Esprit Working Group 8533

NADA — New Hardware Design Methods

Periodic Progress Report May 1996 to August 1997

and Final Report

Contents

Part I
Annual Report

1 Overview

1.1 Goals

The topic of NADA was research on new, mathematically sound methods for the description and design of hardware systems. The term "hardware systems" was interpreted very generally to include architectures and circuits, the hardware/software interface and even biological systems.

Three major subtopics were identified within the Inaugural Meeting in April 1994:

- Modelling;
- NGHDL and design techniques;
- Foundations.

The goal of the research on modelling was to elicit requirements on design methodologies and description languages. It included studying architectures, circuits and emerging new paradigms for hardware systems, as well as various standard technologies.

The work on NGHDL, a Next Generation Hardware Description Language, served to exhibit concepts at a high level of abstraction for shich a clean formal semantics can be given. The term NGHDL has in the meantime been replaced by NIL ("Nada Integrated Language") for better pronounceability and in tying in with the Group's acronym NADA. The NIL concepts include general questions of timing, parameterisation and modularisation.

The investigated design techniques included verification and deductive design. The Group was also investigating the mathematical foundations for hardware design. Appropriate mathematical methods were taken from computation theory, higher order algebra, proof theory, relation algebra and timed process algebra.

1.2 Main Achievements in Year III

The third year significantly advanced the state of the art in the areas of modeling, deductive design and foundations. The output is detailed in Section 3.

On the side of foundations, a major event was the *Third International Workshop on Higher Order Algebra, Logic and Term Rewriting*, held in conjunction with ALP (Algebra and Logic Programming) Southampton, Sept. 2–5, 1997. The workshop was sponsored by the EACSL and NADA members K. Meinke and B. Möller were part of the organizing committee. The proceedings of the workshop have appeared within vol. 1298 of Springer Lecture Notes in Computer Science.

Concerning NIL, there was a severe backdrop imposed on the Group: the development of the language MHDL we had started exploring as a possible basis for NIL has been abandoned. Meanwhile a new development is under way: SLDL (System level description language). However, this language is still in its requirements definition phase and so could not be used within the project duration. We shall formulate our recommendations on NIL as part of the NADA book and forward them to the SLDL committee as input for their further work.

Concerning all of the mentioned research topics, we had the opportunity to present the NADA work to a broad audience at the *Third Annual NADA Meeting* held in conjunction with the conferences CHDL/VUFE/LCMQA '97 at Toledo, 20–25 April, 1997. In particular, a tutorial on four central threads of NADA research was presented (see Section 2.2.1). All this was well-received and gave NADA considerable international visibility.

Two further meetings, an interim meeting at Amsterdam, 25–27 October, 1997, and the Final Meeting at Lübeck, 20–23 September, 1997, provided further useful interaction. At the latter, the plans for the NADA book have been consolidated. In fact, the meeting consisted mainly of presentations and thorough discussions of the contents of the book chapters. The book will appear within the tutorial subseries of Springer Lecture Notes in Computer Science. A schedule has been set up for having the book published in Summer 1998. The contents is detailed in Section 5.

2 NADA Meetings and Conferences

2.1 NADA Interim Meeting, Amsterdam, 25–27 October 1996

The meeting was attended, next to NADA members, by a number of guests, viz. Jos Baeten, TUE Eindhoven, Egbert-Jan van Buiten, UvA Amsterdam, Jan Friso Groote, CWI Amsterdam, Frederic Gruau, CWI Amsterdam/Sussex, Merijn de Jonge, UvA Amsterdam, Ramin Monajemi, UvA Amsterdam and Dieter Spreen, Siegen.

2.1.1 Meeting Programme

Friday 25 October

14.15	Welcome
14.30	*All participants* First discussion on NGHDL concepts
15.30	*Frederic Gruau* (CWI Amsterdam, Sussex) A model of parallel computation for reconfigurable FPGAs
16.30	break
17.00	*Jos Baeten* (TUE Eindhoven) Specification and verification of hardware components using discrete-time process algebra

Saturday 26 October

9.45	*Dieter Spreen* (Univ. Siegen) A new model construction for higher type systems
11.15	*All participants* Second discussion on NGHDL concepts
14.30	*Jan Friso Groote* (CWI Amsterdam) μCRL: specification and verification of processes with data
15.30	break
16.00	*Egbert-Jan van Buiten* (UvA Amsterdam) Beating grid protocols
17.00	*Jan Bergstra* (UvA Amsterdam, UU Utrecht) Network algebra

Sunday 27 October.

9.45 Second discussion on NGHDL concepts

10.45 break

11.15 *Hans Leiss* (LMU München)
Logical relations for dependent types, and representation independence

15.00 *Bernhard Möller* (Un. Augsburg)
Refining ideal behaviours

Preparation of NADA tutorial and Toledo conference

16.00 break

16.30 *Ulrich Berger* (LMU München)
Reasoning with infinite streams

17.30 *All participants*
Discussion on NADA book, and Toledo conference (including NADA tutorials)

2.1.2 Participants

Jos Baeten, TUE Eindhoven
Ulrich Berger, LMU München
Jan Bergstra, UvA Amsterdam, UU Utrecht
Peter Breuer, Madrid
Egbert-Jan van Buiten, UvA Amsterdam
Walter Dosch, Augsburg
Jan Friso Groote, CWI Amsterdam
Frederic Gruau, CWI Amsterdam/Sussex
Merijn de Jonge, UvA Amsterdam
Hans Leiss, LMU München
Ramin Monajemi, UvA Amsterdam
Bernhard Möller, Univ. Augsburg
Jan Philipps, TU München
Alban Ponse, UvA Amsterdam
Peter Scholz, TU München
Dieter Spreen, Un. Siegen
Viggo Stoltenberg-Hansen, Uppsala
John Tucker, Swansea

2.2 3rd NADA Annual Meeting, Toledo, 20–25 April 1997

The meeting was attended, next to NADA members, by two guests, viz. M von der Beeck, Univ. Passau, and M. Kahrs, Rutgers Univ. Being held in conjunction with the conferences CHDL/VUFE/LCMQA '97, it presented surveys of the essential parts of the NADA work to a renowned and truly international audience. In particular, a tutorial on four central threads of NADA research was presented.

2.2.1 Abstracts of the NADA Tutorial *Formal Methods in Hardware Design*

M. Broy: Mathematical Methods in System Engineering. We present a mathematical, scientific, descriptional and methodological basis for system engineering. In its core, there is a mathematical model of a system and formal techniques to describe it. We outline

representative examples of diagrammatic description techniques as they are used in system engineering approaches in practice and show how they are formally related to the system model. These description techniques include in particular

- data models,
- process models,
- structure and distribution models,
- state transition models,
- interface models.

We define a translation of the description techniques into predicate logic. This allows us to combine techniques of formal specification and verification with pragmatic system description techniques. We show how to develop systems with the help of these description techniques in refinement steps. By this, we demonstrate how system engineering methods can be backed up by mathematics.

B. Möller: Towards Deductive Hardware Design. The term *Deductive Design* means the systematic construction of a system implementation, starting from its behavioural specification, according to formal, provably correct rules. The main advantages of this approach are:

- The resulting implementation is correct by construction.
- The rules can be formulated schematically, independent of the particular application area, and hence can be re-used for wide classes of similar problems.
- Since the rules are formal, the design process can be assisted by machine.
- Implementations can be constructed in a modular way, with emphasis on correctness first and subsequent transformation to increase performance.
- The formal derivation also serves as a record of the design decisions taken and hence is an explanatory documentation. Upon modification of the system specification it eases revision of the implementation.

This paradigm has successfully been applied to sequential and, to a lesser extent, also to parallel programs. The tutorial demonstrates its application to hardware design. We present various case studies dealing with basic combinational and sequential circuits. Special emphasis is laid on parametrization and re-usability aspects.

N. Harman: Algebraic Models of Microprocessors. The usefulness of formal methods is limited by the accuracy of the mathematical models of hardware that are available. Important questions in formally representing microprocessors are:

- How can we model structures present in modern microprocessors?
- How can we model time, and the relationship between times at different levels of abstraction, in the presence of out-of-order execution?
- What does it mean for a microprocessor representation to be correct?
- How can we practically establish correctness?

We introduce a set of algebraic techniques for representing microprocessors, readily implementable within a range of available software tools. The techniques introduced are suitable for representations of microprocessors at a number of levels of abstraction, including ISA and RT. The techniques are readily applicable to modern microprocessor implementations, including pipelined and superscalar processors. We consider the precise concept of the correctness of one level of abstraction with respect to another, and how we can simplify the resulting proof obligations. We discuss how these algebraic tools relate to, and may be integrated with, techniques developed by other groups working on formal methods for microprocessor representation. We illustrate our tools with a small superscalar example, supporting out-of-order issue and retirement, with precise architectural state maintained by a reorder buffer.

A. Ponse: Grid protocols based on synchronous communication. A specification format for networks based on synchronous communication, modeling parallel computation, will be presented. This approach starts from the work done on Synchronous Concurrent Algorithms (SCAs) in Swansea (Thompson and Tucker, 1994).

The specification format is ACP-based (Algebra of Communicating Processes, Bergstra and Klop). Main technical construct is the process prefix (generalizing Milner's action prefix), with which parallel input can be easily specified and analyzed.

For certain classes of networks, I/O behaviour (comprising absence of deadlock) can be easily characterized by a recursive equation. The specification format and characterization results are illustrated by some examples (Fibonacci sequence, numerical approximation of a one-dimensional wave equation).

2.2.2 NADA Workshop Presentations

Session N-1: Analog Modelling (Chair: W. Dosch)

Bernhard Möller:	Opening Remarks
Mark Kahrs:	MHDL: A Critical Evaluation (Invited Talk)
Natividad Martínez Madrid:	A Suggested Formal Semantics for the New Elements in VHDL-AMS

Session N-2: Logic and Algebraic Approaches (Chair: H. Schwichtenberg)

Peter T. Breuer:	Hardware Semantics and Time
Neal A. Harman, Anthony C.J. Fox:	Algebraic Models and Correctness of Superscalar Microprocessors
Alban Ponse:	Grid Protocols and Iterative Operations in Process Algebra

Session N-3: Functional Approaches (Chair: M. Broy)

Bernhard Möller:	Stream Algebra
Walter Dosch:	Deductive Design of Digital Circuits
Ulrich Berger, Helmut Schwichtenberg:	Extraction of Functional Programs from Proofs

Session N-4: State-based Approaches (Chair: N. Harman)

Michael von der Beeck:	Statecharts Semantics: An Evaluation (Invited Talk)
Peter Scholz:	System-Level Hardware Design with μ-Charts
Karl Meinke:	Semantics and Verification for Statecharts using Non-Deterministic Synchronous Concurrent Algorithms

2.3 NADA Final Meeting, Lübeck, 20–23 September 1997

The meeting was attended by representatives of almost all sites and two guests, Tiziana Margaria, Dortmund, and Mike Fourman, Edinburgh. It consisted of two invited talks and in-depth discussions on the book contributions (cf. Section 5) by the various group members.

