Home

PRETSY - Precision-Timed Synchronous Reactive Processing

The PRETSY project is a project funded by the German Research Foundation (DFG HA 4407/6-1, ME 1427/6-1), running 2012 - 2015 and 2015 - 2018.

Summary:

Embedded reactive real-time systems are ubiquitous today, and provide increasingly complex functionality for example in modern automotive, avionics or medical products. This rising complexity makes it important to apply high-level design approaches, which, however, traditionally make critical low-level aspects such as timing hard to control. This project investigates a novel, holistic approach for the design of timing-predictable, efficient reactive systems, which considers the modelling and programming level as well as the execution platform.

A key aim is to combine a formal semantical basis, which is provided by the synchronous model of computation and which results in predictable reactive control flow, with recent architectural developments that offer predictable timing at the instruction level. Compared to typical design approaches today, based on C-like languages and processors that optimise the average case at the expense of predictability, the objective is to reduce timing uncertainties at the control-flow level as well as the architectural level. On the practical side the project is developing a model-based design flow and tool chain for implementing timing-predictable, reactive systems, including a synchronous modelling and programming language, a compiler, a timing analyser, and a predictable execution platform derived from the Berkeley/Columbia PRET architecture.

PRETSY (poster)

Coordinators:

Reinhard v. Hanxleden, Kiel University

Michael Mendler, Bamberg University

Collaborating Scientists:

Stephen Edwards, Columbia University

Alain Girault, INRIA Grenoble

Edward Lee, UC Berkeley

Gerald Lüttgen, Bamberg University

Marc Pouzet, ENS Paris

Partha Roop, University of Auckland

Zoran Salcic, University of Auckland

Reinhard Wilhelm, Saarland University

Internal Documents:

Internal Documents

Mailinglists:

If you wish to receive important project updates and announcements in low-traffic mode, this mailing list is just for you:

pretsy mailinglist

Active group members and developers should subscribe to:

pretsy-devel mailinglist

Papers:

[FMSD'12] **Constructive Boolean circuits and the exactness of timed ternary simulation.**

M. Mendler, T. R. Shiple, G. Berry.

Formal Methods in System Design (FMSD), Volume 40, Issue 3, June 2012: 283-329.

> **Abstract**. We classify gate level circuits with cycles based on their stabilization behavior. We define a formal class of combinational circuits, the *cons tructive* circuits, for which signals settle to a unique value in bounded time, for any input, under a simple conservative delay model, called the *up-bounded non-inertial (UN) delay*. Since circuits with combinational cycles can exhibit asynchronous behavior, such as non-determinism or metastability, it is crucial to ground their analysis in a formal delay model, which previous work in this area did not do.

We prove that *ternary simulation*, such as the practical algorithm proposed by Malik, decides the class of *constructive* circuits. We prove that three-valued algebra is able to maintain correct and exact stabilization information under the UN-delay model, and thus provides an adequate electrical interpretation of Malik's algorithm, which has been missing in the literature. Previous work on combinational circuits used the *upbounded inertial (UI) delay* to justify ternary simulation. We show that the match is not exact and that stabilization under the UI-model, in general, cannot be decided by ternary simulation. We argue for the superiority of the UN-model for reasons of complexity, compositionality and electrical adequacy. The UN-model, in contrast to the UI-model, is consistent with the hypothesis that physical mechanisms cannot implement non-deterministic choice in bounded time.

As the corner-stone of our main results we introduce *UN-Logic*, an axiomatic specification language for UN-delay circuits that mediates between the real-time behavior and its abstract simulation in the ternary domain. We present a symbolic simulation calculus for circuit theories expressed in UN-logic and prove it sound and complete for the UN-model. This provides, for the first time, a correctness and exactness result for the timing analysis of cyclic circuits. Our algorithm is a timed extension of Malik's pure ternary algorithm and closely related to the timed algorithm proposed by Riedel and Bruck, which however was not formally linked with real-time execution models.

