/ao.sct - KIELER

```
scchart AO {
  input bool A;
  output int O = 0;
  during / O = O + 4;

  initial state wA
  --> dA with A / O = 2;

  state dA;
}
```

Hierarchy: 128.0
Controlflow thickness: 2.0

Dependency Filter
- [✓] write - write
- [✓] Sausage Folding
- [✓] abs. write - rel. write
- [✓] write - read
- [✓] rel. write - read

Alignment
- [✓] Tick start

6. Lego Mindstorms Preparations

Before we are able to use SCCharts with Lego Mindstorms, some preparation is necessary.

Please note that most limitations that are described here are limitations of leJOS or its Eclipse plugin, not of the KIELER SCCharts tools.

If you do not want to struggle with these extra preparations, then you are invited to use our provided DOWNLOAD STATION. Please use one of the USB sticks to transfer your SCT textual SCChart file to the Download Station.

Note that the installation procedure varies depending on your operating system. In essence, two steps are necessary.

1. Lego Mindstorms driver
2. LeJOS project

First, the Lego device needs to be registered in your operating system. Then the leJOS project needs to be installed. This provides a Java capable firmware for the Lego brick and the uploading facility. The leJOS plugin for Eclipse should be available in the KIELER SCCharts tools already. It will use the installed leJOS project to compile, upload, and run the Java files that are generated from SCCharts.

6.1 Windows

For Windows you will need the **32bit** version of both (a) KIELER SCCharts and (b) Java (see Section 1). Make sure that the 32 Bit Java version is available for the KIELER SCCharts RCA. You may need to check your PATH variable or edit the kieler.ini of the SCCharts RCA in rare cases.

USB Driver (Fantom)

For Windows you will need the Lego Fantom Driver which you may download here or install from the provided USB stick:

https://mi-od-live-s.legocdn.com/r/www/r/mindstorms/-/media/franchises/mindstorms%202014/downloads/firmware%20and%20software/nxt%20software/nxt%20fantom%20drivers%20v120.zip

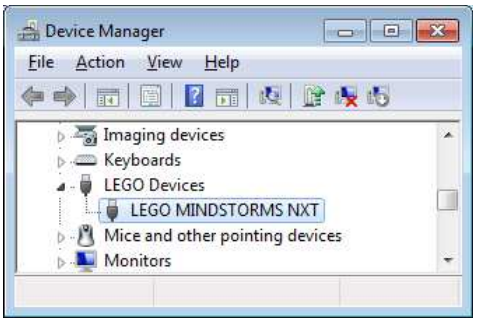
or

http://tinyurl.com/legodriver

or

the file /USBStick/WindowsMacFantomDriver/**NXT Fantom Drivers v120.zip** (from the USB Stick)

After the installation, connect the Mindstorms brick with a USB cable. Check whether you see the Lego Mindstorms device in the Windows Device Manager:


Device Manager

File Action View Help

Imaging devices
Keyboards
LEGO Devices
LEGO MINDSTORMS NXT
Mice and other pointing devices
Monitors

Install LeJOS

You will further need leJOS which you will get from the USB stick or from the following URL: https://sourceforge.net/projects/nxt.lejos.p/files/

or use the following tiny URL: http://tinyurl.com/lejosinstall

Note, that you will need a version that fits to the flashed firmware on the brick. Currently most of our bricks run with **version 0.9.1**.

For Windows, you will mostly like to run the /USBStick/leJOS/**leJOS_NXJ_0.9.1beta-3_win32_setup.exe** executable.

6.2 Mac OSX

For MaxOSX you need to start KIELER SCCharts from the command line. You need to set two path variables before (for details, see below), namely `NXJ_HOME` and `LEJOS_NXT_JAVA_HOME`.

Further, leJOS needs a 32 Bit Java and for MaxOSX the most recent 32 Bit Java is Version 1.6. Hence, you need to install Java 1.6 in addition from Oracle Homepage or from the USB stick.

For KIELER SCCharts you still need Java 1.8.

If you need to install Java 1.6 and Java 1.8, please install Java 1.8 first and then Java 1.6 as the Java 1.8 installation procedure is known to possibly corrupt a previously installed Java 1.6 on MacOSX.

