

INTRODUCING TIME
IN
HETEROGENEOUS MODELING AND DESIGN
FOR
SOFTWARE DEFINED RADIO

by

DANIEL LÁZARO CUADRADO



CSDR
Aalborg University (AaU)
Fredrik Bajers Vej 7, A3
DK-9220 Aalborg
Denmark

ABSTRACT

Software defined radio (SDR) includes complex mixtures of diverse functionalities such as signal processing, feedback control, sequential decision making. Ptolemy II is a good choice for simulating SDR together with the elements interfacing with the environment (such as AD and DA converters, antennas, ports, etc) and the environment itself. In order for ES systems containing SDRs to provide Quality of Service (QoS), SDR must provide with deterministic/predictable timing properties. Moreover, SDRs must provide timing information for upper layers in the system, (such as for example the application layer). The hardware on which SDR executes must provide accurate timing information. Furthermore, the software running above such platforms must be able embrace and integrate the timing information provided by the hardware layer and convey predictable timing to upper layers. Timed Automata is a theory that has been extensively used for modeling and verification of real-time systems and has successfully been applied in a variety of case studies that range from communication protocols to multimedia applications. Timed Automata is a good formalism to model control. Control functionalities can be considered as the conglomerate of the decision making in an application. Decision making involving formalisms that involve time can make an effort in guaranteeing predictability of timing properties. Therefore, it makes sense that timing information should be represented and treated in formalisms modeling control. This report proposes extending the existing FSM implementation in Ptolemy II with clocks. This will result in a Timed Automata domain implementing time automata theory [17] in Ptolemy II. The implementation can extend the existing FSM domain that provides most of the infrastructure required. This will allow timing properties to be validated by simulation for the control functionalities as opposed to verification with model checkers thus overcoming state space explosion problems while still allowing validation.

THE PROBLEM

A general definition of software defined radio (SDR) refers to wireless communication in which the modulation at the transmitter and demodulation at the receiver are performed by means of programmable hardware which is controlled by software. Often SDRs are a part of a larger embedded system (ES).

Embedded Systems is an *interdisciplinary* area that combines theories and techniques from many different fields, for instance electronics, hardware design, computer science and telecommunications [1]. Among the myriad of challenges embedded system design poses, there seems to be a general consensus that the key to overcome them is *abstractions*. To invent or apply abstractions that yield more understandable programs [2]. Better abstractions that do not abstract away from properties that are most important in an embedded system [3] such as time. Raise the levels of abstraction of designs [4]. The key problem becomes identifying the appropriate abstractions for representing the design [5]. The purpose for an abstraction is to hide the details of the implementation below and provide a platform for design from above. Furthermore, existing Computer Science (CS) paradigms have to be enriched and combined with methods found in other areas such as electronic engineering and embrace interdisciplinarity. Untimed and sequential programming abstractions are insufficient, particularly in embedded systems which are intrinsically timed and distributed. There is a need for building a new scientific foundation or simply adopting existing ones in modeling and design practice that integrates computation, physicality and *time*.

Complex mixtures of diverse functionalities such as signal processing, feedback control, sequential decision making are common in SDR which makes Ptolemy II a good choice for simulating SDR together with the elements interfacing with the environment (such as AD and DA converters, antennas, ports, etc) and the environment itself.

The Ptolemy project [6] proposes actor-oriented heterogeneous modeling and design to tackle these issues adopting several abstractions. The focus is on embedded systems, particularly those mixing interdisciplinary technologies, for example analog and digital electronics, hardware, software and mechanical devices. The aim is to model and simulate the embedded software together with the physical substrate on which it runs and the devices through which it interacts with the environment. Typically such environment cannot wait as opposed to interactive systems which are able to synchronize with it. These are called reactive systems [7]. The work is carried under the premise that no single general-purpose model of computation (MoC) is likely to emerge in the near future that will deliver what designers need. Moreover, a general purpose framework would fail in accurately modeling the widely different functionalities and their interaction. Accuracy comes with imposing constraints