[DATE'13] **Sequentially Constructive Concurrency: A Conservative Extension of the Synchronous Model of Computation.** (slides)

R. von Hanxleden, M. Mendler, J. Aguado, B. Duderstadt, I. Fuhrmann, C. Motika, S. Mercer, O. O'Brian.

Design, Automation Test in Europe Conference (DATE'13), March 2013: 581-586.

Abstract. Synchronous languages ensure deterministic concurrency, but at the price of heavy restrictions on what programs are considered valid, or constructive. Meanwhile, sequential languages such as C and Java offer an intuitive, familiar programming paradigm but provide no guarantees with regard to deterministic concurrency. The sequentially constructive (SC) model of computation (MoC) presented here harnesses the synchronous execution model to achieve deterministic concurrency while addressing concerns that synchronous languages are unnecessarily restrictive and difficult to adopt.

In essence, the SC MoC extends the classical synchronous MoC by allowing variables to be read and written in any order as long as sequentiality expressed in the program provides sufficient scheduling information to rule out race conditions. This allows to use programming patterns familiar from sequential programming, such as testing and later setting the value of a variable, which are forbidden in the standard synchronous MoC. The SC MoC is a conservative extension in that programs considered constructive in the common synchronous MoC are also SC and retain the same semantics. In this paper, we investigate classes of shared variable accesses, define SC admissible scheduling as a restriction of "free scheduling," derive the concept of sequential constructiveness, and present a priority-based scheduling algorithm for analyzing and compiling SC programs efficiently.

[CSIJC'13] **A Game Semantics for Instantaneous Esterel Reactions.** (pdf)

J. Aguado.

CSI Journal of Computing (CSIJC). Vol.2, No.1-2, e-ISSN 2277-7091, 2013: 30-44.

Abstract. The present paper explains the constructive semantics of the synchronous language Esterel in terms of winning strategies in finite two–player games. The kernel fragment of Esterel considered here corresponds to combinational circuits and includes the statements for emission, signal test, parallel composition and gives for the first time a game–theoretic interpretation to sequential composition and local signal declaration. This modelling shows how non-classical truth values induced by logic games can be used to characterise the constructive single-step semantics for Esterel.

[ESOP'14] **Grounding Synchronous Deterministic Concurrency in Sequential Programming**. (slides)

J. Aguado, M. Mendler, R. von Hanxleden, I. Fuhrmann.

Proceedings of the 23rd European Symposium on Programming (ESOP'14), April 2014: 229-248.

Abstract. Using a new domain-theoretic characterisation we show that Berry's constructive semantics is a conservative approximation of the recently proposed *sequentially constructive* (SC) model of computation. We prove that every Berry-constructive program is deterministic and deadlock-free under sequentially admissible scheduling. This gives, for the first time, a natural interpretation of Berry-constructiveness for shared-memory, multi-threaded programming in terms of synchronous cycle-based scheduling, where previous results were cast in terms of synchronous circuits. This opens the door to a direct mapping of Esterel's signal mechanism into Boolean variables that can be set and reset under the programmer's control within a tick. We illustrate the practical usefulness of this mapping by discussing how *signal reincarnation* is handled efficiently by this transformation, which is of linear complexity in program size, in contrast to earlier techniques that had quadratic overhead.

[PLDI'14] **SCCharts: Sequentially Constructive Statecharts for Safety-Critical Applications: HW/SW-Synthesis for a Conservative Extension of Synchronous Statecharts.**

R. von Hanxleden, B. Duderstadt, C. Motika, S. Smyth, M. Mendler, J. Aguado, S. Mercer, O. O'Brian.

ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI'14), June 2014.

Abstract. We present a new visual language, SCCharts, designed for specifying safety-critical reactive systems. SCCharts use a statechart notation and provide determinate concurrency based on a synchronous model of computation (MoC), without restrictions common to previous synchronous MoCs. Specifically, we lift earlier limitations on sequential accesses to shared variables, by leveraging the sequentially constructive MoC. The semantics and key features of SCCharts are defined by a very small set of elements, the Core SCCharts, consisting of state machines plus fork/join concurrency. We also present a compilation chain that allows efficient synthesis of software and hardware.