USB Driver (Fantom)

For MacOSX you will need the Lego Fantom Driver which you may download here or install from the provided USB stick:

https://mi-od-live-s.legocdn.com/r/www/r/mindstorms/-/media/franchises/mindstorms%202014/downloads/firmware%20and%20software/nxt%20software/nxt%20fantom%20drivers%20v120.zip

or

http://tinyurl.com/legodriver

or

the file /USBStick/WindowsMacFantomDriver/**NXT Fantom Drivers v120.zip** (from the USB Stick)

Install LeJOS

The OS X edition of leJOS is distributed as a .tar.gz archive. Untar the archive to a location of your choice. You may delete the build subfolder, since it is required for Linux only.

```
tar -xvzf leJOS_NXJ_0.9.1beta-3.tar.gz
```

Set the NXJ_HOME path variable in the KIELER/Eclipse preferences and let it point to your leJOS installation (the root folder, not the bin folder). See explanation in the end.

Also set the leJOS path variable for your shell:

```
export NXJ_HOME="<your leJOS installation root directory>"
```

Install Java Version 1.6 32 Bit

For MaxOSX, you need a 32 Bit version of Java. The latest version of Java that supports 32 Bit on a Mac is 1.6. Unfortunately, all newer version do not even support a 32 Bit mode (which often could be enabled using the -d32 parameter).

Hence, you need to download Java 1.6 32 Bit and install it parallel to you Java 1.8, which you need for working with KIELER SCCharts.

You get the installation bundle here:

http://support.apple.com/downloads/DL1572/en_US/javaforosx.dmg

or

http://tinyurl.com/java16mac

or from the provided USB stick.

For leJOS to find Java 1.6, you need to set an environment variable.

```
export LEJOS_NXT_JAVA_HOME ="<your Java 1.6 installation root
directory>"
```

Start KIELER w/ 32Bit

You need to set the two path variables `NXJ_HOME` and `LEJOS_NXT_JAVA_HOME` before you start KIELER SCCharts from the command line.

Then, navigate to the folder where you expanded KIELER SCCharts and start it using

```
./kieler
```

6.3 Linux

Prepare Java

If you don't have a JDK yet, download and install the newest release from oracle for your system or use the Java SDK provided on the USB stick. LeJOS needs to know the location of this JDK. This can be achieved by creating an environment variable JAVA_HOME. On most Linux systems, environment variables are edited in a file .profile in your home directory. Thus add

```
export JAVA_HOME="/opt/java/jdk1.8.0_60"
```

to the end of ~/.profile. Thereby the path to java must be adapted to your system. Logout and login again to let the change take effect. You can test this by typing `echo $JAVA_HOME`. The command should print the path you configured.

USB Driver

Linux systems require the packages **libusb** and **libusb-dev** to connect via USB with a Mindstorms brick. On Ubuntu, these can be installed from command prompt using

```
sudo apt-get install libusb-0.1 libusb-dev
```

To use the USB connection as non-root user, create a file */etc/udev/rules.d/70-lego.rules with the following content (4 lines in total):*

```
# Lego NXT brick in normal mode
SUBSYSTEM=="usb", DRIVER=="usb", ATTRS{idVendor}=="0694", ATTRS{idProduct}=="0002", GROUP="lego", MODE="0660"
# Lego NXT brick in firmware update mode (Atmel SAM-BA mode)
SUBSYSTEM=="usb", DRIVER=="usb", ATTRS{idVendor}=="03eb", ATTRS{idProduct}=="6124", GROUP="lego", MODE="0660"
```

Then create a group named **lego** and add your user to this group using the following commands:

```
sudo groupadd lego

sudo gpasswd -a <username> lego

sudo udevadm control --reload-rules
```

Install LeJOS

You will further need leJOS, which can be downloaded from the following URL:
https://sourceforge.net/projects/nxt.lejos.p/files/

Note, that you will need a version that fits to the flashed firmware on the brick. Currently, most of our bricks run with **version 0.9.1**.

For Linux, you have to download and extract the tar.gz file:

```
tar -xvzf leJOS_NXJ_0.9.1beta-3.tar.gz
```

Afterwards you will need to build the Java Native Library that ships with leJOS. To do so, you will need ant.

```
sudo apt-get install ant
```

Then navigate to the ./build directory in your leJOS installation and execute ant from terminal. The release should build without errors.