2.3.1 Presentations

Saturday, September 20th

14:00 – 14:10	W. Dosch: Welcome
14:10 – 14:20	J. Tucker: *NADA and* NIL
14:20 – 15:20	N. Harman: *Algebraic models of advanced microprocessor implementations*
15:20 – 15:30	Discussion
15:30 – 15:50	Break
15:50 – 16:20	H. Leiss: *Logical relations for dependent types*
16:20 – 16:30	Discussion
16:30 – 17:15	J. Bergstra: *Process algebra and grid protocols*
17:15 – 17:25	Discussion

Sunday, September 21st

09:00 – 10:00	M. Fourman: *New Hardware Design Methods*
10:00 – 10:15	Discussion
10:15 – 10:45	Break
10:45 – 11:45	H. Schwichtenberg: *Normalization by evaluation*
11:45 – 12:00	Discussion
12:00 – 14:00	Lunch
14:00 – 14:30	K. Stephenson: *An algebraic specification of the JAVA virtual machine*
14:30 – 14:45	Discussion
14:45 – 15:30	M.I. Poole: *Hierarchies of spatially extended systems*
15:30 – 15:45	Discussion
15:45 – 16:15	Break
16:15 – 16:45	V. Stoltenberg-Hansen: *Streams and Domains*
16:45 – 18:00	*1. Discussion: NADA and* NIL

Monday, September 22nd

09:00 – 09:45	B. MÖLLER: *Stream algebra*
09:45 – 10:00	Discussion
10:00 – 10:30	P. SCHOLZ: *Verification and Hardware Generation for Statecharts*
10:30 – 10:45	Discussion
10:45 – 11:15	Break
11:15 – 11:45	P. BREUER: *Infinitesimal languages*
11:45 – 12:00	Discussion
12:00 – 14:00	Lunch
14:00 – 15:00	T. MARGARIA: *The application layer of the MOSEL tool kit: monadic second order logic for hardware verification*
15:00 – 15:15	Discussion
15:15 – 15:45	Break
15:45 – 16:30	K. HANNA: *Reasoning about analog implementations of digital systems*
16:30 – 16:45	Discussion
16:45 – 17:15	Break
17:15 – 19:00	*2. Discussion: NADA and* NIL

Tuesday, September 23rd

09:00 – 09:45	B. MÖLLER: *Towards deductive hardware design*
09:45 – 09:55	Discussion
09:55 – 10:25	W. DOSCH: *Deductive design of iterative circuits*
10:25 – 10:35	Discussion
10:35 – 11:00	Break
11:00 – 12:00	*3. Discussion NADA and* NIL

2.3.2 Participants

Prof. Dr. Jan Bergstra	Faculty of Mathematics University of Amsterdam Kruislaan 403 NL-1098 SJ Amsterdam
Prof. Dr. Bernhard Möller	Institut für Informatik Universität Augsburg Universitätsstraße 2 D–86135 Augsburg
Prof. Dr. Michael P. Fourman	Dept. of Computer Science University of Edinburgh James Clerk Maxwell Building King's Buildings, Mayfield Road Edinburgh EH9 3JZ
Dr. F.K. Hanna	Computing Laboratory University of Kent Canterbury Kent CT2 7NF
Prof. Dr. Walter Dosch Dipl.-Inform. Annette Stümpel Dipl.-Math. Bernd Wiedemann	Institut für Softwaretechnik und Programmiersprachen Medizinische Universität zu Lübeck Seelandstraße 1 a D–23569 Lübeck
Prof. Dr. Carlos Delgado Kloos Dr. Peter Breuer	Ingeniería Telemática Universidad Carlos III de Madrid C/Butarque 15 E–28911 Leganés
Dipl.-Inform. Peter Scholz	Fakultät für Informatik Technische Universität München Arcisstraße 21 D–80290 München

Prof. Dr. Helmut Schwichtenberg Dr. H. Leiss	Mathematisches Institut Ludwig-Maximilians-Universität Theresienstraße 39 D–80333 München
Dr.–Ing. Tiziana Margaria	Lehrstuhl für Programmiersysteme Universität Passau Innstraße 33 D–94032 Passau
Prof. Dr. J.V. Tucker Dr. Neal Harman Dr. M.I. Poole Dr. Karen Stephenson	Dept. of Computer Science University of Wales Swansea Swansea SA2 8PP Singleton Park
Prof. Dr. Viggo Stoltenberg-Hansen Dr. Jens Blanck	Dept. of Mathematics Uppsala University P.O. Box 480 S–75106 Uppsala

3 Scientific Output May 96 to August 97

3.1 Amsterdam

3.1.1 Results

NADA-activity in this period concentrated on further development of Grid Protocols; a specification format geared to Synchronous Concurrent Algorithms (SCAs, Tucker c.s.), and going with simple characterization results and a process algebraic format that is suitable for simulation and animation.

As an example, the Single Pulser (IFIP WG10.5 Hardware Verification Benchmark Suite) was specified as a Grid Protocol, and with a few additional constraints proven correct.

At the University of Amsterdam, 12 December 1996, the PhD. thesis *Experiments in Specification Re-Engineering* was defended by J.A. Hillebrand (supervisors J.A. Bergstra, M.G.J. van den Brand). This thesis contains a chapter "A small language for the specification of Grid Protocols" (35 pp.), relating the below-mentioned publication in SCP to muCRL, a tool-supported version of ACP.

One master's thesis (in the field of Applied Logic) on Beating Grid Protocols (following closely the SCA-format of Tucker c.s., and giving an operational, process algebraic account of SCAs) has been submitted at Utrecht University, Dept. of Philosphy (candidate M. Pouw, supervisor Ponse, ceremony 28 November). In this thesis, the Single Pulser (IFIP WG10.5 Hardware Verification Benchmark Suite) was specified as a Grid Protocol, and with a few additional constraints proven correct. A second master's thesis on Beating Grid Protocols is being written at our own department (candidate R. Monajemi, supervisor Ponse).

In continuation of the NADA work, the University of Amsterdam, Programming Research Group has appointed at 1 September 1997 E.-J. van Buiten as a PhD. candidate. He is to do further research on Grid Protocols (specification formats and characterization results).

3.1.2 Visits

Attendance to the Amsterdam NADA workshop 25-27 October 1996 (Bergstra, Ponse)

Participation in Toledo workshop in 20-25 April 1997, (Ponse) (Tutorial on "Formal methods in Hardware Design: Grid Protocols", Talk on "Grid Protocols and Iterative Operations in Process Algebra")

3.1.3 Relevant publications

BHP95 J.A. Bergstra, J.A. Hillebrand, and A. Ponse: Grid protocols based on synchronous communication: specification and correctness. Technical Report P9511, Programming Research Group, University of Amsterdam, 1995.

We provide a short notation for processes with parallel inputs and outputs. With this specification format, a certain class of networks, comprising synchronous grid protocols, can be easily specified and analyzed. We illustrate this by an example on the approximation of a one-dimensional wave equation.

BMS95 J.A. Bergstra, C.A. Middelburg, and Gh. Ştefănescu. Network algebra for synchronous and asynchronous dataflow. Technical Report P9508, Programming Research Group, University of Amsterdam, 1995

Network algebra is proposed as a uniform algebraic framework for the description and analysis of dataflow networks. An equational theory, called BNA (Basic Network Algebra), is presented. BNA, which is essentially a part of the algebra of flownomials, captures the basic algebraic properties of networks. For synchronous and asynchronous dataflow networks, additional constants and axioms for connections are given; and corresponding process algebra models are introduced. The main difference between these models is in the interpretation of the identity connections, called wires in dataflow networks. The process algebra model for the asynchronous case is compared with previous models.

Hil96 J.A. Hillebrand. A small language for the specification of Grid Protocols, Chapter 4 (35 pp.) in: J.A. Hillebrand, Experiments in Specification Re-Engineering, PhD. Thesis, University of Amsterdam, December 1996

BErHilPon97 J.A. Bergstra, J.A. Hillebrand and A. Ponse: Grid protocols based on synchronous communication, Science of Computer Programming, 29(1&2):199-233, 1997.

We provide a short notation for processes with parallel inputs and outputs. With this specification format synchronous networks or grid protocols can be specified in a straightforward way. For a certain class of connected networks we prove a correctness theorem that characterizes I/O behaviour. We illustrate our approach by an example on the approximation of a one-dimensional wave equation.

3.2 Augsburg

3.2.1 Results

The work in Augsburg centered around Deductive Design, as applied to both hardware (see below) and software [DosMol97,MolPoi97,MolLis97].

In the case of hardware the performed case studies include many of the IFIP WG10.5 Benchmark Verification Problems. A major breakthrough was the switch to formalizing the specifications and derivations using the functional programming language *Gofer*. The polymorphism of this language allows the use of Ştefănescu's network algebra and other algebraic laws both at the level of combinational and sequential circuits. Moreover, many derivations can be performed in a polymorphic way abstracting from concrete applications and hence achieving much better re-usability. This work is reported in [MolHard98].

Another thread of work concerns more algebraic ways of specifying, reasoning about and transforming descriptions of (sets of) streams [MolStr98, MolTemp97, MolMod97, MolStef98].

3.2.2 Relevant Publications

DosMol97 W. Dosch, B. Möller: Calculating a functional module for binary search trees. In: W. Kluge (ed.): Implementation of Functional Languages. Selected Papers from the 8th International Workshop, Bad Godesberg, Germany, September 1996. LNCS 1268. Berlin: Springer 1997, pp. 267–284

MolPoi97 B. Möller: Calculating with pointer structures. In: R. Bird, L. Meertens (eds.): Algorithmic languages and calculi. Proc. IFIP TC2/WG2.1 Working Conference, Le Bischenberg, Feb. 1997. Chapman&Hall 1997, 24–48

MolLis97 B. Möller: Linked lists calculated. Submitted for publication (1997)

MolTemp97 B. Möller: Temporal Operators on Partial Orders. Proc. 3rd Domain Workshop, Munich, 29–31 May 1997. LMU Munich (to appear)

MolMod97 B. Möller: Modal and Temporal Operators on Partial Orders. In: R. Berghammer (ed.): Programmiersprachen und Grundlagen der Programmierung. Insitut für Informatik und Praktische Mathematik, Universität Kiel (to appear)

MolHard98 B. Möller: Towards deductive hardware design. In [NADA 98] (in preparation)

MolStr98 B. Möller: Ideal stream algebra. In [NADA 98] (in preparation)

NADA98 B. Möller, J.V. Tucker (eds.): Prospects for hardware foundations. LNCS (in preparation)

MolStef98 B. Möller, G. Ştefănescu: Network specification algebra (in preparation)

3.2.3 Attended Conferences

- **NADA Meeting Amsterdam, 25–28 Oct 1996**
- **Visit to Oxford, 8–11 Nov 1996**
  Discussions on deductive design and on possible follow-on project to NADA
- **Esprit Long-Term Research: Post-Conference Workshop Brussels, 27–28 Nov 1996**
  Information about possible follow-on project to NADA
- **3rd International Seminar on Relational Methods in Computer Science, Hammamet, Tunisia, 5–12 Jan 1997**
  Of particular interest for NADA were a number of talks on relational calculi applied to modeling and deductive design of systems and algoriths. B. Möller presented [MolLis97].
- **IFIP TC2/WG 2.1 Working Conference on Algorithmic Languages and Calculi, Le Bischenberg, France, 16–21 Feb 1997**
  One of the leading conferences on deductive design. B. Möller presented [MolPoi97].
- **Hardware Description Languages and their Application CHDL'97, and 3rd NADA Annual Meeting, Toledo, 19–26 April 1997**
  B. Möller presented preliminary versions of [MolHard98] at the NADA tutorial and [MolStr98] at the NADA Annual Meeting.
- **3rd Domain Workshop, Munich, 29–31 May 1997.**
  B. Möller presented [MolTemp97].

3.2.4 Visitors

Prof. Ştefănescu from Romania visited both the University of Augsburg and the TUM. His collaboration with M. Broy on visual formalisms in the area of formal system semantics and with B. Möller on the papers [MolHard98] and [MolStef98] in the areas of formal specification and deductive design was most fruitful.

3.3 Kent

3.3.1 Results

Throughout this period, work on *Mixed-level Specification and Verification of Digital Systems* has continued. It concerned extending the well-established methods of specification and verification using higher-order logic downwards to encompass circuit behaviours expressed at the analog level of abstraction.

The motivation for this work is that, in order to achieve higher operating speeds, or lower levels of power consumption, designers of digital systems often need to work at the analog level of abstraction. By this means, the logically redundant stages of amplification necessary to amplify signals to well-defined, digital levels between successive logic devices can be eliminated. Digital circuits realised at the analog level can be up to an order of magnitude faster and more energy efficient than those designed entirely at the digital level.

The main achievement over the last year has been the discovery that the behavioural verification of the digital properties of analog circuits could be reduced to a case of Presburger arithmetic and hence handled automatically. This new approach entirely supersedes the previous approach, which required a mathematically-sophisticated user to guide an interactive theorem-prover.

A prototype implementation of this decision procedure has been developed. This takes as inputs:

- A set of definitions of (conservative) rectilinear specifications of the behaviour of analog devices (resistors, transistors, etc.);
- A specification of the analog/digital abstraction relation (this characterises both the voltage levels and the current flows associated with digital signal levels);
- A digital-level specification for the desired behaviour of a device;
- A netlist for a proposed, analog-level realisation of this device. (This might be, for example, a netlist of a TTL circuit, expressed in terms of the above analog device behaviours.)

These various specifications are all described in a typed, higher-order predicate logic.