resulting in a more specialized MoC unlikely to embrace any complex system. Instead, the Ptolemy project takes an approach that mixes heterogeneous MoCs, while preserving their distinct identity. It is realized by including formal MoCs in a software laboratory for experimenting with heterogeneous modeling [8] called Ptolemy II. This allows the designer to choose the MoC (or the subset of them) that best suits the target application, avoiding over- or under- specified models. Furthermore, it acknowledges the strengths and weaknesses of each MoC. The choices of MoCs strongly affect the quality of a system design. Yet, this approach poses new challenges as it has to define the semantics of the interaction of components governed by different MoCs. This is a non trivial problem area addressed by the project. Moreover, the mixtures of MoCs, convey to the designers the responsibility to cope with heterogeneity, which is alleviated by providing sophisticated, visual user interfaces.

In order for ES systems containing SDRs to provide Quality of Service (QoS), SDR must provide with deterministic/predictable timing properties. Moreover, SDRs must provide timing information for upper layers in the system, (such as for example the application layer). There are two main elements that will allow a SDR to provide with precise timing properties both coming from the hardware and software sides:

- The hardware on which SDR executes must provide accurate timing information. This is an open area of research that has recently caught some momentum. Precision Timed Machines (PRETS) [9] is one of the first attempts to develop high performance platforms with timing predictability. Prototypes of PRETS are developed on FPGAs since the implementation is close physical substrate that provides the timing information. SDR prototypes implementations are often built on FPGAs alone or combined with DSP processors and/or microcontrollers [10][11][12][13][14][15][16] (co-processing architecture).

- The software running above such platforms must be able embrace and integrate the timing information provided by the hardware layer. Furthermore, convey predictable timing to upper layers. A formalism that represents time and enables validation of such timing properties is required for such purpose.

THE SOLUTION

Timed Automata [17] is a theory has been extensively used for modeling and verification of real-time systems. It has been implemented in several verification tools like UppAal [18] and Kronos [19]. Moreover, it has successfully been applied in a variety of case studies that range from communication protocols to multimedia applications [20].

A timed automaton is a finite state machine (FSM) extended with clock variables represented as real numbers that progress synchronously.

Ptolemy represents time, an abstraction that belongs to the physical world. Time materializes in different ways in the different computational models, as *real time* that advances uniformly, either discrete or continuously, or time can be a partial order, sorted by constraints imposed by causality among events. Unluckily, the FSM domain (implementation of a MoC in PtolemyII) does not consider time in its implementation. FSM as well as Timed Automata are formalisms that are best suited to model control. Control functionalities cluster together of the decision making in an application. Decision making modeled with formalisms that involve time can make an effort in guaranteeing predictability of timing properties. Therefore, it makes sense that timing information should be represented and treated in formalisms modeling control.

This report proposes extending the existing FSM implementation in Ptolemy II with clocks. This will result in a Timed Automata domain implementing time automata theory [17]. The implementation can extend the existing FSM domain that provides most of the infrastructure required. This will allow timing properties to be validated by simulation for the control functionalities as opposed to verification with model checkers. The main problem with model checking is the exponential growth of the state space as models become larger (also known as the 'state space explosion' problem) [21]. Using simulation of Timed Automata to validate models is not limited by the size of the models or the state space explosion problem.

Issues to be considered include:

- Interfacing with other timed MoC becomes an interesting issue to be tackled. The natural scenario when modeling a SDR would be to have a Discrete Events (DE) [24] at the top level domain. Communication networks and digital hardware are some of the main application areas of DE. In discrete event simulation only the points in time at which the state of the system changes are represented. The system is modeled as a sequence of events. Each event occurs at an instant in time and marks a change of state in the system. Therefore simulation time is a totally ordered set of values each of them representing an instant in the system. Events come with a time stamp representing the associated that conveys information to the control logic for the decision making. The director keeps track of a global notion of time. Timing information inside the TA domain can be obtained from both the events arriving to it as depicted in Figure 1 or from the external DE director.