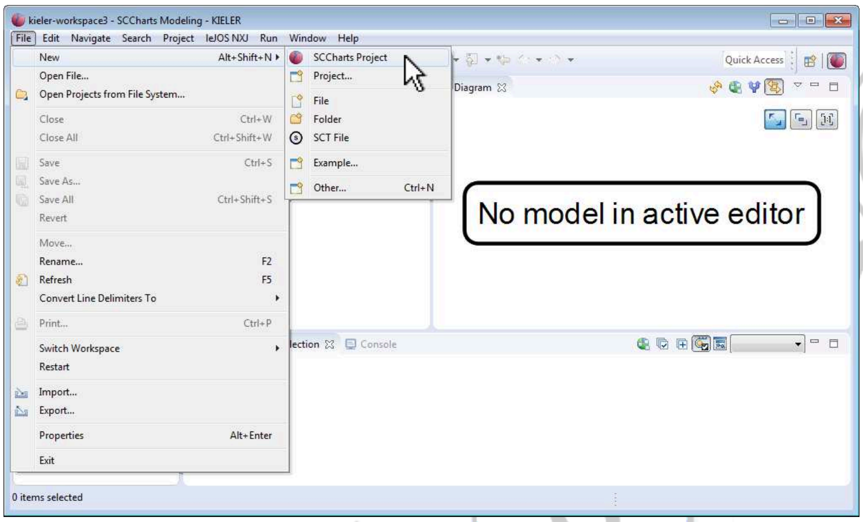
7. Exercise V: SCCharts for Lego Mindstorms

Learning Objective: Familiarize yourself with the SCCharts template environment using the example of Lego Mindstorms as an example embedded target.

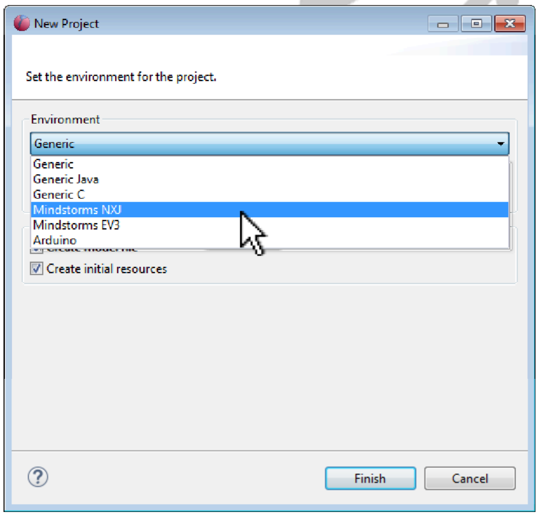
In this exercise you will create a simple SCChart which will wait on the OK button of the Lego Midstorms and output "Hello Synchron" to its display.

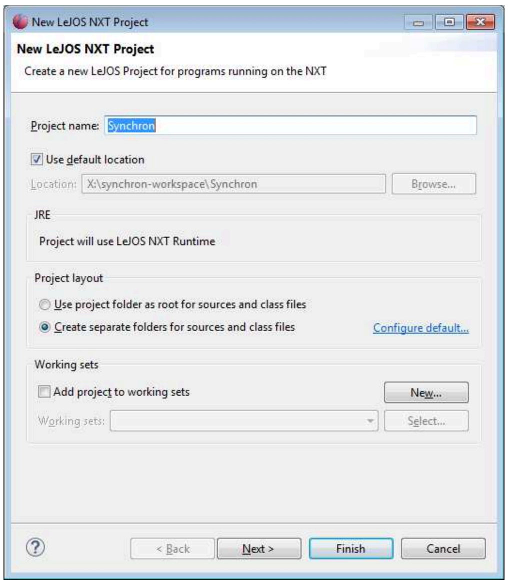
7.1 Creating an SCCharts Project

Click on "File" -> "New" -> "SCCharts Project" in the main menu.



The select "Mindstorms NXJ" and click on "Finish" (left figure). Note that on MacOSX you need to select "Mindstorms NXJ (MacOSX)".