The prototype system takes the above set of inputs and yields a YES/NO answer as to whether the proposed analog level implementation is correct with respect to the given digital specification and abstraction relation. In demonstrating that this can be done, it has the potential to supersede simulation as a major design tool used by the VLSI designer of high-performance devices.

The prototype system is written in Haskell. It begins by constructing the verification condition and then simplifying it (rearranging the quantifiers, and beta-reducing any reducible subexpressions). It then converts it to a clausal normal form and, from this, it extracts a set of linear constraint problems. These are analysed by a constraint analysis procedure which, for efficiency, is written in FORTRAN.

The details of this approach are described in a chapter of the forthcoming NADA book.

3.3.2 Relevant Publications

Hann97 F K Hanna: Reasoning about Digital Devices at the Analog Level of Abstraction, 30pp, submitted for publication, 1997.

3.4 Leeds

3.4.1 Results

During the period April 1996-August 1997, the group at the university of Leeds contributed to case studies of models of dynamical systems. We concentrated on the following types of spatialy extended dynamical systems

- continuous space, continuous time, continuous state (e.g. partial differential equations (P.D.Es) and excitable media
- discrete space, discrete time, continuous state (coupled map lattice (CMLSs) and neural networks (NN));
- discrete space, discrete time, discrete state (cellular automata (CAs));
- and relating these models to time series and images (data streams) genrated from the models or in real experiemnts or clinical observations.

Of primary interest have been the cellular (discrete space) systems, CMLs and CA, and discretized PDEs, because the notion of discrete space and local connectivity correspond closely with the notion of parallel digital systems and architectures. We have been motivated by spatially extensive dynamical system models of wave phenomena in cardiac tissue, and clinical applications to cardiac arrhythmias, defibrillation control algorithms, and noninvasive visualisations of cardiac activity.

3.4.2 Relevant papers

Papers on NADA case studies

A.V.Holden, M.J.Poole, J.V.Tucker Hierarchical reconstructions of cardiac tissue. In preparation for *J Theoretical biology*

V.N. Biktashev, A.V. Holden, Re-entrant waves and their elimination in a model of mammalian ventricular tissue, *Chaos* 1998 in press

V.N. Biktashev, A.V. Holden, H Zhang Modelling the defibrillation of ventricular tissue" *Int. J. Bifurcation and Chaos*, **7**, 477-485 1997

A V Holden Spatiotemporal irregularity of propagating electrical activity in models of cardiac tissue: the genesis of flutter and fibrillation Ischia Spring School Lecture Notes May 1997

A V Holden Approaches to defibrillation: the control of re-entrant activity, and the annihilation of propagating waves of excitation in cardiac tissue. Ischia Spring School Lecture Notes May 1997

M R Boyett, A V Holden, H Zhang Computational investigation of atrial modulation of sino-atrial pacemaker activity, European Heart Journal 17 Abstract Supplement 293 1996

V N Biktashev, A V Holden, H Zhang Framework for a computational theory of defibrillation, Physiological Society Abstracts Leeds Meeting 11-13 Sept 1996.

V N Biktashev, A V Holden, H Zhang A model for the action of external current onto excitable tissue., International J of Bifurcation and Chaos 7 477-485 1997

A.V.Holden Defibrillation in models of cardiac muscle. J Theoretical medicine (in press)

A.V.Holden Nonlinear Science - the impact of biology., J Franklin Institute 334B 971-1014 1997

A V Holden From Structure and function to functional structures - towards a computational biology *Trp 3* (Research and Funding News from the Wellcome Trust) issue 6 Q1 1996 19-21

Other theoretical papers

V.N. Biktashev, A.V. Holden Hypermeander of spiral waves as a deterministic Brownian motion, *Physica D*, submitted

Yu.E. Elkin, V.N. Biktashev, A.V. Holden On the movement of excitation wave breaks" *Chaos Solitons and Fractals*, 1997, in press

E.V. Nikolaev, V.N.Biktashev, A.V.Holden On Feedback Resonant Drift and Interaction with the Boundaries in Circular and Annular Excitable Media *Chaos Solitons and Fractals*, 1998, in press

V.N. Biktashev, A.V. Holden, E.V. Nikolaev Spiral wave meander and symmetry of the plane *Int J Bifurcation and Chaos*, **6** 2433-2440 1996

V N Biktashev, A V Holden Computation of re-entrant propagation in a two-dimensional model of ventricular tissue simulating long QT syndrome, *European Heart Journal***17** Abstract Supplement 597 1996

V N Biktashev, A V Holden Re-entrant activity and its control by resonant drift in a model of mammalian ventricular tissue, *Proc Roy Soc London***B 263** 1373-1382 1996

H Zhang, R.L Winslow, A.V.Holden Re-entrant excitation initiated in model of inhomogeneous atrial tissue., *J theoret biol* (in press))

H Zhang, A.V.Holden One dimensional modelling of the vulnerability to re-entry of homogeneous atrial tissue. *J theoret biol***184** 119-124 1997

Preprints available from http://www.cbiol.leeds.ac.uk/arunpub.html.

3.4.3 Related products

The following grants have been awarded to the Leeds group; these are developments and applications of case studies

- British Heart Foundation: £444,830 Mapping the sinoatrial node. Oct 1996-Sept 2001.
- British Heart Foundation: £56,855 Computational investigation of possible arrhythmogenic role of mechanoelectric feedback by combining magnetic resonance imaging of beating human heart with models of ventricular propagtion. Dec 1996-Nov 1998.
- EPSRC: £6,400 Excitable media behaviours of plankton population dynamics models. Nov 1997-June 1998
- EPSRC: £5755 Hierachies of spayoially extended nonlinear models of physiological systems: algorithms and applications (with John Tucker, Swansea; esentially a local continuation of NADA activities and case studies)

Publicity: the application of SCA theory to the cardiac dynamics and control of arryhthmias was the lead news item of Oct 1997 *Scientific Computing W orld.*

3.4.4 Visits

AV Holden presented papers at

- *Visions of Nonlinear science in the 21st century.* Seville, Spain. June 1996
- *Chaos in Biology* Ischia, Italy. April 1997
- *Spiral waves, Euclidean symmetries and cardiac arrhythmias* Warwick, UK. May 1997
- *Information Processing in Cells and Tissues.* Sheffield, UK. July 1997
- *International Congress for Physics Students*, Vienna, Austria July 1997

and visited Swansea, and hosted visits from Swansea by M. Poole and J.V. Tucker.

3.5 Lübeck

3.5.1 Results

The work in Lübeck concentrated on deductive design — both in the area of hardware and software. The work aimed at abstracting characteristic design patterns from specific case studies.

For a class of digit recurrence algorithms, we derived various implementations as combinational and sequential circuits in a schematic way. The approach was then generalized from linearly recursive functions to cascading recursive functions using Bird's theory of lists.

The homomorphisms widely employ higher-order functions like *map* and *reduce* to encapsulate standard design patterns. In hardware design, the resulting functional algorithms describe tree-structured networks in a parameterized way. In software design, they serve as architecture independent data parallel programs.

In several case studies we systematically derived the combining operations for the list homomorphism from high-level specifications. The formal derivations also lead to simplifications of previously published data parallel algorithms.

3.5.2 Relevant Publications

Dos96a W. Dosch: *Calculating Digital Counters.* In: D. Bjørner, M. Broy, I.V. Pottosin (eds.): Perspectives of Systems Informatics. Second International Andrei Ershov Memorial Conference. Akademgorodok, June 25–28, 1996. Lecture Notes in Computer Science *1181*. Berlin: Springer 1996, 21–39

Deductive design characterizes a method where a system description is deduced from the functional specification of its behaviour applying formal transformations rules. Following this design methodology, we derive circuit descriptions for various combinational and sequential counters from a common functional specification using equational and inductive reasoning.

Dos96b W. Dosch: *Calculating Hamming Streams.* In: K. Yetongnon, S. Hariri (eds.): Proceedings of the Ninth International Conference on Parallel and Distributed Computing Systems, Vol. I. Dijon, September 25–27, 1996. Raleigh, N.J.: International Society for Computers and Their Applications 1996, 529–534

We present an algebraic derivation of the asynchronous dataflow network computing the infinite stream of Hamming numbers. As a subtask, we derive an implementation for the deterministic merge agent from its specification.

DosMol96 W. Dosch, B. Möller: *Calculating a Functional Module for Binary Search Trees.* In: W. Kluge (ed.): Implementation of Functional Languages. 8th International Workshop IFL'96, Bad Godesberg, September 16-18, 1996. Lecture Notes in Computer Science *1268*. Berlin: Springer 1997, 267–284

We formally derive a functional module for binary search trees comprising search, insert, delete, minimum and maximum operations. The derivation starts from an extensional specification that refers only to the multiset of elements stored in the tree. The search tree property is systematically derived as an implementation requirement.

Dos97a W. Dosch: *On Swap Systems.* In: J. Graham, D.-G. Shin (eds.): Intelligent Systems. Proceedings of the 6th International Conference on Intelligent Systems. Boston, June 11–13, 1997. Raleigh, N.C.: International Society for Computers and Their Applications 1997, 32–36

We present a simple computational model for a specific class of systolic arrays. Swap systems achieve a specified global rearrangement of a linear data structure based on local pattern-driven swaps of adjacent elements. We illustrate various applications and investigate fundamental properties — among others confluence, termination and parallelization.

Dos97b W. Dosch: *Design Decisions for the Parallel Evaluation of Polynomials.* In: H.A. Arabnia (ed.): Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications, Vol. II. Las Vegas, June 30 - July 3, 1997. CSREA Press 1997, 636-644

We systematically explore the design decisions for various parallel algorithms evaluating a polynomial in one indeterminate. The functional algorithms show a different degree of parallelism; they can be oriented towards different architectures. The algorithms are formally derived from a common specification using equational and inductive reasoning.

DosWie97a W. Dosch, B. Wiedemann: *Calculating List Homomorphisms For Parallel Bracket Matching.* In: R. Berghammer (Hrsg.): Kolloquium Programmiersprachen und Grundlagen der Programmierung. Avendorf, September 24–26, 1997. Institut für Informatik, Christian-Albrecht-Universität zu Kiel [to appear]

The paper presents a case study for the systematic design of abstract data parallel programs. We formally derive a list homomorphic parallel bracket matching algorithm from a specification. We explore how high-level design decisions influence the final parallel algorithm. We specialize the data parallel algorithm retrieving well-known sequential versions.

Dos97c W. Dosch: *An Improved Data Parallel Algorithm For the LSS-Problem* [submitted]

We present an improved data parallel algorithm for the longest satisfying segment problem. We formally derive the functional algorithm from a set-oriented specification and show different applications.

DosWie97b W. Dosch, B. Wiedemann: *A Data Parallel Algorithm For the Evaluation of Polynomials* [submitted]

We present a list homomorphic algorithm for evaluating a polynomial in one indeterminate. We formally derive the data parallel algorithm from a high-level specification and specialize it obtaining well-known sequential versions.

3.5.3 Attended Conferences

- **15. PARS Workshop "Programmiersprachen und hardwarenahe Programmierung", Bad Honnef, May 13–15, 1996**
  W. Dosch presented [Dos96b].
- **Perspectives of Systems Informatics. Second International Andrei Ershov Memorial Conference, Akademgorodok, June 25-28, 1996**
  W. Dosch presented [Dos96a] and participated in a panel discussion with D. Bjørner, I. Pottosin and N. Wirth on *Perspectives of Systems Informatics.*
- **Eighth International Workshop on Implementation of Functional Languages, Bonn, September 16–18, 1996**
  W. Dosch presented [DosMol96].
- **Ninth International Conference on Parallel and Distributed Computing Systems, Dijon, September 25–27, 1996**
  W. Dosch presented [Dos96b].
- **1. Informatik-Treffen Kiel–Lübeck, Malente-Gremsmühlen, April 11–12, 1997**
  W. Dosch presented a lecture on *Deduktiver Entwurf stromverarbeitender Funktionen.*
- **Annual meeting of the Esprit Working Group 8533, Toledo, April 21–25, 1997**
  W. Dosch gave a talk on *Deductive Design of Digital Circuits.*
- **6th International Conference on Intelligent Agents, Boston, June 11–13, 1997**
  W. Dosch presented [Dos97a].
- **International Conference on Parallel and Distributed Processing Techniques and Applications, Las Vegas, June 30 – July 2, 1997**
  W. Dosch presented [Dos97b].
- **Final Meeting of the Esprit Working Group 8533, Lübeck, September 20–23, 1997**
  W. Dosch organized the workshop and presented a talk on *Deductive Design of Iterative Circuits.*
- **Workshop "Programmierung und Grundlagen von Programmiersprachen", Avenhaus, September 24–26, 1997**
  W. Dosch presented [DosWie97a]
- **2. Informatik-Treffen Kiel–Lübeck, Malente-Gremsmühlen, October 31 – November 1, 1997**
  W. Dosch organized this workshop on "Rechnergestützter Entwurf komplexer Soft- und Hardwaresysteme" and gave a talk on *Konstruktive Entwurfsverfahren für komplexe Soft- und Hardwaresysteme.*
- **2. Lübecker Hochschultag, Kongreß- und Messezentrum Lübeck, November 6, 1997**
  W. Dosch presented a public lecture on *Korrekte Software — eine Illusion?*

3.6 Madrid

3.6.1 Results

The main technical achievements have been

- Study of Analog Hardware Description Languages and the description of continuous phenomena. In particular, study of the operational amplifier paradox and of the analog languages MHDL and VHDL-AMS, giving a formal semantics to the latter.
- Development of a refinement calculus for VHDL, based on the formal semantics developed previously.
- Critical comparison of Hardware Description Languages.