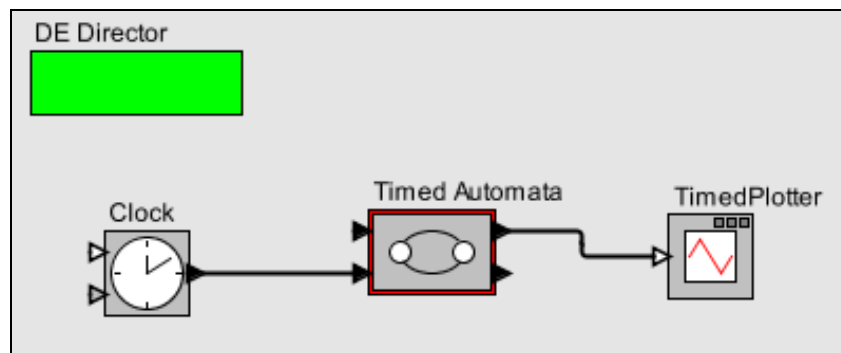


FIGURE 1: TIME FLOW FROM THE DE MoC TO A TIMED AUTOMATA

The decision making can be transferred to the outside through events or refining the different states as in *modal models* [23]. Since clocks in a timed automata progress synchronously modification of their values due to external time updates are straightforward.

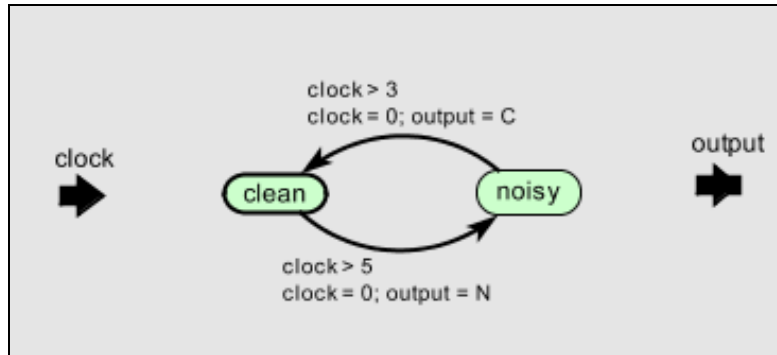


FIGURE 2: MODAL TIMED AUTOMATA

When interacting with continuous time (CT) [24] semantics and specially DA and AD converters, several actors are provided by Ptolemy for conversion. `ZeroOrderHold` converts events to continuous timed signals by holding the value of the discrete event until the next discrete event arrives. `PeriodicSampler` generates discrete events by periodically sampling a continuous signal. A sampling period can be specified.

- The expression language is used to specify guards and actions. In principle clocks can be considered as any other variable in the system so no further modification should be required. For the clocks to be used as expressions, they must be declared as `Parameters` and be in scope. `PortParameters` [22] provide default values and can be used to update the value of clocks on the arrival of events

- A hook for verification through model checking with for example UppAal will allow performing combined validation and verification of control functionalities. For that purpose either a model exchange data format must be created or conversions between the existing formats.

By creating SDR models that combine TA, DE and CT semantics, timing properties can be utilized by the control logic to provide predictable timing properties. Those can be then conveyed to other layers or subsystems of the containing ES. The size limitations in model checking as a result of the state space explosion can be overcome by validation through simulation. A hook for UppAal can allow verification of critical control logic parts. A combination of verification and validation of the overall system combined will enable higher robustness and predictability.