New LeJOS NXT Project

Create a new LeJOS Project for programs running on the NXT

Project name: Synchron

[✓] Use default location

Location: X:\synchron-workspace\Synchron

JRE

Project will use LeJOS NXT Runtime

Project layout

( ) Use project folder as root for sources and class files

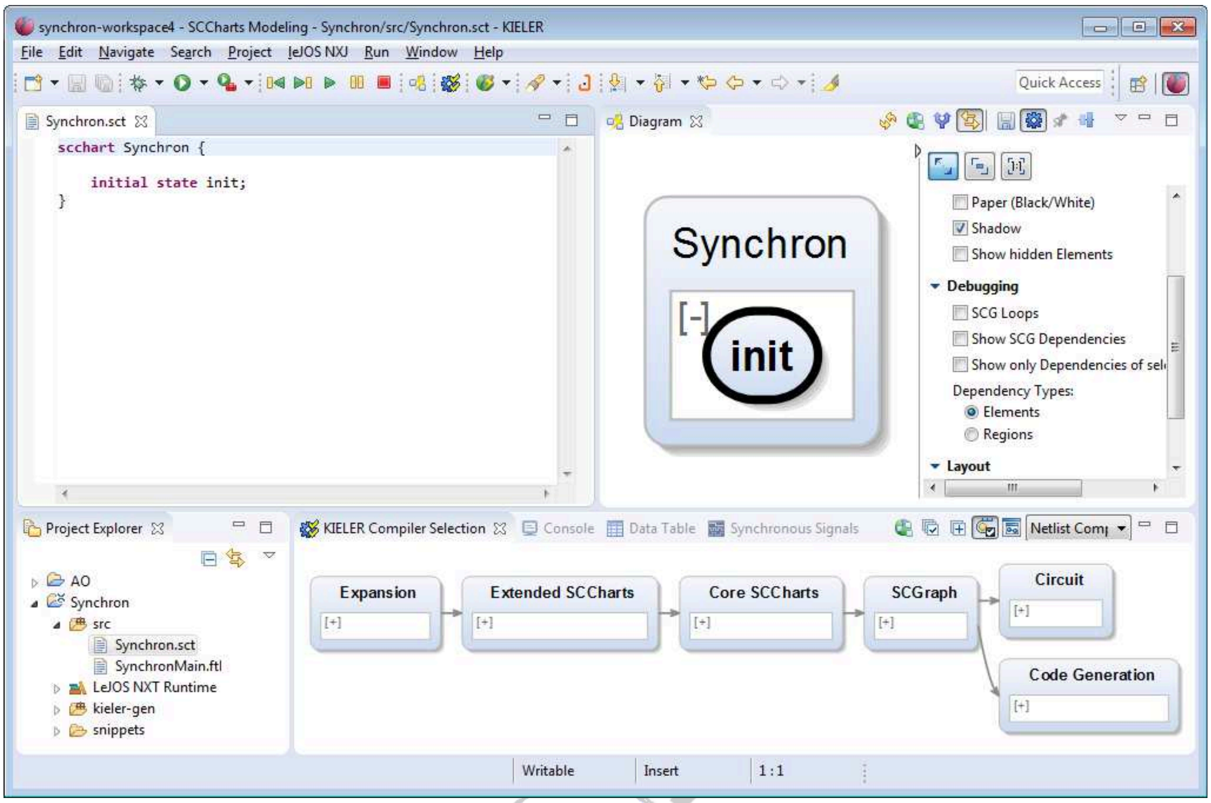
(●) Create separate folders for sources and class files

Working sets

[ ] Add project to working sets

You will be prompted another New Java Wizard now (right figure). Enter a project name, e.g., "Synchron" and again click on "Finish". **Do not switch to the Java perspective, if you are prompted to do so**. **<u>Attention:</u> Due to leJOS limitations the name of the project must not exceed 8 characters!**

The screen should look as follows:


synchron-workspace4 - SCCharts Modeling - Synchron/src/Synchron.sct - KIELER

```
scchart Synchron {

   initial state init;
}
```

- [ ] Paper (Black/White)
- [x] Shadow
- [ ] Show hidden Elements

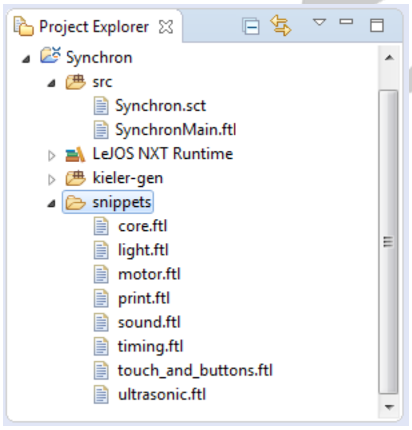
Debugging
- [ ] SCG Loops
- [ ] Show SCG Dependencies
- [ ] Show only Dependencies of sel

Dependency Types: [● Elements] ○ Regions

Writable | Insert | 1 : 1

7.2 Modeling SCCharts with Environment Snippets

You can now start modeling with SCCharts and its Lego Environment Extension. Note the "snippets" folder where you can draw from various Lego I/O examples.



Modify the example as follows:

```
scchartSynchron {

    @Wrapper Button, ENTER
    input bool isEnterDown;

    @Wrapper Print
    output string outputText;

    initial state init
    --> done with isEnterDown / outputText = "Hello Synchron";

    state done;
}
```

The wrapper annotation "Enter" will enforce that the automatically generated environment wrapper code will be mapped to the "isEnterDown" boolean variable such that "isEnterDown" is true iff the Lego Mindstorms Enter button is pressed.
The wrapper annotation "Print" will enforce that the automatically generated environment wrapper code will print the content of the "outputText" string variable on the Lego Minstorms display unit. Further snippets are available in the "snippets" folder (see above). Each snippet is a Freemarker template and comes with a short example.