The main organisational achievements were

- Organisation of the NADA Workshop in Toledo from 20-25 April 1997 in the context of TOLEDO'97, which also included IFIP WG 10.5 CHDL'97 (Computer Hardware Description Languages), VUFE'97 (VHDL Users' Forum in Europe) and LCW&QA'97 (Libraries, Component Modelling and Quality Assurance). A NADA tutorial session was organised as well as presentations of NADA partners. This gave the NADA project considerable visibility.
- Attendance by invitation to SLDL'97 (System Level Design Language) in Il Ciocco (Italy) and presentation of some ideas of NIL, the NADA Integrated Language. At this workshop, promoted by the PTAB (Program Technical Advisory Board) of the US EDA Industry Council, and organised by ECSI, a new language is sought for the description at system level. At this workshop, requirements for the language were defined. The presentation of NADA ideas implied a technology transfer from the project.

3.6.2 Relevant Publications

BreuEtAl96a Peter T. Breuer, Natividad Martínez Madrid, Luis Sánchez, Andrés Marín, Carlos Delgado Kloos: A Formal Method for Specification and Refinement of Real-Time Systems, 8th EuroMicro Workshop on Real Time Systems, L'Aquila, Italy, June 96

BreuEtAl96b Peter T. Breuer, Carlos Delgado Kloos, Natividad Martínez Madrid, Luis Sánchez, Andrés Marín: A Refinement Calculus for VHDL, EuroDAC with EuroVHDL'96, Geneva, Oct. 96

MarEtAl97 Natividad Martínez Madrid, Peter T. Breuer and Carlos Delgado Kloos: A Sketch Semantics for VHDL-AMS, CHARME'97, Montreal, Canada, Oct. 97

Del97 Carlos Delgado Kloos: Diseño e Ingeniería de Sistemas Electrónicos, in: J. Cuena (ed.): Panorama Informático, 97

DelEtAl97 Carlos Delgado Kloos, Natividad Martínez Madrid, Andrés Marín López, Luis Sánchez Fernández: Lenguajes de Descripción Hardware: Conceptos y Perspectivas, Novática, Apr. 97

BreuEtAl97 Peter T. Breuer, Carlos Delgado Kloos, Natividad Martínez Madrid, Andrés Marín, Luis Sánchez Fernández: A Refinement Calculus for the Design of Hardware Systems in VHDL, TOPLAS, July 97, 30 pp.

3.6.3 Attended Conferences

- **NADA Workshop, Toledo, Spain, April 97**
  Peter T. Breuer gave a talk *Continuous Hardware Time Semantics: A Small Mystery.* Natividad Martínez Madrid presented *A Formal Semantics for VHDL-AMS.*
- **SLDL'97 Workshop, Il Ciocco, Italy, July 97**
  Carlos Delgado Kloos gave an Invited Talk *Requirements for a System Level Language.*
- **III Jornadas de Informática, El Puerto de Santa María, Spain, July 97**
  Carlos Delgado Kloos gave an Invited Talk *Métodos formales aplicados al diseño hardware.*

3.7 LMU Munich

3.7.1 Results

The LMU Munich group at the Mathematics Institute worked on
- formal verification of hardware,
- normalization by evaluation,
- finitary notation systems for infinite terms.

Also, a book with the title by Anne S. Troelstra and Helmut Schwichtenberg, has just been published:

> Anne S. Troelstra and Helmut Schwichtenberg: Basic Proof Theory.
> Cambridge University Press 1996

3.7.2 Relevant Publications

SchwiStro Helmut Schwichtenberg and Karl Stroetmann: From terms to circuits: the minmax example. Invited talk. In: M.L. Dalla Chiara, K. Doets, D. Mundici and J. van Benthem (eds.): 10th International Congress for Logic, Methodology and the Philosophy of Science (ICLMPS), Florence 1995, Volume One. Synthese Library **259**. Dordrecht: Kluwer 1997, 209–220

The paper deals with an "applied" aspect of computability theory in higher types: its possible use for the simultaneous design (from given components) and formal verification of hardware.

The basic observation is that many hardware units can be viewed as stream transformers, converting some input (control or data streams) into an output stream. This is possible even for bidirectional circuits since, in most cases, these can be modeled as pairs of unidirectional circuits. Here a stream is simply a function from the natural numbers (used to model time, i.e., the system clock) into the booleans (for control streams) or into some kind of data.

We consider some simple schemata (explicit definition and a form of primitive recursion) to define computable functionals. The resulting terms can be thought of as stream transformers, i.e., as circuits. The form of our schemata then makes it possible to directly translate a term into a circuit. On the other hand, we now have the term as a compact formal representation of the circuit, in the context of a reasonable theory, i.e., computability theory for partial continuous functionals. This is particularly useful for purposes of formal verification.

Our approach offers a number of benefits.

1. It opens the possibility to treat some questions on hardware synthesis and verification in a "mathematically civilized" setting. After all, it is an old experience in computability theory that it pays if one does not unnecessarily restrict (higher order) arguments. Moreover, it provides the proper mathematical framework to deal with undefined or error objects.
2. Both the design and the verification of a circuit can be done in a modular way. In particular, the design and verification can start from components that are only specified abstractly by higher order formulas (cf. Gordon 1986).
3. The approach sketched can deal with various types of data abstractions. For example, in the example presented we will abstract completely from the bit level representation of the data.

It is these benefits that distinguish our approach from others that are based on representing circuitry by finite automata.

BerEtAl98 Ulrich Berger, Matthias Eberl and Helmut Schwichtenberg: Normalization by evaluation. Contribution NADA book

As elaborated in the above paper with Stroetmann, it is both feasible and useful to represent circuits by means of terms in a language PR^ω of higher order primitive recursion (Gödel's system T). This can and should be done with machine support; in the Minlog system under development in Munich we put special emphasis on the deductive aspects of such systems, since this is an area where commercial systems need to be supplemented.

In the course of interactively generating proofs involving such typed terms it is crucial to have a term rewriting machinery available, in order to ease the burden of equational reasoning. Quite often term rewrite rules can be treated as normalization problems, and therefore it is essential to have an efficient implementation for the normalization ot terms available. By the same token, one then can also effectively normalize whole proofs (which can be written as derivation terms, using the Curry-Howard correspondece). This leads to further applications of proof theoretic tools, e.g., to the possibility to extract terms (representing circuits) from formal proofs. However, we do not go any further into this matter here, but rather concentrate on the issue of how to implement normalization.

It is well known that implementing normalization of λ-terms with free variables in the usual recursive fashion is quite inefficient. However, it is possible to compute the long normal form of such a term by evaluating it in an appropriate model. This makes it possible to use the built-in evaluation mechanism of e.g., Scheme (a pure Lisp dialect) to efficiently implement normalization. In fact, one can write down easily a Scheme program which does normalization by evaluation. The essential idea here is to find an inverse to evaluation, converting a semantic object into a sytactic term.

The aim of the paper is to develop the theory of normalization by evaluation from scratch, up to and including (some generalizations of) the system PR^ω of higher order primitive recursion (Gödel's system T).

Schwi97a Helmut Schwichtenberg: Finite notations for infinite terms. Invited Talk at the Conference on Recursion Theory, Oberwolfach 1996. In: B. Soare, T. Slaman, K. Ambos Spies (eds.): APAL special volume Recursion Theory '96 (to appear)

In 1991 Buchholz presented a method to build notation systems for infinitary sequent-style derivations, analogous to well-known systems of notation for ordinals. The essential feature is that from a notation one can read off by a *primitive* (not ε_0-) recursive function

its n-th predecessor and e.g., the last rule applied. Here we extend the method to the more general setting of infinitary (typed) terms, in order to make it applicable in other proof-theoretic contexts as well as in recursion theory. As an example, we use the method to give a new proof of a well-known trade-off theorem, which says that detours through higher types can be eliminated at the expense of transfinite recursion along higher ordinals.

Schwi97b Helmut Schwichtenberg: Programmentwicklung durch Beweistransformation: Das Maximalsegmentproblem. In: Sitzungsber. d. Bayer. Akad. d. Wiss., Math.-Nat. Kl. (to appear)

Schwi97c Helmut Schwichtenberg: Termination of permutative conversions in intuitionistic Gentzen calculi. To appear: TCS (Gentzen Volume)

BerEtAl97 Ulrich Berger, Helmut Schwichtenberg and Monika Seisenberger: From Proofs to Programs in the Minlog System. The Warshall Algorithm and Higman's Lemma (submitted for publication)

BenEtAl97 Holger Benl, Ulrich Berger, Helmut Schwichtenberg, Monika Seisenberger and Wolfgang Zuber: Proof theory at work: Program development in the Minlog system (submitted for publication)

Lei97 H. Leiss: Logical relations for dependent types

Type systems with types depending on individuals are useful in hardware description languages to express, for example, the type of bit-strings of size n, the individual time-scale of a hardware component c, the space of neighbours of a component c in a network, etc. They are also useful to describe the modular structure of large soft- or hardware packages as systems of modules depending on other modules.

We provide a basic notion for reasoning about dependent type systems: logical relations. For the simply typed lambda calculus, logical relations (between terms or elements of models) are inductively defined along the structure of types. Statman (1982) showed that the main theorems of simply typed lambda calculus can be proven using logical relations. Mitchell and Meyer (1984) introduced logical relations for second order lambda calculus, and Mitchell (1986) showed how they can be used to demonstrate a 'representation independence' theorem for second order lambda calculus: programs using logically related implementations of abstract data types are observationally equivalent. He posed the problem of developing logical relations for dependent types in order to demonstrate representation independence for SML with its sophisticated modules system.

We developed a notion of logical relation for the lambda calculus with dependent types and proved a 'Fundamental Theorem' like Statman's. It implies a representation independence theorem for SML: when substituting logically related terms into the free variable of an expression, the resulting expressions are related as well. It seems that the result can be applied in hardware description languages to prove, for example, (a) that the (functional) behaviour of a hardware unit does not change when subcomponents are replaced by suitably related ones, or (b) that the values of computations do not depend on a choice among logically related representations of data.

3.8 TU Munich

3.8.1 Results

At TUM two orthogonal research activities can be identified that are related to the NADA Working Group. They are briefly outlined in the sequel.

Manfred Broy presented a mathematical, scientific, descriptional and methodological basis for system engineering [Bro96]. He outlines representative examples of diagrammatic description techniques as they are used in system engineering approaches in practice and shows how they are formally related to the system model. These description techniques include in particular data models, process models, structure and distribution models, state transition models, and interface models.

He defines a translation of the description techniques into predicate logic. This allows to combine techniques of formal specification and verification with pragmatic system description techniques. He shows how to develop systems with the help of these description techniques in refinement steps. By this, he demonstrates how system engineering methods can be backed up by mathematics.

This contribution was also presented by Manfred Broy in the Tutorial "Formal Methods in Hardware Design" at Toledo.

Statecharts are the second topic of the research activities at TUM [SNR96, Sch96, PS97a, PS97b, PS97c, PS97d]. Jan Philipps, Peter Scholz et al. define a compositional, mathematical semantics for a Statecharts dialect and show that this semantics can be used for formal verification using model checking as well as for hardware generation.