REFERENCES

- [1] F. Vahid, T. Givargis
Embedded System Design - A Unified Hardware/Software Introduction
John Wiley & Sons, Inc., 2002
- [2] E. A. Lee
What are the key challenges in Embedded Software?
System Design Frontier, Volume 2, Number 1, January 2005
<http://www.hwswworld.com/>
- [3] A. P. Black
Better Abstractions: an Agenda for Embedded Systems Research
Panel Presentation at DARPA-NSF Workshop on Embedded & Hybrid Systems
- [4] Artist
Embedded Systems Design. The ARTIST Roadmap for Research and Development
Lecture Notes in Computer Science, Vol 3436.
<http://www.artist-embedded.org>
- [5] E. A. Lee
What's Ahead for Embedded Software?
IEEE Computer, September 2000, p 18-26.
- [6] Department of EECS, UC Berkeley
Ptolemy Project
<http://ptolemy.eecs.berkeley.edu/>
- [7] D. Harel, A. Pnuelli
On the development of reactive systems.
NATO Advanced Study Institute on Logics and Models for Verification and Specification of Concurrent Systems.
Springer Verlag, 1985.
- [8] C. Brooks et. al
Heterogeneous Concurrent Modeling and Design in Java. Volume 1: Introduction to Ptolemy.
EECS Department, University of California, Berkeley, UCB/EECS-2007-7, 2007.
- [9] S. Edwards, E A. Lee
The Case for the Precision Timed (PRET) Machine
EECS Department, UC, Berkeley, Tech. Report No. UCB/EECS-2006-149, 2006
- [10] Y. Zhuan, J. Grosspietsch, G. Memik
An FPGA Based All-Digital Transmitter with Radio Frequency Output for Software Defined Radio
Design, Automation & Test in Europe 2007 (DATE '07) 16-20 April 2007 Page(s):1 – 6
- [11] Di Stefano, A.; Fiscelli, G.; Giaconia, C.G.;
An FPGA-Based Software Defined Radio Platform for the 2.4GHz ISM Band
Research in Microelectronics and Electronics 2006, Ph. D. 12-15 June 2006 Page(s):73 – 76
- [12] Myler, H.R.; Bagasrawala, S.A.; Narayana, N.V.;
A concurrent processing approach for software defined radio baseband design
Technical, Professional and Student Development Workshop, 2005, Page(s):20 – 24
- [13] Barrandon, L.; Crand, S.; Houzet, D.;
Behavioral modeling and simulation of mixed signal front-end for software defined radio terminals
Industrial Electronics, 2004 IEEE International Symposium 2004 Page(s):181 - 185 vol. 1
- [14] Reves, X.; Marojevic, V.; Ferrus, R.; Gelonch, A.;
FPGA's middleware for software defined radio applications

- [15] *Field Programmable Logic and Applications, 2005. International Conference* Page(s):598 – 601
Blaickner, A.; Albl, S.; Scherr, W.;
Configurable computing architectures for wireless and software defined radio - a FPGA prototyping experience using high level design-tool-chains
System-on-Chip, 2004. Proceedings. 2004 International Symposium Page(s):111 – 116
- [16] Altera
Software Defined Radio
<http://www.altera.com/end-markets/wireless/software/sdr/wir-sdr.html>
- [17] Rajeev Alur, David L. Dill
A Theory of Timed Automata
Theoretical Computer Science, 126:183-235, 1994.
- [18] Kim G. Larsen, Paul Pettersson and Wang Yi
Uppaal in a Nutshell
Springer International Journal of Software Tools for Technology Transfer 1(1+2), 1997.
- [19] S. Yorvine
Kronos: a verification tool for real-time systems
Journal on Software Tools for Technology transfer
- [20] Gerd Behrmann, Alexandre David, and Kim G. Larsen
A Tutorial on Uppaal
In proceedings of SFM-RT'04. LNCS 3185.
- [21] Hendriks, Martijn
Model checking timed automata : techniques and applications
Ph. D. Thesis, Radboud University.
http://webdoc.ubn.ru.nl/mono/h/hendriks_m/modectia.pdf
- [22] C. Brooks et. al
Heterogeneous Concurrent Modeling and Design in Java. Volume 1: Introduction to Ptolemy.
EECS Department, University of California, Berkeley, UCB/EECS-2007-7, 2007.
- [23] C. Brooks et. al
Heterogeneous Concurrent Modeling and Design in Java. Volume 2: Ptolemy II Software Architecture.
EECS Department, University of California, Berkeley, UCB/EECS-2007-8, 2007
- [24] C. Brooks et. al
Heterogeneous Concurrent Modeling and Design in Java. Volume 3: Ptolemy II Domains.
EECS Department, University of California, Berkeley, UCB/EECS-2007-9, 2007