Coping with concurrency in hardware: teaching experiences

Alex Yakovlev, University of Newcastle, UK
(alex.yakovlev@ncl.ac.uk, http://async.org.uk)

Extended abstract of the invited lecture at TeaCon'06 (Workshop on Teaching Concurrency, within International conferences on Petri nets and ACSD, Turku, Finland, June 2006)

In this lecture I will share my experience of teaching concurrency in hardware design context, ranging from tutorials at international conferences (both PN and circuit-oriented) down to masters courses, as well as ranging from teaching concurrency to hardware designers and teaching circuit aspects of concurrency to concurrency-knowledgeable audience.

Two types of courses have been taught by me in the last six years:

(1) Advanced Tutorials, Advanced Course Lectures, Invited Lectures on Hardware Design and Petri nets – at Petri nets conferences and Petri nets courses (predominant for "Theoretical Audience") [1]-[3].

The content of this material:

a) Basic Rationale for use of concurrency (such as Petri nets) models in hardware design particularly in the view of the key challenges of the International Roadmap for Semiconductors (ITRS), e.g. Design section of ITRS 2005 (cf. http://www.itrs.net/Common/2005ITRS/Design2005.pdf)
b) Brief introduction to the classes of Petri nets required for modelling hardware
c) Simple examples of using Petri nets in low level modelling of digital circuits (using labelled PT nets, Signal Transition Graphs) and high-level modelling of systems such as CPU (using labelled ordinary PT nets and CPNs)
d) General use of Petri net models for detecting deadlocks, safeness and liveness properties, more detailed and specific models of circuits for some hardware-specific effects such as hazards (relationship between hazard-freedom in level-based modelling and persistency, and freedom from computation interference in event-based modelling and 1-safness), atomic-level models of conflicts and metastability, difficulties with modelling true metastability and other analogue effects in circuits
e) Petri net models of AND and OR causality in hardware, relation to domain theory, distributive and semi-modular lattices on Parikh vectors.
f) Region theory and its use in hardware design, examples of using regions in deriving circuit implementations and circuit visualisation.
g) Use of true concurrency semantics and Petri net unfoldings for analysis of digital circuits.

h) Use of Time(d) Petri nets and Stochastic Petri nets for modelling timed asynchronous circuits and checking for hazards under timing assumptions, as well as performance analysis.

(Some of the) lessons learnt:

a) Theory people have difficulty in grasping intuitive relationships between circuits and formal models, e.g.
b) Theorists often stop at somewhat trivial examples of circuits, i.e. don't go to model systems with arbitration and OR causality, or models with read-arcs.
c) Theorists concentrate on complexity and decidability results rather than on practicality of algorithms that work "somehow", with lots of heuristics.

(2) Tutorials on Petri nets, STG and Synthesis tools in Asynchronous Circuits and Systems Design – at ASYNC symposia, VLSI design conferences, invited course on asynchronous design for designers in industry, mini-course on asynchronous design and Petrify tool to MSc Microelectronics students in Newcastle
(predominantly for "Circuit design" audience) [4]-[7].

The content of this material:

a) Relationship between timing diagrams and STGs and Petri nets
b) Understanding circuit behaviour by walk-through its switching events (pseudo-concurrency)
c) Basic modelling of circuits and specification of asynchronous control logic in Petri nets and STGs
d) Understanding the trace and state space semantics of circuit specifications in PNs and STGs
e) Understanding relationship between STG and state space properties and their implementability in logic (consistent state assignment, output persistence, complete state coding)
f) Deriving logic equations from state graphs obtained from STGs
g) Transformations of STGs and state graphs for state encoding conflict resolution
h) Logic decomposition and its relation to STG and Petri net levels
i) Visualisation of asynchronous behaviour in state graphs and STGs
j) Analysis of logic circuits for hazards and glitches using Petri nets, traces and state graphs
k) Performance analysis of logic circuits using Petri nets.