Both activities will be reported in the NADA Book, each in a separate chapter.

3.8.2 Relevant Publications

Bro96 Manfred Broy: Mathematical Methods in System and Software Engineering. To appear in: Manfred Broy and Birgit Schieder (Edts.). Mathematical Methods in Program Development, Marktoberdorf, August 1996, Springer NATO ASI Series, Series F: Computer and Systems Sciences

BroSte96 Manfred Broy, Gheorghe Ştefănescu: The Algebra of Stream Processing Functions. TU Munich, Report No. TUM-I9620, 1996,
http://www4.informatik.tu-muenchen.de/paper_db/reports/TUM-I9620.html

Dataflow networks are a model of concurrent computation. They consist of a collection of concurrent asynchronous processes which communicate by sending data over FIFO channels. In this paper we study the algebraic structure of the dataflow networks and base their semantics on stream processing functions. The algebraic theory is provided by the calculus of flownomials which gives a unified presentation of regular algebra and iteration theories. The kernel of the calculus is an equational axiomatization called Basic Network Algebra (BNA) for flowgraphs modulo graph isomorphism. We show that the algebra of stream processing functions called SPF (used for deterministic networks) and the algebra of sets of stream processing functions called PSPF (used for nondeterministic networks) are BNA algebras. As a byproduct this shows that both semantic models are compositional. We also identify the additional axioms satisfied by the branching components that correspond to constants in these two algebraic theories. For the deterministic case we study in addition the coarser equivalence relation on networks given by the input-output behaviour and provide a correct and complete axiomatization.

GSB98 Radu Grosu, Gheorghe Ştefănescu, M. Broy: Visual Formalisms Revisited. Submitted for publication.

SNR96 Peter Scholz, Dieter Nazareth, Franz Regensburger. Mini-Statecharts: A Compositional Way to Model Parallel Systems. In: 9th International Conference on Parallel and Distributed Computing Systems, Dijon, France, 25-27 September 1996

Statecharts are well accepted in industrial applications for specifying parallel, reactive systems. Tools have been developed that supply the Statecharts specification method. Unfortunately, a reference semantics has not been developed until now. Therefore, formal semantics and simulator-tool semantics often differ. This contribution outlines how to specify parallel systems with Mini-Statecharts, a special subclass of Statecharts. A compositional semantics of Mini-Statecharts is given in a fully functional way. It can be immediately executed by a suitable interpreter. Thus, we do not only define a theoretical semantics, but in addition provide the basis to write a simple program for simulating and prototyping Mini-Statecharts.

Sch96 Peter Scholz. A Light-Weight Formalism for the Specification of Reactive Systems. In: SOFSEM'96, Milovy, Czech Republic, 23-30 November 1996

Statecharts are a visual specification mechanism for reactive, embedded systems. They are implemented in commercial tools like Statemate. However, some syntactic constructs impede the modular system specification and have a confusing semantics. We have developed Mini-Statecharts, a lean version of Statecharts. Mini-Statecharts are restricted to the most important syntactic elements of Statecharts but are nevertheless powerful enough to specify complex systems. In this contribution, we extend the core language with local variables and integer-valued signals to avoid state explosion. We show that the formal semantics of the core language smoothly carries over to the semantics of the extended language.

SH97 Peter Scholz, Erwin Harbeck. Task Assignment for Distributed Computing. Advances in Parallel and Distributed Computing APDC'97, Shanghai, IEEE Computer Society Press, 1997

In the course of the development of reactive systems often real time constraints have to be met. In such time critical applications heterogeneous multi-processor systems are used in order to fulfill these time constraints. This paper presents mixed integer linear programming methods that support the development of time critical systems. The goal of these methods is to fix for every task of a so-called task-graph the processor that will execute it together with its starting time. The main issue is a high-level-synthesis-like method for constructing a problem-specific multi-processor board. The presented methods have been fully implemented and tested.

PS97a Jan Philipps, Peter Scholz. System-Level Hardware Design with μ-Charts. In: Hardware Description Languages and their Application CHDL'97, Toledo, Chapman and Hall, April 1997

μ-charts are a synchronous specification language for reactive systems with a compositional semantics. We show how a μ-chart can be implemented in hardware, using a register and a combinatorial logic block that represents the transition relation of the system.

PS97b Jan Philipps, Peter Scholz. Compositional Specification of Embedded Systems with Statecharts. In: Theory and Practice of Software Development TAPSOFT'97, Lille, Springer LNCS 1214, April 1997

During the last years, Statecharts have gained wide acceptance for the specification of reactive, embedded systems. However, most semantics suggested so far are either informal or hard to grasp. In this contribution, we present a Statecharts dialect that permits nondeterministic specifications, offers zero-delay broadcast communication, and handles negation in trigger expressions in a new way. We give a compositional formal semantics for this dialect, which is abstract enough for formal reasoning and yet easy to operationalize for simulators, model checking tools and code generation.

PS97c Jan Philipps, Peter Scholz. Formal Verification of Statecharts With Instantaneous Chain Reactions. In: Tools and Algorithms for the Construction and Analysis of Systems TACAS'97, Twente, Springer LNCS 1217, April 1997

We present a method for symbolic model checking of μ-Charts, a Statecharts dialect with instantaneous broadcast communication. Due to this communication concept, μ-Charts satisfy the perfect synchrony hypothesis. The well-known causality conflicts that arise under instantaneous feedback from negative trigger conditions are resolved semantically through oracle signals. We have implemented a prototypical tool that translates μ-Charts specifications into μ-calculus formulae. These formulae are checked against temporal specifications using a μ-calculus verifier.

SN97 Peter Scholz, Dieter Nazareth. Communication Concepts for Statecharts: A Semantic Foundation. In: Transformation-Based Reactive Systems Development ARTS'97, Las Palmas de Mallorca, Springer LNCS 1231, May 1997

Statecharts have proved their usefulness for development of real-time systems; one imagines that a precise semantics of communication between components is required for any successful application to embedded systems. This contribution shows how to do this. First, sequential automata and a parallel composition operator are introduced. Then, we introduce a scoping construct for 'feedback' (signal sharing between explicitly-named components of a parallel composition). The syntactic closure of these operations is called the class of 'Mini-Statecharts'. We then discuss three different semantics that may be given to the feedback operator. They all have advantages and disadvantages, mainly mathematical, which are forthrightly discussed in depth and with clarity, using examples where necessary. The discussion is conducted within the framework of stream processing functions.

PS97d Jan Philipps, Peter Scholz. Synthesis of Digital Circuits from Hierarchical State Machines. In: 5th GI/ITG/GMM Workshop, Linz, 9.-11.4.97. Universitätsverlag Trauner, April 1997

μ-charts are a synchronous specification language for reactive systems. In contrast to other Statecharts-like formalisms, μ-charts have a formal, compositional semantics. This semantics is well-suited for operational descriptions needed for automatic verification. As a further result of our work on symbolic verification of μ-charts, we show how a μ-chart can be implemented in hardware, using a register and a combinational logic block that represents the transition relation of the system.

3.8.3 Attended Conferences

- **9th International Conference on Parallel and Distributed Computing Systems, Dijon, France, 25-27 September 1996**
  Peter Scholz presented [SNR96].

- **SOFSEM'96, Milovy, Czech Republic, 23-30 November 1996**
  Peter Scholz presented [Sch96].
- **Advances in Parallel and Distributed Computing APDC'97, Shanghai, IEEE Computer Society Press, 1997**
  Peter Scholz presented [SH97].
- **Hardware Description Languages and their Application CHDL'97, Toledo, Chapman and Hall, April 1997**
  Peter Scholz presented [PS97a].
- **Theory and Practice of Software Development TAPSOFT'97, Lille, Springer LNCS 1214, April 1997**
  Peter Scholz presented [PS97b].
- **Tools and Algorithms for the Construction and Analysis of Systems TACAS'97, Twente, Springer LNCS 1217, April 1997**
  Jan Philipps presented [PS97c].
- **Transformation-Based Reactive Systems Development ARTS'97, Las Palmas de Mallorca, Springer LNCS 1231, May 1997**
  Peter Scholz presented [SN97].
- **5th GI/ITG/GMM Workshop, Linz, 9.-11.4.97.**
  Peter Scholz presented [PS97d].

3.8.4 Visitors

Prof. Ştefănescu from Romania visited both the University of Augsburg and the TUM. The scientific output of his stay is reported in the section of Augsburg.

3.9 Swansea

3.9.1 Results

The Swansea results are presented in the four chapters of the NADA book to which Swansea members of NADA will be contributing. Three of the research areas underlying these chapters are detailed below; the fourth one is dealt with in the Uppsala section.

Algebraic Models of Microprocessors

Researchers: A C J Fox, N A Harman and J V Tucker.

Substantial progress has been made on modeling advanced implementations of microprocessors, especially *pipelined* and *superscalar* processors. A substantial example has been developed in detail, and is the subject of [FoxHar98]. Attention has also been given to the problem of verifying the correctness of microprocessor implementations.

An Algebraic Framework for Linking Program and Machine Semantics

Researchers: K Stephenson and J V Tucker.

The semantics of a number of real-world programming languages have been defined operationally in terms of the behaviour of *abstract* or *virtual machines*. Such machines enable semantics to be described in a manner which is independent of any particular architecture that the language is actually implemented and installed on. As such, virtual machines provide an ideal intermediate level in our abstraction hierarchy.

We produce algebraic specifications to model the semantics of *Java bytecode*. This low-level language is an intermediate step between high-level Java programs and architecture-specific machine-level languages. The work is to appear as [Steph98].

Currently work is being done at both higher and lower levels of the Java Virtual Machine; the semantics of Java itself is being algebraically specified, and other lower levels involved in the execution process, in the form of operating systems and hardware implementations of the Java Virtual Machine are being investigated.

SCAs and Dynamical Systems

Researchers: A V Holden, M J Poole and J V Tucker.

Concepts of hierarchies of spatially extended systems have been developed within the algebraic framework of synchronous concurrent algorithms (SCAs). These concepts are built upon notions of abstraction and approximation between the structure and behaviour of two SCAs, which are in turn built upon timing abstractions, spatial abstractions and state abstractions. This work is summarised in [HolPooTuc98].

The general theory has been applied to the modeling of spatially extended biological systems. Specifically, we have considered a number of models of electrical behaviour of the whole-heart. These models are mathematically diverse (cellular automata, coupled map lattices, coupled ordinary differential equations and partial differential equations) and reconstruct different levels of cardiac behaviour (single cell biophysics, local and global electrical wave propagation) at different space- and time-scales. The models have been formally related using general notions of SCA abstraction and approximation. The models and relationships have been used in the construction of novel hierarchical and hybrid SCA models of cardiac activity where different component models are coupled; hence local biophysical activity influences, and is influenced by, global heart behaviour. The model has been exploited in experiments reproducing well-known cardiac behaviours and pathologies. This work will appear in a paper: A V Holden, M J Poole and J V Tucker, *Hierarchical Reconstructions of Cardiac Tissue*, to be submitted to the Journal of Theoretical Biology.

FoxHar98 A C J Fox and N A Harman: Algebraic models of superscalar microprocessor implementations: A case study. In B Möller and J V Tucker (eds.): Prospects for Hardware Foundations. Springer LNCS (in preparation)

HolPooTuc98 A V Holden, M J Poole, and J V Tucker: Hierarchies of spatially extended systems. In B Möller and J V Tucker (eds.): Prospects for Hardware Foundations. Springer LNCS (in preparation)

Steph98 K Stephenson: Algebraic specification of the Java virtual machine. In B Möller and J V Tucker (eds.): Prospects for Hardware Foundations. Springer LNCS (in preparation)

3.9.2 Attended Conferences

FMCAD 96 A.C.J. Fox and N.A. Harman attended *Formal Methods in Computer Aided Design*, Palo Alto, in November 1996, and presented a paper [2].

NADA/CHDL 97 A.C.J. Fox and N.A. Harman attended the Toledo 97 meeting: in particular, the NADA annual meeting and *Computer Hardware Description Languages 1997*. N.A. Harman gave a talk to NADA on the work he is doing with A.C.J. Fox on modeling superscalar microprocessors. He also gave a tutorial to CHDL 97.