Further Snippet Examples

Light-Sensor:

```
@Wrapper LightSensor, S3
input int light;
```

Motor-Actuator:

```
@Wrapper MotorSpeed, A
@Wrapper MotorSpeed, B
output int speed;
```

Time Input:

```
@Wrapper Clock, "1000"
input bool second;
```

Touch Sensor:

```
@Wrapper TouchSensor, S1
input bool GO;
```

Lamp:

```
@Wrapper RCXLamp, A
output bool blink;
```

Beep:

```
@Wrapper Beep
output bool beep;
```

Draw String to pos (x,y):

```
@Wrapper DrawString, "0", "2"
output string blackString = "black: ";
```

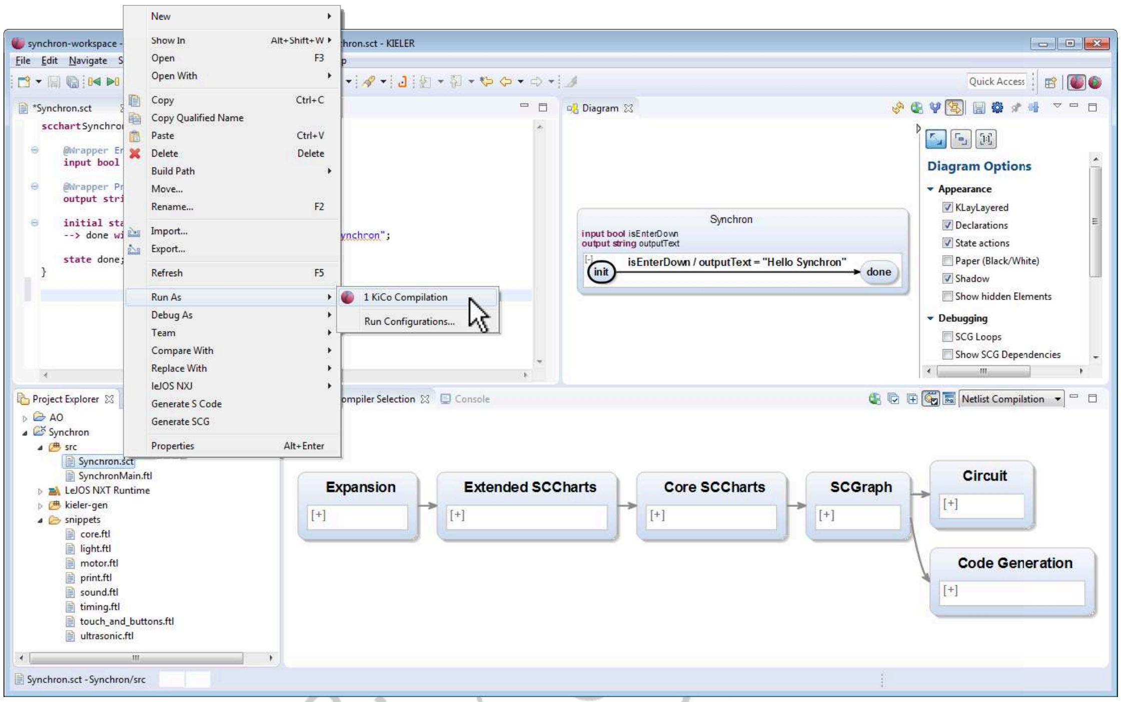
Draw Int to pos (x,y):

```
@Wrapper DrawInt, "8", "2"
output int blackValue;
```

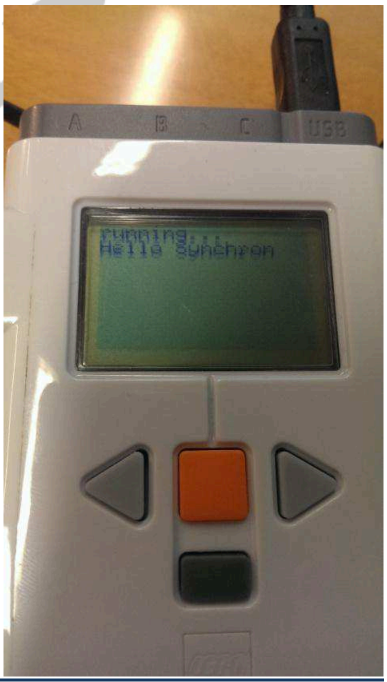
7.3 Download & Run

Please ensure that the **Lego Mindstorms brick is switched on now** and connected via a USB connection.

To upload right-click the sct file and select "Run As"->"KiCo Compilation" (see next page). This will automatically generate a Java file for the SCChart under a new "kieler-gen" folder. It will further generate environment wrapper code and try to upload and run the class files to the Lego brick.



