T-106.5840 Seminar on embedded systems Timing Analysis for Real-Time Processing
Embedded System Design Techniques for Timing Analysis

Antti P Miettinen
antti.p.miettinen@nokia.com

May 25, 2011

Abstract

Embedded system design employs a multitude of techniques for analysis, design, modelling, simulation, validation, verification and testing. The multitude of techniques reflects the cross-disciplinary nature of the area. Timing is a central topic of interest for many embedded systems. Therefore many of the emplyed techniques offer varying levels of support for timing analysis. This report lists selected embedded system design techniques relevant for timing analysis.

1 Introduction

Many of the listed techniques are from [1] which covers many specification and modelling techniques for embedded system hardware and software design and has also sections about evaluation, validation, application mapping, optimization and testing. Many techniques are very briefly covered in the book but refrences are provided for further information.

2 Techniques

2.1 Timed automata

Timed atomata [2] are extended finite state machines. The input for a timed automaton consists of timed words and time is modelled with real valued clocks. Clocks can be reset and used as constraints for transitions. Tools [3, 4] exist for analyzing models specified as timed automata.

2.2 StateCharts

StateCharts [5] is a modelling technique supporting a hierarchy for finite state machines. The model hierarchy consists of super-states and sub-states. Sub-states of a super-state can be parallel (for AND-super-states) or exclusive (for OR-super-states). The technique supports also timers via timeout transitions.

2.3 Specification and secription language

SDL [6] supports modelling with constructs including states, message passing, control flow, timers, processes and a block hierarchy describing the system structure.

2.4 Kahn process networks

Kahn process networks [7] model data flow. The nodes of a KPN model computation processes and edges the data flow between processes via FIFOs. Data sending is asynchronous and size of FIFOs is potentially infinite. Data reception is blocking.

2.5 Synchronous data flow

Synchronous data flow [8] is a restricted class of data flow specification where the number of data items produced and consumed by each computation node is specified a priori.

2.6 Petri nets

Petri nets [9, 10] consists of places, transitions and tokens. The possible states for a petri net are markings which specify the number of tokens in each place. The transitions take tokens from their source places and produce tokens to their destination places. Petri nets are partilularly suitable for modelling control flow in distributed communication. A large number of tools [11] are available for analyzing petri net models.

2.7 Hardware design languages

Timing is a central issue for hardware design. Contemporary design flows utilize testing for verifying the designs which requires extensive test bench developement. Static analysis can be used for hardware design languages [12, 13] in a manner similar to software timing analysis.

2.8 Software programming languages

Worst case execution time analysis for software is surveyed in [14]. Techniques include static analysis, abstract interpretation, implicit path enumeration, integer linear programming and measurement based methods. Using WCET as an optimization criterion for compilation is described in [15].

2.9 Operating systems

Static analysis of operating system WCET [16] suffers from unknown parameters that affect, e.g., loop bounds. For tightening the WCET estimates, application information is highly desirable [17]. Also the system can often be designed to be more friendly towards timing analysis without compromising overall performance [18].

References

[1] Peter Marwedel. *Embedded System Design, Embedded Systems, Foudations of Cyber-Physical Systems*. Springer, second edition, 2011.

[2] Rajeev Alur and David L. Dill. A theory of timed automata. *Theor. Comput. Sci.*, 126:183–235, April 1994.

[3] Marius Bozga, Conrado Daws, Oded Maler, Alfredo Olivero, Stavros Tripakis, and Sergio Yovine. KRONOS: A Model-Checking Tool for Real-Time Systems (Tool-Presentation for FTRTFT '98). In *FTRTFT'98*, pages 298–302, 1998.

[4] Kim G. Larsen, Paul Pettersson, and Wang Yi. UPPAAL in a Nutshell, 1997.

[5] David Harel. Statecharts: a visual formalism for complex systems. *Science of Computer Programming*, 8(3):231 – 274, 1987.

[6] Ferenc Belina and Dieter Hogrefe. The CCITT-specification and description language SDL. *Computer Networks and ISDN Systems*, 16(4):311 – 341, 1989.

[7] G. Kahn. The Semantics of a Simple Language for Parallel Programming. In J. L. Rosenfeld, editor, *Information Processing '74: Proceedings of the IFIP Congress*, pages 471–475. North-Holland, New York, NY, 1974.

[8] E.A. Lee and D.G. Messerschmitt. Synchronous data flow. *Proceedings of the IEEE*, 75(9):1235 – 1245, sept. 1987.

[9] C. A. Petri. *Kommunikation mit Automaten*. PhD thesis, Institut für instrumentelle Mathematik, Bonn, 1962.

[10] T. Murata. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77(4):541 –580, apr 1989.

[11] Petri Nets Tools and Software. `http://www.informatik.uni-hamburg.de/TGI/PetriNets/tools/`.

[12] Marc Schlickling and Markus Pister. A Framework for Static Analysis of VHDL Code. In Christine Rochange, editor, *7th Intl. Workshop on Worst-Case Execution Time (WCET) Analysis*, Dagstuhl, Germany, 2007. Internationales Begegnungs- und Forschungszentrum f"ur Informatik (IBFI), Schloss Dagstuhl, Germany.

[13] Stephan Thesing. Modeling a system controller for timing analysis. In *Proceedings of the 6th ACM & IEEE International conference on Embedded software*, EMSOFT '06, pages 292–300, New York, NY, USA, 2006. ACM.

[14] Reinhard Wilhelm, Jakob Engblom, Andreas Ermedahl, Niklas Holsti, Stephan Thesing, David Whalley, Guillem Bernat, Christian Ferdinand, Reinhold Heckmann, Tulika Mitra, Frank Mueller, Isabelle Puaut, Peter Puschner, Jan Staschulat, and Per Stenström. The worst-case execution-time problem —-overview of methods and survey of tools. *ACM Trans. Embed. Comput. Syst.*, 7:36:1–36:53, May 2008.

[15] H. Falk. WCET-aware register allocation based on graph coloring. In *Design Automation Conference, 2009. DAC '09. 46th ACM/IEEE*, pages 726 –731, july 2009.

[16] Mingsong Lv, Nan Guan, Yi Zhang, Qingxu Deng, Ge Yu, and Jianming Zhang. A Survey of WCET Analysis of Real-Time Operating Systems. In *Embedded Software and Systems, 2009. ICESS '09. International Conference on*, pages 65 –72, may 2009.

[17] Jörn Schneider. Why You Can't Analyze RTOSs without Considering Applications and Vice Versa. In *In Proc. of the 2nd International Workshop on Worst-Case Execution Time Analysis (WCET 2002)*, 2002.

[18] A. Marti Campoy, S. Saez, A. Perles, and J. V. Busquets. Schedulability analysis in edf scheduler with cache memories, 2004.