[TECS'14] **Sequentially Constructive Concurrency — A Conservative Extension of the Synchronous Model of Computation.**

R. von Hanxleden, M. Mendler, J. Aguado, B. Duderstadt, I. Fuhrmann, C. Motika, S. Mercer, O. O'Brian, P. Roop.

ACM Transactions on Embedded Computing Systems (ACM TECS) — Special Issue on Applications of Concurrency to System Design, Vol. 13, Issue 4s, June 2014: 144:1-144:26

Abstract. This is a journal extended version of the [DATE'13] publication which includes in addition: (i) a description of the mapping from the Sequentially Constructive (SC) language to the SC graph (SCG); (ii) detailed discussions on thread and statement reincarnation; (iii) a full section on the formalisation of SC based on free scheduling of SCGs; (iv) a more general SC Model of Computation based on the notion of confluence; (v) a revised (positive) definition of SC-Admissibility; (vi) definition of valid SC-schedules; (vii) proof that every ASC schedulable program is indeed SC; (viii) detailed discussion on conservative approximations and (ix) additional examples for illustrating ineffective writes, failure despite deterministic outcome, data-dependency of SC and enforced determinism via reduction of admissible runs.

[ISoLA'14] ***Compiling SCCharts — a case-study on interactive model-based compilation.***

C. Motika, S. Smyth, and R. von Hanxleden.

Proceedings of the 6th International Symposium on Leveraging Applications of Formal Methods, Verication and Validation, LNCS 8802, October 2014: 443-462.

Abstract. SCCharts is a recently proposed statechart language designed for specifying safety-critical reactive systems. We have developed an Eclipse-based compilation chain that synthesizes SCCharts into either hardware or software. The user edits a textual description which is visualized as SCChart and subsequently transformed into VHDL or C code via a series of model-to-model (M2M) transformation steps. An interactive environment gives the user control over which transformations are applied and allows the user to inspect intermediate transformation results. This *Si ngle-Pass Language-Driven Incremental Compilation* (SLIC) approach should conceptually be applicable to other languages as well. Key benets are: (1) a compact, light-weight denition of the core semantics, (2) intermediate transformation results open to inspection and support for certication, (3) high-level formulations of transformations that define advanced language constructs, (4) a divide-and-conquer validation strategy, (5) simplied language/compiler subsetting and DSL construction.

Workshops/Seminars:

[SYNCHRON'12] **Sequentially Constructive Concurrency: A Conservative Extension of the Synchronous Model of Computation**. (slides)

M. Mendler.

Summary: *Presentations of the [DATE'13] and [TECS'14] publications respectively.*

Also presented at

- 19th International Workshop on Synchronous Programming (SYNCHRON'12), Le Croisic, France. November 2012
- Electrical Engineering Research Seminar "PRETzel Forum" (PRETzel'14), Auckland University, New Zealand, January 2014.

[PDAY'13] **Is timing analysis a refinement of causality analysis?** (slides)

J. Aguado.

PRET Day, Inria Grenoble Rhône-Alpes France, March 2013.

Summary: It has been recognised that the synchronous model of computation together with a suitable execution platform (precision-timed, or PRET architectures) facilitates system-level timing predictability. This talk discusses a logical and game-theoretic framework for capturing worst-case reaction time (WCRT) for Esterel-style synchronous reactive programming. This framework will provide a formal grounding for the WCRT problem, and allow to improving upon earlier heuristics by accurately and modularly characterising timing interfaces. This approach will not only allows verifying the correctness of WCRT analyses methods, but also will allow capturing functionality and timing together.

[PDAY'13] **Constructive Boolean Networks and the Exactness of Timed Ternary Simulation.** (slides)

M. Mendler.

Summary: Presentations of materials included in the [FMSD'12] publication.

