SCCharts



Sequentially Constructive Statecharts (SCCharts)

SCCharts [1] is a visual synchronous language that is designed for specifying safety-critical reactive systems. SCCharts uses a new statechart notation similar to Harel Statecharts [3] and provides deterministic concurrency based on a synchronous model of computation (MoC), without restrictions common to previous synchronous MoCs like the Esterel constructive semantics [2]. Specifically, we lift earlier limitations on sequential accesses to shared variables, by leveraging the sequentially constructive MoC [4]. Thus SCCharts in short are SyncCharts [5] syntax plus Sequentially Constructive semantics.

The key features of SCCharts are defined by a very small set of elements, the Core SCCharts, consisting of state machines plus fork/join concurrency.

Conversely, Extended SCCharts contain a rich set of advanced features, such as different abort types, signals, history transitions, etc., all of which can be reduced via semantics preserving model-to-model (M2M) transformations into Core SCCharts. Extended SCCharts features are syntactic sugar because they can be expressed by a combination of Core SCCharts features.
On the one hand this eases the compilation and makes it more robust because it reduces its complexity. On the other hand, using Extended SCCharts features, a modeler is able to abstract away complexity of his or her SCCharts model which increases robustness and readability of a model. This approach enables a simple yet efficient compilation strategy and aids verification and certification.

[1] Reinhard von Hanxleden, Björn Duderstadt, Christian Motika, Steven Smyth, Michael Mendler, Joaquín Aguado, Stephen Mercer, and Owen O'Brien. SCCharts: Sequentially Constructive Statecharts for Safety-Critical Applications. In Proc. ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI'14), Edinburgh, UK, June 2014.

[2] Christian Motika, Steven Smyth, and Reinhard von Hanxleden. Compiling SCCharts — A case-study on interactive model-based compilation. In Proceedings of the 6th International Symposium on Leveraging Applications of Formal Methods, Verification and Validation (ISoLA 2014) , volume 8802 of LNCS, pages 443–462, Corfu, Greece, October 2014.

[3] Francesca Rybicki, Steven Smyth, Christian Motika, Alexander Schulz-Rosengarten, and Reinhard von Hanxleden. Interactive model-based compilation continued — interactive incremental hardware synthesis for SCCharts. In Proceedings of the 7th International Symposium on Leveraging Applications of Formal Methods, Verification and Validation (ISoLA 2016), LNCS, 2016.

[4] Reinhard von Hanxleden, Michael Mendler, Joaquín Aguado, Björn Duderstadt, Insa Fuhrmann, Christian Motika, Stepjen Mercer, and Owen O'Brien. Sequentially Constructive Concurrency - A conservative extension of the synchronous model of computation, In Proc. Design, Automation and Test in Europe Conference (DATE'13), Grenoble, France, March 2013.

[5] Charles André. Semantics of SyncCharts, Technical Report ISRN I3S/RR-2003-24-FR, I3S Laboratory, Sophia-Antipolis, France, April 2003.

[6] Gerad Berry. The foundations of Esterel, In G. Plotkin, C. Stirling, and M. Tofte, editors, Proof, Language, and Interaction: Essays in Honour of Robin Milner, pages 425-454, Cambridge, MA, USA, 2000.

[7] David Harel. Statecharts: A visual formalism for complex systems, Science of Computer Programming, 8(3):231-274, June 1987.

Documentation

- Quick Start Guide
- What's New in Version 1.0?
- Syntax
  - Annotations and Pragmas
    - leJOS
  - Object Orientation
  - Timed Automata
- Examples
- Converting Legacy Models (sct)
- For Developers
  - Building Kieler without Eclipse
  - Test and Benchmark Framework
- LEGO Mindstorms - leJOS and SCCharts
- Simulation Visualization (KViz)
- Previous Versions (Deprecated Documentation)
  - Environment Visualization (KEV)
    - Simulation Visualization using KEV
  - Execution Manager (KIEM)
    - KIEM - Launch Configuration
    - Regression Testing (JUnit & KART)
    - Simulation DataComponent
  - Interactive Timing Analysis
    - How to try out Interactive Timing Analysis
  - Kieler Compiler
    - KIELER Command Line Compiler

        - KIELER Command Line Rendering
- Leveraging Ptolemy Semantics (KlePto)
- Project Creation, Initialization and Launch (Prom)
    - Arduino and SCCharts
    - LEGO Mindstorms with leJOS and SCCharts
- Project Management (Prom)
- SCCharts (pre 1.0)
    - Command Line Compiler
    - Command Line Rendering
    - Examples (sct)
    - Online Compiler
    - PLDI'14 Artifact on SCCharts
    - Priority-Based Compilation
    - Quick Start Guide (pre 0.13)
    - Textual SCCharts Language SCT
- Simulation Visualization (KiVis)
- SyncCharts
    - SCCharts-Editor based on Yakindu
        - Comparison of the graphical representation
        - Comparison of the textual description language
        - Interfaces declarations
        - Transitions priorities
        - Project
            - User Guide
                - How to open YAKINDU-SCCharts Editor
            - Details
                - Yakindu-SCCharts textual desciption language
                - Project Progress: A SyncChart-Editor based on Yakindu
                - ABRO Example on Yakindu SCCharts Editor
        - Experiments
            - How to add Transition Types in Yakindu SCT Editor
            - How to modify Initial and Final States in Yakindu SCT Editor
        - Yakindu SCCharts / Syncchart Examples
    - The Thin Kieler SyncCharts Editor (ThinKCharts)
        - ThinKCharts User Guide
- The SC Language (SCL)
    - SC Language & Graph
    - SCL Extensions
    - SCL Transformation
    - Hardware Synthesis from SCCharts
        - Handmade ABO Implementations
        - ESO to core ESO transformation
        - ESO to VHDL Testbench
        - SCL to VHDL Transformation
        - Regressiontest - Automated Testing
- Transformation Mapping (KTM)

Downloads

Downloads - KIELER SCCharts Product

The KIELER SCCharts Product includes the SCCharts editor and compiler.

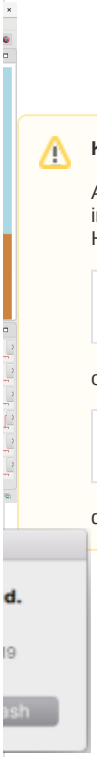
Release 1.2.0

KIELER SCCharts Product v. 1.2.0 (2021-07-05, based on Eclipse 2021-06). Available platforms:

- Linux: x86 64bit
- Mac OS X: x86 64bit
- Windows: x86 64bit

Nightly Builds

- http://rtsys.informatik.uni-kiel.de/~kieler/files/nightly/sccharts/



KIELER App on Mac OS

After downloading KIELER, Mac OS quarantines the application and, for some reason, considers the app broken and wants you to move it into the Trash.
Hence, if you want to use KIELER on your Mac, you have to remove the quarantine flag manually. You can use the following command:

```
xattr -rc com.apple.quarantine ./Kieler.app
```

or

```
xattr -rd com.apple.quarantine ./Kieler.app
```

depending on the Mac OS version