The program should be uploaded to the brick and started. You should see the following screen on the display unit (left photo):



After pressing the (orange) Enter button, the output "Hello Synchron" should appear on the display unit (right photo).

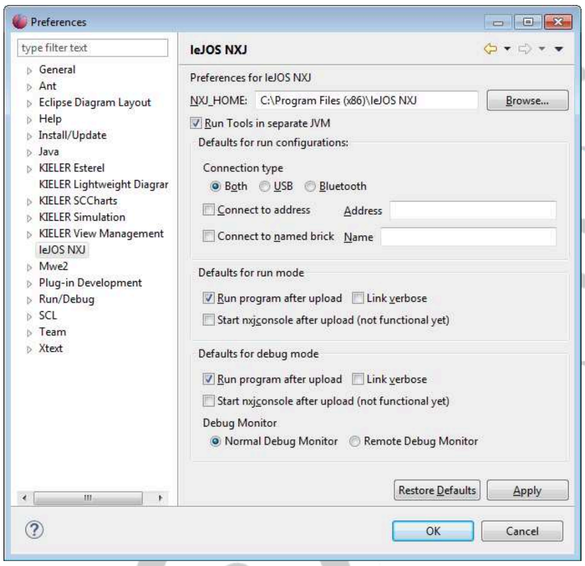
If you encounter Problems

1. If you get the following error:

```
Linking ...
Program has been linked successfully
Uploading ...
leJOS NXJ> Searching for any NXT using Bluetooth inquiry
BlueCove version 2.1.0 on winsock
leJOS NXJ> Failed to find any NXTs
leJOS NXJ> Failed to connect to any NXT
No NXT found - is it switched on and plugged in (for USB)?
```

Then most likely the NXT is not turned on (any more) or it is not connected via USB. Note that the NXT switches off automatically after some time.

2. If you have problems uploading your SCCharts to the Lego Mindstorms brick, you may need to check the LeJos configuration in the preference settings.
Click on "Window"->"Preferences" and select "leJOS NXJ". Then, check the path to the leJOS installation.


Preferences

leJOS NXJ

Preferences for leJOS NXJ

NXJ_HOME: C:\Program Files (x86)\leJOS NXJ

[✓] Run Tools in separate JVM

Defaults for run configurations:

Connection type: [● Both] ○ USB ○ Bluetooth

[ ] Connect to address Address

[ ] Connect to named brick Name

Defaults for run mode

[✓] Run program after upload [ ] Link verbose

[ ] Start nxjconsole after upload (not functional yet)

Defaults for debug mode

[✓] Run program after upload [ ] Link verbose

[ ] Start nxjconsole after upload (not functional yet)

Debug Monitor: [● Normal Debug Monitor] ○ Remote Debug Monitor

8. Exercise VI: SCCharts Pathfinder

Learning Objective: Use your knowledge about SCCharts and the KIELER SCCharts tooling to solve a more complex embedded system task.

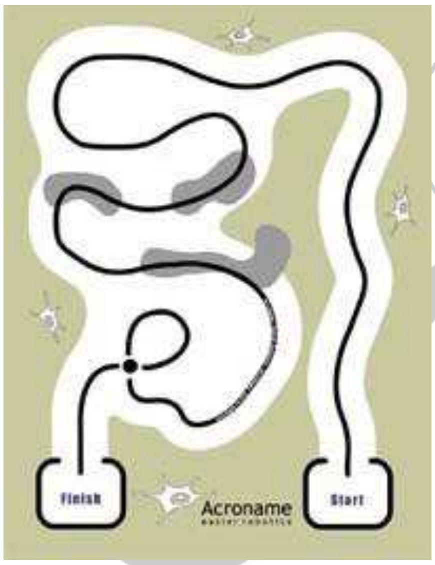
This exercise is not meant to be completely solved in the short time frame of this tutorial workshop slot.

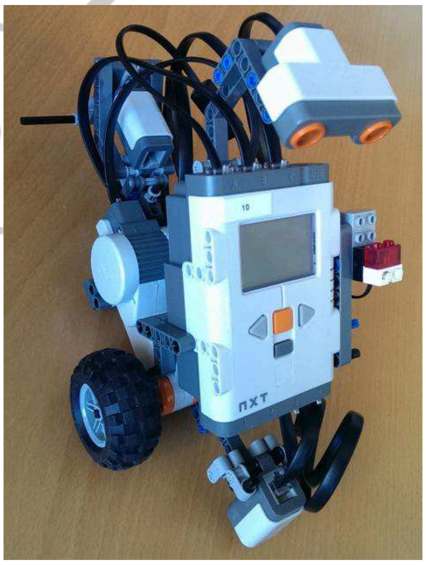
However, you are invited to solve it during the SYNCHRON workshop until Friday morning whenever time permits (breaks, evenings, ...). We will leave the robots, the Download Station, and the mat for your pleasure (hopefully ☺). We may assist you any time; just let us know.