Cyclic boolean systems, such as those arising in asynchronous circuits the semantics of synchronous programming languages, do not admit a unique canonical execution semantics. Instead, different approaches impose different restrictions on their stabilization behavior. This talk concerns the class of constructive circuits, for which signals settle to a unique value in bounded time, for any input, under a simple conservative delay model, called the up-bounded non-inertial (UN) delay. The main result is that ternary simulation decides the class of constructive circuits. It shows that three-valued algebra is able to maintain correct and exact stabilization information under the UN-delay model, which thus provides an adequate electrical interpretation of ternary algebra, which has been missing in the literature. Previous work on combinational circuits used the up-bounded inertial (UI) delay to justify ternary simulation. It can be shown that the match is not exact and that stabilization under the UI-model, in general, cannot be decided by ternary simulation. As the corner-stone of our main results we introduce UN-Logic, an axiomatic specification language for UN-delay circuits that mediates between the real-time behavior and its abstract simulation in the ternary domain. We present a symbolic simulation calculus for circuit theories expressed in UN-logic and prove it sound and complete for the UN-model. This provides, for the first time, a correctness and exactness result for the timing analysis of cyclic circuits such as the timed extension of Malik's or Brzozowski and Seger's pure ternary algorithm or the timed algorithm proposed by Riedel and Bruck, which were not formally linked with real-time execution models.

Also presented at:

- [D-CON'13] **Constructive Boolean Circuits and the Exactness of Timed Ternary Simulation.**
  German Chapter Concur 2013, Institute for Software Engineering and Programming Languages, University of Lübeck, Germany, March 2013.
- [ECERS'14] **Constructive Circuits and the Synchrony Hypothesis.**
  Joint Electrical Engineering and Computer Science Departmental Seminar, Auckland University, New Zealand, January 2014.

[PTCONF'13] **SCCharts: Sequentially Constructive Charts.** (poster)

C. Motika, S. Smyth, R. von Hanxleden, M. Mendler.

10th Biennial Ptolemy Miniconference (PTCONF'13), Berkeley, CA, USA, November 2013.

Summary: This poster describes Sequentially Constructive Charts (SCCharts), the visual language for specifying safety-critical reactive systems employed in the [PLDI'14] publication.

[SYNCHRON'13] **Compiling SCCharts (and other Sequentially Constructive Programs) to Hardware and Software.**

R. von Hanxleden.

20th International Workshop on Synchronous Programming (SYNCHRON'13), Dagstuhl Germany, November 2013.

Summary: SCCharts extend SyncCharts with sequential constructiveness (SC) and other features. We developed a compilation chain that first, in a high-level compilation phase, performs a sequence of model-to-model transformations at the SCCharts-level such that they can be mapped directly to SC Graphs (SCGs). Then two alternative low-level compilation approaches allow mapping to hardware and software; the circuit approach generates a netlist, the priority approach simulates concurrency with interleaved threads.

[SYNCHRON'13] **SCCharts – Sequentially Constructive Charts**. (slides)

C. Motika.

20th International Workshop on Synchronous Programming (SYNCHRON'13), Dagstuhl Germany, November 2013.

Summary: We present a new visual language, SCCharts, designed for specifying safety-critical reactive systems. SCCharts uses a new statechart notation similar to Harel Statecharts and provides deterministic concurrency based on a synchronous model of computation (MoC), without restrictions common to previous synchronous MoCs like the Esterel constructive semantics. Specifically, we lift earlier limitations on sequential accesses to shared variables, by leveraging the sequentially constructive MoC. Thus SCCharts in short are SyncCharts 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.

[SYNCHRON'13] **Berry-Constructive Programs are Sequentially Constructive, or: Synchronous Programming from a Scheduling Perspective.** (slides)

M. Mendler.

20th International Workshop on Synchronous Programming (SYNCHRON'13), Dagstuhl Germany, November 2013.

Summary: We introduce an abstract value domain I(D) and associated fixed point semantics for reasoning about concurrent and sequential variable valuations within a synchronous cycle-based model of computation. We use this domain for a new behavioural definition of Berry's causality analysis for Esterel in terms of approximation intervals. This gives a compact and more uniform understanding of causality and generalises to other data-types. We also prove that Esterel's ternary domain and its semantics is conservatively extended by the recently proposed sequentially constructive (SC) model of computation. This opens the door to a direct mapping of Esterel's signal mechanism into boolean variables that can be set and reset arbitrarily within a tick. We illustrate the practical usefulness of this mapping by discussing how signal reincarnation is handled efficiently by this transformation, which is of complexity that is linear in program size, in contrast to earlier techniques that had, at best, potentially quadratic overhead.