3.10 Uppsala

3.10.1 Results

The work in Uppsala related to NADA has primarily been concerned with providing foundations for a theory of streams and stream transformers where streams may have not only discrete time and data but also continuous time (modelled by a subspace of $\mathbb{R}$) and data. The properties considered are continuity and computability. We use Scott-Ershov domains to model streams and stream transformers. Here we utilise the fact that domains provide topologically and computably good representations of structures such as $\mathbb{R}$, as established by Stoltenberg-Hansen, Tucker, Edalat, Weihrauch, Blanck and others, and the facility to build function spaces between domains within the category of domains.

The above work has been carried out in co-operation with Swansea. The results will be described in a chapter of the NADA book by Blanck, Stoltenberg-Hansen and Tucker.

Our work has also included the use of certain constructive non-standard models which we believe may be useful when modelling streams and stream transformers for continuous time. We have given a model of Martin-Löf type theory in [Normann, Palmgren, Stoltenberg-Hansen 97] using this method which coincides extensionally, but not computationally, with the domain model [Waagbö 97].

3.10.2 Relevant Publications

Blanck 97a J Blanck, Domain representability of metric spaces, Annals of Pure and Applied Logic 83 (1997), 225-247

Blanck 97b J Blanck, Computability on topological spaces by effective domain representations, Ph D Thesis, Uppsala Dissertations in Mathematics 7, 1997

Blanck 97c J Blanck, Domain representations of topological spaces. U.U.D.M. Report 1997:26, Uppsala University (submitted for publication)

Blanck, Stoltenberg-Hansen, Tucker 97 J Blanck, V Stoltenberg-Hansen and J V Tucker, Computable stream transformers and domain theory (to appear in the NADA book)

Normann, Palmgren, Stoltenberg-Hansen 97 D Normann, E Palmgren and V Stoltenberg-Hansen, Hyperfinite type structures, U.U.D.M. Report 1997:10, Uppsala University (submitted for publication)

Sigstam, Stoltenberg-Hansen 97 I Sigstam and V Stoltenberg-Hansen, Representability of locally compact regular spaces by domains and formal spaces, Theoretical Computer Science 179 (1997), 319-331

Stoltenberg-Hansen, Tucker 96 V Stoltenberg-Hansen and J V Tucker, Computable rings and fields, U.U.D.M. Report 1996:29, Uppsala University. Also in: E Griffor (ed.): Handbook of Recursion Theory. North-Holland (to appear)

Waagbö 97 G Waagbö, Domains-with totality semantics for intuitionistic type theory, Ph D Thesis, University of Oslo, 1997

3.10.3 Examination

Jens Blanck obtained an F.D. (Ph. D.) in Mathematical Logic on 22 May 1997 from Uppsala University.

3.11 Attended Conferences

- **Computation and Approximation II meeting in Darmstadt, 25–28 September 1996**
  V Stoltenberg-Hansen was an invited speaker.
- **Dagstuhl-Seminar 9717 on Computability and Complexity in Analysis, Schloss Dagstuhl, 21–25 April, 1997**
  V Stoltenberg-Hansen was a keynote speaker and J Blanck participated.
- **CCA '96: Second Workshop on Computability and Complexity in Analysis, Trier, 22–23 August, 1996**
  J Blanck participated.
- **Computation and Approximation III meeting in Birmingham, 11–13 September 1997**
  J Blanck presented his paper *Domain representations of topological spaces.*

Part II
Final Report

4 Overall Project Achievements

4.1 Amsterdam

Our participation in NADA triggered a lot of design and specification activity, ranging from refinement of the concept 'early input' (a primitive to model value-passing among concurrent processes and used to model stream processing) to design and application of Grid Protocols. We reported earlier on actual achievements, relevant publications and so on. Here we sketch the background of our motivation, and discuss some overall achievements. Our participation in NADA was driven by two general goals:

1. To find simple and adequate primitives, and elegant algebraic methods to model data transfer between concurrent processes, and
2. To give a process algebraic account of applications that are relevant to NADA, viz. the modeling of hardware phenomena, and development of concerning analysis techniques.

As an example we mention our work on Synchronous Concurrent Algorithms as developed by Tucker *et al.* Most certainly, the goals described above have not been fully achieved, and research will be continued. The following achievements of our participation in NADA can be distinguished:

- A better understanding of scientific work done elsewhere. This applies in particular to the modeling of hardware and other concurrent phenomena in which parallel input/output is a major ingredient.
- Much more focus on a systematic treatment of data manipulation in process algebra, as exemplified by 'early input'. Traditionally, process algebraic modeling often concentrated on operational aspects of concurrency, and treatment of data manipulation (by processes) was ad hoc or left implicit. To mention some more examples of our activities in this field: primitives for file-transfer have been distinguished, and further research to an algebraic treatment of (concurrent) assignment is being undertaken. Furthermore, we are following closely the development of μCRL ("process algebra with data") at CWI in Amsterdam.
- A concurrent and relevant development in our work concerns research to process algebra with recursive operations, such as binary Kleene star and nesting. This topic seems to attract the attention (since 1994, at least 14 papers on this topic were published). We apply recursive operations in combination with early input and value-passing.

4.2 Augsburg

The focus of the work at Augsburg has been on the method of *Deductive Design*, i.e., on the systematic construction of a system implementation, starting from its behavioural specification, according to formal, provably correct rules. The main advantages of this approach are:

- The resulting implementation is correct by construction.
- The rules can be formulated schematically, independent of the particular application area, and hence can be re-used for wide classes of similar problems.
- Since the rules are formal, the design process can be assisted by machine.

- Implementations can be constructed in a modular way, with emphasis on correctness first and subsequent transformation to increase performance.
- The formal derivation also serves as a record of the design decisions taken and hence is an explanatory documentation. Upon modification of the system specification it eases revision of the implementation.

This paradigm has successfully been applied to sequential and, to a lesser extent, also to parallel programs. Since hardware consists of "frozen" programs, it is an obvious idea to apply this method to hardware design. A large amount of work in this area has been done by M. Sheeran and others.

The major novel ingredients and achievements in the Augsburg work are the following:

- specification at the level of predicate logic, not necessarily algorithmic yet,
- a clearer disentangling of the abstract idea of an algorithm from the concrete layout that realizes it,
- in particular, introduction of wiring operators in a late stage of the derivations, thus avoiding a lot of burden and clutter,
- a simpler approach to retiming that avoids the concept of anti-delays,
- in the asynchronous case a strongly algebraic approach to streams, notably to questions about fairness.

While the first two items had been established already in the CIP group to which NADA members B. Möller and W. Dosch belonged, the latter three items were elaborated within the NADA project.

The case studies include many of the IFIP WG10.5 Benchmark Verification Problems. Next to dealing with basic combinational and sequential circuits, a very simple treatment of systolic circuits was achieved. Finally, concerning higher-level hardware concepts, an easy formal account of pipelining became possible.

Special emphasis was laid on parametrization and re-usability aspects.

In the synchronous case, a major breakthrough was the switch to formalizing the specifications and derivations using the functional programming language *Gofer*, a subset of *Haskell*. The polymorphism of this language allows the use of Ştefănescu's network algebra and other algebraic laws both at the level of combinational and sequential circuits. Also fixpoint induction and related proof principles can be applied directly. Moreover, many derivations can be performed in a polymorphic way abstracting from concrete applications and hence achieving much better re-usability. This work is reported in the Chapter [MolHard98] of the NADA book.

In the asynchronous case, the algebraic basis has been consolidated. This thread of work concerns more algebraic ways of specifying, reasoning about and transforming descriptions of (sets of) streams. Particular emphasis is laid on the use of regular algebra in describing stream patterns, notably in connection with questions of fairness. This research is described in the Chapter [MolStr98] of the NADA book. This algebraic approach has also been tied in with other specification formalisms such as temporal and modal logic [MolTemp97, MolMod97, MolStef98].

4.3 Kent

The work undertaken at Kent has focussed on two themes: the use of dependent types in specifications for digital systems and in extending formal verification from the digital domain towards the analog domain.

The motivation for using dependent types (loosely, types that are parameterised by values and/or other types) for formal specification is that they offer both greater expressiveness and greater generality. During the course of the NADA project, we have:

- Carried out an in-depth investigation of the use of dependent types in specifying and in formally verifying complex systolic array architectures;
- Shown how it is possible to capture, within a formal theory, the interrelation between the structural and the behavioural aspects of a circuit. Circuits are treated as typed graphs and their behavioural specifications as predicates. The key difficulty that has to be overcome is that, in a structural description, the types of ports have to be treated as values whereas, in the behavioural description, they appear as types. Using dependent types, it was found possible to devise a sufficiently expressive type for interfaces that guaranteed the consistency of structural and behavioural specifications. The approach was demonstrated using Veritas, an implementation of a dependently-typed higher-order logic.

The motivation for extending formal specification and verification techniques from the digital domain towards the analog domain is that, in practice, significant sections of a digital system often have to be designed at the analog electronics level of abstraction. By doing this, logically redundant signal conditioning and buffering stages can be eliminated, leading to circuitry which can be up to an order of magnitude faster, and with lower power consumption, than the corresponding circuitry designed (in terms of gates and flipflops, etc) at the digital level of abstraction. During the course of the NADA project, we have:

- Demonstrated how, using higher-order logic, the behavioural characteristics of typical electronic components (transistors, diodes, resistors) can be specified in terms of analog voltages and currents.
- Shown how the behavioural specification of an electronic circuit (for example, one that uses pass-transistor logic) can be inferred from the analog specifications of the component parts and their interconnections and how specifications at the analog and digital levels are related.
- Devised and implemented a decision procedure that, given a description of an analog circuit and behavioural specifications of its component parts, determines whether it correctly implements a given behavioural specification.

We believe that this work is likely to impact strongly upon the design of future generations of mixed-level hardware design languages which, at present, lack rigorous foundations.

4.4 Leeds

The focus of the work at Leeds has been on the development and evaluation of case studies, based on biomedical applications, that are practical illustrations of what current problems exist in the computational modeling of complicated, highly structured, spatially extended systems that are represented mathematically by different classes of models - cellular automata, coupled

map lattices, coupled ordinary differential equation lattices, and partial differential equations, and that interact with data streams. The main idea of the approach is to use the theory of synchronous concurrent algorithms as a framework within which different types of model are embedded, and can be coupled. The advantage of this modular approach is that newer generation models can readily be incorporated, and the framework is applicable to modeling a wide range of hierarchical, stuctured system, not just the case studies, based on cardiac muscle, arryhthmias, and their control.

The principal results of the Leeds work have been:

- The development of a family of partial differential equation models for different normal and patholgical cardiac tissues, and the use of these models to simulate cardiac arryhthmias.
- The design of a novel means of controlling re-entrant cardiac arryhthmias, and the evaluation of its practicability.
- The development of a geometric, anisotropic model for the ventricles of the heart, and the coupling of this geometry with simple (coupled map lattice, simple ordinary differential equation models) of excitability and with partial differential equation models.
- The development of a formal approach, within the theory of synchronous concurrent algorithms, for coupling and verifying the above models.
- The application of the same approach to other biological systems - neural nets, and oceanic plankton population dynamics.

This approach has been primarily in collaboration with the Swansea group, and has led to extensive funding from UK research council and medical charities, in the applications of these models to clinical problems. It has also led to UK funding to continue the collaboration with the Swansea group.

4.5 Lübeck

The work in Lübeck concentrated on deductive design — both in the area of hardware and software [DosMol96]. This methodology aims at systematically deriving a system implementation from its behavioural specification following sound transformation rules. The derivation documents the design decisions and guides the redesign upon changing requirements. Within a structured design methodology, we initially abstract from layout and timing issues and concentrate on algorithmic design principles. Synchronous descriptions are based on the algebra of finite sequences modelling the linear succession in space or in time; asynchronous descriptions employ stream processing functions.

The major contributions in the work of Lübeck consist in

- isolating and formalizing important design steps [Dos96a],
- abstracting characteristic design patterns from specific case studies [Dos98],
- deriving standard implementations for iterative and tree structured networks using higher-order functions [Dos98],
- reasoning about ordered streams by set abstraction [Dos96b].

For a class of digit recurrence algorithms, we derived different implementations as combinational and sequential circuits in a schematic way. This class of iterative circuits was then generalized to tree structured networks originating from cascade recursive functions. The derivations are parameterized supporting their reuse in different applications.