8.1 The Task

We brought several Lego Mindstorms that were build by some of your students. Additionally, we brought you an exercise mat with a printed path.

Your task is to build an SCCharts controller for the Mindstorms robot such that the robot follows the line from "Start" to "Finish" reliably and as fast as possible.





8.2 Your Solution

We would be happy if you share your solution with us. You are happy to add comments about what you liked and what you would like us to improve in the SCCharts language and tooling.

We would appreciate if you send us your solution via email to:
kieler@informatik.uni-kiel.de

9. Questions & Feedback

We'd like to encourage you to give us your valuable feedback on anything that comes to your mind when using the KIELER SCCharts tools.

9.1 Questions

For questions, you may use the following link that takes you to our Confluence Questions section:

https://rtsys.informatik.uni-kiel.de/confluence/questions

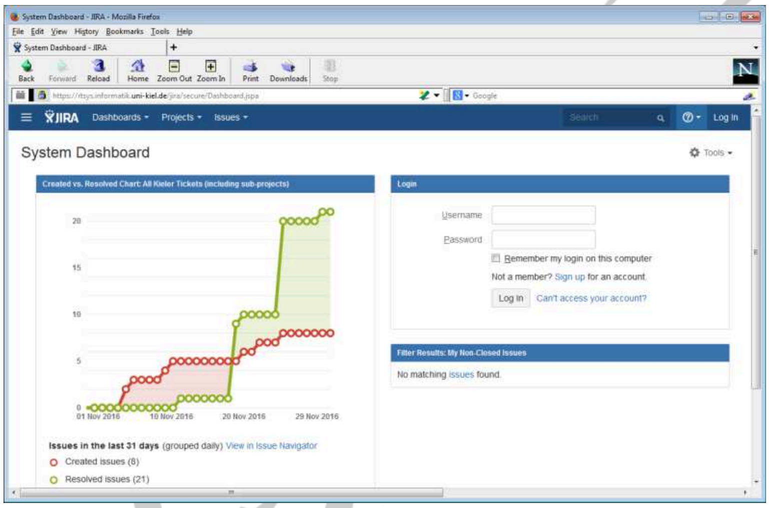
or

http://tinyurl.com/kielerquestions

9.2 Report Errors

If you encounter any errors or faulty behavior or if you have concrete feature requests, then you may use our Jira bug tracker here:

https://rtsys.informatik.uni-kiel.de/jira/ or http://tinyurl.com/kielerbug


System Dashboard - JIRA - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Back Forward Reload Home Zoom Out Zoom In Print Downloads Stop
https://rtsys.informatik.uni-kiel.de/jira/secure/Dashboard.jspa
JIRA Dashboards Projects Issues
Search
Log In
Tools
Username
Password
Remember my login on this computer
Log In
01 Nov 2016
10 Nov 2016
20 Nov 2016
29 Nov 2016

System Dashboard

Created vs. Resolved Chart: All Kieler Tickets (including sub-projects)

Issues in the last 31 days (grouped daily) View in Issue Navigator

Created issues (8)

Resolved issues (21)

Login

Not a member? Sign up for an account.

Can't access your account?

Filter Results: My Non-Closed Issues

No matching issues found.

If you do not have an account yet, please sign up for one. If you do not like to open an account, you may also send us your bug report via email: **kieler@informatik.uni-kiel.de**

9.3 Feedback

We'd also like to hear about your valuable feedback in general. We kindly invite you to participate in the following survey (see next page).

Optionally, you may also send us an email: **kieler@informatik.uni-kiel.de**

Rev. 1.1t

SCCHARTS SURVEY FORM

Have you use KIELER SCCharts **before**?

☐ **Yes**
Please let us know in the comments below, where you used it.

☐ **No**

Can you image to use KIELER SCCharts in the future?

☐ **Yes** ☐ in class ☐ in projects ☐ for fun

☐ **No**

Rate the quality of the SCCharts development tools you worked with.