[RePP'14] **Towards Interactive Timing Analysis for Designing Reactive Systems.** (slides)

I. Fuhrmann (talk), David Broman, Steven Smyth, Reinhard von Hanxleden.

Workshop on Reconciling Performance with Predictability (RePP'14), Grenoble France, April 2014.

Summary: Reactive systems are increasingly developed using high-level modeling tools. Such modeling tools may facilitate formal reasoning about concurrent programs, but provide little help when timing-related problems arise and deadlines are missed when running a real system. In these cases, the modeler has typically no information about timing properties and costly parts of the model; there is little or no guidance on how to improve the timing characteristics of the model. Here, we propose a design methodology where interactive timing analysis is an integral part of the modeling process. This methodology concerns how to aggregate timing values in a user-friendly manner and how to define timing analysis requests. We also introduce and formalize a new timing analysis interface that is designed for communicating timing information between a high-level modeling tool and a lower-level timing analysis tool.

[RePP'14] **Worst Case Reaction Time analysis of Synchronous Programs: Studying the Tick Alignment Problem.** (slides)

M. Mendler.

Workshop on Reconciling Performance with Predictability (RePP'14), Grenoble France, April 2014.

Summary: Synchronous programs are ideally suited for the design of safety critical systems as they provide guarantees on determinism and deadlock freedom. In addition to such functional guarantees, guarantees on timing are also necessary. In this talk, we study the problem of static worst case reaction (WCRT) time analysis of synchronous programs. While, there have been many recent attempts at addressing this problem from the point of view of scalability and precision, one crucial aspect is yet to be examined from a fundamental viewpoint. Concurrent threads in a synchronous program must align during every reaction, a problem that has been termed as the tick alignment problem (TAP), i.e., infeasible ticks that never align in practice must be ruled out for precision. We, for the first time, study TAP in the guise of a number theoretic formulation in order to not only explore its lower bound complexity, but also to develop heuristics that work well in practice.

Technical Reports:

[TR1308] **Sequentially Constructive Concurrency. A Conservative Extension of the Synchronous Model of Computation.** (pdf)

R. von Hanxleden, M. Mendler, J. Aguado, B. Duderstadt, I. Fuhrmann, C. Motika, S. Mercer, O. O'Brien, P. Roop.

Technical Report 1308, Christian-Albrechts-Universitaet zu Kiel, Department of Computer Science, ISSN 2192-6247, August 2013.

[TR1311] **SCCharts: Sequentially Constructive Statecharts for Safety-Critical Applications.** (pdf)

R. von Hanxleden, B. Duderstadt, C. Motika, S. Smyth, M. Mendler, J. Aguado, S. Mercer, and O. O'Brien.

Technical Report 1311, Christian-Albrechts-Universitaet zu Kiel, Department of Computer Science, ISSN 2192-6247 December 2013.

[EECS-14-26] **Towards Interactive Timing Analysis for Designing Reactive Systems.** (pdf)

I. Fuhrmann, D. Broman, S. Smyth, R. von Hanxleden.

Technical Report No. UCB/EECS-2014-26, University of California, Berkeley, EECS Department, April 2014.

[TR94] **Grounding Synchronous Deterministic Concurrency in Sequential Programming.** (pdf)

J. Aguado, M. Mendler, R. von Hanxleden, I. Fuhrmann.

Bamberger Beiträge zur Wirtschaftsinformatik und Angewandten Informatik Nr. 94, Bamberg University, ISSN 0937-3349, August 2014.

[TR95] **WCRT analysis of Synchronous Programs: Studying the Tick Alignment Problem** (pdf)

M. Mendler, B. Bodin, P. S. Roop, J.-J. Wang.

Bamberger Beiträge zur Wirtschaftsinformatik und Angewandten Informatik Nr. 95, Bamberg University, ISSN 0937-3349, August 2014.