Dos96a W. Dosch: *Calculating Digital Counters.* In: D. Bjørner, M. Broy, I.V. Pottosin (eds.): Perspectives of Systems Informatics. Second International Andrei Ershov Memorial Conference. Akademgorodok, June 25–28, 1996. Lecture Notes in Computer Science *1181*. Berlin: Springer 1996, 21–39

Dos96b W. Dosch: *Calculating Hamming Streams.* In: K. Yetongnon, S. Hariri (eds.): Proceedings of the Ninth International Conference on Parallel and Distributed Computing Systems, Vol. I. Dijon, September 25–27, 1996. Raleigh, N.J.: International Society for Computers and Their Applications 1996, 529–534

DosMol96 W. Dosch, B. Möller: *Calculating a Functional Module for Binary Search Trees.* In: W. Kluge (ed.): Implementation of Functional Languages. 8th International Workshop IFL'96, Bad Godesberg, September 16-18, 1996. Lecture Notes in Computer Science *1268*. Berlin: Springer 1997, 267–284

Dos98 W. Dosch: *Deductive Design of Iterative Circuits.* In: B. Möller, J.V. Tucker (eds.): Prospects For Hardware Foundations. Lecture Notes in Computer Science (to appear)

4.6 Madrid

Over the past three years our work has been concentrated in three main areas. The first area is the semantics of hardware description languages, chiefly in connection with the formalization of VHDL and the development of specification, refinement and verification calculi for that language and real-time systems in general.

Secondly we have been working with the specification language LOTOS. We have produced methods for designing hardware systems and generating VHDL code automatically from the design. Thirdly we have been researching in the area of codesign also using LOTOS as the system description language.

We have given several presentations to the project about the last two areas and we have been actively developing the first are within the project itself. The project has been specially useful to us in the further development of our logic for real-time system design from our approach to VHDL semantics. With regard to the latter, we have debugged the refinement calculus of VHDL, finally proving it to be complete with respect to our formal semantics for the language. We have also taken the opportunity to investigate the problems of design in non-discrete regimes using our formalisms and those of others. The operational amplifier has been a particularly instructive example of an analog device operating in continuous time. From our attempts to describe it we have concluded that compositionality under these regimes derives from the constraints on a system and not from the solutions to (subsets of) those constraints. If we try to do so, then we run into problems with causality requirements. As a result of these investigations we have been able to suggest a semantic model for the (now accepted by the IEEE) analog extension to VHDL, developing a prototype implementation.

We feel we have benefitted greatly from involvement with a group with such expertise in formal logic and we have attempted, with some success, to apply the techniques of formal logic to our treatments of hardware specification languages.

4.7 LMU Munich

We have worked on a variety of subjects all subsumable under the heading 'applied proof theory'. This was in conjunction with the development of an interactive prover Minlog well suited for the purpose and in particular for hardware verification and development. Most notably Ulrich Berger and Matthias Eberl have been working on NADA related questions. Details of their work have been reported earlier and will not be repeated here. However, the following paper has not yet been mentioned.

BerSchwi96 Ulrich Berger and Helmut Schwichtenberg: The greatest common divisor: a case study for program extraction from classical proofs. In: S. Berardi and M. Coppo (eds.): Types for Proofs and Programs. International Workshop TYPES '95, Torino, Italy, June 1995. Selected Papers. Lecture Notes in Computer Science **1158**. Berlin: Springer 1996, 36–46

Yiannis Moschovakis suggested the following example of a classical existence proof with a quantifier-free kernel which does not obviously contain an algorithm: the gcd of two natural numbers a_1 and a_2 is a linear combination of the two. Here we treat that example as a case study for program extraction from classical proofs. We apply H. Friedman's A-translation followed by a modified realizability interpretation to extract a program from this proof. However, to obtain a reasonable program it is essential to use a refinement of the A-translation introduced in Berger/Schwichtenberg 1995. This refinement makes it possible that not all atoms in the proof are A-translated, but only those with a "critical" relation symbol. In our example only the divisibility relation $\cdot|\cdot$ turns out to be critical.

4.8 TU Munich

In the past years, the design methodology Focus for distributed systems has been developed at the TUM. As the original intention of Focus was to describe software systems, the NADA project has been a unique opportunity to verify to which extent Focus is applicable for the specification of hardware, too. To achieve this, a number of case studies, like the formal development of a production cell, a distributed min/max component, interfaces, modulo-n counter, and the Alpha AXP^{TM} shared memory have been carried out using Focus.

Apart from these case studies, which have shown that Focus is indeed suitable for the description of hardware oriented systems, it also turned out that Focus provides a mathematical framework to define a formal semantics for hardware description languages like VHDL and system level languages like Statecharts. Furthermore, it was proven that Focus is not only appropriate to describe pure software or hardware components but in addition can be used to specify mixed hardware/software systems, i.e. can serve as a formal foundation for Hardware/Software Codesign.

4.9 Stockholm

4.9.1 Research Activity

K. Meinke has concentrated on the theory and applications of higher-order algebra within formal hardware specification and verification. This research has involved both theoretical and practical work. Much of it was carried out in collaboration with PhD students at Swansea, namely Dr Barry Hearn (now at Formal Methods Group, Logica UK Ltd, Cambridge) and Dr. Jason Steggles (now at the Dept. of Computer Science, University of Newcastle). Both students received their PhDs for research related to NADA, although they were funded by the national science council in Britain (EPSRC).

K. Meinke also collaborated with B. Möller (Augsburg) and J. Heering (CWI Amsterdam) in organising a series of international workshops on higher-order algebra, logic and term rewriting, under the auspices of the NADA project. The proceedings of these workshops are published as Heering et al [1994], Dowek et al [1996] and Hanus et al [1997].

Below we describe research carried out under three headings. Finally we list publication output.

4.9.2 Foundations of Hardware Specification

K. Meinke has investigated the foundations of hardware specification using higher-order algebra. Roughly this corresponds with the equational fragment of many-sorted higher-order logic, where higher types are function types. Thus it is related to Church's system of finite types. This framework is natural for hardware description since it allows the direct representation of hardware devices as stream transformers. This observation has also been confirmed elsewhere, e.g. the HOL community.

Significant achievements in this area have been:

(a) An exact characterisation of the specification power of higher-order equations in Meinke [1997]. Earlier work on this problem was published in Kosiuczenko and Meinke [1995]. This power was shown to be Π^1_1, thus it properly includes the arithmetical hierarchy. Furthermore, the expressive power of the hierarchy of specifications (second-order, third-order, etc.) collapses to second-order. In this sense, (quantifier free) second-order equational logic is slightly more powerful than first-order arithmetic.

(b) A proof theory for higher-order equations which exactly characterises the case where higher-order equational reasoning is reducible to first-order equational reasoning through the existence of normal form proofs for higher-order equations. This result makes use of a particularly interesting topology on higher-order algebras which seems to capture a definition of observational equivalence for higher-order operations (e.g. stream transformers). Case study research with L.J. Steggles (see Meinke and Steggles [1994]) on Kung's systolic convolution algorithm found this to be a useful result for theorem proving and hardware verification.

4.9.3 Practical Case Studies

Together with L.J. Steggles, K. Meinke studied some well known hardware algorithms in order to gain insight into their specification and verification requirements. These included a typical systolic algorithm (convolution) and a typical dataflow algorithm (Hamming Stream). The convolution algorithm lead to fundamental insights in the proof theory of verification for systolic algorithms (see 1.(b) above). The dataflow algorithm revealed the power of higher-order algebra for semantical verification.

L.J. Steggles extended the theory of higher-order algebraic specification to transfinite types, and showed that this formed a suitable framework to capture the parametricity of families of hardware algorithms. (See Steggles [1996])

L.J. Steggles also developed a theory of parameterised higher-order algebraic specifications, which allowed to take advantages of the polymorphism inherent in the convolution algorithm to simplify specification (see Steggles [1997].)

K. Meinke investigated the impact of the proof theory of higher-order equations on higher-order equational theorem proving, and verification of hardware by means of term rewriting in Meinke [1996].

4.9.4 Software Tools

Together with B.M. Hearn, K. Meinke designed and implemented a tool for parsing and executing higher-order algebraic specifications by rewriting of terms and types. This tool was based on earlier published research of K. Meinke. The tool and its input language is called ATLAS (A Typed Language for Algebraic Specification) and is based on many-level algebraic specifications, which can be used for equational specification both of data and types (by means of type equations). ATLAS is implemented in C under UNIX and has a Motif graphical user interface. ATLAS was used to specify and verify hardware algorithms, as well as other case studies, including order-sorted, polymorphic and recursive types. Research on ATLAS is published in Hearn and Meinke [1994].

The research on ATLAS was also taken further by E. Visser at the CWI Amsterdam (outside the Nada project).

4.9.5 Publications

G. Dowek, J. Heering, K. Meinke, B. Möller (eds), Higher-Order Algebra, Logic and Term Rewriting, (HOA '95), Lecture Notes in Computer Science, 1074, Springer Verlag, Berlin, 1996.

M. Hanus, J. Heering, K. Meinke (eds), Algebraic and Logic Programming (Joint ALP '97 and HOA '97), Lecture Notes in Computer Science, 1298, Springer Verlag, Berlin, 1997.

B.M. Hearn and K. Meinke, ATLAS: a typed language for algebraic specification, 146-168 in J. Heering et al (eds), Higher-Order Algebra, Logic and Term Rewriting (HOA '93), Lecture Notes in Computer Science, 816, Springer Verlag, Berlin, 1994.

J. Heering, K. Meinke, B. Möller, T. Nipkow (eds), Higher-Order Algebra, Logic and Term Rewriting (HOA '93), Lecture Notes in Computer Science, 816, Springer Verlag, Berlin, 1994.

K. Meinke and L.J. Steggles, Specification and verification in higher-order algebra: a case study of convolution, 189-222 in J. Heering et al (eds), Higher-Order Algebra, Logic and Term Rewriting (HOA '93), Lecture Notes in Computer Science, 816, Springer Verlag, Berlin, 1994.

K. Meinke and P. Kosiuczenko, On the power of higher-order algebraic specifications, Information and Computation 124, 85-101, 1995.

K. Meinke, Higher-order equational logic for specification, simulation and testing, 124-143 in:G. Dowek et al (eds), Higher-Order Algebra, Logic and Term Rewriting, (HOA '95), Lecture Notes in Computer Science, 1074, Springer Verlag, Berlin, 1996.

K. Meinke, A completeness theorem for the expressive power of higher-order algebraic specifications, Journal of Computer and Systems Sciences 54, 502-518, 1997.

K. Meinke, Proof theory of higher-order equations, conservativity, normal forms and term rewriting, Information and Computation, to appear 1997.

L.J. Steggles, Higher-order algebra with transfinite types, 238-263 in G. Dowek et al (eds), Higher-Order Algebra, Logic and Term Rewriting, (HOA '95), Lecture Notes in Computer Science, 1074, Springer Verlag, Berlin, 1996.

L.J. Steggles, Parameterised higher-order algebraic specifications, 76-98 in M. Hanus et al (eds), Algebraic and Logic Programming (Joint ALP '97 and HOA '97), Lecture Notes in Computer Science, 1298, Springer Verlag, Berlin, 1997.

4.10 Swansea

4.10.1 Algebraic Models of Microprocessors

Work on microprocessor modeling has progressed from basic representations of simple processors, at the level of the programmer, through models of microprogrammed implementations [4, 5], to models of advanced processor implementations [2, 3]. In addition, considerable attention has been given to the problems of verifying the correctness of microprocessor implementations.

At the start of the NADA project,*iterated map* microprocessor models were restricted to a simple PDP-8-derived example, at the level of the programmer. An implementation of the PDP-8 was developed, and in parallel a technique was developed for substantially simplifying the process of formal verification. This required that a number of conditions be met, that were trivially satisfied by iterated map models of microprocessors, and their associated timing abstraction functions. The PDP-8 appeared in [4]. A more sophisticated example, with input and output, was then developed. This was based on a standard example, *Gordon's Computer* [5].

The substantial body of work on microprocessor representation has been concerned with pipelined, and especially superscalar processors. The problems that these bring are (a) substantially increased complexity of implementations; and (b) a timing model that differs substantially from that of the programmer's level model, or architecture. Techniques were developed for representing superscalar microprocessors, and existing correctness conditions were modified. Preliminary results appeared in [2]. A substantial example superscalar microprocessor has been developed, and will appear in [3]. Work continues on developing techniques for simplifying the verification of superscalar processors. A full account of the work on superscalar processors will appear in [1].