Creation/**Modeling** of models:

☐ **Professional**
as other mature open source or commercial products

☐ **Advanced**
as other smaller commercial or open source products

☐ **Ok**
as beta version of commercial software or Freeware

☐ **Hardly usable**
like alpha versions or private/obsolete projects

Debugging of models:

☐ **Professional**
as other mature open source or commercial products

☐ **Advanced**
as other smaller commercial or open source products

☐ **Ok**
as beta version of commercial software or Freeware

☐ **Hardly usable**
like alpha versions or private/obsolete projects

Interactive compilation/code generation:

☐ **Professional**
as other mature open source or commercial products

☐ **Advanced**
as other smaller commercial or open source products

☐ **Ok**
as beta version of commercial software or Freeware

☐ **Hardly usable**
like alpha versions or private/obsolete projects

Wrapper\Template code generation:

☐ **Professional**
as other mature open source or commercial products

☐ **Advanced**
as other smaller commercial or open source products

☐ **Ok**
as beta version of commercial software or Freeware

☐ **Hardly usable**
like alpha versions or private/obsolete projects

Understanding the language semantics:

☐ **Professional**
as other mature open source or commercial products

☐ **Advanced**
as other smaller commercial or open source products

☐ **Ok**
as beta version of commercial software or Freeware

☐ **Hardly usable**
like alpha versions or private/obsolete projects

User interface:

☐ **Professional**
as other mature open source or commercial products

☐ **Advanced**
as other smaller commercial or open source products

☐ **Ok**
as beta version of commercial software or Freeware

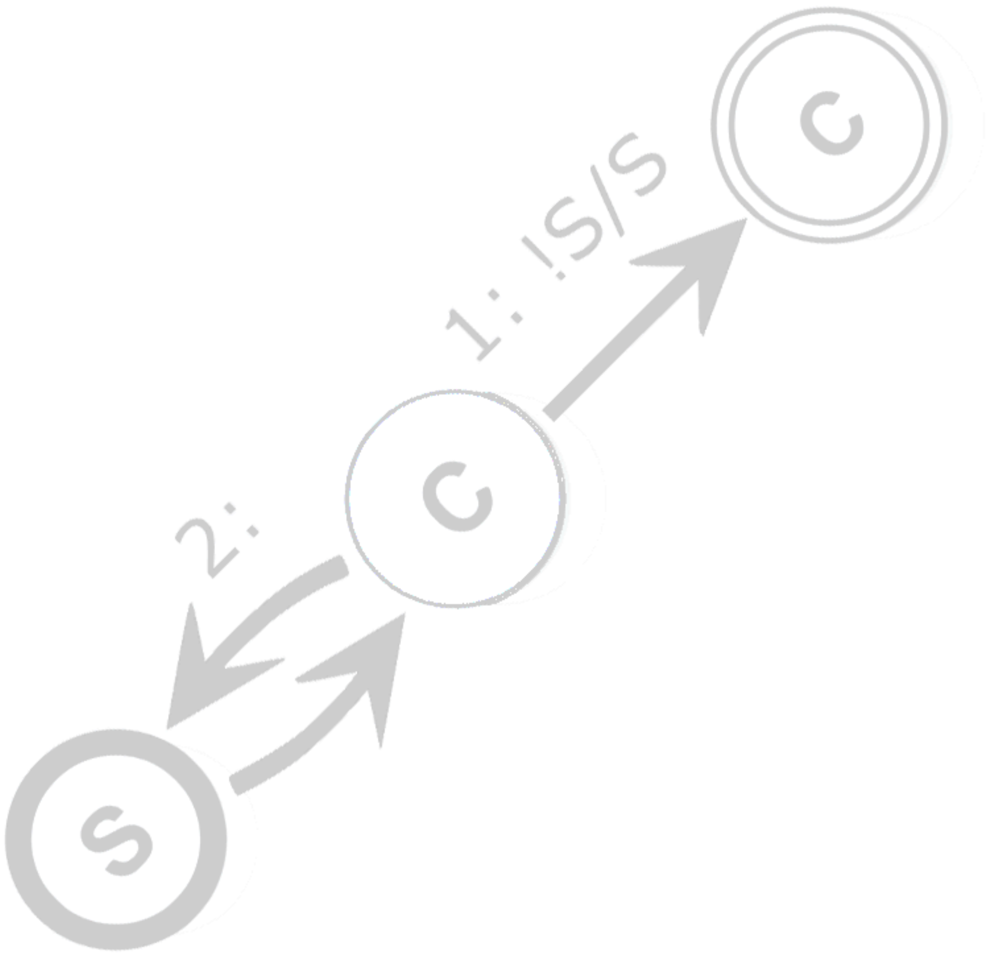
☐ **Hardly usable**
like alpha versions or private/obsolete projects

Comments on KIELER SCCharts:
(E.g., what do you think is missing for the release of the KIELER SCCharts development tools?)

Comments on this tutorial:
(E.g., what can we do to improve it? Was it helpful? Did you enjoy it?)

You can use the back page of this form to leave more detailed comments and/or suggestions.

We appreciate your feedback! Thank you very much!


2:
1: !S/S
S
C
C