(Some of the) lessons learnt:

a) Designers have difficulty in abstract notion of causality, PN marking, reachability and other effects in cocncurrency formalisms, therefore the best way to teach them seems to be from the relationship to timing diagrams for low level control and interface models, and from the relationship to data and control flow graphs at the higher level.

b) The main problem with teaching electronics students is to let them understand the notions of confluence, diamonds etc in the interleaving semantics and corresponding views in true concurrency semantics. Notions of cuts and sets of states in which certain events remain enabled are very helpful.

c) Good visualisation of state spaces, e.g. with parallel graphics for diamond structures are essential. Good visualisation of firing processes and signal-like arrangement of process semantics of STGs would be essential. Use of tools such as Visual STG laboratory for STG capture and dot-based visualisation of various intermediate STG and State Graphs is crucial for understanding of the concurrent behaviours and state encoding. Visualisation of STG by means of unfoldings is also very helpful.

d) Visualisation support for pipelined structures and data-flow oriented design models is essential. Lack of tools is clear here.

e) Large scale hardware with complex concurrency-related effects such as superscalar CPUs with branch prediction, register access conflicts require adequate modelling and simulation using either domain specific tools such as those in Balsa or use of Design/CPN tools.

References:

[1] J. Cortadella, L. Lavagno and A. Yakovlev, Advanced Tutorials on Hardware Design and Petri nets, Petri nets 2000 (Aarhus) and 2005 (Miami), http://www.daimi.au.dk/pn2000/proceedings/pn2000_hardwaretut.pdf, http://www.daimi.au.dk/pn2000/proceedings/pn2000_hardwareadd.pdf, http://www.staff.ncl.ac.uk/alex.yakovlev/home.formal/tutorial_pn2005.html

[2] A. Yakovlev, “Hardware Design and Petri nets”, Advanced Course on Petri nets, Eichstaett, Sept 2003, http://www.staff.ncl.ac.uk/alex.yakovlev/home.formal/adv-course-pn2003-handouts.6up.pdf.

[3] A. Yakovlev. “Is the Die Cast for the Token Game?” (invited paper) Proc. of 23rd ICATPN, Adelaide, Australia, June 2002, LNCS 2360, Springer, pp. 70-79.

[4] J. Cortadella, M.Kishinevsky, A. Kondratyev, L. Lavagno and A. Yakovlev, “Petrify: Method and Tool for Synthesis of Asynchronous Controllers and Interfaces”, Tutorial at ASYNC 2003, Vancouver, May 2003, and Lectures at ACiD-WG Summer School, Grenoble, and Winter School, Cambridge, January 2005, http://www.staff.ncl.ac.uk/alex.yakovlev/home.formal/async03-tutorial-petrify.pdf

[5] J. Cortadella, M.Kishinevsky, A. Kondratyev, L. Lavagno and A. Yakovlev, “Synthesis of control circuits from STG specifications: Practical Exercise Manual”, July 2002, http://www.staff.ncl.ac.uk/alex.yakovlev/home.formal/async03-petrify-exercise-manual.2up.pdf.

[6] J. Cortadella, M.Kishinevsky, A. Kondratyev, L. Lavagno and A. Yakovlev, “Synthesis of control circuits from STG specifications: Problems”, July 2002, http://www.staff.ncl.ac.uk/alex.yakovlev/home.formal/async03-petrify-problems.pdf.

[7] J.Cortadella, J. Garside and A. Yakovlev, “Logic Design of Asynchronous Circuits”, Tutorial at ASPDAC/VLSI Design 2002, Bangalore, http://www.staff.ncl.ac.uk/alex.yakovlev/home.formal/talks/vlsi2002/vlsi02.zip

[8] J. Cortadella and A. Yakovlev, Proc. Workshop on Token Based Computing (ToBaCo2004), satellite to 25th Int. Conf. on Appl. and Theory of Petri nets, Bologna, Italy, 22 June 2004.

[9] A. Yakovlev, L. Gomes and L. Lavagno (Eds.) Hardware Design and Petri Nets. Kluwer Academic Publishers, Boston, ISBN 0-7923-7791-5, March 2000, 344 pp

[10] A.V. Yakovlev and A.M. Koelmans. Petri nets and Digital Hardware Design Lectures on Petri Nets II: Applications. Advances in Petri Nets, Lecture Notes in Computer Science, vol. 1492, Springer-Verlag, 1998, pp. 154-236.

[11] J. Cortadella, M. Kishinevsky, A. Kondratyev, L. Lavagno and A. Yakovlev. Logic Synthesis of Asynchronous Controllers and Interfaces, Springer Series in Advanced Microelectronics, vol. 8, Springer, 2002, ISBN-3-540-43152-7.