N.A. Harman is writing a monograph on the algebraic representation of microprocessors as iterated maps; a proposal for publication is expected to be submitted in December 1997.

4.10.2 An Algebraic Framework for Linking Program and Machine Semantics

During the period 1994–7, KS and J.V. Tucker constructed an algebraic framework in which the process of executing a high-level program can be described at the level of hardware. There are many layers of abstraction involved between these two extremes. Each layer of the hierarchy consists of an algebraic specification for the semantics of the language at that level. To produce specifications for each layer requires the syntax and a semantic mapping of the language to be defined. In addition, to compare levels of the hierarchy and to define the relative correctness of levels to each other, these layers are structured further.

General methods of defining syntax are defined in [14] based on the notion of *filtering* context-free supersets to produce algebraic specifications of non-context-free languages. This work is set against a background of research into the relationship between context-free grammars and closed term algebras.

An operational model of semantics is defined in [11] and extended in [13], in which the sequences of states produced by the execution of programs are explicitly clocked by time. This enables computable models of operational semantics to be developed and hence algebraically specified. In particular, this provides an algebraic approach to structural operational semantics of high-level languages. The approach is also applicable to the semantics of low-level languages; this work is set against a background of research into machine semantics developed by J.V. Tucker and N.A. Harman within NADA.

In addition, each layer is structured so that the specifications have a common signature.

This allows the process of compilation between the layers to be expressed as a homomorphism. Correctness of one level of the hierarchy with respect to another can then be stated in terms of commutative diagrams. This work is set against a background of research in algebraic semantics.

A result of [4] is extended in [11] (to deal with concepts at a higher level of abstraction) to show that the act of proving correctness can be reduced to establishing the correctness of the commutative diagram before execution (trivially) and at the end of the first time cycle. As these stages do not require structural induction to be used, this both dramatically reduces the work required, and also makes the process feasible for mechanical checking by a suitable theorem prover.

In [11] the first two levels of this hierarchy are specified and are (manually) shown to be correct relative to each other for a non-trivial case study of a simple **while** programming language and a very abstract machine-level language. This work is summarised in [12]. This framework is now being extended to deal at lower levels of the hierarchy, with the aim of linking our work to that of N.A. Harman regarding the correctness of microprocessors.

4.10.3 SCAs and Dynamical Systems

Work by J.V. Tucker and M.J. Poole , in collaboration with A.V. Holden (University of Leeds) has focussed on algebraic models of spatially extended computational systems. The theory of synchronous concurrent algorithms (SCAs) has been applied and extended to unify the modeling of a range of different types of hardware system and biological system.

In our biological system applications, we have been primarily interested in coupled map lattice, cellular automaton, neural network, coupled ordinary differential equation and partial differential equation models of spatially extended biological phenomena. We have concentrated on models of electrical phenomena in cardiac tissue. On making discrete approximations of continuous spaces and times, the diverse range of available models of cardiac excitation are all examples of SCAs and may therefore be studied in a unified way, from the point of view of parallel deterministic computation. The notion of discrete space and local connectivity in cardiac models corresponds closely with the notion of parallel digital systems; continuous state systems are closely related to analogue computing systems.

Notions of hierarchies of spatially extended systems have been developed within the framework of SCAs. Our formal concepts of hierarchy are derived from notions of abstraction and approximation between the structure and behaviour of two SCAs, which in turn are built upon abstractions of time, space and state. The general theory has been exploited in the investigation of systolic algorithms and, especially, of spatially extended biological systems. Our concepts of hierarchy have led to the rigorous comparison of the behaviours of mathematically and biologically diverse models of cardiac electrical behaviour that are unified by SCA theory. Further, they have led to the construction of hierarchical and hybrid SCA models of cardiac activity combining many component models and reconstructing cardiac behaviour at many different, interacting, levels of biological detail.

In 1995 the development of a monograph on general SCA theory and applications was begun by M.J. Poole and J.V. Tucker; a proposal for publication is expected to be submitted in early 1998. This text will include many of the results from our collaboration with A.V. Holden as part of the NADA Working Group.

Current publications are: [10, 6, 8, 7].

References

[1] A C J Fox. *Algebraic Representation of Advanced Microprocessors.* PhD thesis, Department of Computer Science, University of Wales Swansea, 1997.

[2] A C J Fox and N A Harman. An algebraic model of correctness for superscalar microprocessors. In *Formal Methods in Computer-Aided Design*, pages 346 – 361. Lecture Notes in Computer Science 1166, Springer-Verlag, 1996.

[3] A C J Fox and N A Harman. Algebraic models of superscalar microprocessor implementations: A case study. In B Möller and J V Tucker, editors, *Prospects for Hardware Foundations.* Springer-Verlag, to appear, 1998.

[4] N A Harman and J V Tucker. Algebraic models of microprocessors: Architecture and organisation. *Acta Informatica*, 33:421 – 456, 1996.

[5] N A Harman and J V Tucker. Algebraic models of microprocessors: the verification of a simple computer. In V Stavridou, editor, *Proceedings of the 1995 IMA Conference on Mathematics for Dependable Systems.* Oxford University Press, 1997.

[6] A V Holden, M J Poole, and J V Tucker. Reconstructing the heart. *Chaos, Solitons and Fractals*, 5:691–706, 1994.

[7] A V Holden, M J Poole, and J V Tucker. An algorithmic model of the mammalian heart: propagation, vulnerability, re-entry and fibrillation. *International Journal of Bifurcation and Chaos*, 6:1623–1635, 1996.

[8] A V Holden, M J Poole, and J V Tucker. Theoretical framework for analysis and synthesis of networks of neurones. In R Moreno-Diaz and J Mira-Mira, editors, *Brain Processes, Theories and Models*, pages 320–329. MIT Press, 1996.

[9] A V Holden, M J Poole, and J V Tucker. Hierarchies of spatially extended systems. In B Möller and J V Tucker, editors, *Prospects for Hardware Foundations.* Springer-Verlag, to appear, 1998.

[10] A V Holden, M J Poole, J V Tucker, and H Zhang. Coupling cmls and the synchronisation of a multilayer neural computing system. *Chaos, Solitons and Fractals*, 4:2249–2268, 1994.

[11] K Stephenson. *An Algebraic Approach to Syntax, Semantics and Compilation.* PhD thesis, University of Wales Swansea Computer Science Department, 1996.

[12] K Stephenson. Compiler Correctness using Algebraic Operational Semantics. Technical Report CSR 1-97, Department of Computer Science, University of Wales Swansea, January 1997.

[13] K Stephenson. Algebraic specification of the java virtual machine. In B Möller and J V Tucker, editors, *Prospects for Hardware Foundations.* Springer-Verlag, to appear, 1998.

[14] K Stephenson. The reporting and correcting of syntax errors in an algebraic environment. Technical report, Department of Computer Science, University of Wales Swansea, 1998. in preparation.

4.11 Uppsala

The work of the Uppsala site has primarily been concerned with semantical foundations of data types, streams and stream transformers. The main tool used has been the theory of domains. Domain theory arose from generalisations of the theory of recursion and computability in the work of D. Scott, Yu. Ershov and others. Its use in denotational semantics is well known. We have also considered the use of formal spaces and of certain non-standard models.

Our work has been concentrated on considering domain theory as a theory of computation on data types or topological algebras, where approximations are the primary objects. That is, computations are performed on concrete approximations and the results are then transferred to ideal or non-concrete elements (the elements of the considered topological algebra) via a limit process. A domain captures this idea well. The method used is that of domain representability. It was first explicitly formulated in [Stoltenberg-Hansen, Tucker 85], and [Stoltenberg-Hansen, Tucker 88] and has been much developed since then by several people. In [Stoltenberg-Hansen, Tucker 95] the method was extended to cover locally compact regular spaces such as the reals $\mathbb{R}$. It has been further extended to all metric spaces by [Blanck 97a]. The thesis [Blanck 97b] contains a thorough study of domain representability in general and the concerns of computability in particular.

Another tool useful for the study of computability for topological structures, and for obtaining constructive versions of non constructive theorems such as the Tychonoff or Hahn-Banach theorems, is the theory of formal spaces. In [Stoltenberg-Hansen, Sigstam 97] we have made a comparison between this method and the method of domain representability showing their equivalence in certain precise senses for regular locally compact spaces. We have also found a connection between domain theory and model theory where saturation plays a key role. In [Palmgren, Stoltenberg-Hansen 95] we gave an initial account of such a connection. Then, in [Palmgren, Stoltenberg-Hansen 94], we used this to give a logical presentation of the Kleene-Kreisel continuous functionals.

Finally we have considered certain constructive non-standard models which we believe will be useful for foundational questions addressed by the NADA project. We have given a model of Martin-Löf type theory in [Normann, Palmgren, Stoltenberg-Hansen 97] using this method which coincides extensionally, but not computationally, with the domain model [Waagbö 97].

Blanck 97a J Blanck, Domain representability of metric spaces, Annals of Pure and Applied Logic 83 (1997), 225-247.

Blanck 97b J Blanck, Computability on topological spaces by effective domain representations, Ph D Thesis, Uppsala Dissertations in Mathematics 7, 1997.

Blanck 97c J Blanck, Domain representations of topological spaces. U.U.D.M. Report 1997:26, Uppsala University

Blanck, Stoltenberg-Hansen, Tucker 97 J Blanck, V Stoltenberg-Hansen and J V Tucker, Computable stream transformers and domain theory. To appear in the NADA book.

Normann, Palmgren, Stoltenberg-Hansen D Normann, E Palmgren and V Stoltenberg-Hansen, Hyperfinite type structures, U.U.D.M. Report 1997:10, Uppsala University (Submitted for publication)

Palmgren, Stoltenberg-Hansen, 94 E Palmgren and V Stoltenberg-Hansen, A logical presentation of the continuous functionals, U.U.D.M. Report 1994:48. (To appear in the Journal of Symbolic Logic.)

Palmgren, Stoltenberg-Hansen, 95 E Palmgren and V Stoltenberg-Hansen, Logically presented domains , LICS Proceedings 1995, 455-463.

Sigstam, Stoltenberg-Hansen 97 I Sigstam and V Stoltenberg-Hansen, Representability of locally compact regular spaces by domains and formal spaces, Theoretical Computer Science 179 (1997), 319-331.

Stoltenberg-Hansen, Tucker 85 V Stoltenberg-Hansen and J V Tucker, Complete local rings as domains, CTCS Report 1.85, University of Leeds, 1985.

Stoltenberg-Hansen, Tucker 88 V Stoltenberg-Hansen and J V Tucker, Complete local rings as domains, Journal of of Symbolic Logic 53(1988), 603 - 624.

Waagbö 97 G Waagbö, Domains-with totality semantics for intuitionistic type theory, Ph D Thesis, University of Oslo, 1997.

5 The NADA Book

The group has found it worthwhile to present its results in an in-depth fashion to a wider audience. The book will serve as the detailed Final Report of the NADA Working Group.

We have been able to place this book within the Tutorial Subseries of Springer Lecture Notes in Computer Science. It is envisaged that it be published in Summer 1998. The current plan for the book's structure is as follows.

PROSPECTS FOR HARDWARE FOUNDATIONS

Editors: B Möller & J V Tucker

Springer Lecture Notes in Computer Science

Preface
(on the NADA project)

Introduction
(to the area and the book -
discussion of NIL in this context)

I MATHEMATICAL FOUNDATIONS

J Blank, V Stoltenberg-Hansen and J V Tucker
Computable stream transformations and domain representations 30pp

B Möller
Ideal stream algebra 30pp

U Berger, M Eberl and H Schwichtenberg
Normalisation by evaluation 30pp

II HARDWARE AND DYNAMICAL SYSTEMS

A C J Fox and N A Harman
Algebraic models of superscalar microprocessor implementations: A case study 40pp

A V Holden, M J Poole and J V Tucker
Hierarchies of spatially extended systems 40pp

K Stephenson
Algebraic specification of the Java virtual machine 40pp

J A Bergstra and A Ponse
Grid protocol specifications 40pp

III VERIFICATION AND DEDUCTIVE DESIGN

K Hanna
Reasoning about imperfect digital systems 30pp

J Philipps and P Scholz
Formal verification and hardware design using state charts 32pp

K Stølen and M Fuchs
An exercise in conditional refinement 28pp

B Möller
Deductive hardware design: a functional approach 40pp

W Dosch
Deductive design of iterative circuits 25pp