PN'2015 Advanced Tutorial: Modeling, Synthesis and Verification of Hardware

Alex Yakovlev, Victor Khomenko, Andrey Mokhov and Danil Sokolov,

Newcastle University, UK

Contact: Alex.Yakovlev@ncl.ac.uk

In this tutorial we will present an up-to-date vision of and approach to applying Petri nets and related concurrency models to modelling, verification and synthesis of electronic circuits. Petri nets are seen here as a unifying modelling language for reasoning about the behaviour of digital circuits and systems, where various application-specific and engineering-specific modelling notations can be used as front-end notations. In the first half of the tutorial we will introduce theoretical aspects of our methodology, while the second half will be devoted to the familiarisation with the toolkit Workcraft, which supports (and is constantly in the process of expansion!) a range of front-end notations for digital hardware, captured at different levels of abstraction and granularity. The examples of the front end notation we will be presenting include both structurally oriented models, such as data-flow structures and logic circuit net-lists, as well a behavioural formalisms, such as state-graphs and conditional partial order graphs.

While the spectrum of hardware that can be modelled and designed includes both synchronous (aka globally clocked) and asynchronous (aka self-timed) circuits, our main focus is on the support for designing asynchronous logic. Why? The reason for that is that self-timed circuits are inherently concurrent and the role played by Petri nets for them is as fundamental as the role played by finite-state machines in traditional synchronous design. Thus, the hardware whose behavioural semantics is conveniently underpinned by Petri nets includes digital processors, pipelines, arbiters, interfaces and controllers, and most notably digital control for analog electronics, such as power converters.

Schedule:

Morning

1) *Overview of Petri nets and Hardware Design*
2) *Basics of Asynchronous Circuit Design*
3) *Modeling of Asynchronous Circuits (High level and Low level)*
4) *Petri net Synthesis for Hardware*
5) *Verification of Asynchronous Circuits*
6) *Synthesis of Asynchronous Circuits using Signal Transition Graphs*
7) *Examples: processor, data-flow structures, arbiters, interfaces*

Afternoon

1) *Introduction to Workcraft*
2) *Design of a simple asynchronous circuit*
3) *Automated circuit synthesis*
4) *Formal verification*
5) *Advanced models/tools*
6) *Exercises and tuition*

IMPORTANT:

In order to enable hands-on work with Workcraft, we strongly recommend the participants to bring a laptop with the following capabilities:

- Operating system: Linux or Windows (MacOS is not fully supported).
- Java Runtime Environment: Oracle Java 7 or newer (Java 6 is NOT sufficient).
- The latest version of Workcraft from http://workcraft.org
- A mouse is highly recommended.

More details can be found on http://workcraft.org